In [1]:

```
import numpy as np
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from seaborn import load_dataset
from sklearn.linear_model import LogisticRegression, LinearRegression
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
df = load_dataset('iris')
df.head()
```

Out[2]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [4]:

```
df['species'].unique()
```

Out[4]:

```
array(['setosa', 'versicolor', 'virginica'], dtype=object)
```

In [5]:

```
df1=df[df['species']!='versicolor']
df1
```

Out[5]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

100 rows × 5 columns

In [6]:

```
df1['species'].unique()
```

Out[6]:

```
array(['setosa', 'virginica'], dtype=object)
```

In [7]:

```
lr = LinearRegression()
```

In [8]:

```
lb = LabelEncoder()
```

In [9]:

```
df1['species'] = lb.fit_transform(df1['species'])
df1
```

Out[9]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	1
146	6.3	2.5	5.0	1.9	1
147	6.5	3.0	5.2	2.0	1
148	6.2	3.4	5.4	2.3	1
149	5.9	3.0	5.1	1.8	1

100 rows × 5 columns

In [10]:

```
x = df1.iloc[:,0].values.reshape(-1,1)
y = df1.iloc[:,4].values.reshape(-1,1)
```

In [11]:

```
x
```

```
       [5.8],
       [5.7],
       [5.4],
       [5.1],
       [5.7],
       [5.1],
       [5.4],
       [5.1],
       [4.6],
       [5.1],
       [4.8],
       [5.  ],
       [5.  ],
       [5.2],
       [5.2],
       [4.7],
       [4.8],
       [5.4],
       [5.2],
       [5.5],
```

In [12]:

```
y
```

```
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
       [1],
```

In [13]:

```
lr.fit(x,y)
```

Out[13]:

```
LinearRegression()
```

In [ ]:

```

```

In [14]:

```
x_train, x_test, y_train, y_test = train_test_split(x, y,test_size=0.3, random_state
```

In [15]:

```
lr.fit(x_train, y_train)
```

Out[15]:

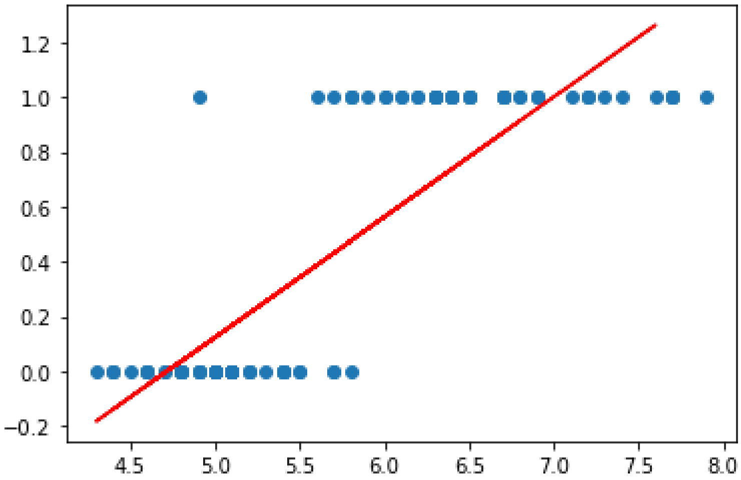
```
LinearRegression()
```

In [16]:

```
plt.scatter(x,y)
plt.plot(x_test, lr.predict(x_test), color = 'r')
```

Out[16]:

```
[<matplotlib.lines.Line2D at 0x1ef6a857460>]
```



In [17]:

```
loglr = LogisticRegression()
```

In [18]:

```
loglr.fit(x_test, y_test)
```

Out[18]:

```
LogisticRegression()
```

In [24]:

```
x_test
```

Out[24]:

```
array([[5. ],
       [6.3],
       [4.7],
       [7.6],
       [7.2],
       [6.8],
       [5.4],
       [6.3],
       [6.5],
       [6.7],
       [6.3],
       [5.8],
       [6.4],
       [4.3],
       [5. ],
       [4.8],
       [4.6],
       [4.8],
       [5.5],
       [4.4],
       [5. ],
       [6.8],
       [4.6],
       [5.6],
       [4.8],
       [5.3],
       [4.6],
       [5.9],
       [6.4],
       [6.2]])
```

