Exercises to Pattern Recognition
Exercise Sheet 2 - Solutions

Exercises: Dipl.-Math. Stefan Wirtz
Lectures: Dr.-Ing. Marcin Grzegorzek
Fachbereich Informatik, Universität Koblenz–Landau

Date of Issue:	November 23, 2009
Submission Deadline:	December 1, 2009
Submission Form:	Email to **wirtzstefan@uni-koblenz.de** with an PDF attachment
File Name:	PR-ES2-Surname1-Surname2.pdf

1 Linear Classifiers - The Perceptron Algorithm (8 Points)

Consider a two-class problem with two-dimensional feature vectors $\boldsymbol{x} = [x_1, x_2]^{\mathrm{T}}$ distributed in each of the classes ω_1 and ω_2 in the following way

$$p(\boldsymbol{x}|\omega_1) = \frac{1}{\left(\sqrt{2\pi\sigma_1^2}\right)^2} \exp\left(-\frac{1}{2\sigma_1^2}(\boldsymbol{x}-\boldsymbol{\mu_1})^{\mathrm{T}}(\boldsymbol{x}-\boldsymbol{\mu_1})\right)$$

$$p(\boldsymbol{x}|\omega_2) = \frac{1}{\left(\sqrt{2\pi\sigma_2^2}\right)^2} \exp\left(-\frac{1}{2\sigma_2^2}(\boldsymbol{x}-\boldsymbol{\mu_2})^{\mathrm{T}}(\boldsymbol{x}-\boldsymbol{\mu_2})\right)$$

with

$$\boldsymbol{\mu}^{\mathrm{T}}{}_1 = [1,1], \qquad \boldsymbol{\mu}^{\mathrm{T}}{}_2 = [0,0], \qquad \sigma_1^2 = \sigma_2^2 = 0.2 \qquad .$$

Produce four feature vectors for each class (e. g., by coding a short script for this). To guarantee linear separability of the classes, disregard vectors with $x_1 + x_2 < 1$ for ω_1 and vectors with $x_1 + x_2 > 1$ for ω_2. Use these vectors to design a linear classifier using the perceptron algorithm, whereas $\rho = 0.7$ for all iterations.

Solutions:

- Choose $w(0)$ randomly
- Choose ρ_0
- $t = 0$
- Repeat
  - Set $Y = \emptyset$
  - For j = 1 to K
    - If $\delta_{x_j} w(j)^T x_j \geq 0 then Y = Y \cup \{x_j\}$

- End For
- $w(t+1) = w(t) - \rho_t \sum_{x \in Y} \delta_x x$
- Adjust ρ_t
- Iterate $t = t + 1$

- Until $Y = \emptyset$

Beispiel:

$$x = \begin{pmatrix} 1 & 1.1 & 1 & 0.9 & 0 & 0.1 & 0 & -0.1 \\ 1 & 1 & 1.1 & 1 & 0 & 0 & 0.1 & 0 \end{pmatrix}$$

$$\delta = \begin{pmatrix} 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \end{pmatrix}$$

$$w = \begin{pmatrix} 0.5 & 0.5 & 0.5 \end{pmatrix}^T$$

$$\rho = 0.7$$

$t = \delta_{x_j} w(j)^T x_j \geq 0?$

t
1.5
1.55
1.55
1.45
-0.5
-0.55
-0.55
-0.45

$$w = \begin{pmatrix} -2.3 & -2.37 & -2.3 \end{pmatrix}^T$$

t
-6.97
-7.2
-7.207
-6.74
2.3
2.53
2.537
2.07

$$w = \begin{pmatrix} -2.3 & -2.3 & 0.5 \end{pmatrix}^T$$

t
-4.1
-4.33
-4.33
-3.87
-0.5
-0.27
-0.27
-0.73

$$w = \begin{pmatrix} -2.3 & -2.3 & 0.5 \end{pmatrix}^T$$

2 Linear Classifiers - Sum of Error Squares Estimation (6P)

Consider again the problem of Task 1, but this time three feature vectors of each class without any assumptions[1] have to be produced. Design a classifier for this problem using the sum of error squares criterion.

Solutions:

$$\hat{w} = (X^T X)^{-1} X^T y$$

$$x = \begin{pmatrix} 0.3702 & 0.0003 & 0.5376 & -0.1433 & -0.5113 & -0.4018 \\ 0.5863 & 0.1561 & 0.8573 & -0.4476 & -0.5560 & -0.2823 \end{pmatrix}$$

$$y = \begin{pmatrix} 1 & 1 & 1 & -1 & -1 & -1 \end{pmatrix}$$

$$X^T * X = \begin{pmatrix} 0.4808 & 0.0916 & 0.7017 & -0.3155 & -0.5153 & -0.3143 \\ 0.0916 & 0.0244 & 0.1340 & -0.0699 & -0.0869 & -0.0442 \\ 0.7017 & 0.1340 & 1.0239 & -0.4607 & -0.7515 & -0.4580 \\ -0.3155 & -0.0699 & -0.4607 & 0.2209 & 0.3221 & 0.1839 \\ -0.5153 & -0.0869 & -0.7515 & 0.3221 & 0.5706 & 0.3624 \\ -0.3143 & -0.0442 & -0.4580 & 0.1839 & 0.3624 & 0.2411 \end{pmatrix}$$

