Fundamentos de las Redes Neuronales

Algoritmo de Retropropagación

Clase 5

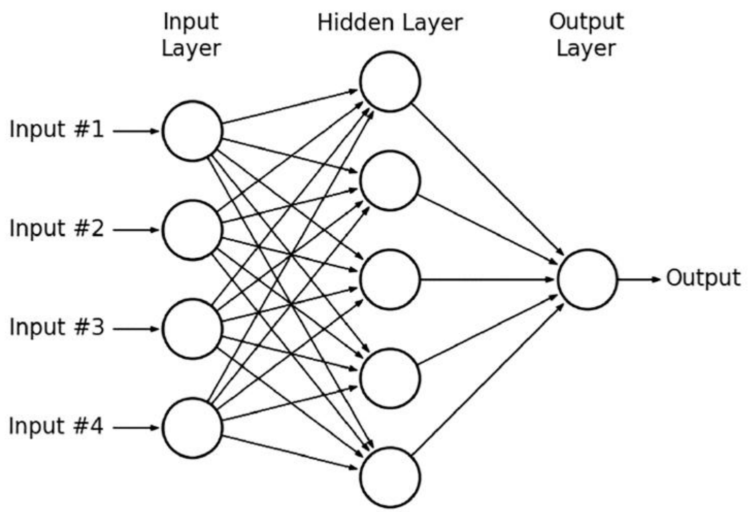
Dra. Wendy Aguilar

Modelos Generativos Profundos

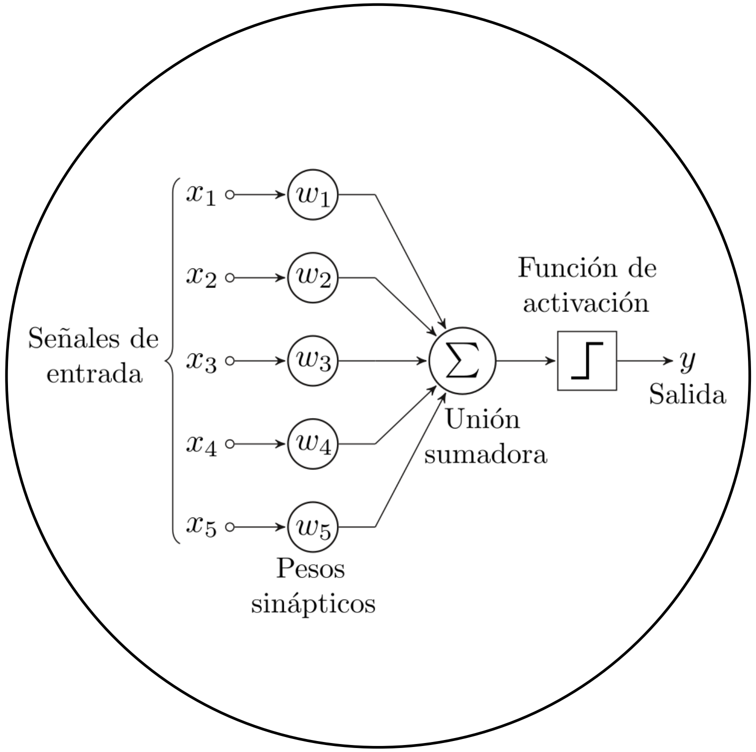
UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

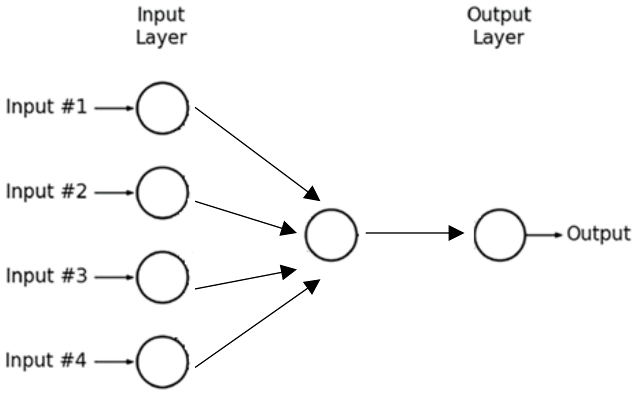
¿cómo podemos entrenar un perceptrón con capas ocultas de forma eficiente?

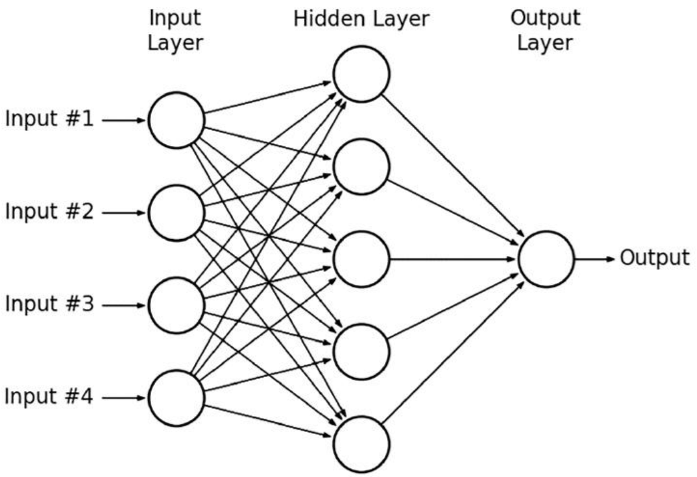
¿cómo actualizar los pesos y sesgos?

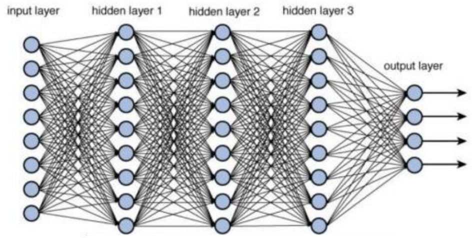


Perceptrones Multicapa y Aprendizaje de Pesos









En el perceptrón, una regla de aprendizaje para actualizar pesos es relativamente fácil de derivar, mientras no existan capas ocultas.

La entrada se transforma una sola vez por el perceptrón para calcular un valor de salida → los pesos pueden ajustarse directamente para producir la salida.

Cuando existen capas ocultas entre la entrada y salida, el problema se vuelve más complejo.

¿cuándo debemos modificar los pesos internos para calcular las activaciones que alimentan la salida final?

¿Cómo debemos ajustarlos en relación con los pesos de entrada?

Perceptrones Multicapa y Retropropagación

- En la década de los **80s**, el **algoritmo de retropropagación** surgió como una solución práctica para el cálculo de los pesos ocultos.

La idea clave es que:

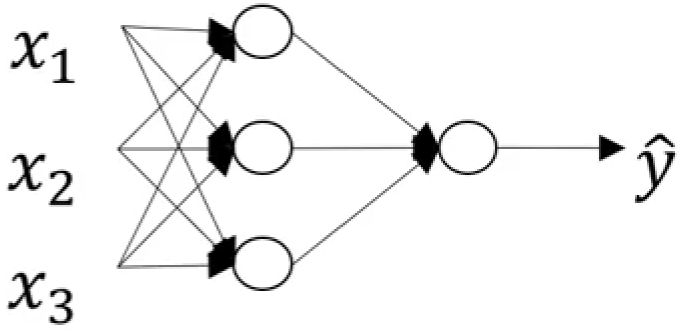
- podemos utilizar la **regla de la cadena** del cálculo diferencial para **calcular de manera eficiente las derivadas de cada parámetro** de una red con respecto a **una función de pérdida**,
- y **combinándola con una regla de aprendizaje**, obtener un método escalable para entrenar redes multicapa.

Notación

* Se le llama oculta porque sus valores son desconocidos (no observables) para el conjunto de entrenamiento.

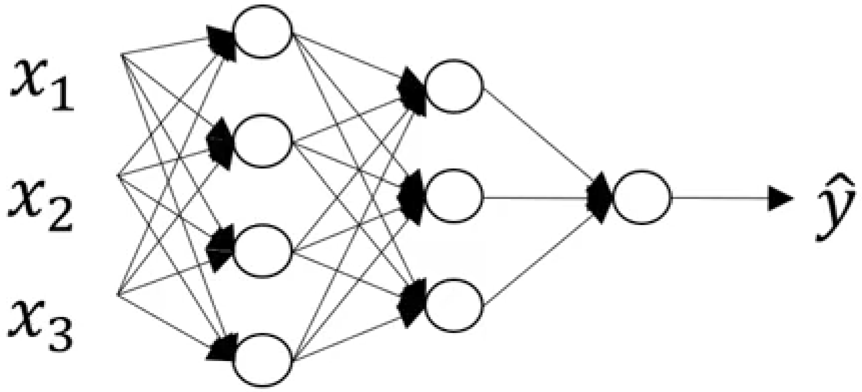
Capa de entrada | Capa oculta | Capa salida

Número de capa

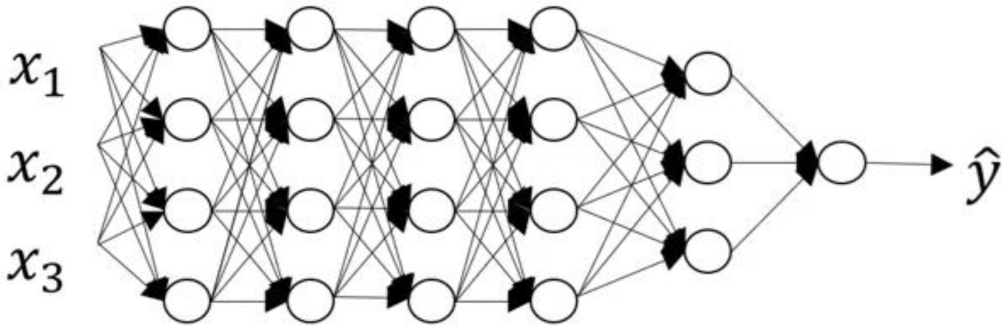


RN con 1 capa oculta y 2 capas totales

No se cuenta la capa de entrada.
Solo se cuentan las capas con parámetros entrenables.



RN con ? capas ocultas y ? capas totales

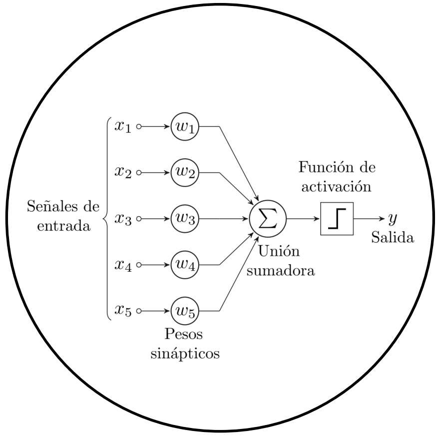


RN con ? capas ocultas y ? capas totales

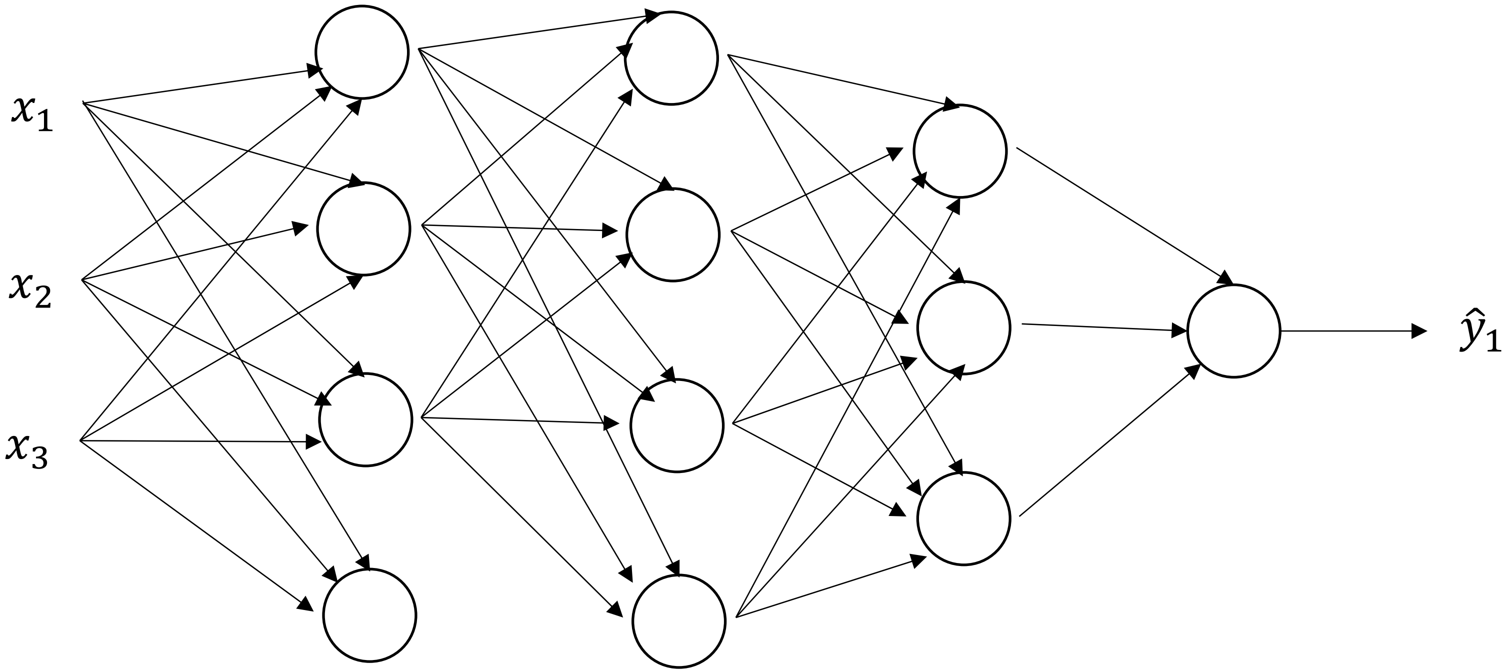
Notación

RN de 4 capas

$L = número\ de\ capas = 4$



Capa de entrada	Capa oculta 1	Capa oculta 2	Capa oculta 3	Capa salida 4



$número\ de\ neuronas\ de\ la\ capa\ l$

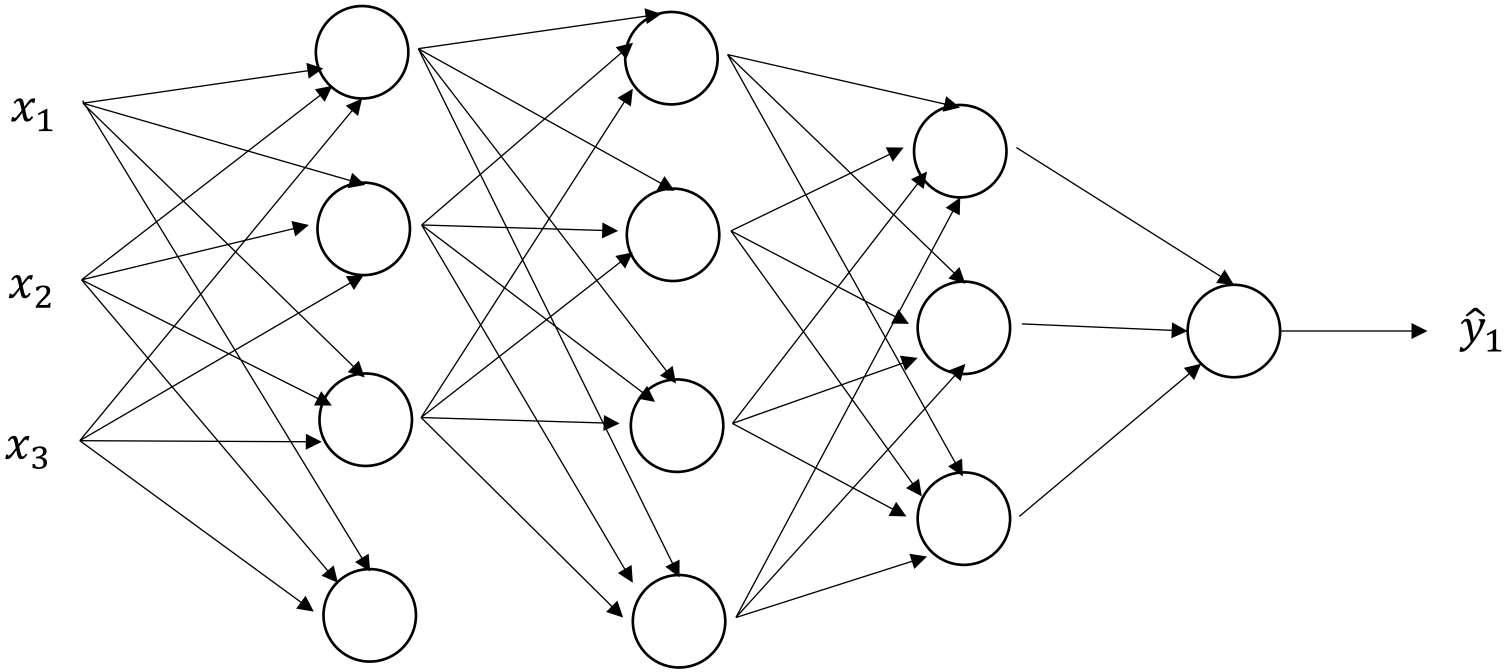
4 neuronas	4 neuronas	3 neuronas	1 neurona
$n^{[1]} = 4$	$n^{[2]} = 4$	$n^{[3]} = 3$	$n^{[4]} = 1$

Notación

RN de 4 capas

$L = número\ de\ capas = 4$

Capa de entrada	Capa oculta	Capa oculta	Capa oculta	Capa salida
0	1	2	3	4



	4 neuronas	4 neuronas	3 neuronas	1 neurona
$n^{[0]} = n_X = 3$	$n^{[1]} = 4$	$n^{[2]} = 4$	$n^{[3]} = 3$	$n^{[4]} = 1$

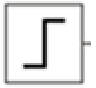
Notación

RN de 4 capas

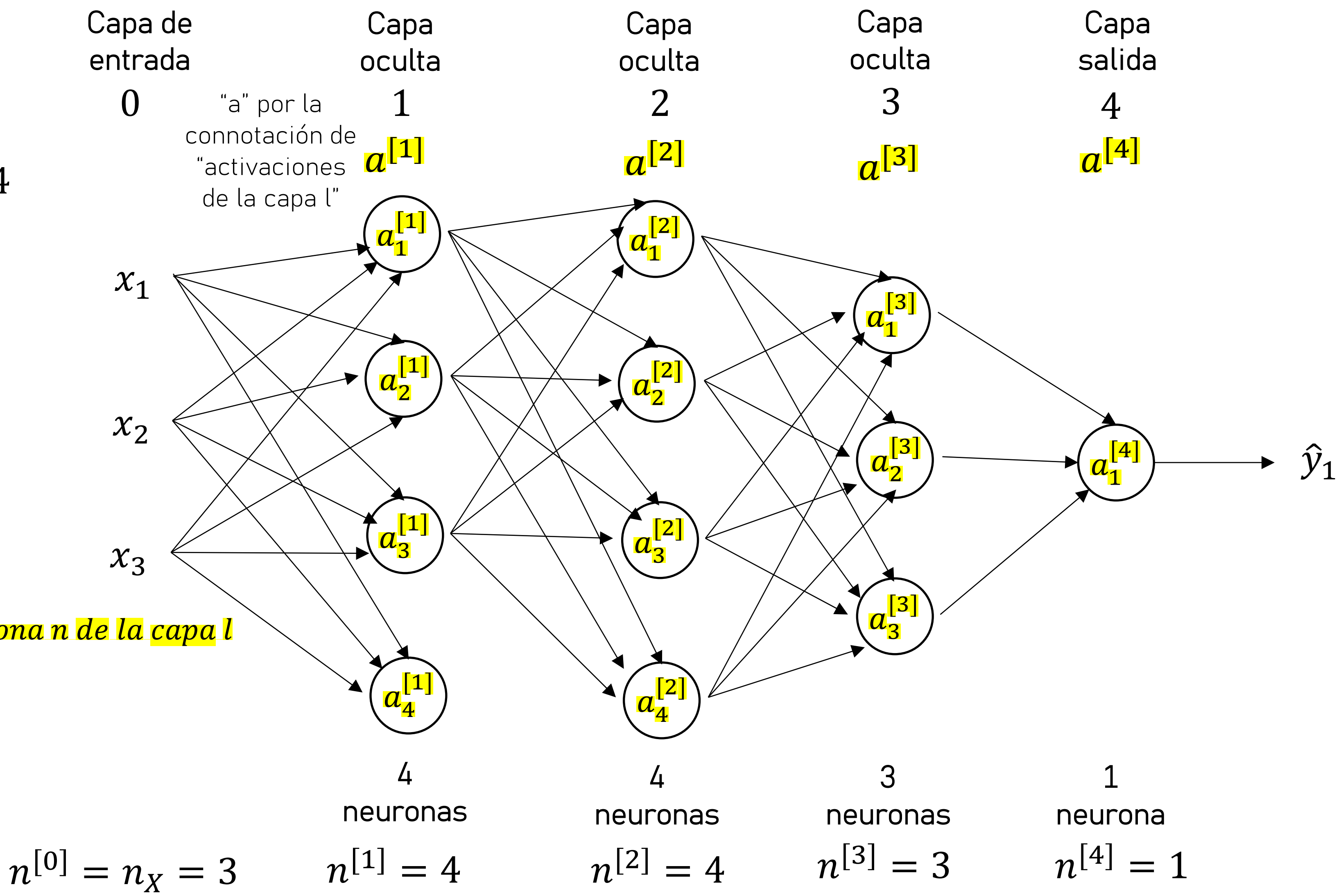
$L = número\ de\ capas = 4$

$\sum_{i=1}^{N} w_i x_i + b$

$a^{[l]} = f^{[l]}(z^{[l]})$



$a_n^{[l]} = la\ activación\ de\ la\ la\ neurona\ n\ de\ la\ capa\ l$



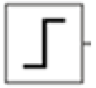
Notación

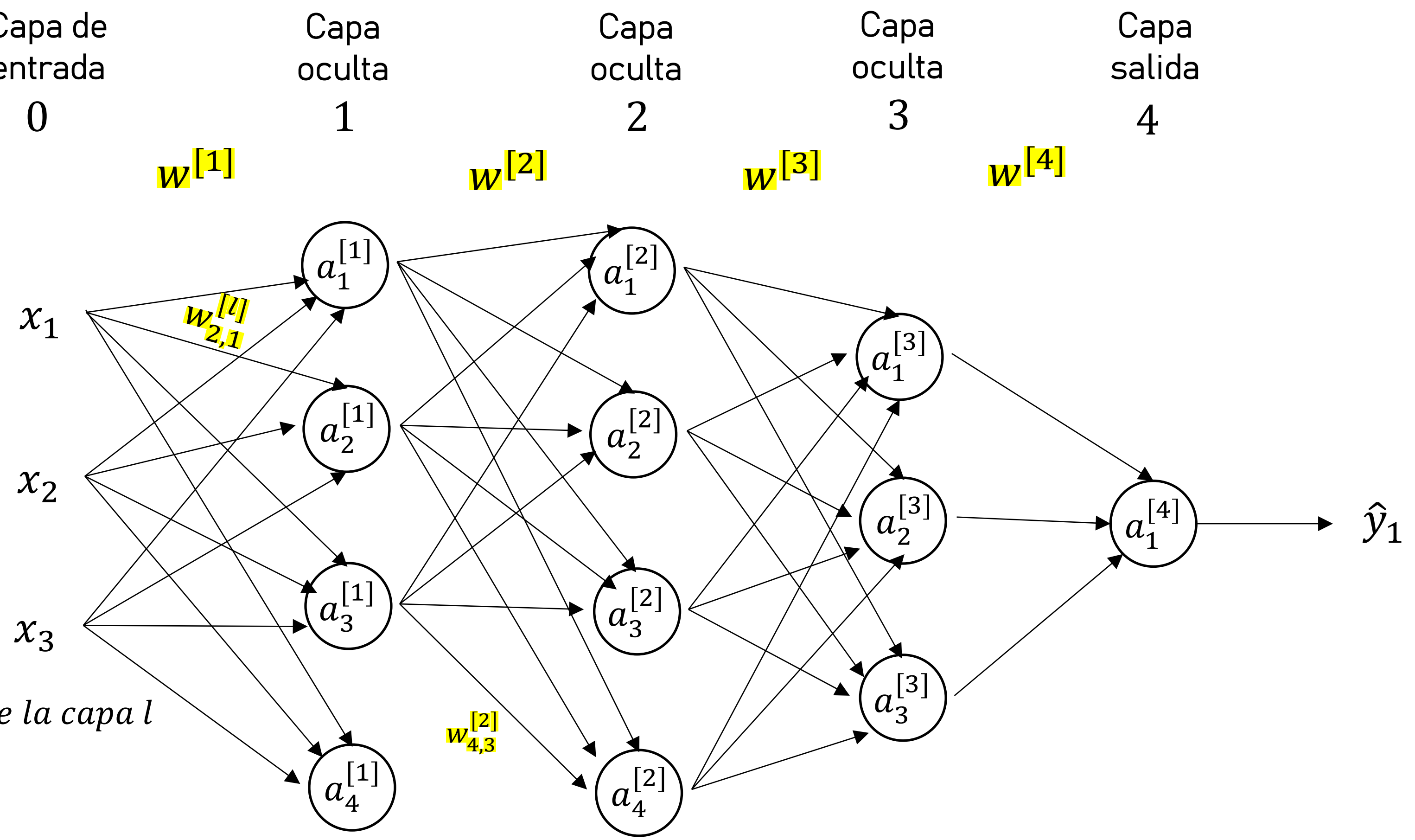
RN de 4 capas

$L = número\ de\ capas = 4$

$\sum_{i=1}^{N} w_i x_i + b$

$a^{[l]} = g^{[l]}(z^{[l]})$





$a_n^{[l]} = la\ activación\ de\ la\ la\ neurona\ n\ de\ la\ capa\ l$

$w^{[l]} = pesos\ de\ z^{[l]}$

$w_{i,j}^{[l]} = el\ peso\ a\ la\ neurona\ destino\ i\ (de\ la\ capa\ l) de\ la\ neurona\ de\ origen\ j$

$n^{[0]} = n_X = 3$ $\quad n^{[1]} = 4$ $\quad n^{[2]} = 4$ $\quad n^{[3]} = 3$ $\quad n^{[4]} = 1$

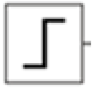
Notación

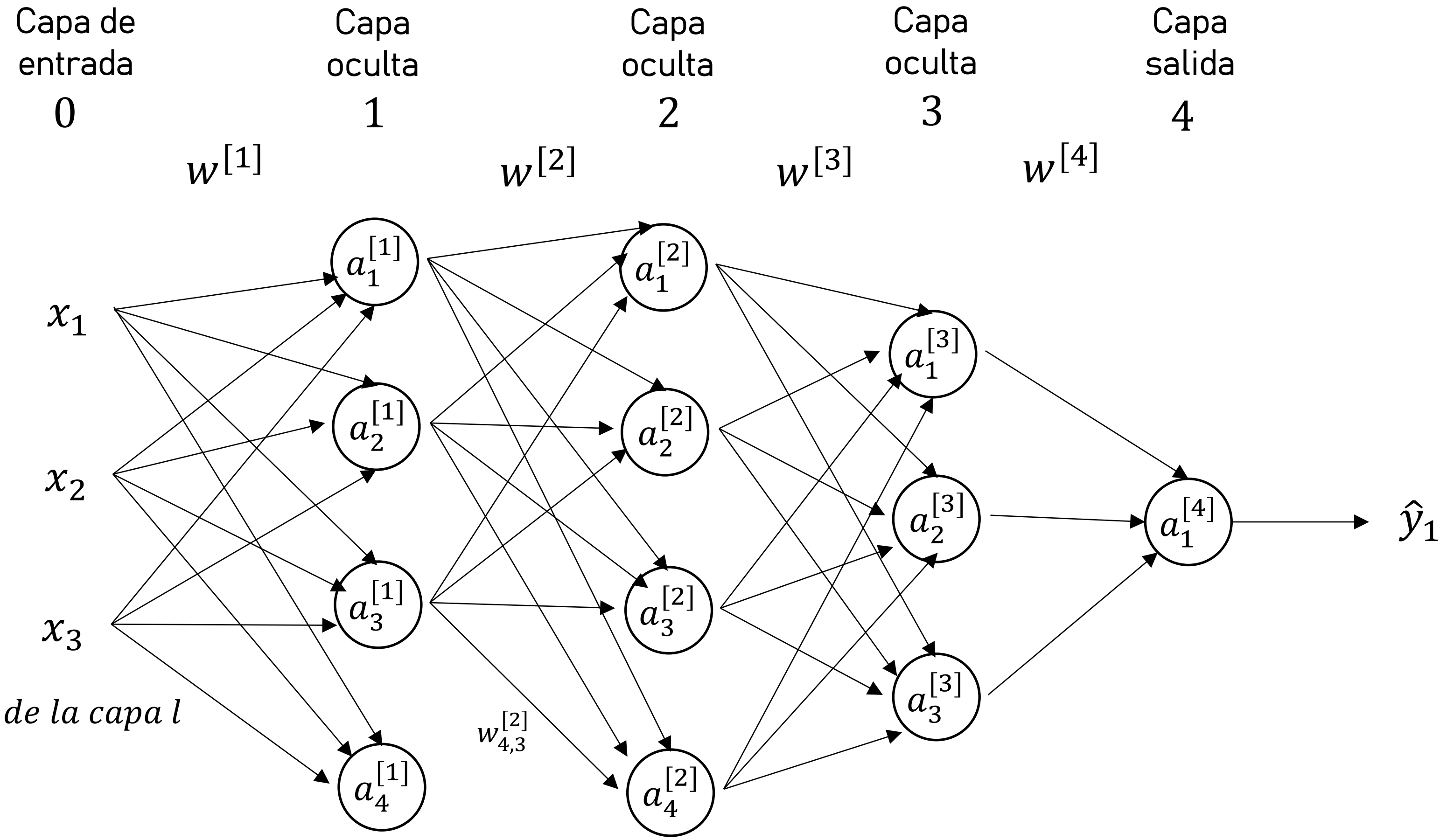
RN de 4 capas

$L = número\ de\ capas = 4$

$\sum_{i=1}^{N} w_i x_i + b$

$a^{[l]} = g^{[l]}(z^{[l]})$





$a_n^{[l]} = la\ activación\ de\ la\ la\ neurona\ n\ de\ la\ capa\ l$

$w^{[l]} = pesos\ de\ z^{[l]}$

$w_{i,j}^{[l]} = el\ peso\ a\ la\ neurona\ destino\ i\ (de\ la\ capa\ l) de\ la\ neurona\ de\ origen\ j$

$b^{[l]} = sesgo\ de\ z^{[l]}$

$n^{[0]} = n_X = 3$ $n^{[1]} = 4$ $n^{[2]} = 4$ $n^{[3]} = 3$ $n^{[4]} = 1$

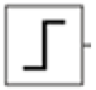
Notación

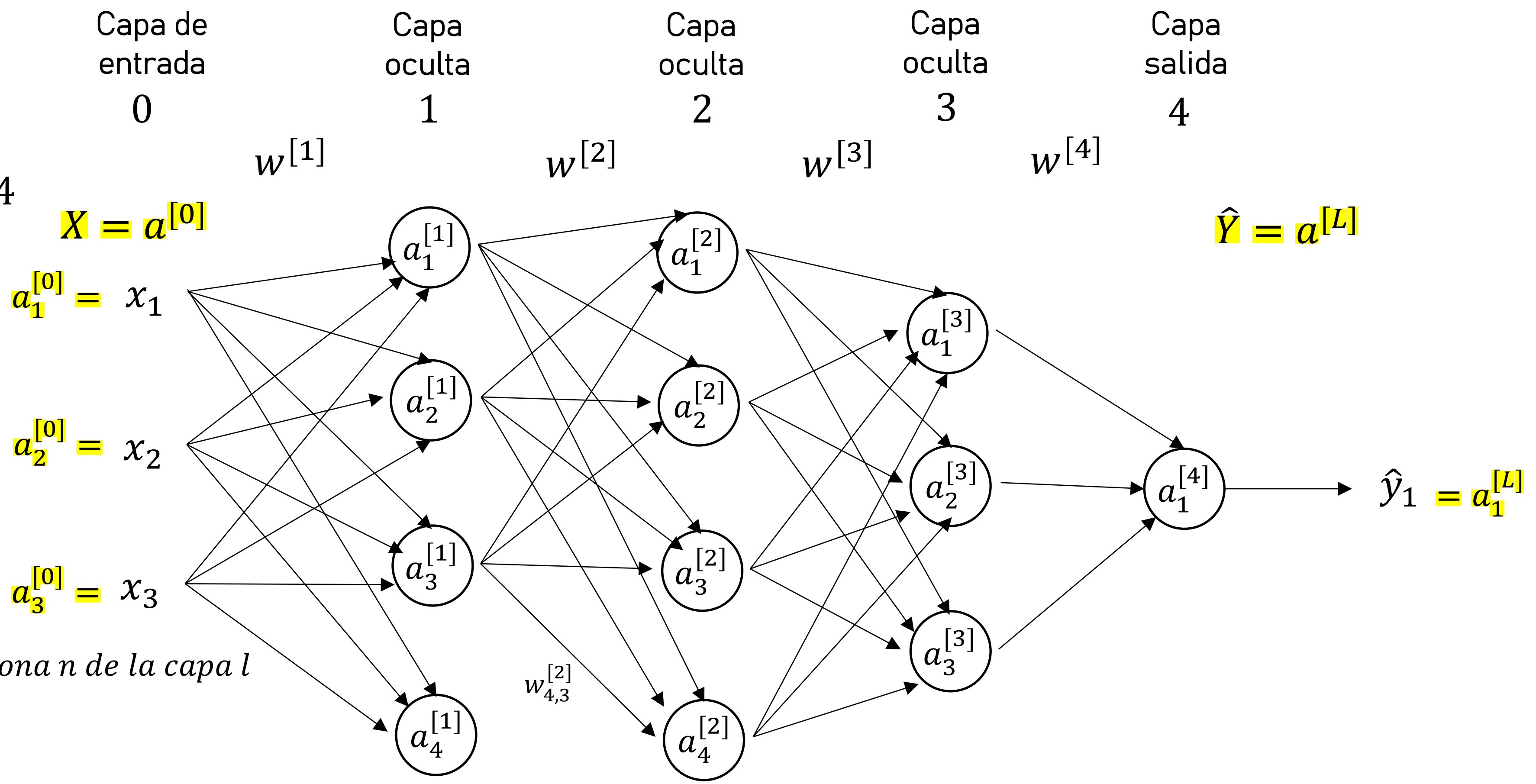
RN de 4 capas

$L = número\ de\ capas = 4$

$$\sum_{i=1}^{N} w_i x_i + b$$

$a^{[l]} = g^{[l]}(z^{[l]})$





$a_n^{[l]} = la\ activación\ de\ la\ la\ neurona\ n\ de\ la\ capa\ l$

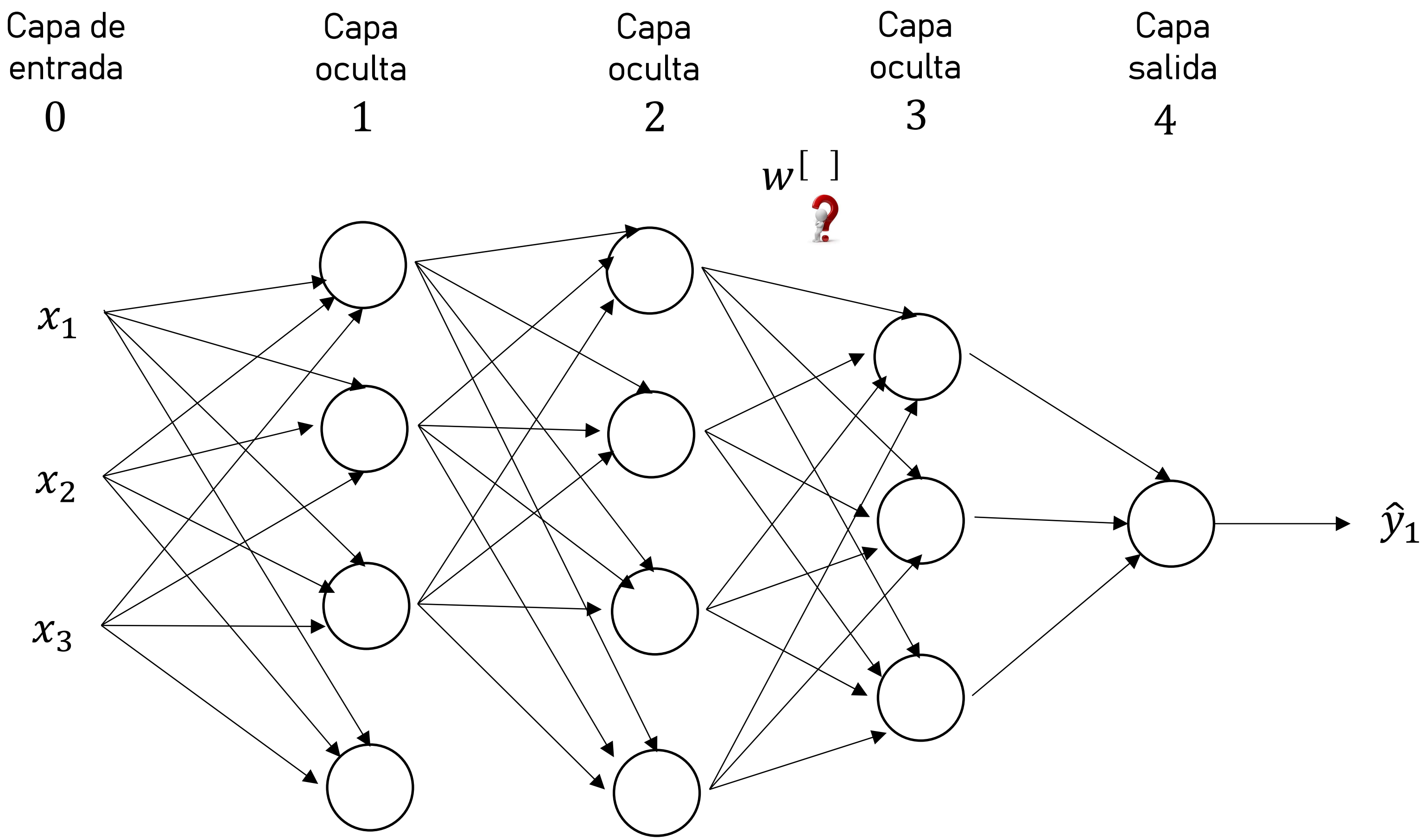
$w^{[l]} = pesos\ de\ z^{[l]}$

$w_{i,j}^{[l]} = el\ peso\ a\ la\ neurona\ destino\ i\ (de\ la\ capa\ l) de\ la\ neurona\ de\ origen\ j$

$b^{[l]} = sesgo\ de\ z^{[l]}$

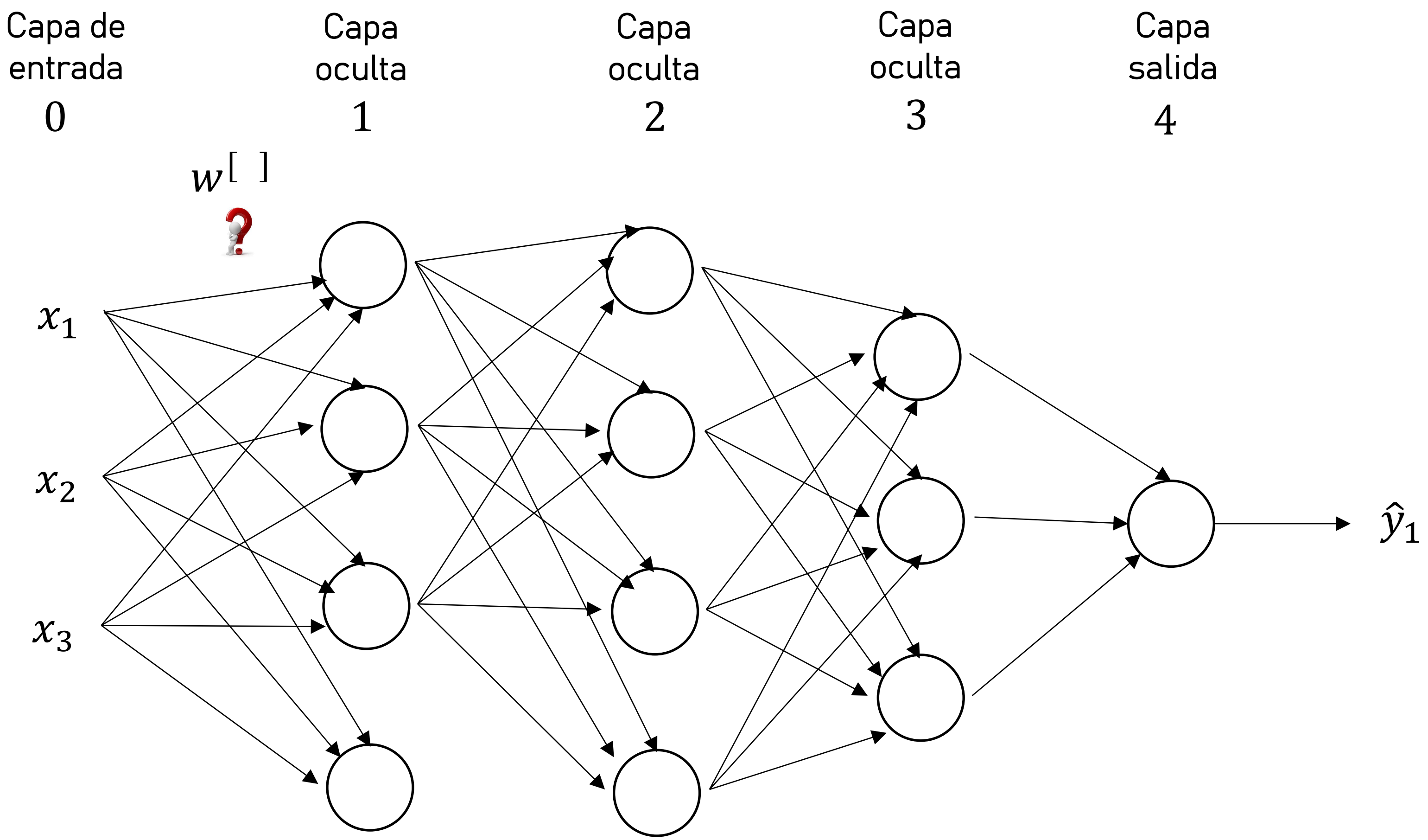
$n^{[0]} = n_X = 3$ $\quad n^{[1]} = 4$ $\quad n^{[2]} = 4$ $\quad n^{[3]} = 3$ $\quad n^{[4]} = 1$

Notación



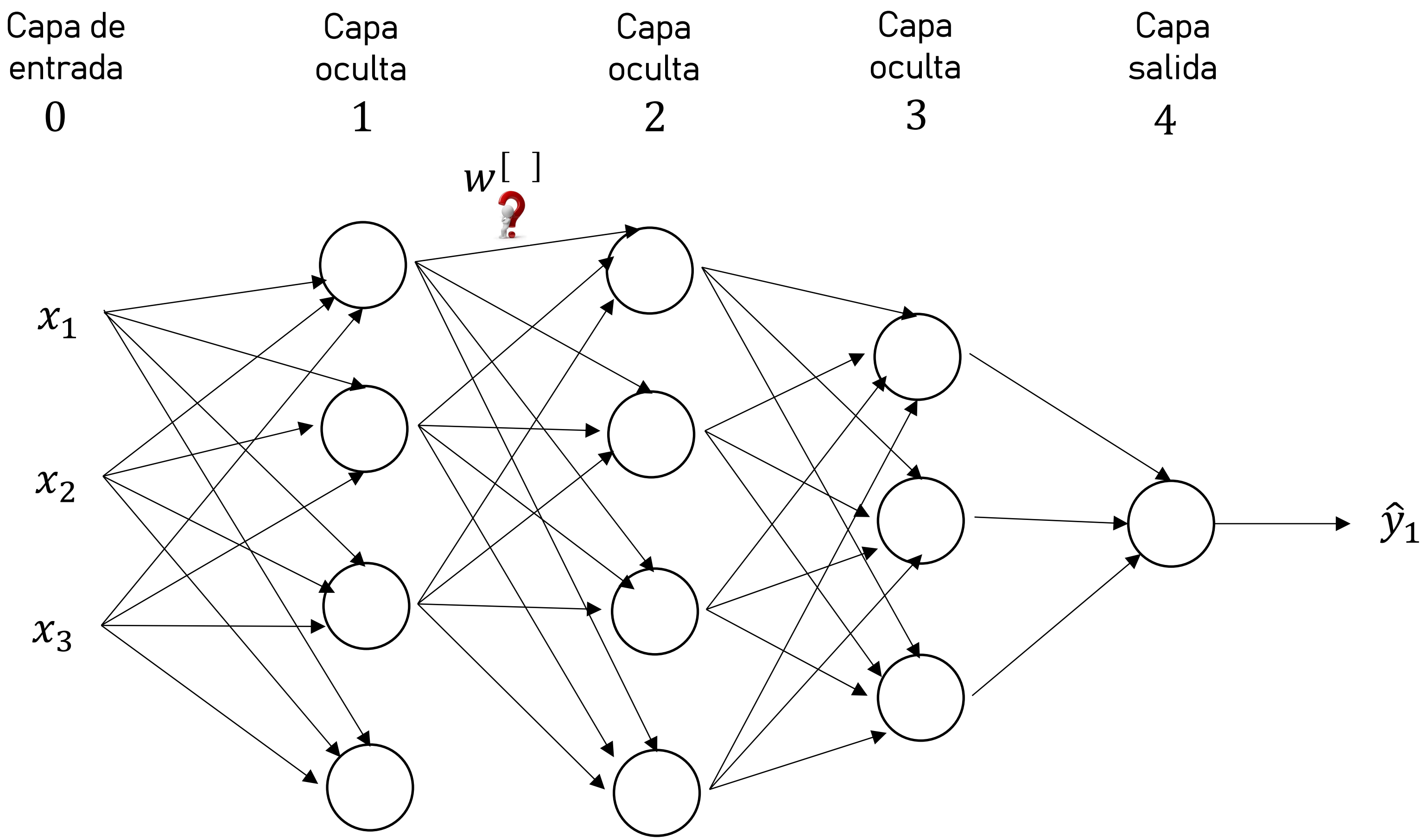
Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
$w^{[\ ]}$
x_1
x_2
x_3
$\hat{y}_1$

Notación

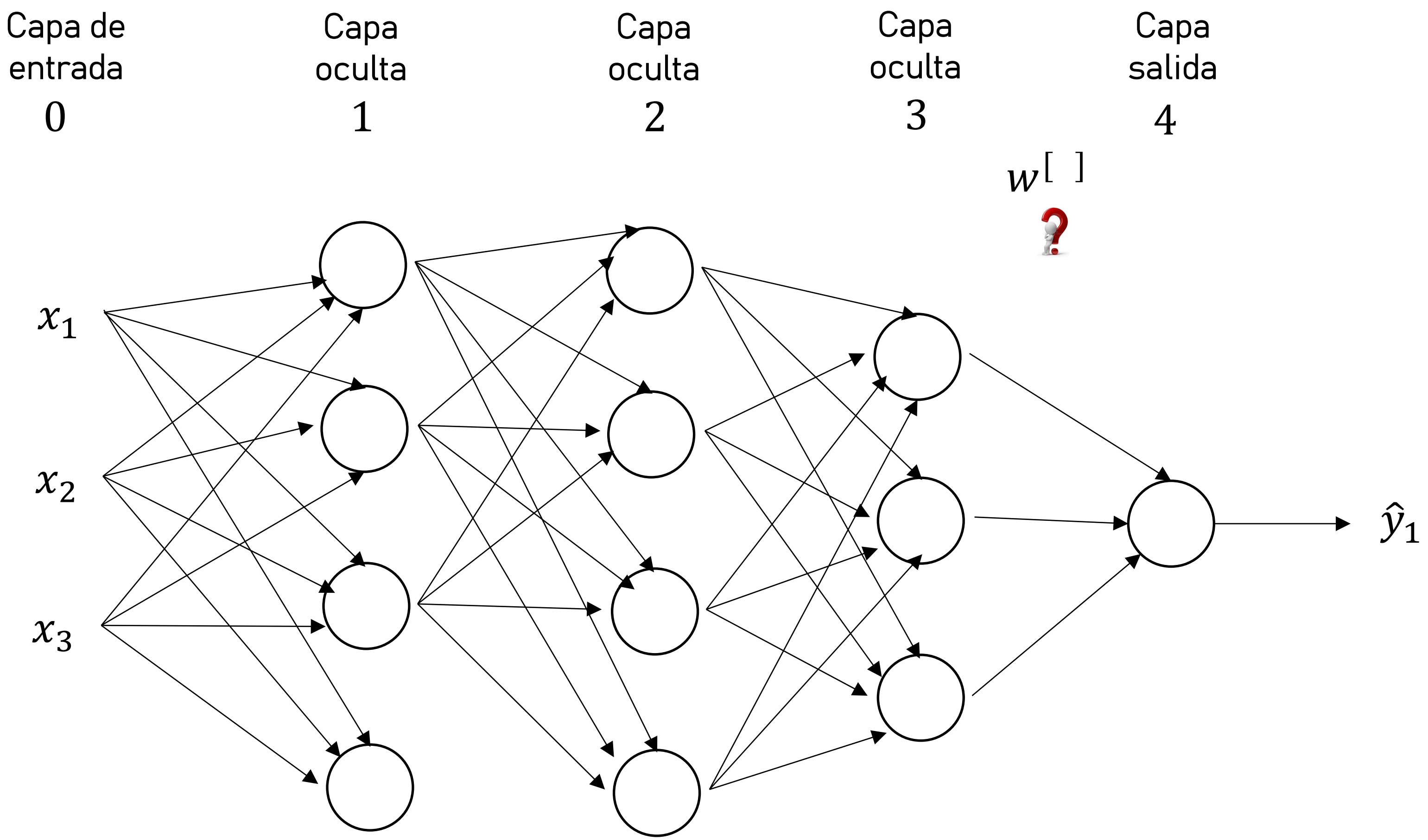


Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
$w^{[\ ]}$
x_1
x_2
x_3
$\hat{y}_1$

Notación

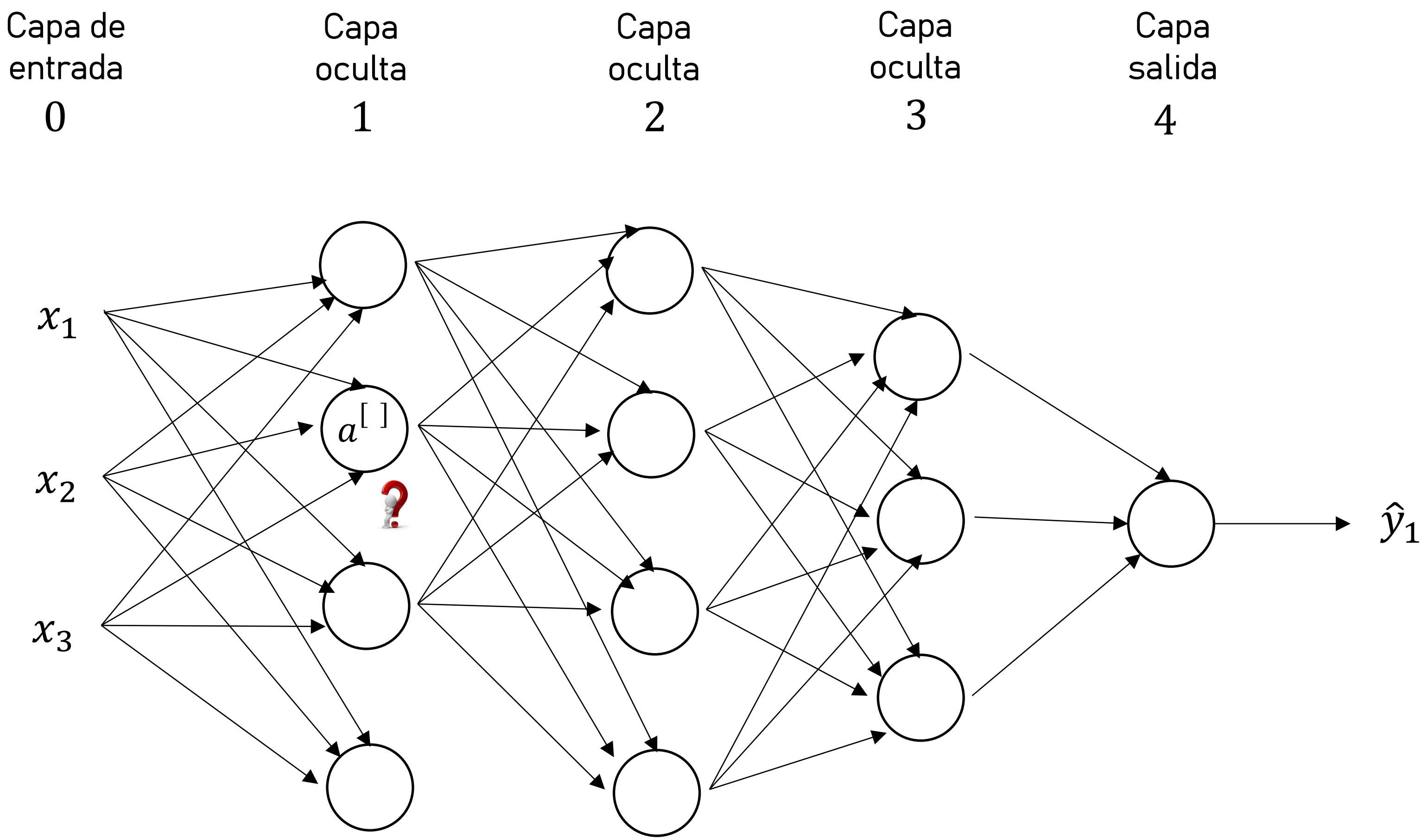

Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
$w^{[\ ]}$
x_1
x_2
x_3
$\hat{y}_1$

Notación

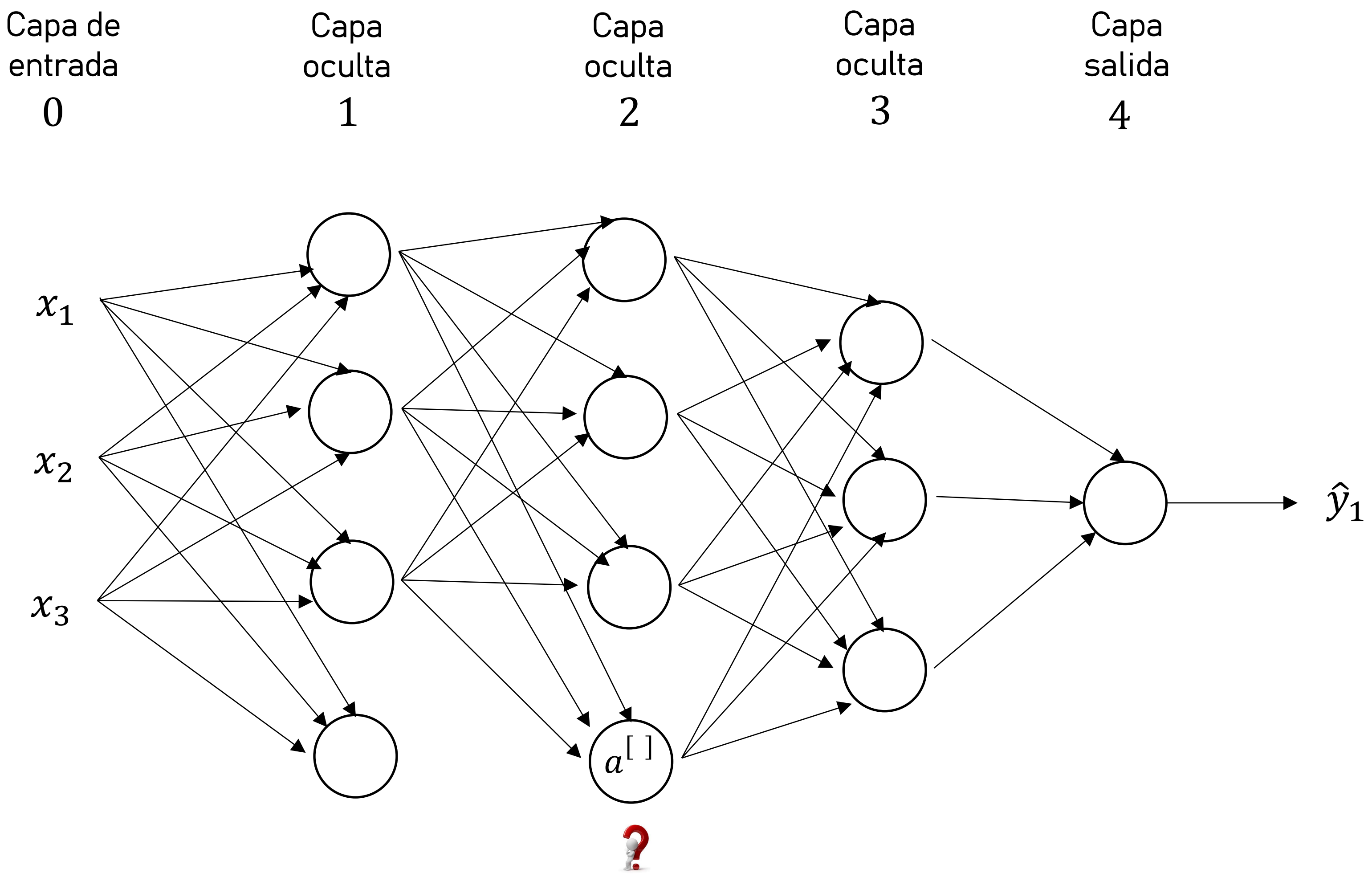


Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
$w^{[\ ]}$
x_1
x_2
x_3
$\hat{y}_1$

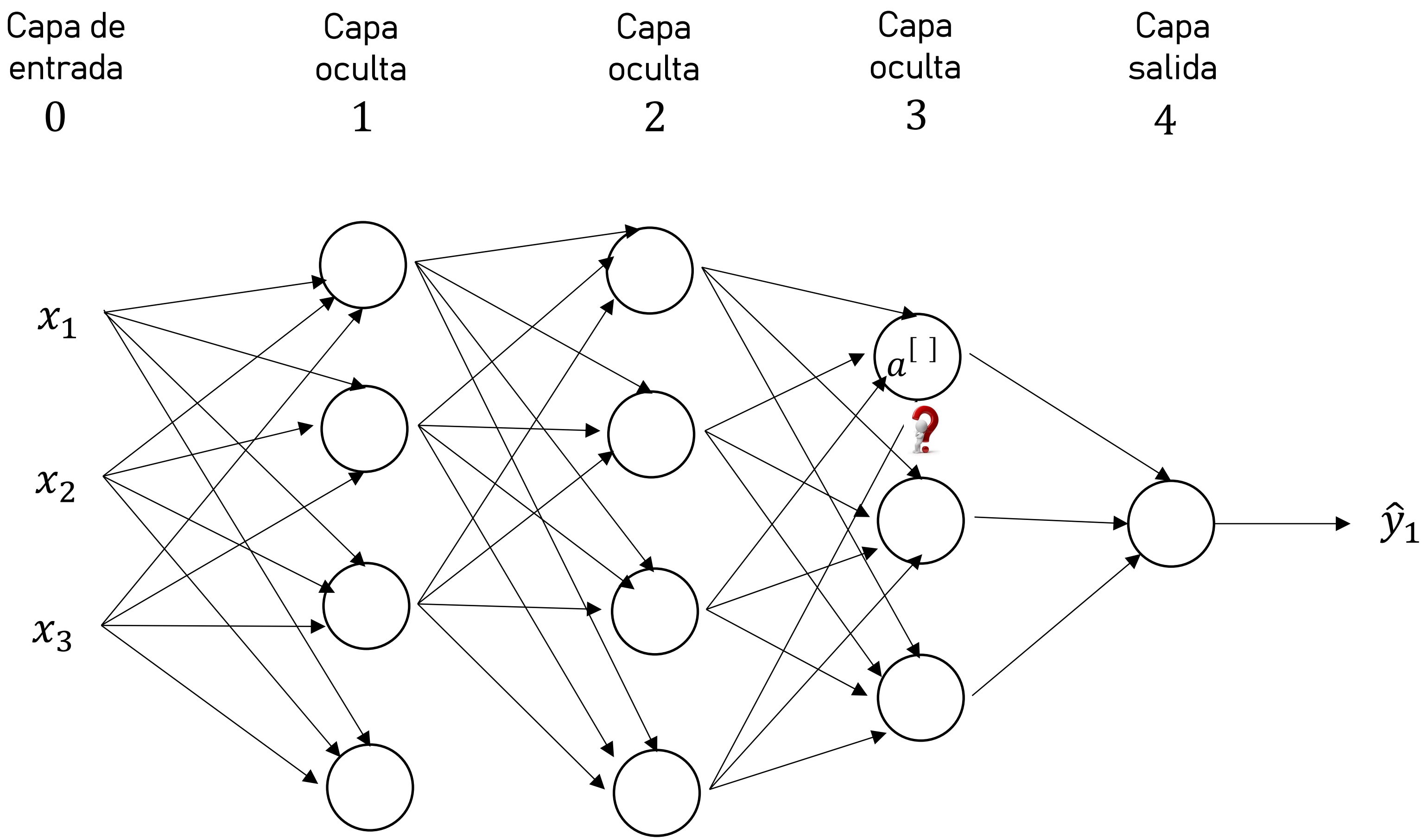
Notación


Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
x_1
x_2
x_3
$a^{[\ ]}$
$\hat{y}_1$

Notación

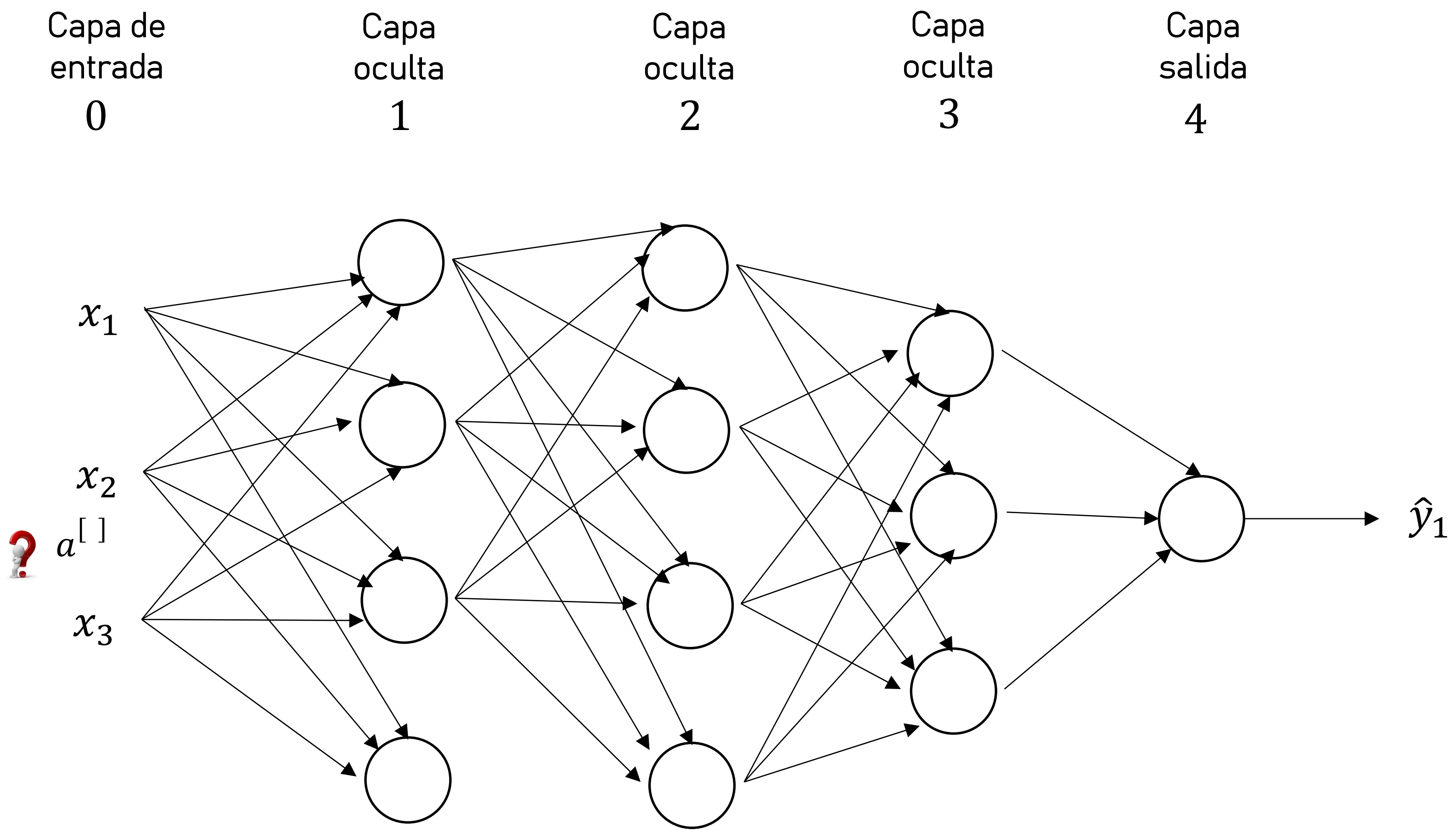

Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
x_1
x_2
x_3
$a^{[\ ]}$
$\hat{y}_1$

Notación



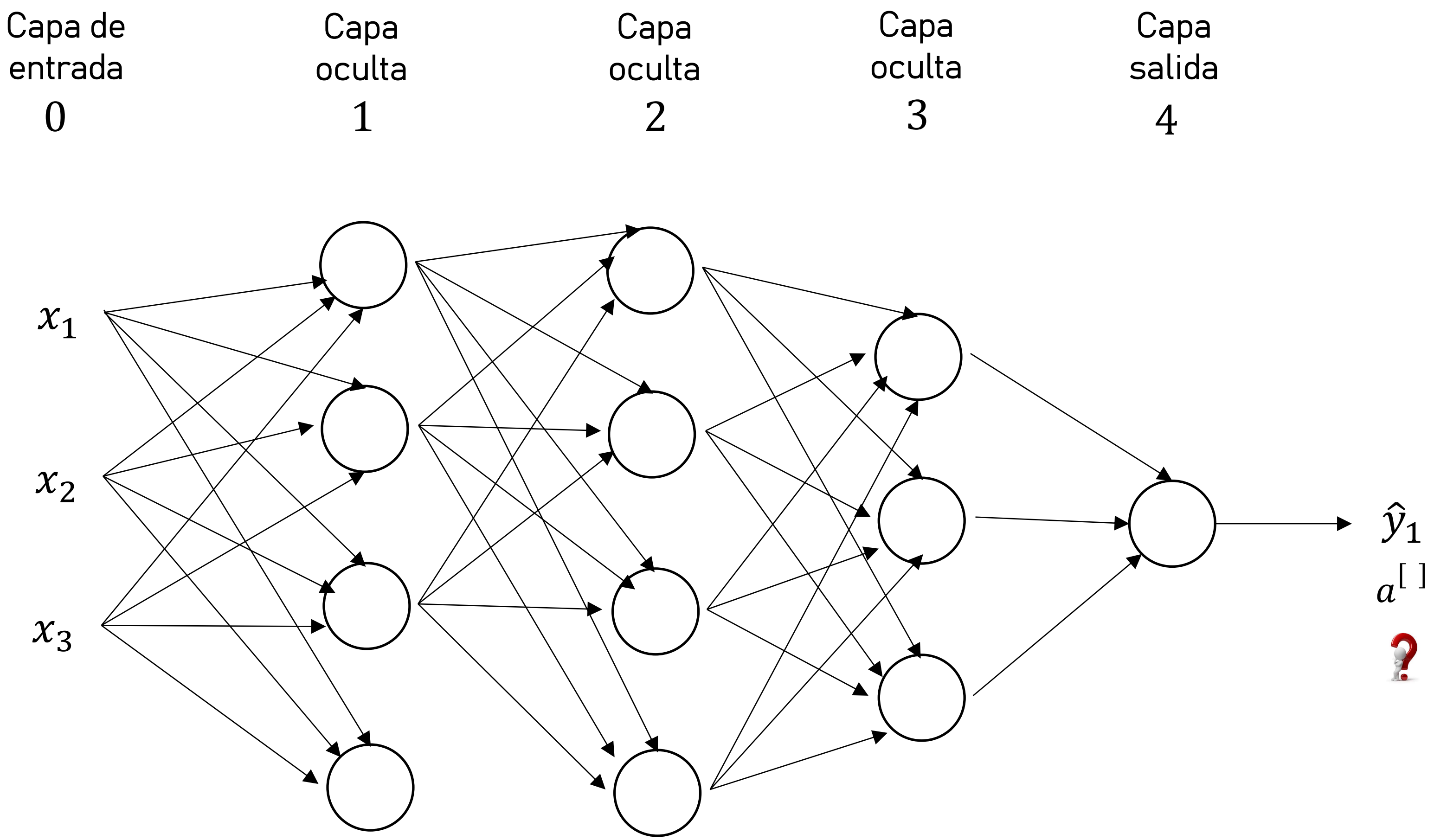
Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
x_1
x_2
x_3
$a^{[\,]}$
?
$\hat{y}_1$

Notación



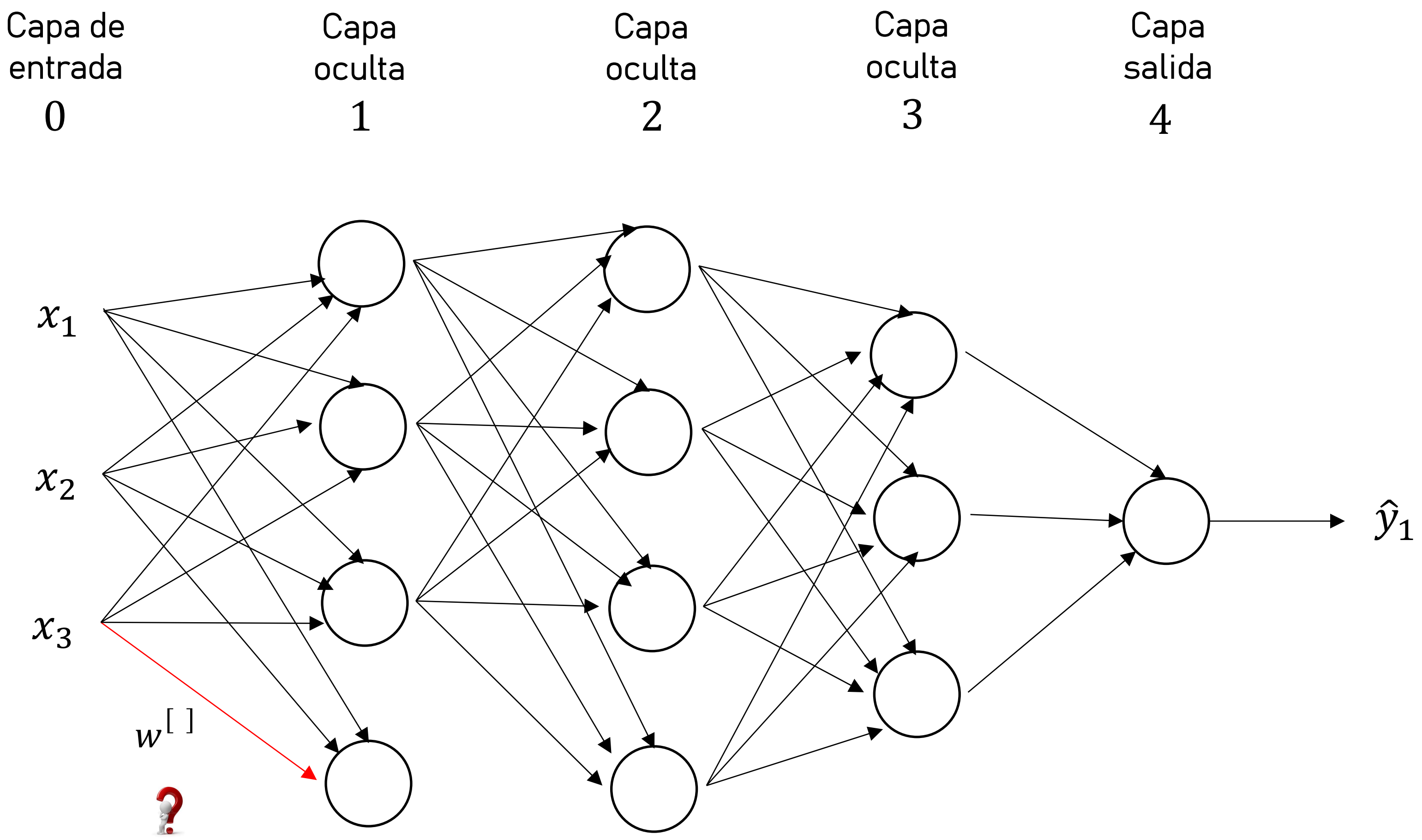
Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
x_1
x_2
$a^{[\,]}$
x_3
$\hat{y}_1$

Notación



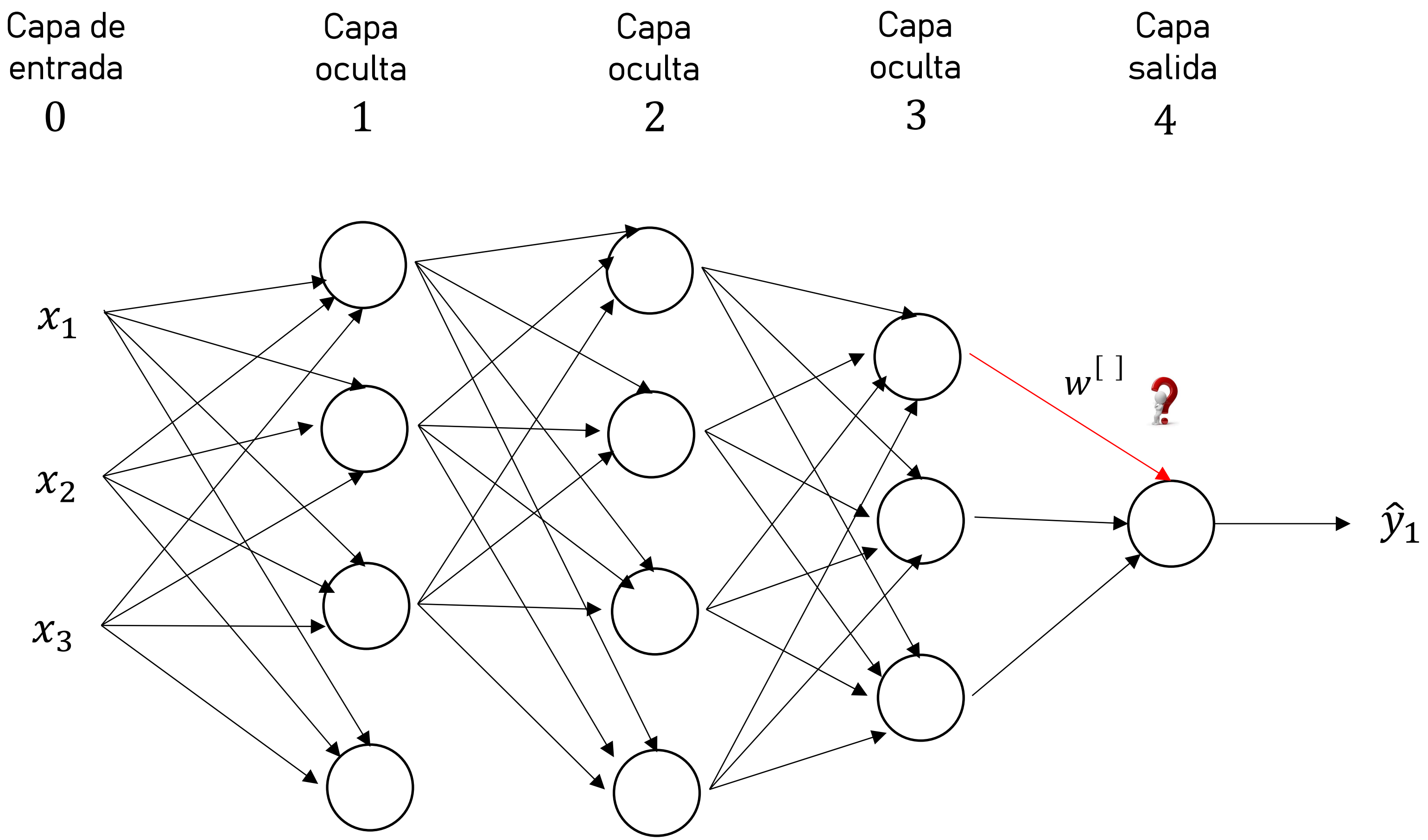
Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
x_1
x_2
x_3
$\hat{y}_1$
$a^{[\ ]}$

Notación



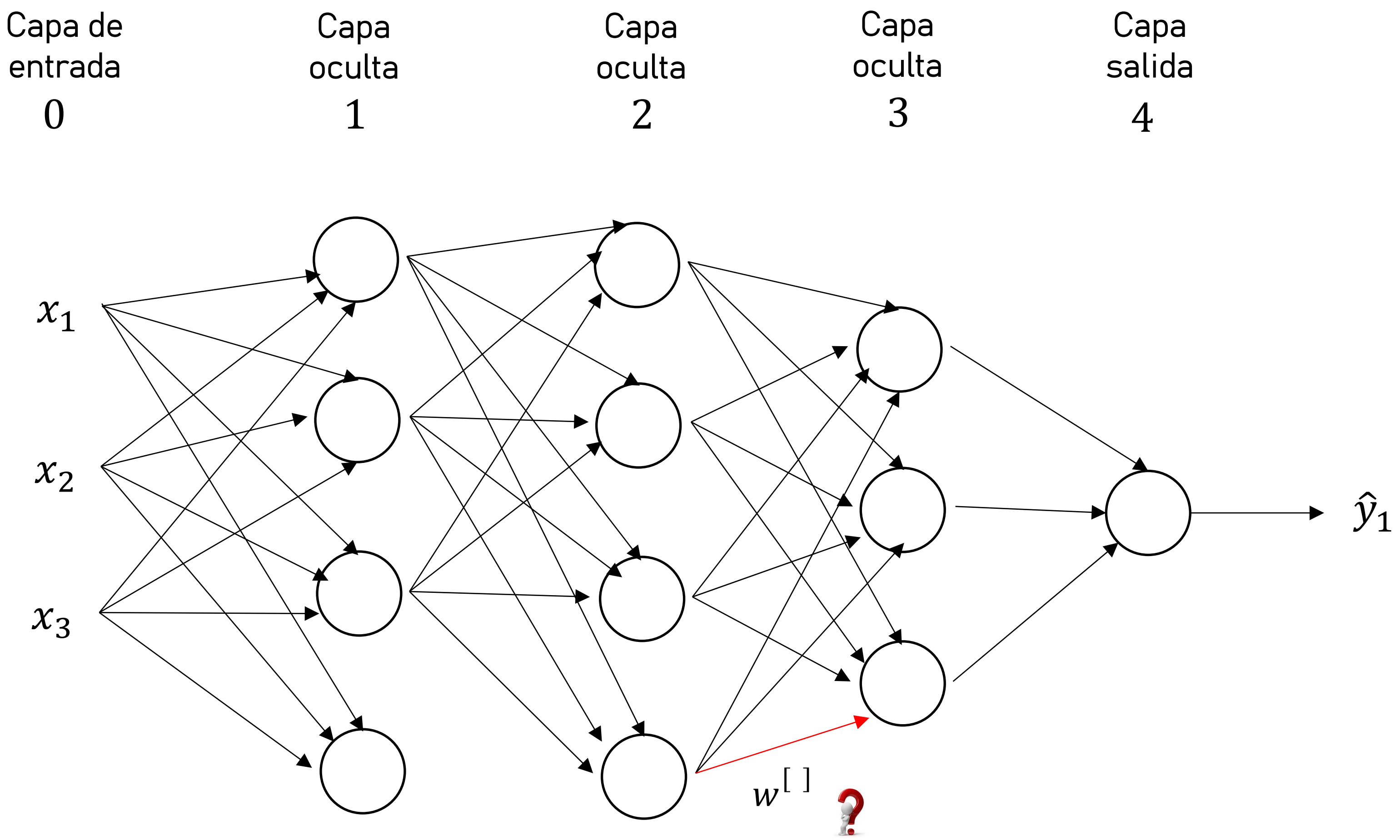
Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
x_1
x_2
x_3
$w^{[\ ]}$
$\hat{y}_1$