In [23]:

```
y_pred = loglr.predict(x_test)
y_pred
```

Out[23]:

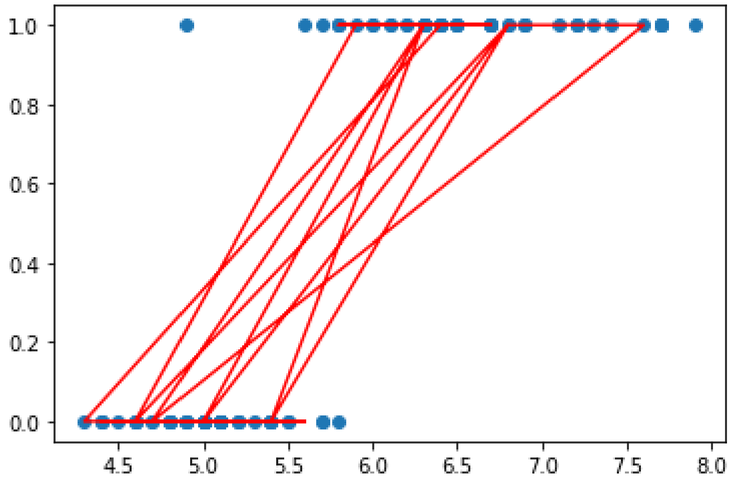
```
array([0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 1, 1, 1])
```

In [19]:

```
plt.scatter(x,y)
plt.plot(x_test, loglr.predict(x_test), color = 'r')
```

Out[19]:

```
[<matplotlib.lines.Line2D at 0x1ef6a978610>]
```

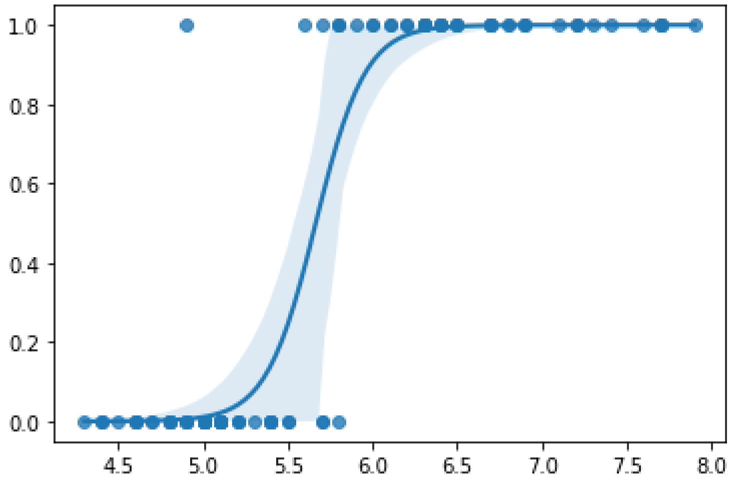


In [20]:

```
sns.regplot(x,y,logistic=True)
```

Out[20]:

```
<AxesSubplot:>
```



Confusion Matrix

In [30]:

```
df1 = pd.DataFrame(y_test)
df2 = pd.DataFrame(y_pred)
pd.concat([df1, df2],axis = 1)
```

Out[30]:

	0	0
0	0	0
1	1	1
2	0	0
3	1	1
4	1	1
5	1	1
6	0	0
7	1	1
8	1	1
9	1	1
10	1	1
11	1	1
12	1	1
13	0	0
14	0	0
15	0	0
16	0	0
17	0	0
18	0	0
19	0	0
20	0	0
21	1	1
22	0	0
23	1	0
24	0	0
25	0	0
26	0	0
27	1	1
28	1	1
29	1	1

In [26]:

```
confusion_matrix(y_test, y_pred)
```

Out[26]:

```
array([[15,  0],
       [ 1, 14]], dtype=int64)
```

In [31]:

```
accuracy_score(y_test,y_pred)
```

Out[31]:

```
0.9666666666666667
```

In [ ]:

```

```