In [1]:
```
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.preprocessing import StandardScaler
```

In [2]:
```
from sklearn.linear_model import LogisticRegression
```

In [3]:
```
df=pd.read_csv(r"E:\154\C6_bmi - C6_bmi.csv").dropna()
df
```

Out[3]:

	Gender	Height	Weight	Index
0	Male	174	96	4
1	Male	189	87	2
2	Female	185	110	4
3	Female	195	104	3
4	Male	149	61	3
...	...	...	...	...
495	Female	150	153	5
496	Female	184	121	4
497	Female	141	136	5
498	Male	150	95	5
499	Male	173	131	5

500 rows × 4 columns

In [4]:
```
df.head()
```

Out[4]:

	Gender	Height	Weight	Index
0	Male	174	96	4
1	Male	189	87	2
2	Female	185	110	4
3	Female	195	104	3
4	Male	149	61	3

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 500 entries, 0 to 499
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Gender  500 non-null    object
 1   Height  500 non-null    int64 
 2   Weight  500 non-null    int64 
 3   Index   500 non-null    int64 
dtypes: int64(3), object(1)
memory usage: 19.5+ KB
```

In [6]:
```
df.describe()
```

Out[6]:

	Height	Weight	Index
count	500.000000	500.000000	500.000000
mean	169.944000	106.000000	3.748000
std	16.375261	32.382607	1.355053
min	140.000000	50.000000	0.000000
25%	156.000000	80.000000	3.000000
50%	170.500000	106.000000	4.000000
75%	184.000000	136.000000	5.000000
max	199.000000	160.000000	5.000000

In [7]:
```
df.columns
```

Out[7]:
```
Index(['Gender', 'Height', 'Weight', 'Index'], dtype='object')
```

In [13]:
```
feature_matrix = df[['Height','Weight']]
target_vector = df[['Index']]
```

In [14]:
```
fs=StandardScaler().fit_transform(feature_matrix)
logr=LogisticRegression()
logr.fit(fs,target_vector)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

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

In [20]:
```
observation=[[1,2]]
```

In [21]:
```
prediction=logr.predict(observation)
print(prediction)
```

```
[5]
```

In [22]:
```
logr.classes_
```

Out[22]:
```
array([0, 1, 2, 3, 4, 5], dtype=int64)
```

In [23]:
```
logr.predict_proba(observation)[0][0]
```

Out[23]:
```
5.5956697582538237e-11
```

In [24]:
```
logr.predict_proba(observation)[0][1]
```

Out[24]:
```
6.059900360819463e-10
```

In [ ]:

In [ ]: