In [1]:
```
import pandas as pd
import numpy as np
import tensorflow as tf
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import ConfusionMatrixDisplay, accuracy_score, classificati
```

In [2]:
```
df = pd.read_csv('index.csv')
```

In [3]:
```
df.isnull().sum()
```

Out[3]:
```
pregnancies                   0
plasma glucose concentration  0
diastolic blood pressure      0
triceps skinfold thickness    0
insulin                       0
body mass index               0
diabetes pedigree function    0
age                           0
diabetic                      0
dtype: int64
```

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

Out[4]:

	pregnancies	plasma glucose concentration	diastolic blood pressure	triceps skinfold thickness	insulin	body mass index	diabetes pedigree function	age	diab
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	

In [5]:
```
X = df.iloc[:,:-1].values
Y = df.iloc[:,-1].values
```

In [6]:
```
Xtrain,Xtest,Ytrain,Ytest = train_test_split(X,Y,test_size=0.2,random_state=42)
```

In [7]:
```
SS = StandardScaler()
```

In [8]:
```
Xtrain = SS.fit_transform(Xtrain)
```

In [9]:
```
Xtest = SS.transform(Xtest)
```

In [10]:
```
model = tf.keras.models.Sequential()
```

In [11]:
```
model.add(tf.keras.layers.Dense(units = 8, activation='relu'))
```

In [12]:
```
model.add(tf.keras.layers.Dense(units = 1, activation='sigmoid'))
```

In [13]:
```
model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])
```

In [14]:
```
model.fit(Xtrain,Ytrain,batch_size=8,epochs=50)
```

```
Epoch 1/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 3s 6ms/step - accuracy: 0.5863 - loss: 4.2695
Epoch 2/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.5912 - loss: 4.0834
Epoch 3/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.5961 - loss: 3.9394
Epoch 4/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6059 - loss: 3.8277
Epoch 5/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6091 - loss: 3.6051
Epoch 6/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6156 - loss: 3.5121
Epoch 7/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6270 - loss: 3.4096
Epoch 8/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6287 - loss: 3.3598
Epoch 9/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6303 - loss: 3.2896
Epoch 10/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6319 - loss: 3.2267
Epoch 11/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.6368 - loss: 3.1830
Epoch 12/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6368 - loss: 2.9807
Epoch 13/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6352 - loss: 2.7867
Epoch 14/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6368 - loss: 2.6796
Epoch 15/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6401 - loss: 2.6462
Epoch 16/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6401 - loss: 2.5722
Epoch 17/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6482 - loss: 2.3718
Epoch 18/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6547 - loss: 2.2115
Epoch 19/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6564 - loss: 1.9741
Epoch 20/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6612 - loss: 1.6021
Epoch 21/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.6661 - loss: 1.0346
Epoch 22/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6694 - loss: 0.8677
Epoch 23/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6678 - loss: 0.8453
Epoch 24/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.6694 - loss: 0.8307
Epoch 25/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.6694 - loss: 0.8194
Epoch 26/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6759 - loss: 0.8096
Epoch 27/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6775 - loss: 0.7456
Epoch 28/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.6792 - loss: 0.7286
Epoch 29/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.6840 - loss: 0.7210
Epoch 30/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6857 - loss: 0.7140
```

```
Epoch 31/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6857 - loss: 0.7080
Epoch 32/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6906 - loss: 0.7018
Epoch 33/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6938 - loss: 0.6969
Epoch 34/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6922 - loss: 0.6914
Epoch 35/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.6938 - loss: 0.6866
Epoch 36/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.6971 - loss: 0.6822
Epoch 37/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.7020 - loss: 0.6598
Epoch 38/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.7003 - loss: 0.6516
Epoch 39/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.7052 - loss: 0.6479
Epoch 40/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.7085 - loss: 0.6442
Epoch 41/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.7101 - loss: 0.6404
Epoch 42/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.7150 - loss: 0.6191
Epoch 43/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.7313 - loss: 0.6114
Epoch 44/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.7296 - loss: 0.6083
Epoch 45/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.7329 - loss: 0.6046
Epoch 46/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.7362 - loss: 0.6022
Epoch 47/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.7378 - loss: 0.5833
Epoch 48/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.7280 - loss: 0.5987
Epoch 49/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.7264 - loss: 0.5668
Epoch 50/50
77/77 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.7150 - loss: 0.5553
```

Out[14]: <keras.src.callbacks.history.History at 0x2852f121fd0>

In [15]:
```
Ypred = model.predict(Xtest)
```

```
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step
```

In [16]:
```
Ypred= [1 if y>0.5 else 0 for y in Ypred]
```

In [17]:
```
Ypred
```

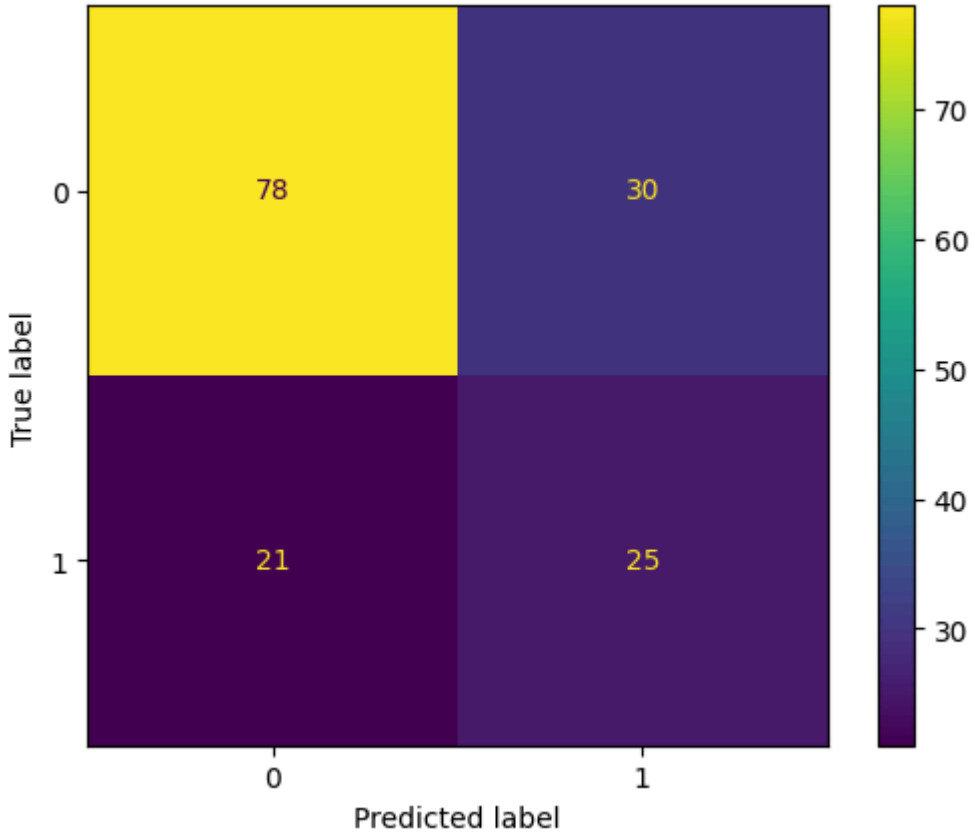
Out[17]:
```
[0,
 0,
 0,
 0,
 0,
 1,
 0,
 0,
 0,
 1,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 1,
 1,
 1,
 0,
 1,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 1,
 1,
 1,
 0,
 1,
 1,
 1,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 1,
 0,
 1,
 0,
```

```
1,
0,
1,
0,
0,
1,
0,
0,
0,
1,
0,
0,
1,
0,
0,
0,
0,
0,
0,
0,
0,
0,
0,
0,
1,
1,
0,
0,
0,
0,
0,
0,
0,
1,
1,
1,
0,
0,
0,
1,
0,
0,
0,
1,
0,
0,
0,
1,
1,
1,
0,
0,
0,
0,
0,
0,
0,
1,
0,
0,
1,
```

```
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 1,
 1,
 1,
 1,
 1,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 1,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 1,
 0,
 1]
```

In [18]:
```
ConfusionMatrixDisplay.from_predictions(Ypred,Ytest)
```

Out[18]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x2852f121400
>
```



In [19]:
```
print(f'accuracy: {accuracy_score(Ypred,Ytest)}')
print(f'{classification_report(Ypred,Ytest)}')
```

```
accuracy: 0.6688311688311688
              precision    recall  f1-score   support

           0       0.79      0.72      0.75       108
           1       0.45      0.54      0.50        46

    accuracy                           0.67       154
   macro avg       0.62      0.63      0.62       154
weighted avg       0.69      0.67      0.68       154
```

In [ ]: