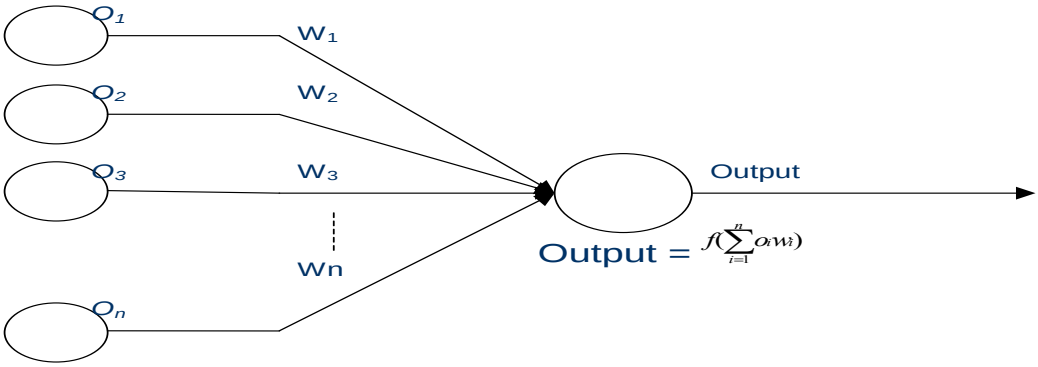
Data Mining

Apprendre ? Les réseaux neuronaux

Le neurone artificiel est modélisé ainsi :



avec des fonctions d'activations au choix :

Pas unitaire		$f(x) = \begin{cases} 0 \text{ if } 0 > x \\ 1 \text{ if } x \geq 0 \end{cases}$
Sigmoïde		$f(x) = \frac{1}{1+e^{-\beta x}}$
Linéaire Seuillée		$f(x) = \begin{cases} 0 & \text{if } x \leq x_{min} \\ mx+b & \text{if } x_{max} > x > x_{min} \\ 1 & \text{if } x \geq x_{max} \end{cases}$
Gaussienne		$f(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$
Identité		$f(x) = x$

Tutorial. On va en Suisse.

Rendez vous sur le site de l'Université de Lausanne :
http://diwww.epfl.ch/mantra/tutorial/french/

1. **Neurone Artificiel** :

En utilisant la fonction d'activation de pas unitaire, on va essayer de déterminer **sur papier** un ensemble de poids (et une valeur de seuil) qui va produire la classification suivante :

x1	x2	sortie
-0.2	0.5	0
0.2	-0.5	0
0.8	-0.8	1
0.8	0.8	1

- Représenter dans un repère, le problème de classification.
- Représentez le neurone artificiel correspondant.
- En utilisant la fonction d'activation linéaire, déterminez un ensemble de poids (et une valeur de seuil) qui va produire la classification exigée.
- En utilisant la fonction d'activation échelon ou pas (fonction linéaire seuillée pour $x_{min}=x_{max}=0$), déterminez un ensemble de poids (et une valeur de seuil) qui va produire la classification exigée.
- Testez l'ensemble de ce qui précède grâce à l'applet http://diwww.epfl.ch/mantra/tutorial/french/aneuron/html/index.html .
- Seriez-vous capable de tenir le raisonnement que précédemment avec la fonction d'activation sigmoïde ($\beta=1$) ?. Qu'apporte une telle fonction pour le problème de classification ?
- Sur l'ensemble de classification suivant, ce modèle permettrait-t-il de répondre à la question ? Si non, proposez un modèle neuronal possible.

x1	x2	sortie
-0.2	0.5	0
0.2	-0.5	1
0.8	-0.8	0
0.8	0.8	1

- Testez l'ensemble de ce qui précède grâce à l'applet http://diwww.epfl.ch/mantra/tutorial/french/aneuron/html/index.html .
- Quel comportement différent induit le changement de fonction d'activation ?

2. **Apprentissage sur perceptron simple** :

- Lancer l'applet correspondante : http://diwww.epfl.ch/mantra/tutorial/french/perceptron/html/index.html
- Modifier les paramètres de Learning Rate, Number of Iterations et Error Threshold pour chaque fonction d'activation pour apprendre la configuration (0,0,0), (1,0,0), (0,1,0) et (1,1,1) avec (x,y,classe). Commentez.
- Même question pour le problème du XOR. (N'utilisez pas la case à cocher SOLVE XOR).

On rappelle le principe de l'algorithme d'apprentissage par rétro-propagation de l'erreur :

Box 6. The Back-Propagation Training Algorithm

The back-propagation training algorithm is an iterative gradient algorithm designed to minimize the mean square error between the actual output of a multilayer feed-forward perceptron and the desired output. It requires continuous differentiable non-linearities. The following assumes a sigmoid logistic non-linearity is used where the function $f(\alpha)$ in Fig. 1 is

$$f(\alpha) = \frac{1}{1 + e^{-(\alpha - \theta)}}$$

Step 1. Initialize Weights and Offsets

Set all weights and node offsets to small random values.

Step 2. Present Input and Desired Outputs

Present a continuous valued input vector $x_0, x_1, \ldots x_{N-1}$ and specify the desired outputs $d_0, d_1, \ldots d_{M-1}$. If the net is used as a classifier then all desired outputs are typically set to zero except for that corresponding to the class the input is from. That desired output is 1. The input could be new on each trial or samples from a training set could be presented cyclically until weights stabilize.

Step 3. Calculate Actual Outputs

Use the sigmoid nonlinearity from above and formulas as in Fig. 15 to calculate outputs $y_0, y_1 \ldots y_{M-1}$.

Step 4. Adapt Weights

Use a recursive algorithm starting at the output nodes and working back to the first hidden layer. Adjust weights by

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x'_i$$

In this equation $w_{ij}(t)$ is the weight from hidden node i or from an input to node j at time t, x'_i is either the output of node i or is an input, η is a gain term, and δ_j is an error term for node j. If node j is an output node, then

$$\delta_j = y_j(1 - y_j)(d_j - y_j),$$

where d_j is the desired output of node j and y_j is the actual output.

If node j is an internal hidden node, then

$$\delta_j = x'_j(1 - x'_j) \sum_k \delta_k w_{jk},$$

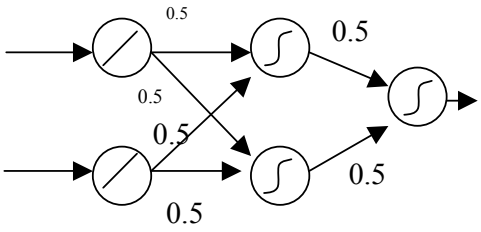
where k is over all nodes in the layers above node j. Internal node thresholds are adapted in a similar manner by assuming they are connection weights on links from auxiliary constant-valued inputs. Convergence is sometimes faster if a momentum term is added and weight changes are smoothed by

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x'_i + \alpha(w_{ij}(t) - w_{ij}(t-1)),$$

where $0 < \alpha < 1$.

Step 5. Repeat by Going to Step 2

- Sur l'ensemble d'apprentissage suivant, et à partir du réseau à une couche cachée initial suivant, faites tourner **sur papier** l'algorithme d'apprentissage par rétro-propagation du gradient (back-propagation) sur 2 itérations.



x1	x2	sortie
0	1	1
0	0	0
1	1	0
1	0	1

- Expliquez l'intérêt d'utiliser un réseau neuronal puis définissez les étapes d'apprentissage pour les données du cas marketing du TP-Data. Imaginez les applications dans le cadre du Web Mining.

3. Applet Adaline, Perceptron et Rétro-propagation :

- **Cas idéal:** placez 10 points rouges (classe 1) et 10 points bleus (classe 0) dans deux groupes similaires, distincts et linéairement séparables et apprenez avec l'algorithme de rétropropagation. Quelles valeurs de taux d'apprentissage (learning rate) donne les meilleurs résultats ?

- **Différentes dispersions de groupe**: Placez 20 poitns rouges (1) dans un très petit groupe (points fortement corrélés) et 5 points bleus (0) dans un très large groupe de manière à ce que les deux classes soient linéairement séparables et apprenez avec l'algorithme de rétropropagation. Quelles valeurs de taux d'apprentissage (learning rate) donne les meilleurs résultats ?
- **Séparation non parfaite:** Placez 10 points rouges (1) et 10 points bleus (0) dans deux groupes similaires et linéairement séparables. Placez ensuite des points bleus supplémentaires à l'intérieur du groupe de points rouges et apprenez avec l'algorithme de rétropropagation. Quelles valeurs de taux d'apprentissage (learning rate) donne les meilleurs résultats ?

3. Applet Perceptron multi-couche (sortie des neurones: {0;1})

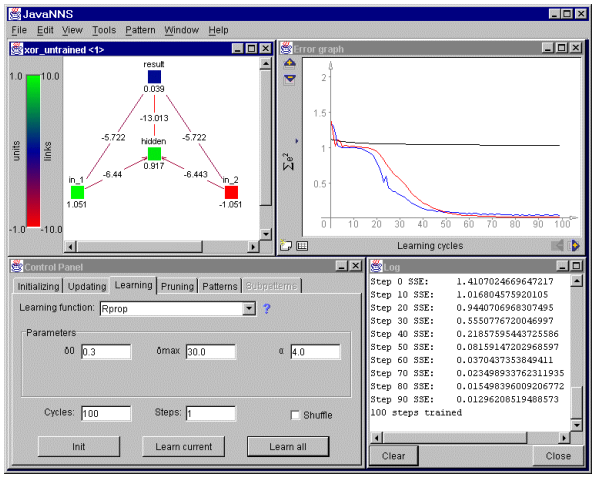
- Résolvez le problème du XOR.
- Placez trois groupes de points en ligne : un rouge avec 3 points, un bleu avec 6 points et encore un rouge avec 3 points. Est-ce que le perceptron multi-couche le plus simple est capable de résoudre ce problème ? Si ce n'est pas le cas, quelle est la structure minimale qui permet de résoudre ce problème ? Précisez le momentum et le taux d'apprentissage (learning rate).
- Placez un groupe de points rouges au centre, entouré par des points bleus. Quel structure de réseau et quels paramètres (momentum et taux d'apprentissage) permettent de résoudre un tel problème ?

Cas génétique : le logiciel JavaNNS qui nous vient de Stuttgart.

Ce logiciel gratuit permet de faire de l'apprentissage sur des données plus conséquentes que les précédentes.

Un rapide tour d'horizon

- Ouvrez les fichiers xor_untrained.net et xor.pat qui stockent un réseau simple et le fichier de motifs à apprendre correspondant : le bien connu problème du xor.



- Pour visualiser le réseau si nécessaire dans la fenêtre principale, choisissez View/Nework.
- Les neurones et les liens ont des couleurs différentes symbolisant les différentes valeurs des unités d'activation et des poids des connexions.
- A présent, entraînons le réseau de neurones à reconnaître les motifs, c'est-à-dire reprogrammons les poids pour que le réseau fournisse les sorties désirées quand un motif d'entrée est fourni. Pour cela, ouvrez le Control Panel à partir du menu Tools. C'est la fenêtre la plus importante car toutes les modifications et manipulations du réseau y seront effectuées. Ouvrez également la fenêtre d'Error Graph pour observez l'évolution de processus d'apprentissage et Log pour des informations.

- Dans le Control Panel, rendez-vous à l'onglet Learning. Choisissez la fonction d'apprentissage classique Backpropagation et étudiez les paramètres par défaut. A quoi correspondent-ils pour chacun d'eux ?
- Lancez l'apprentissage en pressant le bouton « Learn all ». Qu'observez-vous et expliquez ?

Quelques indications pour créer un réseau adapté à vos données :

- Avec un éditeur de texte de votre choix, visualisez le fichier de motifs good_train.pat. Donnez-en une interprétation possible dans le domaine du marketing par exemple.
- Choisissez File/New pour supprimer le réseau en cours du simulateur. Puis File/Open good_train.pat et good_valid.pat pour rendre disponibles les deux fichiers de motifs en question.

Nous allons pouvoir créer un réseau adapté à ces motifs à 4 neurones d'entrée, 2 neurones de sortie et une couche cachée à 3 neurones.

- Choose height "4", unit type "Input" and click "Create" to create a new layer. For the next layer, set height to 3 and the unit type to "Hidden" and click "Create" again. Finally, create the output layer with the height of 2 and unit type "Output" and close the window. To connect the created units, use Create/Connections from the Tools menu. Simply choose "Connect feed-forward" and click "Connect". Doing that, you have created a simple feed-forward neural network, with 4 input, 3 hidden and 2 output unit. You can now close the Connections window, too
- Pour voir comment deux ensembles de motifs sont utilisés pour l'apprentissage et la validation, chargez les deux fichiers good_train.pat et good_valid.pat à partir de l'onglet Pattern.
- Revenez à l'onglet Learning et entraînez le réseau. Durant l'apprentissage deux courbes sont affichées dans Graph Error, l'une qui affiche l'erreur sur l'ensemble d'apprentissage et l'autre en rose qui représente l'erreur sur l'ensemble de validation. Observez éventuellement l 'effet de sur-apprentissage. A quel comportement et de quel courbe cela correspond-t-il ?
- A présent, utilisez vos données *Dataset.data* qui se trouve dans le répertoire *splice* récupéré auparavant à Toronto. Ce fichier est composée de lignes avec sur chaque ligne la classe à laquelle appartient la séquence EI, IE, ou N puis la séquence ADN concernée. Ce fichier doit permettre de prévoir si le nucléotide central de la séquence proposée marque une transition entre une partie codante (EXON) et non-codante (INTRON) (classe EI), ou la transition contraire (classe IE), ou bien ni l'une ni l'autre (classe N pour Neutre).Vous allez apprendre un réseau de neurones puis le tester en suivant les indications suivantes :
    1. Prenez 1/3 des données pour l'apprentissage et 2/3 pour le test (pensez à la commande awk vue précédemment).
    2. Créez un réseau neuronal à 2 couches cachées et entraînez-le.

S'il vous reste du temps, faites les exercices suivants sur le site de Lausanne.

1. Généralisation avec un Perceptron multi-couche (sortie des neurones: {0;1}).
2. Reconnaissance de caractères optique (OCR) avec un Perceptron multi-couche.
3. Prédiction avec un Perceptron multi-couche.
4. Amusez vous avec le fichier *titanic.tar.gz*. et JavaNNS