import tensorflow as tf

import numpy as np

import matplotlib.pyplot as plt

import itertools

import math

import pandas as pd

import os

import seaborn as sns

#We will try to import our dataset

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

d.head()

d.isnull()

d.drop\_duplicates()

col\_val=[]

col\_num=[]

for column in data.columns:

if data[column].nunique()<=20:

col\_val.append(column)

else:

col\_num.append(column)

col\_val

col\_num

d.describe()

for i, col in enumerate(col\_val, 1):

plt.subplot(1,9,i)

d[d.target == 1][col].hist(bins=20, color='red', alpha=0.5, label='Disease: YES')

d[d.target == 0][col].hist(bins=20, color='blue', alpha=0.5, label='Disease: NO')

plt.xlabel(col)

plt.legend()

d.info()

#Forming Correlation Matrix between the different methods

plt.figure(figsize=(24,6))

sns.heatmap(d.corr(),annot=True)

#To Find gender distribution whether the person is male or female according to the Target variable

sns.countplot(x='sex',hue="target",data=data)

plt.xticks([1,0],['Male','Female'])

plt.legend(labels = ['NOT-DETECTED','DETECTED'])

plt.show()

sns.set\_theme(style='darkgrid')

import numpy as np

from sklearn.preprocessing import StandardScaler

import numpy as np

from sklearn.preprocessing import StandardScaler

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

X1 = data.drop('target', axis=1)

Y1 = data['target']

X1\_\_train, X1\_\_test, y\_\_train, y\_\_test=train\_test\_split(X1,y,test\_size=0.01, random\_state=57)

y\_\_train

d.head()

from sklearn.linear\_model import LogisticRegression

log\_reg = LogisticRegression()

log\_reg.fit(X1\_\_train, y\_\_train)

X\_Trainpredict=log\_reg.predict(X1\_\_train)

from sklearn.metrics import accuracy\_score

Train\_DataAccuracy= accuracy\_score(X\_Trainpredict,y\_\_train)

print('The accuracy of the training data = ',Train\_DataAccuracy)

X\_Testpredict=log\_reg.predict(X1\_\_test)

Test\_DataAccuracy= accuracy\_score(X\_Testpredict,y\_\_test)

print('The accuracy on the data of testing model = ',Test\_DataAccuracy)

#Forming a PREDICTIVE SYSTEM

input\_d= (41,0,130,204,0,0,172,0,1,4,2,0,2)

input\_d\_as\_numpy\_array=np.asarray(input\_d)

input\_d\_reshaped= input\_d\_as\_numpy\_array.reshape(1,-1)

d\_prediction=log\_reg.predict(input\_d\_reshaped)

print(d\_prediction)

if (d\_prediction[0]==0):

print('No Heart Disease has been detected')

else:

print('Heart Disease has been detected')

from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()

rf.fit(X1\_\_train,y\_\_train)

rf.fit(X1,y)

import joblib

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

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

model.predict(input\_d\_reshaped)

from tkinter import \*

import joblib

import numpy as np

from sklearn import \*

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()