Nutrition_Analysis.M

```
clc; clear;
figure
data = readtable('cleaned_nutrition_dataset.csv');

% Nutrient list
nutrients = {'VitaminC', 'VitaminB11', 'Sodium', 'Calcium', ...
             'Carbohydrates', 'Iron', 'CaloricValue', 'Sugars', ...
             'DietaryFiber', 'Fat', 'Protein'};
%Choice Menu
while true
    mainChoice = menu('Nutritional Data Analysis Menu', ...
                      '1. Show summary of statistics', ...
                      '2. Sort data by a nutrient', ...
                      '3. Filter foods by nutrient threshold', ...
                      '4. Compare two nutrients (scatter plot)', ...
                      '5. Predict Caloric Value', ...
                      '6. Exit');

    switch mainChoice
        case 1  % Summary of all food statistics
            stats = getStats(data);
            disp(stats);

        case 2  % Sort by nutrient
            sortChoice = menu('Select a nutrient to sort by:', nutrients);
            if sortChoice > 0
                nutrient = nutrients{sortChoice};
                sorted = sortrows(data, nutrient, 'descend');
                fprintf('\nTop 10 foods sorted by %s:\n', nutrient);
                disp(sorted(1:10, {'food', nutrient}));
            else
                disp('No selection made.');
```

```
    end

case 3  % Filter foods by threshold
    filterChoice = menu('Select a nutrient to filter:', nutrients);
    if filterChoice > 0
        nutrient = nutrients{filterChoice};
        threshold = input(sprintf('Enter maximum value for %s: ', nutrient));
        result = filterFoods(data, nutrient, threshold);
        disp(result(:, {'food', nutrient}));
    else
        disp('No selection made.');
    end

case 4  % Compare nutrients in foods
    xChoice = menu('Select X-axis nutrient:', nutrients);
    yChoice = menu('Select Y-axis nutrient:', nutrients);
    if xChoice > 0 && yChoice > 0
        xname = nutrients{xChoice};
        yname = nutrients{yChoice};
        scatter(data.(xname), data.(yname), 'filled');
        xlabel(xname); ylabel(yname);
        title([xname ' vs ' yname]);
    else
        disp('One or both selections were canceled.');
    end

case 5  % Predict caloric value in food with nutrients
    x = data.Fat + data.Protein;
    y = data.('CaloricValue');
    p = polyfit(x, y, 1);
    fprintf('Model: CaloricValue = %.2f * (Fat + Protein) + %.2f\n', p(1), p(2));
    inputVal = input('Enter total Fat + Protein value: ');
    prediction = polyval(p, inputVal);
```

```
            fprintf('Predicted Caloric Value: %.2f\n', prediction);

        case 6
            disp('prgrm shdwn. Goodbye!');
            break;

        otherwise
            disp('Invalid choice.');
    end
end
```

getStats.m:

```
function statsTable = getStats(data)
    % Computes mean and standard deviation for all numeric nutrient columns
    numericVars = varfun(@isnumeric, data, 'OutputFormat', 'uniform');
    subdata = data(:, numericVars);

    % Calculate mean and standard deviation
    means = varfun(@mean, subdata);
    stds  = varfun(@std, subdata);

    fixedNames = {'Vitamin_C', 'Vitamin_B11', 'Sodium', 'Calcium', ...
                  'Carbohydrates', 'Iron', 'CaloricValue', 'Sugars', ...
                  'DietaryFiber', 'Fat', 'Protein'};

    means.Properties.VariableNames = fixedNames;
    stds.Properties.VariableNames  = fixedNames;

    means.Properties.RowNames = {'Mean'};
    stds.Properties.RowNames  = {'Standard_Deviation'};
```

```
    statsTable = [means; stds];
end
```

FilterFoods.m

```
function filtered = filterFoods(data, nutrient, threshold)
    % Filters rows where the selected nutrient is <= threshold.
    if ismember(nutrient, data.Properties.VariableNames)
        filtered = data(data.(nutrient) <= threshold, :);
    else
        disp('Nutrient not found. Returning empty table.');
        filtered = table();
    end
end
```