

Apprentissage Automatique

Mini-projet M1 GIL : Réseau
de neurones appliqué au
jeu des bâtons

5 mai 2017

HFIDHI Hajer
Ben HMIDA Ali

Table des matières

I.	Introduction.....	3
II.	Travail à rendre	3
III.	Les bases et le mode simple.....	3
1.	Présenté par le script1.py.....	3
IV.	Mode intermédiaire	4
V.	Implémentation du réseau de neurones.....	5
1.	Méthode testNeuron de la classe Neuron	5
2.	Méthode chooseConnectedNeuron de la classe Neuron	5
3.	Méthode playHard de la classe CPUPlayer	5
4.	Méthode recompenseConnection de la classe Neuron	6
VI.	Apprentissage.....	6
VII.	Jeu final.....	7
VIII.	Question optionnel (entrainement.py)	7

I. Introduction

Dans le cadre du TP d'Apprentissage, il nous a été demandé de créer des scripts de test et de compléter les différentes méthodes et classe **Neuron** et **CPUPlayer**.

II. Travail à rendre

Les scripts Python correspondant aux différents scénarios :

SCRIPT1.py : permet de tester le mode de jeu easy

SCRIPT2.py : permet de jouer 800 parties en mode hard

SCRIPT3.py : permet de jouer 800 parties des différents modes contre eux

SCRIPT4.py : permet de choisir le mode de jeu et si le joueur humain joue au premier

entrainement.py : permet de jouer 1000000 parties en mode hard pour apprendre

III. Les bases et le mode simple

1. Présenté par le script1.py

Dans le mode easy et lors du dernier tour. L'ordinateur ne peut pas faire le contrôle où il reste soit 3 ou 4 bâtons en ayant toutes les chances de gagner. C'est-à-dire :

