In [1]:

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

In [2]:

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

In [3]:

```python
df=pd.read_csv(r"framingham")
```

In [4]:

```python
df
```

Out[4]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BM
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.9
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.7
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.3
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.5
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.1
...	...	...	...	...	...	...	...	...	...	...	...	...	..
4233	1	50	1.0	1	1.0	0.0	0	1	0	313.0	179.0	92.0	25.9
4234	1	51	3.0	1	43.0	0.0	0	0	0	207.0	126.5	80.0	19.7
4235	0	48	2.0	1	20.0	NaN	0	0	0	248.0	131.0	72.0	22.0
4236	0	44	1.0	1	15.0	0.0	0	0	0	210.0	126.5	87.0	19.1
4237	0	52	2.0	0	0.0	0.0	0	0	0	269.0	133.5	83.0	21.4

4238 rows × 16 columns

In [5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4238 entries, 0 to 4237
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   male             4238 non-null   int64  
 1   age              4238 non-null   int64  
 2   education        4133 non-null   float64
 3   currentSmoker    4238 non-null   int64  
 4   cigsPerDay       4209 non-null   float64
 5   BPMeds           4185 non-null   float64
 6   prevalentStroke  4238 non-null   int64  
 7   prevalentHyp     4238 non-null   int64  
 8   diabetes         4238 non-null   int64  
 9   totChol          4188 non-null   float64
 10  sysBP            4238 non-null   float64
 11  diaBP            4238 non-null   float64
 12  BMI              4219 non-null   float64
 13  heartRate        4237 non-null   float64
 14  glucose          3850 non-null   float64
 15  TenYearCHD       4238 non-null   int64  
dtypes: float64(9), int64(7)
memory usage: 529.9 KB
```

In [6]:

```
df.dropna(inplace=True)
```

In [7]:

```
df.columns
```

Out[7]:

```
Index(['male', 'age', 'education', 'currentSmoker', 'cigsPerDay', 'BPMeds',
       'prevalentStroke', 'prevalentHyp', 'diabetes', 'totChol', 'sysBP',
       'diaBP', 'BMI', 'heartRate', 'glucose', 'TenYearCHD'],
      dtype='object')
```

In [8]:

```python
df
```

Out[8]:

rentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHI
0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.97	80.0	77.0	
0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.73	95.0	76.0	
1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.34	75.0	70.0	
1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.58	65.0	103.0	
1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.10	85.0	85.0	
...	...	...	...	...	...	...	...	...	...	...	...	.
0	0.0	0.0	0	1	0	187.0	141.0	81.0	24.96	80.0	81.0	
0	0.0	0.0	0	1	0	176.0	168.0	97.0	23.14	60.0	79.0	
1	1.0	0.0	0	1	0	313.0	179.0	92.0	25.97	66.0	86.0	
1	43.0	0.0	0	0	0	207.0	126.5	80.0	19.71	65.0	68.0	
0	0.0	0.0	0	0	0	269.0	133.5	83.0	21.47	80.0	107.0	

In [9]:

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

In [10]:

```python
logr =LogisticRegression()
```

In [12]:

```python
feature_matrix=df[['age', 'education', 'cigsPerDay', 'BPMeds',
          'prevalentStroke',
          'totChol', 'sysBP',
          'diaBP', 'BMI', 'heartRate', 'glucose']]
target_vector=df['currentSmoker']
```

In [13]:

```python
feature_matrix.shape
```

Out[13]:

```
(3656, 11)
```

In [14]:
```
target_vector.shape
```
Out[14]:
```
(3656,)
```

In [15]:
```
from sklearn.preprocessing import StandardScaler
```

In [16]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [17]:
```
logr=LogisticRegression()
logr.fit(fs,target_vector)
```
Out[17]:
```
LogisticRegression()
```

In [22]:
```
observation=[[1,2,3,4,5,6,7,8,9,10,11]]
```

In [23]:
```
prediction = logr.predict(observation)
print(prediction)
```
```
[1]
```

In [24]:
```
logr.classes_
```
Out[24]:
```
array([0, 1], dtype=int64)
```

In [25]:
```
logr.predict_proba(observation)[0][1]
```
Out[25]:
```
1.0
```

In [26]:
```
logr.predict_proba(observation)[0][0]
```
Out[26]:
```
0.0
```

Linear regression 2

In [27]:

```python
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
from sklearn.model_selection import train_test_split
```

In [28]:

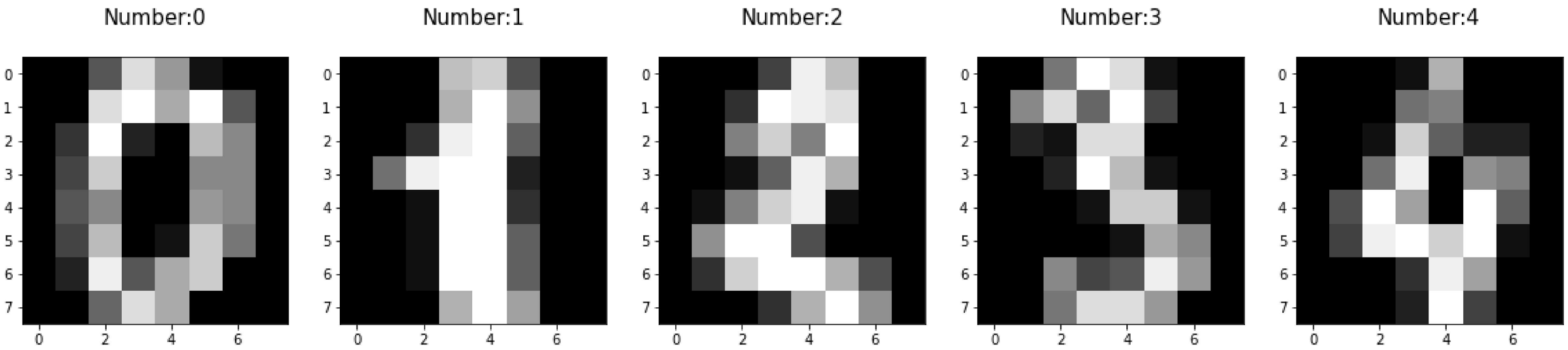
```python
digits =load_digits()
digits
```

Out[28]:

```
{'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 [29]:

```
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)
```



Number:0 Number:1 Number:2 Number:3 Number:4

In [30]:

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

In [31]:

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

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

In [32]:

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

Out[32]:

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

In [33]:

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

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

In [34]:

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

```
0.9555555555555556
```