Problem : We are given data like age and bought_insurance. Apply Logistic Regression Model and predict whether a person takes insurance or not based on his age.

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')

from sklearn.linear_model import LogisticRegression
```

In [2]:
```
#load the data into dataframe
df = pd.read_csv("insurance_Data.csv")
df
```

Out[2]:

	age	bought_insurance
0	22	0
1	25	0
2	47	1
3	52	0
4	46	1
5	56	1
6	55	0
7	60	1
8	62	1
9	61	1
10	18	0
11	28	0
12	27	0
13	29	0
14	49	1

In [3]:
```
#retreive the data
x = df.iloc[:, :-1].values #retreive only 0th column
x
```

Out[3]:
```
array([[22],
       [25],
       [47],
       [52],
       [46],
       [56],
       [55],
       [60],
       [62],
       [61],
       [18],
       [28],
       [27],
       [29],
       [49]])
```

Note: We need to get a 2D column afterwards, retreiving it as [:, :0] will keep it 1D and it wont work. So we use :-1 to get a column data to multiply it with y and get 2D

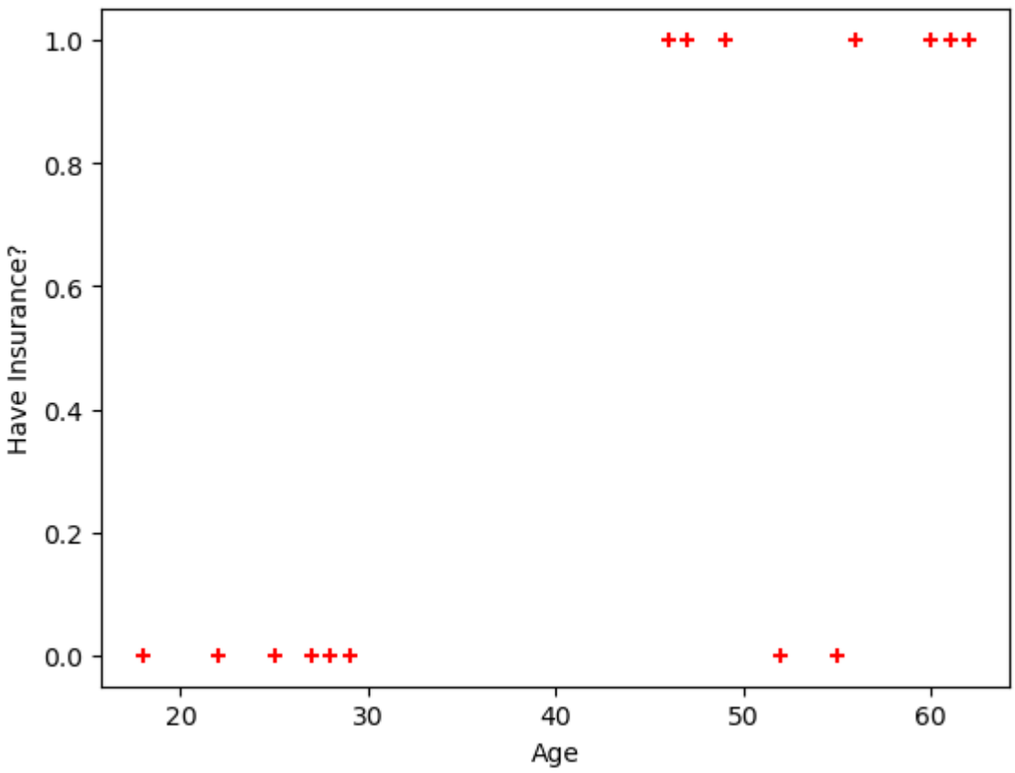
In [4]:
```
y = df.iloc[:, 1].values #retreive 1st column
y
```

Out[4]:
```
array([0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1])
```

Display the scatter plot to know how the datapoints are aligned -

In [5]:
```
plt.xlabel('Age')
plt.ylabel('Have Insurance?')
plt.scatter(x,y, marker='+', color='red')
```

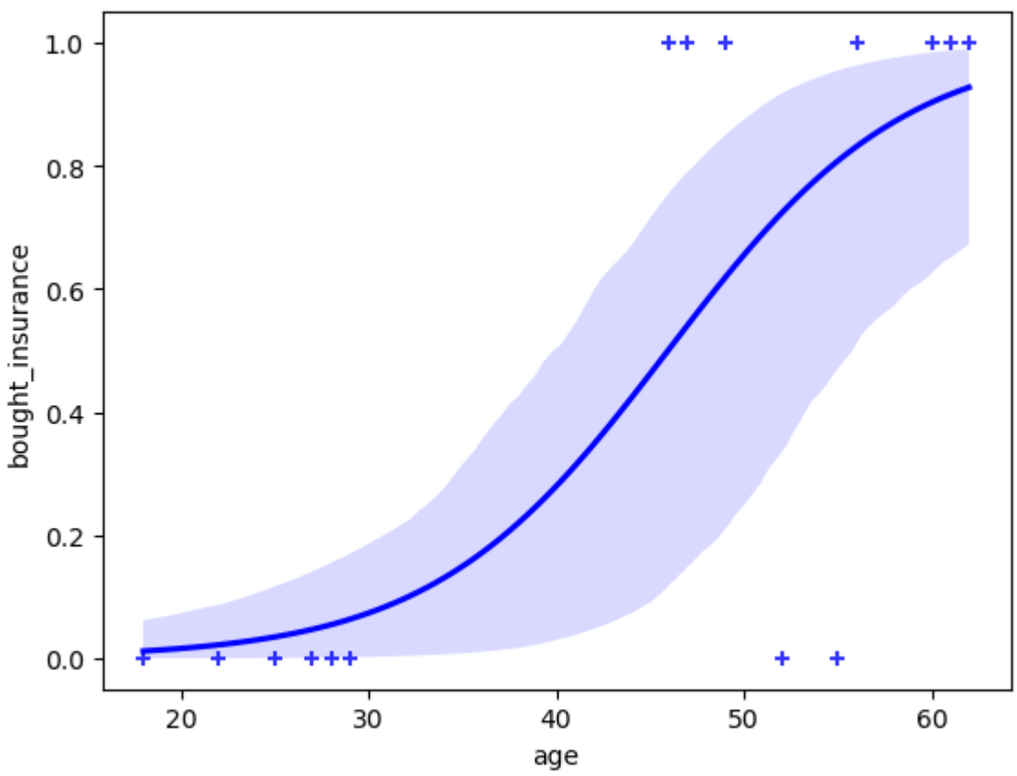