Notación



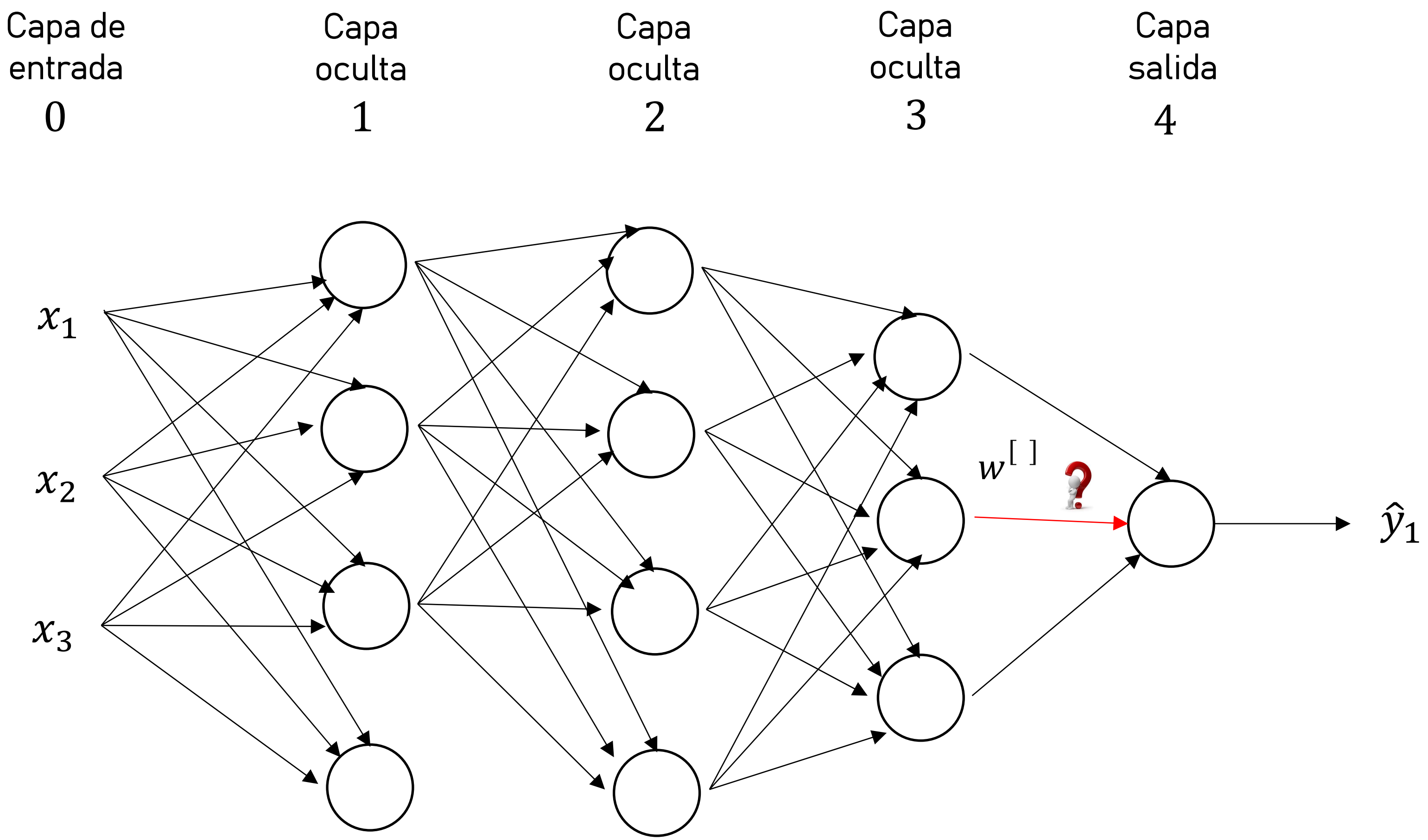
Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
x_1
x_2
x_3
$w^{[\ ]}$
$\hat{y}_1$

Notación



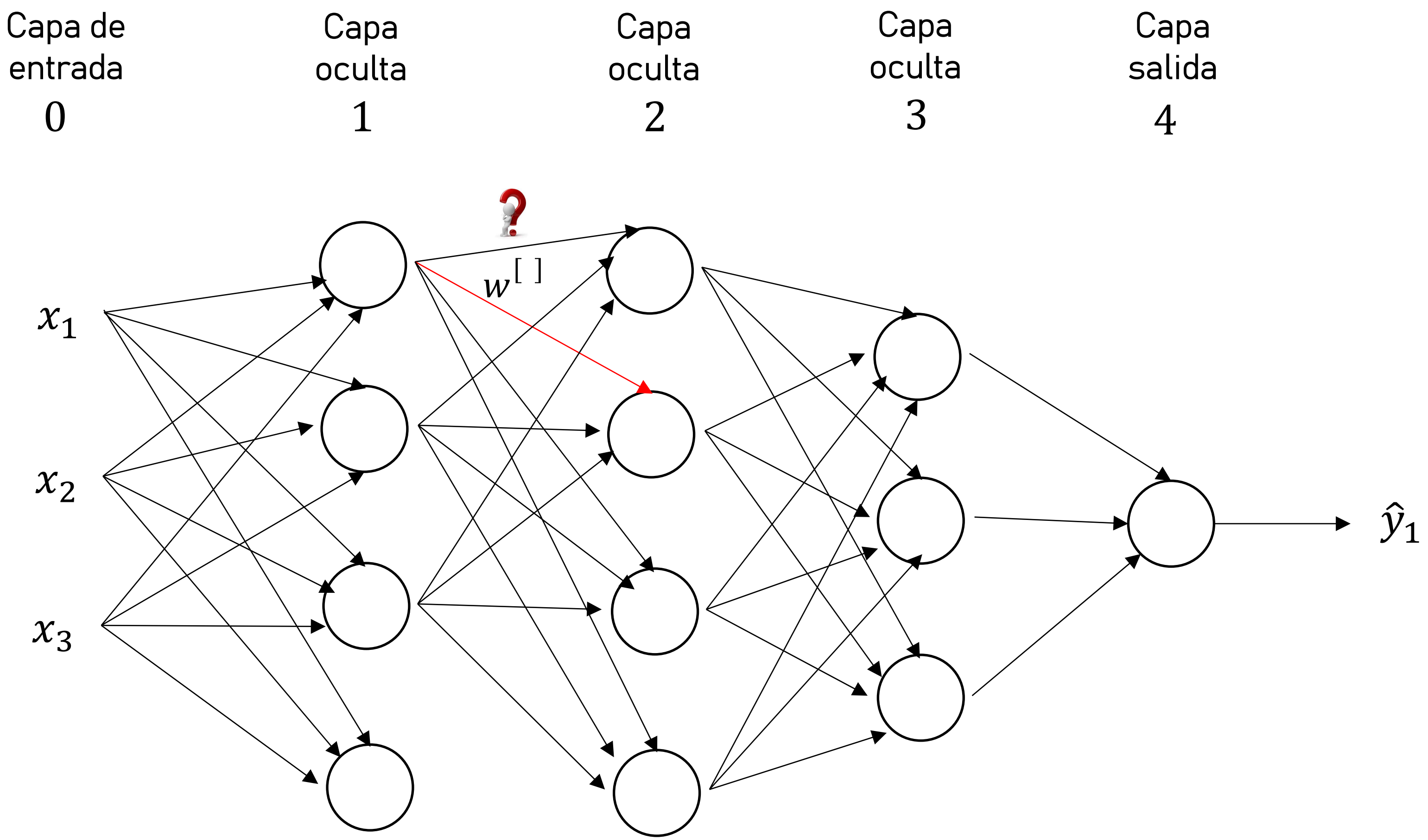
Capa de entrada 0
Capa oculta 1
Capa oculta 2
Capa oculta 3
Capa salida 4
x_1
x_2
x_3
$\hat{y}_1$
$w^{[\ ]}$

Notación



Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
x_1
x_2
x_3
$w^{[\ ]}$
$\hat{y}_1$

Notación



Capa de entrada
0
Capa oculta
1
Capa oculta
2
Capa oculta
3
Capa salida
4
$w^{[\ ]}$
x_1
x_2
x_3
$\hat{y}_1$

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación
Hacia adelante

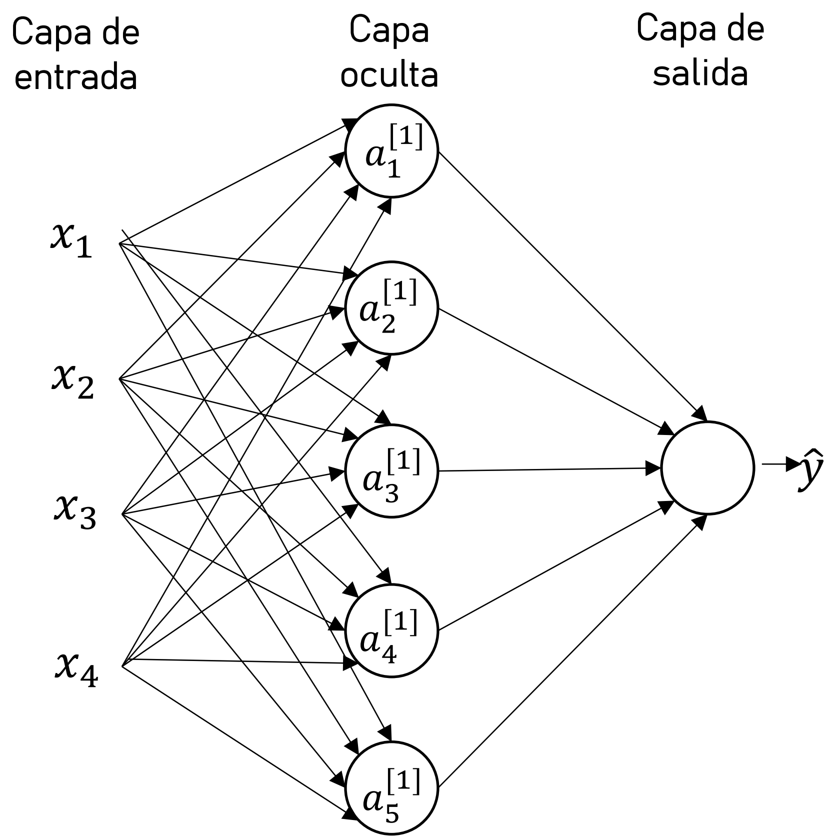
Alternativa:
- Métodos evolutivos (neuroevolución)
    - Últil, p.ej., cuando la función de pérdida no es diferenciable.

Propagación
Hacia atrás

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación
Hacia adelante



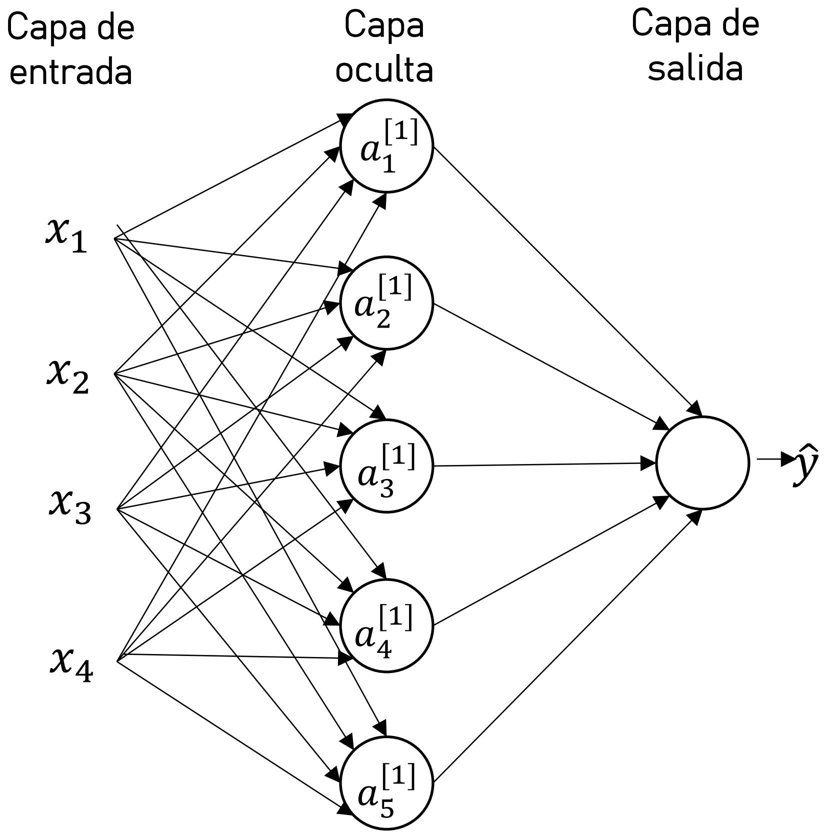
Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación Hacia adelante

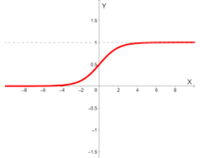


1. Los datos de entrada alimentan la red, capa por capa.

2. En cada capa, se calcula la suma pesada de las entradas más el sesgo.

$$z_1^{[1]} = \sum_{i=1}^{4} w_{1,i}^{[1]}\, x_i + b^{[1]}$$

3. Se aplica la función de activación para producir la salida de la capa.



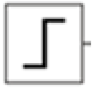
$$\sigma(x) = \frac{1}{1+e^{-x}}$$

4. Este proceso continúa hasta que la capa de salida produce la predicción de la red.

Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

En el perceptrón original:



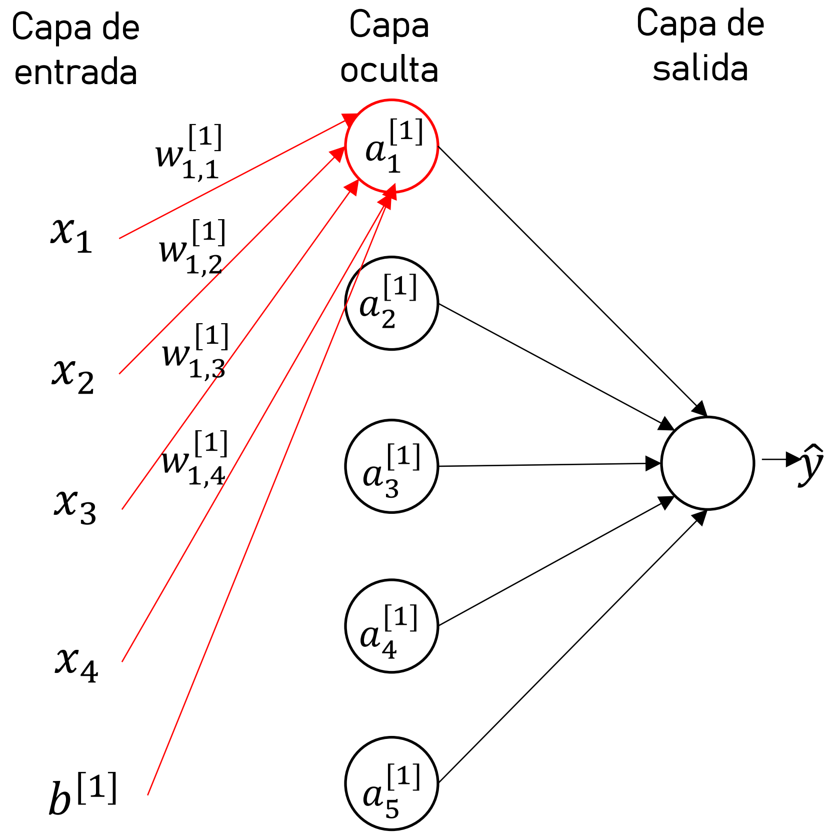
$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

… pero, no es diferenciable.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación Hacia adelante



1. Los datos de entrada alimentan la red, capa por capa.

2. En cada capa, se calcula la suma pesada de las entradas más el sesgo.

$$\boxed{z_1^{[1]} = \sum_{i=1}^{4} w_{1,i}^{[1]}\, x_i + b^{[1]}}$$

3. Se aplica la función de activación para producir la salida de la capa.

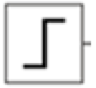
$$a_1^{[1]} = \boldsymbol{\sigma}\left(z_1^{[1]}\right) = \frac{1}{1 + e^{-z_1^{[1]}}}$$

4. Este proceso continúa hasta que la capa de salida produce la predicción de la red.

Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

En el perceptrón original:



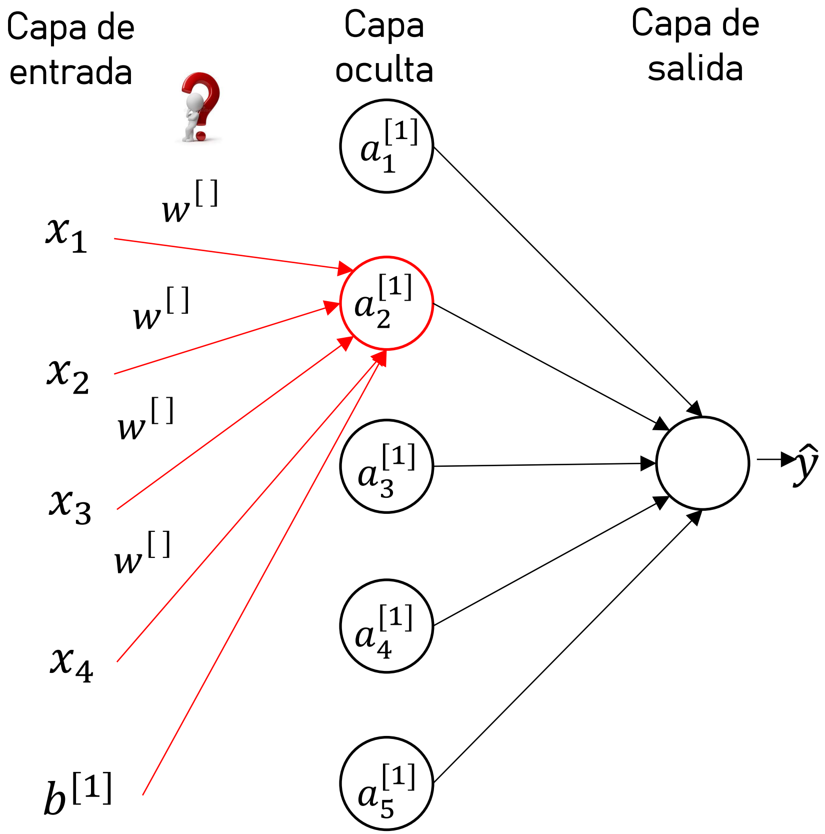
$$f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$$

… pero, no es diferenciable.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación Hacia adelante



1. Los datos de entrada alimentan la red, capa por capa.

2. En cada capa, se calcula la suma pesada de las entradas más el sesgo.

$$\boxed{z^{[\,]} = \sum_{i=1}^{4} w \quad x_i + b^{[1]}}$$

3. Se aplica la función de activación para producir la salida de la capa.

$$\boxed{a_1^{[\,]} = \boldsymbol{\sigma}\left(z^{[\,]}\right) = \frac{1}{1 + e^{-z^{[\,]}}}}$$

4. Este proceso continúa hasta que la capa de salida produce la predicción de la red.

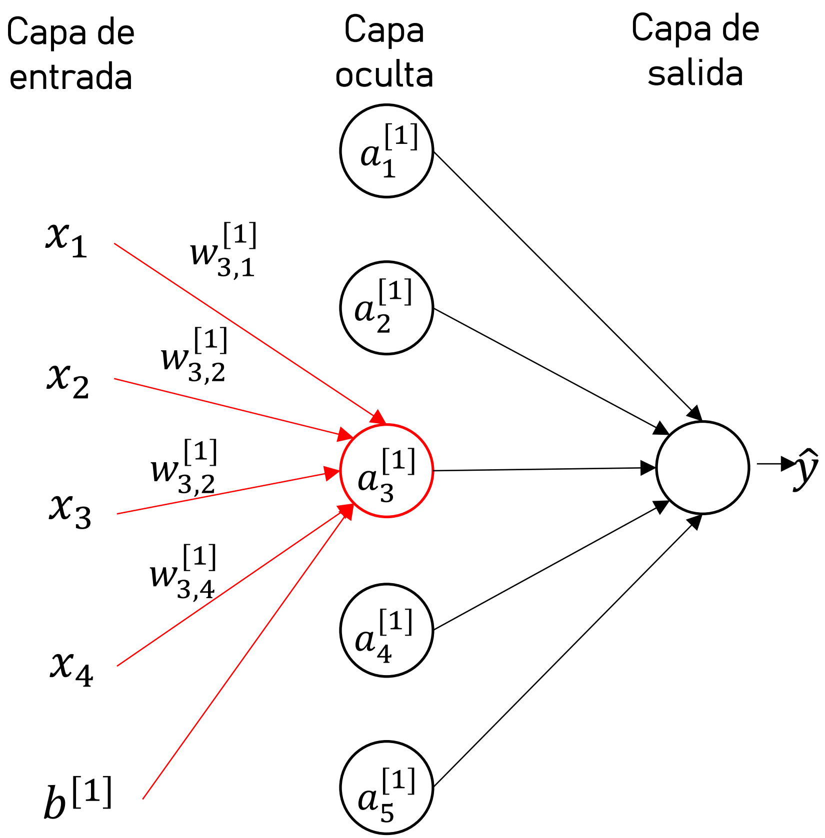
Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación Hacia adelante



1. Los datos de entrada alimentan la red, capa por capa.

2. En cada capa, se calcula la suma pesada de las entradas más el sesgo.

$$\boxed{z_3^{[1]} = \sum_{i=1}^{4} w_{3,i}^{1}\, x_i + b^{[1]}}$$

3. Se aplica la función de activación para producir la salida de la capa.

$$\boxed{a_3^{[1]} = \boldsymbol{\sigma}\left(z_3^{[1]}\right) = \frac{1}{1 + e^{-z_3^{[1]}}}}$$

4. Este proceso continúa hasta que la capa de salida produce la predicción de la red.

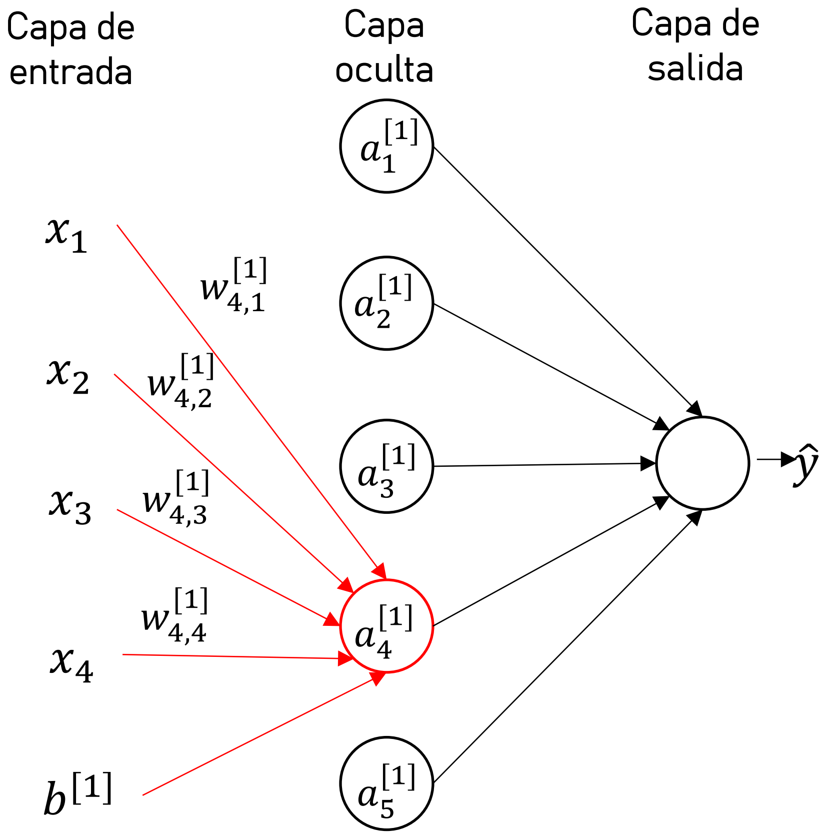
Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación Hacia adelante



1. Los datos de entrada alimentan la red, capa por capa.

2. En cada capa, se calcula la suma pesada de las entradas más el sesgo.

$$\boxed{z_4^{[1]} = \sum_{i=1}^{4} w_{4,i}^{1}\, x_i + b^{[1]}}$$

3. Se aplica la función de activación para producir la salida de la capa.

$$\boxed{a_4^{[1]} = \boldsymbol{\sigma}\left(z_4^{[1]}\right) = \frac{1}{1 + e^{-z_4^{[1]}}}}$$

4. Este proceso continúa hasta que la capa de salida produce la predicción de la red.

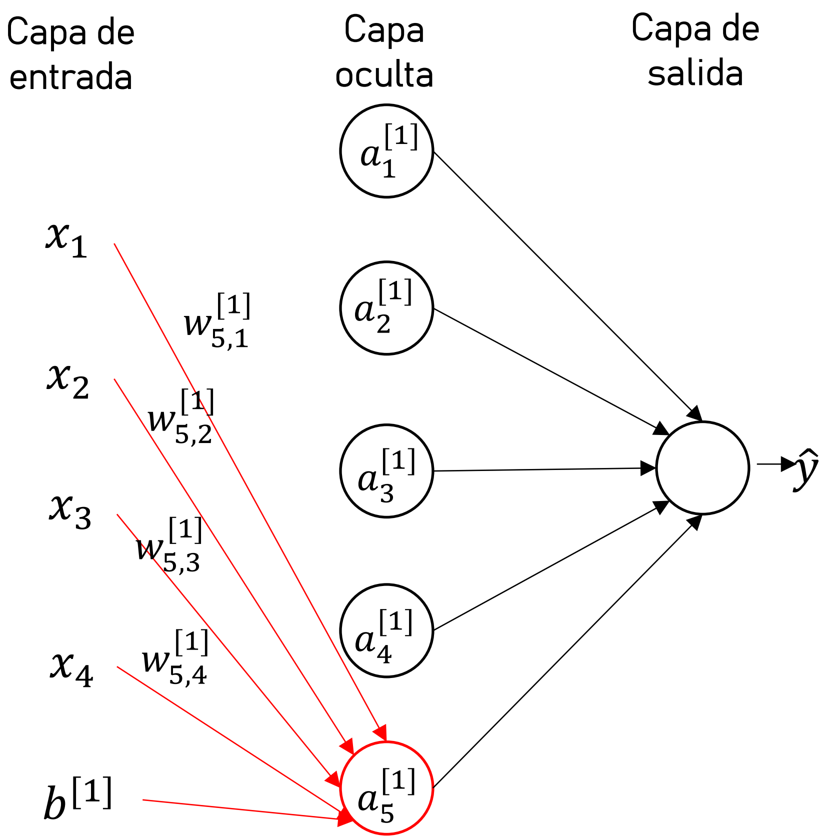
Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación Hacia adelante



