**Advancing Heart Disease Prediction: Neural Network Models for Accurate Diagnosis**

**Code:**

import sys

import pandas as pd

import numpy as np

import sklearn

import matplotlib

import keras

import matplotlib.pyplot as plt

from pandas.plotting import scatter\_matrix

IMPORTING DATASET:

# import the heart disease dataset

url = "http://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data"

names = ['age',

'sex',

'cp',

'trestbps',

'chol',

'fbs',

'restecg',

'thalach',

'exang',

'oldpeak',

'slope',

'ca',

'thal',

'class']

cleveland = pd.read\_csv(url, names=names)

print ('format(cleveland.shape')

print (cleveland.loc[1])

cleveland.loc[280:]

# remove missing data (indicated with a "?")

data = cleveland[~cleveland.isin(['?'])]

data.loc[280:]

data = data.dropna(axis=0)

data.loc[280:]

print (data.shape)

print (data.dtypes)

data = data.apply(pd.to\_numeric)

data.dtypes

data.describe()

data.hist(figsize = (12, 12))

plt.show()

TRAINING AND TESTING DATASETS:

from sklearn import model\_selection

X = np.array(data.drop(columns=['class']))

y = np.array(data['class'])

X\_train, X\_test, y\_train, y\_test = model\_selection.train\_test\_split(X, y, test\_size = 0.2)

from keras.utils import to\_categorical

Y\_train = to\_categorical(y\_train, num\_classes=None)

Y\_test = to\_categorical(y\_test, num\_classes=None)

print (Y\_train.shape)

print (Y\_train[:10])

MODEL TRAINING:

from keras.models import Sequential

from keras.layers import Dense

from keras.optimizers import Adam

# define a function to build the keras model

def create\_model():

model = Sequential()

model.add(Dense(8, input\_dim=13, kernel\_initializer='normal', activation='relu'))

model.add(Dense(4, kernel\_initializer='normal', activation='relu'))

model.add(Dense(5, activation='softmax'))

adam = Adam(learning\_rate=0.001)

model.compile(loss='categorical\_crossentropy', optimizer=adam, metrics=['accuracy'])

return model

model = create\_model()

print(model.summary())

# fit the model to the training data

model.fit(X\_train, Y\_train, epochs=100, batch\_size=10, verbose = 1)

BINARY CLASSIFICATION MODEL:

# convert into binary classification problem - heart disease or no heart disease

Y\_train\_binary = y\_train.copy()

Y\_test\_binary = y\_test.copy()

print("Binary Classification Model")

Y\_train\_binary[Y\_train\_binary > 0] = 1

Y\_test\_binary[Y\_test\_binary > 0] = 1

print (Y\_train\_binary[::])

# define a new keras model for binary classification

def create\_binary\_model():

model = Sequential()

model.add(Dense(8, input\_dim=13, kernel\_initializer='normal', activation='relu'))

model.add(Dense(4, kernel\_initializer='normal', activation='relu'))

model.add(Dense(1, activation='sigmoid'))

# Compile model

adam = Adam(learning\_rate=0.0001)

model.compile(loss='binary\_crossentropy', optimizer=adam, metrics=['accuracy'])

return model

binary\_model = create\_binary\_model()

print(binary\_model.summary())

# fit the binary model on the training data

binary\_model.fit(X\_train, Y\_train\_binary, epochs=100, batch\_size=10, verbose = 1)

EVALUATION:

# generate classification report using predictions for categorical model

from sklearn.metrics import classification\_report, accuracy\_score

categorical\_pred = np.argmax(model.predict(X\_test), axis=1)

print('Results for Categorical Model')

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

print(classification\_report(y\_test, categorical\_pred))

# generate classification report using predictions for binary model

binary\_pred = np.round(binary\_model.predict(X\_test)).astype(int)

print('Results for Binary Model')

print(accuracy\_score(Y\_test\_binary, binary\_pred))

print(classification\_report(Y\_test\_binary, binary\_pred))