HEART DISEASE PREDICTION

Source code

### 1. Importing the Libraries

import pandas as pd

### 2. Importing the Dataset

data = pd.read\_csv('heart.csv')

### 3. Taking Care of Missing Values

data.isnull().sum()

### 4. Taking Care of Duplicate Values

data\_dup = data.duplicated().any()

data\_dup

data = data.drop\_duplicates()

data\_dup = data.duplicated().any()

data\_dup

### 5. Data Processing

cate\_val = []

cont\_val = []

for column in data.columns:

if data[column].nunique() <=10:

cate\_val.append(column)

else:

cont\_val.append(column)

cate\_val

cont\_val

### 6. Encoding Categorical Data

cate\_val

data['cp'].unique()

cate\_val.remove('sex')

cate\_val.remove('target')

data = pd.get\_dummies(data,columns = cate\_val,drop\_first=True)

data.head()

### 7. Feature Scaling

data.head()

from sklearn.preprocessing import StandardScaler

st = StandardScaler()

data[cont\_val] = st.fit\_transform(data[cont\_val])

data.head()

### 8. Splitting The Dataset Into The Training Set And Test Set

X = data.drop('target',axis=1)

y = data['target']

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=42)

y\_test

### 9. Logistic Regression

data.head()

from sklearn.linear\_model import LogisticRegression

log = LogisticRegression()

log.fit(X\_train,y\_train)

y\_pred1 = log.predict(X\_test)

from sklearn.metrics import accuracy\_score

accuracy\_score(y\_test,y\_pred1)

### 10. SVC

from sklearn import svm

svm = svm.SVC()

svm.fit(X\_train,y\_train)

y\_pred2 = svm.predict(X\_test)

accuracy\_score(y\_test,y\_pred2)

### 11. KNeighbors Classifier

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()

knn.fit(X\_train,y\_train)

y\_pred3=knn.predict(X\_test)

accuracy\_score(y\_test,y\_pred3)

score = []

for k in range(1,40):

knn=KNeighborsClassifier(n\_neighbors=k)

knn.fit(X\_train,y\_train)

y\_pred=knn.predict(X\_test)

score.append(accuracy\_score(y\_test,y\_pred))

score

import matplotlib.pyplot as plt

plt.plot(score)

plt.xlabel("K Value")

plt.ylabel("Acc")

plt.show()

knn=KNeighborsClassifier(n\_neighbors=2)

knn.fit(X\_train,y\_train)

y\_pred=knn.predict(X\_test)

accuracy\_score(y\_test,y\_pred)

### Non-Linear ML Algorithms

data = pd.read\_csv('heart.csv')

data = data.drop\_duplicates()

X = data.drop('target',axis=1)

y=data['target']

X\_train,X\_test,y\_train,y\_test= train\_test\_split(X,y,test\_size=0.2,

random\_state=42)

### 12. Decision Tree Classifier

from sklearn.tree import DecisionTreeClassifier

dt = DecisionTreeClassifier()

dt.fit(X\_train,y\_train)

y\_pred4= dt.predict(X\_test)

accuracy\_score(y\_test,y\_pred4)

### 13. Random Forest Classifier

from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()

rf.fit(X\_train,y\_train)

y\_pred5= rf.predict(X\_test)

accuracy\_score(y\_test,y\_pred5)

### 14. Gradient Boosting Classifier

from sklearn.ensemble import GradientBoostingClassifier

gbc = GradientBoostingClassifier()

gbc.fit(X\_train,y\_train)

y\_pred6 = gbc.predict(X\_test)

accuracy\_score(y\_test,y\_pred6)

final\_data = pd.DataFrame({'Models':['LR','SVM','KNN','DT','RF','GB'],

'ACC':[accuracy\_score(y\_test,y\_pred1)\*100,

accuracy\_score(y\_test,y\_pred2)\*100,

accuracy\_score(y\_test,y\_pred3)\*100,

accuracy\_score(y\_test,y\_pred4)\*100,

accuracy\_score(y\_test,y\_pred5)\*100,

accuracy\_score(y\_test,y\_pred6)\*100]})

final\_data

import seaborn as sns

sns.barplot(final\_data['Models'],final\_data['ACC'])

X=data.drop('target',axis=1)

y=data['target']

from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()

rf.fit(X,y)

### 15. Prediction on New Data

import pandas as pd

new\_data = pd.DataFrame({

'age':52,

'sex':1,

'cp':0,

'trestbps':125,

'chol':212,

'fbs':0,

'restecg':1,

'thalach':168,

'exang':0,

'oldpeak':1.0,

'slope':2,

'ca':2,

'thal':3,

},index=[0])

new\_data

p = rf.predict(new\_data)

if p[0]==0:

print("No Disease")

else:

print("Disease")

### 16. Save Model Using Joblib

import joblib

joblib.dump(rf,'model\_joblib\_heart')

model = joblib.load('model\_joblib\_heart')

model.predict(new\_data)

data.tail()

### GUI

from tkinter import \*

import joblib

def show\_entry\_fields():

p1=int(e1.get())

p2=int(e2.get())

p3=int(e3.get())

p4=int(e4.get())

p5=int(e5.get())

p6=int(e6.get())

p7=int(e7.get())

p8=int(e8.get())

p9=int(e9.get())

p10=float(e10.get())

p11=int(e11.get())

p12=int(e12.get())

p13=int(e13.get())

model = joblib.load('model\_joblib\_heart')

result=model.predict([[p1,p2,p3,p4,p5,p6,p7,p8,p8,p10,p11,p12,p13]])

if result == 0:

Label(master, text="No Heart Disease").grid(row=31)

else:

Label(master, text="Possibility of Heart Disease").grid(row=31)

master = Tk()

master.title("Heart Disease Prediction System")

label = Label(master, text = "Heart Disease Prediction System"

, bg = "black", fg = "white"). \

grid(row=0,columnspan=2)

Label(master, text="Enter Your Age").grid(row=1)

Label(master, text="Male Or Female [1/0]").grid(row=2)

Label(master, text="Enter Value of CP").grid(row=3)

Label(master, text="Enter Value of trestbps").grid(row=4)

Label(master, text="Enter Value of chol").grid(row=5)

Label(master, text="Enter Value of fbs").grid(row=6)

Label(master, text="Enter Value of restecg").grid(row=7)

Label(master, text="Enter Value of thalach").grid(row=8)

Label(master, text="Enter Value of exang").grid(row=9)

Label(master, text="Enter Value of oldpeak").grid(row=10)

Label(master, text="Enter Value of slope").grid(row=11)

Label(master, text="Enter Value of ca").grid(row=12)

Label(master, text="Enter Value of thal").grid(row=13)

e1 = Entry(master)

e2 = Entry(master)

e3 = Entry(master)

e4 = Entry(master)

e5 = Entry(master)

e6 = Entry(master)

e7 = Entry(master)

e8 = Entry(master)

e9 = Entry(master)

e10 = Entry(master)

e11 = Entry(master)

e12 = Entry(master)

e13 = Entry(master)

e1.grid(row=1, column=1)

e2.grid(row=2, column=1)

e3.grid(row=3, column=1)

e4.grid(row=4, column=1)

e5.grid(row=5, column=1)

e6.grid(row=6, column=1)

e7.grid(row=7, column=1)

e8.grid(row=8, column=1)

e9.grid(row=9, column=1)

e10.grid(row=10, column=1)

e11.grid(row=11, column=1)

e12.grid(row=12, column=1)

e13.grid(row=13, column=1)

Button(master, text='Predict', command=show\_entry\_fields).grid()

mainloop()