1. Los datos de entrada alimentan la red, capa por capa.

2. En cada capa, se calcula la suma pesada de las entradas más el sesgo.

$$\boxed{z_5^{[1]} = \sum_{i=1}^{4} w_{5,i}^{1}\, x_i + b^{[1]}}$$

3. Se aplica la función de activación para producir la salida de la capa.

$$\boxed{a_5^{[1]} = \boldsymbol{\sigma}\left(z_5^{[1]}\right) = \frac{1}{1 + e^{-z_5^{[1]}}}}$$

4. Este proceso continúa hasta que la capa de salida produce la predicción de la red.

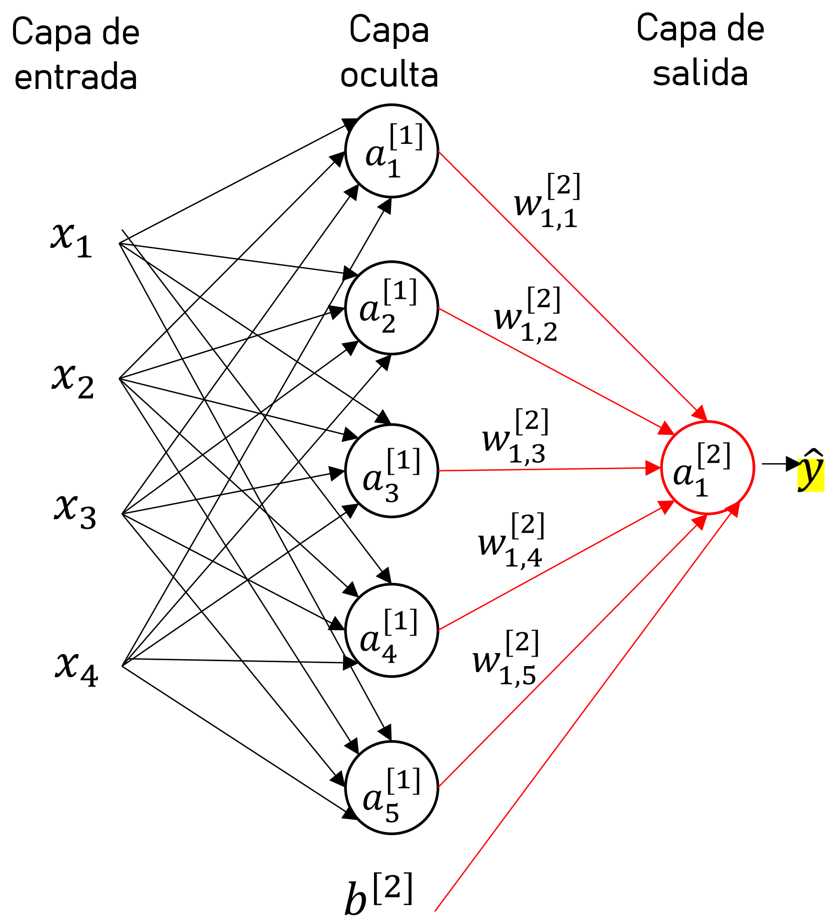
Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación Hacia adelante



1. Los datos de entrada alimentan la red, capa por capa.

2. En cada capa, se calcula la suma pesada de las entradas más el sesgo.

3. Se aplica la función de activación para producir la salida de la capa.

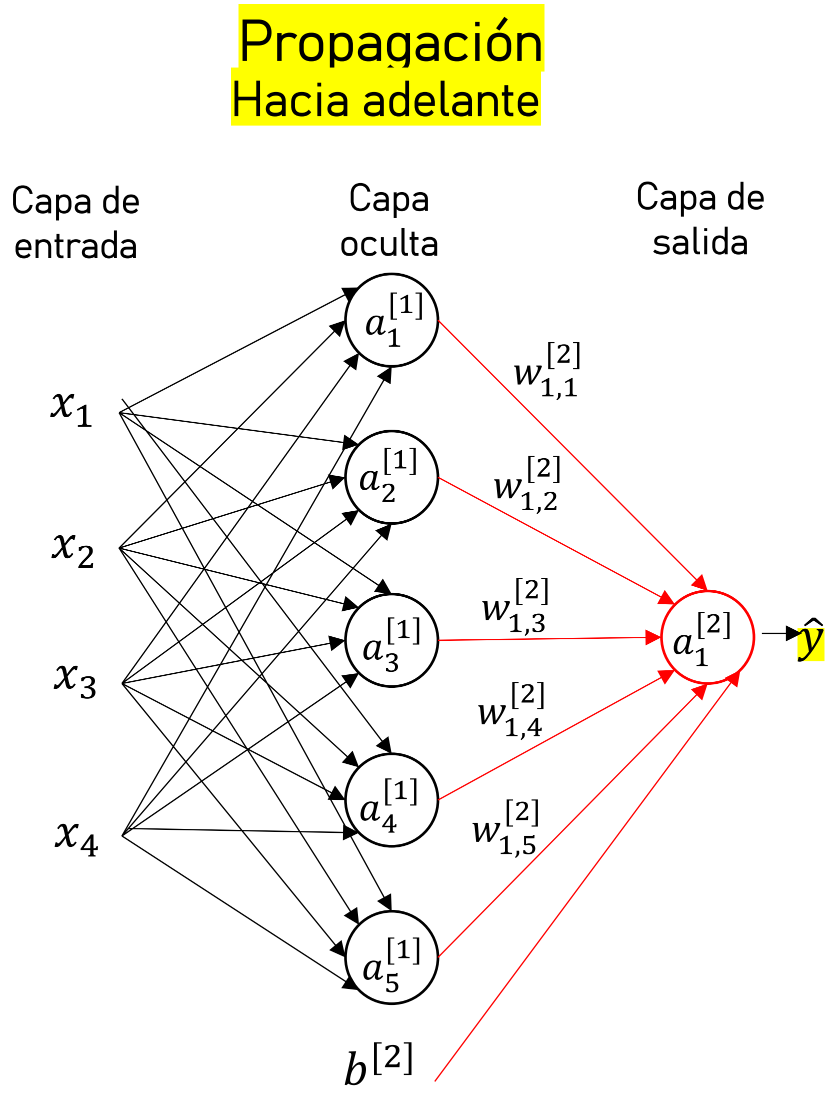
4. Este proceso continúa hasta que la capa de salida produce la predicción de la red.

Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.



5. Cálculo del error: Se calcula el error en la capa de salida entre la predicción de la red y el valor esperado.

Error cuadrático medio

$$E = \frac{1}{2}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(y_{j,k} - \hat{y}_{j,k}\right)^2$$

Para $J = \#ejemplos\ y\ K = \#salidas$

En este caso $J = 1\ y\ K = 1$

$$E = \frac{1}{2}(y - \hat{y})^2$$

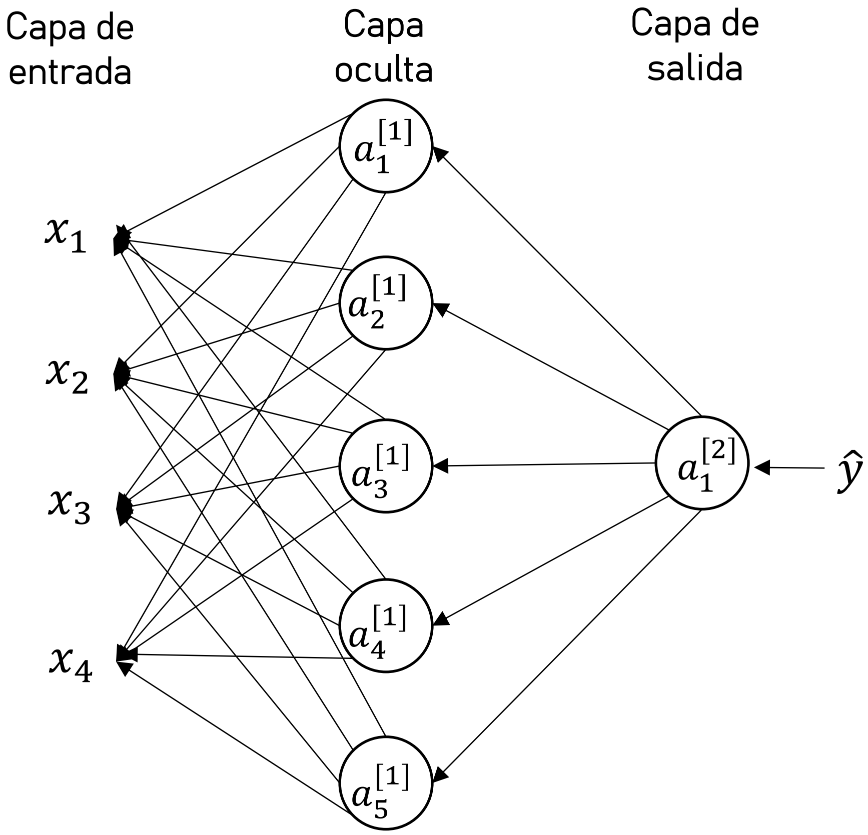
Meta: Obtener el error de la red considerando todos los ejemplos disponibles.

- Calculamos el valor de $\hat{Y}$, para los datos de entrada X
- Calculamos el error entre Y y $\hat{Y}$.

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

Propagación
Hacia atrás



Meta: Calcular los pesos $\boldsymbol{W}$ $\boldsymbol{y}$ $\boldsymbol{sesgos}$ $\boldsymbol{B}$, de tal forma que se minimice la función del error.

$$E = \frac{1}{2}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(y_{j,k} - \hat{y}_{j,k}\right)^2$$

Algoritmo de Retropropagación de errores

Es el algoritmo central usado para entrenar a la mayoría de las ANN.

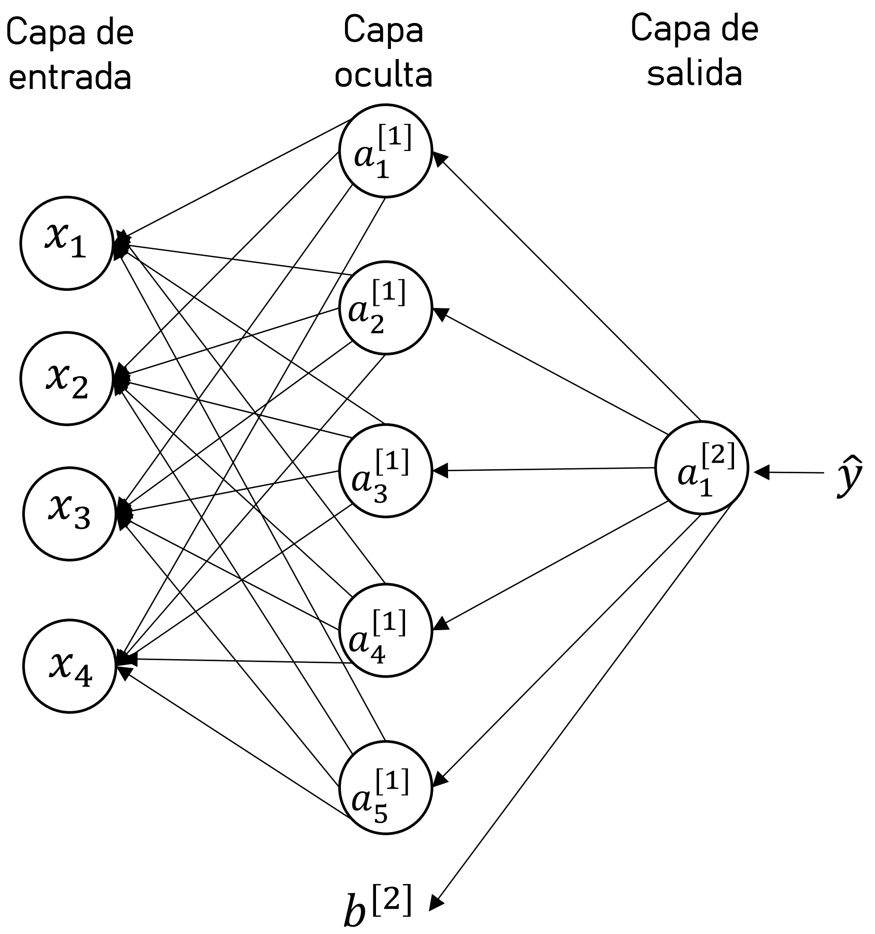
Propagación Hacia atrás

Meta: Calcular los pesos W y sesgos B, de tal forma que se minimice la función del error.

1. **Cálculo de los gradientes:** A partir del error obtenido en la propagación hacia adelante, se deriva la función de pérdida con respecto a las salidas de la red. Esto nos dice cuánto cambia el error si se cambian un poco las salidas.

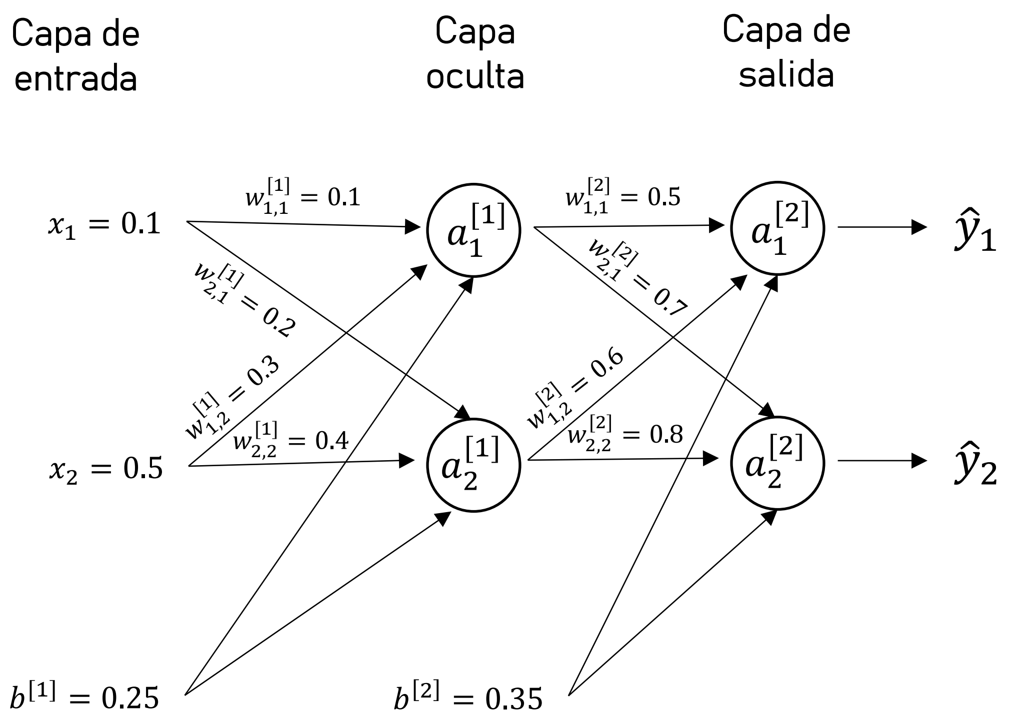
2. **Propagación del error:** Este error se propaga hacia atrás de la red, capa por capa. Usando la regla de la cadena, se calcula el gradiente de la pérdida con respecto a los pesos y sesgos de cada capa. Esto implica que debemos calcular el gradiente local de cada neurona (derivada de la función de activación) y luego multiplicarlo por el gradiente del error de la capa subsecuente.

3. **Actualización de pesos y sesgos:** Una vez que se calcularon todos los gradientes de la pérdida con respecto a todos los pesos y sesgos, estos gradientes se usan para optimizar los pesos y los sesgos usando un algoritmo de optimización (p.ej. **descenso por gradiente**). La regla de actualización normalmente implica restar una pequeña fracción del gradiente (escalado por la tasa de aprendizaje) a el valor actual del parámetro.



Perceptrones Multicapa y Retropropagación

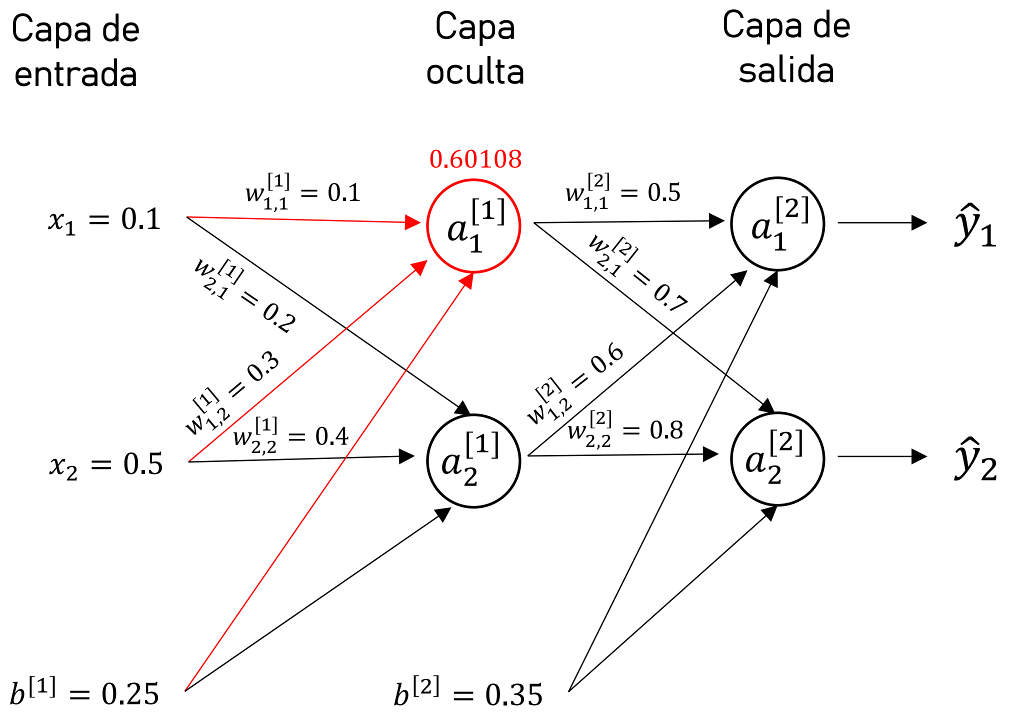
Ejemplo



- Las neuronas de la capa oculta usan la función de activación sigmoide.
- Las neuronas de la capa de salida usa la función lineal.

Perceptrones Multicapa y Retropropagación

Ejemplo



Propagación
Hacia adelante

Calculamos $a_1^{[1]}$

$$z_1^{[1]} = w_{1,1}^{[1]} x_1 + w_{1,2}^{[1]} x_2 + b^{[1]}$$

$$= (0.1)(0.1) + (0.3)(0.5) + 0.25 = 0.41$$

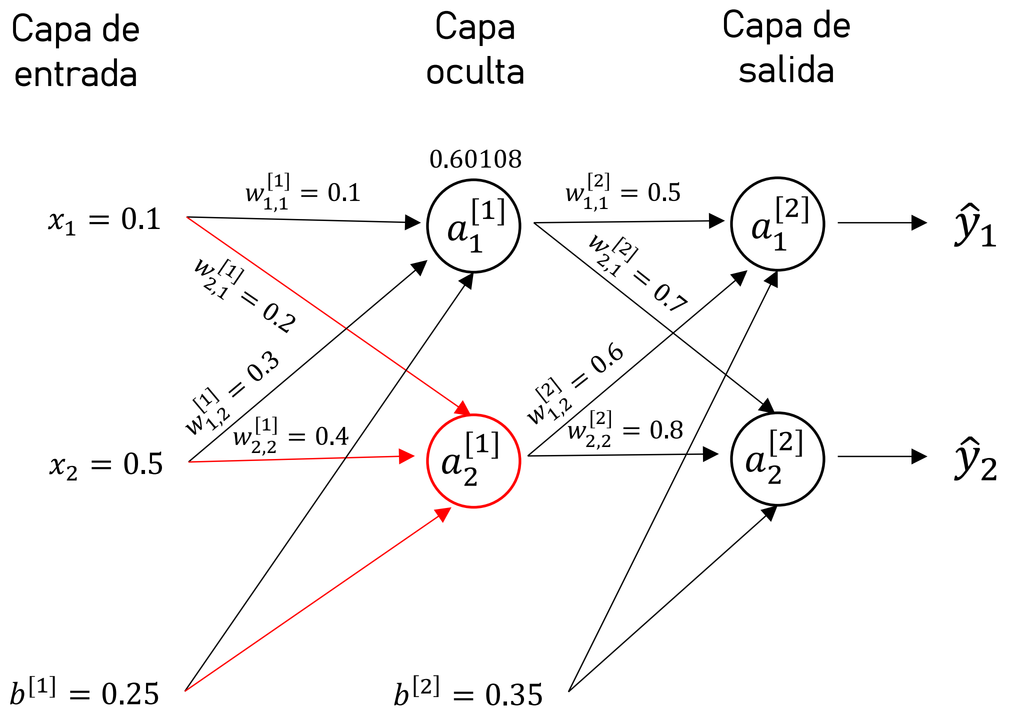
$$a_1^{[1]} = \boldsymbol{\sigma}\left(z_1^{[1]}\right) = \frac{1}{1 + e^{-z_1^{[1]}}}$$

$$= \frac{1}{1 + e^{-0.41}} = 0.60108$$

Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia adelante



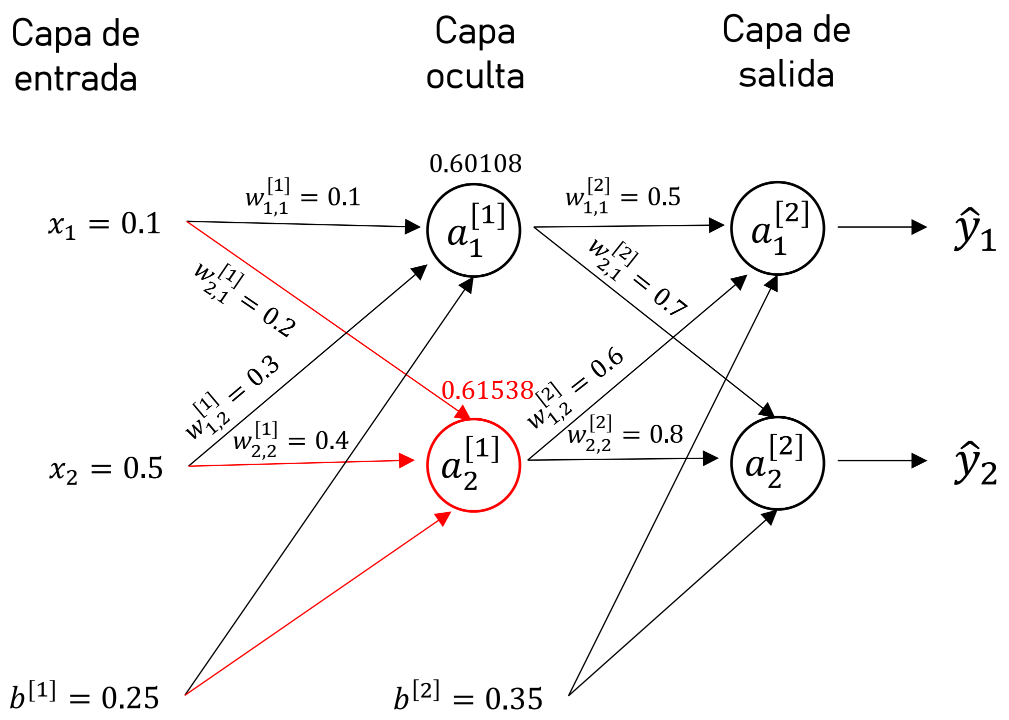
Calculamos $a_2^{[1]}$

$$z_2^{[1]} =$$

$$a_2^{[1]} = \boldsymbol{\sigma}\left(z_2^{[1]}\right) = \frac{1}{1 + e^{-z_2^{[1]}}} =$$

Perceptrones Multicapa y Retropropagación

Ejemplo



Propagación

Hacia adelante

Calculamos $a_2^{[1]}$

$$z_2^{[1]} = w_{2,1}^{[1]} x_1 + w_{2,2}^{[1]} x_2 + b^{[1]}$$

$$= (0.2)(0.1) + (0.4)(0.5) + 0.25 = 0.47$$

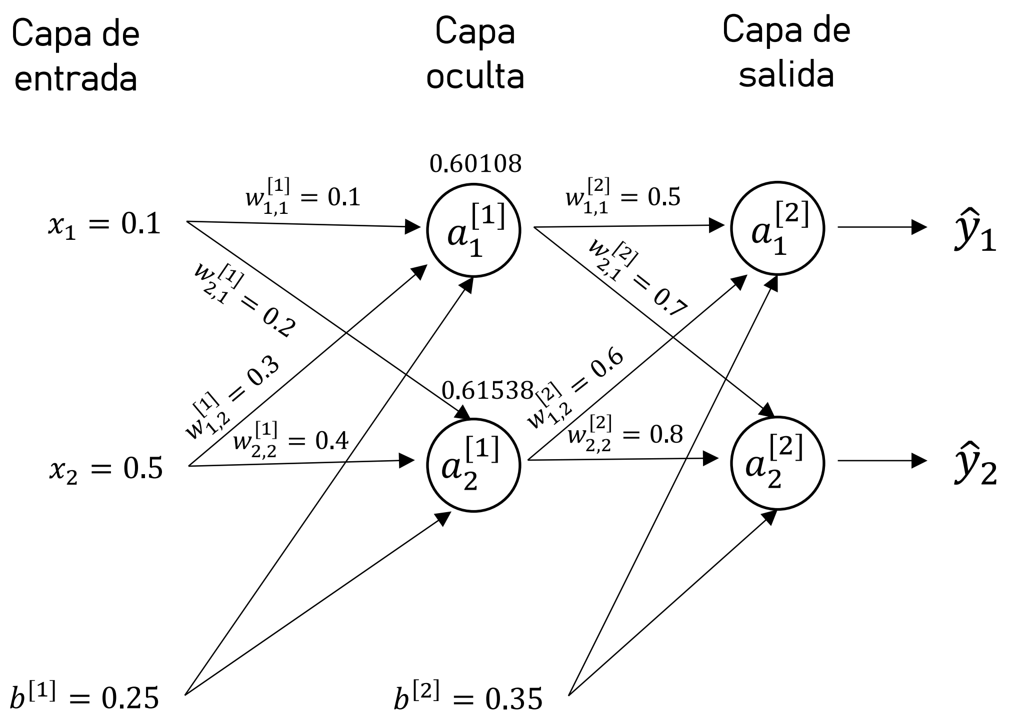
$$a_2^{[1]} = \boldsymbol{\sigma}\left(z_2^{[1]}\right) = \frac{1}{1 + e^{-z_2^{[1]}}}$$

$$= \frac{1}{1 + e^{-0.47}} = 0.61538$$

Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia adelante



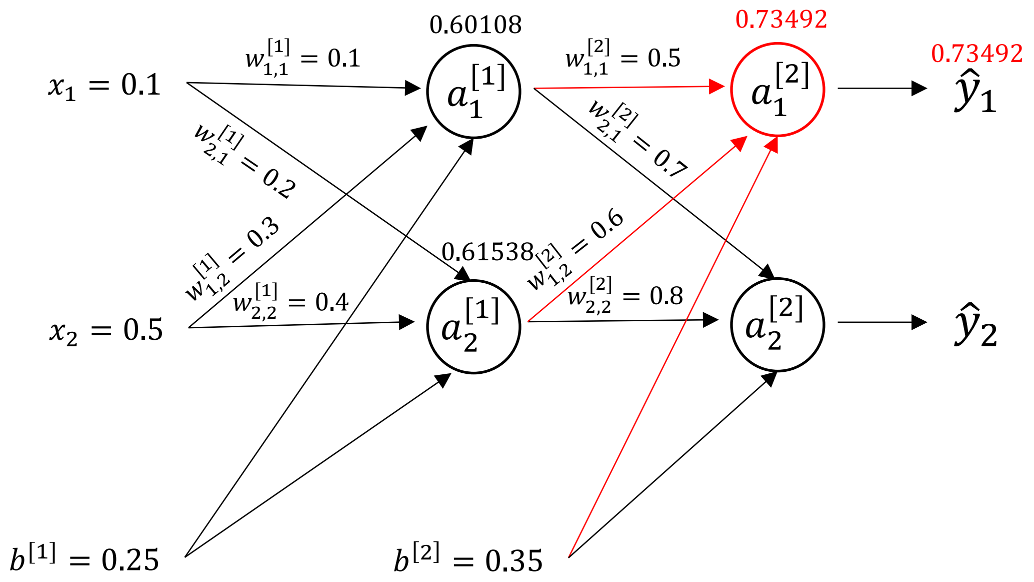
Calculamos $a_1^{[2]}$

$$z_1^{[2]} =$$

$$a_1^{[2]} = \boldsymbol{\sigma}\left(z_1^{[2]}\right) = \frac{1}{1 + e^{-z_1^{[2]}}} =$$

Perceptrones Multicapa y Retropropagación

Ejemplo



Propagación
Hacia adelante

Calculamos $a_1^{[2]}$

$$z_1^{[2]} = w_{1,1}^{[2]} x_1 + w_{1,2}^{[2]} x_2 + b^{[2]}$$

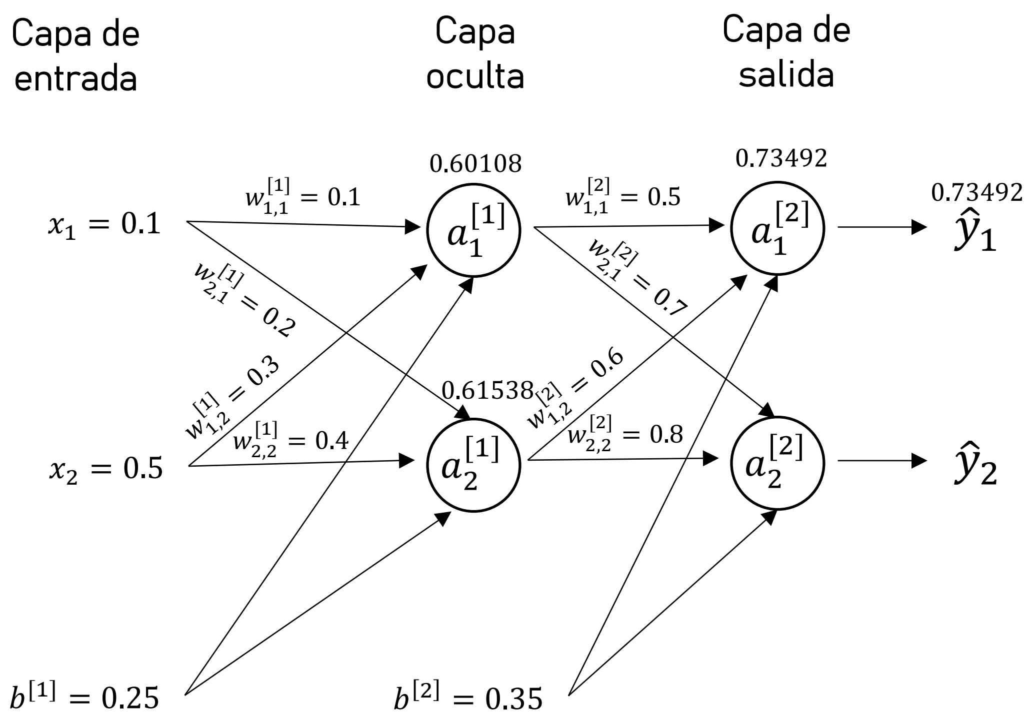
$$= (0.5)(0.60108) + (0.6)(0.61538) + 0.35$$
$$= 1.019768$$

$$a_1^{[2]} = \boldsymbol{\sigma}\left(z_1^{[2]}\right) = \frac{1}{1 + e^{-z_1^{[2]}}}$$

$$= \frac{1}{1 + e^{-1.019768}} = 0.73492$$

Perceptrones Multicapa y Retropropagación

Ejemplo



Propagación
Hacia adelante

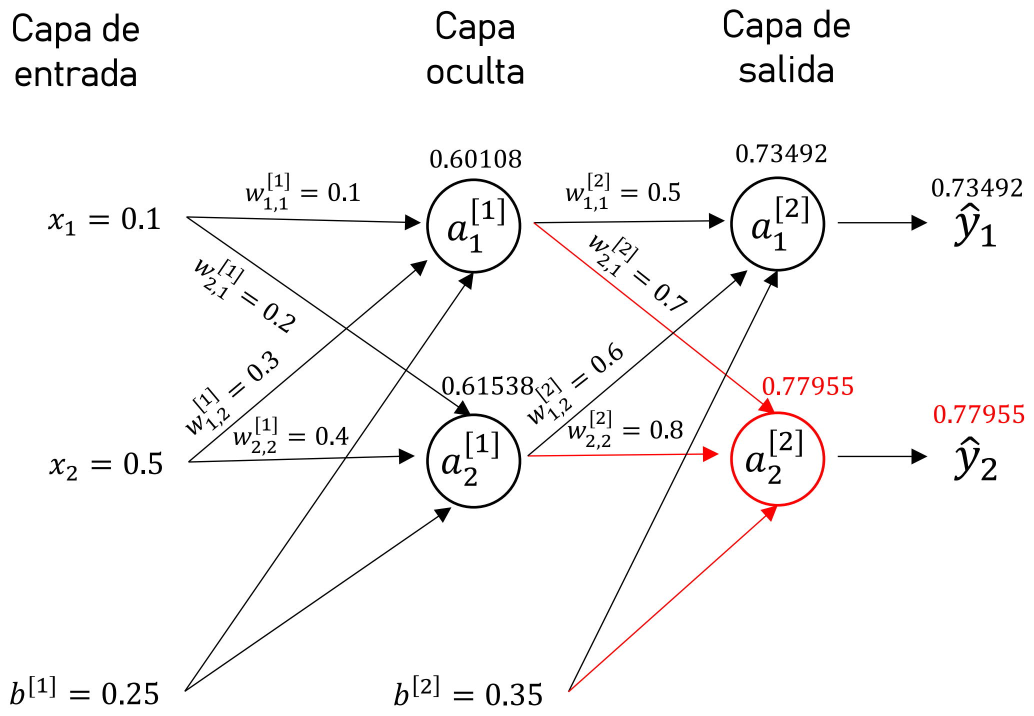
Calculamos $a_1^{[2]}$

$$z_2^{[2]} = $$

$$a_2^{[2]} = \boldsymbol{\sigma}\left(z_2^{[2]}\right) = \frac{1}{1 + e^{-z_2^{[2]}}} = $$

Perceptrones Multicapa y Retropropagación

Ejemplo



Propagación

Hacia adelante

Calculamos $a_1^{[2]}$

$$z_2^{[2]} = w_{2,1}^{[2]} x_1 + w_{2,2}^{[2]} x_2 + b^{[2]}$$

$$= (0.7)(0.60108) + (0.8)(0.61538) + 0.35$$
$$= 1.26306$$

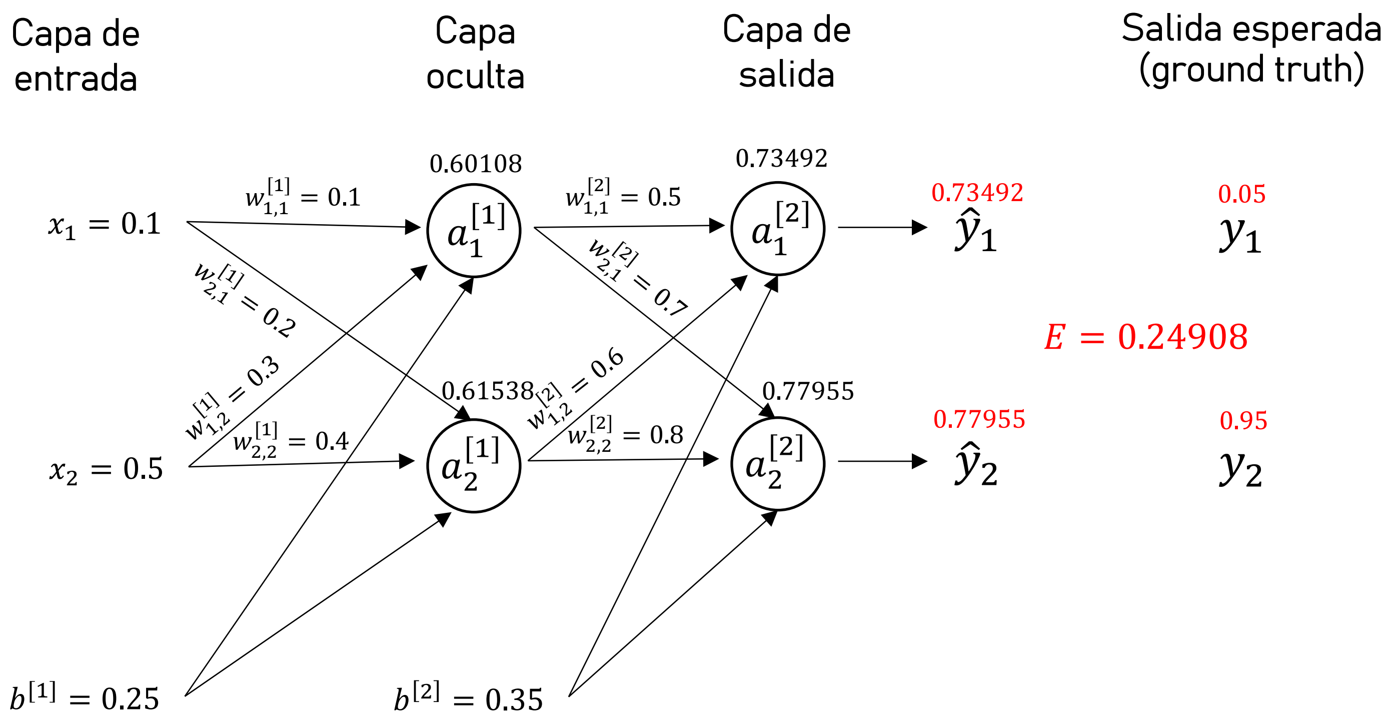
$$a_1^{[2]} = \boldsymbol{\sigma}\left(z_1^{[2]}\right) = \frac{1}{1 + e^{-z_1^{[2]}}}$$

$$= \frac{1}{1 + e^{-1.26306}} = 0.77955$$

Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia adelante



Calculamos el error

$$E = \frac{1}{2}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(y_{j,k} - \hat{y}_{j,k}\right)^2$$

En este ejemplo $J = 1$ y $K = 2$

$$E = \frac{1}{2}\sum_{k=1}^{K}(y_k - \hat{y}_k)^2$$

$$= \frac{1}{2}\left((0.05 - 0.73492)^2 + (0.95 - 0.77955)^2\right)$$

$$= \frac{1}{2}\left((-0.68492)^2 + (0.17045)^2\right)$$

$$= \frac{1}{2}(0.46911 + 0.02905) = \frac{0.46911}{2} + \frac{0.02905}{2}$$

$$= 0.23456 + 0.01452 = \boxed{0.24908}$$

Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{1,1}^{[2]} = 0.5$ en E?

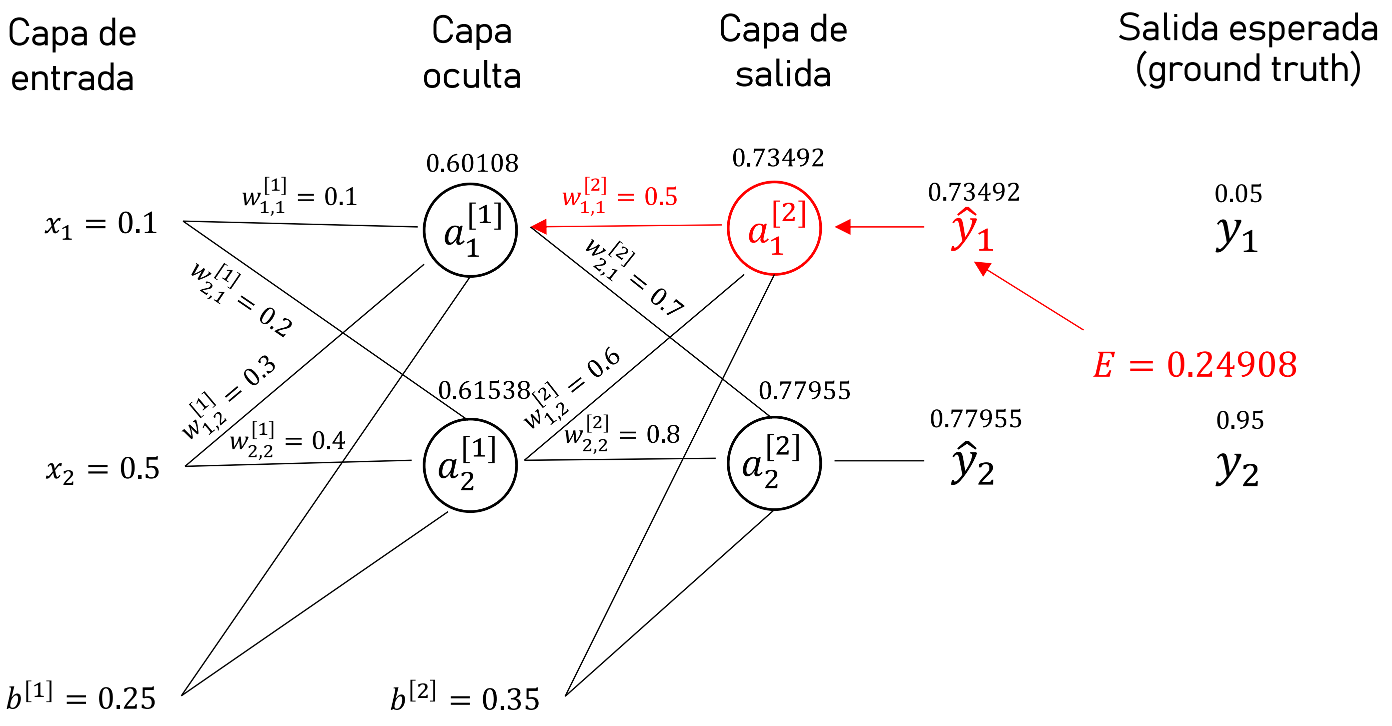
E está siendo afectado por $\hat{y}_1$

$\hat{y}_1$ está siendo afectado por $a_1^{[2]} \rightarrow z_1^{[2]}$

$z_1^{[2]}$ está siendo afectado por $w_{1,1}^{[2]}$

Por la regla de la cadena:

$$\underbrace{\frac{dE}{dw_{1,1}^{[2]}}}_{\text{¿Cómo cambia } E \text{ conforme cambia } w_{1,1}^{[2]}?} = \underbrace{\frac{dE}{d\hat{y}_1}}_{\text{¿Cómo cambia } E \text{ conforme cambia } \hat{y}_1?} \underbrace{\frac{d\hat{y}_1}{dz_1^{[2]}}}_{\text{¿Cómo cambia } \hat{y}_1 \text{ conforme cambia } a_1^{[2]}?} \underbrace{\frac{dz_1^{[2]}}{dw_{1,1}^{[2]}}}_{\text{¿Cómo cambia } a_1^{[2]} \text{ conforme cambia } w_{1,1}^{[2]}?}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{1,1}^{[2]} = 0.5$ en E?

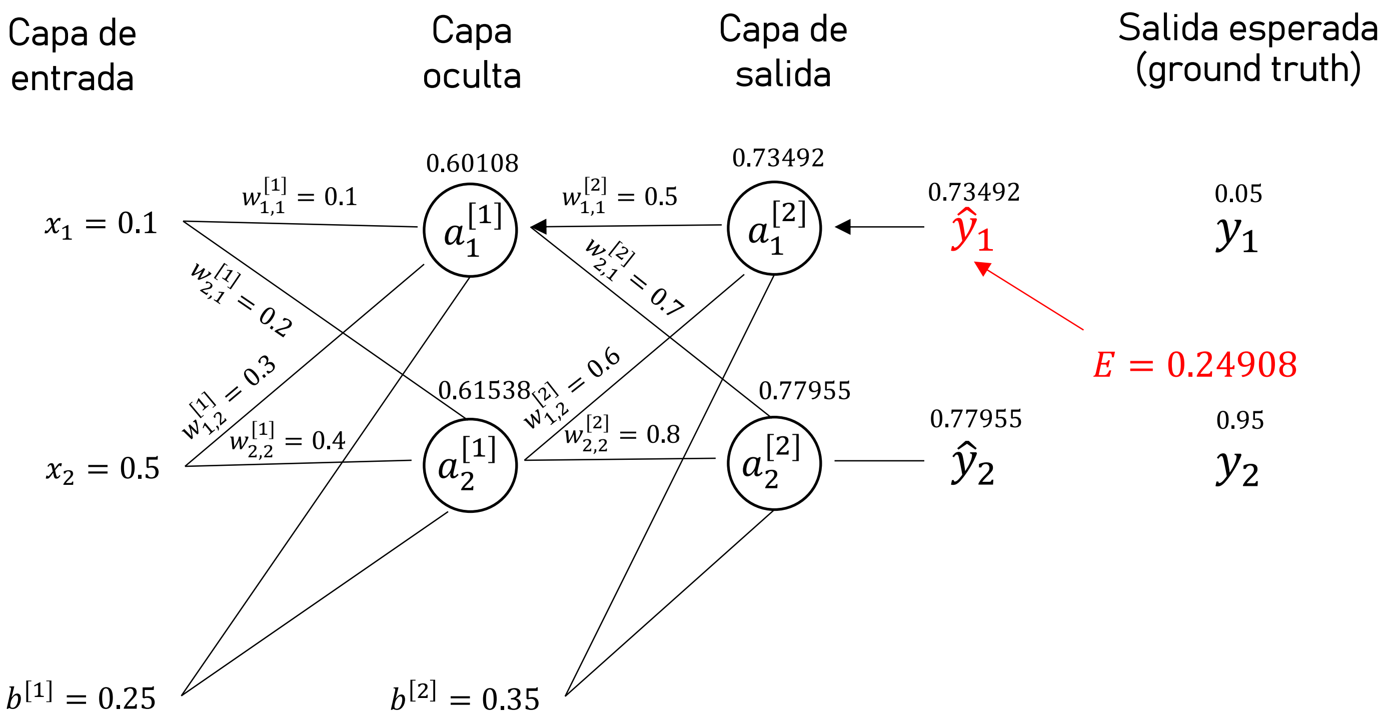
$$\frac{dE}{dw_{1,1}^{[2]}} = \boxed{\frac{dE}{d\hat{y}_1}} \frac{d\hat{y}_1}{dz_1^{[2]}} \frac{dz_1^{[2]}}{dw_{1,1}^{[2]}}$$

$$E = \frac{(y_1 - \hat{y}_1)^2}{2} + \frac{(y_2 - \hat{y}_2)^2}{2}$$

$$\frac{dE}{d\hat{y}_1} = \frac{2(y_1 - \hat{y}_1)}{2}(-1) = \hat{y}_1 - y_1$$

$$= 0.73492 - 0.05$$

$$= \boxed{-0.68492}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{1,1}^{[2]} = 0.5$ en E?

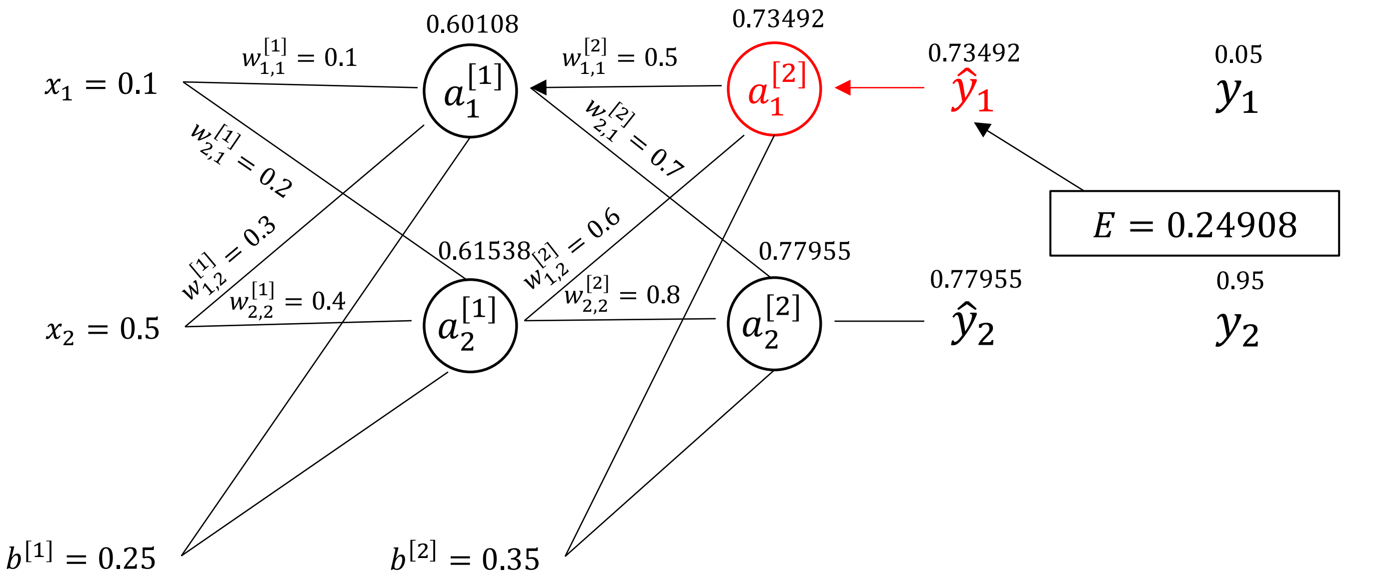
$$\frac{dE}{dw_{1,1}^{[2]}} = \frac{dE}{d\hat{y}_1} \boxed{\frac{d\hat{y}_1}{dz_1^{[2]}}} \frac{dz_1^{[2]}}{dw_{1,1}^{[2]}}$$

$$\hat{y}_1 = a_1^{[2]} = \boldsymbol{\sigma}\left(z_1^{[2]}\right) = \frac{1}{1 + e^{-z_1^{[2]}}}$$

$$\frac{d\hat{y}_1}{dz_1^{[2]}} =$$

Calcular:

$$\frac{d}{dx}\boldsymbol{\sigma}(x)$$



Nota

Derivada de la función sigmoide

$$\sigma(x) = \frac{1}{1+e^{-x}} = (1+e^{-x})^{-1}$$

$$\sigma'(x) = -1 \cdot (1+e^{-x})^{-2} \cdot \underbrace{\frac{d}{dx}(1+e^{-x})}_{-e^{-x}}$$

Nota:
$D_x\, e^u = e^u D_x u$

$$= (1+e^{-x})^{-2} \cdot e^{-x}$$

$$= \frac{e^{-x}}{(1+e^{-x})^2}$$

Como $\sigma(x) = \frac{1}{1+e^{-x}}$ y $1-\sigma(x) = 1 - \frac{1}{1+e^{-x}}$

$$= \frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}$$

$$= \frac{1+e^{-x}-1}{1+e^{-x}}$$

$$= \frac{e^{-x}}{1+e^{-x}}$$

$$\sigma(x)(1-\sigma(x)) = \frac{e^{-x}}{(1+e^{-x})^2}$$

Por lo tanto:

$$\boxed{\sigma'(x) = \sigma(x)\big(1-\sigma(x)\big)}$$

- Evita tener exponentes negativos en la fórmula,
- hace explícito que la derivada depende solo de σ(x)\sigma(x)σ(x),
- y se usa mucho en redes neuronales para ahorrar cálculos.

Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{1,1}^{[2]} = 0.5$ en E?

$$\frac{dE}{dw_{1,1}^{[2]}} = \frac{dE}{d\hat{y}_1} \boxed{\frac{d\hat{y}_1}{dz_1^{[2]}}} \frac{dz_1^{[2]}}{dw_{1,1}^{[2]}}$$

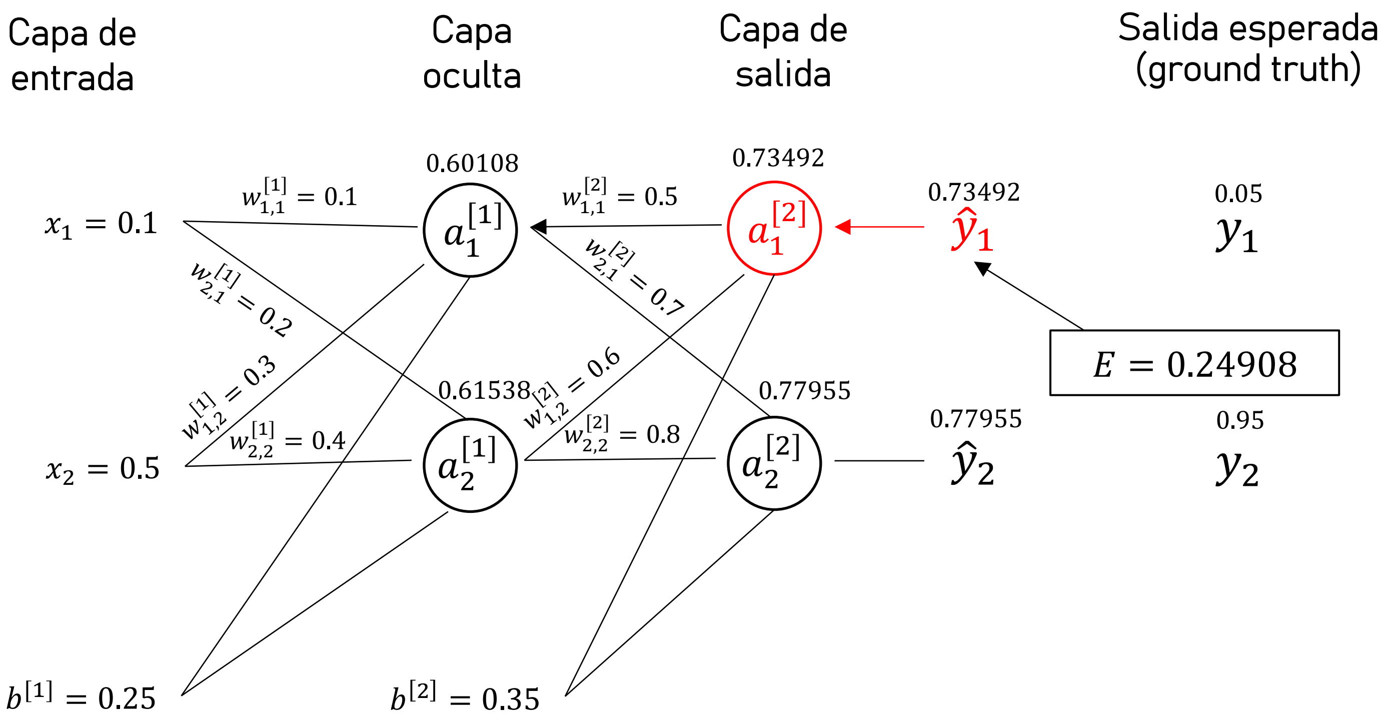
$$\hat{y}_1 = a_1^{[2]} = \boldsymbol{\sigma}\left(z_1^{[2]}\right) = \frac{1}{1 + e^{-z_1^{[2]}}}$$

$$\frac{d\hat{y}_1}{dz_1^{[2]}} = \boldsymbol{\sigma}(z_1^{[2]})(1 - \boldsymbol{\sigma}\left(z_1^{[2]}\right))$$

Nota: $\frac{d}{dx}\boldsymbol{\sigma}(x) = \boldsymbol{\sigma}(x)(1 - \boldsymbol{\sigma}(x))$

$$= \hat{y}_1\ (1 - \hat{y}_1) = 0.73492(1 - 0.73492)$$

$$= \boxed{0.19480}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

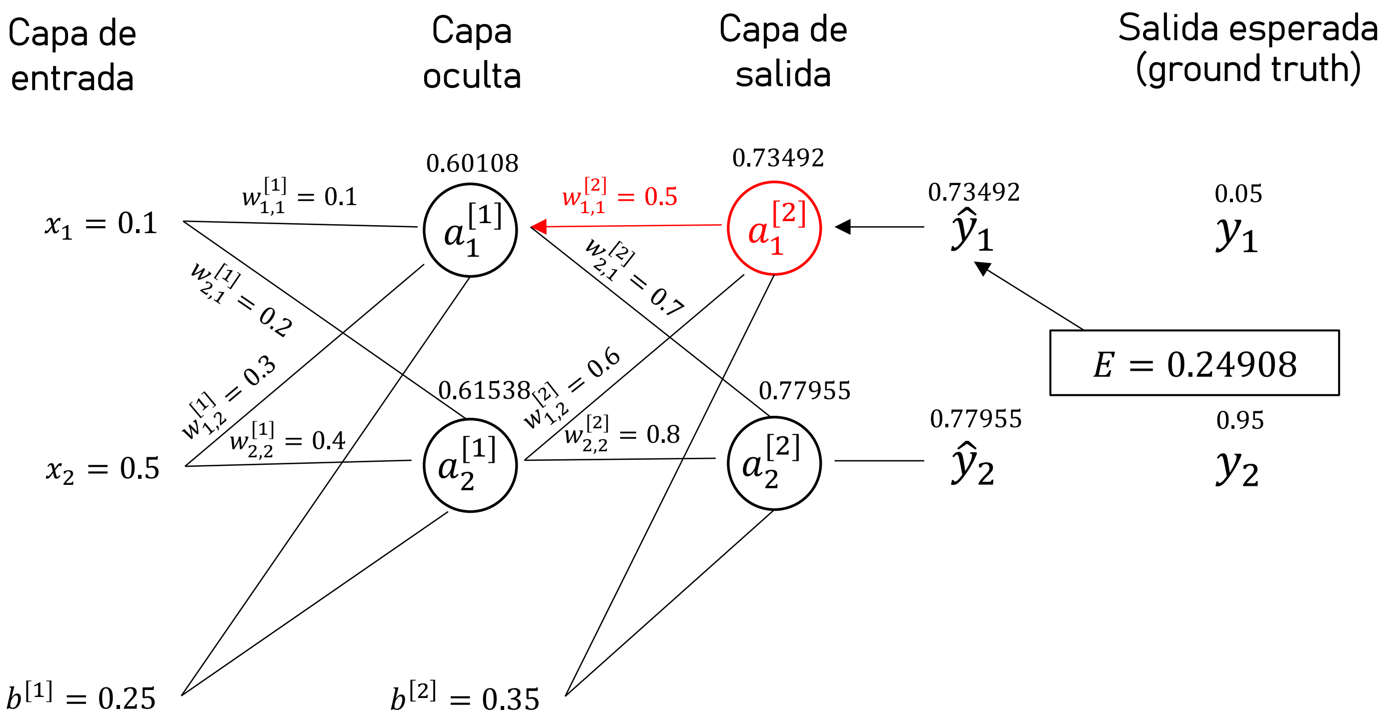
Propagación
Hacia atrás

¿Cuánto contribuye $w_{1,1}^{[2]} = 0.5$ en E?

$$\frac{dE}{dw_{1,1}^{[2]}} = \frac{dE}{d\hat{y}_1}\frac{d\hat{y}_1}{dz_1^{[2]}}\boxed{\frac{dz_1^{[2]}}{dw_{1,1}^{[2]}}}$$

$$z_1^{[2]} = w_{1,1}^{[2]}a_1^{[1]} + w_{1,2}^{[2]}a_2^{[1]} + b^{[2]}$$

$$\frac{dz_1^{[2]}}{dw_{1,1}^{[2]}} = a_1^{[1]} = \boxed{0.60108}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

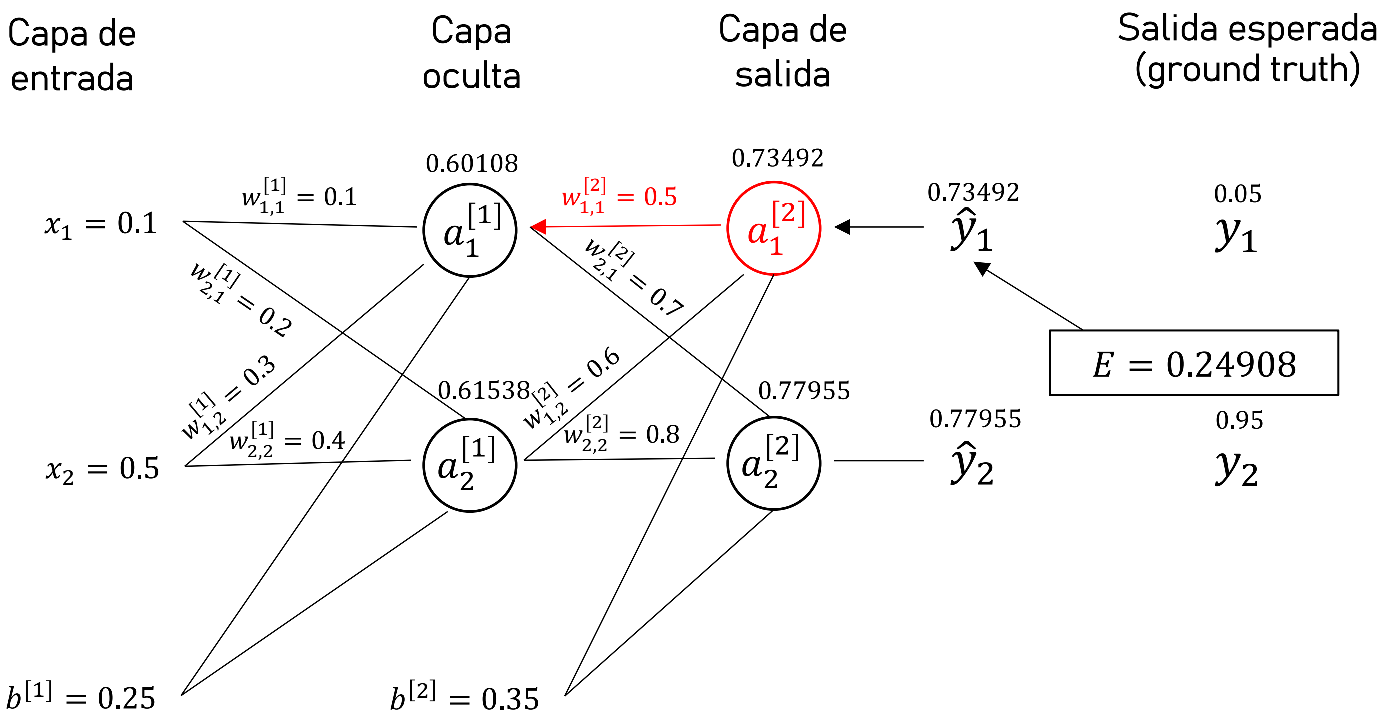
Propagación
Hacia atrás

¿Cuánto contribuye $w_{1,1}^{[2]} = 0.5$ en E?

$$\boxed{\frac{dE}{dw_{1,1}^{[2]}}} = \frac{dE}{d\hat{y}_1}\frac{d\hat{y}_1}{dz_1^{[2]}}\frac{dz_1^{[2]}}{dw_{1,1}^{[2]}}$$

$$= 0.68492 * 0.1980 * 0.60108$$

$$= \boxed{0.0802}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

Actualizamos $w_{1,1}^{[2]}$

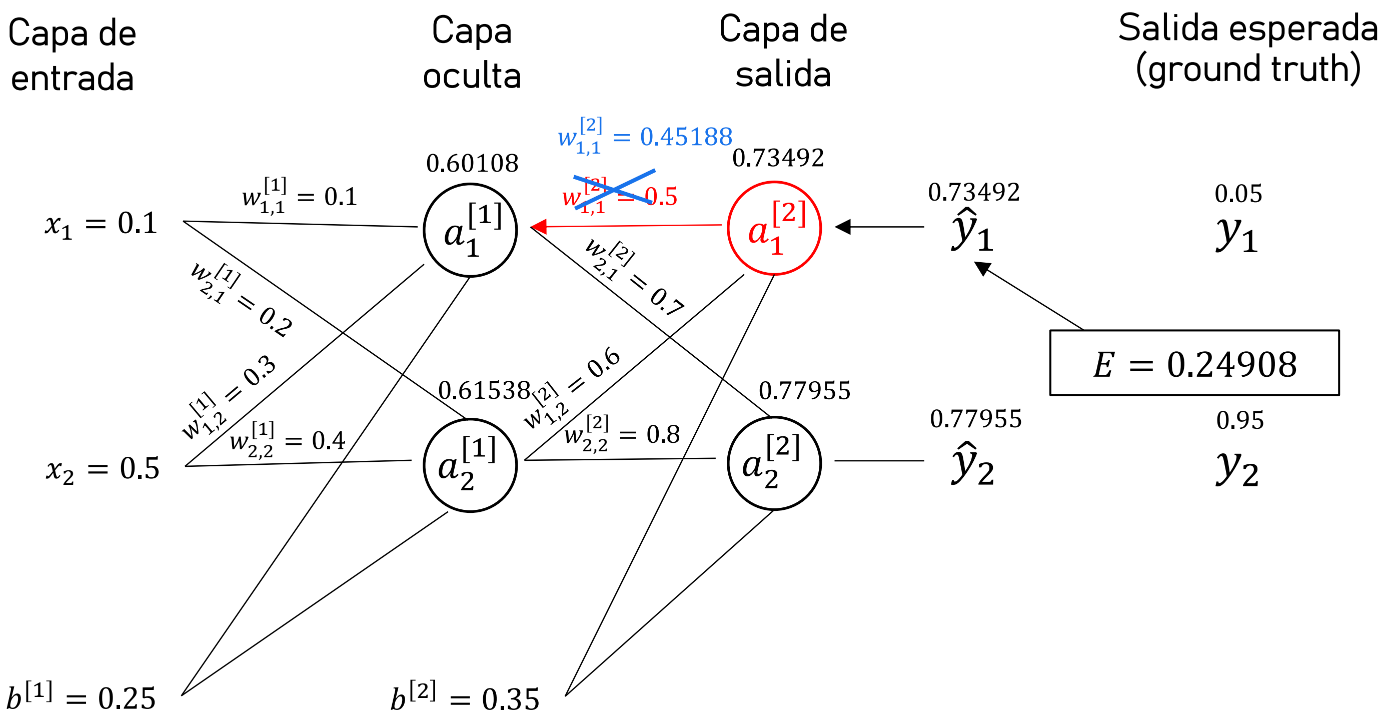
$\boldsymbol{\eta} = 0.6$

$$\boxed{w_{1,1}^{[2]}} = w_{1,1}^{[2]} - \boldsymbol{\eta}\left(\frac{dE}{dw_{1,1}^{[2]}}\right)$$

$$= 0.5 - 0.6(0.0802)$$

$$= 0.5 - 0.04812$$

$$= \boxed{0.45188}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{1,2}^{[2]} = 0.6$ en E?

$$\boxed{\frac{dE}{dw_{1,2}^{[2]}}} = \frac{dE}{d\hat{y}_1}\frac{d\hat{y}_1}{dz_1^{[2]}}\frac{dz_1^{[2]}}{dw_{1,2}^{[2]}}$$

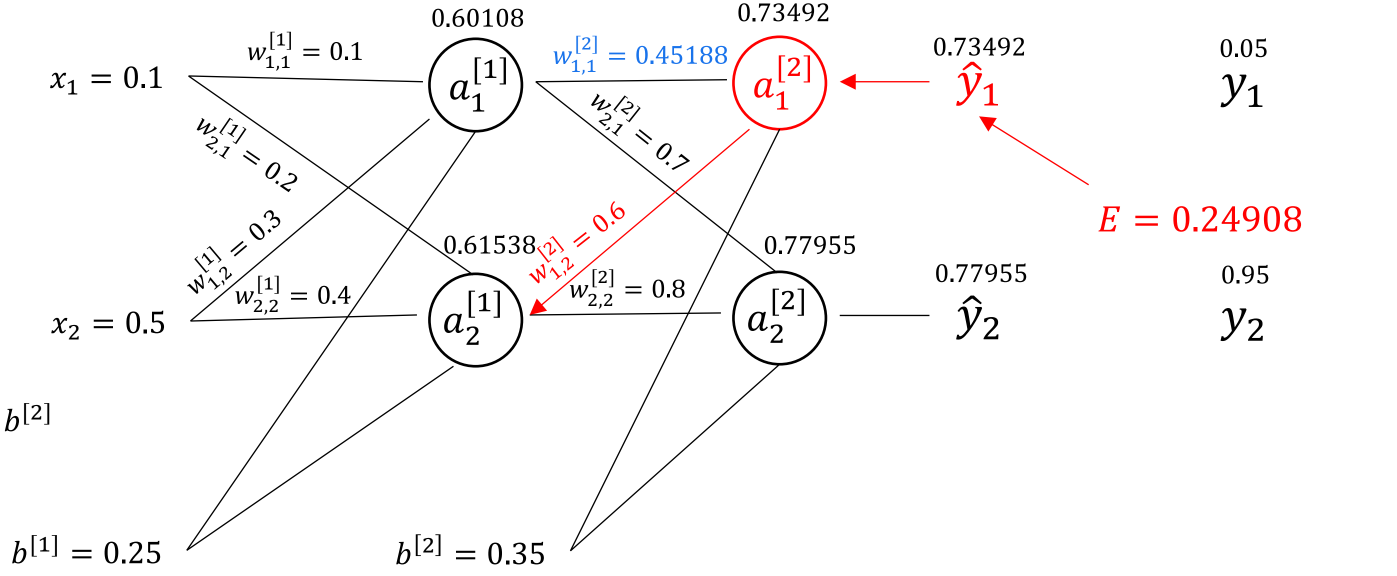
$\frac{dE}{d\hat{y}_1} \rightarrow 0.68492$ ✓

$\frac{d\hat{y}_1}{dz_1^{[2]}} \rightarrow 0.19480$ ✓

$\frac{dz_1^{[2]}}{dw_{1,2}^{[2]}} \rightarrow a_2^{[1]}$ = 0.61538

$z_1^{[2]} = w_{1,1}^{[2]}a_1^{[1]} + w_{1,2}^{[2]}a_2^{[1]} + b^{[2]}$

$= \boxed{0.08211}$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

Actualizamos $w_{1,2}^{[2]}$

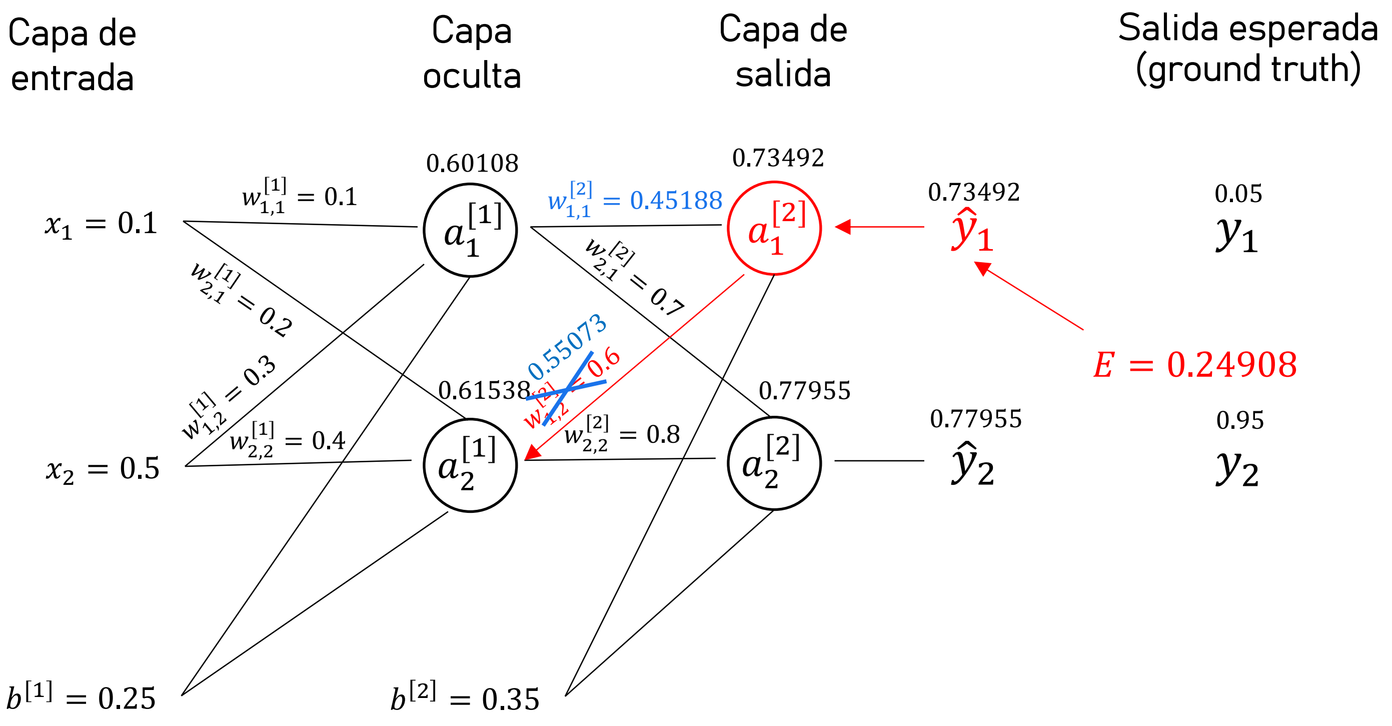
$\boldsymbol{\eta} = 0.6$

$$\boxed{w_{1,2}^{[2]}} = w_{1,2}^{[2]} - \boldsymbol{\eta}\left(\frac{dE}{dw_{1,2}^{[2]}}\right)$$

$$= 0.6 - 0.6(0.08211)$$

$$= 0.6 - 0.04926$$

$$= \boxed{0.55073}$$



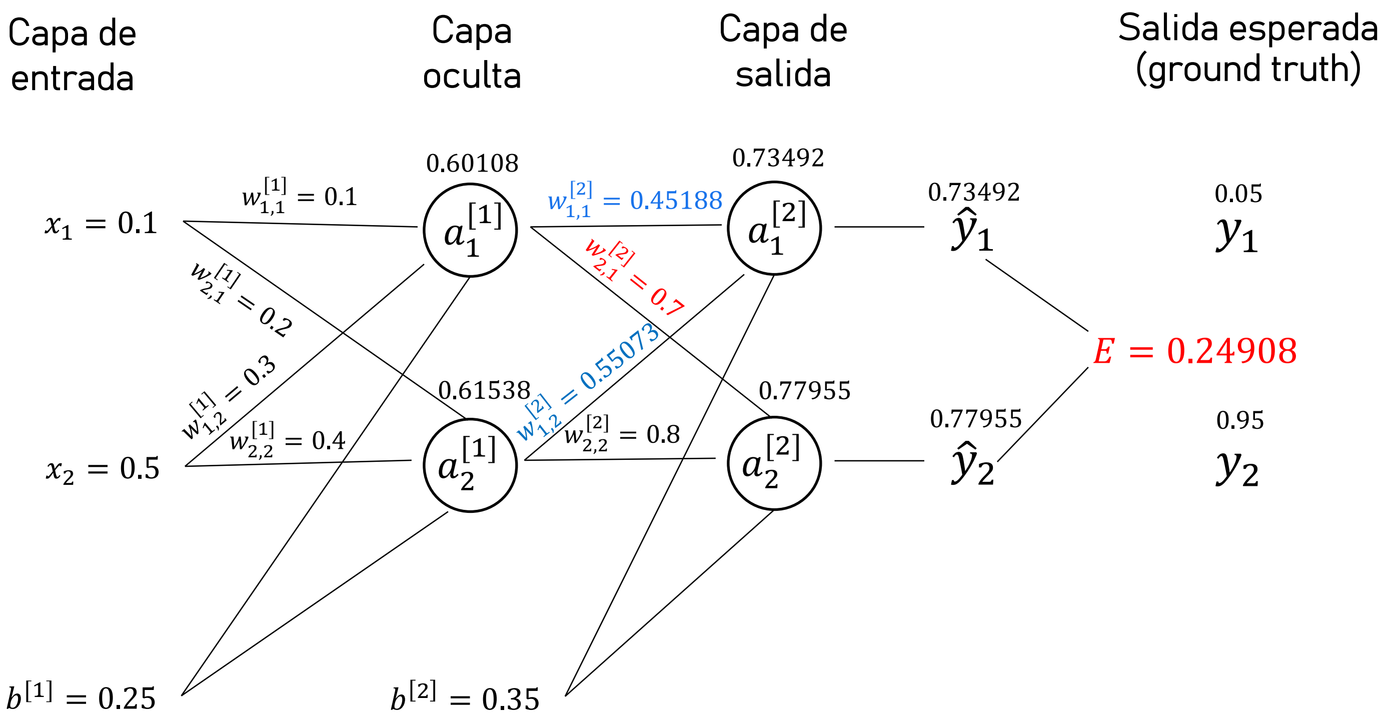
Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?



