­­­

|  |
| --- |
| Asignatura: |
| Inteligencia Artificial II |

|  |
| --- |
| Título del documento: |
| **Laboratorio 02** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Preparado por: |  |  |  |  |
| Nombre |  | Gonzalo Monedero  Nicolás Salgado  Iñigo Vázquez  Marcos Molina |  | 14/03/2022 | |
|  |  |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nombre de fichero: |  | Fecha: |  | Edición: |  | Página: |
| **Memoria\_LAB02\_GB01** |  | **14/03/2022** |  | **7** |  | **01/67** |

# Índice

[1 Índice 2](#_Toc98191110)

[2 Introducción 4](#_Toc98191111)

[3 Práctica 1 5](#_Toc98191112)

[3.1 Introducción 5](#_Toc98191113)

[3.2 Estudio de valores. 6](#_Toc98191114)

[3.2.1 Estudio umbral y coeficiente para AND 6](#_Toc98191115)

[3.2.2 Estudio periodo para AND 7](#_Toc98191116)

[3.2.3 Valores para la red AND 7](#_Toc98191117)

[3.2.4 Estudio umbral y coeficiente para XOR 8](#_Toc98191118)

[3.2.5 Estudio periodo para XOR 8](#_Toc98191119)

[3.2.6 Valores para la red XOR 9](#_Toc98191120)

[3.3 Cuestiones 9](#_Toc98191121)

[3.3.1 Función AND 9](#_Toc98191122)

[3.3.1.1 Tabla de iteraciones 9](#_Toc98191123)

[3.3.1.2 Ecuación del hiperplano 12](#_Toc98191124)

[3.3.2 Función XOR 13](#_Toc98191125)

[3.3.2.1 Tabla de iteraciones 13](#_Toc98191126)

[3.3.2.2 Ecuación del hiperplano Gonzalo 15](#_Toc98191127)

[4 Práctica 2 16](#_Toc98191128)

[4.1 Introducción 16](#_Toc98191129)

[4.2 Cuestiones 18](#_Toc98191130)

[4.2.1 Función RELU 18](#_Toc98191131)

[4.2.1.1 Tablas RELU 18](#_Toc98191132)

[4.2.1.2 Graficas RELU 27](#_Toc98191133)

[4.2.2 Función SIGMOIDEA 28](#_Toc98191134)

[4.2.2.1 Tablas SIGMOIDEA 28](#_Toc98191135)

[4.2.2.2 Graficas SIGMOIDEA 36](#_Toc98191136)

[4.2.3 Función RELU y SIGMOIDEA 37](#_Toc98191137)

[4.2.3.1 Tablas RELU y SIGMOIDEA 37](#_Toc98191138)

[4.2.3.2 Graficas RELU y SIGMOIDEA 45](#_Toc98191139)

[4.2.4 Función SIGMOIDEA y RELU 46](#_Toc98191140)

[4.2.4.1 Tablas SIGMOIDEA y RELU 46](#_Toc98191141)

[4.2.4.2 Grafica RELU y SIGMOIDEA 54](#_Toc98191142)

[4.2.5 Conclusión de mejor función 55](#_Toc98191143)

[5 Práctica 3 57](#_Toc98191144)

[5.1 Introducción 57](#_Toc98191145)

[5.2 Cuestiones 57](#_Toc98191146)

[5.2.1 Proceso de categorización y normalización de los datos 57](#_Toc98191147)

[5.2.2 Estudio de arquitectura 58](#_Toc98191148)

[58](#_Toc98191149)

[5.2.3 Error de la arquitectura 62](#_Toc98191150)

[5.2.4 Influencia de las variables 63](#_Toc98191151)

[6 Conclusión 66](#_Toc98191152)

[7 Bibliografía 67](#_Toc98191153)

# Introducción

En esta práctica realizaremos los problemas propuestos por el enunciado para, no solo quedarnos con la compresión teórica sino también acercarnos al campo practico.

Es así como aprenderemos a hacer un perceptrón, que realice las operaciones lógicas de AND y XOR. Realizaremos esta última con un perceptrón multicapa y la librería keras. Y por último crearemos un perceptrón multicapa no lineal que clasifique de forma correcta casos de cáncer en base al dataset proporcionado por el enunciado.

Así es como comprenderemos y asimilaremos los contenidos vistos en las clases teóricas aportando resoluciones que demuestren nuestros conocimientos en esta área.

# Práctica 1

## Introducción

La primera práctica de este laboratorio es la elaboración de un perceptrón que nos permita resolver la función AND y XOR. Este perceptrón será entrenado mediante la ley de Hebb y tendrá pesos comprendidos entre .

Como bien hemos visto en clase, para realizar correctamente un perceptrón hay que seguir una serie de pasos de forma estricta.

1. **Generación de un vector de pesos aleatorios.**



1. **Introducción de entrada.**



1. **Cálculo del error.**



1. **Si el error es = 0, se vuelve al paso 2 con una nueva entrada.**
2. Imagen que contiene objeto, reloj, señal

   Descripción generada automáticamente**Aplicación de la Ley de Hebb.**

1. **Actualización de pesos.**



1. **Repetir desde el paso dos con una nueva entrada.**

Estos pasos los seguimos constantemente de manera iterativa tantas veces como iteraciones realicemos.

## Estudio de valores.

Al principio planteamos un estudio secuencial donde primero averiguaríamos el umbral que mayor tasa de aciertos obtenía, luego proseguíamos con el coeficiente y por ultimo el periodo, pero nos dimos cuenta de que no había una razón por la que primero deberíamos estudiar el umbral y luego el coeficiente de aprendizaje o viceversa. Es por ello por lo que decidimos plantear una nueva metodología de estudio que combina umbral y coeficiente para hacer el estudio a la par e intentar perder la menos información posible. Como en la práctica anterior hemos diseñado un sistema que automatiza la pruebas que queremos realizar, de este modo solo tenemos que pasarle una lista por cada parámetro de lo que queremos estudiar.

### Chart Description automatically generatedEstudio umbral y coeficiente para AND

Cada curva en esta grafica representa un estudio del coeficiente con un umbral diferente que empieza en 0 y termina en 1. Como podemos observar en el intervalo [0 , 0.2] del eje de abscisas todas la curvas se disparan hacia una precisión del 100% , lo único que vemos al aumentar el umbral es cuánto dura esta precisión a lo largo del eje de a abscisas, es decir a mayor es el umbral más larga es la tendencia asintótica es por ello que solo nos interesa quedarnos con un valor dentro del intervalo mencionado ya que es donde la mayoría de curvas coindicen es por ello que cogeremos el punto medio que es donde todas llegan a una precisión del 100% este punto siendo el 0.1, este valor dado nuestro estudio es el óptimo para el coeficiente de aprendizaje.

Chart, line chart

Description automatically generated Para el umbral como ya hemos mencionado, generalmente hablando a mayor es el umbral, mayor es la tendencia asintótica. Para asegurarnos hemos aplicado el mismo método, pero esta vez cada curva indica un coeficiente de aprendizaje diferente.

Como podemos observar a mayor es el umbral más permisivo es a la hora de elegir nuestro coeficiente, pero debemos tener en cuenta la gráfica anterior ya que si nos conformamos con los valores (coeficiente y umbral) más alto perdemos precisión es por eso por lo que nos figaremos en el intervalo [0.4 , 0.6] ya que es donde los coeficientes de aprendizaje más bajos se estabilizan y como hemos visto el 0.1 es el coeficiente optimo es por ello que aligeremos para el umbral el punto medio siendo este 0.5.

### Shape, rectangle Description automatically generatedEstudio periodo para AND

En el caso del periodo hemos usado los valores de coeficiente y umbral mencionados anteriormente para el estudio. Aplicando la misma metodología que en la anterior practica entrenamos para cada periodo la red y vemos la precisión de dicha red, como podemos observar con solo 20 iteraciones somos capaces de obtener un 100% de aciertos.

### Valores para la red AND

* Coeficiente de aprendizaje: **0.1**
* Umbral: **0.5**
* Periodo: **20**

### Estudio umbral y coeficiente para XOR

Shape

Description automatically generatedPara el estudio de la puerta XOR hemos usado la misma metodología que para AND y cómo podemos ver curiosamente a partir del 0.1 como coeficiente se estabiliza todo entre [0.75 , 0.8] de media de aciertos dependiendo este rango del umbral. Es por ello por lo que usaremos 0.1 como coeficiente de aprendizaje para XOR.

A picture containing chart

Description automatically generatedPara ver que umbral nos acerca más al 0.8 de precisión implementamos la misma gráfica, pero invertida es decir ahora cada curva representa un coeficiente diferente. Como podemos ver otra vez el intervalo [0.4 , 0.6] es el que nos optimiza la gran mayoría de curvas siendo las más importante la de 0.1 de coeficiente. En el punto 0.5 de umbral obtenemos un 0.8 de precisión es por lo que nos conformaremos con este valor ya que no podemos obtener un 100% de precisión debido a que no es un problema que pueda resolver linealmente, al menos con una sola recta.

### Estudio periodo para XOR

Shape, rectangle

Description automatically generatedComo podemos ver a partir de 20 iteraciones obtenemos una tendencia asintótica así que usaremos 20 iteraciones para entrenar a nuestra red. Ya que maximiza la precisión sin aumentar la complejidad computacional.

### Valores para la red XOR

* Coeficiente de aprendizaje: **0.1**
* Umbral: **0.5**
* Periodo: **20**

Curiosamente son los mismos que para nuestra red AND.

## Cuestiones

*Elabora una memoria de la práctica con una tabla para cada caso (AND & XOR). Discútelas según lo visto en teoría- Escribe de forma explícita la ecuación del hiperplano que se ha generado en cada uno.*

### Función AND

#### Tabla de iteraciones

A continuación, se presenta la tabla de las iteraciones que realiza nuestro perceptrón durante la validación. Todos los pesos de la tabla se generan aleatoriamente y vemos como en cada iteración se ajustan automáticamente. En este caso observamos la tabla de la función AND.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Iteración | Entradas | Pesos Iniciales | Esperado | Obtenido | Error | Pesos Finales |
| 1 | **[0, 0]** | **[-0.328,-0.255]** | **0** | **0** | **0** | **[-0.328 -0.255]** |
| 1 | [0, 1] | [-0.328 -0.255] | 0 | 0 | 0 | [-0.328 -0.255] |
| 1 | **[1, 0]** | **[-0.328 -0.255]** | **0** | **0** | **0** | **[-0.328 -0.255]** |
| 1 | [1, 1] | [-0.328 -0.255] | 1 | 0 | 1 | [-0.228 -0.155] |
| 2 | **[0, 0]** | **[-0.228 -0.155]** | **0** | **0** | **0** | **[-0.228 -0.155]** |
| 2 | [0, 1] | [-0.228 -0.155] | 0 | 0 | 0 | [-0.228 -0.155] |
| 2 | **[1, 0]** | **[-0.228 -0.155]** | **0** | **0** | **0** | **[-0.228 -0.155]** |
| 2 | [1, 1] | [-0.228 -0.155] | 1 | 0 | 1 | [-0.128 -0.055] |
| 3 | **[0, 0]** | **[-0.128 -0.055]** | **0** | **0** | **0** | **[-0.128 -0.055]** |
| 3 | [0, 1] | [-0.128 -0.055] | 0 | 0 | 0 | [-0.128 -0.055] |
| 3 | **[1, 0]** | **[-0.128 -0.055]** | **0** | **0** | **0** | **[-0.128 -0.055]** |
| 3 | [1, 1] | [-0.128 -0.055] | 1 | 0 | 1 | [-0.028 0.045] |
| 4 | **[0, 0]** | **[-0.028 0.045]** | **0** | **0** | **0** | **[-0.028 0.045]** |
| 4 | [0, 1] | [-0.028 0.045] | 0 | 0 | 0 | [-0.028 0.045] |
| 4 | **[1, 0]** | **[-0.028 0.045]** | **0** | **0** | **0** | **[-0.028 0.045]** |
| 4 | [1, 1] | [-0.028 0.045] | 1 | 0 | 1 | [0.072 0.145] |
| 5 | **[0, 0]** | **[0.072 0.145]** | **0** | **0** | **0** | **[0.072 0.145]** |
| 5 | [0, 1] | [0.072 0.145] | 0 | 0 | 0 | [0.072 0.145] |
| 5 | **[1, 0]** | **[0.072 0.145]** | **0** | **0** | **0** | **[0.072 0.145]** |
| 5 | [1, 1] | [0.072 0.145] | 1 | 0 | 1 | [0.172 0.245] |
| 6 | **[0, 0]** | **[0.172 0.245]** | **0** | **0** | **0** | **[0.172 0.245]** |
| 6 | [0, 1] | [0.172 0.245] | 0 | 0 | 0 | [0.172 0.245] |
| 6 | **[1, 0]** | **[0.172 0.245]** | **0** | **0** | **0** | **[0.172 0.245]** |
| 6 | [1, 1] | [0.172 0.245] | 1 | 0 | 1 | [0.272 0.345] |
| 7 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 7 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 7 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 7 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 8 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 8 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 8 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 8 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 9 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 9 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 9 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 9 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 10 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 10 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 10 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 10 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 11 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 11 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 11 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 11 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 12 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 12 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 12 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 12 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 13 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 13 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 13 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 13 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 14 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 14 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 14 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 14 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 15 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 15 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 15 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 15 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 16 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 16 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 16 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 16 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 17 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 17 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 17 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 17 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 18 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 18 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 18 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 18 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 19 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 19 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 19 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 19 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |
| 20 | **[0, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 20 | [0, 1] | [0.272 0.345] | 0 | 0 | 0 | [0.272 0.345] |
| 20 | **[1, 0]** | **[0.272 0.345]** | **0** | **0** | **0** | **[0.272 0.345]** |
| 20 | [1, 1] | [0.272 0.345] | 1 | 1 | 0 | [0.272 0.345] |

Se puede apreciar que a medida que pasan las iteraciones los pesos se mueven menos hasta que llega a un punto donde ya no se mueve eso se debe a que como tiene un acierto del 100% ya no es necesario el moviente de dichos pesos, da igual cuantas iteraciones pasen los pesos no se mueven ya que clasifica todo correctamente esto se puede apreciar desde la iteración 7 de la tabla superior ya que a partir de ese punto no hay error y por lo tanto no se puede aprender.

