# Fruit recognition validation from the book

clc

clear all

% Load input data and target values

input\_data = readmatrix('validation/input\_p.txt');

target\_data = readmatrix('validation/target\_t.txt');

% Format data into prototypes

prototypes = cell(size(input\_data, 1), 1);

for i = 1:size(input\_data, 1)

prototypes{i} = {input\_data(i, :)', target\_data(i, :)'};

end

% Hyperparameters

input\_size = size(input\_data, 2); % Number of classes

max\_epochs = 100; % Maximum number of epochs

learning\_rate = 0.2; % Delta value

% Create and train the ADALINE network

stop\_code = 0;

ntry = 0;

number\_of\_retries = 1; % If training faill retry n times

validation\_adaline = adalineNetwork(input\_size, 'classification'); % Set for classification

while (stop\_code ~= 1 && ntry < number\_of\_retries)

[validation\_adaline, stop\_code] = validation\_adaline.train(prototypes, max\_epochs, learning\_rate);

ntry = ntry + 1;

end

% Save final weights and bias to files

w\_file = 'validation/w.txt';

b\_file = 'validation/b.txt';

% Write weights & bias to file

writematrix(validation\_adaline.Weights, w\_file, 'Delimiter', 'tab');

writematrix(validation\_adaline.Bias, b\_file, 'Delimiter', 'tab');

disp('Mode:');

disp(validation\_adaline.Mode);

disp('Weights:');

disp(validation\_adaline.Weights);

disp('Bias:');

disp(validation\_adaline.Bias);

validation\_adaline.classify(input\_data(1,:)') % -1 Orange

validation\_adaline.classify(input\_data(2,:)') % 1 Apple

# Example 1 classification 2 classes

### From book P10.2

clc

clear all

% Load input data and target values

input\_data = readmatrix('c1/input\_p.txt');

target\_data = readmatrix('c1/target\_t.txt');

% Format data into prototypes

prototypes = cell(size(input\_data, 1), 1);

for i = 1:size(input\_data, 1)

prototypes{i} = {input\_data(i, :)', target\_data(i, :)'};

end

% Hyperparameters

input\_size = size(input\_data, 2); % Number of classes

max\_epochs = 100; % Maximum number of epochs

learning\_rate = 0.3; % Delta value

% Create and train the ADALINE network

stop\_code = 0;

ntry = 0;

number\_of\_retries = 1; % If training faill retry n times

c1\_adaline = adalineNetwork(input\_size, 'classification'); % Set for classification

while (stop\_code ~= 1 && ntry < number\_of\_retries)

[c1\_adaline, stop\_code] = c1\_adaline.train(prototypes, max\_epochs, learning\_rate);

ntry = ntry + 1;

end

% Save final weights and bias to files

w\_file = 'c1/w.txt';

b\_file = 'c1/b.txt';

% Write weights & bias to file

writematrix(c1\_adaline.Weights, w\_file, 'Delimiter', 'tab');

writematrix(c1\_adaline.Bias, b\_file, 'Delimiter', 'tab');

disp('Mode:');

disp(c1\_adaline.Mode);

disp('Weights:');

disp(c1\_adaline.Weights);

disp('Bias:');

disp(c1\_adaline.Bias);

c1\_adaline.classify(input\_data(1,:)') % -1

c1\_adaline.classify(input\_data(2,:)') % -1

c1\_adaline.classify(input\_data(3,:)') % 1

# Example 2 classification 2 classes

clc

clear all

% Load input data and target values

input\_data = readmatrix('c2/input\_p.txt');

target\_data = readmatrix('c2/target\_t.txt');

% Format data into prototypes

prototypes = cell(size(input\_data, 1), 1);

for i = 1:size(input\_data, 1)

prototypes{i} = {input\_data(i, :)', target\_data(i, :)'};

end

% Hyperparameters

input\_size = size(input\_data, 2); % Number of classes

max\_epochs = 100; % Maximum number of epochs

learning\_rate = 0.1; % Delta value

% Create and train the ADALINE network

stop\_code = 0;

ntry = 0;

number\_of\_retries = 1; % If training faill retry n times

c2\_adaline = adalineNetwork(input\_size, 'classification'); % Set for classification

while (stop\_code ~= 1 && ntry < number\_of\_retries)

[c2\_adaline, stop\_code] = c2\_adaline.train(prototypes, max\_epochs, learning\_rate);

ntry = ntry + 1;

end

% Save final weights and bias to files

w\_file = 'c2/w.txt';

b\_file = 'c2/b.txt';

% Write weights & bias to file

writematrix(c2\_adaline.Weights, w\_file, 'Delimiter', 'tab');

writematrix(c2\_adaline.Bias, b\_file, 'Delimiter', 'tab');

disp('Mode:');

disp(c2\_adaline.Mode);

disp('Weights:');

disp(c2\_adaline.Weights);

disp('Bias:');

disp(c2\_adaline.Bias);

c2\_adaline.classify(input\_data(1,:)') % -1

c2\_adaline.classify(input\_data(2,:)') % -1

c2\_adaline.classify(input\_data(3,:)') % -1

c2\_adaline.classify(input\_data(4,:)') % 1

c2\_adaline.classify(input\_data(5,:)') % 1

c2\_adaline.classify(input\_data(6,:)') % 1

# Example 3 classification 4 classes

### From book P10.8

clc

clear all

% Load input data and target values

input\_data = readmatrix('c3/input\_p.txt');

target\_data = readmatrix('c3/target\_t.txt');

