import pandas as pd

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

from sklearn.ensemble import GradientBoostingRegressor

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score

import matplotlib.pyplot as plt

# Load the dataset

data = pd.read\_csv(" ", encoding='gbk')

# Remove rows with missing values

data = data.dropna()

# Features and target variable

X = data[['concentration', 'temperature']] # Features: Concentration and Temperature

y = data['response'] # Target variable: Response

# Split the data into training and testing sets

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

# Train the Gradient Boosting Regression model

gb\_model = GradientBoostingRegressor(n\_estimators=100, learning\_rate=0.1, max\_depth=3, random\_state=42)

gb\_model.fit(X\_train, y\_train)

# Make predictions

gb\_y\_pred = gb\_model.predict(X\_test)

# Define multiple sets of data to predict

new\_data = pd.DataFrame({

'concentration': [10, 10, 10, 10, 10, 10, 10, 30, 30, 30, 30, 30, 30, 30, 50, 50, 50, 50, 50, 50, 50], # Concentration values for multiple data points

'temperature': [25, 50, 100, 150, 200, 250, 300, 25, 50, 100, 150, 200, 250, 300, 25, 50, 100, 150, 200, 250, 300] # Temperature values for multiple data points

})

# Use the trained Gradient Boosting Regression model to make predictions for the new data

predicted\_responses = gb\_model.predict(new\_data)

# Print the predicted responses for each set of data

for i, (conc, temp) in enumerate(zip(new\_data['concentration'], new\_data['temperature'])):

print(f"Predicted Response for concentration {conc} and temperature {temp}: {predicted\_responses[i]}")