#### Ecuación del hiperplano

A continuación, podemos observar la ecuación del Hiperplano de la función AND. Esta función la calculamos mediante la fórmula:

Diagram

Description automatically generated

Nos permite ver gráficamente la división que se realiza entre los 4 distintos valores. Podemos ver como esta recta divide los valores verdes y el rojo de manera correcta, mostrándonos, así como nuestro perceptrón está bien entrenado.

Line chart

Description automatically generated with medium confidence

### Función XOR

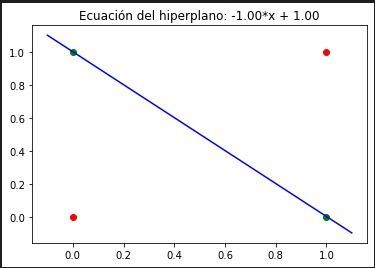
#### Tabla de iteraciones

A continuación, se presenta la tabla de las iteraciones que realiza nuestro perceptrón durante la validación. Todos los pesos de la tabla se generan aleatoriamente y vemos como en cada iteración se ajustan automáticamente. En este caso observamos la tabla de la función XOR.

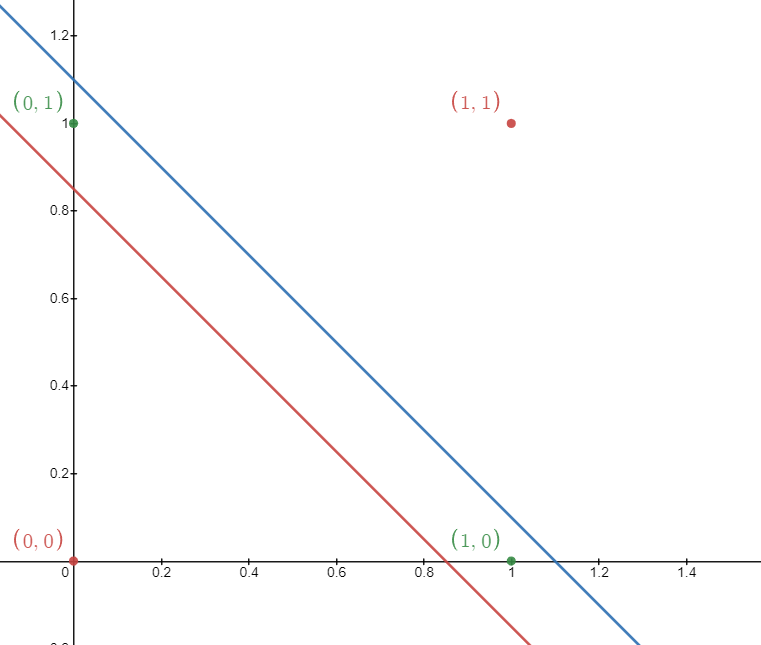
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Iteración | Entradas | Pesos Iniciales | Esperado | Obtenido | Error | Pesos Finales |
| 1 | [0, 0] | [0.673, 0.659] | 0 | 0 | 0 | [0.673 0.659] |
| 1 | [0, 1] | [0.673 0.659] | 1 | 1 | 0 | [0.673 0.659] |
| 1 | [1, 0] | [0.673 0.659] | 1 | 1 | 0 | [0.673 0.659] |
| 1 | [1, 1] | [0.673 0.659] | 0 | 1 | 1 | [0.573 0.559] |
| 2 | [0, 0] | [0.573 0.559] | 0 | 0 | 0 | [0.573 0.559] |
| 2 | [0, 1] | [0.573 0.559] | 1 | 1 | 0 | [0.573 0.559] |
| 2 | [1, 0] | [0.573 0.559] | 1 | 1 | 0 | [0.573 0.559] |
| 2 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 3 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 3 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 3 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 3 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 4 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 4 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 4 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 4 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 5 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 5 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 5 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 5 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 6 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 6 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 6 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 6 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 7 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 7 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 7 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 7 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 8 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 8 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 8 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 8 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 9 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 9 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 9 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 9 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 10 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 10 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 10 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 10 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 11 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 11 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 11 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 11 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 12 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 12 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 12 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 12 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 13 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 13 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 13 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 13 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 14 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 14 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 14 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 14 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 15 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 15 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 15 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 15 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 16 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 16 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 16 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 16 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 17 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 17 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 17 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 17 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 18 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 18 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 18 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 18 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 19 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 19 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 19 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 19 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |
| 20 | [0, 0] | [0.473 0.459] | 0 | 0 | 0 | [0.473 0.459] |
| 20 | [0, 1] | [0.473 0.459] | 1 | 0 | 1 | [0.473 0.559] |
| 20 | [1, 0] | [0.473 0.559] | 1 | 0 | 1 | [0.573 0.559] |
| 20 | [1, 1] | [0.573 0.559] | 0 | 1 | 1 | [0.473 0.459] |

En este caso podemos ver como el problema no es linealmente separable nunca termina de aprender porque nunca encuentra la recta que divida los puntos de manera correcta y es por ello que si lo dejemos iterar infinitamente siempre se modificaran sus pesos ya que nunca será capaz de clasificar todas las entradas de manera correcta.

#### Ecuación del hiperplano Gonzalo

Al igual que en la función del AND, en este caso calcularemos la recta en el plano bidimensional que separa a los puntos del XOR.

Como podemos ver en el dibujo, es imposible realizar la función XOR con un perceptrón, ya que no existe ninguna recta en el hiperplano capaz de dividir los puntos rojos de los puntos verdes. Es por ello por lo que como bien hemos visto en las gráficas de apartados anteriores, nunca conseguiremos una salida correcta, y si la conseguimos, será de forma aleatoria.

Nótese que si pudiésemos usar dos rectas es decir dos perceptrones seriamos capaz de resolver linealmente este problema.

# Práctica 2

## Introducción

El perceptrón multicapa es un modelo feedfoward no lineal, entrenada de aprendizaje supervisado. Este es una variación del modelo Adaline con aprendizaje por minimización del MSE mediante Regla Delta. Sus principales aplicaciones son la clasificación, reconocimiento de señales y control adaptativo entre otros y cuenta con ventajas como: Capacidad de generalización y entrenada, produce en un runtime resultados muy rápidos.

Esta se llama multicapa ya que es una red neuronal que cuenta con al menos 3 entradas interconectadas al completo.

Los principales problemas de este modelo es saber el número de capas ocultas y el número de neuronas en cada una. Ya que vemos que en función de estas nuestro modelo puede sufrir de underfitting o al contrario de overfitting.

Para verificar que el número de neuronas en una capa oculta está bien empíricamente tenemos la siguiente formula:

Un reloj de aguja

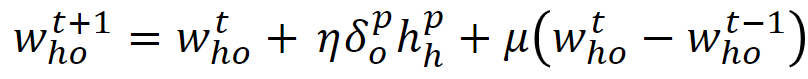
Descripción generada automáticamente con confianza baja

Donde M es el número de ejemplos del conjunto de entrenamiento, N es el número de neuronas de la capa de entrada y n es el número de neuronas en la capa intermedia.

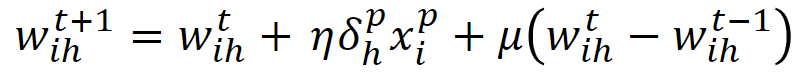
Este modelo se caracteriza por tener un conjunto de datos del cual sabemos la salida esperada, de este debemos separar el 20% para las pruebas, de esta forma verificaremos que no se ha producido overfitting. Cuando se llega a la salida debemos hacer Backpropagation (Regla Delta Generalizada), esto calcula el error producido por cada neurona, de tal forma que obtenemos el vector gradiente para poder aplicar a cada capa el descenso de gradiente. Su objetivo es encontrar el error mínimo.

Con lo cual para cada capa tenemos una ecuación de pesos:

Matriz de pesos oculta-salida:



Matriz de pesos entrada-oculta:



Por eso para realizar este modelo estos son los pasos para seguir:

1. Inicializar pesos con valores aleatorios
2. Presentar el vector de entrada
3. Calcular la salida de las neuronas de la capa oculta
4. Calcular la salida de la red (neuronas de la capa de salido)
5. Aplicar la Regla Delta Generalizada
6. Calcular la sensibilidad de las neuronas de la capa de salida
7. Calcular la sensibilidad de las neuronas de la capa oculta
8. Actualizar los pesos de la matriz de la capa oculta
9. Actualizar los pesos de la matriz de la capa de salida
10. Repetir desde 2 para todo el conjunto de entrenamiento
11. Calcular el Error
12. Repetir desde 2 si no se da la condición de parada

En este apartado de la práctica utilizamos keras, una biblioteca de redes neuronales la cual nos ayudara a automatizar los pasos previamente expuestos.

## Cuestiones

Para continuar debemos introducir dos funciones, la función de activación RELU y la Sigmoidea.

La función de activación RELU nos aporta al modelo una funcionalidad, en la que, los valores negativos que entran pasan a 0 y los positivos se mantienen igual.

Por otro lado, tenemos a la función sigmoidea, transforma los valores introducidos a una escala de (0,1), donde los valores altos tienden asintóticamente a 1 y los bajos a 0. El problema de esta es que satura y mata el gradiente que es lo que queremos para calcular el error. Gráfico, Gráfico de líneas

Descripción generada automáticamente Diagrama

Descripción generada automáticamente

### Función RELU

#### Tablas RELU

|  |  |  |  |
| --- | --- | --- | --- |
| Neuronas Capa Oculta | *Epochs* | *Salida* | *Error* |
| 2 | 525 | [0.] | 0 |
| 2 | 525 | [0.] | 1 |
| 2 | 525 | [1.] | 0 |
| 2 | 525 | [0.] | 0 |
| 3 | 525 | [0.] | 0 |
| 3 | 525 | [0.] | 1 |
| 3 | 525 | [1.] | 0 |
| 3 | 525 | [0.] | 0 |
| 4 | 525 | [0.] | 0 |
| 4 | 525 | [0.] | 1 |
| 4 | 525 | [0.] | 1 |
| 4 | 525 | [0.] | 0 |
| 5 | 525 | [0.] | 0 |
| 5 | 525 | [1.] | 0 |
| 5 | 525 | [0.] | 1 |
| 5 | 525 | [0.] | 0 |
| 6 | 525 | [0.] | 0 |
| 6 | 525 | [1.] | 0 |
| 6 | 525 | [0.] | 1 |
| 6 | 525 | [0.] | 0 |
| 7 | 525 | [0.] | 0 |
| 7 | 525 | [0.] | 1 |
| 7 | 525 | [0.] | 1 |
| 7 | 525 | [0.] | 0 |
| 8 | 525 | [0.] | 0 |
| 8 | 525 | [1.] | 0 |
| 8 | 525 | [1.] | 0 |
| 8 | 525 | [0.] | 0 |
| 9 | 525 | [0.] | 0 |
| 9 | 525 | [1.] | 0 |
| 9 | 525 | [0.] | 1 |
| 9 | 525 | [0.] | 0 |
| 10 | 525 | [0.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [0.] | 0 |
| 11 | 525 | [0.] | 0 |
| 11 | 525 | [1.] | 0 |
| 11 | 525 | [1.] | 0 |
| 11 | 525 | [0.] | 0 |
| 12 | 525 | [0.] | 0 |
| 12 | 525 | [0.] | 1 |
| 12 | 525 | [0.] | 1 |
| 12 | 525 | [0.] | 0 |
| 13 | 525 | [0.] | 0 |
| 13 | 525 | [1.] | 0 |
| 13 | 525 | [1.] | 0 |
| 13 | 525 | [0.] | 0 |
| 14 | 525 | [0.] | 0 |
| 14 | 525 | [0.] | 1 |
| 14 | 525 | [1.] | 0 |
| 14 | 525 | [0.] | 0 |
| 15 | 525 | [0.] | 0 |
| 15 | 525 | [0.] | 1 |
| 15 | 525 | [1.] | 0 |
| 15 | 525 | [0.] | 0 |
| 16 | 525 | [0.] | 0 |
| 16 | 525 | [0.] | 1 |
| 16 | 525 | [1.] | 0 |
| 16 | 525 | [0.] | 0 |
| 17 | 525 | [0.] | 0 |
| 17 | 525 | [1.] | 0 |
| 17 | 525 | [1.] | 0 |
| 17 | 525 | [0.] | 0 |
| 18 | 525 | [0.] | 0 |
| 18 | 525 | [1.] | 0 |
| 18 | 525 | [0.] | 1 |
| 18 | 525 | [0.] | 0 |
| 19 | 525 | [0.] | 0 |
| 19 | 525 | [0.] | 1 |
| 19 | 525 | [1.] | 0 |
| 19 | 525 | [0.] | 0 |
| 20 | 525 | [0.] | 0 |
| 20 | 525 | [1.] | 0 |
| 20 | 525 | [0.] | 1 |
| 20 | 525 | [0.] | 0 |
| 21 | 525 | [0.] | 0 |
| 21 | 525 | [1.] | 0 |
| 21 | 525 | [1.] | 0 |
| 21 | 525 | [0.] | 0 |
| 22 | 525 | [0.] | 0 |
| 22 | 525 | [1.] | 0 |
| 22 | 525 | [0.] | 1 |
| 22 | 525 | [0.] | 0 |
| 23 | 525 | [0.] | 0 |
| 23 | 525 | [1.] | 0 |
| 23 | 525 | [1.] | 0 |
| 23 | 525 | [0.] | 0 |
| 24 | 525 | [0.] | 0 |
| 24 | 525 | [0.] | 1 |
| 24 | 525 | [0.] | 1 |
| 24 | 525 | [0.] | 0 |
| 25 | 525 | [0.] | 0 |
| 25 | 525 | [1.] | 0 |
| 25 | 525 | [0.] | 1 |
| 25 | 525 | [0.] | 0 |
| 26 | 525 | [0.] | 0 |
| 26 | 525 | [1.] | 0 |
| 26 | 525 | [0.] | 1 |
| 26 | 525 | [0.] | 0 |
| 27 | 525 | [0.] | 0 |
| 27 | 525 | [1.] | 0 |
| 27 | 525 | [0.] | 1 |
| 27 | 525 | [0.] | 0 |
| 28 | 525 | [0.] | 0 |
| 28 | 525 | [1.] | 0 |
| 28 | 525 | [1.] | 0 |
| 28 | 525 | [0.] | 0 |
| 29 | 525 | [0.] | 0 |
| 29 | 525 | [0.] | 1 |
| 29 | 525 | [1.] | 0 |
| 29 | 525 | [0.] | 0 |
| 30 | 525 | [0.] | 0 |
| 30 | 525 | [1.] | 0 |
| 30 | 525 | [0.] | 1 |
| 30 | 525 | [0.] | 0 |
| 31 | 525 | [0.] | 0 |
| 31 | 525 | [1.] | 0 |
| 31 | 525 | [1.] | 0 |
| 31 | 525 | [0.] | 0 |
| 32 | 525 | [0.] | 0 |
| 32 | 525 | [1.] | 0 |
| 32 | 525 | [1.] | 0 |
| 32 | 525 | [0.] | 0 |
| 33 | 525 | [0.] | 0 |
| 33 | 525 | [0.] | 1 |
| 33 | 525 | [0.] | 1 |
| 33 | 525 | [0.] | 0 |
| 34 | 525 | [0.] | 0 |
| 34 | 525 | [0.] | 1 |
| 34 | 525 | [0.] | 1 |
| 34 | 525 | [0.] | 0 |
| 35 | 525 | [0.] | 0 |
| 35 | 525 | [1.] | 0 |
| 35 | 525 | [0.] | 1 |
| 35 | 525 | [0.] | 0 |
| 36 | 525 | [0.] | 0 |
| 36 | 525 | [0.] | 1 |
| 36 | 525 | [0.] | 1 |
| 36 | 525 | [0.] | 0 |
| 37 | 525 | [0.] | 0 |
| 37 | 525 | [0.] | 1 |
| 37 | 525 | [0.] | 1 |
| 37 | 525 | [0.] | 0 |
| 38 | 525 | [0.] | 0 |
| 38 | 525 | [1.] | 0 |
| 38 | 525 | [0.] | 1 |
| 38 | 525 | [0.] | 0 |
| 39 | 525 | [0.] | 0 |
| 39 | 525 | [1.] | 0 |
| 39 | 525 | [0.] | 1 |
| 39 | 525 | [0.] | 0 |
| 40 | 525 | [0.] | 0 |
| 40 | 525 | [0.] | 1 |
| 40 | 525 | [1.] | 0 |
| 40 | 525 | [0.] | 0 |
| 41 | 525 | [0.] | 0 |
| 41 | 525 | [1.] | 0 |
| 41 | 525 | [1.] | 0 |
| 41 | 525 | [0.] | 0 |
| 42 | 525 | [0.] | 0 |
| 42 | 525 | [0.] | 1 |
| 42 | 525 | [1.] | 0 |
| 42 | 525 | [0.] | 0 |
| 43 | 525 | [0.] | 0 |
| 43 | 525 | [1.] | 0 |
| 43 | 525 | [1.] | 0 |
| 43 | 525 | [0.] | 0 |
| 44 | 525 | [0.] | 0 |
| 44 | 525 | [1.] | 0 |
| 44 | 525 | [0.] | 1 |
| 44 | 525 | [0.] | 0 |
| 45 | 525 | [0.] | 0 |
| 45 | 525 | [1.] | 0 |
| 45 | 525 | [1.] | 0 |
| 45 | 525 | [0.] | 0 |
| 46 | 525 | [0.] | 0 |
| 46 | 525 | [0.] | 1 |
| 46 | 525 | [1.] | 0 |
| 46 | 525 | [0.] | 0 |
| 47 | 525 | [0.] | 0 |
| 47 | 525 | [1.] | 0 |
| 47 | 525 | [1.] | 0 |
| 47 | 525 | [0.] | 0 |
| 48 | 525 | [0.] | 0 |
| 48 | 525 | [1.] | 0 |
| 48 | 525 | [0.] | 1 |
| 48 | 525 | [0.] | 0 |
| 49 | 525 | [0.] | 0 |
| 49 | 525 | [0.] | 1 |
| 49 | 525 | [0.] | 1 |
| 49 | 525 | [0.] | 0 |
| 50 | 525 | [0.] | 0 |
| 50 | 525 | [1.] | 0 |
| 50 | 525 | [0.] | 1 |
| 50 | 525 | [0.] | 0 |

|  |  |  |  |
| --- | --- | --- | --- |
| Neuronas Capa Oculta | *Epochs* | *Salida* | *Error* |
| 10 | 50 | [0.] | 0 |
| 10 | 50 | [1.] | 0 |
| 10 | 50 | [0.] | 1 |
| 10 | 50 | [1.] | 1 |
| 10 | 75 | [1.] | 1 |
| 10 | 75 | [1.] | 0 |
| 10 | 75 | [1.] | 0 |
| 10 | 75 | [0.] | 0 |
| 10 | 100 | [1.] | 1 |
| 10 | 100 | [0.] | 1 |
| 10 | 100 | [1.] | 0 |
| 10 | 100 | [1.] | 1 |
| 10 | 125 | [0.] | 0 |
| 10 | 125 | [0.] | 1 |
| 10 | 125 | [0.] | 1 |
| 10 | 125 | [0.] | 0 |
| 10 | 150 | [0.] | 0 |
| 10 | 150 | [1.] | 0 |
| 10 | 150 | [0.] | 1 |
| 10 | 150 | [1.] | 1 |
| 10 | 175 | [1.] | 1 |
| 10 | 175 | [1.] | 0 |
| 10 | 175 | [1.] | 0 |
| 10 | 175 | [0.] | 0 |
| 10 | 200 | [1.] | 1 |
| 10 | 200 | [0.] | 1 |
| 10 | 200 | [1.] | 0 |
| 10 | 200 | [0.] | 0 |
| 10 | 225 | [0.] | 0 |
| 10 | 225 | [1.] | 0 |
| 10 | 225 | [0.] | 1 |
| 10 | 225 | [1.] | 1 |
| 10 | 250 | [1.] | 1 |
| 10 | 250 | [1.] | 0 |
| 10 | 250 | [0.] | 1 |
| 10 | 250 | [0.] | 0 |
| 10 | 275 | [0.] | 0 |
| 10 | 275 | [0.] | 1 |
| 10 | 275 | [1.] | 0 |
| 10 | 275 | [1.] | 1 |
| 10 | 300 | [0.] | 0 |
| 10 | 300 | [1.] | 0 |
| 10 | 300 | [0.] | 1 |
| 10 | 300 | [1.] | 1 |
| 10 | 325 | [1.] | 1 |
| 10 | 325 | [1.] | 0 |
| 10 | 325 | [1.] | 0 |
| 10 | 325 | [0.] | 0 |
| 10 | 350 | [0.] | 0 |
| 10 | 350 | [1.] | 0 |
| 10 | 350 | [0.] | 1 |
| 10 | 350 | [1.] | 1 |
| 10 | 375 | [1.] | 1 |
| 10 | 375 | [1.] | 0 |
| 10 | 375 | [0.] | 1 |
| 10 | 375 | [0.] | 0 |
| 10 | 400 | [0.] | 0 |
| 10 | 400 | [0.] | 1 |
| 10 | 400 | [1.] | 0 |
| 10 | 400 | [1.] | 1 |
| 10 | 425 | [0.] | 0 |
| 10 | 425 | [1.] | 0 |
| 10 | 425 | [0.] | 1 |
| 10 | 425 | [0.] | 0 |
| 10 | 450 | [1.] | 1 |
| 10 | 450 | [1.] | 0 |
| 10 | 450 | [0.] | 1 |
| 10 | 450 | [0.] | 0 |
| 10 | 475 | [0.] | 0 |
| 10 | 475 | [1.] | 0 |
| 10 | 475 | [1.] | 0 |
| 10 | 475 | [1.] | 1 |
| 10 | 500 | [0.] | 0 |
| 10 | 500 | [1.] | 0 |
| 10 | 500 | [0.] | 1 |
| 10 | 500 | [1.] | 1 |
| 10 | 525 | [0.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 1 |
| 10 | 550 | [0.] | 0 |
| 10 | 550 | [1.] | 0 |
| 10 | 550 | [1.] | 0 |
| 10 | 550 | [1.] | 1 |
| 10 | 575 | [0.] | 0 |
| 10 | 575 | [1.] | 0 |
| 10 | 575 | [1.] | 0 |
| 10 | 575 | [1.] | 1 |
| 10 | 600 | [0.] | 0 |
| 10 | 600 | [1.] | 0 |
| 10 | 600 | [1.] | 0 |
| 10 | 600 | [0.] | 0 |
| 10 | 625 | [0.] | 0 |
| 10 | 625 | [1.] | 0 |
| 10 | 625 | [1.] | 0 |
| 10 | 625 | [1.] | 1 |
| 10 | 650 | [0.] | 0 |
| 10 | 650 | [1.] | 0 |
| 10 | 650 | [0.] | 1 |
| 10 | 650 | [0.] | 0 |
| 10 | 675 | [0.] | 0 |
| 10 | 675 | [1.] | 0 |
| 10 | 675 | [1.] | 0 |
| 10 | 675 | [1.] | 1 |
| 10 | 700 | [0.] | 0 |
| 10 | 700 | [1.] | 0 |
| 10 | 700 | [1.] | 0 |
| 10 | 700 | [1.] | 1 |
| 10 | 725 | [1.] | 1 |
| 10 | 725 | [1.] | 0 |
| 10 | 725 | [1.] | 0 |
| 10 | 725 | [0.] | 0 |
| 10 | 750 | [0.] | 0 |
| 10 | 750 | [1.] | 0 |
| 10 | 750 | [1.] | 0 |
| 10 | 750 | [0.] | 0 |
| 10 | 775 | [0.] | 0 |
| 10 | 775 | [1.] | 0 |
| 10 | 775 | [1.] | 0 |
| 10 | 775 | [0.] | 0 |
| 10 | 800 | [0.] | 0 |
| 10 | 800 | [0.] | 1 |
| 10 | 800 | [1.] | 0 |
| 10 | 800 | [1.] | 1 |
| 10 | 825 | [0.] | 0 |
| 10 | 825 | [1.] | 0 |
| 10 | 825 | [1.] | 0 |
| 10 | 825 | [0.] | 0 |
| 10 | 850 | [0.] | 0 |
| 10 | 850 | [1.] | 0 |
| 10 | 850 | [1.] | 0 |
| 10 | 850 | [0.] | 0 |
| 10 | 875 | [0.] | 0 |
| 10 | 875 | [1.] | 0 |
| 10 | 875 | [1.] | 0 |
| 10 | 875 | [1.] | 1 |
| 10 | 900 | [0.] | 0 |
| 10 | 900 | [1.] | 0 |
| 10 | 900 | [1.] | 0 |
| 10 | 900 | [0.] | 0 |
| 10 | 925 | [0.] | 0 |
| 10 | 925 | [1.] | 0 |
| 10 | 925 | [1.] | 0 |
| 10 | 925 | [0.] | 0 |
| 10 | 950 | [0.] | 0 |
| 10 | 950 | [1.] | 0 |
| 10 | 950 | [1.] | 0 |
| 10 | 950 | [0.] | 0 |
| 10 | 975 | [0.] | 0 |
| 10 | 975 | [1.] | 0 |
| 10 | 975 | [0.] | 1 |
| 10 | 975 | [1.] | 1 |
| 10 | 1000 | [0.] | 0 |
| 10 | 1000 | [1.] | 0 |
| 10 | 1000 | [1.] | 0 |
| 10 | 1000 | [1.] | 1 |

#### Graficas RELU

Ambas tablas se representan gráficamente a continuación:

Text, icon

Description automatically generated

Text

Description automatically generated

### Función SIGMOIDEA

#### Tablas SIGMOIDEA

|  |  |  |  |
| --- | --- | --- | --- |
| Neuronas Capa Oculta | *Epochs* | *Salida* | *Error* |
| 2 | 525 | [1.] | 1 |
| 2 | 525 | [1.] | 0 |
| 2 | 525 | [0.] | 1 |
| 2 | 525 | [0.] | 0 |
| 3 | 525 | [1.] | 1 |
| 3 | 525 | [0.] | 1 |
| 3 | 525 | [1.] | 0 |
| 3 | 525 | [0.] | 0 |
| 4 | 525 | [1.] | 1 |
| 4 | 525 | [0.] | 1 |
| 4 | 525 | [1.] | 0 |
| 4 | 525 | [0.] | 0 |
| 5 | 525 | [0.] | 0 |
| 5 | 525 | [1.] | 0 |
| 5 | 525 | [1.] | 0 |
| 5 | 525 | [1.] | 1 |
| 6 | 525 | [0.] | 0 |
| 6 | 525 | [1.] | 0 |
| 6 | 525 | [1.] | 0 |
| 6 | 525 | [1.] | 1 |
| 7 | 525 | [0.] | 0 |
| 7 | 525 | [1.] | 0 |
| 7 | 525 | [0.] | 1 |
| 7 | 525 | [1.] | 1 |
| 8 | 525 | [0.] | 0 |
| 8 | 525 | [1.] | 0 |
| 8 | 525 | [0.] | 1 |
| 8 | 525 | [1.] | 1 |
| 9 | 525 | [0.] | 0 |
| 9 | 525 | [1.] | 0 |
| 9 | 525 | [1.] | 0 |
| 9 | 525 | [1.] | 1 |
| 10 | 525 | [0.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 1 |
| 11 | 525 | [0.] | 0 |
| 11 | 525 | [1.] | 0 |
| 11 | 525 | [0.] | 1 |
| 11 | 525 | [1.] | 1 |
| 12 | 525 | [0.] | 0 |
| 12 | 525 | [1.] | 0 |
| 12 | 525 | [1.] | 0 |
| 12 | 525 | [1.] | 1 |
| 13 | 525 | [0.] | 0 |
| 13 | 525 | [1.] | 0 |
| 13 | 525 | [1.] | 0 |
| 13 | 525 | [0.] | 0 |
| 14 | 525 | [0.] | 0 |
| 14 | 525 | [1.] | 0 |
| 14 | 525 | [1.] | 0 |
| 14 | 525 | [0.] | 0 |
| 15 | 525 | [0.] | 0 |
| 15 | 525 | [1.] | 0 |
| 15 | 525 | [1.] | 0 |
| 15 | 525 | [0.] | 0 |
| 16 | 525 | [0.] | 0 |
| 16 | 525 | [1.] | 0 |
| 16 | 525 | [1.] | 0 |
| 16 | 525 | [0.] | 0 |
| 17 | 525 | [0.] | 0 |
| 17 | 525 | [1.] | 0 |
| 17 | 525 | [1.] | 0 |
| 17 | 525 | [1.] | 1 |
| 18 | 525 | [0.] | 0 |
| 18 | 525 | [1.] | 0 |
| 18 | 525 | [1.] | 0 |
| 18 | 525 | [1.] | 1 |
| 19 | 525 | [1.] | 1 |
| 19 | 525 | [0.] | 1 |
| 19 | 525 | [1.] | 0 |
| 19 | 525 | [0.] | 0 |
| 20 | 525 | [0.] | 0 |
| 20 | 525 | [1.] | 0 |
| 20 | 525 | [1.] | 0 |
| 20 | 525 | [1.] | 1 |
| 21 | 525 | [0.] | 0 |
| 21 | 525 | [1.] | 0 |
| 21 | 525 | [1.] | 0 |
| 21 | 525 | [1.] | 1 |
| 22 | 525 | [0.] | 0 |
| 22 | 525 | [1.] | 0 |
| 22 | 525 | [1.] | 0 |
| 22 | 525 | [0.] | 0 |
| 23 | 525 | [0.] | 0 |
| 23 | 525 | [1.] | 0 |
| 23 | 525 | [1.] | 0 |
| 23 | 525 | [1.] | 1 |
| 24 | 525 | [1.] | 1 |
| 24 | 525 | [0.] | 1 |
| 24 | 525 | [1.] | 0 |
| 24 | 525 | [0.] | 0 |
| 25 | 525 | [0.] | 0 |
| 25 | 525 | [1.] | 0 |
| 25 | 525 | [1.] | 0 |
| 25 | 525 | [1.] | 1 |
| 26 | 525 | [0.] | 0 |
| 26 | 525 | [1.] | 0 |
| 26 | 525 | [1.] | 0 |
| 26 | 525 | [1.] | 1 |
| 27 | 525 | [0.] | 0 |
| 27 | 525 | [1.] | 0 |
| 27 | 525 | [1.] | 0 |
| 27 | 525 | [1.] | 1 |
| 28 | 525 | [0.] | 0 |
| 28 | 525 | [1.] | 0 |
| 28 | 525 | [1.] | 0 |
| 28 | 525 | [0.] | 0 |
| 29 | 525 | [0.] | 0 |
| 29 | 525 | [1.] | 0 |
| 29 | 525 | [1.] | 0 |
| 29 | 525 | [1.] | 1 |
| 30 | 525 | [0.] | 0 |
| 30 | 525 | [1.] | 0 |
| 30 | 525 | [1.] | 0 |
| 30 | 525 | [0.] | 0 |
| 31 | 525 | [0.] | 0 |
| 31 | 525 | [1.] | 0 |
| 31 | 525 | [0.] | 1 |
| 31 | 525 | [1.] | 1 |
| 32 | 525 | [0.] | 0 |
| 32 | 525 | [1.] | 0 |
| 32 | 525 | [1.] | 0 |
| 32 | 525 | [1.] | 1 |
| 33 | 525 | [0.] | 0 |
| 33 | 525 | [1.] | 0 |
| 33 | 525 | [1.] | 0 |
| 33 | 525 | [0.] | 0 |
| 34 | 525 | [0.] | 0 |
| 34 | 525 | [1.] | 0 |
| 34 | 525 | [1.] | 0 |
| 34 | 525 | [0.] | 0 |
| 35 | 525 | [0.] | 0 |
| 35 | 525 | [1.] | 0 |
| 35 | 525 | [1.] | 0 |
| 35 | 525 | [0.] | 0 |
| 36 | 525 | [0.] | 0 |
| 36 | 525 | [1.] | 0 |
| 36 | 525 | [1.] | 0 |
| 36 | 525 | [1.] | 1 |
| 37 | 525 | [0.] | 0 |
| 37 | 525 | [1.] | 0 |
| 37 | 525 | [1.] | 0 |
| 37 | 525 | [1.] | 1 |
| 38 | 525 | [0.] | 0 |
| 38 | 525 | [1.] | 0 |
| 38 | 525 | [1.] | 0 |
| 38 | 525 | [1.] | 1 |
| 39 | 525 | [0.] | 0 |
| 39 | 525 | [1.] | 0 |
| 39 | 525 | [1.] | 0 |
| 39 | 525 | [1.] | 1 |
| 40 | 525 | [0.] | 0 |
| 40 | 525 | [1.] | 0 |
| 40 | 525 | [1.] | 0 |
| 40 | 525 | [0.] | 0 |
| 41 | 525 | [0.] | 0 |
| 41 | 525 | [1.] | 0 |
| 41 | 525 | [1.] | 0 |
| 41 | 525 | [1.] | 1 |
| 42 | 525 | [0.] | 0 |
| 42 | 525 | [1.] | 0 |
| 42 | 525 | [1.] | 0 |
| 42 | 525 | [1.] | 1 |
| 43 | 525 | [0.] | 0 |
| 43 | 525 | [1.] | 0 |
| 43 | 525 | [1.] | 0 |
| 43 | 525 | [1.] | 1 |
| 44 | 525 | [0.] | 0 |
| 44 | 525 | [1.] | 0 |
| 44 | 525 | [1.] | 0 |
| 44 | 525 | [0.] | 0 |
| 45 | 525 | [0.] | 0 |
| 45 | 525 | [1.] | 0 |
| 45 | 525 | [0.] | 1 |
| 45 | 525 | [1.] | 1 |
| 46 | 525 | [0.] | 0 |
| 46 | 525 | [1.] | 0 |
| 46 | 525 | [1.] | 0 |
| 46 | 525 | [1.] | 1 |
| 47 | 525 | [0.] | 0 |
| 47 | 525 | [1.] | 0 |
| 47 | 525 | [1.] | 0 |
| 47 | 525 | [1.] | 1 |
| 48 | 525 | [0.] | 0 |
| 48 | 525 | [1.] | 0 |
| 48 | 525 | [1.] | 0 |
| 48 | 525 | [1.] | 1 |
| 49 | 525 | [0.] | 0 |
| 49 | 525 | [1.] | 0 |
| 49 | 525 | [1.] | 0 |
| 49 | 525 | [0.] | 0 |
| 50 | 525 | [0.] | 0 |
| 50 | 525 | [1.] | 0 |
| 50 | 525 | [1.] | 0 |
| 50 | 525 | [1.] | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| Neuronas Capa Oculta | *Epochs* | *Salida* | *Error* |
| 10 | 50 | [1.] | 0 |
| 10 | 50 | [0.] | 1 |
| 10 | 50 | [1.] | 1 |
| 10 | 75 | [1.] | 1 |
| 10 | 75 | [1.] | 0 |
| 10 | 75 | [1.] | 0 |
| 10 | 75 | [0.] | 0 |
| 10 | 100 | [1.] | 1 |
| 10 | 100 | [0.] | 1 |
| 10 | 100 | [1.] | 0 |
| 10 | 100 | [1.] | 1 |
| 10 | 125 | [0.] | 0 |
| 10 | 125 | [0.] | 1 |
| 10 | 125 | [0.] | 1 |
| 10 | 125 | [0.] | 0 |
| 10 | 150 | [0.] | 0 |
| 10 | 150 | [1.] | 0 |
| 10 | 150 | [0.] | 1 |
| 10 | 150 | [1.] | 1 |
| 10 | 175 | [1.] | 1 |
| 10 | 175 | [1.] | 0 |
| 10 | 175 | [1.] | 0 |
| 10 | 175 | [0.] | 0 |
| 10 | 200 | [1.] | 1 |
| 10 | 200 | [0.] | 1 |
| 10 | 200 | [1.] | 0 |
| 10 | 200 | [0.] | 0 |
| 10 | 225 | [0.] | 0 |
| 10 | 225 | [1.] | 0 |
| 10 | 225 | [0.] | 1 |
| 10 | 225 | [1.] | -1 |
| 10 | 250 | [1.] | -1 |
| 10 | 250 | [1.] | 0 |
| 10 | 250 | [0.] | 1 |
| 10 | 250 | [0.] | 0 |
| 10 | 275 | [0.] | 0 |
| 10 | 275 | [0.] | 1 |
| 10 | 275 | [1.] | 0 |
| 10 | 275 | [1.] | 1 |
| 10 | 300 | [0.] | 0 |
| 10 | 300 | [1.] | 0 |
| 10 | 300 | [0.] | 1 |
| 10 | 300 | [1.] | 1 |
| 10 | 325 | [1.] | 1 |
| 10 | 325 | [1.] | 0 |
| 10 | 325 | [1.] | 0 |
| 10 | 325 | [0.] | 0 |
| 10 | 350 | [0.] | 0 |
| 10 | 350 | [1.] | 0 |
| 10 | 350 | [0.] | 1 |
| 10 | 350 | [1.] | 1 |
| 10 | 375 | [1.] | 1 |
| 10 | 375 | [1.] | 0 |
| 10 | 375 | [0.] | 1 |
| 10 | 375 | [0.] | 0 |
| 10 | 400 | [0.] | 0 |
| 10 | 400 | [0.] | 1 |
| 10 | 400 | [1.] | 0 |
| 10 | 400 | [1.] | 1 |
| 10 | 425 | [0.] | 0 |
| 10 | 425 | [1.] | 0 |
| 10 | 425 | [0.] | 1 |
| 10 | 425 | [0.] | 0 |
| 10 | 450 | [1.] | 1 |
| 10 | 450 | [1.] | 0 |
| 10 | 450 | [0.] | 1 |
| 10 | 450 | [0.] | 0 |
| 10 | 475 | [0.] | 0 |
| 10 | 475 | [1.] | 0 |
| 10 | 475 | [1.] | 0 |
| 10 | 475 | [1.] | 1 |
| 10 | 500 | [0.] | 0 |
| 10 | 500 | [1.] | 0 |
| 10 | 500 | [0.] | 1 |
| 10 | 500 | [1.] | 1 |
| 10 | 525 | [0.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 1 |
| 10 | 550 | [0.] | 0 |
| 10 | 550 | [1.] | 0 |
| 10 | 550 | [1.] | 0 |
| 10 | 550 | [1.] | 1 |
| 10 | 575 | [0.] | 0 |
| 10 | 575 | [1.] | 0 |
| 10 | 575 | [1.] | 0 |
| 10 | 575 | [1.] | 1 |
| 10 | 600 | [0.] | 0 |
| 10 | 600 | [1.] | 0 |
| 10 | 600 | [1.] | 0 |
| 10 | 600 | [0.] | 0 |
| 10 | 625 | [0.] | 0 |
| 10 | 625 | [1.] | 0 |
| 10 | 625 | [1.] | 0 |
| 10 | 625 | [1.] | 1 |
| 10 | 650 | [0.] | 0 |
| 10 | 650 | [1.] | 0 |
| 10 | 650 | [0.] | 1 |
| 10 | 650 | [0.] | 0 |
| 10 | 675 | [0.] | 0 |
| 10 | 675 | [1.] | 0 |
| 10 | 675 | [1.] | 0 |
| 10 | 675 | [1.] | 1 |
| 10 | 700 | [0.] | 0 |
| 10 | 700 | [1.] | 0 |
| 10 | 700 | [1.] | 0 |
| 10 | 700 | [1.] | 1 |
| 10 | 725 | [1.] | 1 |
| 10 | 725 | [1.] | 0 |
| 10 | 725 | [1.] | 0 |
| 10 | 725 | [0.] | 0 |
| 10 | 750 | [0.] | 0 |
| 10 | 750 | [1.] | 0 |
| 10 | 750 | [1.] | 0 |
| 10 | 750 | [0.] | 0 |
| 10 | 775 | [0.] | 0 |
| 10 | 775 | [1.] | 0 |
| 10 | 775 | [1.] | 0 |
| 10 | 775 | [0.] | 0 |
| 10 | 800 | [0.] | 0 |
| 10 | 800 | [0.] | 1 |
| 10 | 800 | [1.] | 0 |
| 10 | 800 | [1.] | 1 |
| 10 | 825 | [0.] | 0 |
| 10 | 825 | [1.] | 0 |
| 10 | 825 | [1.] | 0 |
| 10 | 825 | [0.] | 0 |
| 10 | 850 | [0.] | 0 |
| 10 | 850 | [1.] | 0 |
| 10 | 850 | [1.] | 0 |
| 10 | 850 | [0.] | 0 |
| 10 | 875 | [0.] | 0 |
| 10 | 875 | [1.] | 0 |
| 10 | 875 | [1.] | 0 |
| 10 | 875 | [1.] | 1 |
| 10 | 900 | [0.] | 0 |
| 10 | 900 | [1.] | 0 |
| 10 | 900 | [1.] | 0 |
| 10 | 900 | [0.] | 0 |
| 10 | 925 | [0.] | 0 |
| 10 | 925 | [1.] | 0 |
| 10 | 925 | [1.] | 0 |
| 10 | 925 | [0.] | 0 |
| 10 | 950 | [0.] | 0 |
| 10 | 950 | [1.] | 0 |
| 10 | 950 | [1.] | 0 |
| 10 | 950 | [0.] | 0 |
| 10 | 975 | [0.] | 0 |
| 10 | 975 | [1.] | 0 |
| 10 | 975 | [0.] | 1 |
| 10 | 975 | [1.] | 1 |
| 10 | 1000 | [0.] | 0 |
| 10 | 1000 | [1.] | 0 |
| 10 | 1000 | [1.] | 0 |
| 10 | 1000 | [1.] | 1 |

#### Graficas SIGMOIDEA

Ambas tablas se representan gráficamente a continuación:

A picture containing text

Description automatically generated

Text

Description automatically generated

### Función RELU y SIGMOIDEA

En este caso la capa de oculta usa la función RELU y la capa de salida usa la función SIGMOIDEA.

#### Tablas RELU y SIGMOIDEA

|  |  |  |  |
| --- | --- | --- | --- |
| Neuronas Capa Oculta | *Epochs* | *Salida* | *Error* |
| 2 | 525 | [0.] | 0 |
| 2 | 525 | [0.] | 1 |
| 2 | 525 | [1.] | 0 |
| 2 | 525 | [0.] | 0 |
| 3 | 525 | [0.] | 0 |
| 3 | 525 | [1.] | 0 |
| 3 | 525 | [1.] | 0 |
| 3 | 525 | [0.] | 0 |
| 4 | 525 | [0.] | 0 |
| 4 | 525 | [1.] | 0 |
| 4 | 525 | [0.] | 1 |
| 4 | 525 | [0.] | 0 |
| 5 | 525 | [0.] | 0 |
| 5 | 525 | [0.] | 1 |
| 5 | 525 | [1.] | 0 |
| 5 | 525 | [0.] | 0 |
| 6 | 525 | [0.] | 0 |
| 6 | 525 | [1.] | 0 |
| 6 | 525 | [1.] | 0 |
| 6 | 525 | [0.] | 0 |
| 7 | 525 | [0.] | 0 |
| 7 | 525 | [1.] | 0 |
| 7 | 525 | [1.] | 0 |
| 7 | 525 | [0.] | 0 |
| 8 | 525 | [0.] | 0 |
| 8 | 525 | [1.] | 0 |
| 8 | 525 | [0.] | 1 |
| 8 | 525 | [0.] | 0 |
| 9 | 525 | [0.] | 0 |
| 9 | 525 | [1.] | 0 |
| 9 | 525 | [1.] | 0 |
| 9 | 525 | [0.] | 0 |
| 10 | 525 | [0.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [0.] | 0 |
| 11 | 525 | [0.] | 0 |
| 11 | 525 | [1.] | 0 |
| 11 | 525 | [1.] | 0 |
| 11 | 525 | [0.] | 0 |
| 12 | 525 | [0.] | 0 |
| 12 | 525 | [1.] | 0 |
| 12 | 525 | [1.] | 0 |
| 12 | 525 | [0.] | 0 |
| 13 | 525 | [0.] | 0 |
| 13 | 525 | [1.] | 0 |
| 13 | 525 | [1.] | 0 |
| 13 | 525 | [0.] | 0 |
| 14 | 525 | [0.] | 0 |
| 14 | 525 | [1.] | 0 |
| 14 | 525 | [1.] | 0 |
| 14 | 525 | [0.] | 0 |
| 15 | 525 | [0.] | 0 |
| 15 | 525 | [1.] | 0 |
| 15 | 525 | [1.] | 0 |
| 15 | 525 | [0.] | 0 |
| 16 | 525 | [0.] | 0 |
| 16 | 525 | [1.] | 0 |
| 16 | 525 | [1.] | 0 |
| 16 | 525 | [0.] | 0 |
| 17 | 525 | [0.] | 0 |
| 17 | 525 | [1.] | 0 |
| 17 | 525 | [1.] | 0 |
| 17 | 525 | [0.] | 0 |
| 18 | 525 | [0.] | 0 |
| 18 | 525 | [1.] | 0 |
| 18 | 525 | [1.] | 0 |
| 18 | 525 | [0.] | 0 |
| 19 | 525 | [0.] | 0 |
| 19 | 525 | [1.] | 0 |
| 19 | 525 | [1.] | 0 |
| 19 | 525 | [0.] | 0 |
| 20 | 525 | [0.] | 0 |
| 20 | 525 | [1.] | 0 |
| 20 | 525 | [1.] | 0 |
| 20 | 525 | [0.] | 0 |
| 21 | 525 | [0.] | 0 |
| 21 | 525 | [1.] | 0 |
| 21 | 525 | [1.] | 0 |
| 21 | 525 | [0.] | 0 |
| 22 | 525 | [0.] | 0 |
| 22 | 525 | [1.] | 0 |
| 22 | 525 | [1.] | 0 |
| 22 | 525 | [0.] | 0 |
| 23 | 525 | [0.] | 0 |
| 23 | 525 | [1.] | 0 |
| 23 | 525 | [1.] | 0 |
| 23 | 525 | [0.] | 0 |
| 24 | 525 | [0.] | 0 |
| 24 | 525 | [1.] | 0 |
| 24 | 525 | [1.] | 0 |
| 24 | 525 | [0.] | 0 |
| 25 | 525 | [0.] | 0 |
| 25 | 525 | [1.] | 0 |
| 25 | 525 | [1.] | 0 |
| 25 | 525 | [0.] | 0 |
| 26 | 525 | [0.] | 0 |
| 26 | 525 | [1.] | 0 |
| 26 | 525 | [1.] | 0 |
| 26 | 525 | [0.] | 0 |
| 27 | 525 | [0.] | 0 |
| 27 | 525 | [1.] | 0 |
| 27 | 525 | [1.] | 0 |
| 27 | 525 | [0.] | 0 |
| 28 | 525 | [0.] | 0 |
| 28 | 525 | [1.] | 0 |
| 28 | 525 | [1.] | 0 |
| 28 | 525 | [0.] | 0 |
| 29 | 525 | [0.] | 0 |
| 29 | 525 | [1.] | 0 |
| 29 | 525 | [1.] | 0 |
| 29 | 525 | [0.] | 0 |
| 30 | 525 | [0.] | 0 |
| 30 | 525 | [1.] | 0 |
| 30 | 525 | [1.] | 0 |
| 30 | 525 | [0.] | 0 |
| 31 | 525 | [0.] | 0 |
| 31 | 525 | [1.] | 0 |
| 31 | 525 | [1.] | 0 |
| 31 | 525 | [0.] | 0 |
| 32 | 525 | [0.] | 0 |
| 32 | 525 | [1.] | 0 |
| 32 | 525 | [1.] | 0 |
| 32 | 525 | [0.] | 0 |
| 33 | 525 | [0.] | 0 |
| 33 | 525 | [1.] | 0 |
| 33 | 525 | [1.] | 0 |
| 33 | 525 | [0.] | 0 |
| 34 | 525 | [0.] | 0 |
| 34 | 525 | [1.] | 0 |
| 34 | 525 | [1.] | 0 |
| 34 | 525 | [0.] | 0 |
| 35 | 525 | [0.] | 0 |
| 35 | 525 | [1.] | 0 |
| 35 | 525 | [1.] | 0 |
| 35 | 525 | [0.] | 0 |
| 36 | 525 | [0.] | 0 |
| 36 | 525 | [1.] | 0 |
| 36 | 525 | [1.] | 0 |
| 36 | 525 | [0.] | 0 |
| 37 | 525 | [0.] | 0 |
| 37 | 525 | [1.] | 0 |
| 37 | 525 | [1.] | 0 |
| 37 | 525 | [0.] | 0 |
| 38 | 525 | [0.] | 0 |
| 38 | 525 | [1.] | 0 |
| 38 | 525 | [1.] | 0 |
| 38 | 525 | [0.] | 0 |
| 39 | 525 | [0.] | 0 |
| 39 | 525 | [1.] | 0 |
| 39 | 525 | [1.] | 0 |
| 39 | 525 | [0.] | 0 |
| 40 | 525 | [0.] | 0 |
| 40 | 525 | [1.] | 0 |
| 40 | 525 | [1.] | 0 |
| 40 | 525 | [0.] | 0 |
| 41 | 525 | [0.] | 0 |
| 41 | 525 | [1.] | 0 |
| 41 | 525 | [1.] | 0 |
| 41 | 525 | [0.] | 0 |
| 42 | 525 | [0.] | 0 |
| 42 | 525 | [1.] | 0 |
| 42 | 525 | [1.] | 0 |
| 42 | 525 | [0.] | 0 |
| 43 | 525 | [0.] | 0 |
| 43 | 525 | [1.] | 0 |
| 43 | 525 | [1.] | 0 |
| 43 | 525 | [0.] | 0 |
| 44 | 525 | [0.] | 0 |
| 44 | 525 | [1.] | 0 |
| 44 | 525 | [1.] | 0 |
| 44 | 525 | [0.] | 0 |
| 45 | 525 | [0.] | 0 |
| 45 | 525 | [1.] | 0 |
| 45 | 525 | [1.] | 0 |
| 45 | 525 | [0.] | 0 |
| 46 | 525 | [0.] | 0 |
| 46 | 525 | [1.] | 0 |
| 46 | 525 | [1.] | 0 |
| 46 | 525 | [0.] | 0 |
| 47 | 525 | [0.] | 0 |
| 47 | 525 | [1.] | 0 |
| 47 | 525 | [1.] | 0 |
| 47 | 525 | [0.] | 0 |
| 48 | 525 | [0.] | 0 |
| 48 | 525 | [1.] | 0 |
| 48 | 525 | [1.] | 0 |
| 48 | 525 | [0.] | 0 |
| 49 | 525 | [0.] | 0 |
| 49 | 525 | [1.] | 0 |
| 49 | 525 | [1.] | 0 |
| 49 | 525 | [0.] | 0 |
| 50 | 525 | [0.] | 0 |
| 50 | 525 | [1.] | 0 |
| 50 | 525 | [1.] | 0 |
| 50 | 525 | [0.] | 0 |

|  |  |  |  |
| --- | --- | --- | --- |
| Neuronas Capa Oculta | Epochs | Salida | Error |
| 10 | 50 | [0.] | 0 |
| 10 | 50 | [1.] | 0 |
| 10 | 50 | [1.] | 0 |
| 10 | 50 | [0.] | 0 |
| 10 | 75 | [0.] | 0 |
| 10 | 75 | [1.] | 0 |
| 10 | 75 | [0.] | 1 |
| 10 | 75 | [1.] | 1 |
| 10 | 100 | [0.] | 0 |
| 10 | 100 | [1.] | 0 |
| 10 | 100 | [1.] | 0 |
| 10 | 100 | [0.] | 0 |
| 10 | 125 | [1.] | 1 |
| 10 | 125 | [0.] | 1 |
| 10 | 125 | [1.] | 0 |
| 10 | 125 | [0.] | 0 |
| 10 | 150 | [0.] | 0 |
| 10 | 150 | [1.] | 0 |
| 10 | 150 | [1.] | 0 |
| 10 | 150 | [0.] | 0 |
| 10 | 175 | [0.] | 0 |
| 10 | 175 | [1.] | 0 |
| 10 | 175 | [1.] | 0 |
| 10 | 175 | [0.] | 0 |
| 10 | 200 | [1.] | 1 |
| 10 | 200 | [0.] | 1 |
| 10 | 200 | [1.] | 0 |
| 10 | 200 | [0.] | 0 |
| 10 | 225 | [0.] | 0 |
| 10 | 225 | [1.] | 0 |
| 10 | 225 | [1.] | 0 |
| 10 | 225 | [0.] | 0 |
| 10 | 250 | [0.] | 0 |
| 10 | 250 | [1.] | 0 |
| 10 | 250 | [1.] | 0 |
| 10 | 250 | [0.] | 0 |
| 10 | 275 | [0.] | 0 |
| 10 | 275 | [1.] | 0 |
| 10 | 275 | [1.] | 0 |
| 10 | 275 | [0.] | 0 |
| 10 | 300 | [0.] | 0 |
| 10 | 300 | [1.] | 0 |
| 10 | 300 | [1.] | 0 |
| 10 | 300 | [0.] | 0 |
| 10 | 325 | [0.] | 0 |
| 10 | 325 | [1.] | 0 |
| 10 | 325 | [1.] | 0 |
| 10 | 325 | [0.] | 0 |
| 10 | 350 | [0.] | 0 |
| 10 | 350 | [1.] | 0 |
| 10 | 350 | [1.] | 0 |
| 10 | 350 | [0.] | 0 |
| 10 | 375 | [0.] | 0 |
| 10 | 375 | [1.] | 0 |
| 10 | 375 | [1.] | 0 |
| 10 | 375 | [0.] | 0 |
| 10 | 400 | [0.] | 0 |
| 10 | 400 | [1.] | 0 |
| 10 | 400 | [1.] | 0 |
| 10 | 400 | [0.] | 0 |
| 10 | 425 | [0.] | 0 |
| 10 | 425 | [1.] | 0 |
| 10 | 425 | [1.] | 0 |
| 10 | 425 | [0.] | 0 |
| 10 | 450 | [0.] | 0 |
| 10 | 450 | [1.] | 0 |
| 10 | 450 | [1.] | 0 |
| 10 | 450 | [0.] | 0 |
| 10 | 475 | [1.] | 1 |
| 10 | 475 | [1.] | 0 |
| 10 | 475 | [1.] | 0 |
| 10 | 475 | [0.] | 0 |
| 10 | 500 | [0.] | 0 |
| 10 | 500 | [1.] | 0 |
| 10 | 500 | [1.] | 0 |
| 10 | 500 | [0.] | 0 |
| 10 | 525 | [0.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [0.] | 0 |
| 10 | 550 | [0.] | 0 |
| 10 | 550 | [1.] | 0 |
| 10 | 550 | [1.] | 0 |
| 10 | 550 | [0.] | 0 |
| 10 | 575 | [0.] | 0 |
| 10 | 575 | [1.] | 0 |
| 10 | 575 | [1.] | 0 |
| 10 | 575 | [0.] | 0 |
| 10 | 600 | [0.] | 0 |
| 10 | 600 | [1.] | 0 |
| 10 | 600 | [1.] | 0 |
| 10 | 600 | [0.] | 0 |
| 10 | 625 | [0.] | 0 |
| 10 | 625 | [1.] | 0 |
| 10 | 625 | [1.] | 0 |
| 10 | 625 | [0.] | 0 |
| 10 | 650 | [0.] | 0 |
| 10 | 650 | [1.] | 0 |
| 10 | 650 | [1.] | 0 |
| 10 | 650 | [0.] | 0 |
| 10 | 675 | [1.] | 1 |
| 10 | 675 | [1.] | 0 |
| 10 | 675 | [1.] | 0 |
| 10 | 675 | [0.] | 0 |
| 10 | 700 | [0.] | 0 |
| 10 | 700 | [1.] | 0 |
| 10 | 700 | [1.] | 0 |
| 10 | 700 | [0.] | 0 |
| 10 | 725 | [0.] | 0 |
| 10 | 725 | [1.] | 0 |
| 10 | 725 | [1.] | 0 |
| 10 | 725 | [0.] | 0 |
| 10 | 750 | [0.] | 0 |
| 10 | 750 | [1.] | 0 |
| 10 | 750 | [1.] | 0 |
| 10 | 750 | [0.] | 0 |
| 10 | 775 | [0.] | 0 |
| 10 | 775 | [1.] | 0 |
| 10 | 775 | [1.] | 0 |
| 10 | 775 | [0.] | 0 |
| 10 | 800 | [0.] | 0 |
| 10 | 800 | [1.] | 0 |
| 10 | 800 | [1.] | 0 |
| 10 | 800 | [0.] | 0 |
| 10 | 825 | [0.] | 0 |
| 10 | 825 | [1.] | 0 |
| 10 | 825 | [1.] | 0 |
| 10 | 825 | [0.] | 0 |
| 10 | 850 | [0.] | 0 |
| 10 | 850 | [1.] | 0 |
| 10 | 850 | [1.] | 0 |
| 10 | 850 | [0.] | 0 |
| 10 | 875 | [0.] | 0 |
| 10 | 875 | [1.] | 0 |
| 10 | 875 | [1.] | 0 |
| 10 | 875 | [0.] | 0 |
| 10 | 900 | [0.] | 0 |
| 10 | 900 | [1.] | 0 |
| 10 | 900 | [1.] | 0 |
| 10 | 900 | [0.] | 0 |
| 10 | 925 | [0.] | 0 |
| 10 | 925 | [1.] | 0 |
| 10 | 925 | [1.] | 0 |
| 10 | 925 | [0.] | 0 |
| 10 | 950 | [0.] | 0 |
| 10 | 950 | [1.] | 0 |
| 10 | 950 | [1.] | 0 |
| 10 | 950 | [0.] | 0 |
| 10 | 975 | [0.] | 0 |
| 10 | 975 | [1.] | 0 |
| 10 | 975 | [1.] | 0 |
| 10 | 975 | [0.] | 0 |
| 10 | 1000 | [0.] | 0 |
| 10 | 1000 | [1.] | 0 |
| 10 | 1000 | [1.] | 0 |
| 10 | 1000 | [0.] | 0 |

#### Graficas RELU y SIGMOIDEA

Ambas tablas se representan gráficamente a continuación:

A picture containing text

Description automatically generated

Icon

Description automatically generated

### Función SIGMOIDEA y RELU

En este caso la función de activación de la capa oculta es SIGMOIDEA y la de la capa de salida RELU.

#### Tablas SIGMOIDEA y RELU

|  |  |  |  |
| --- | --- | --- | --- |
| Neuronas Capa Oculta | Epochs | Salida | Error |
| 2 | 525 | [0.] | 0 |
| 2 | 525 | [0.] | 1 |
| 2 | 525 | [0.] | 1 |
| 2 | 525 | [0.] | 0 |
| 3 | 525 | [1.] | 1 |
| 3 | 525 | [0.] | 1 |
| 3 | 525 | [1.] | 0 |
| 3 | 525 | [0.] | 0 |
| 4 | 525 | [0.] | 0 |
| 4 | 525 | [0.] | 1 |
| 4 | 525 | [1.] | 0 |
| 4 | 525 | [1.] | 1 |
| 5 | 525 | [0.] | 0 |
| 5 | 525 | [0.] | 1 |
| 5 | 525 | [0.] | 1 |
| 5 | 525 | [0.] | 0 |
| 6 | 525 | [0.] | 0 |
| 6 | 525 | [1.] | 0 |
| 6 | 525 | [1.] | 0 |
| 6 | 525 | [1.] | 1 |
| 7 | 525 | [0.] | 0 |
| 7 | 525 | [0.] | 1 |
| 7 | 525 | [1.] | 0 |
| 7 | 525 | [1.] | 1 |
| 8 | 525 | [0.] | 0 |
| 8 | 525 | [0.] | 1 |
| 8 | 525 | [0.] | 1 |
| 8 | 525 | [0.] | 0 |
| 9 | 525 | [0.] | 0 |
| 9 | 525 | [0.] | 1 |
| 9 | 525 | [0.] | 1 |
| 9 | 525 | [0.] | 0 |
| 10 | 525 | [1.] | 1 |
| 10 | 525 | [1.] | 0 |
| 10 | 525 | [0.] | 1 |
| 10 | 525 | [0.] | 0 |
| 11 | 525 | [0.] | 0 |
| 11 | 525 | [0.] | 1 |
| 11 | 525 | [0.] | 1 |
| 11 | 525 | [0.] | 0 |
| 12 | 525 | [0.] | 0 |
| 12 | 525 | [1.] | 0 |
| 12 | 525 | [1.] | 0 |
| 12 | 525 | [1.] | 1 |
| 13 | 525 | [0.] | 0 |
| 13 | 525 | [1.] | 0 |
| 13 | 525 | [1.] | 0 |
| 13 | 525 | [1.] | 1 |
| 14 | 525 | [0.] | 0 |
| 14 | 525 | [1.] | 0 |
| 14 | 525 | [1.] | 0 |
| 14 | 525 | [1.] | 1 |
| 15 | 525 | [0.] | 0 |
| 15 | 525 | [0.] | 1 |
| 15 | 525 | [0.] | 1 |
| 15 | 525 | [0.] | 0 |
| 16 | 525 | [0.] | 0 |
| 16 | 525 | [0.] | 1 |
| 16 | 525 | [0.] | 1 |
| 16 | 525 | [0.] | 0 |
| 17 | 525 | [0.] | 0 |
| 17 | 525 | [0.] | 1 |
| 17 | 525 | [0.] | 1 |
| 17 | 525 | [0.] | 0 |
| 18 | 525 | [0.] | 0 |
| 18 | 525 | [0.] | 1 |
| 18 | 525 | [0.] | 1 |
| 18 | 525 | [0.] | 0 |
| 19 | 525 | [0.] | 0 |
| 19 | 525 | [0.] | 1 |
| 19 | 525 | [0.] | 1 |
| 19 | 525 | [0.] | 0 |
| 20 | 525 | [0.] | 0 |
| 20 | 525 | [0.] | 1 |
| 20 | 525 | [0.] | 1 |
| 20 | 525 | [0.] | 0 |
| 21 | 525 | [0.] | 0 |
| 21 | 525 | [0.] | 1 |
| 21 | 525 | [1.] | 0 |
| 21 | 525 | [1.] | 1 |
| 22 | 525 | [0.] | 0 |
| 22 | 525 | [1.] | 0 |
| 22 | 525 | [1.] | 0 |
| 22 | 525 | [1.] | 1 |
| 23 | 525 | [0.] | 0 |
| 23 | 525 | [0.] | 1 |
| 23 | 525 | [0.] | 1 |
| 23 | 525 | [0.] | 0 |
| 24 | 525 | [0.] | 0 |
| 24 | 525 | [1.] | 0 |
| 24 | 525 | [1.] | 0 |
| 24 | 525 | [1.] | 1 |
| 25 | 525 | [0.] | 0 |
| 25 | 525 | [1.] | 0 |
| 25 | 525 | [1.] | 0 |
| 25 | 525 | [0.] | 0 |
| 26 | 525 | [0.] | 0 |
| 26 | 525 | [1.] | 0 |
| 26 | 525 | [1.] | 0 |
| 26 | 525 | [0.] | 0 |
| 27 | 525 | [0.] | 0 |
| 27 | 525 | [0.] | 1 |
| 27 | 525 | [0.] | 1 |
| 27 | 525 | [0.] | 0 |
| 28 | 525 | [0.] | 0 |
| 28 | 525 | [1.] | 0 |
| 28 | 525 | [1.] | 0 |
| 28 | 525 | [1.] | 1 |
| 29 | 525 | [0.] | 0 |
| 29 | 525 | [1.] | 0 |
| 29 | 525 | [1.] | 0 |
| 29 | 525 | [1.] | 1 |
| 30 | 525 | [0.] | 0 |
| 30 | 525 | [1.] | 0 |
| 30 | 525 | [1.] | 0 |
| 30 | 525 | [1.] | 1 |
| 31 | 525 | [0.] | 0 |
| 31 | 525 | [1.] | 0 |
| 31 | 525 | [1.] | 0 |
| 31 | 525 | [1.] | 1 |
| 32 | 525 | [0.] | 0 |
| 32 | 525 | [0.] | 1 |
| 32 | 525 | [0.] | 1 |
| 32 | 525 | [0.] | 0 |
| 33 | 525 | [0.] | 0 |
| 33 | 525 | [1.] | 0 |
| 33 | 525 | [1.] | 0 |
| 33 | 525 | [1.] | 1 |
| 34 | 525 | [0.] | 0 |
| 34 | 525 | [1.] | 0 |
| 34 | 525 | [1.] | 0 |
| 34 | 525 | [1.] | 1 |
| 35 | 525 | [0.] | 0 |
| 35 | 525 | [0.] | 1 |
| 35 | 525 | [0.] | 1 |
| 35 | 525 | [0.] | 0 |
| 36 | 525 | [0.] | 0 |
| 36 | 525 | [1.] | 0 |
| 36 | 525 | [1.] | 0 |
| 36 | 525 | [1.] | 1 |
| 37 | 525 | [0.] | 0 |
| 37 | 525 | [0.] | 1 |
| 37 | 525 | [0.] | 1 |
| 37 | 525 | [0.] | 0 |
| 38 | 525 | [0.] | 0 |
| 38 | 525 | [0.] | 1 |
| 38 | 525 | [0.] | 1 |
| 38 | 525 | [0.] | 0 |
| 39 | 525 | [0.] | 0 |
| 39 | 525 | [1.] | 0 |
| 39 | 525 | [1.] | 0 |
| 39 | 525 | [1.] | 1 |
| 40 | 525 | [0.] | 0 |
| 40 | 525 | [1.] | 0 |
| 40 | 525 | [1.] | 0 |
| 40 | 525 | [1.] | 1 |
| 41 | 525 | [0.] | 0 |
| 41 | 525 | [0.] | 1 |
| 41 | 525 | [1.] | 0 |
| 41 | 525 | [1.] | 1 |
| 42 | 525 | [0.] | 0 |
| 42 | 525 | [1.] | 0 |
| 42 | 525 | [1.] | 0 |
| 42 | 525 | [1.] | 1 |
| 43 | 525 | [0.] | 0 |
| 43 | 525 | [0.] | 1 |
| 43 | 525 | [0.] | 1 |
| 43 | 525 | [0.] | 0 |
| 44 | 525 | [0.] | 0 |
| 44 | 525 | [0.] | 1 |
| 44 | 525 | [1.] | 0 |
| 44 | 525 | [1.] | 1 |
| 45 | 525 | [0.] | 0 |
| 45 | 525 | [0.] | 1 |
| 45 | 525 | [0.] | 1 |
| 45 | 525 | [0.] | 0 |
| 46 | 525 | [0.] | 0 |
| 46 | 525 | [1.] | 0 |
| 46 | 525 | [1.] | 0 |
| 46 | 525 | [0.] | 0 |
| 47 | 525 | [0.] | 0 |
| 47 | 525 | [0.] | 1 |
| 47 | 525 | [0.] | 1 |
| 47 | 525 | [0.] | 0 |
| 48 | 525 | [0.] | 0 |
| 48 | 525 | [1.] | 0 |
| 48 | 525 | [1.] | 0 |
| 48 | 525 | [0.] | 0 |
| 49 | 525 | [0.] | 0 |
| 49 | 525 | [0.] | 1 |
| 49 | 525 | [0.] | 1 |
| 49 | 525 | [0.] | 0 |
| 50 | 525 | [0.] | 0 |
| 50 | 525 | [0.] | 1 |
| 50 | 525 | [0.] | 1 |
| 50 | 525 | [0.] | 0 |

|  |  |  |  |
| --- | --- | --- | --- |
| Neuronas Capa Oculta | Epochs | Salida | Error |
| 10 | 50 | [1.] | 1 |
| 10 | 50 | [0.] | 1 |
| 10 | 50 | [1.] | 0 |
| 10 | 50 | [0.] | 0 |
| 10 | 75 | [0.] | 0 |
| 10 | 75 | [0.] | 1 |
| 10 | 75 | [0.] | 1 |
| 10 | 75 | [0.] | 0 |
| 10 | 100 | [0.] | 0 |
| 10 | 100 | [0.] | 1 |
| 10 | 100 | [0.] | 1 |
| 10 | 100 | [0.] | 0 |
| 10 | 125 | [1.] | 1 |
| 10 | 125 | [1.] | 0 |
| 10 | 125 | [1.] | 0 |
| 10 | 125 | [1.] | 1 |
| 10 | 150 | [1.] | 1 |
| 10 | 150 | [0.] | 1 |
| 10 | 150 | [1.] | 0 |
| 10 | 150 | [0.] | 0 |
| 10 | 175 | [1.] | 1 |
| 10 | 175 | [0.] | 1 |
| 10 | 175 | [1.] | 0 |
| 10 | 175 | [0.] | 0 |
| 10 | 200 | [0.] | 0 |
| 10 | 200 | [0.] | 1 |
| 10 | 200 | [0.] | 1 |
| 10 | 200 | [0.] | 0 |
| 10 | 225 | [0.] | 0 |
| 10 | 225 | [0.] | 1 |
| 10 | 225 | [0.] | 1 |
| 10 | 225 | [0.] | 0 |
| 10 | 250 | [1.] | 1 |
| 10 | 250 | [0.] | 1 |
| 10 | 250 | [1.] | 0 |
| 10 | 250 | [0.] | 0 |
| 10 | 275 | [0.] | 0 |
| 10 | 275 | [0.] | 1 |
| 10 | 275 | [1.] | 0 |
| 10 | 275 | [1.] | 1 |
| 10 | 300 | [1.] | 1 |
| 10 | 300 | [1.] | 0 |
| 10 | 300 | [0.] | 1 |
| 10 | 300 | [0.] | 0 |
| 10 | 325 | [0.] | 0 |
| 10 | 325 | [0.] | 1 |
| 10 | 325 | [0.] | 1 |
| 10 | 325 | [0.] | 0 |
| 10 | 350 | [0.] | 0 |
| 10 | 350 | [0.] | 1 |
| 10 | 350 | [1.] | 0 |
| 10 | 350 | [1.] | 1 |
| 10 | 375 | [0.] | 0 |
| 10 | 375 | [1.] | 0 |
| 10 | 375 | [0.] | 1 |
| 10 | 375 | [1.] | 1 |
| 10 | 400 | [0.] | 0 |
| 10 | 400 | [0.] | 1 |
| 10 | 400 | [0.] | 1 |
| 10 | 400 | [0.] | 0 |
| 10 | 425 | [0.] | 0 |
| 10 | 425 | [1.] | 0 |
| 10 | 425 | [1.] | 0 |
| 10 | 425 | [1.] | 1 |
| 10 | 450 | [0.] | 0 |
| 10 | 450 | [1.] | 0 |
| 10 | 450 | [1.] | 0 |
| 10 | 450 | [0.] | 0 |
| 10 | 475 | [0.] | 0 |
| 10 | 475 | [0.] | 1 |
| 10 | 475 | [0.] | 1 |
| 10 | 475 | [0.] | 0 |
| 10 | 500 | [0.] | 0 |
| 10 | 500 | [0.] | 1 |
| 10 | 500 | [1.] | 0 |
| 10 | 500 | [1.] | 1 |
| 10 | 525 | [0.] | 0 |
| 10 | 525 | [0.] | 1 |
| 10 | 525 | [0.] | 1 |
| 10 | 525 | [0.] | 0 |
| 10 | 550 | [0.] | 0 |
| 10 | 550 | [1.] | 0 |
| 10 | 550 | [0.] | 1 |
| 10 | 550 | [1.] | 1 |
| 10 | 575 | [0.] | 0 |
| 10 | 575 | [0.] | 1 |
| 10 | 575 | [0.] | 1 |
| 10 | 575 | [0.] | 0 |
| 10 | 600 | [1.] | 1 |
| 10 | 600 | [1.] | 0 |
| 10 | 600 | [0.] | 1 |
| 10 | 600 | [0.] | 0 |
| 10 | 625 | [0.] | 0 |
| 10 | 625 | [0.] | 1 |
| 10 | 625 | [0.] | 1 |
| 10 | 625 | [0.] | 0 |
| 10 | 650 | [0.] | 0 |
| 10 | 650 | [0.] | 1 |
| 10 | 650 | [0.] | 1 |
| 10 | 650 | [0.] | 0 |
| 10 | 675 | [0.] | 0 |
| 10 | 675 | [1.] | 0 |
| 10 | 675 | [0.] | 1 |
| 10 | 675 | [1.] | 1 |
| 10 | 700 | [0.] | 0 |
| 10 | 700 | [0.] | 1 |
| 10 | 700 | [0.] | 1 |
| 10 | 700 | [0.] | 0 |
| 10 | 725 | [0.] | 0 |
| 10 | 725 | [0.] | 1 |
| 10 | 725 | [0.] | 1 |
| 10 | 725 | [0.] | 0 |
| 10 | 750 | [0.] | 0 |
| 10 | 750 | [0.] | 1 |
| 10 | 750 | [0.] | 1 |
| 10 | 750 | [0.] | 0 |
| 10 | 775 | [0.] | 0 |
| 10 | 775 | [1.] | 0 |
| 10 | 775 | [1.] | 0 |
| 10 | 775 | [0.] | 0 |
| 10 | 800 | [0.] | 0 |
| 10 | 800 | [1.] | 0 |
| 10 | 800 | [1.] | 0 |
| 10 | 800 | [0.] | 0 |
| 10 | 825 | [0.] | 0 |
| 10 | 825 | [1.] | 0 |
| 10 | 825 | [1.] | 0 |
| 10 | 825 | [0.] | 0 |
| 10 | 850 | [0.] | 0 |
| 10 | 850 | [1.] | 0 |
| 10 | 850 | [1.] | 0 |
| 10 | 850 | [0.] | 0 |
| 10 | 875 | [0.] | 0 |
| 10 | 875 | [1.] | 0 |
| 10 | 875 | [1.] | 0 |
| 10 | 875 | [0.] | 0 |
| 10 | 900 | [0.] | 0 |
| 10 | 900 | [0.] | 1 |
| 10 | 900 | [0.] | 1 |
| 10 | 900 | [0.] | 0 |
| 10 | 925 | [0.] | 0 |
| 10 | 925 | [1.] | 0 |
| 10 | 925 | [1.] | 0 |
| 10 | 925 | [0.] | 0 |
| 10 | 950 | [0.] | 0 |
| 10 | 950 | [1.] | 0 |
| 10 | 950 | [1.] | 0 |
| 10 | 950 | [0.] | 0 |
| 10 | 975 | [0.] | 0 |
| 10 | 975 | [1.] | 0 |
| 10 | 975 | [1.] | 0 |
| 10 | 975 | [0.] | 0 |
| 10 | 1000 | [0.] | 0 |
| 10 | 1000 | [1.] | 0 |
| 10 | 1000 | [1.] | 0 |
| 10 | 1000 | [0.] | 0 |

#### Grafica RELU y SIGMOIDEA

Ambas tablas se representan gráficamente a continuación:

Icon

Description automatically generated

A picture containing histogram

Description automatically generated

### Conclusión de mejor función

Capa oculta RELU capa de salida SIGMOIDE

Icon

Description automatically generatedA picture containing text

Description automatically generated

Capa oculta SIGMOIDE capa de salida RELU

A picture containing histogram

Description automatically generatedIcon

Description automatically generated

Capa oculta SIGMOIDE capa de salida SIGMOIDE

Text

Description automatically generatedA picture containing text

Description automatically generated

Capa oculta RELU capa de salida RELU

Text

Description automatically generatedText, icon

Description automatically generated

Como podemos observar la única combinación de funciones de activación que dan resultados estables de clasificación son Capa oculta RELU capa de salida SIGMOIDE, las demás solo muestran picos de estabilidad y no se puede apreciar ninguna tendencia.

# Práctica 3

## Introducción

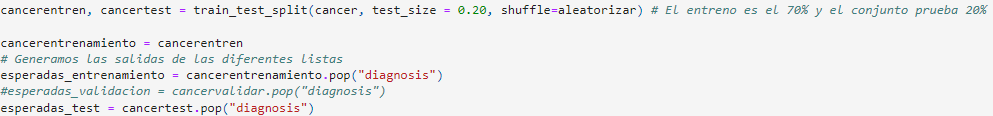
En esta práctica 3 se nos da un dataset con diferentes datos acerca del cáncer y su importancia, dándonos la resolución de un caso, indicándonos en cada uno si es importante o no. Debemos realizar está utilizando Keras/TensorFlow como en el ejercicio anterior, pero demostrando en este caso poderlo aplicar a un caso real y más serio que la resolución de una operación lógica.

## Cuestiones

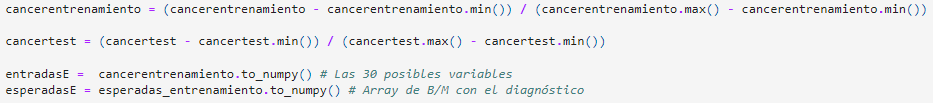
### Proceso de categorización y normalización de los datos

Antes de pasar a realizar el estudio del cáncer es necesario realizar tanto una limpieza como una normalización y categorización de los datos. De esta manera, podremos fácilmente operar con keras.

Primero de todo realizaremos una división de los datos entre el conjunto de entrenamiento y el conjunto de predicción. Esta división será de 70% / 20%70% / 20%. Hay que matizar que, para realizar esta división, se hace de forma aleatoria.



Una vez tenemos los datos divididos, normalizaremos todas las columnas de ambos dataset. Esta normalización la haremos de tal manera que el máximo valor de la columna sea 1 y de ahí una regla de tres para el resto. Además, una vez estén normalizadas las columnas, las pasaremos a un *numpy array* para que keras pueda procesarlo correctamente.



Por último, realizaremos una categorización de los datos de diagnóstico. La categorización consiste en pasar del diagnóstico en letras (M o B) a una salida binaria. En este caso la M será una salida de 1 y la B una salida de 0. Esto nos facilita el trabajo con keras, que solo permite este tipo de salidas para los MLP. A continuación, se adjunta el código de la categorización:

A screenshot of a computer

Description automatically generated with medium confidence

### Estudio de arquitectura

Para estudiar la arquitectura óptima hemos realizado un automatizador para encontrar los valores óptimos para la red. Los valores que estamos buscando son el número de capas, el número de neuronas por capa y el número de epoch.

Capas ocultas

Salida

Capa de salida

Capa de entrada

…

Entradas

Este es el esquema que estamos modelando. La capa de entrada tiene tantas neuronas como características tienen los datos. El número de capas ocultas y de neuronas por capa lo vamos cambiando hasta encontrar el más adecuado y la última capa tiene una sola neurona con la función de activación sigmoide. Para encontrar el valor más adecuado, usábamos una lista de opciones que nos parecían viables, y vamos creando una nueva red cada vez y guardando el loss y la precisión(accuracy) de cada iteración, para posteriormente mostrarlo en una gráfica. Aquí muestro las gráficas de las diferentes variaciones con la correspondiente tabla de datos.

**Búsqueda de número de neuronas:**

A picture containing graphical user interface

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Elemento | Loss | Accuracy |
| 0 | 0.04323010966181755 | 0.9385964870452881 |
| 1 | 0.03302073366940021 | 0.9473684430122375 |
| 2 | 0.02264605201780796 | 0.9736841917037964 |
| 3 | 0.021064944565296173 | 0.9385964870452881 |
| 4 | 0.025049705244600772 | 0.9298245906829834 |
| 5 | 0.01903420966118574 | 0.9736841917037964 |
| 6 | 0.01720892544835806 | 0.9736841917037964 |

Como podemos ver el número de neuronas por capa no afecta demasiado al resultado de la red. Se ve que ligeramente con 15 neuronas por capa es mejor, por lo tanto usaremos ese valor para la red final.

**Búsqueda de número de capas:**

Graphical user interface

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Elemento | Loss | Accuracy |
| 1 | 0.02046978697180748 | 0.9561403393745422 |
| 2 | 0.01814328636974096 | 0.9824561476707458 |
| 3 | 0.11516766306012868 | 0.9736841917037964 |
| 4 | 0.11636080816388131 | 0.9649122953414917 |
| 5 | 0.20245255753397942 | 0.5789473652839661 |

Con esta gráfica podemos ver muy claramente que a partir de las 14 capas ocultas se produce un overfitting y la exactitud de la red decae drásticamente. Lo que en la leyenda llamamos error es realmente el loss de la red.

En este caso, la lista de neuronas probadas es: [2, 6, 10, 14, 18]

**Búsqueda de número de iteraciones:**

**A picture containing line chart

Description automatically generated**

|  |  |  |
| --- | --- | --- |
| Elemento | Loss | Accuracy |
| 0 | 0.07338605970144271 | 0.8684210777282715 |
| 1 | 0.08748536407947541 | 0.9561403393745422 |
| 2 | 0.03174797594547272 | 0.9912280440330505 |
| 3 | 0.05002635344862938 | 0.8859649300575256 |
| 4 | 0.03871805630624294 | 0.9736841917037964 |
| 5 | 0.04657958298921585 | 0.9561403393745422 |
| 6 | 0.03183360323309899 | 0.9649122953414917 |
| 7 | 0.06852509379386902 | 0.9561403393745422 |
| 8 | 0.11421036198735238 | 0.9649122953414917 |
| 9 | 0.01865769289433956 | 0.9649122953414917 |
| 10 | 0.027789442241191863 | 0.9736841917037964 |
| 11 | 0.027141755819320677 | 0.9561403393745422 |
| 12 | 0.026569827646017074 | 0.9561403393745422 |
| 13 | 0.02550237365067005 | 0.9649122953414917 |
| 14 | 0.03071775995194912 | 0.9736841917037964 |
| 15 | 0.06590496934950352 | 0.9736841917037964 |
| 16 | 0.028716180101037027 | 0.9824561476707458 |
| 17 | 0.021175287663936615 | 0.9824561476707458 |

En esta tabla podemos ver que en general, en cuanto superamos las 125 iteraciones, la precisión del modelo se mantiene más o menos constante.

### Error de la arquitectura

Usando la arquitectura óptima, es decir:

* Número de neuronas por capa: 15
* Número de capas: 13
* Epoch o número de iteraciones: 200

Con esta arquitectura, hemos obtenido el resultado óptimo, con un balance entre la exactitud y el número de iteraciones más bajo necesario, sin llegar al overfitting. Hemos reservado un 30% de los datos del dataset originar para calcular el error obtenido en la evaluación de los datos de testeo. Con esta arquitectura, el loss ha sido de 0.10733694583177567 y el accuracy ha sido de 0.8947368264198303. Dibujamos la gráfica obtenida al entrenar la red en función del epoch.

A picture containing graphical user interface

Description automatically generated

### Influencia de las variables

Toda la información de este apartado ha sido extraída del siguiente artículo científico (Sopena, 2008) donde se estudia la realización de la selección de funciones con perceptrones multicapa.

Una vez tenemos todas las cuestiones anteriores resueltas es el momento de encontrar cual es la influencia real de las variables de cáncer. El problema es el siguiente: Dado un conjunto de n características, seleccione un subconjunto que funcione mejor bajo ciertos criterios de evaluación. Desde el punto de vista computacional, la definición de esto conduce a un problema de búsqueda en un espacio de elementos. Por lo tanto, se deben especificar dos componentes: la búsqueda procedimiento a través del espacio de subconjuntos de características y la característica criterio de evaluación del subconjunto.

Idealmente, se espera que el rendimiento de la red mejore hasta que quede un subconjunto de características para las cuales la eliminación de más variables da como resultado una degradación del rendimiento.

Es por ello por lo que como bien dictamina el artículo, el mejor algoritmo es el siguiente:

Graphical user interface, text, application, email

Description automatically generated

De esta manera podemos observar cómo los pasos que hay que seguir para la elección de las mejores características son los siguientes:

1. La red es entrenada con el conjunto de características disponibles obteniendo un valor de error. A este lo llamaremos: *error\_red*
2. De forma iterativa se van eliminando una a una las características y reentrenando la red. Cada vez que se elimina una y se reentrena, se guarda el error de la red sin esa característica en *errores\_caracteristicas*
3. Si el menor error de *errores\_caracteristicas* es menor que *error\_red* significa que esa característica no era significativa y se elimina definitivamente
4. Se repite el proceso hasta que el menor error de *errores\_caracteristicas* sea un 10% mayor que *error\_red*

El cálculo de cuál es la variable para eliminar es alto, pero en el artículo se realiza un estudio y se concluye que los resultados experimentales en conjuntos de datos artificiales y de referencia indican que el aumento en el coste computacional asociado con volver a entrenar la red con cada función eliminada temporalmente antes de calcular su prominencia se recompensa con una mejora significativa del rendimiento.

Una vez sabemos la teoría del algoritmo a desarrollar, procedemos a su codificación. A continuación, se adjuntan dos capturas del algoritmo en su totalidad, basado en el pseudocódigo del artículo científico ya mencionado y realizando ligeros cambios para adaptarlo a nuestra práctica. Hay que mencionar que obviamente es muy costoso a nivel de tiempo su ejecución, pero como bien hemos comentado, la reducción de la dimensionalidad supone un cambio muy notorio a la hora de ejecución de las posibles pruebas.

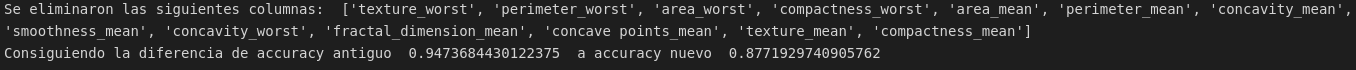
Text

Description automatically generated

Text

Description automatically generated

A continuación, mostramos la ejecución de nuestro algoritmo, que ha tardado en ejecutarse 250 minutos y que ha dado como resultado la eliminación de 13 columnas, y suponiendo un 45% menos de dimensionalidad y solo asumiendo un 10% del accuracy. Como podemos ver la exactitud de la red antes de la reducción era de un 94%, tras reducir la red un 45% obtenemos una exactitud del 88%, lo cual es un cambio muy aceptable para casi un 50% menos de cómputo.



# Conclusión

Una vez concluidas las tres practicas hemos logrado comprender y asimilar la teoría vista en clase, asociando los conocimientos a problemas de la vida real utilizando métodos de inteligencia artificial.

# Bibliografía

Brownlee, J. (s.f.). *Display Deep Learning Model Training History in Keras*. Obtenido de https://machinelearningmastery.com/display-deep-learning-model-training-history-in-keras/

Burgdorf, C. (s.f.). *Understanding XOR with Keras and TensorFlow*. Obtenido de https://blog.thoughtram.io/machine-learning/2016/11/02/understanding-XOR-with-keras-and-tensorlow.html

Damavis. (s.f.). Obtenido de https://blog.damavis.com/el-perceptron-simple-implementacion-en-python/

https://ichi.pro/es/7-funciones-de-activacion-populares-que-debes-conocer-en-deep-learning-y-como-usarlas-con-keras-y-tensorflow-2-43657675202534. (s.f.). *7 funciones de activación populares que debes conocer en Deep Learning y cómo usarlas con Keras y TensorFlow 2*.

Keras. (s.f.). Obtenido de https://keras.io/

Rosebrock, A. (s.f.). Obtenido de https://pyimagesearch.com/2019/02/18/breast-cancer-classification-with-keras-and-deep-learning/

Sopena, E. R. (2008). *Performing Feature Selection.* Obtenido de https://www.cs.upc.edu/~eromero/Publications/Downloads/2008-tnn.pdf

Tensorflow. (s.f.). Obtenido de https://www.tensorflow.org/?hl=es-419

unipython. (s.f.). Obtenido de https://unipython.com/rendimiento-de-los-modelos-de-deep-learning/