% Format data into prototypes

prototypes = cell(size(input\_data, 1), 1);

for i = 1:size(input\_data, 1)

prototypes{i} = {input\_data(i, :)', target\_data(i, :)'};

end

% Hyperparameters

input\_size = size(input\_data, 2); % Number of classes

max\_epochs = 100; % Maximum number of epochs

learning\_rate = 0.1; % Delta value

% Create and train the ADALINE network

stop\_code = 0;

ntry = 0;

number\_of\_retries = 1; % If training faill retry n times

c3\_adaline = adalineNetwork(input\_size, 'classification'); % Set for classification

while (stop\_code ~= 1 && ntry < number\_of\_retries)

[c3\_adaline, stop\_code] = c3\_adaline.train(prototypes, max\_epochs, learning\_rate);

ntry = ntry + 1;

end

% Save final weights and bias to files

w\_file = 'c3/w.txt';

b\_file = 'c3/b.txt';

% Write weights & bias to file

writematrix(c3\_adaline.Weights, w\_file, 'Delimiter', 'tab');

writematrix(c3\_adaline.Bias, b\_file, 'Delimiter', 'tab');

disp('Mode:');

disp(c3\_adaline.Mode);

disp('Weights:');

disp(c3\_adaline.Weights);

disp('Bias:');

disp(c3\_adaline.Bias);

c3\_adaline.classify(input\_data(1,:)') % -1,-1

c3\_adaline.classify(input\_data(2,:)') % -1,-1

c3\_adaline.classify(input\_data(3,:)') % -1,1

c3\_adaline.classify(input\_data(4,:)') % -1,1

c3\_adaline.classify(input\_data(5,:)') % 1,-1

c3\_adaline.classify(input\_data(6,:)') % 1,-1

c3\_adaline.classify(input\_data(7,:)') % 1,1

c3\_adaline.classify(input\_data(8,:)') % 1,1

# Example 4 classification 4 classes

clc

clear all

% Load input data and target values

input\_data = readmatrix('c4/input\_p.txt');

target\_data = readmatrix('c4/target\_t.txt');

% Format data into prototypes

prototypes = cell(size(input\_data, 1), 1);

for i = 1:size(input\_data, 1)

prototypes{i} = {input\_data(i, :)', target\_data(i, :)'};

end

% Hyperparameters

input\_size = size(input\_data, 2); % Number of classes

max\_epochs = 100; % Maximum number of epochs

learning\_rate = 0.1; % Delta value

% Create and train the ADALINE network

stop\_code = 0;

ntry = 0;

number\_of\_retries = 1; % If training faill retry n times

c3\_adaline = adalineNetwork(input\_size, 'classification'); % Set for classification

while (stop\_code ~= 1 && ntry < number\_of\_retries)

[c3\_adaline, stop\_code] = c3\_adaline.train(prototypes, max\_epochs, learning\_rate, 0.00001);

ntry = ntry + 1;

end

% Save final weights and bias to files

w\_file = 'c4/w.txt';

b\_file = 'c4/b.txt';

% Write weights & bias to file

writematrix(c3\_adaline.Weights, w\_file, 'Delimiter', 'tab');

writematrix(c3\_adaline.Bias, b\_file, 'Delimiter', 'tab');

disp('Mode:');

disp(c3\_adaline.Mode);

disp('Weights:');

disp(c3\_adaline.Weights);

disp('Bias:');

disp(c3\_adaline.Bias);

c3\_adaline.classify(input\_data(1,:)') % -1,-1

c3\_adaline.classify(input\_data(2,:)') % -1,-1

c3\_adaline.classify(input\_data(3,:)') % -1,-1

c3\_adaline.classify(input\_data(4,:)') % -1,1

c3\_adaline.classify(input\_data(5,:)') % -1,1

c3\_adaline.classify(input\_data(6,:)') % -1,1

c3\_adaline.classify(input\_data(7,:)') % 1,-1

c3\_adaline.classify(input\_data(8,:)') % 1,-1

c3\_adaline.classify(input\_data(9,:)') % 1,-1

c3\_adaline.classify(input\_data(10,:)') % 1,1

c3\_adaline.classify(input\_data(11,:)') % 1,1

c3\_adaline.classify(input\_data(12,:)') % 1,1

# Example resgresion

clc

clear all

% Load input data and target values

input\_data = readmatrix('1/input\_p.txt');

target\_data = readmatrix('1/target\_t.txt');

% Format data into prototypes

prototypes = cell(size(input\_data, 1), 1);

for i = 1:size(input\_data, 1)

prototypes{i} = {input\_data(i, :)', target\_data(i, :)'}; % Column vectors

end

% Hyperparameters

input\_size = size(input\_data, 2); % R

max\_epochs = 100; % Maximum number of epochs

learning\_rate = 0.15; % Delta value

error\_threshold = 0.0001; % Threshold for training error

% Create and train the ADALINE network

stop\_code = 0;

ntry = 0;

number\_of\_retries = 1;

decoder\_adaline = adalineNetwork(input\_size, 'regression');

while (stop\_code ~= 1 && ntry < number\_of\_retries)

[decoder\_adaline, stop\_code] = decoder\_adaline.train(prototypes, max\_epochs, learning\_rate, error\_threshold);

ntry = ntry + 1;

end

% Save final weights and bias to files

w\_file = '1/w.txt';

b\_file = '1/b.txt';

% Write weights & bias to file

writematrix(decoder\_adaline.Weights, w\_file, 'Delimiter', 'tab');

writematrix(decoder\_adaline.Bias, b\_file, 'Delimiter', 'tab');

disp('Mode:');

disp(decoder\_adaline.Mode);

disp('Weights:');

disp(decoder\_adaline.Weights);

disp('Bias:');

disp(decoder\_adaline.Bias);

decoder\_adaline.classify([1; 0; 1]) %5

decoder\_adaline.classify([0; 0; 1]) %1

decoder\_adaline.classify([1; 1; 1]) %7