import numpy as np

import pandas as pd

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

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import mean\_squared\_error

from gmdhpy import gmdh

# Load data from a CSV file

data = pd.read\_csv(r'C:\Users\Elena\Documents\GitHub\steel\_strength\metals\_data.csv')

# Assume the first 13 columns are features and the last three are targets

features = data.iloc[:, :13]

outputs = data.iloc[:, -3:]

# Scale features

scaler = StandardScaler()

features\_scaled = scaler.fit\_transform(features)

# Split the scaled features and outputs into training and testing datasets

X\_train, X\_test, y\_train\_all, y\_test\_all = train\_test\_split(features\_scaled, outputs, test\_size=0.2, random\_state=42)

# Prepare to store models and mse for each target

models = []

mses = []

# Train a separate model for each output column

for i in range(outputs.shape[1]):

# Select the specific target column for this loop

y\_train = y\_train\_all.iloc[:, i]

y\_test = y\_test\_all.iloc[:, i]

# Initialize and train the GMDH model

model = gmdh.MultilayerGMDH()

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

# Store the model

models.append(model)

# Make predictions with the trained model

predictions = model.predict(X\_test)

# Evaluate the model's performance using Mean Squared Error (MSE)

mse = mean\_squared\_error(y\_test, predictions)

mses.append(mse)

# Print MSE for each target

print(f'MSE for target {i}:', mse)

# Optionally, print the MSEs

print("MSEs:", mses)

(steel\_strength) C:\Users\Elena\Documents\GitHub\steel\_strength>C:/ProgramData/Anaconda3/envs/steel\_strength/python.exe c:/Users/Elena/Documents/GitHub/steel\_strength/GMDH.py

train layer0 in 0.07 sec

train layer1 in 0.21 sec

train layer2 in 0.20 sec

train layer3 in 0.18 sec

train layer4 in 0.21 sec

train layer5 in 0.21 sec

train layer6 in 0.21 sec

train layer7 in 0.20 sec

train layer8 in 0.22 sec

train layer9 in 0.18 sec

train layer10 in 0.22 sec

train layer11 in 0.22 sec

train layer12 in 0.22 sec

train layer13 in 0.22 sec

train layer14 in 0.20 sec

train layer15 in 0.20 sec

train layer16 in 0.20 sec

train layer17 in 0.21 sec

train layer18 in 0.19 sec

train layer19 in 0.21 sec

train layer20 in 0.19 sec

train layer21 in 0.20 sec

train layer22 in 0.20 sec

train layer23 in 0.21 sec

train layer24 in 0.19 sec

train layer25 in 0.20 sec

train layer26 in 0.20 sec

train layer27 in 0.20 sec

train layer28 in 0.22 sec

train layer29 in 0.20 sec

train layer30 in 0.19 sec

train layer31 in 0.21 sec

train layer32 in 0.20 sec

train layer33 in 0.22 sec

train layer34 in 0.20 sec

train layer35 in 0.24 sec

train layer36 in 0.20 sec

train layer37 in 0.21 sec

train layer38 in 0.21 sec

train layer39 in 0.20 sec

train layer40 in 0.21 sec

train layer41 in 0.19 sec

train layer42 in 0.20 sec

train layer43 in 0.21 sec

train layer44 in 0.19 sec

train layer45 in 0.22 sec

train layer46 in 0.20 sec

train layer47 in 0.19 sec

train layer48 in 0.20 sec

train layer49 in 0.21 sec

MSE for target 0: 2.46397767903223e-05

train layer0 in 0.05 sec

train layer1 in 0.20 sec

train layer2 in 0.21 sec

train layer3 in 0.21 sec

train layer4 in 0.19 sec

train layer5 in 0.20 sec

train layer6 in 0.21 sec

train layer7 in 0.20 sec

train layer8 in 0.22 sec

train layer9 in 0.20 sec

train layer10 in 0.21 sec

train layer11 in 0.22 sec

train layer12 in 0.20 sec

train layer13 in 0.20 sec

train layer14 in 0.22 sec

train layer15 in 0.21 sec

train layer16 in 0.20 sec

train layer17 in 0.21 sec

train layer18 in 0.20 sec

train layer19 in 0.23 sec

train layer20 in 0.20 sec

train layer21 in 0.21 sec

train layer22 in 0.20 sec

train layer23 in 0.21 sec

train layer24 in 0.20 sec

train layer25 in 0.23 sec

train layer26 in 0.21 sec

train layer27 in 0.20 sec

train layer28 in 0.21 sec

train layer29 in 0.20 sec

train layer30 in 0.22 sec

train layer31 in 0.20 sec

train layer32 in 0.22 sec

train layer33 in 0.22 sec

train layer34 in 0.23 sec

train layer35 in 0.20 sec

train layer36 in 0.21 sec

train layer37 in 0.22 sec

train layer38 in 0.24 sec

train layer39 in 0.26 sec

train layer40 in 0.21 sec

train layer41 in 0.23 sec

train layer42 in 0.21 sec

train layer43 in 0.22 sec

train layer44 in 0.21 sec

train layer45 in 0.22 sec

train layer46 in 0.20 sec

train layer47 in 0.21 sec

train layer48 in 0.22 sec

train layer49 in 0.21 sec

MSE for target 1: 3.121281254310028e-06

train layer0 in 0.04 sec

train layer1 in 0.20 sec

train layer2 in 0.21 sec

train layer3 in 0.22 sec

train layer4 in 0.24 sec

train layer5 in 0.20 sec

train layer6 in 0.22 sec

train layer7 in 0.19 sec

train layer8 in 0.22 sec

train layer9 in 0.21 sec

train layer10 in 0.20 sec

train layer11 in 0.21 sec

train layer12 in 0.21 sec

train layer13 in 0.21 sec

train layer14 in 0.21 sec

train layer15 in 0.20 sec

train layer16 in 0.21 sec

train layer17 in 0.22 sec

train layer18 in 0.22 sec

train layer19 in 0.20 sec

train layer20 in 0.20 sec

train layer21 in 0.21 sec

train layer22 in 0.20 sec

train layer23 in 0.22 sec

train layer24 in 0.21 sec

train layer25 in 0.20 sec

train layer26 in 0.21 sec

train layer27 in 0.22 sec

train layer28 in 0.21 sec

train layer29 in 0.21 sec

train layer30 in 0.20 sec

train layer31 in 0.22 sec

train layer32 in 0.22 sec

train layer33 in 0.21 sec

train layer34 in 0.20 sec

train layer35 in 0.21 sec

train layer36 in 0.20 sec

train layer37 in 0.22 sec

train layer38 in 0.20 sec

train layer39 in 0.22 sec

train layer40 in 0.21 sec

train layer41 in 0.20 sec

train layer42 in 0.22 sec

train layer43 in 0.20 sec

train layer44 in 0.20 sec

train layer45 in 0.20 sec

train layer46 in 0.22 sec

train layer47 in 0.20 sec

train layer48 in 0.22 sec

train layer49 in 0.22 sec

MSE for target 2: 3.561976679748538e-06

MSEs: [2.46397767903223e-05, 3.121281254310028e-06, 3.561976679748538e-06]