With scikit learn

In [3]:
```
# Step 1: Import packages, functions, and classes
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
# Step 2: Get data
x = np.arange(10).reshape(-1, 1)
y = np.array([0, 1, 0, 0, 1, 1, 1, 1, 1, 1])
# Step 3: Create a model and train it
model = LogisticRegression(solver='liblinear', C=10.0, random_state=0)
model.fit(x, y)
# Step 4: Evaluate the model
p_pred = model.predict_proba(x)
y_pred = model.predict(x)
score_ = model.score(x, y)
conf_m = confusion_matrix(y, y_pred)
report = classification_report(y, y_pred)
```

In [4]:
```
print('x:', x, sep='\n')
```

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

In [5]:
```
print('y:', y, sep='\n', end='\n\n')
```

```
y:
[0 1 0 0 1 1 1 1 1 1]
```

In [6]:
```
print('intercept:', model.intercept_)
print('coef:', model.coef_, end='\n\n')
print('p_pred:', p_pred, sep='\n', end='\n\n')
```

```
intercept: [-1.51632619]
coef: [[0.703457]]

p_pred:
[[0.81999686 0.18000314]
 [0.69272057 0.30727943]
 [0.52732579 0.47267421]
 [0.35570732 0.64429268]
 [0.21458576 0.78541424]
 [0.11910229 0.88089771]
 [0.06271329 0.93728671]
 [0.03205032 0.96794968]
 [0.0161218  0.9838782 ]
 [0.00804372 0.99195628]]
```

In [7]:
```
print('y_pred:', y_pred, end='\n\n')
```

```
y_pred: [0 0 0 1 1 1 1 1 1 1]
```

In [9]:
```
print('score_:', score_, end='\n\n')

print('conf_m:', conf_m, sep='\n', end='\n\n')
```

```
score_: 0.8

conf_m:
[[2 1]
 [1 6]]
```

In [11]:
```
print('report:', report, sep='\n')
```

```
report:
              precision    recall  f1-score   support

           0       0.67      0.67      0.67         3
           1       0.86      0.86      0.86         7

    accuracy                           0.80        10
   macro avg       0.76      0.76      0.76        10
weighted avg       0.80      0.80      0.80        10
```

With stats models

In [12]:
```
# Step 1: Import Packages
import numpy as np
import statsmodels.api as sm
```

In [14]:
```
# Step 2: Get Data
x = np.arange(10).reshape(-1, 1)
y = np.array([0, 1, 0, 0, 1, 1, 1, 1, 1, 1])
x = sm.add_constant(x)
x, y

# The first column of x corresponds to the intercept b₀.
# The second column contains the original values of x.
```

Out[14]:
```
(array([[1., 0.],
        [1., 1.],
        [1., 2.],
        [1., 3.],
        [1., 4.],
        [1., 5.],
        [1., 6.],
        [1., 7.],
        [1., 8.],
        [1., 9.]]),
 array([0, 1, 0, 0, 1, 1, 1, 1, 1, 1]))
```

In [16]:
```
# Step 3: Create a Model and Train It
#Note that the first argument here is y, followed by x.
model = sm.Logit(y, x)
result = model.fit(method='newton')
```

```
Optimization terminated successfully.
         Current function value: 0.350471
         Iterations 7
```

In [17]:
```
# obtain the values of b₀ and b₁ respectively with .params:
result.params
```

Out[17]:
```
array([-1.972805  ,  0.82240094])
```

In [18]:
```
# Step 4: Evaluate the Model
result.predict(x)
```

Out[18]:
```
array([0.12208792, 0.24041529, 0.41872657, 0.62114189, 0.78864861,
       0.89465521, 0.95080891, 0.97777369, 0.99011108, 0.99563083])
```

In [19]:
```
(result.predict(x) >= 0.5).astype(int)
# array contains the predicted output values
```

Out[19]:
```
array([0, 0, 0, 1, 1, 1, 1, 1, 1, 1])
```

In [21]:
```
# obtain the confusion matrix with .pred_table():
result.pred_table()
```

Out[21]:
```
array([[2., 1.],
       [1., 6.]])
```

In [23]:
```
# .summary() and .summary2() get output data that you might find
# useful in some circumstances:
result.summary()
```

Out[23]:

Logit Regression Results

Dep. Variable:	y	**No. Observations:**	10
Model:	Logit	**Df Residuals:**	8
Method:	MLE	**Df Model:**	1
Date:	Wed, 21 Sep 2022	**Pseudo R-squ.:**	0.4263
Time:	11:37:26	**Log-Likelihood:**	-3.5047
converged:	True	**LL-Null:**	-6.1086
Covariance Type:	nonrobust	**LLR p-value:**	0.02248

	coef	std err	z	P>|z|	[0.025	0.975]
const	-1.9728	1.737	-1.136	0.256	-5.377	1.431
x1	0.8224	0.528	1.557	0.119	-0.213	1.858

In [24]:
```
result.summary2()
```

Out[24]:

Model:	Logit	Pseudo R-squared:	0.426
Dependent Variable:	y	AIC:	11.0094
Date:	2022-09-21 11:37	BIC:	11.6146
No. Observations:	10	Log-Likelihood:	-3.5047
Df Model:	1	LL-Null:	-6.1086
Df Residuals:	8	LLR p-value:	0.022485
Converged:	1.0000	Scale:	1.0000
No. Iterations:	7.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
const	-1.9728	1.7366	-1.1360	0.2560	-5.3765	1.4309
x1	0.8224	0.5281	1.5572	0.1194	-0.2127	1.8575

In [ ]:
```

```