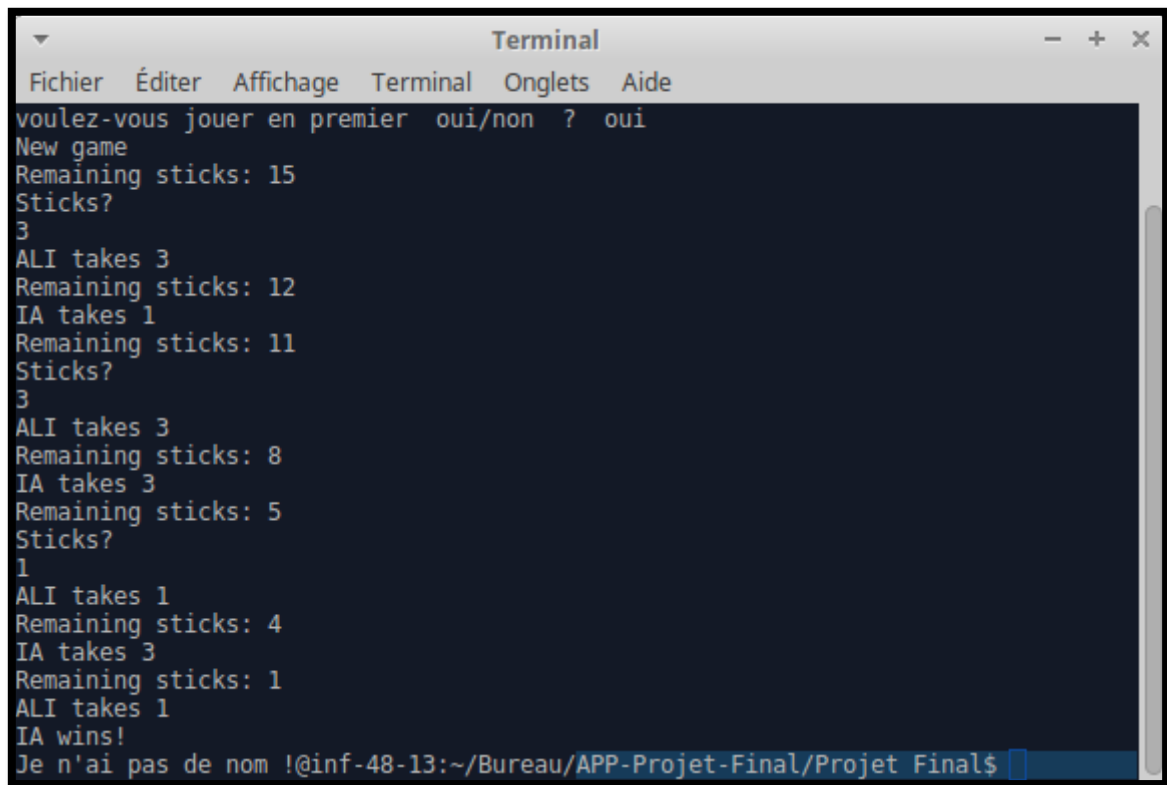
S'il reste 3 bâtons

L'ordinateur peut prendre un seul bâton et de ce fait, le joueur va prendre 1 bâton et l'ordinateur ne va pas gagner. Or que l'idéale est de prendre 2 bâtons.

S'il reste 4 bâtons

L'optimale est de prendre 3 bâtons. Mais l'ordinateur peut prendre 1 ou 2 bâtons, alors il ne va pas gagner.

❖ Script 1



```
Terminal
Fichier Éditer Affichage Terminal Onglets Aide
voulez-vous jouer en premier oui/non ? oui
New game
Remaining sticks: 15
Sticks?
3
ALI takes 3
Remaining sticks: 12
IA takes 1
Remaining sticks: 11
Sticks?
3
ALI takes 3
Remaining sticks: 8
IA takes 3
Remaining sticks: 5
Sticks?
1
ALI takes 1
Remaining sticks: 4
IA takes 3
Remaining sticks: 1
ALI takes 1
IA wins!
Je n'ai pas de nom !@inf-48-13:~/Bureau/APP-Projet-Final/Projet Final$
```

IV. Mode intermédiaire

On peut faire un contrôle pour éviter que l'ordinateur ne commette une erreur au dernier tour lorsqu'il a toutes les chances de gagner (le nombre de bâtons restants est de 2, 3 ou 4)

```
Def playMedium(self,sticks) :
if sticks==4 : move=3
elif sticks==3 : move=2
elif sticks==2 : move=1
else : move=random.randint(1,3)
return move
```

En ajoutant ces conditions en mode medium, on est sûre que l'ordinateur gagnera dans le cas où il reste 2, 3, ou 4 bâtons.

Mais par contre **ce n'est pas de l'apprentissage**, vu que ce sont des conditions imposées à l'ordinateur.

V. Implémentation du réseau de neurones

1. Méthode testNeuron de la classe Neuron

```
def testNeuron(self,inValue):  
    return (inValue - self.index) in range(1, 4)
```

2. Méthode chooseConnectedNeuron de la classe Neuron

```
def chooseConnectedNeuron(self,shift):  
    connections = self.connections.copy()  
    neuron = self.weighted_choice(connections)  
    while not neuron == None and not  
neuron.testNeuron(self.index - shift):  
        connections.pop(neuron)  
        neuron = self.weighted_choice(connections)  
    return neuron
```

3. Méthode playHard de la classe CPUPlayer

```
def playHard(self,sticks):  
    nb = 0  
    if self.previousNeuron == None:  
        currentNeuron = self.netw.getNeuron(sticks)  
        neuron = currentNeuron.chooseConnectedNeuron(0)  
        nb = sticks - neuron.index  
        self.netw.activateNeuronPath(currentNeuron, neuron)  
        self.previousNeuron = neuron  
    else:  
        shift = self.previousNeuron.index - sticks  
        neuron = self.previousNeuron.chooseConnectedNeuron(shift)  
        if not neuron == None:  
            nb = sticks - neuron.index  
            self.netw.activateNeuronPath(self.previousNeuron, neuron)  
            self.previousNeuron = neuron  
        else:  
            nb = 1  
    return nb
```

4. Méthode recompenseConnection de la classe Neuron

```
def recompenseConnection(self,neuron):  
    self.connections[neuron] = self.connections[neuron] + RECOMPENSE
```

Si on teste l'utilisation du réseau de neurones en faisant jouer la machine d'abord contre nous-même (mode "hard", nombre de bâtons = 15 et mode "verbose" à True). On va sûrement gagner car jusqu'à ce moment l'IA ne sait pas encore comment jouer.

VI. Apprentissage

Si l'IA joue contre lui-même en mode "hard". Donc cette méthode s'appelle : **Apprentissage**.

- On va constater que le joueur qui commence en premier c'est celui qui va systématiquement gagner.
- Pour le mode Easy le Medium va gagner
- Pour le mode medium il y a pourcentage de 50% pour que le 1ere joueur gagne.
- Pour le mode hard toujours le 1ere joueur qui commence celui qui gagne.
- Le test nous permet de choisir dès le début le niveau ou le mode de jeu ainsi de préciser qui joue contre qui.

❖ Script 2

```
Terminal  
Fichier Éditer Affichage Terminal Onglets Aide  
benhmali@inf-48-13:~/Bureau/APP-Projet-Final/Projet Final$ python3 SQRIP2.py  
script pour jouer 800 partie CPU contre lui meme dans le mode hard  
CPU1_hard : 0 - 800 : CPU2_hard  
benhmali@inf-48-13:~/Bureau/APP-Projet-Final/Projet Final$
```

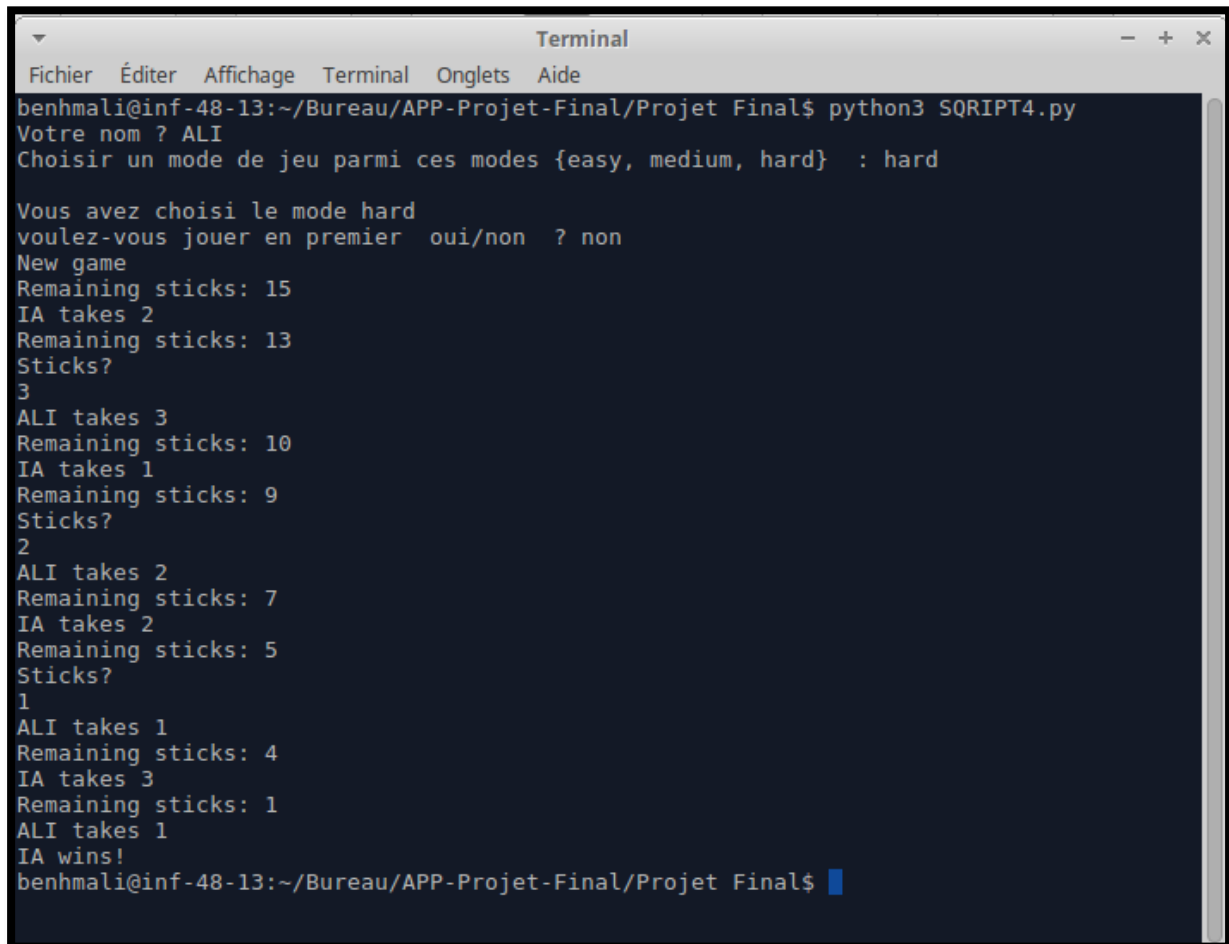
❖ Script 3

```
Terminal  
Fichier Éditer Affichage Terminal Onglets Aide  
benhmali@inf-48-13:~/Bureau/APP-Projet-Final/Projet Final$ python3 SQRIP3.py  
script pour jouer 800 partie CPU contre lui meme dans les deffirent modes  
CPU5_easy : 404 - 396 : CPU6_easy  
CPU5_easy : 132 - 668 : CPU4_medium  
CPU5_easy : 0 - 800 : CPU2_hard  
CPU3_medium : 388 - 412 : CPU4_medium  
CPU3_medium : 0 - 800 : CPU2_hard  
CPU1_hard : 1 - 799 : CPU2_hard  
benhmali@inf-48-13:~/Bureau/APP-Projet-Final/Projet Final$
```

VII. Jeu final

l'IA va savoir tous les astuces qu'il le faut pour gagner, donc il est désormais impossible de gagner, vu qu'il va jouer de tel sorte qu'il va nous laisse que 5 bâtons.

❖ Script 4

A screenshot of a terminal window titled "Terminal" with a menu bar containing "Fichier", "Éditer", "Affichage", "Terminal", "Onglets", and "Aide". The terminal shows the execution of a Python script named "SQRIPT4.py". The user "benhmali@inf-48-13" runs the script from the directory "~/Bureau/APP-Projet-Final/Projet Final". The script prompts for a name (ALI) and a game mode (hard). It then simulates a game of sticks, alternating between the user (ALI) and an AI player. The game ends with "IA wins!".

```
benhmali@inf-48-13:~/Bureau/APP-Projet-Final/Projet Final$ python3 SQRIPT4.py
Votre nom ? ALI
Choisir un mode de jeu parmi ces modes {easy, medium, hard} : hard

Vous avez choisi le mode hard
voulez-vous jouer en premier oui/non ? non
New game
Remaining sticks: 15
IA takes 2
Remaining sticks: 13
Sticks?
3
ALI takes 3
Remaining sticks: 10
IA takes 1
Remaining sticks: 9
Sticks?
2
ALI takes 2
Remaining sticks: 7
IA takes 2
Remaining sticks: 5
Sticks?
1
ALI takes 1
Remaining sticks: 4
IA takes 3
Remaining sticks: 1
ALI takes 1
IA wins!
benhmali@inf-48-13:~/Bureau/APP-Projet-Final/Projet Final$
```

VIII. Question optionnel (entrainement.py)

—Après avoir joué contre lui-même et évaluez le taux d'erreur en jouant un grand nombre de fois avec le réseau neuronal final on peut dire que le joueur qui commence en premier a perdu dans 0,0020696%.

—On a créé un algorithme pour calculer le taux d'erreur qui effectue 1 000 000 parties avec toujours la même intelligence artificielle qui commence.

— Sur ces 1 000 000 parties, l'intelligence artificielle ne gagne que 823 447 parties. On ne peut pas donc attendre jusqu'à ce que les gagnants de la part de l'intelligence artificielle commencent.