

Heart Disease Prediction By Random Forest

IMPORT DATASET

```
In [1]: import numpy as np
import pandas as pd
```

```
In [2]: df=pd.read_csv("heart.csv")
df.head()
```

Out[2]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
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

Age : Age of the patient

Sex : Sex of the patient

exang: exercise induced angina (1 = yes; 0 = no)

ca: number of major vessels (0-3)

cp : Chest Pain type chest pain type

Value 1: typical angina Value 2: atypical angina Value 3: non-anginal pain Value 4: asymptomatic trtbps :
resting blood pressure (in mm Hg)

chol : cholestoral in mg/dl fetched via BMI sensor

fbs : (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false) rest_ecg : resting electrocardiographic results

Value 0: normal Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV) Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria thalach : maximum heart rate achieved

target : 0= less chance of heart attack 1= more chance of heart attack

DATA INFO

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

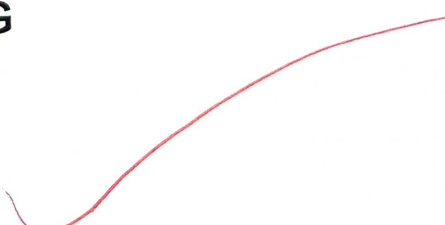
```
Out[3]: age      0
sex      0
cp       0
trtbps   0
chol     0
fbs      0
restecg  0
thalachh 0
exng     0
oldpeak  0
slp      0
caa      0
thall    0
output   0
dtype: int64
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
#   Column      Non-Null Count  Dtype
---  -
0   age         303 non-null   int64
1   sex         303 non-null   int64
2   cp          303 non-null   int64
3   trtbps      303 non-null   int64
4   chol        303 non-null   int64
5   fbs         303 non-null   int64
6   restecg     303 non-null   int64
7   thalachh    303 non-null   int64
8   exng        303 non-null   int64
9   oldpeak     303 non-null   float64
10  slp         303 non-null   int64
11  caa         303 non-null   int64
12  thall       303 non-null   int64
13  output      303 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

DATA SPLITTING

```
In [5]: x=df.iloc[:,13]
        y=df.iloc[:,13]
```



In [6]: x.head()

Out[6]:

	age	sex	cp	trtbps	chol	fb	restecg	thalachh	exng	oldpeak	slp	caa	thalf
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2

In [7]: y.head()

Out[7]: 0 1
1 1
2 1
3 1
4 1

Name: output, dtype: int64

In [8]: print("Shape of X : ",x.shape)
print("Shape of Y : ",y.shape)

Shape of X : (303, 13)

Shape of Y : (303,)

In [9]: from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=51)

In [10]: print("Shape of X_train : ",x_train.shape)
print("Shape of Y_train : ",y_train.shape)
print("Shape of X_test : ",x_test.shape)
print("Shape of Y_test : ",y_test.shape)

Shape of X_train : (242, 13)

Shape of Y_train : (242,)

Shape of X_test : (61, 13)

Shape of Y_test : (61,)

In [11]: x_train,x_test,y_train,y_test=x_train.values,x_test.values,y_train.values,y_test.value

CREATING LINEAR RANDOM FOREST MODEL

In [12]: from sklearn.ensemble import RandomForestClassifier
classifier= RandomForestClassifier(n_estimators= 10, criterion="entropy")
classifier.fit(x_train, y_train)

Out[12]: RandomForestClassifier(criterion='entropy', n_estimators=10)

PREDICTION

```
[13]: classifier.score(x_test,y_test)
```

```
[13]: 0.8360655737704918
```

Let check for data prediction from a datapoint in x_test

```
[14]: x_test[6]
```

```
[14]: array([ 44.,  1.,  1., 120., 220.,  0.,  1., 170.,  0.,  0.,  2.,  
         0.,  2.])
```

Actual output of data point

```
[15]: if y_test[6]==0:  
      print("Person predicted to have LOW chance of heart disease!")  
      else:  
      print("Person predicted to have HIGHER chance of heart disease!")
```

Person predicted to have HIGHER chance of heart disease!

Prediction from Random Forest Classifier

```
[16]: result=classifier.predict([x_test[6]])  
      if result==0:  
      print("Person predicted to have LOW chance of heart disease!")  
      else:  
      print("Person predicted to have HIGHER chance of heart disease!")
```

Person predicted to have HIGHER chance of heart disease!

