CORRECTION EXAMEN CLASSIFICATION - 3TSI

Mardi 9 Décembre 2014 (14h-15h45)

Partiel avec documents autorisés

Exercice 1 : Bayesian Classifier

We consider a classification problem with two equiprobable classes ω_1 and ω_2 whose densities are

$$f(x|\omega_i) = \frac{\frac{1}{\pi b}}{1 + \left(\frac{x-a_i}{b}\right)^2}, \quad x \in \mathbb{R}, i \in \{1,2\} \tag{1}$$

with $b > 0$ and $a_2 > a_1$.

1. Derive the Bayesian classification rule associated with this problem.
Response : Since the classes are equiprobable, the Bayesian classifier accepts the class ω_1 if

$$f(x|\omega_1) \geq f(x|\omega_2)$$

or equivalently

$$1 + \left(\frac{x-a_1}{b}\right)^2 \leq 1 + \left(\frac{x-a_2}{b}\right)^2 \Leftrightarrow 2x(a_1 - a_2) \leq a_2^2 - a_1^2.$$

Since $a_2 > a_1$, the class ω_1 is accepted if

$$x \leq S(a_1, a_2) = \frac{a_1 + a_2}{2}.$$

2. Express the error probability of the Bayesian classifier as functions of $a_2 - a_1$ and b. Compute the limit of this probability when $a_2 - a_1$ tends to $+\infty$ and explain this result. Same question when b tends to $+\infty$.
Response : The error probability of the previous Bayesian classifier can be computed as follows

$$P_e = \int_{S(a_1,a_2)}^{+\infty} f(x|\omega_1)P(\omega_1)dx + \int_{-\infty}^{S(a_1,a_2)} f(x|\omega_2)P(\omega_2)dx.$$

After replacing the densities $f(x|\omega_1)$ and $f(x|\omega_2)$ by their expressions and making standard changes of variables, the following result is obtained

$$P_e = \int_{T(a_1,a_2)}^{+\infty} \frac{1}{2\pi}\frac{1}{1+u^2}du + \int_{-\infty}^{-T(a_1,a_2)} \frac{1}{2\pi}\frac{1}{1+u^2}du.$$

with $T(a_1, a_2) = \frac{a_2-a_1}{2b}$. The two integrals defying P_e are equal by symmetry and can be computed easily. One finally obtains

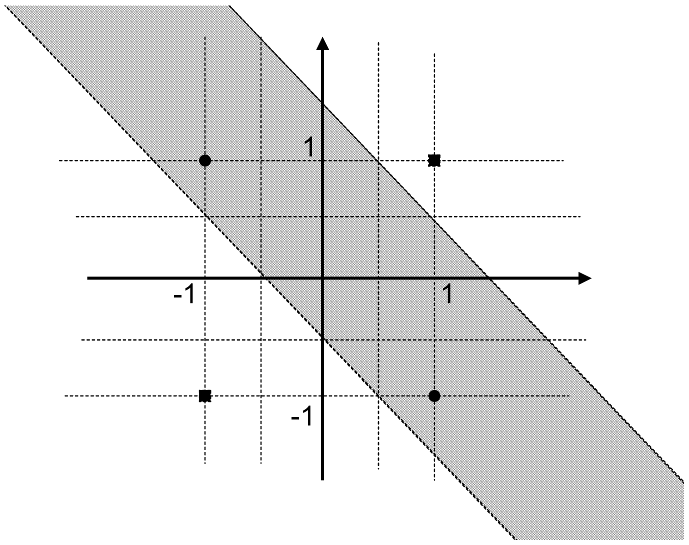
$$P_e = \frac{1}{2} - \frac{1}{\pi}\arctan\left(\frac{a_2 - a_1}{2b}\right).$$

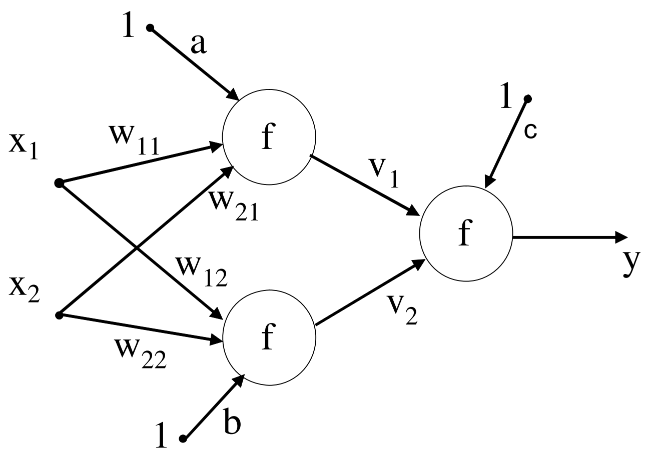
The error probability tends to 0 when $a_2 - a_1$ tend to $+\infty$ or when b tends to 0. It is a decreasing function of $\frac{a_2-a_1}{2b}$ which can be considered as a kind of of signal to noise ratio for the classification problem.

