

TRABAJO PRACTICO N°4

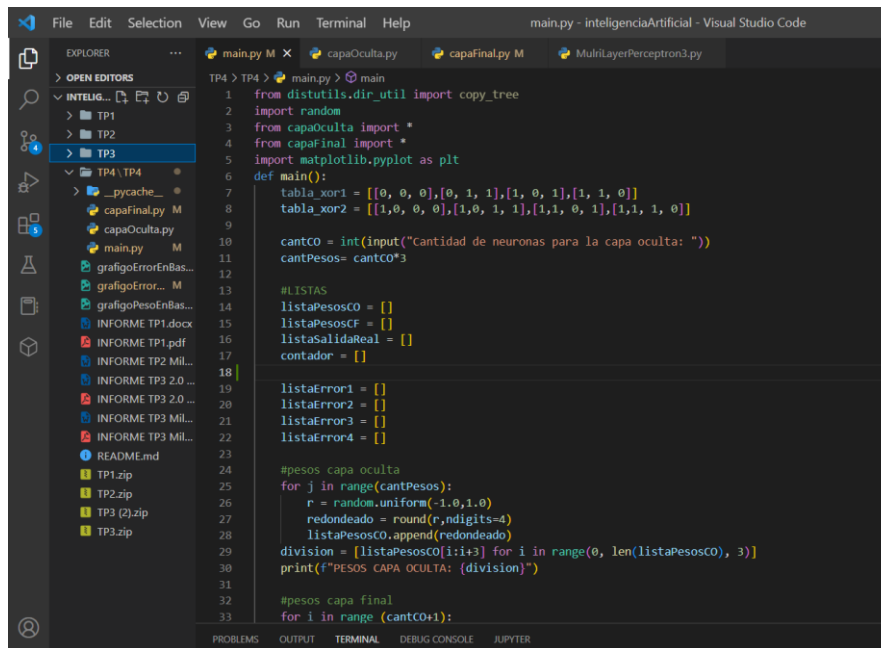
PERCEPTRÓN MULTICAPA CON “N” NEURONAS EN LA CAPA OCULTA

Mi trabajo cuentas con 3 archivos .py:

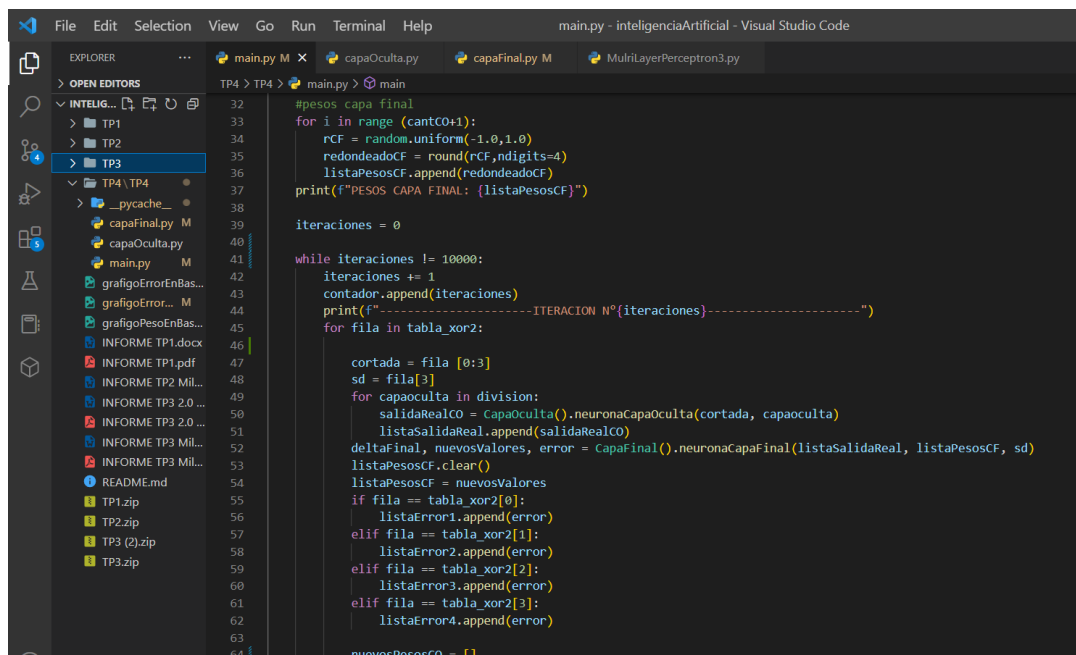
- main.py
- capaOculta.py
- capaFinal.py

