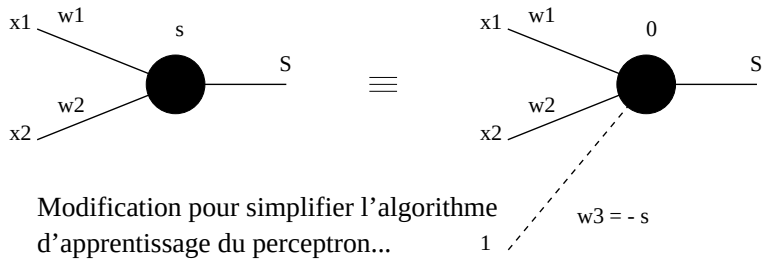
– TD 1 – Le Perceptron –

TD de Connexionnisme - M1 informatique de Lyon 2

Didier Puzenat
didier.puzenat@univ-lyon2.fr

1 Introduction

Nous allons implémenter en Python un « perceptron » et lui apprendre des fonctions booléennes à deux variables. Afin de simplifier l'algorithme, le seuil est considéré comme un poids supplémentaire connecté à une entrée toujours à 1 :



2 Algorithme

1 Initialiser aléatoirement les poids W = (w1 w2 w3)
2 Choisir un exemple X = (x1 x2 1)
3 Activer le « perceptron », i.e. calculer $S = \mathcal{H}(W^t X) = \mathcal{H}(\sum_{i=1}^{3} w_i x_i)$
où $\mathcal{H}$ est la fonction de Heaviside vue en cours
4 Modifier les poids suivant la règle suivante : $W \leftarrow W + \eta \times E \times X$
où E est la différence entre la sortie **désirée** et la sortie **obtenue** (S)
et η (êta) est un coefficient d'apprentissage
5 Retourner en 2 « si nécessaire »

3 À faire obligatoirement

Pour la fonction OU puis la fonction ET :

1. apprendre la fonction avec l'algorithme du Perceptron, voir la section 5 pour un peu d'aide sur une structure possible de la base d'exemples ;
2. afficher dans la console l'évolution de l'erreur durant l'apprentissage ;
3. afficher dans la console d'équation de la droite de séparation que forment les poids du perceptron, sous la forme $w_1 x_1 + w_2 x_2 + w_3 = 0$;
4. tracer la droite de séparation (voir la section 6 pour un peu d'aide), lancer plusieurs fois le programme pour obtenir autant de droites de séparation différentes puisque les vecteurs de départ sont aléatoires.

4 À faire si vous avez le temps

Si vous avez terminé avant vos camarades, vous pouvez faire une version du programme montrant l'évolution de l'apprentissage en traçant les droites successives, par exemple avec des couleurs différentes.

5 Aide sur la structure de la base d'exemples

On peut déclarer pour chaque base d'exemples (celle du OU et celle du ET) une liste de 4 tuples avec le vecteur exemple sous la forme d'une liste comme premier terme du tuple et la sortie désirée comme second terme. La base d'exemples comprend tous les cas possibles, ce qui n'est évidemment pas le cas dans un problème réel.

```
OU = [ ([0,0,1], 0),
       ([0,1,1], 1),
       ([1,0,1], 1),
       ([1,1,1], 1) ]
```

```
ET = [ ([0,0,1], 0),
       ([0,1,1], 0),
       ([1,0,1], 0),
       ([1,1,1], 1) ]
```

Dès lors, pour l'étape 2 de l'algorithme, vous pouvez « choisir » un exemple `X` et la sortie désirée correspondante `d` par une simple affectation, par exemple `X, d = OU[0]`.

6 Aide pour le tracé de la droite

Le module `pylab` propose de nombreuses fonctions pour le tracé des courbes, par exemple : `plot`, `xlim` et `ylim` pour définir le repère, `title` pour donner un titre au graphique, etc. À noter que la fonction `plot` nécessite d'expliciter la formule de la droite sous la forme $y = ax + b$. Si vous débutez avec `pylab`, le code suivant devrait bien vous aider :

```
from matplotlib.pylab import plot, xlim, ylim, show, title

xlim([0,1])
ylim([0,1])
title("Hyperplan separateur")

y = []
y.append(b)
y.append(a+b)
plot([0,1], y)

show()
```

7 Usage du module `numpy`

Sans que ce soit obligatoire pour le rendu, vous pouvez fortement simplifier le code en déclarant les exemples avec des `array` plutôt que comme des listes :

```
from numpy import array

OU = [ (array([0,0,1]), 0),
       (array([0,1,1]), 1),
       (array([1,0,1]), 1),
       (array([1,1,1]), 1), ]
```

Dès lors vous pouvez utiliser le produit scalaire (`dot`) pour calculer l'activation du Perceptron, et faire directement des opérations sur les vecteurs (`W += eta * erreur * X`) pour peu que `W` et `X` soient aussi de type `array`. Le code est du coup beaucoup plus lisible.

Idem pour le tracé :

```
x = array([0,1])
y = a * x + b
plot(x, y)
```

8 Questions

Pour le Perceptron et le problème jouet de cette fiche de TD :

1. quelles remarques peut-on faire sur les droites de séparation trouvées par l'algorithme, et que peut-on en déduire sur ses capacités en généralisation ?
2. Essayer le programme avec le OU exclusif : que se passe-t-il ? Comment « résoudre » ce problème, et quelle est alors la difficulté ?

Dans le cas d'un problème réel, avec beaucoup d'exemples à apprendre et des entrées réelles ($X \in \mathbb{R}^n$), que faire dans l'éventualité où le Perceptron :

1. ait des difficultés à converger ?
2. n'arrive pas du tout à converger ?

9 Rendu durant la séance

À rendre durant la séance, sur le Moodle du CM :

1. le code, si possible sous la forme d'un seul fichier enchaînant le travail demandé ;
2. un **mini** rapport d'une page en PDF avec les figures et la réponse aux questions.

Si vous n'avez pas terminé durant la séance, merci de faire votre rendu dès que possible, mais la note pourra éventuellement être impactée.