TP Bandits Manchots

BARREYRE Alexis – FAUCHERY Benoit – GOJARD Romain

# Exercice 1 : Le bandit manchot

1.

class Bandit:

    def \_\_init\_\_(self):

        self.avg = random.gauss(0, 1)

2.

def play(self):

        return random.gauss(self.avg, 1)

3.

from classes import Bandit

import matplotlib.pyplot as plt

bandit1, bandit2, bandit3 = Bandit(), Bandit(), Bandit()

points1, points2, points3 = [], [], []

for i in range(1000):

    value1, value2, value3 = bandit1.play(), bandit2.play(), bandit3.play()

    points1.append(value1) # affichage matplotlib

    points2.append(value2)

    points3.append(value3)

    print(value1, value2, value3)

# Exercice 2 : Le Ban-10

1.

class BanDix:

    def \_\_init\_\_(self):

        self.tab = []

        for i in range(10):

            self.tab.append(Bandit())

2.

def \_\_init\_\_(self):

        self.tab = []

        maxAvg = 0

        maxBanditIndex = 0

        for i in range(10):

            newBandit = Bandit()

            self.tab.append(newBandit)

            if(newBandit.avg > maxAvg):

                maxAvg = newBandit.avg

                maxBanditIndex = i

        self.banditMaxAvg = maxBanditIndex

3.

def play(self, arm\_number):

        if arm\_number > 9 or arm\_number < 0:

            raise ValueError("Valeur impossible, erreur")

        else:

            return self.tab[arm\_number].play()

# Exercice 3 : Algorithme ε-greedy

1.

class GreedyPlayer:

    def \_\_init\_\_(self, n, eps):

        self.n = n

        self.eps = eps

2.

self.action\_values = [0] \* 10

      self.eval\_count = [0] \* 10

3 & 4.

def get\_action(self):

        explore = random.random() < self.eps

        return explore

5.

def \_greedy\_action(self):

        best\_actions = []

        highest\_value = 0

        for i in self.action\_values:

            if(self.action\_values[i]>highest\_value):

                best\_actions = []

                best\_actions.append(i)

            elif(self.action\_values[i]==highest\_value):

                best\_actions.append(i)

        return random.choice(best\_actions)

6.

def \_random\_action(self):

        return random.choice(self.action\_values)

7.

if explore < self.eps:

            self.\_random\_action()

        else:

            self.\_greedy\_action()

8.

def reward(self, action, reward):

        self.action\_values[action] += reward

        self.eval\_count[action] += 1

9. & 10

ban10 = BanDix()

greedy = GreedyPlayer(0.1)

for i in range(1000):

    action = greedy.get\_action()

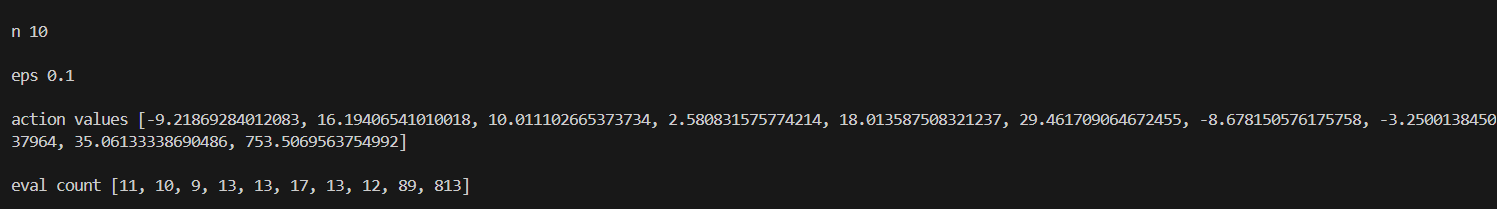
    reward = ban10.play(action)

    greedy.reward(action, reward)

    ban10.\_\_str\_\_()

    greedy.\_\_str\_\_()

11.



# Exercice 4 : Graphiques simples

1.

2.

3.

4.

5.

6.