% Load the dataset

data = readtable('Iran ready mix2.csv');

% Extract input parameters and output variables

X = data(:, {'Coarse3\_4', 'Coarse3\_8', 'Fine', 'Cement', 'CementType', 'Admixture1', 'TypeOfAdmixture1', 'Admixture2', 'TypeOfAdmixture2', 'Water', 'ConcreteTemperature', 'AmbientTemperature', 'Slump', 'WaterToCementRatio', 'AirContent'});

Y = data(:, {'CompressiveStrength', 'Cost', 'CO2Emissions'});

% Ask the user for the required classification

availableClassifications = unique(data.Classification);

fprintf('Available Classifications:\n');

for i = 1:numel(availableClassifications)

fprintf('%d. %s\n', i, availableClassifications{i});

end

classificationChoice = input('Enter the number corresponding to the required classification: ');

% Filter the dataset based on the user's classification choice

classifiedData = data(strcmp(data.Classification, availableClassifications{classificationChoice}), :);

if strcmp(availableClassifications{classificationChoice}, 'Non-structural Concrete')

availableApplications = {'precast wall', 'channel', 'culvert'};

else

availableApplications = unique(classifiedData.Application);

end

fprintf('Available Applications:\n');

for i = 1:numel(availableApplications)

fprintf('%d. %s\n', i, availableApplications{i});

end

applicationChoice = input('Enter the number corresponding to the required Application: ');

% Filter the dataset based on the user's application choice

selectedData = classifiedData(strcmp(classifiedData.Application, availableApplications{applicationChoice}), :);

if isempty(selectedData)

fprintf('No mix design found for the specified classification and application.\n');

else

% Extract input parameters and output variables for the selected data

X = selectedData(:, {'Coarse3\_4', 'Coarse3\_8', 'Fine', 'Cement', 'CementType', 'Admixture1', 'TypeOfAdmixture1', 'Admixture2', 'TypeOfAdmixture2', 'Water', 'ConcreteTemperature', 'AmbientTemperature', 'Slump', 'WaterToCementRatio', 'AirContent'});

Y = selectedData(:, {'CompressiveStrength', 'Cost', 'CO2Emissions'});

% Ask the user for required Strength, slump, and ambient temperature

requiredStrength = input('Enter the required Strength (MPa): ');

requiredSlump = input('Enter the required slump (cm): ');

requiredAmbientTemp = input('Enter the required ambient temperature (°C): ');

% Define the maximum allowable deviation in ambient temperature

maxAmbientTempDeviation = 3; % Maximum allowable deviation from the required ambient temperature

% Define the maximum allowable deviation in slump

maxSlumpDeviation = 1; % Maximum allowable deviation from the required slump

% Define the maximum allowable variance in compressive strength

maxStrengthVariance = 2; % Maximum allowable variance from the required compressive strength

% Find the indices of samples that meet the required Strength, slump, and ambient temperature

indices = find(X.Slump <= requiredSlump + maxSlumpDeviation & ...

X.Slump >= requiredSlump - maxSlumpDeviation & ...

(X.AmbientTemperature <= requiredAmbientTemp + maxAmbientTempDeviation & X.AmbientTemperature >= requiredAmbientTemp - maxAmbientTempDeviation) & ...

Y.CompressiveStrength >= (requiredStrength - maxStrengthVariance));

if ~isempty(indices)

% Sort the indices based on ascending cement content

[~, sortedIndices] = sort(X.Cement(indices), 'ascend'); % Sorting based on cement content

% Get the sorted indices

sortedIndices = indices(sortedIndices);

% Generate options for lowest cost, lowest CO2, and highest compressive strength

optionInputs = cell(3, 1);

optionObjectives = cell(3, 1);

% Find indices for lowest cost, lowest CO2, and highest compressive strength

costIndices = find(X.Slump <= requiredSlump + maxSlumpDeviation & ...

X.Slump >= requiredSlump - maxSlumpDeviation & ...

(X.AmbientTemperature <= requiredAmbientTemp + maxAmbientTempDeviation & X.AmbientTemperature >= requiredAmbientTemp - maxAmbientTempDeviation) & ...

Y.CompressiveStrength >= (requiredStrength - maxStrengthVariance));

[~, costIdx] = min(Y.Cost(costIndices));

costIndices = costIndices(costIdx);

co2Indices = find(X.Slump <= requiredSlump + maxSlumpDeviation & ...

X.Slump >= requiredSlump - maxSlumpDeviation & ...

(X.AmbientTemperature <= requiredAmbientTemp + maxAmbientTempDeviation & X.AmbientTemperature >= requiredAmbientTemp - maxAmbientTempDeviation) & ...

Y.CompressiveStrength >= (requiredStrength - maxStrengthVariance));

[~, co2Idx] = min(Y.CO2Emissions(co2Indices));

co2Indices = co2Indices(co2Idx);

strengthIndices = find(X.Slump <= requiredSlump + maxSlumpDeviation & ...

X.Slump >= requiredSlump - maxSlumpDeviation & ...

(X.AmbientTemperature <= requiredAmbientTemp + maxAmbientTempDeviation & X.AmbientTemperature >= requiredAmbientTemp - maxAmbientTempDeviation) & ...

Y.CompressiveStrength >= (requiredStrength - maxStrengthVariance));

[~, strengthIdx] = max(Y.CompressiveStrength(strengthIndices));

strengthIndices = strengthIndices(strengthIdx);

% Retrieve the corresponding input parameters and objective values for lowest cost, lowest CO2, and highest compressive strength

optionInputs{1} = X(costIndices, :);

optionObjectives{1} = Y(costIndices, :);

optionInputs{2} = X(co2Indices, :);

optionObjectives{2} = Y(co2Indices, :);

optionInputs{3} = X(strengthIndices, :);

optionObjectives{3} = Y(strengthIndices, :);

% Plot the trade-offs between Compressive Strength, CO2 Emissions, and Cost for the options

figure;

scatter3(optionObjectives{1}.CompressiveStrength, optionObjectives{1}.Cost, optionObjectives{1}.CO2Emissions, 100, 'r', 'filled');

hold on;

for i = 2:3

scatter3(optionObjectives{i}.CompressiveStrength, optionObjectives{i}.Cost, optionObjectives{i}.CO2Emissions, 'filled');

end

% Set the axis labels

xlabel('Compressive Strength (MPa)');

ylabel('Cost (US $/m3)');

zlabel('CO2 Emissions (kg/m3)');

% Create the legend labels

legendLabels = {

'Option 1 (Lowest Cost)', ...

'Option 2 (Lowest CO2)', ...

'Option 3 (Highest Strength)'

};

% Display the legend

legend(legendLabels);

% Print the user-defined strength, slump, and ambient temperature

fprintf('User-defined Strength (MPa): %.2f\n', requiredStrength);

fprintf('User-defined Slump (cm): %.2f\n', requiredSlump);

fprintf('User-defined Ambient Temperature (°C): %.2f\n', requiredAmbientTemp);

fprintf('\n');

for i = 1:3

fprintf('Option %d (%s):\n', i, legendLabels{i});

fprintf('Mix Design Information:\n');

disp(optionInputs{i});

fprintf('Objective Values:\n');

fprintf('Compressive Strength: %.2f MPa\n', optionObjectives{i}.CompressiveStrength);

fprintf('Cost: %.2f US $/m3\n', optionObjectives{i}.Cost);

fprintf('CO2 Emissions: %.2f kg/m3\n', optionObjectives{i}.CO2Emissions);

fprintf('\n');

end

else

fprintf('No valid mix design found for the specified requirements.\n');

end

end