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\C5_health care diabetes.csv")
a
```

Out[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFun
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

In [3]:

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

In [4]:

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

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFuncti
0	6	148	72	35	0	33.6	0.6
1	1	85	66	29	0	26.6	0.3
2	8	183	64	0	0	23.3	0.6
3	1	89	66	23	94	28.1	0.1
4	0	137	40	35	168	43.1	2.2
5	5	116	74	0	0	25.6	0.2
6	3	78	50	32	88	31.0	0.2
7	10	115	0	0	0	35.3	0.1
8	2	197	70	45	543	30.5	0.1
9	8	125	96	0	0	0.0	0.2

In [13]:

```
c=a.iloc[:,0:9]
d=a.iloc[:,-1]
```

In [14]:

```
c.shape
```

Out[14]:

```
(10, 9)
```

In [15]:

```
d.shape
```

Out[15]:

```
(10,)
```

In [16]:

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

In [17]:

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

In [18]:

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

Out[18]:

```
LogisticRegression()
```

In [19]:

```
e=[[2,5,77,8,6,5,4,66,88]]
```

In [5]:

```
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 [20]:

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

Out[20]:

```
array([1], dtype=int64)
```

In [21]:

```
logr.classes_
```

Out[21]:

```
array([0, 1], dtype=int64)
```

In [22]:

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

Out[22]:

```
0.0
```

In [23]:

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

Out[23]:

```
1.0
```

In [6]:

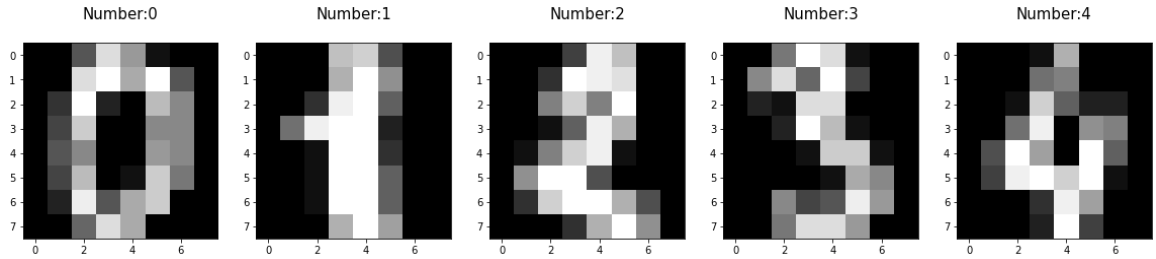
```
digits=load_digits()
digits
```

Out[6]:

```
{'data': array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
        [ 0.,  0.,  0., ..., 10.,  0.,  0.],
        [ 0.,  0.,  0., ..., 16.,  9.,  0.],
        ...,
        [ 0.,  0.,  1., ...,  6.,  0.,  0.],
        [ 0.,  0.,  2., ..., 12.,  0.,  0.],
        [ 0.,  0., 10., ..., 12.,  1.,  0.]]),
 'target': array([0, 1, 2, ..., 8, 9, 8]),
 'frame': None,
 'feature_names': ['pixel_0_0',
  'pixel_0_1',
  'pixel_0_2',
  'pixel_0_3',
  'pixel_0_4',
  'pixel_0_5',
  'pixel_0_6',
  'pixel_0_7',
  'pixel_1_0',
```

In [7]:

```
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 [8]:

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

In [9]:

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

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

In [10]:

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

Out[10]:

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

In [11]:

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

Out[11]:

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

In [12]:

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

Out[12]:

```
0.9537037037037037
```

In [ ]: