```
clear all;
clc

% Define file paths
input_file = "C:\Users\Hp\OneDrive\Desktop\Cleaned_OUT_Table.csv";
output_file = "C:\Users\Hp\OneDrive\Desktop\Cleaned_IN_Table.csv";

% Read input and output data
data1 = readmatrix(input_file);
data2 = readmatrix(output_file);
data = [data1, data2];
```

Extracting data

```
input = 5;  % Number of input variables
output = 2; % Number of output variables

input_matrix = data(:, 1:input); % Matrix containing input data
output_matrix = data(:, 6:7);    % Matrix containing output data

% Splitting data into training (70%) and validation (30%) sets
train_index = randperm(length(output_matrix), round(0.7 * length(output_matrix)));
% Selecting 70% random data
valid_index = setdiff(1:length(output_matrix), train_index); % Remaining 30% data

% Extracting train and validation data
train_input = input_matrix(train_index, :);
valid_input = input_matrix(valid_index, :);
train_output = output_matrix(train_index, :);
valid_output = output_matrix(valid_index, :);

% Combining train data for easy indexing
train_data = [train_input, train_output];
valid_data = [valid_input, valid_output];

% Number of nearest neighbors for JIT model
k = 60;

% Initialize prediction matrix
y_predict_lin = zeros(length(valid_output), output);

% Loop over each validation sample
for i = 1:length(valid_output)
    querypt = valid_input(i, 1:input); % Extracting each row of input validation
data

    % Find k nearest neighbors
    index_neighbour = knnsearch(train_input, querypt, 'dist','euclidean', 'K', k,
'IncludeTies', false);
```

```
    % Extract neighbor input and output data
    neighbour_input = train_data(index_neighbour, 1:input);
    neighbour_output = train_data(index_neighbour, 6:7);

    % Ensure numeric values and remove missing data
    neighbour_input = rmmissing(neighbour_input);
    neighbour_output = rmmissing(neighbour_output);

    if isempty(neighbour_input) || isempty(neighbour_output)
        y_predict_lin(i, :) = NaN; % Handle case where all neighbors had missing
data
        continue;
    end

    % Convert to double for fitlm compatibility
    neighbour_input = double(neighbour_input);
    neighbour_output = double(neighbour_output);

    % Fit separate linear models for each output variable
    model1 = fitlm(neighbour_input, neighbour_output(:, 1), "linear");
    model2 = fitlm(neighbour_input, neighbour_output(:, 2), "linear");

    % Predict output using both models
    y_predict_lin(i, 1) = predict(model1, querypt);
    y_predict_lin(i, 2) = predict(model2, querypt);
end

% Compute mean squared error
mean_error = mean((valid_output - y_predict_lin).^2, 'omitnan');
disp(["The mean error for trained model is:", mean_error]);
```

```
    "The mean error for trained model is:"    "0.00078682"    "0.0050899"
```

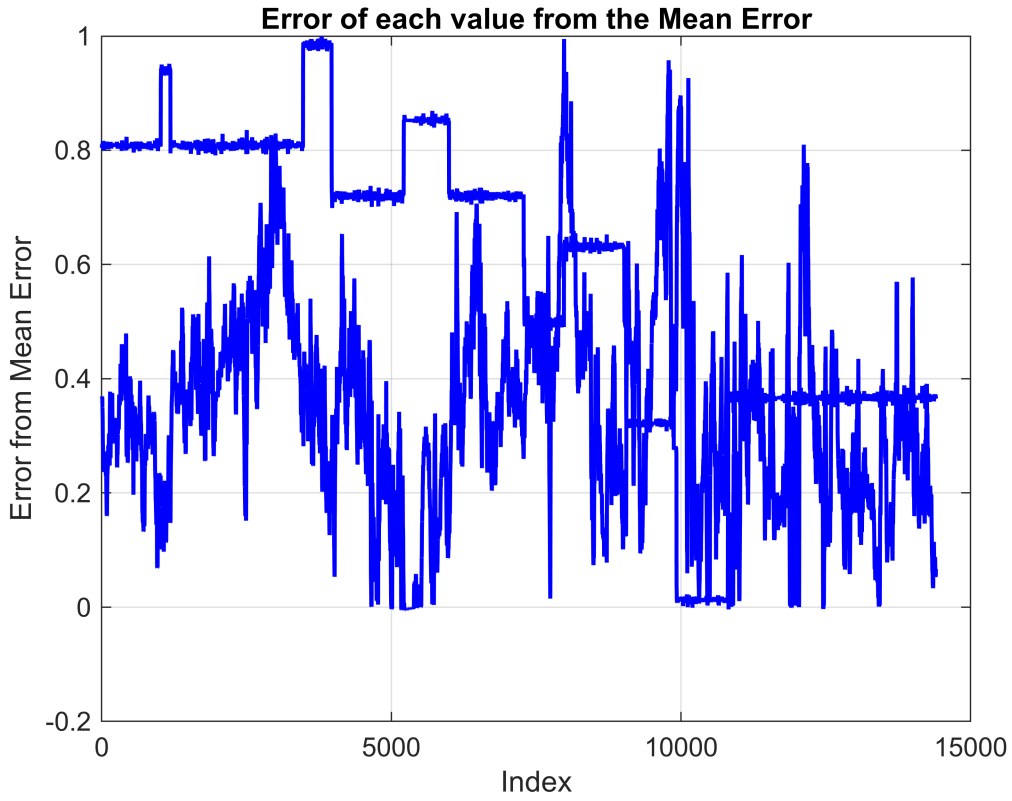
```
% Compute R-squared value
residual_sum_squares = sum((valid_output - y_predict_lin).^2, 'omitnan');
total_sum_squares = sum((valid_output - mean(valid_output, 'omitnan')).^2,
'omitnan');
R_squared = 1 - (residual_sum_squares ./ total_sum_squares);
disp(["The R-squared for trained model is:", R_squared]);
```

```
    "The R-squared for trained model is:"    "0.98731"    "0.8439"
```

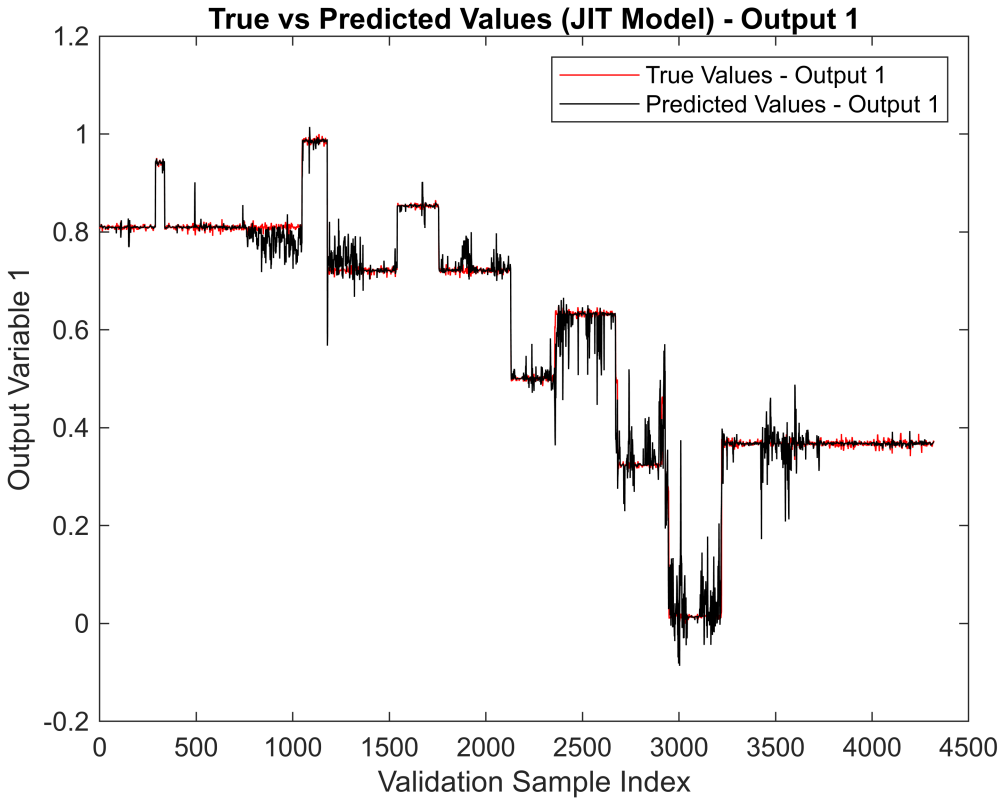
```
% Compute error for each sample
error = output_matrix - mean_error;

% Plot the errors
figure;
plot(1:length(output_matrix), error, 'b', 'LineWidth', 1.5);
xlabel('Index');
```

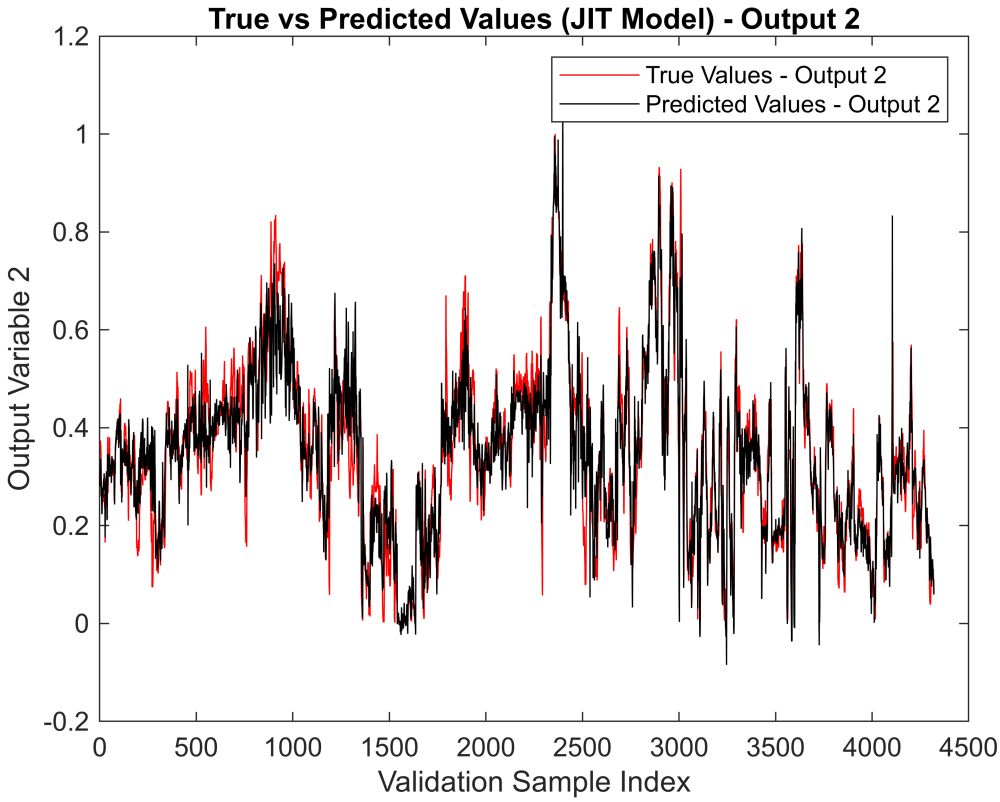
```
ylabel('Error from Mean Error');
title('Error of each value from the Mean Error');
grid on;
```




```
% Plot true vs predicted values
figure;
plot(1:length(valid_output), valid_output(:, 1), 'r', 'DisplayName', 'True Values - Output 1');
hold on;
plot(1:length(valid_output), y_predict_lin(:, 1), 'k', 'DisplayName', 'Predicted Values - Output 1');
xlabel('Validation Sample Index');
ylabel('Output Variable 1');
legend;
title('True vs Predicted Values (JIT Model) - Output 1');
hold off;
```




```
figure;
plot(1:length(valid_output), valid_output(:, 2), 'r', 'DisplayName', 'True Values -
Output 2');
hold on;
plot(1:length(valid_output), y_predict_lin(:, 2), 'k', 'DisplayName', 'Predicted
Values - Output 2');
xlabel('Validation Sample Index');
ylabel('Output Variable 2');
legend;
title('True vs Predicted Values (JIT Model) - Output 2');
hold off;
```


True vs Predicted Values (JIT Model) - Output 2
True Values - Output 2
Predicted Values - Output 2
Output Variable 2
Validation Sample Index
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
500
1000
1500
2000
2500
3000
3500
4000
4500