$$w = \begin{pmatrix} -0.1432 & 1.8199 & -0.0987 \end{pmatrix}^T$$

3 Linear Classifiers - Support Vector Machines (8P)

Design a two-class SVM classifier for the following training features $\omega_1 \rightarrow \{[0,1]^{\mathrm{T}}, [1,0]^{\mathrm{T}}\}$ and $\omega_2 \rightarrow \{[-1,0]^{\mathrm{T}}, [0,-1]^{\mathrm{T}}\}$ using the KKT conditions.

Solutions:

$$L(w, w_0, \lambda) = \frac{1}{2} w^T w - \sum_{i=1}^{N} \lambda_i [y_i (w^T x_i + w_0) - 1]$$

KKT Krieterien:

[1] The classes do not have to be linearly separable.

1. $\frac{\partial L}{\partial w}(w, w_0, \lambda) = 0$
    - $w = \sum_{i=1}^{N} \lambda_i y_i x_i$
2. $\frac{\partial L}{\partial w_0}(w, w_0, \lambda) = 0$
    - $\sum_{i=1}^{N} \lambda_i y_i = 0$
3. $\lambda_i \geq 0; \bigvee i = 1, \dots, N$
4. $\lambda_i[y_i(w^T x_i + w_0) - 1] = 0 \bigvee i = 1, \dots, N$

Gegeb.
$\omega_1 \rightarrow \{[0,1]^{\mathrm{T}}, [1,0]^{\mathrm{T}}\}$ and $\omega_2 \rightarrow \{[-1,0]^{\mathrm{T}}, [0,-1]^{\mathrm{T}}\}$
Einsetzen:

$$L(w, w_0, \lambda) = \frac{1}{2}(w_1^2 + w_2^2) - \lambda_1(w_2 + w_0 - 1) - \lambda_2(w_1 + w_0 - 1) - \lambda_3(w_1 - w_0 - 1) - \lambda_4(w_2 - w_0 - 1)$$

KKT Resultate:

$$\frac{\partial L}{\partial w_1} = w_1 - \lambda_2 - \lambda_3 = 0$$
$$\frac{\partial L}{\partial w_2} = w_2 - \lambda_1 - \lambda_4 = 0$$
$$\frac{\partial L}{\partial w_0} = -\lambda_1 - \lambda_2 + \lambda_3 + \lambda_4 = 0$$

Oder schneller:

$$w = \sum_{i=1}^{N} \lambda_i y_i x_i$$
$$(w_1, w_2)^T = \lambda_1 * (0,1)^T + \lambda_1 * (1,0)^T - \lambda_3 * (-1,0)^T - \lambda_4 * (0,-1)^T$$
$$w_1 = \lambda_2 + \lambda_3 \wedge w_2 = \lambda_1 + \lambda_4$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0$$
$$\frac{\partial L}{\partial w_0} = -\lambda_1 - \lambda_2 + \lambda_3 + \lambda_4 = 0$$

aus iv)

$$\lambda_1[y_1(w^T x_1 + w_0) - 1] = 0$$
$$\lambda_1[w_2 + w_0 - 1] = 0$$
$$\lambda_2[w_1 + w_0 - 1] = 0$$
$$\lambda_3[w_1 - w_0 - 1] = 0$$
$$\lambda_4[w_2 - w_0 - 1] = 0$$

$$\Rightarrow w_0 = 0 \wedge w_1 = 1 \wedge w_2 = 1$$

$$\Rightarrow g(x) = x_1 + x_2 = 0$$

---

4 Nonlinear Classifiers - The Backpropagation Algorithm (8P)

Consider a simple neural network having two layers with three nodes in the hidden and one node in the output layer. Perform two iterations of the backpropagation algorithm updating the network weights for a single training pair ($\boldsymbol{x} = [1, 1, 1]^{\mathrm{T}}, y = 0.5$).

