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

df = pd.read\_csv('heart\_disease\_data.csv')

df.head()

<del></del>		age	sex	ср	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
	0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
	1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
	2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
	3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
	4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

df.tail()

₹		age	sex	ср	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
	298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
	299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
	300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
	301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
	302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

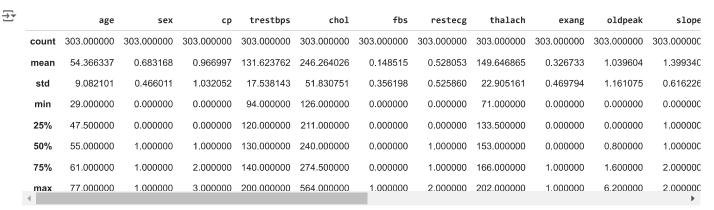
df.shape

**→** (303, 14)

df.info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 303 entries, 0 to 302 Data columns (total 14 columns): # Column Non-Null Count Dtype -----303 non-null 0 age int64 303 non-null 1 sex int64 303 non-null int64 trestbps 303 non-null int64 303 non-null int64 fbs 303 non-null int64 restecg 303 non-null int64 303 non-null int64 thalach exang oldpeak 303 non-null int64 303 non-null float64 int64 303 non-null 10 slope 11 ca 12 thal int64 303 non-null 303 non-null int64 13 target 303 non-null int64 dtypes: float64(1), int64(13) memory usage: 33.3 KB

df.describe()



df['target'].value\_counts()



Double-click (or enter) to edit

## 1 : Defective Heart

## 0: Healthy Heart

X = df.drop(columns='target',axis=1)

**₹** chol fbs thalach oldpeak slope thal age sex ср trestbps restecg exang ca 2.3 3.5 1.4 8.0 0.6 ... 0.2 1.2 3.4 1.2 0.0 303 rows x 13 columns

Y =df['target']

Υ

Х

```
<del>_</del>__
             target
        0
                    1
        1
                    1
        2
        3
                    1
       298
                    0
       299
       300
                    0
       301
                    0
       302
                    0
      303 rows × 1 columns
```

## **SPLITTING DATASET INTO TRAIN AND TEST**

```
X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, stratify=Y, random\_state=23)
print(X_test.shape,X_train.shape)
→ (61, 13) (242, 13)
MODEL TRAINING
model = LogisticRegression()
#Train the model with Training_data
model.fit(X_train, Y_train)
yusr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:460: ConvergenceWarning: lbfgs failed to converge (status=
     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
     Please also refer to the documentation for alternative solver options:
        https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
       n_iter_i = _check_optimize_result(
     ▼ LogisticRegression
     LogisticRegression()
Y_Pred = model.predict(X_test)
```

Y\_train

```
Heart Disease ipynb - Colab
<del>_</del>__
          target
      61
              1
score = accuracy_score(Y_test,Y_Pred)
print("The accuracy score is", score)
→ The accuracy score is 0.8524590163934426
BUILDUING A PREDECTIVE SYSTEM
input_data = (62, 0, 0, 140, 268, 0, 0, 160, 0, 3.6, 0, 2,2)
#Change the input_data into array
arr = np.array(input_data)
print(arr)
→ [ 62. 0. 0. 140. 268. 0. 0. 160. 0. 3.6 0. 2.
       2.]
prediction = model.predict(arr.reshape(1,-1))
🚁 /usr/local/lib/python3.10/dist-packages/sklearn/base.py:465: UserWarning: X does not have valid feature names, but LogisticRegressic
      warnings.warn(
    4
print(prediction)
→ [0]
if prediction == 0:
 print("The preson is Healthy")
else:
 print("Consult Your Doctor")

    The preson is Healthy
```