Exercice 2: Neural Network

We consider a classification problem with two classes ω_1 and ω_2 and the training samples $\boldsymbol{x}_1 = (1,1)^T$, $\boldsymbol{x}_2 = (-1,-1)^T$, $\boldsymbol{x}_3 = (1,-1)^T$ and $\boldsymbol{x}_4 = (-1,1)^T$. We know that $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are associated with class ω_1 whereas $\boldsymbol{x}_3$ and $\boldsymbol{x}_4$ are associated with class ω_2.

1. We build a neural network with one hidden layer with two nodes and one output y such that $y = 1$ if $\boldsymbol{x}$ belongs to the dark gray area displayed in the figure below and $y = 0$ else. All non-linearities used in this neural network are Heaviside functions such that $f_s(u) = 1$ if $u > 0$ and $f_s(u) = 0$ else. We also assume that the output of node 1 of the first layer is equal to 1 when $\boldsymbol{x}$ falls on one side of one of the two straight lines delimiting the dark gray area (the side corresponding to the region inside the dark gray area) and is equal to 0 else. The output of node 2 is defined similarly with the other straight line delimiting the dark gray area. Provide the values of the different weights of this neural net which is displayed in the second figure below.





Response : The dark gray area is delimited by the two lines of equations

$$x_1 + x_2 + \frac{1}{2} = 0 \quad \text{and} \quad x_1 + x_2 - \frac{3}{2} = 0.$$

More precisely, it is defined by

$$x_1 + x_2 + \frac{1}{2} > 0 \quad \text{and} \quad -x_1 - x_2 + \frac{3}{2} > 0.$$

These two equation can be rewritten as

$$w_{11}x_1 + w_{21}x_2 - a > 0 \quad w_{12}x_1 + w_{22}x_2 - b > 0$$

with $w_{11} = w_{21} = 1$, $w_{12} = w_{22} = -1$, $a = -\frac{1}{2}$ and $b = -\frac{3}{2}$. The two straight lines defined before can be generated by adjusting the weights of the two first nodes with a non-linearity providing 0 is the input is negative and 1 if the input is positive. If we denote the outputs of the two first nodes as y_1 and y_2, the node of the last layer has to perform an "and" operation. This operation can be obtained as follows

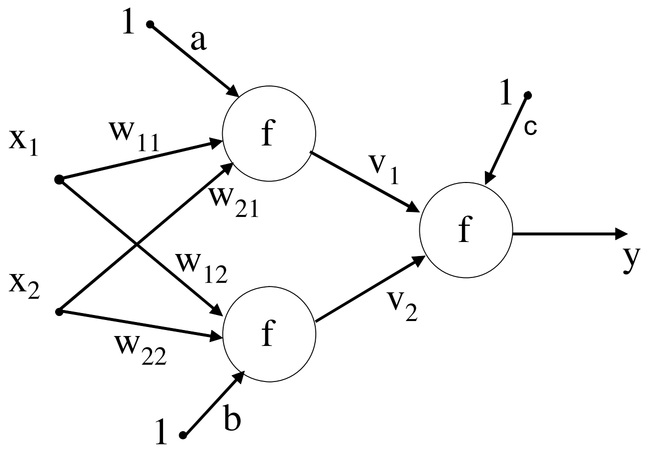
$$y = f(v_1y_1 + v_2y_2 - c)$$

with $v_1 = v_2 = 1$ and $c \in]1, 2[$, (e.g., $c = \frac{3}{2}$).

2. The neural net investigated in the first question is displayed in the figure below, where $\boldsymbol{w} = (w_{11}, w_{12}, w_{22}, w_{22})^T$ is the weight vector of the first layer with offsets a and b, and $\boldsymbol{v} = (v_1, v_2)^T$ is the weight vector of the output with offset c. We recall the following relations

$$y_1 = f(w_{11}, x_1 + w_{21}x_2 - a),\ y_2 = f(w_{12}, x_1 + w_{22}x_2 - b) \text{ and } y = f(v_1y_1 + v_2y_2 - c).$$

where we assume $f(t) = \frac{\exp(\alpha t)}{1+\exp(\alpha t)}$.What are the back propagation update rules associated with this neural net?