Out[5]:
```
<matplotlib.collections.PathCollection at 0x7f653f0a12d0>
```



For the above data we cannot use **Linear regression** as we can see from the graph itself that there is a linear relationship does not exist here, so we use a **Logistic Regression Model** for this -

In [6]:
```
plt.xlabel('Age')
plt.ylabel('Probability of Buying Insurance')
sns.regplot(data=df, x='age', y='bought_insurance', logistic=True, marker='+', colo
```

Out[6]:
```
<Axes: xlabel='age', ylabel='bought_insurance'>
```



You might get a RuntimeWarning: overflow encountered in [exp t = np.exp(-z)], but we can ignore it using the filter warnings that we imported from the warnings library

In [7]:
```
#create logistic regression model
model = LogisticRegression()
```

In [8]:
```
#train the model
model.fit(x,y)
```

Out[8]:
```
▾ LogisticRegression

LogisticRegression()
```

In [9]:
```
#find the accuracy of the model
model.score(x,y)
```

Out[9]:
```
0.8666666666666667
```

Prediction 1 : Let us now predict if a 66 years aged person will buy insurance or not -

In [10]:
```
model.predict([[56]]) #array([[1]]) means Yes
```

Out[10]:
```
array([1])
```

Prediction 2 : Let us now predict if a 23 years old person will buy insurance or not -

In [11]:
```
#predict if 23 years aged person will buy insurance or not
model.predict([[23]]) #array([[0]]) means No
```

Out[11]:
```
array([0])
```