Solutions:

Step 1

Forward Computation

1. $0.1 + 0.7 + 0.1 = 0.9$
2. $0.2 + 0.3 + 0.7 = 1.2$
3. $0.8 + 0.1 + 0.1 = 1.0$

Out:

1. $\frac{1}{1+\epsilon^{-x}} = 0.7109$
2. $\frac{1}{1+\epsilon^{-x}} = 0.7685$
3. $\frac{1}{1+\epsilon^{-x}} = 0.7311$

Output slice:
$0.7109 * 0.2 + 0.7685 * 0.2 + 0.7311 * 0.2 = 0.4421$
$\frac{1}{1+\epsilon^{-x}} = 0.6088$
Error

$$F = 0.6088 - 0.5 = 0.1088$$

Output Error

$$\delta = ae_j(i)f(x)(1 - f(x)) = -0.1088 * 0.6088 * 0.3912 = -0.0259$$

New weights
$w_1' = w_1 + (\delta_1^r y_1^{r-1}) = 0.2 - 0.0259 * 0.9 = 0.1816$
$w_2' = w_2 + (\delta_2^r y_2^{r-1}) = 0.1801$
$w_3' = w_3 + (\delta_3^r y_3^{r-1}) = 0.1811$

Error for hidden layers

$$\delta_1 = \sum_{m=0}^{k_{r-1}} [\delta_k^r(i) w_{kj}^r] * f(v_m^{r-1}) * (1 - f(v_m^{r-1})) = -0.0259 * 0.1816 * 0.7109 * (1 - 0.7109) = -0.009666$$

$$\delta_2 = \sum_{m=0}^{k_{r-1}} [\delta_k^r(i) w_{kj}^r] * f(v_m^{r-1}) * (1 - f(v_m^{r-1})) = -0.008299$$

$$\delta_3 = \sum_{m=0}^{k_{r-1}} [\delta_k^r(i) w_{kj}^r] * f(v_m^{r-1}) * (1 - f(v_m^{r-1})) = -0.9221$$

New hidden layer weights
$w_4' = w_4 + (\delta_1^r y_1^{r-1}) = 0.1 + (-0.009666) * 1 = 0.0990$
$w_5' = w_5 + (\delta_2^r y_2^{r-1}) = 0.1992$
$w_6' = w_6 + (\delta_3^r y_3^{r-1}) = 0.7991$
$w_7' = w_7 + (\delta_1^r y_1^{r-1}) = 0.6990$
$w_8' = w_8 + (\delta_1^r y_1^{r-1}) = 0.2992$
$w_9' = w_9 + (\delta_1^r y_1^{r-1}) = 0.0991$
$w_1 0' = w_1 0 + (\delta_1^r y_1^{r-1}) = 0.0990$
$w_1 1' = w_1 1 + (\delta_1^r y_1^{r-1}) = 0.6992$
$w_1 2' = w_1 2 + (\delta_1^r y_1^{r-1}) = 0.0991$

Step 2 Forward Computation

1. 0.8971

2. 1.1975

3. 0.9972

Out:

1. $\frac{1}{1+\epsilon^{-x}} = 0.7104$
2. $\frac{1}{1+\epsilon^{-x}} = 0.7681$
3. $\frac{1}{1+\epsilon^{-x}} = 0.7305$

Output slice:
$Out = 0.3996$
$\frac{1}{1+\epsilon^{-x}} = 0.5986$
Error

$$F = 0.5986 - 0.5 = 0.0986$$

Output Error

$$\delta = ae_j(i)f(x)(1 - f(x)) = -0.0237$$

New weights
$w_1' = w_1 + (\delta_1^r y_1^{r-1}) = 0.1648$
$w_2' = w_2 + (\delta_2^r y_2^{r-1}) = 0.1619$
$w_3' = w_3 + (\delta_3^r y_3^{r-1}) = 0.1638$

Error for hidden layers

$$\delta_1 = \sum_{m=0}^{k_{r-1}} [\delta_k^r(i) w_{kj}^r] * f(v_m^{r-1}) * (1 - f(v_m^{r-1})) = -0.008030$$

$$\delta_2 = \sum_{m=0}^{k_{r-1}} [\delta_k^r(i) w_{kj}^r] * f(v_m^{r-1}) * (1 - f(v_m^{r-1})) = -0.006832$$

$$\delta_3 = \sum_{m=0}^{k_{r-1}} [\delta_k^r(i) w_{kj}^r] * f(v_m^{r-1}) * (1 - f(v_m^{r-1})) = -0.007637$$

New hidden layer weights
$w_4' = w_4 + (\delta_1^r y_1^{r-1}) = 0.0982$
$w_5' = w_5 + (\delta_2^r y_2^{r-1}) = 0.1985$
$w_6' = w_6 + (\delta_3^r y_3^{r-1}) = 0.7983$
$w_7' = w_7 + (\delta_1^r y_1^{r-1}) = 0.6982$
$w_8' = w_8 + (\delta_1^r y_1^{r-1}) = 0.2985$
$w_9' = w_9 + (\delta_1^r y_1^{r-1}) = 0.0983$
$w_1 0' = w_1 0 + (\delta_1^r y_1^{r-1}) = 0.0982$
$w_1 1' = w_1 1 + (\delta_1^r y_1^{r-1}) = 0.6985$
$w_1 2' = w_1 2 + (\delta_1^r y_1^{r-1}) = 0.0983$

Out:
$Out = 0.5892$
Error

$$F = 0.5892 - 0.5 = 0.0892$$