Response : we know that $f'(x) = \alpha f(x)[1 - f(x)]$. The gradient rule for the weights v_i can be written

$$v_i(n+1) = v_i(n) - \mu \left.\frac{\partial e^2(n)}{\partial v_i}\right|_{v_i=v_i(n)}$$

for $i = 1, 2$. Using

$$e^2(n) = [d(n) - y(n)]^2 = [d(n) - f(v_1y_1(n) + v_2y_2(n) - c)]^2$$

we obtain

$$\frac{\partial e^2(n)}{\partial v_i} = -2e(n)\frac{\partial e(n)}{\partial v_i} = -2\alpha e(n)y_i(n)y(n)[1 - y(n)]$$

hence

$$v_i(n+1) = v_i(n) + \delta e(n) y_i(n) y(n)[1 - y(n)], \quad i = 1, 2.$$

Similarly, the update of the offset parameter c is

$$c(n+1) = c(n) + \delta_c e(n) y(n)[1 - y(n)].$$

Using the definitions of y_1 and y_2

$$y_1 = f(w_{11}x_1 + w_{21}x_2 - a) \quad \text{and} \quad y_2 = f(w_{12}x_1 + w_{22}x_2 - b)$$

we obtain

$$\begin{aligned}
\frac{\partial y(n)}{\partial w_{11}} &= \alpha^2 y(n)[1 - y(n)] v_1(n) x_1(n) y_1(n)[1 - y_1(n)] \\
\frac{\partial y(n)}{\partial w_{21}} &= \alpha^2 y(n)[1 - y(n)] v_1(n) x_2(n) y_1(n)[1 - y_1(n)] \\
\frac{\partial y(n)}{\partial w_{12}} &= \alpha^2 y(n)[1 - y(n)] v_2(n) x_1(n) y_2(n)[1 - y_2(n)] \\
\frac{\partial y(n)}{\partial w_{21}} &= \alpha^2 y(n)[1 - y(n)] v_2(n) x_2(n) y_2(n)[1 - y_2(n)]
\end{aligned}$$

or equivalently

$$\frac{\partial y(n)}{\partial w_{ij}} = \alpha^2 y(n)[1 - y(n)] v_j(n) x_i(n) y_j(n)[1 - y_j(n)].$$

Similarly, the partial derivatives of $y(n)$ with respect to the the offsets are

$$\begin{aligned}
\frac{\partial y(n)}{\partial a} &= \alpha^2 y(n)[1 - y(n)] v_1(n)(-1) y_1(n)[1 - y_1(n)] \\
\frac{\partial y(n)}{\partial b} &= \alpha^2 y(n)[1 - y(n)] v_2(n)(-1) y_2(n)[1 - y_2(n)].
\end{aligned}$$

Finally, the updating rules for the weight w_{ij} and the offsets a and b are

$$\begin{aligned}
w_{ij}(n+1) &= w_{ij}(n) + \delta_{ij} e(n) \left.\frac{\partial y(n)}{\partial w_{ij}}\right|_{w_{ij}=w_{ij}(n)}, \quad i, j = 1, 2. \\
a(n+1) &= a(n) + \delta_a e(n) \left.\frac{\partial y(n)}{\partial a}\right|_{a=a(n)}, \\
b(n+1) &= b(n) + \delta_b e(n) \left.\frac{\partial y(n)}{\partial b}\right|_{b=b(n)}.
\end{aligned}$$

Questions related to the working paper

1. What are the advantages of hyperspectral sensors with respect to multispectral sensors?
   Response : Hyperspectral sensors have a higher spectral resolution since they provide measurements acquired in hundreds of observation channels.

2. Explain what the authors mean by the "curse of dimensionality" (Second column of page 1778).
   Response : when the dimensionality of the problem (size of the vector to classify) is too large, we need a training set with a lot of examples to obtain a reasonable classification performance. In practical applications, it is often difficult to acquire all these examples and thus we need to map the data into a lower dimensional subspace. Methods that can be used for this mapping include the principal component analysis or the linear discriminant analysis.

3. Explain the differences between supervised and unsupervised classification rules.
Response : supervised rules are used when the number of classes is known and when some examples associated with each class (training samples) are available. Conversely, unsupervised classification rules aim at separating the data into different clusters without having a training set.

