**RANDOM FOREST CLASSIFIER**

1. **Import libraries :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline//instruct plots to show directly below the code cell

**load the data set and visualising :**

df=pd.read\_csv('/content/heart\_v2 (1).csv')//load the csv

print(df.head())//view csv

sns.countplot(df['heart disease'])//count plot

plt.title('Value count of heart disease patients')//adding title

plt.show()//show the plot

1. **Droping the existing and storing in new variable(SPLITTING INPUT AND OUTPUT SEPARATELY):**

x=df.drop('heart disease',axis=1)//droping column heart disease

y=df['heart disease']//storing in new

1. **Model selection and spliting traing and testing data:**

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.7,random\_state=42)//splitting the data as test data and train data

x\_train.shape,x\_test.shape//visualising the dimension

1. **Model building with training and testing time:**

import time

from sklearn.ensemble import RandomForestClassifier

classifier\_rf=RandomForestClassifier(random\_state=42,n\_jobs=-1,max\_depth=5,n\_estimators=100,oob\_score=True)//model initialization

start\_time=time.time()//training and timing

classifier\_rf.fit(x\_train,y\_train)//training ad timing

end\_time=time.time()//training and timing

print(f"Training completed in {end\_time - start\_time:.4f} seconds")

//output training time

1. **Classification and model evaluation:**

from sklearn.metrics import classification\_report,confusion\_matrix

y\_pred=classifier\_rf.predict(x\_test) //make predictions

print(confusion\_matrix(y\_test,y\_pred)) //confusion matrix creation

print(classification\_report(y\_test,y\_pred)) //classification report

1. **Feature importance and visualization:**

import matplotlib.pyplot as plt

importances=classifier\_rf.feature\_importances\_ //feature importance

plt.barh(range(len(importances)),importances) //bar plot

plt.xlabel('Feature Importance')//adding labels

plt.ylabel('Feature Index')//adding labels

plt.title('Random Forest feature importnace')//adding title

plt.show()