Fotos del código:

main.py



```
1 from distutils.dir_util import copy_tree
2 import random
3 from capaOculta import *
4 from capaFinal import *
5 import matplotlib.pyplot as plt
6 def main():
7     tabla_xor1 = [[0, 0, 0],[0, 1, 1],[1, 0, 1],[1, 1, 0]]
8     tabla_xor2 = [[1,0, 0, 0],[1,0, 1, 1],[1,1, 0, 1],[1,1, 1, 0]]
9
10    cantCO = int(input("Cantidad de neuronas para la capa oculta: "))
11    cantPesos= cantCO*3
12
13    #LISTAS
14    listaPesosCO = []
15    listaPesosCF = []
16    listaSalidaReal = []
17    contador = []
18
19    listaError1 = []
20    listaError2 = []
21    listaError3 = []
22    listaError4 = []
23
24    #pesos capa oculta
25    for j in range(cantPesos):
26        r = random.uniform(-1.0,1.0)
27        redondeado = round(r,ndigits=4)
28        listaPesosCO.append(redondeado)
29    division = [listaPesosCO[i:i+3] for i in range(0, len(listaPesosCO), 3)]
30    print(f"PESOS CAPA OCULTA: {division}")
31
32    #pesos capa final
33    for i in range (cantCO+1):
```



```
34    #pesos capa final
35    for i in range (cantCO+1):
36        rCF = random.uniform(-1.0,1.0)
37        redondeadoCF = round(rCF,ndigits=4)
38        listaPesosCF.append(redondeadoCF)
39    print(f"PESOS CAPA FINAL: {listaPesosCF}")
40
41    iteraciones = 0
42
43    while iteraciones != 10000:
44        iteraciones += 1
45        contador.append(iteraciones)
46        print(f"-----ITERACION N°{iteraciones}-----")
47        for fila in tabla_xor2:
48
49            cortada = fila [0:3]
50            sd = fila[3]
51            for capaoculta in division:
52                salidaRealCO = CapaOculta().neuronaCapaOculta(cortada, capaoculta)
53                listaSalidaReal.append(salidaRealCO)
54            deltaFinal, nuevosValores, error = CapaFinal().neuronaCapaFinal(listaSalidaReal, listaPesosCF, sd)
55            listaPesosCF.clear()
56            listaPesosCF = nuevosValores
57            if fila == tabla_xor2[0]:
58                listaError1.append(error)
59            elif fila == tabla_xor2[1]:
60                listaError2.append(error)
61            elif fila == tabla_xor2[2]:
62                listaError3.append(error)
63            elif fila == tabla_xor2[3]:
64                listaError4.append(error)
65
66    nuevosPesosCO = []
```

```

main.py - inteligenciaArtificial - Visual Studio Code

EXPLORER
main.py M
capaOculta.py
capaFinal.py M
MultiLayerPerceptron3.py

OPEN EDITORS
TP4 > TP4 > main.py > main

30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88

def main():
    listaError1.append(error)
    elif fila == tabla_xor2[1]:
        listaError2.append(error)
    elif fila == tabla_xor2[2]:
        listaError3.append(error)
    elif fila == tabla_xor2[3]:
        listaError4.append(error)

    nuevosPesosCO = []
    lr=0.5
    for peso in range(len(division)):
        Soc_general = listaSalidaReal[peso]*(1-listaSalidaReal[peso])*deltaFinal
        for c in range(len(cortada)):
            delta_general = lr*cortada[c]*Soc_general
            w_general = division[peso][c] + delta_general
            nuevosPesosCO.append(w_general)
        listaSalidaReal.clear()
        division.clear()
        division = [nuevosPesosCO[i:i+3] for i in range(0, len(nuevosPesosCO), 3)]

    fig, ax = plt.subplots()
    ax.plot(contador, listaError1, label="Error fila 1")
    ax.plot(contador, listaError2, label="Error fila 2")
    ax.plot(contador, listaError3, label="Error fila 3")
    ax.plot(contador, listaError4, label="Error fila 4")
    ax.set_xlabel("ITERACIONES", fontdict = {'fontsize':14, 'fontweight':'bold', 'color':'tab:green'})
    ax.set_ylabel("ERRORES", fontdict = {'fontsize':14, 'fontweight':'bold', 'color':'tab:blue'})
    plt.title("Error en base de las iteracion")
    plt.legend()
    plt.savefig("graficoErroresEnBaseIteracion")
    plt.show()