4. Explain the principles of the basic sequential forward selection (SFS) (mentioned in the first column of page 1779) and an example of feature selection criterion that can be used for the SFS method.
Response : the basic SFS algorithm selects features sequentially in order to obtain maximize an appropriate cost function C. A classical choice of cost function is $C = 1 - P_e$, where P_e is the error probability of the classifier. In this case, we begin by choosing the feature f_1 which minimizes P_e. After determining this feature, we compute the feature f_2 such that the pair (f_1, f_2) minimizes P_e, and so on.

5. How can we obtain Eq. (5) from (4)?
Response : see course on support vector machines

6. What is the leave-one-out (LOO) rule mentioned in the first column of page 1782?
Response : The leave-one-out (LOO) rule is a rule for computing the error probability of a classifier. Assume that n samples are available in a database. The LOO rule leaves one sample out of this database for testing and uses the $n - 1$ remaining sample for training. This operation is repeated n times and the error probability of the classifier is computed by counting the number of errors obtained during these n operations.

7. Provide the mathematical expressions of the discriminant functions used for the one-against-all (OAA) rule mentioned after Eq. (16)?
Response : the discriminant functions used for the one-against-all (OAA) rule are defined as $f_k(\boldsymbol{x}) = y_i(\boldsymbol{w}_k \,.\, \boldsymbol{x} + b_k)$, where $\boldsymbol{w}_k$ and b_k define the separating hyperplane for the kth support vector machine.

8. What are the minimum and maximum value of the score function $S_i(x)$? When do we obtain these minimum and maximum values?
Response : The maximum value of the score function is $S_i(x) = T - 1$, which is obtained when the class ω_i has been preferred to all other classes ω_j, for $j \neq i$. The minimum value of the score function is $S_i(x) = -(T - 1)$, which is obtained when the class ω_i has never been preferred to another class ω_j, for $j \neq i$.

9. Explain how the tree of Fig. 6 (a) has been obtained and how a vector x is classified using this tree.
Response : in order to build the tree of Fig. 6 (a), we first need to build two groups of classes $\Omega^0_{A,0}$ and $\Omega^0_{B,0}$ having similar prior probabilities using the class priors $P(\omega_i) = \frac{n_i}{n}$ (where n_i is the number of training samples of the ith class and $n = \sum_i n_i$) . By using the numbers of samples indicated in Table I, we obtain $P(\Omega^0_{A,0}) = P(\omega_1) + P(\omega_2) + P(\omega_7) = 0.51$ and $P(\Omega^0_{B,0}) = 0.49$. We run an SVM classifier for these two groups of classes denoted as SVM 1 which separates the two groups into two sets of samples. The procedure is repeated for each output of the classifier SVM 1. For instance, in the right branch of the tree, one group is formed of class 7 (with prior close to 0.27) and the other group is formed of the two classes ω_1 and ω_2 (with prior close to 0.25). The procedure stops when a leaf of the tree contains a unique class. In order to classify the vector $\boldsymbol{x}$, we run the different SVM classifiers sequentially with $\boldsymbol{x}$ as an input and we stop when $\boldsymbol{x}$ is assigned to a leaf corresponding to a given class.

10. Explain how the tree of Fig. 6 (b) has been obtained. In particular, justify the order of the different classes (i.e., ω_7 first, ω_1 second, ...) and explain how a vector x is classified using this tree.
Response : the tree of Fig. 6 (b) is obtained by running different SVM classifiers. Each classifier

is built from two groups of classes : the first group contains the most probable class (according to its prior) and the second group contains the remaining classes. For instance, at the first step, class ω_7 which is the most likely is chosen for group A and the group B contains all other classes. At the second step, class ω_1 constitues group A since it has the maximum prior probability, etc... n order to classify the vector $\boldsymbol{x}$, we run the different SVM classifiers sequentially with $\boldsymbol{x}$ as an input and we stop when $\boldsymbol{x}$ is assigned to a leaf corresponding to a given class.

11. Assume that we have $n = 200$ test samples ($n_0 = 100$ from class ω_0 and $n_1 = 100$ from class ω_1). A classifier correctly classifies 70 samples from class ω_0 (called true negatives) and 80 samples from class ω_1 (called true positives). What is the overall accuracy for this problem? (the overall accuracy is indicated in Table V for the classification of hyperspectral pixels).
Response : the overall accuracy is the ratio between 1) the sum of true positives and true negatives and 2) the total number of samples. For the problem considered above, we have $OA = (70 + 80)/200 = 0.75$.

12. How do the authors of this paper justify the poor performance of k-nearest neighbor rule for their problem?
Response : it is mentioned p. 1786 that “the small number of training samples is not sufficient to fill in a proper way the emptiness of the hyper dimensional feature space” which explains the relatively poor classification accuracies of the K-nn classifier.