TP 2 : Programmation des réseaux de Neurones

Objectifs :

Apprendre à développer un programme utilisant les réseaux de Neurones Artificiels ;

Apprendre à utiliser Scikit Learn

1. Programmation des réseaux de Neurones

Pour développer le programme de Réseaux de Neurones Artificiels, on va utiliser un Dataset (X,y) de 100 lignes sachant que X est un vecteur de deux composantes x1 et x2 et y est la fonction objectif.

1. On va donc commencer par générer cette Dataset en utilisant make\_blobs de sclearn.datasets.

Import numpy as np

Import matplotlib.pyplot as plt

From sclearn.datasets import make\_blobs

X, y = make\_blobs( n\_samples = 100, n\_features = 2, centers = 2, random\_state = 0)

y = reshape((y.shape[0], 1))

print ( ‘ dimension de X : ‘, X.shape)

print(‘ dimension de y :’, y.shape)

plt.scatter(X[ :,0], X[ :,1], c=y, cmap=’summer’)

plt.show()

Exécuter ce programme. Expliquer son fonctionnement et ces résultats.

1. Initialisation

Saisir le programme suivant :

def initialisation(X):

W = np.random.randn(X.shape[1],1)

b = np.random.randn(1)

return(W, b)

W, b=initialisation(X)

Exécuter le programme, expliquer son fonctionnement et donner W et b.

1. Modèle

Saisir le programme suivant :

def model(X, W, b):

z=np.dot(X,W)+b

A= 1/(1+np.exp(-z))

return A

A= model(X, W, b)

A.shape

Exécuter le programme. Quelle est la dimension de A ? pourquoi ?

1. La fonction Coût

Saisir le programme suivant :

def log\_loss (A,y):

cout = (1/len(y))\*np.sum(-y\*np.log(A)-(1-y)\*np.log(1-A))

return cout

Cout= log\_loss(A,y)

Cout

Exécuter le programme. Quelle est la valeur de Cout ?

1. Calcul des gradients

Saisir le programme suivant :

def gradients (A, X, y):

dW = (1 / len(y))\* np.dot(X.T, A-y)

db = (1/len(y)) \* np.sum(A-y)

return (dW, db)

Exécuter le programme. Donner les dimensions de dW et db.

1. Mise à jour des paramètres

Saisir le programme suivant :

def update (W, b, dW, db, alpha):

W = W - alpha\*dW

b = b - alpha\*db

return (W, b)

Comment appelle-t-on alpha ? Quel est son rôle ? Quelles sont les valeurs de W et b ?

Rassembler toutes ces fonctions pour créer une seule fonction « artificial\_neurones » dont les arguments sont A, X, y, alpha et n\_iter (nombre d’itérations).

Au début, on fixe alpha = 0 .1 et n\_iter = 100 (pour effectuer des itérations, utiliser l’instruction

for i in range (n\_iter) : ). Implanter le programme,

Visualiser le cout (pour cela, ajouter une liste cout=[] et utiliser l’instruction cout.append(log\_loss(A, y)) à l’intérieur de la boucle d’itération)

Tracer la fonction Coût. Qu’est ce que vous pouvez en déduire.

Maintenant que vous avez un modèle, il faut prédire des éléments. Pour cela, vous allez créer une fonction de prédiction

def predict(X, W, b)

A=model(X, W, b)

return A>=0.5

Que fait cette fonction ?

Ajouter ces lignes à la fin de votre programme et calculer « accuracy »

From sklearn.metrics import accuracy\_score

y\_pred = predict(X, W,b)

print(accuracy\_score(y , y\_pred))

Exécuter ces lignes, que vaut accuracy ? interpréter

Maintenant, on va créer de nouveaux exemples : X\_nouveau={(2,1) ; (2, 5) ; (3, 1.5) ; (2.5, 1.5)}

Ecrire le programme suivant :

X\_nouveau = np.array ([[2,1] , [2,5] , [3,1.5] , [2.5, 1.5]])

Pour s’assurer que vous avez crée ces exemples, il faut voir sur le graphique et mettre ces nouveaux points en rouge.

plt.scatter (X[ : , 0], X[ : , 1], c = y, cmap = ’summer’)

plt.scatter (X\_nouveau[ : , 0] , X\_nouveau[ : , 1] , c = ’red’)

Exécuter le programme. Vérifier que les points ajoutés figurent sur le graphique.

Prédire la classe de ces nouveaux exemples en utilisant le modèle développé.

predict(X\_nouveau, W, b)

Exécuter l’instruction. Est-ce que le résultat obtenu est correct ? Expliquer.

On veut tracer la frontière des décision qui correspond à Z=0, c’est-à-dire l’ensemble des points pour lesquels (W[0]\*X0 + W[1]\*X1 + b= 0) donc (X1= (-W[0]\*X0 - b)/W[1])

Ajouter les lignes suivantes :

X0=np.linspace(-1, 4, 100)

X1= (- W[0] \* X0 - b) / W[1]

plt.plot(X0 , X1, c=’orange’, lw = 3)

Exécuter le programme, compter le nombre d’erreur et interpréter.