In [43]:

```
import numpy as np
import pandas as pd
from pathlib import Path
import os
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Set random seed for reproducibility
np.random.seed(0)

# Generate synthetic dataset
n_samples = 100
X = np.random.randn(100,2)#the two independent variables

y= np.random.randint( 0, 2, 100)# the binary target variable(dependant)

X[y == 0,1] = X[y == 0,0] +3
X[y == 1,1] = X[y == 1,0] -3

# Create a DataFrame to store the dataset
df = pd.DataFrame(data= X , columns=['Exercise_duration','Heart_rate'])
df['STATUS'] = y
```

In [32]:

```
df
```

Out[32]:

	Exercise_duration	Heart_rate	STATUS
0	1.764052	4.764052	0
1	0.978738	-2.021262	1
2	1.867558	4.867558	0
3	0.950088	-2.049912	1
4	-0.103219	2.896781	0
...	...	...	...
95	-1.292857	-4.292857	1
96	-0.039283	-3.039283	1
97	0.523277	-2.476723	1
98	0.771791	3.771791	0
99	2.163236	5.163236	0

100 rows × 3 columns

In [44]:

```
data = pd.read_csv("C:\\Users\\santo\\Downloads\\Exercise_study_dataset.csv")
# Save DataFrame to CSv
```

In [45]:

```
data
```

Out[45]:

```
     Unnamed: 0  Exercise_duration  Heart_rate  STATUS
 0            0           1.764052    4.764052       0
 1            1           0.978738   -2.021262       1
 2            2           1.867558    4.867558       0
 3            3           0.950088   -2.049912       1
 4            4          -0.103219    2.896781       0
...         ...                ...         ...     ...
95           95          -1.292857   -4.292857       1
96           96          -0.039283   -3.039283       1
97           97           0.523277   -2.476723       1
98           98           0.771791    3.771791       0
99           99           2.163236    5.163236       0

100 rows × 4 columns
```

In [46]:

```
# defining the independent and target variables
X = np.array(data[[ "Exercise_duration","Heart_rate"]])
y = np.array(data["STATUS"])
```

In [47]:

```
X
```

```
Out[47]: array([[ 1.76405235,  4.76405235],
               [ 0.97873798, -2.02126202],
               [ 1.86755799,  4.86755799],
               [ 0.95008842, -2.04991158],
               [-0.10321885,  2.89678115],
               [ 0.14404357, -2.85595643],
               [ 0.76103773,  3.76103773],
               [ 0.44386323, -2.55613677],
               [ 1.49407907, -1.50592093],
               [ 0.3130677 ,  3.3130677 ],
               [-2.55298982,  0.44701018],
               [ 0.8644362 , -2.1355638 ],
               [ 2.26975462,  5.26975462],
               [ 0.04575852, -2.95424148],
               [ 1.53277921,  4.53277921],
               [ 0.15494743, -2.84505257],
               [-0.88778575,  2.11221425],
               [-0.34791215,  2.65208785],
               [ 1.23029068,  4.23029068],
               [-0.38732682,  2.61267318],
               [-1.04855297, -4.04855297],
               [-1.70627019, -4.70627019],
               [-0.50965218,  2.49034782],
               [-1.25279536, -4.25279536],
               [-1.61389785,  1.38610215],
               [-0.89546656, -3.89546656],
               [-0.51080514, -3.51080514],
               [-0.02818223,  2.97181777],
               [ 0.06651722, -2.93348278],
               [-0.63432209,  2.36567791],
               [-0.67246045,  2.32753955],
               [-0.81314628,  2.18685372],
               [ 0.17742614,  3.17742614],
               [-1.63019835,  1.36980165],
               [-0.90729836, -3.90729836],
               [ 0.72909056, -2.27090944],
               [ 1.13940068,  4.13940068],
               [ 0.40234164,  3.40234164],
               [-0.87079715,  2.12920285],
               [-0.31155253,  2.68844747],
               [-1.16514984, -4.16514984],
               [ 0.46566244, -2.53433756],
               [ 1.48825219,  4.48825219],
               [ 1.17877957,  4.17877957],
               [-1.07075262, -4.07075262],
               [-0.40317695,  2.59682305],
               [ 0.20827498, -2.79172502],
               [ 0.3563664 , -2.6436336 ],
               [ 0.01050002, -2.98949998],
               [ 0.12691209, -2.87308791],
               [ 1.8831507 , -1.1168493 ],
               [-1.270485  , -4.270485  ],
               [-1.17312341, -4.17312341],
               [-0.41361898, -3.41361898],
               [ 1.92294203, -1.07705797],
               [ 1.86755896, -1.13244104],
               [-0.86122569, -3.86122569],
       [-0.26800337,  2.73199663],
       [ 0.94725197,  3.94725197],
       [ 0.61407937, -2.38592063],
       [ 0.37642553, -2.62357447],
       [ 0.29823817, -2.70176183],
       [-0.69456786, -3.69456786],
       [-0.43515355,  2.56484645],
       [ 0.67229476,  3.67229476],
       [-0.76991607, -3.76991607],
       [-0.67433266,  2.32566734],
       [-0.63584608,  2.36415392],
       [ 0.57659082, -2.42340918],
       [ 0.39600671,  3.39600671],
       [-1.49125759, -4.49125759],
       [ 0.1666735 ,  3.1666735 ],
       [ 2.38314477, -0.61685523],
       [-0.91282223,  2.08717777],
       [-1.31590741,  1.68409259],
       [-0.06824161, -3.06824161],
       [-0.74475482,  2.25524518],
       [-0.09845252,  2.90154748],
       [ 1.12663592,  4.12663592],
       [-1.14746865,  1.85253135],
       [-0.49803245,  2.50196755],
       [ 0.94942081, -2.05057919],
       [-1.22543552,  1.77456448],
       [-1.00021535, -4.00021535],
       [ 1.18802979,  4.18802979],
       [ 0.92085882,  3.92085882],
       [ 0.85683061, -2.14316939],
       [-1.03424284, -4.03424284],
       [-0.80340966, -3.80340966],
       [-0.4555325 , -3.4555325 ],
       [-0.35399391,  2.64600609],
       [-0.6436184 ,  2.3563816 ],
       [ 0.62523145, -2.37476855],
       [-1.10438334,  1.89561666],
       [-0.739563  , -3.739563  ],
       [-1.29285691, -4.29285691],
       [-0.03928282, -3.03928282],
       [ 0.52327666, -2.47672334],
       [ 0.77179055,  3.77179055],
       [ 2.16323595,  5.16323595]])
```

```
In [50]: y
```

```
Out[50]: array([0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1,
                0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0,
                1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1,
                0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1,
                1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0], dtype=int64)
```