```

capaOculta.py

```

View Go Run Terminal Help
capaOculta.py - inteligenciaArtificial - Visual Studio Code

main.py M
capaOculta.py X
capaFinal.py M

TP4 > TP4 > capaOculta.py > CapaOculta > neuronaCapaOculta

1 import math
2 import numpy as np
3 class CapaOculta():
4     def neuronaCapaOculta(self, filaXOR, listaPesos):
5         x = np.multiply(filaXOR, listaPesos)
6         sumatoria = sum(x)
7         salidaReal = 1 / (1 + (math.exp(-sumatoria)))
8         return salidaReal
9
10
11

```

capaFinal.py

```

File Edit Selection View Go Run Terminal Help
capaFinal.py - inteligenciaArtificial - Visual Studio Code

EXPLORER
main.py M
capaOculta.py
capaFinal.py M X

OPEN EDITORS
TP4 > TP4 > capaFinal.py > CapaFinal > neuronaCapaFinal

1 import math
2 import numpy as np
3 class CapaFinal():
4     def neuronaCapaFinal(self, salidaRealCO, listaPesosCF, sd):
5         # print(f"-----NEURONA 4-----\n")
6         # print(f"PESOS ULTIMA NEURONA: {listaPesosCF}")
7         lr = 0.5
8         vias = 1
9         listaPesosCortada = listaPesosCF[1:]
10        listax = []
11        # print(f"Lista pesos cortada capa final:{listaPesosCF}")
12
13        x = listaPesosCF[0]* vias
14        listax.append(x)
15        for i in range(len(listaPesosCortada)):
16            #print(listaPesosCortada[i])
17            x2 = (listaPesosCortada[i]*salidaRealCO[i])
18            listax.append(x2)
19        sumatoria = sum(listax)
20        salidaReal = 1 / (1 + (math.exp(-sumatoria)))
21        error = sd - salidaReal
22        deltaFinal = salidaReal*(1-salidaReal)*error
23
24        nuevosPesos = []
25        lr = 0.5
26        vias = 1
27        listaPesosCortada = listaPesosCF[1:]
28
29        delta_w1 = lr * vias * deltaFinal
30        w1 = listaPesosCF[0] + delta_w1
31        nuevosPesos.append(w1)
32
33        for i in range(len(salidaRealCO)):
34            delta_w2 = lr*salidaRealCO[i]*deltaFinal
35            w2 = listaPesosCortada[i] + delta_w2
36            nuevosPesos.append(w2)
37        print("\nNeurona 4")
38        print(f"SALIDA REAL CAPA FINAL: {salidaReal}")
39
40        return deltaFinal, nuevosPesos, error

```

```
PS D:\Mili\Documentos\facultad\CURSADO\4º AÑO\segundo semestre\inteligenciaArtificial>
Cantidad de neuronas para la capa oculta: 8
```

Seleccione 8 neuronas y 10000 iteraciones:

Resultado de la salida real de la última neurona en la iteración 10000

```
-----ITERACION N°10000-----

Neurona 4
SALIDA REAL CAPA FINAL: 0.018017158750843806

Neurona 4
SALIDA REAL CAPA FINAL: 0.9825517756953905

Neurona 4
SALIDA REAL CAPA FINAL: 0.9829279329074444

Neurona 4
SALIDA REAL CAPA FINAL: 0.021155934309730392
```

Grafico de errores:

