Einführung in die Neuroinformatik

Tim Luchterhand, Paul Nykiel (Gruppe P)

15. Mai 2018

1 Lernschritt im Perzeptron-Lernalgorithmus

(a) TODO

(b)

$$\begin{aligned} w^* &= \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} \\ x^* &= \begin{pmatrix} 2 \\ 2 \\ 1 \end{pmatrix} \end{aligned}$$

(c) TODO

(d) Überprüfen ob bereits korrekt klassifiziert:

$$\begin{aligned} (w^*)^T \cdot x^* &= 2 + 4 - 1 = 5 \geq 0 \\ &\Rightarrow \quad x^* \in \omega_1 \end{aligned}$$

Lernschritt durchführen:

$$\begin{aligned} \tilde{w}^* &= w^* - \nu \cdot x^* = \begin{pmatrix} -1 \\ 0 \\ -2 \end{pmatrix} \\ \Rightarrow \tilde{w} &= \begin{pmatrix} -1 \\ 0 \end{pmatrix} \\ \tilde{x} &= -2 \end{aligned}$$

(e) TODO

(f) Überprüfen ob bereits korrekt klassifiziert:

$$\begin{aligned}(\tilde{w}^*)^T \cdot x^* &= -2+0-2=-4 \\ &\Rightarrow \quad x^* \in \omega_{-1}\end{aligned}$$

Kein Lernschritt ist notwendig $\Rightarrow w$ wird nicht verändert

2 Perzeptron-Lernalgorithmus

Matlab script:

```
%% Initialization
data = [-3  1  -1;
        -3  3   1;
        -2  1  -1;
        -2  4   1;
        -1  3   1;
        -1  4   1;
         2  2  -1;
         2  4   1;
         3  2  -1;
         4  1  -1;];


% Create all vectors
inputs = data(:,1:2);
inputsExtended = [inputs ones(size(inputs,1),1)];
classes = data(:,3);
w = [0,0,0];

% Extended weight vectors; each iteration adds one more row
% Since we don't know the exact number of rows in advance, we
    preallocate the matrix with a maximum size and crop the
    result in the end
maxVectors = 100;
vectorDimension = 3;
wExtendedMat = zeros(maxVectors, vectorDimension);
L = size(data, 1);
numberOfOptimizations = 1;
wExtendedMat(numberOfOptimizations,:) = w;

changesInLastIteration = 1;
while changesInLastIteration > 0
```

```
31      changesInLastIteration = 0;
32      for c = 1:size(inputs,1)
33          currentInput = inputsExtended(c,:);
34          desiredClass = classes(c,:);
35          calculatedOutput = w * transpose(currentInput);
36
37          % Wrong class
38          if calculatedOutput <= 0 && desiredClass > 0
39              w += currentInput;
40          elseif calculatedOutput >= 0 && desiredClass < 0
41              w -= currentInput;
42          end
43
44          % Plot the updated weights
45          if calculatedOutput * desiredClass <= 0
46              changesInLastIteration += 1;
47              numberOfOptimizations += 1;
48              wExtendedMat(numberOfOptimizations,:) = w;
49
50              x = y = 0;
51              if w(2) == 0
52                  y = -6:0.1:6;
53                  x = -w(3)/w(1);
54              else
55                  x = -6:0.1:6;
56                  y = -(w(1)*x + w(3))/w(2);
57              end
58              clf();
59              plot(x,y);
60              hold on;
61              scatter(data(:,1), data(:,2), [], data(:,3));
62              axis([-6 6 -6 6]);
63
64              while waitforbuttonpress ~= 1
65              end
66          end
67
68      end
69  end
70
71  % Remove unused weight vector entries
72  wExtendedMat = wExtendedMat(1:numberOfOptimizations, :);
```