In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:

In [3]:
```
dftrain=pd.read_csv(r"C:\USERS\user\Downloads\C5_health care diabetes - C5_hea
```

Out[3]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunctio
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
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.1
764	2	122	70	27	0	36.8	0.3
765	5	121	72	23	112	26.2	0.2
766	1	126	60	0	0	30.1	0.3
767	1	93	70	31	0	30.4	0.3

768 rows × 9 columns

In [4]:

Out[4]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [6]:
```
a=dftrain[['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','
a
```

Out[6]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunctic
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
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.1
764	2	122	70	27	0	36.8	0.3
765	5	121	72	23	112	26.2	0.2
766	1	126	60	0	0	30.1	0.3
767	1	93	70	31	0	30.4	0.3

768 rows × 9 columns

In [7]:
```
b=dftrain.head(10)
```

Out[7]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction
0	6	148	72	35	0	33.6	0.627
1	1	85	66	29	0	26.6	0.351
2	8	183	64	0	0	23.3	0.672
3	1	89	66	23	94	28.1	0.167
4	0	137	40	35	168	43.1	2.288
5	5	116	74	0	0	25.6	0.201
6	3	78	50	32	88	31.0	0.248
7	10	115	0	0	0	35.3	0.134
8	2	197	70	45	543	30.5	0.158
9	8	125	96	0	0	0.0	0.232

In [8]:
```
a=b[['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','
```

Out[8]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction
0	6	148	72	35	0	33.6	0.627
1	1	85	66	29	0	26.6	0.351
2	8	183	64	0	0	23.3	0.672
3	1	89	66	23	94	28.1	0.167
4	0	137	40	35	168	43.1	2.288
5	5	116	74	0	0	25.6	0.201
6	3	78	50	32	88	31.0	0.248
7	10	115	0	0	0	35.3	0.134
8	2	197	70	45	543	30.5	0.158
9	8	125	96	0	0	0.0	0.232

In [9]:
```
c=a.iloc[:,0:9]
```

In [10]:

Out[10]:
```
(10, 9)
```

In [11]:

Out[11]:
```
(10,)
```

In [12]:

In [13]:

In [14]:
```
logr=LogisticRegression()
```

Out[14]:
```
LogisticRegression()
```

In [15]:

In [16]:
```
prediction=logr.predict(observation)
```

Out[16]:
```
array([1], dtype=int64)
```

In [17]:

Out[17]:
```
array([0, 1], dtype=int64)
```

In [18]:

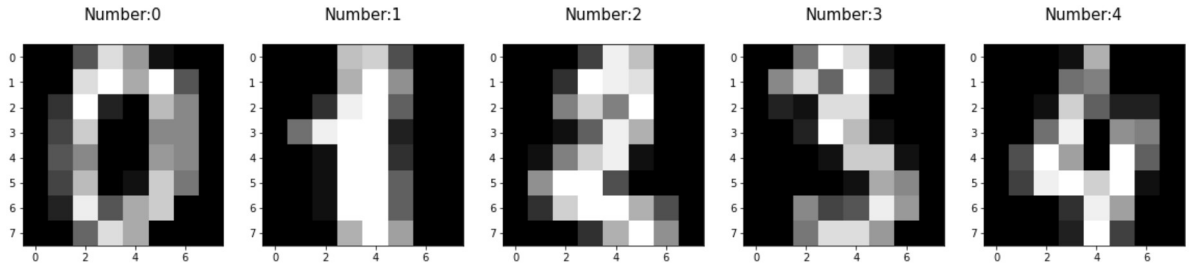
Out[18]:
```
2.220446049250313e-16
```

In [19]:
```
import re
from sklearn.datasets import load_digits
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
```

In [20]:
```
digits=load_digits()
```

Out[20]:
```
{'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',
  'pixel_1_1',
```

In [21]:
```
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)
```



In [22]:

In [23]:
```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
```

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

In [24]:
```
logre=LogisticRegression(max_iter=10000)
```

Out[24]:
```
LogisticRegression(max_iter=10000)
```

In [25]:

```
[5 9 5 7 7 9 1 2 1 0 1 1 7 0 2 0 7 5 4 0 9 5 1 9 2 2 4 4 8 6 9 6 1 2 9 3 9
 6 8 2 3 9 1 8 6 5 5 0 8 0 3 3 8 4 1 2 8 0 0 1 2 6 6 0 8 0 3 6 1 3 3 6 2 7
 4 5 3 6 8 2 7 0 6 0 5 9 1 6 6 5 5 9 6 1 1 7 2 5 9 7 0 7 1 5 1 6 1 9 6 2 3
 3 2 9 4 8 3 6 1 4 2 0 8 8 1 1 9 9 1 6 2 2 5 0 0 2 2 4 4 3 3 1 9 7 8 0 8 1
 6 1 1 4 9 0 4 1 8 3 7 7 8 5 1 5 0 0 3 1 1 4 8 2 9 6 1 8 6 6 9 8 4 8 8 1 6
 1 4 5 3 5 9 0 5 4 4 5 9 1 2 9 9 8 4 3 6 9 3 9 1 4 2 0 5 5 1 5 5 0 6 0 0 8
 5 7 8 8 4 5 3 8 5 6 3 7 2 1 7 1 9 9 7 2 8 0 0 6 0 1 7 3 2 6 8 1 9 9 1 3 3
 9 4 8 7 0 8 0 4 4 9 7 3 4 6 7 8 0 3 2 4 9 7 5 9 1 5 4 5 0 5 5 5 1 0 1 6 0
 4 7 4 3 1 7 7 8 4 3 1 4 1 8 4 3 2 2 0 1 4 6 6 2 9 5 9 5 7 3 5 5 8 0 4 0 6
 3 6 2 4 6 4 2 8 8 6 7 3 7 5 4 5 3 4 0 6 3 2 7 0 4 7 6 8 4 1 2 7 0 9 5 0 4
 7 2 1 1 5 5 0 2 1 3 6 1 6 0 5 1 7 1 1 9 6 5 2 6 6 8 7 4 5 9 0 2 4 8 9 9 5
 3 9 2 6 4 3 1 1 3 9 9 9 6 4 2 5 2 9 2 0 5 4 0 8 4 6 4 5 6 7 4 8 7 6 6 7 3
 1 2 9 8 7 1 0 5 3 9 8 3 1 9 0 8 4 0 7 8 9 0 7 2 2 5 2 5 8 6 5 1 7 7 2 3 6
 9 2 6 2 6 8 7 7 6 0 1 7 0 5 4 0 6 8 1 5 2 3 9 2 8 0 3 2 2 5 2 1 3 8 9 2 1
 6 5 6 1 1 7 5 4 4 4 8 3 4 7 6 1 3 6 6 7 5 6]
```

In [26]:

```
0.9518518518518518
```

In [ ]: