In [66]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [67]:
```
from sklearn.linear_model import LogisticRegression
```

In [68]:
```
df=pd.read_csv(r"C:\Users\Admin\Downloads\C5_health care diabetes - C5_health care diabetes
df
```

Out[68]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcor
0	6	148	72	35	0	33.6	0.627	50	
1	1	85	66	29	0	26.6	0.351	31	
2	8	183	64	0	0	23.3	0.672	32	
3	1	89	66	23	94	28.1	0.167	21	
4	0	137	40	35	168	43.1	2.288	33	
...	...	...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.171	63	
764	2	122	70	27	0	36.8	0.340	27	
765	5	121	72	23	112	26.2	0.245	30	
766	1	126	60	0	0	30.1	0.349	47	
767	1	93	70	31	0	30.4	0.315	23	

768 rows × 9 columns

In [69]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [70]:
```
df1=df.dropna()
```

In [71]:
```
d1=df1.iloc[:,0:5]
d2=df1.iloc[:,-1]
```

In [72]:
```
d1.shape
```

Out[72]:
```
(768, 5)
```

In [73]:
```
d2.shape
```

Out[73]:
```
(768,)
```

In [74]:
```
from sklearn.preprocessing import StandardScaler
```

In [75]:
```
a=StandardScaler().fit_transform(d1)
```

In [76]:
```
lr=LogisticRegression()
lr.fit(a,d2)
```

Out[76]:
```
LogisticRegression()
```

In [77]:
```
obs=[[10,12,13,14,5]]
```

In [78]:
```
pdt=lr.predict(obs)
print(pdt)
```

```
[1]
```

In [79]:
```
lr.classes_
```

Out[79]:
```
array([0, 1], dtype=int64)
```

In [80]:
```
lr.predict_proba(obs)[0][0]
```

Out[80]:
```
3.874164766770605e-09
```

In [81]:
```
lr.predict_proba(obs)[0][1]
```

Out[81]:
```
0.9999999961258352
```

In [ ]:

In [ ]: