

Redes Neuronales

The background of the slide is a photograph of a clear night sky. The Milky Way galaxy is visible as a bright, hazy band of light stretching across the upper half of the image. Numerous individual stars are scattered throughout the dark blue sky. In the lower portion of the image, the dark, silhouetted branches of evergreen trees are visible, framing the bottom and right sides of the scene.

Aprendizaje Automático - Esquema

Ejemplos
(etiquetados)

X				Y
sepal_length	sepal_width	petal_length	petal_width	name
5	2	3,5	1	1
6	2,2	4	1	1
4,7	3,2	1,3	0,2	0
5,1	3,8	1,6	0,2	0
4,6	3,2	1,4	0,2	0

Modelo

Parámetros
(aleatorios)

Algoritmo de
Aprendizaje

- Descenso de Gradiente
- otros

Error a optimizar

- Error cuadrático
- Entropía
- Otros

Ejemplos nuevos

X			
sepal_length	sepal_width	petal_length	petal_width
5	2	3,5	1
6	2,2	4	1
4,7	3,2	1,3	0,2

Modelo

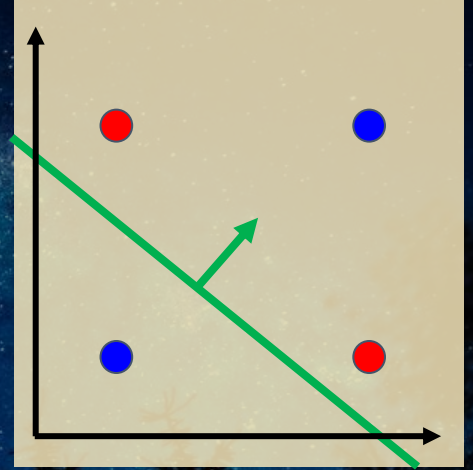
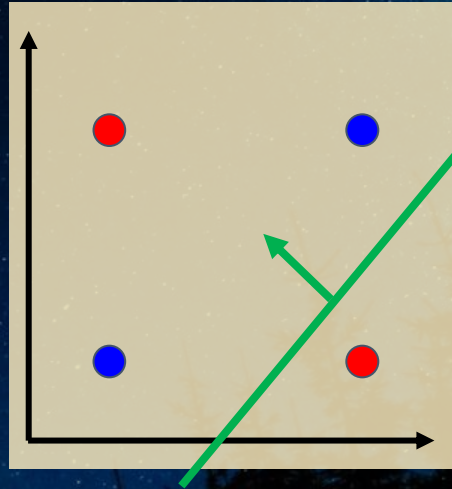
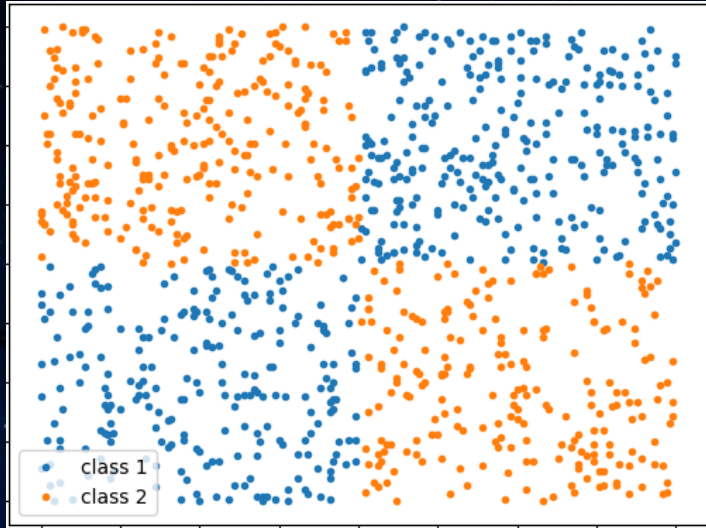
Parámetros
(entrenados)

Predicciones

Y
name
1
1
0
0

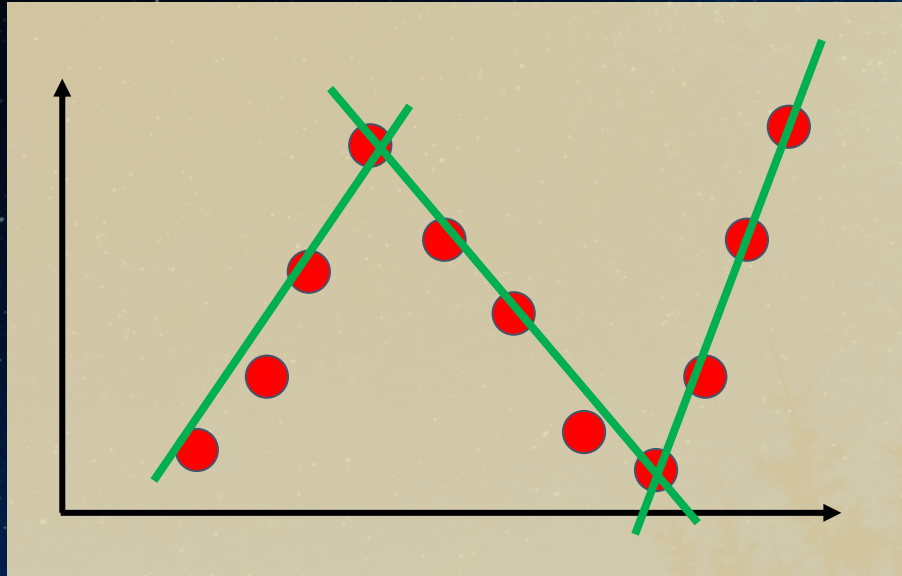
Límite de modelo lineales

El problema del XOR



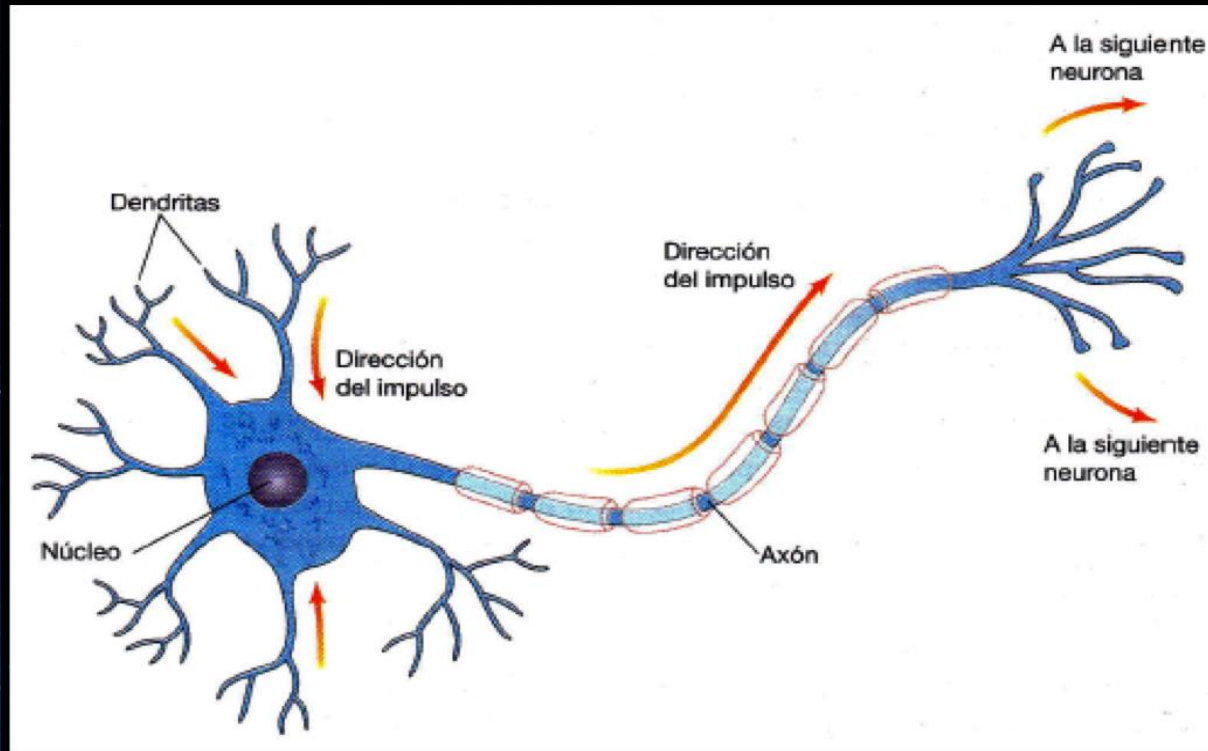
Límite de modelo lineales

Regresión no lineal



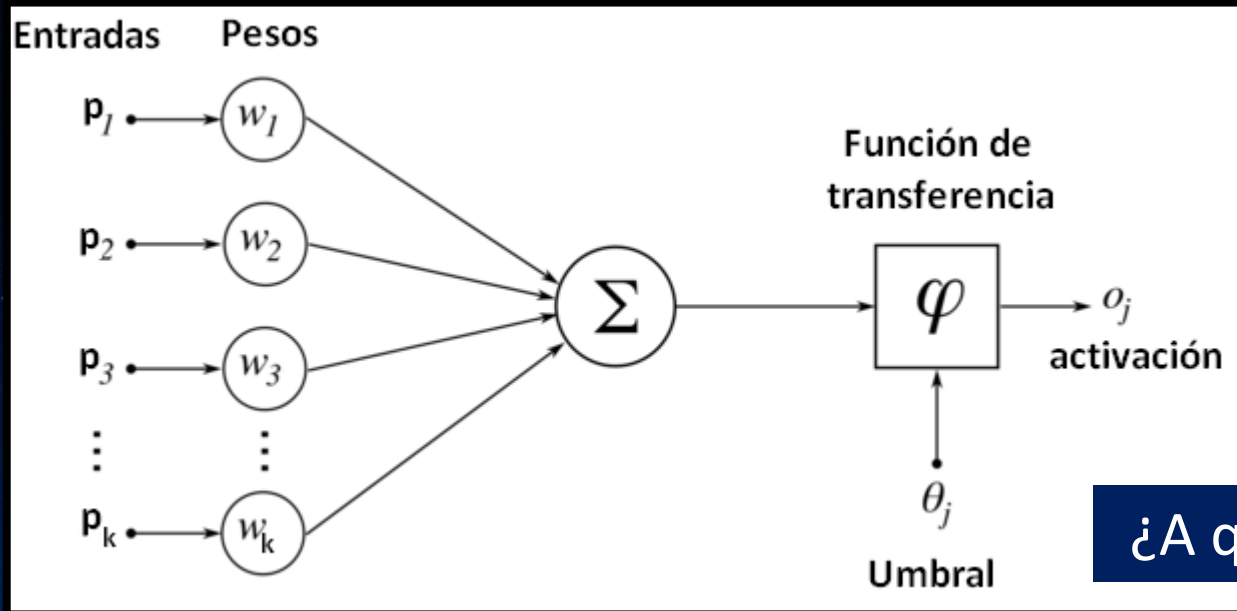
Redes Neuronales Artificiales

Analogía con las redes neuronales biológicas



Redes Neuronales Artificiales

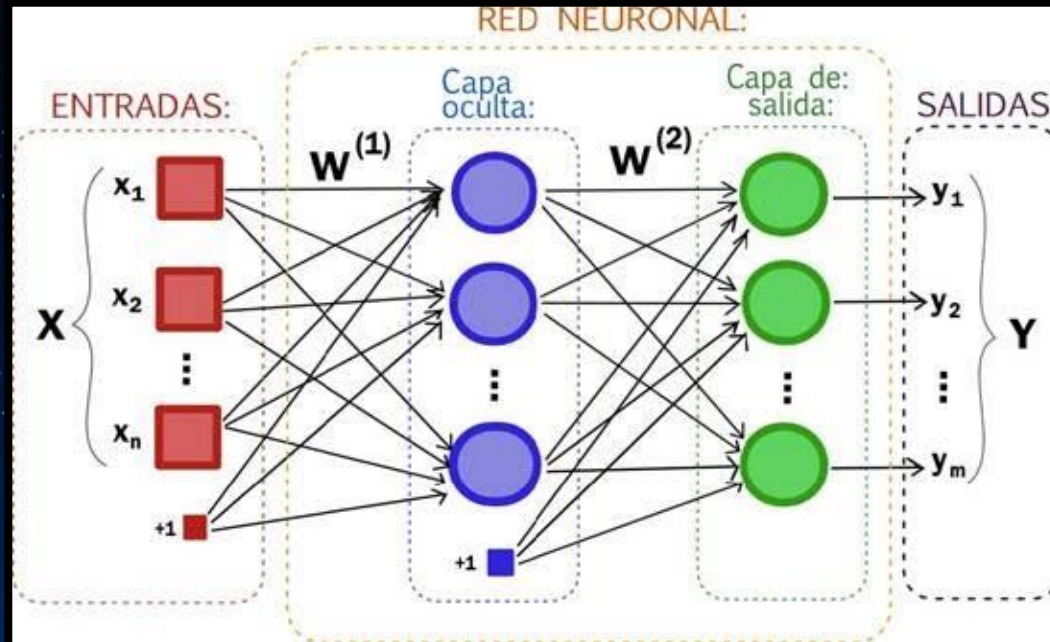
Concepto de Neurona Artificial (Perceptrón)



¿A qué se parece?

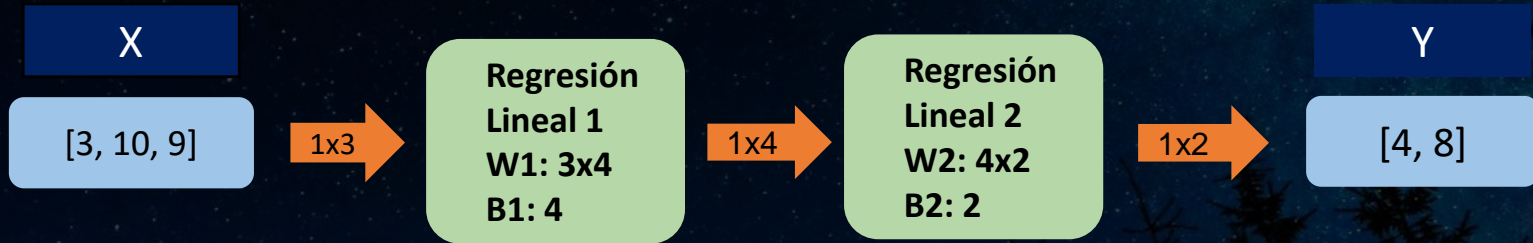
Redes Neuronales Artificiales

Una red neuronal no es más que muchos regresores logísticos conectados de diversas maneras. Se le conoce como “Perceptrón Multicapa”, o “Feedforward Neural Network”.



Redes Neuronales Artificiales

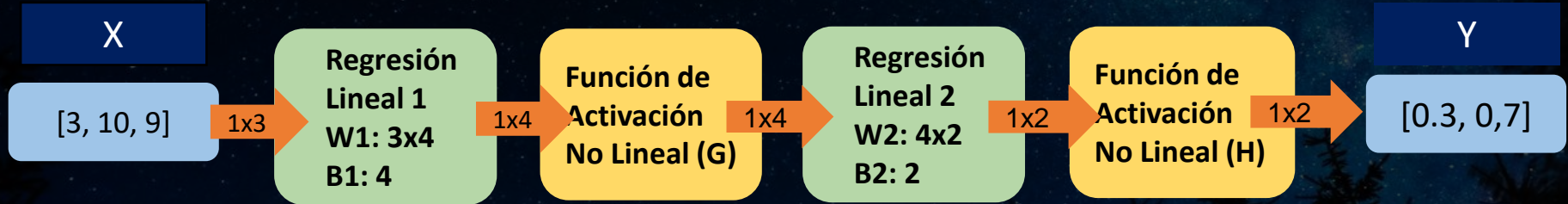
Una representación más moderna.



$$Y = ((X W1 + B1) W2 + B2)$$

Redes Neuronales Artificiales



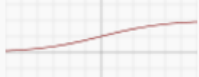



Una representación más moderna.



$$Y = H(G(X W1 + B1) W2 + B2)$$

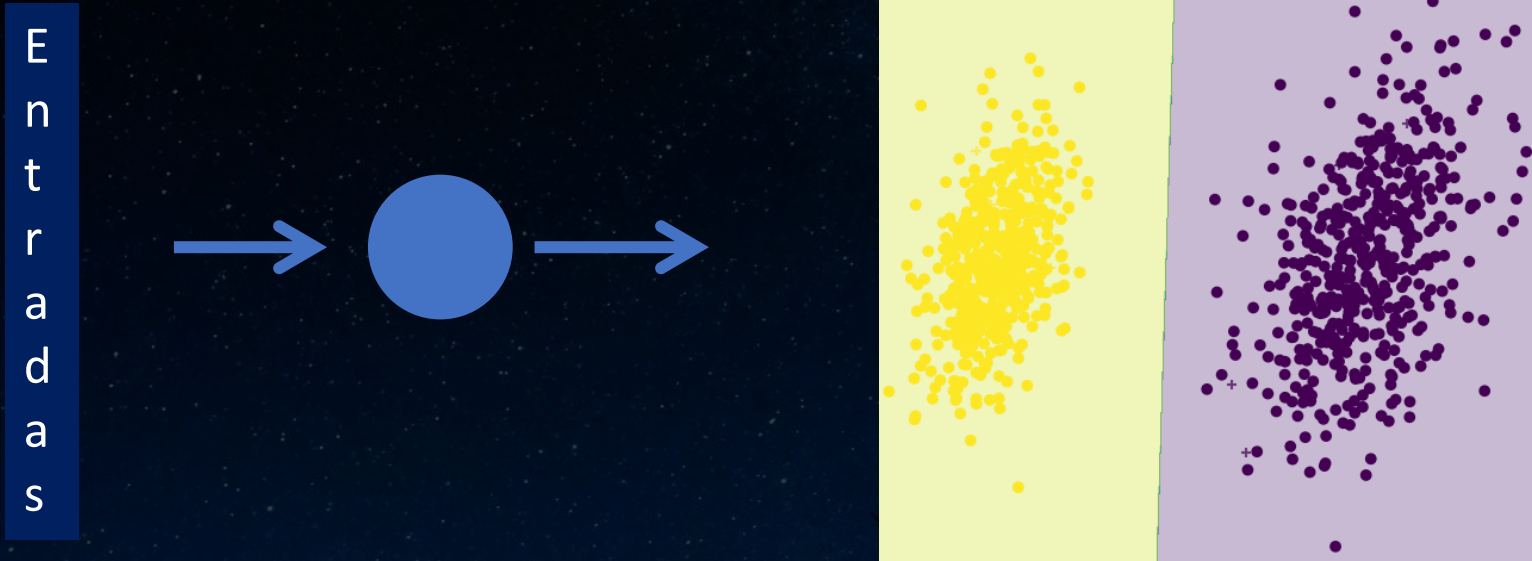
Redes Neuronales Artificiales

Funciones de transferencia/activación para las capas ocultas.

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$

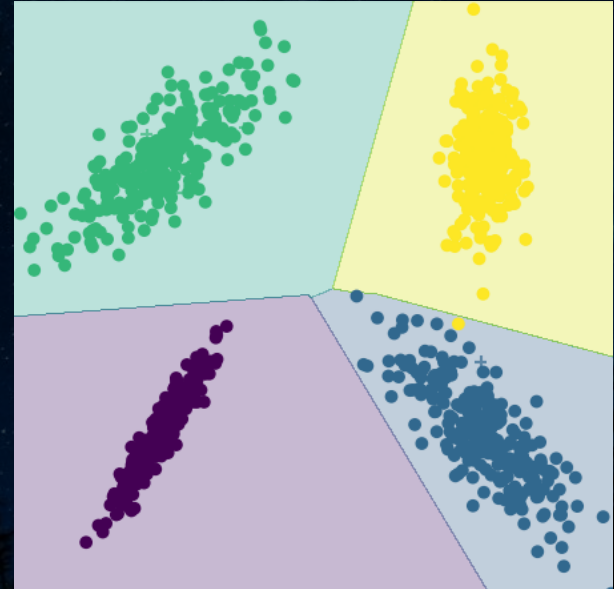
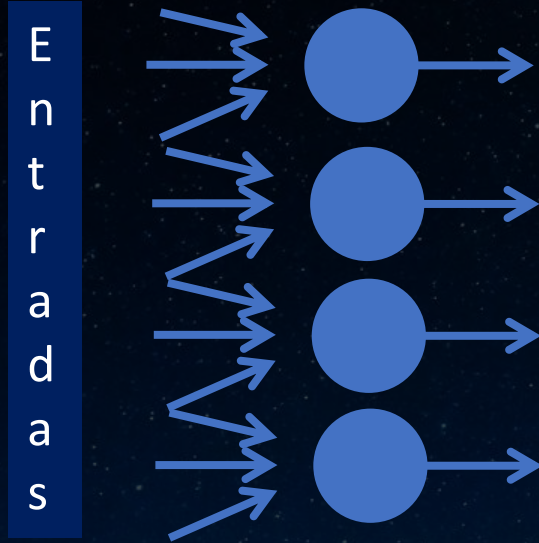
Redes Neuronales Artificiales

Cada neurona no es más que un combinador lineal (regresor logístico).
Para clasificar dos clases sólo necesitamos una neurona de salida.



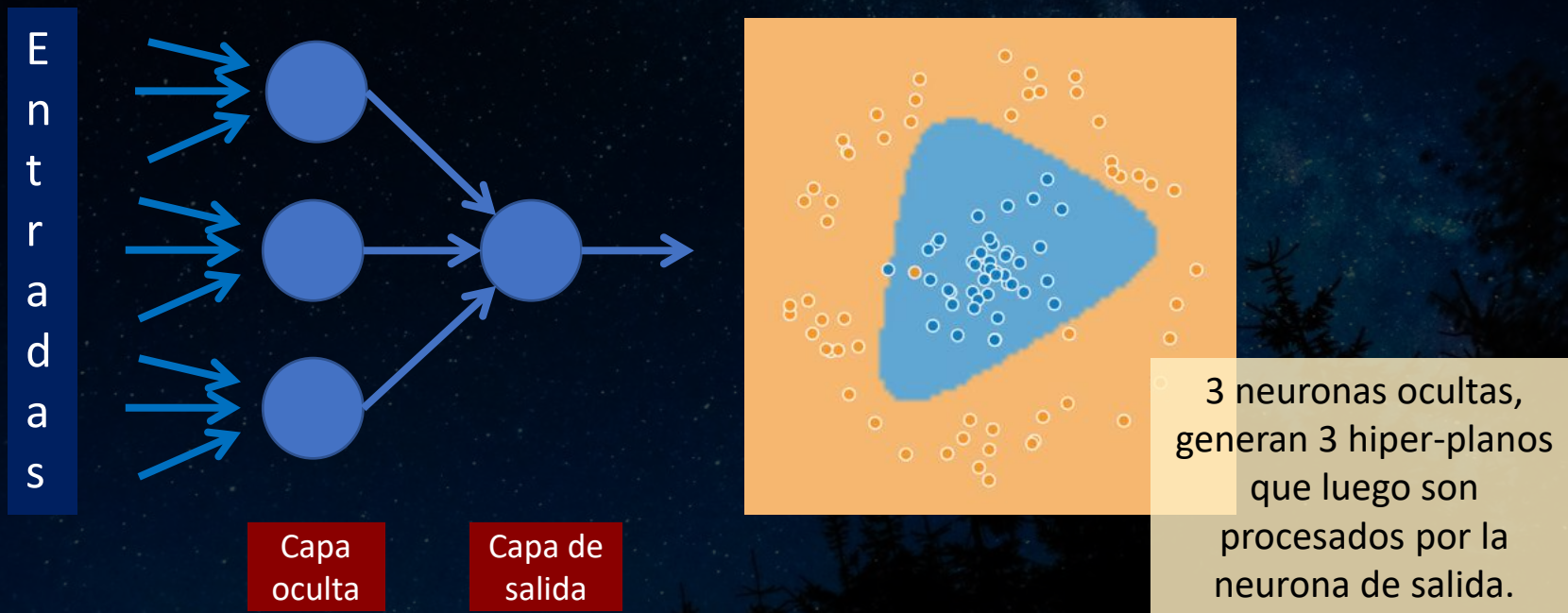
Redes Neuronales Artificiales

Una red neuronal que permite clasificar varias clases tiene tantas neuronas de salida como clases.

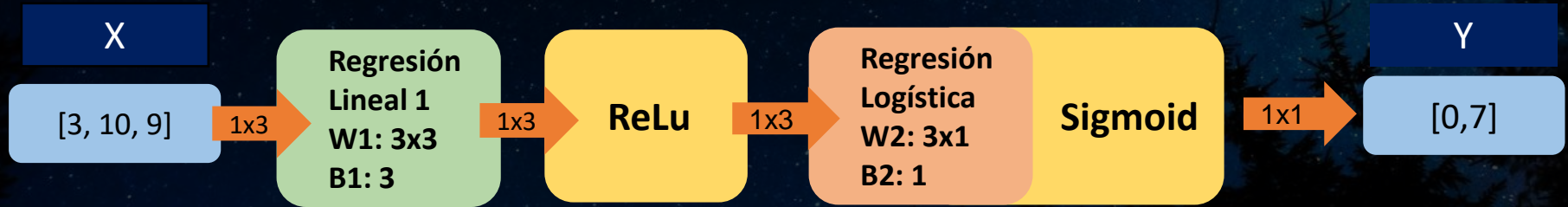
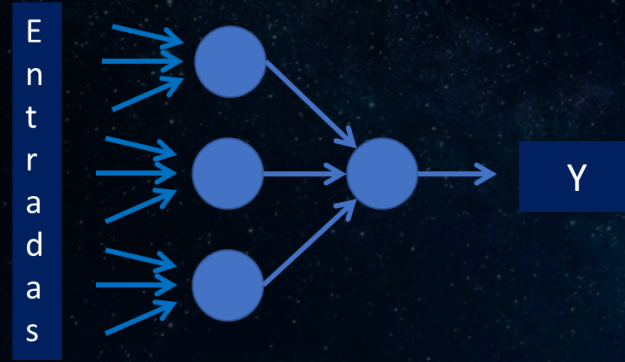


Redes Neuronales Artificiales

Cómo hace una red neuronal para clasificar patrones que no son linealmente separables?: Capas ocultas!

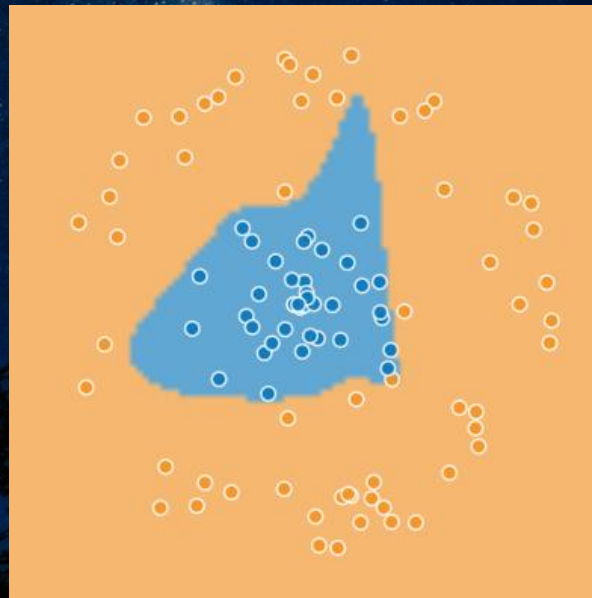
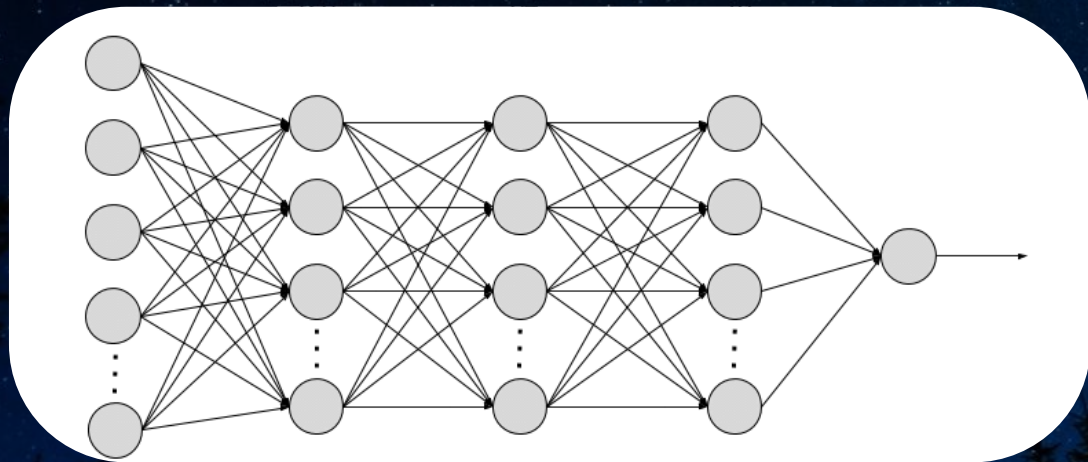


Redes Neuronales Artificiales



Complejidad del modelo

Cuanto más grande es la red (más capas y más neuronas por capa) permite modelar polinomios más complejos. A veces ayuda, pero también podemos caer en mínimos locales (no converger) o en overfitting.

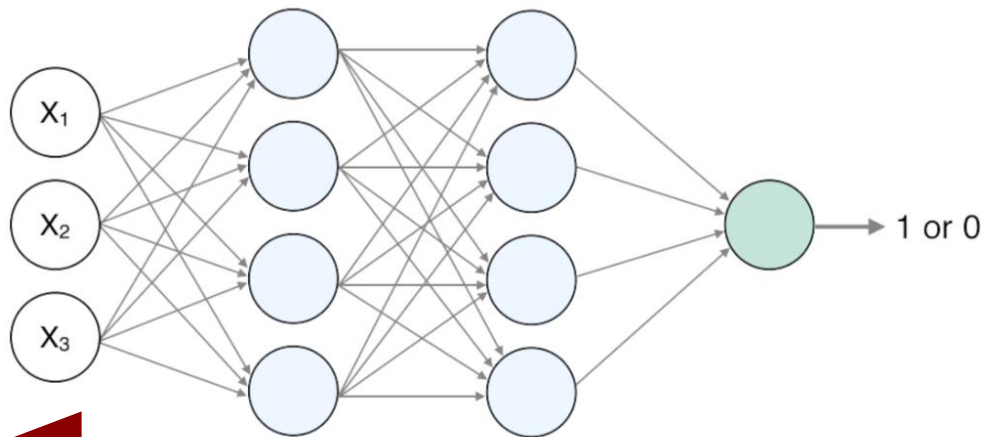


Redes Neuronales Artificiales - Entrenamiento

Backpropagation: Entrenamiento similar a Regresión Logística.

El error sólo se puede calcular en la última capa, y luego se va “propagando” hacia las capas anteriores, actualizando los pesos de la red.

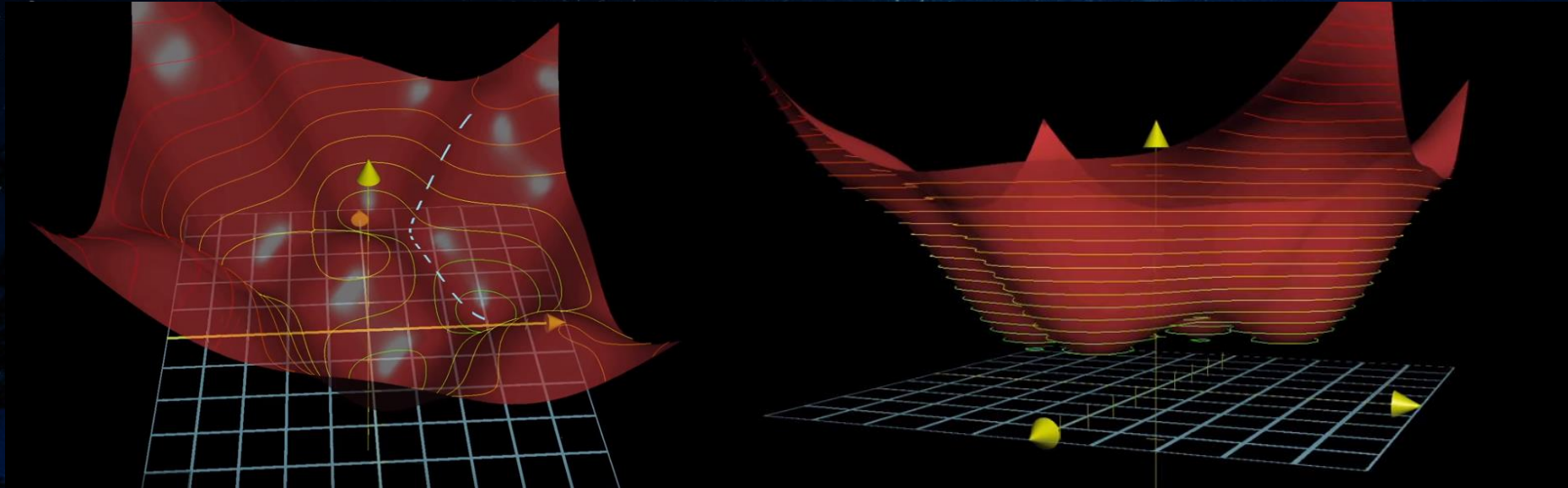
Solo necesitamos que cada capa sea derivable y aplicamos regla de la cadena.



Propagación del error

Redes Neuronales Artificiales - Entrenamiento

La función de error ya no es convexa.
El descenso debe sortear mínimos locales.



Visualización de Redes Neuronales

Red entrenada para clasificar imágenes de números:

<http://scs.ryerson.ca/~aharley/vis/fc/>

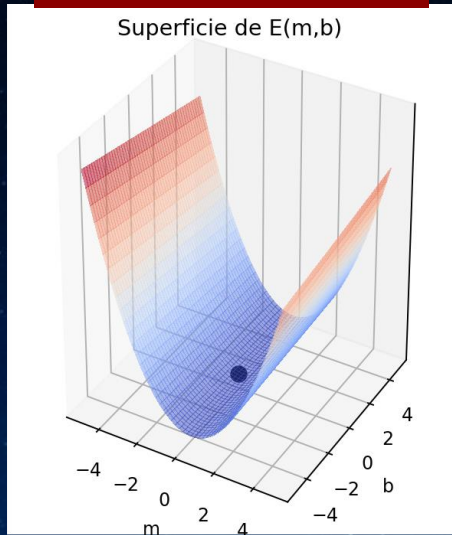
Google. Clasificación o Regresión simple:

<https://playground.tensorflow.org/>

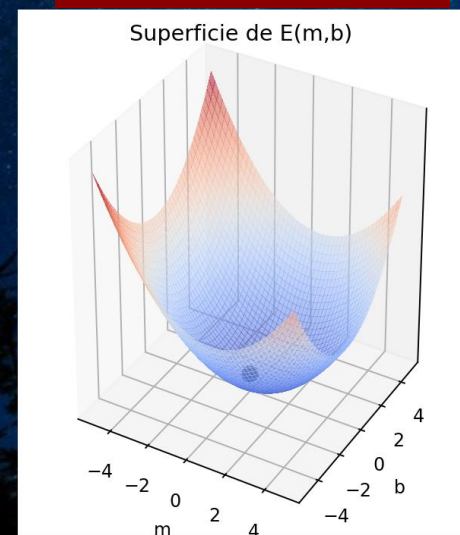
Normalización

Las redes Neuronales son muy sensibles a la escala de los datos. Distintos rangos en las variables puede generar curvas de error difíciles de optimizar.

Sin normalización



Normalizado



Redes Neuronales en Scikit-Learn

```
from sklearn.neural_network import MLPClassifier
```

```
LAYERS_SIZES= (20, 5)
```

Dos capas ocultas, la primera de 20, la segunda de 5 neuronas.

La cant neuronas de salida depende del problema

```
modelo= MLPClassifier(hidden_layer_sizes=LAYERS_SIZES)
```

```
modelo.fit(x,y)  
y_pred= modelo.predict(x)
```

Igual que antes

```
w= modelo.coefs_  
b= modelo.intercepts_
```

Redes Neuronales en Scikit-Learn

```
from sklearn.neural_network import MLPClassifier
```

```
LAYERS_SIZES= (20, 5)
```

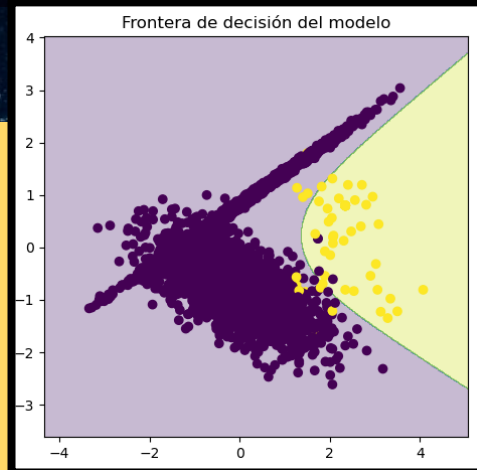
```
modelo= MLPClassifier(hidden_layer_sizes=LAYERS_SIZES)
```

```
modelo.fit(x,y)
```

```
y_pred= modelo.predict(x)
```

```
w= modelo.coefs_
```

```
b= modelo.intercepts_
```



Índice	Tipo	Tamaño	Valor	W
0	Array of float64	(2, 20)	[[0.05390301 0.28123331 -0.53185908 ... 0.29553219... -0 ...	
1	Array of float64	(20, 5)	[[0.44405885 -0.14717654 -0.1061151 0.11926259 0... [-0.45 ...	
2	Array of float64	(5, 1)	[[-1.65641551] [-1.78752241]	

Índice	Tipo	Tamaño	Valor	b
0	Array of float64	(20,)	[0.36207107 0.24910854 0.23401538 ... 0.4545... -0.0 ...	
1	Array of float64	(5,)	[0.3953002 1.08168394 0.1246819 0.38941156 -0.5136422]	
2	Array of float64	(1,)	[-1.74008599]	

Parametrización del modelo

Algunos parámetros a tener en cuenta:

learning_rate_init : default 0.001. The initial learning rate used. It controls the step-size in updating the weights. Only used when solver='sgd' or 'adam'.

activation : default 'relu'. Activation function for the hidden layer. {'identity', 'logistic', 'tanh', 'relu'},

tol: default 1e-4. Tolerance for the optimization. When the loss or score is not improving by at least tol for two consecutive iterations, unless learning_rate is set to 'adaptive', convergence is considered to be reached and training stops.

max_iter: default=200. Maximum number of iterations.

verbose: default=False. Whether to print progress messages to stdout.

Redes Neuronales: Regresión

```
from sklearn.neural_network import MLPRegressor  
modelo= MLPRegressor(hidden_layer_sizes=LAYERS_SIZES,activation ='tanh')
```

- Las redes Neuronales también pueden utilizarse para problemas de Regresión.
- Solo cambiamos la función de activación de la capa de salida.
- En vez de Sigmoid, Lineal