In [51]:

```
# spliting the data into training and testing dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size= 0.2, rando
```

In [52]:

```
# buiding the model
model = LogisticRegression()
```

In [54]:

```
# fitting the model
model.fit(X_train, y_train)
```

Out[54]:

```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [56]:

```
y_pred = model.predict(X_test)
```

In [58]:

```
y_pred
```

Out[58]:

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

In [61]:

```
accuracy = accuracy_score(y_test, y_pred)
print("accuaracy", accuracy)
```

```
accuaracy 1.0
```

model optimization

In [87]:

```
data = pd.read_csv("C:\\Users\\santo\\Downloads\\Exercise_study_dataset.csv ")
```

In [88]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV
```

In [89]:
```
data
```

Out[89]:
```
    Unnamed: 0  Exercise_duration  Heart_rate  STATUS
 0           0           1.764052    4.764052       0
 1           1           0.978738   -2.021262       1
 2           2           1.867558    4.867558       0
 3           3           0.950088   -2.049912       1
 4           4          -0.103219    2.896781       0
...        ...                ...         ...     ...
95          95          -1.292857   -4.292857       1
96          96          -0.039283   -3.039283       1
97          97           0.523277   -2.476723       1
98          98           0.771791    3.771791       0
99          99           2.163236    5.163236       0

100 rows × 4 columns
```

In [90]:
```
# defining the independent and target variables
X = np.array(data[[ "Exercise_duration","Heart_rate"]])
y = np.array(data["STATUS"])
```

In [91]:
```
# spliting the data into training and testing dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size= 0.2, rando
```

In [92]:
```
model = LogisticRegression()
```

In [102]:
```
param_grid = {
    "C" :[0.01,0.1,1],
    "penalty": ["l1","l2"],
    "max_iter": [10,100,1000]

}
```

```
In [103]: grid_search = GridSearchCV(model, param_grid, cv = 5)
          grid_search.fit(X_train, y_train)
```

```
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:
460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:
460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:
460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:
460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:
460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
```

```
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:
460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:
460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\model_selection\_validatio
n.py:425: FitFailedWarning:
45 fits failed out of a total of 90.
The score on these train-test partitions for these parameters will be set to
nan.
If these failures are not expected, you can try to debug them by setting erro
r_score='raise'.

Below are more details about the failures:
-----------------------------------------------------------------------------
---
45 fits failed with the following error:
Traceback (most recent call last):
  File "d:\Users\santo\anaconda3\Lib\site-packages\sklearn\model_selection\_v
alidation.py", line 732, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "d:\Users\santo\anaconda3\Lib\site-packages\sklearn\base.py", line 115
1, in wrapper
    return fit_method(estimator, *args, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logi
stic.py", line 1168, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
             ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logi
stic.py", line 56, in _check_solver
    raise ValueError(
ValueError: Solver lbfgs supports only 'l2' or 'none' penalties, got l1 penal
ty.
```

```
  warnings.warn(some_fits_failed_message, FitFailedWarning)
d:\Users\santo\anaconda3\Lib\site-packages\sklearn\model_selection\_search.p
y:976: UserWarning: One or more of the test scores are non-finite: [nan  1. n
an  1. nan  1. nan  1. nan  1. nan  1. nan  1. nan  1.]
  warnings.warn(
```

Out[103]:

```
GridSearchCV(cv=5, estimator=LogisticRegression(),
             param_grid={'C': [0.01, 0.1, 1], 'max_iter': [10, 100, 1000],
                         'penalty': ['l1', 'l2']})
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [100]:

```
best_params = grid_search.best_params_
```

In [109]:

```

print("best_param", best_param)
best_model = LogisticRegression(**best_params)
best_model.fit(X_train,y_train)
```

```
best_param LogisticRegression(C=[0.001, 0.01, 0.1, 1], max_iter=[10, 100, 100
0],
                   penalty=['l1', 'l2'])

d:\Users\santo\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:
460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
```

Out[109]:

```
LogisticRegression(max_iter=10)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [120]:

```
y_pred = best_model.predict(X_test)
```

In [126]:

```
y_pred
```

Out[126]:

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

```
In [114]: accuaracy = accuracy_score(y_test, y_pred)
```

```
In [117]: accuaracy

Out[117]: 1.0
```

```
In [118]: best_params

Out[118]: {'max_iter': 10, 'penalty': 'l2'}
```

```
In [ ]:
```