Perceptrones Multicapa y Retropropagación

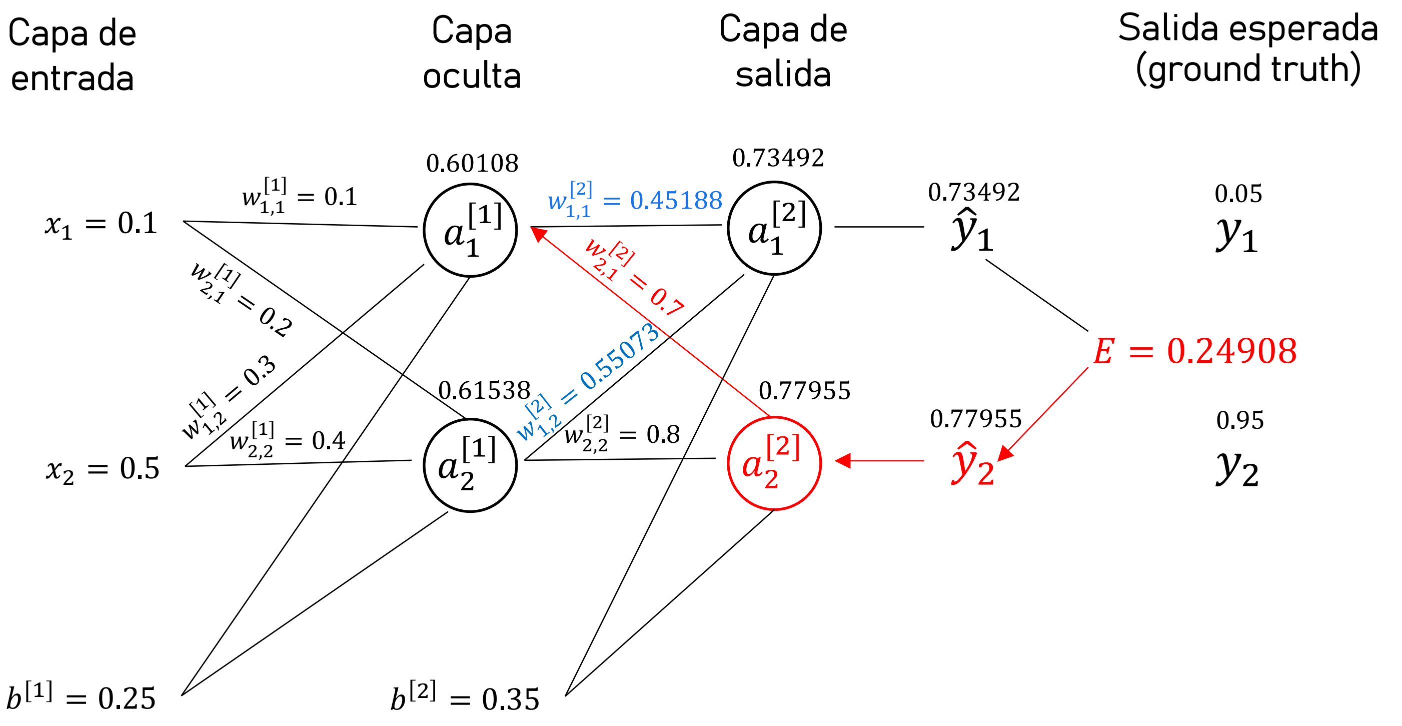
Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?

$$\frac{dE}{dw_{2,1}^{[2]}} = \frac{dE}{d\hat{y}_2} \frac{d\hat{y}_2}{dz_2^{[2]}} \frac{dz_2^{[2]}}{dw_{2,1}^{[2]}}$$



Perceptrones Multicapa y Retropropagación

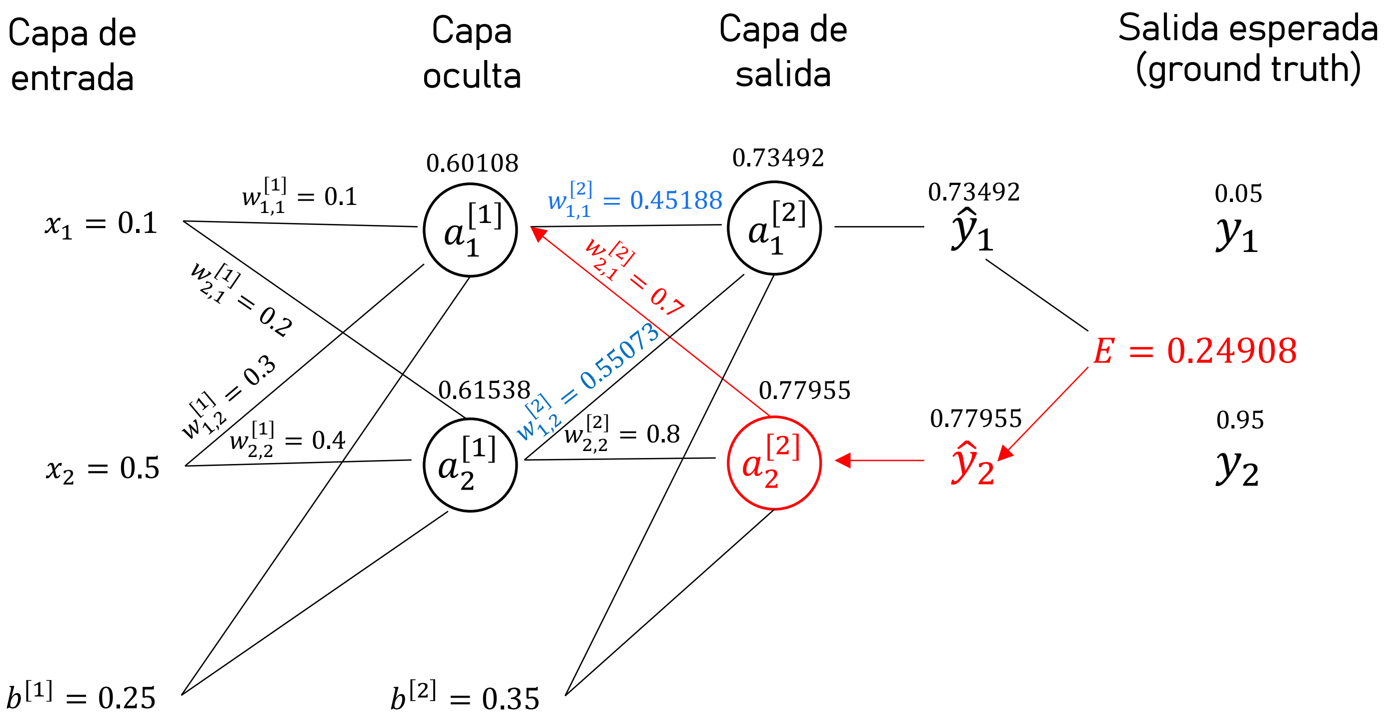
Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?

$$\frac{dE}{dw_{2,1}^{[2]}} = \boxed{\frac{dE}{d\hat{y}_2}} \frac{d\hat{y}_2}{dz_2^{[2]}} \frac{dz_2^{[2]}}{dw_{2,1}^{[2]}}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?

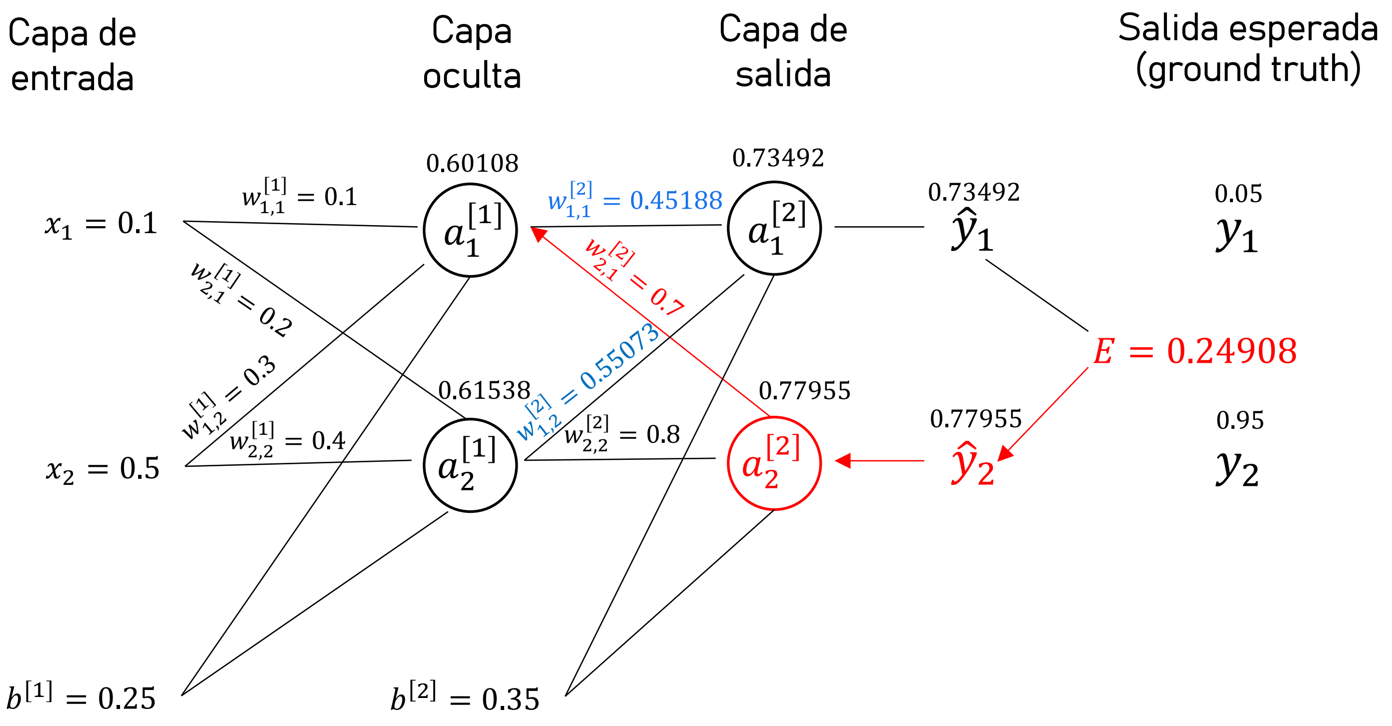
$$\frac{dE}{dw_{2,1}^{[2]}} = \boxed{\frac{dE}{d\hat{y}_2}} \frac{d\hat{y}_2}{dz_2^{[2]}} \frac{dz_2^{[2]}}{dw_{2,1}^{[2]}}$$

$$E = \frac{(y_1 - \hat{y}_1)^2}{2} + \frac{(y_2 - \hat{y}_2)^2}{2}$$

$$\frac{dE}{d\hat{y}_2} = \frac{2(y_2 - \hat{y}_2)}{2}(-1) = \hat{y}_2 - y_2$$

$$= 0.77955 - 0.95$$

$$= \boxed{-0.17044}$$



Perceptrones Multicapa y Retropropagación

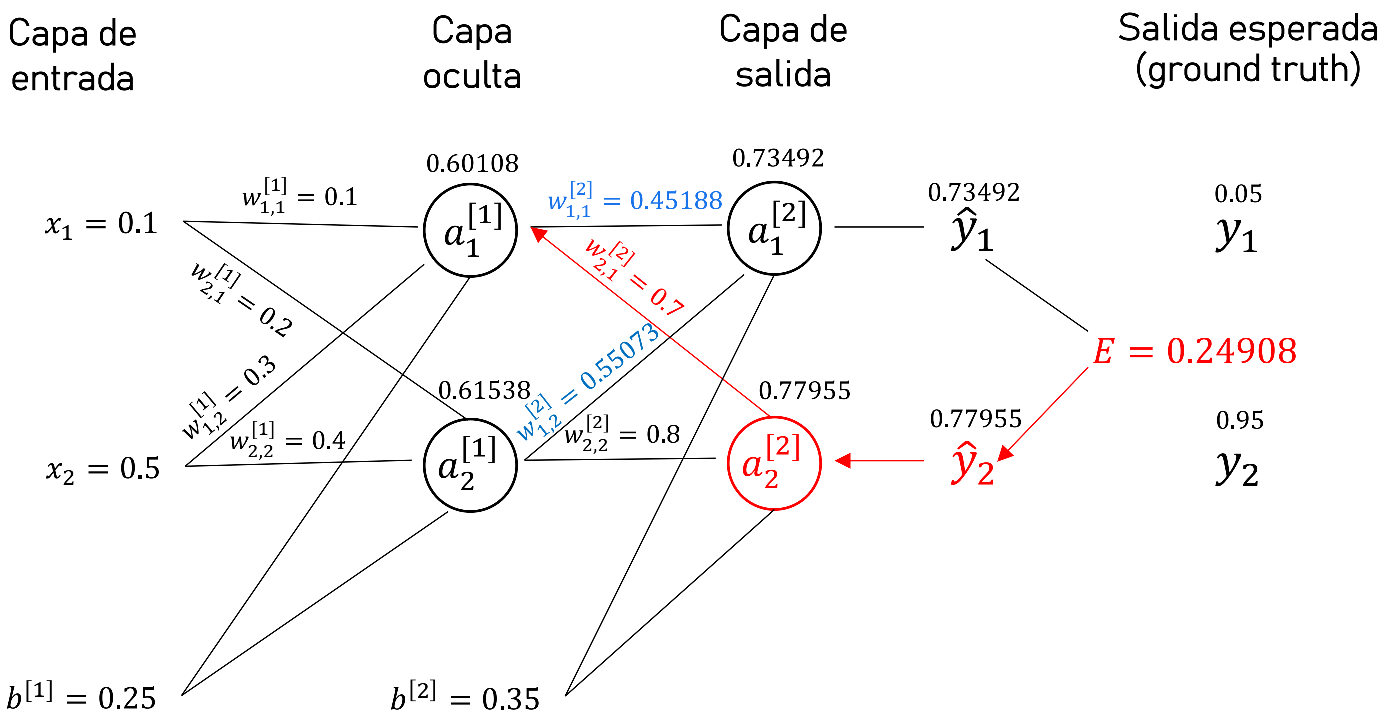
Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?

$$\frac{dE}{dw_{2,1}^{[2]}} = \frac{dE}{d\hat{y}_2} \boxed{\frac{d\hat{y}_2}{dz_2^{[2]}}} \frac{dz_2^{[2]}}{dw_{2,1}^{[2]}}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?

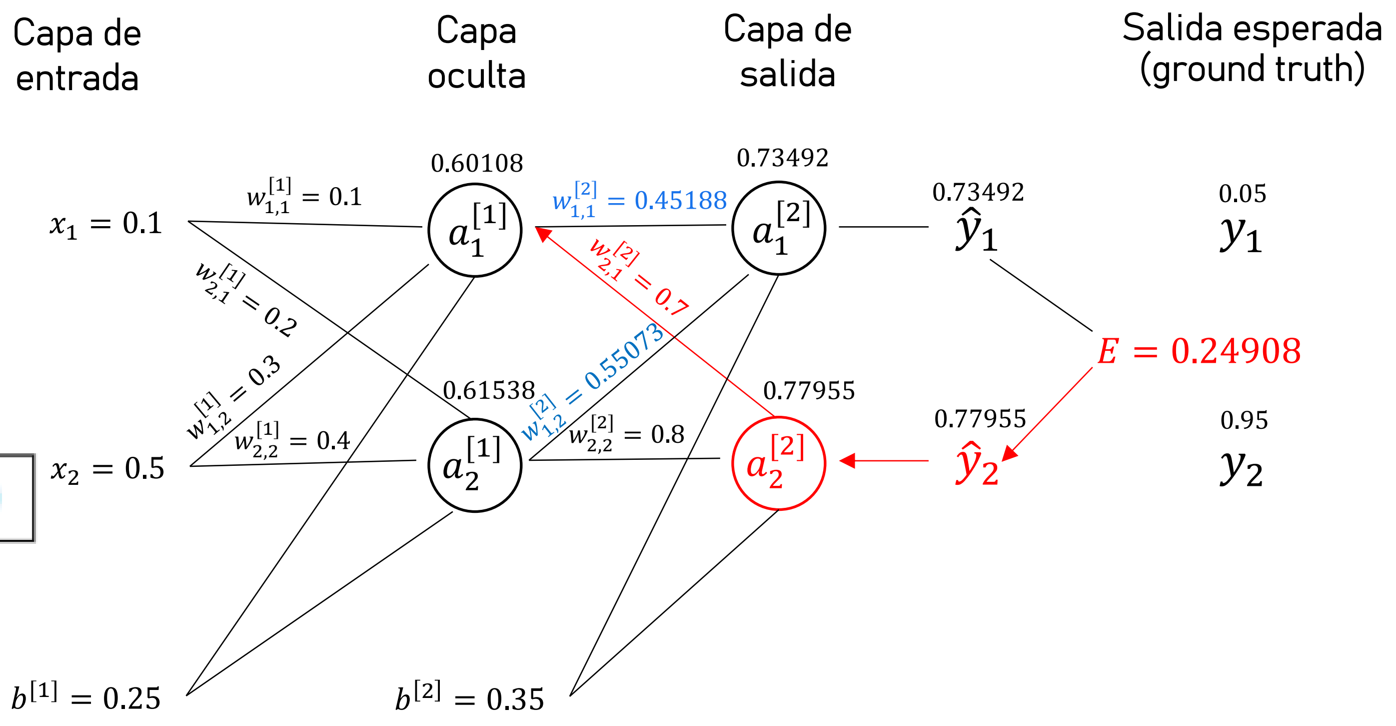
$$\frac{dE}{dw_{2,1}^{[2]}} = \frac{dE}{d\hat{y}_2} \boxed{\frac{d\hat{y}_2}{dz_2^{[2]}}} \frac{dz_2^{[2]}}{dw_{2,1}^{[2]}}$$

$$\hat{y}_2 = \boldsymbol{\sigma}\left(z_2^{[2]}\right) \qquad \boxed{\sigma'(x) = \sigma(x)(1 - \sigma(x))}$$

$$\frac{d\hat{y}_2}{dz_2^{[2]}} = \boldsymbol{\sigma}\left(z_2^{[2]}\right)\left(1 - \boldsymbol{\sigma}\left(z_2^{[2]}\right)\right)$$

$$= 0.77955(1 - 0.77955)$$

$$= \boxed{0.17184}$$



Perceptrones Multicapa y Retropropagación

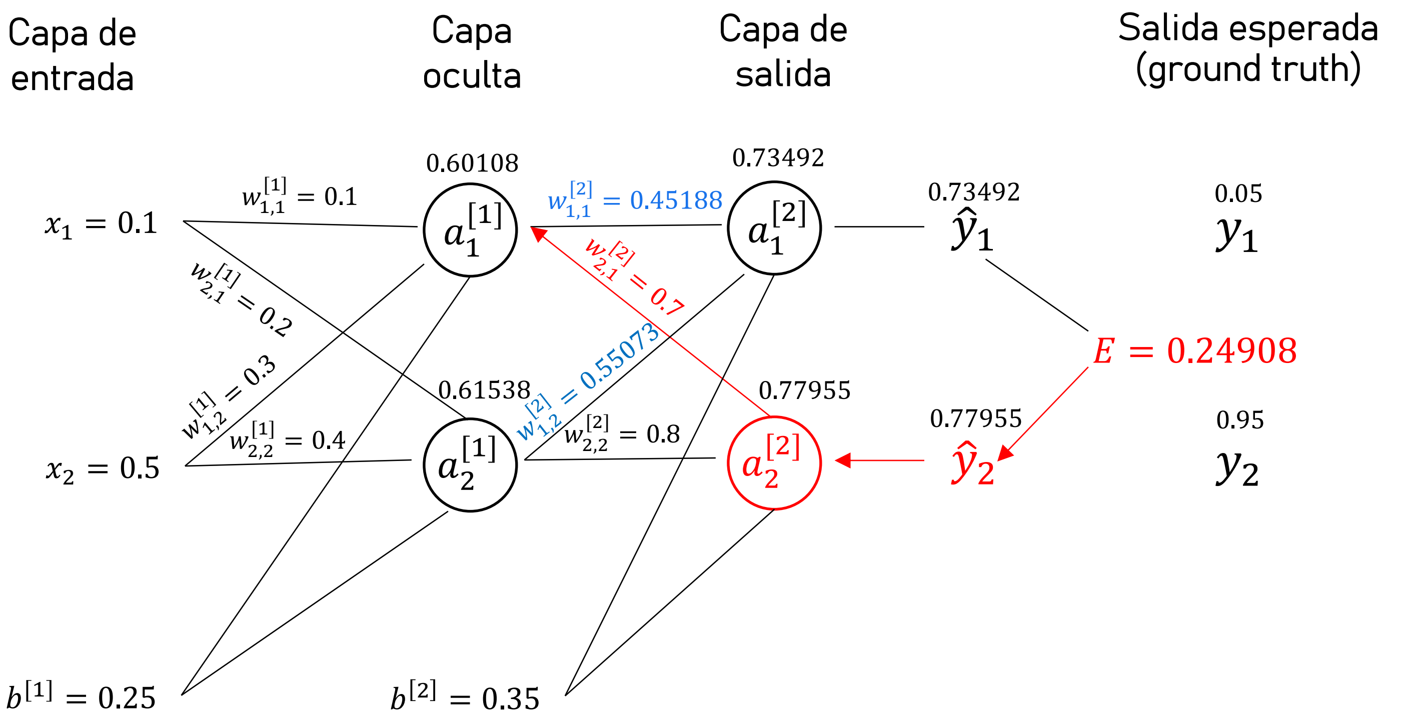
Ejemplo

Propagación
Hacia atrás

¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?

$$\frac{dE}{dw_{2,1}^{[2]}} = \frac{dE}{d\hat{y}_2}\frac{d\hat{y}_2}{dz_2^{[2]}}\boxed{\frac{dz_2^{[2]}}{dw_{2,1}^{[2]}}}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

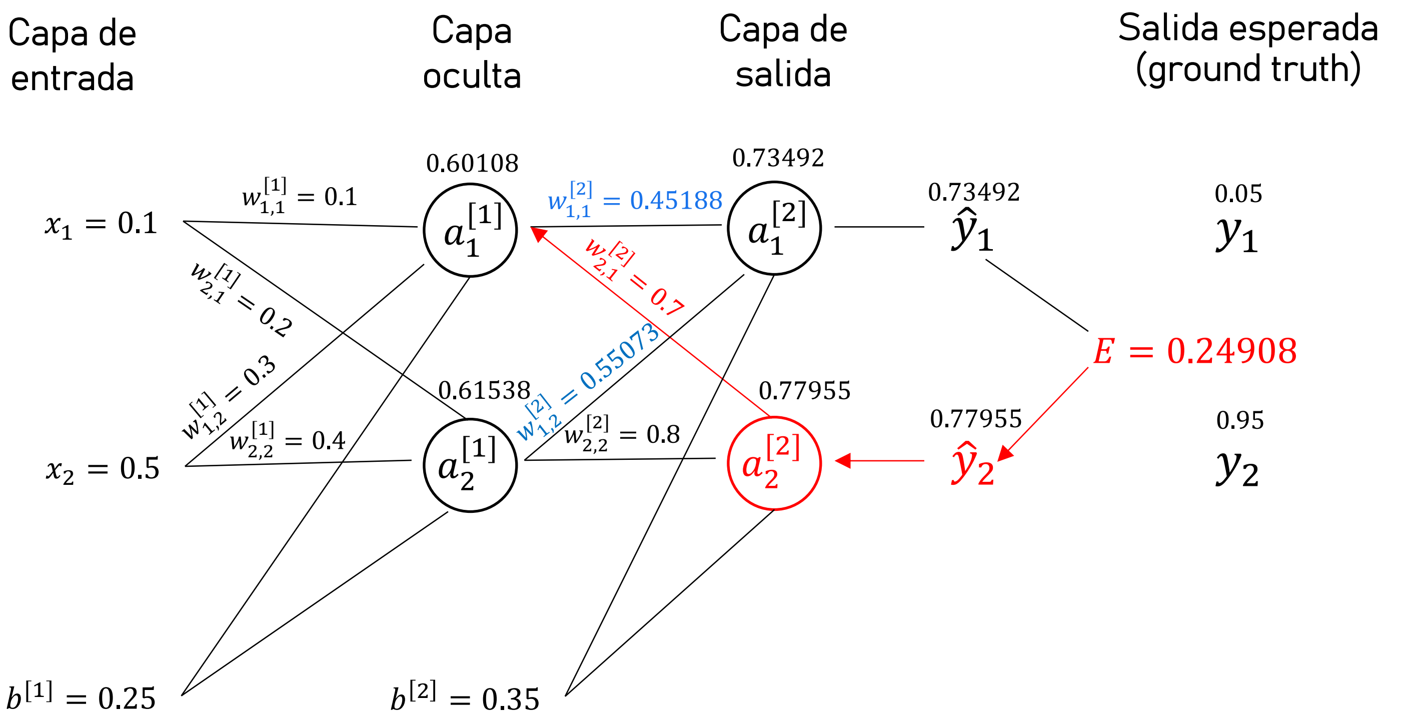
¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?

$$\frac{dE}{dw_{2,1}^{[2]}} = \frac{dE}{d\hat{y}_2}\frac{d\hat{y}_2}{dz_2^{[2]}}\boxed{\frac{dz_2^{[2]}}{dw_{2,1}^{[2]}}}$$

$$z_2^{[2]} = w_{2,1}^{[2]}a_1^{[1]} + w_{2,2}^{[2]}a_2^{[1]}$$

$$\frac{dz_2^{[2]}}{dw_{2,1}^{[2]}} = a_1^{[1]} = \boxed{0.60108}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

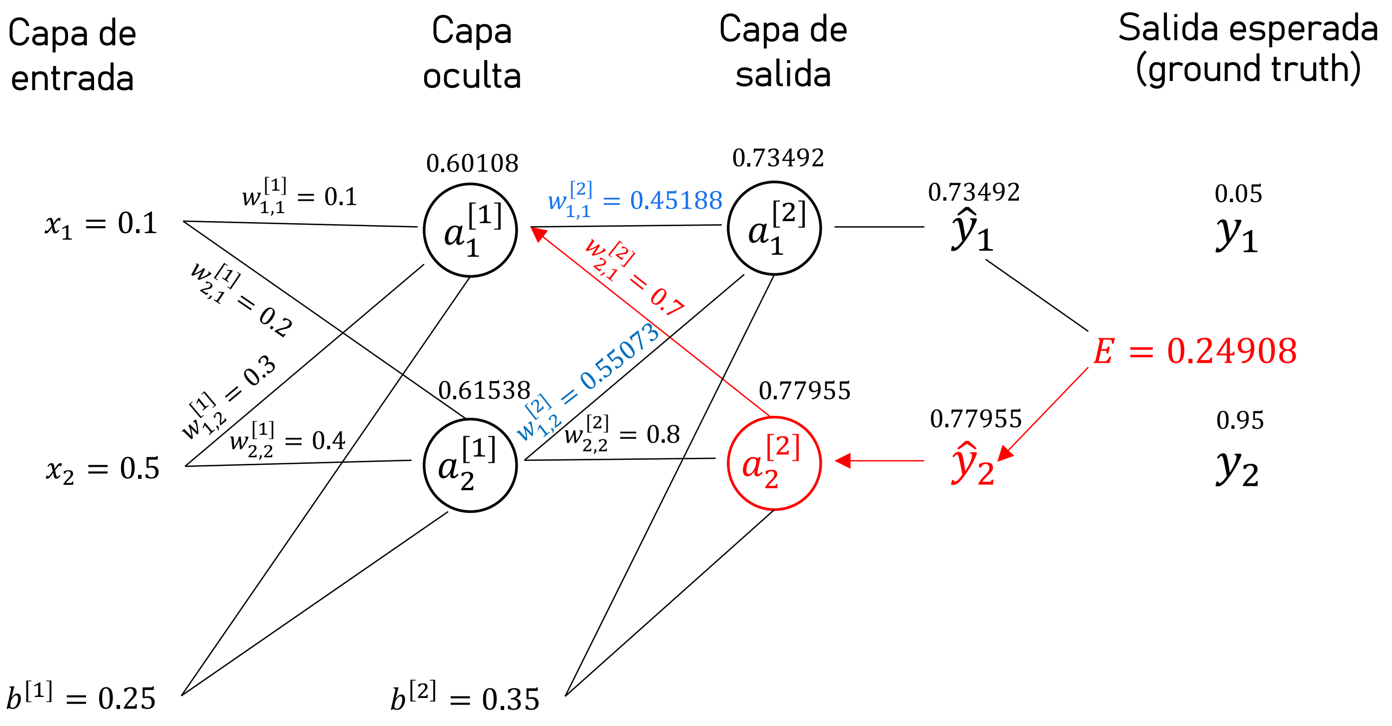
¿Cuánto contribuye $w_{2,1}^{[2]} = 0.7$ en E?

¿Qué derivada hay que calcular?

$$\boxed{\frac{dE}{dw_{2,1}^{[2]}}} = \frac{dE}{d\hat{y}_2}\frac{d\hat{y}_2}{dz_2^{[2]}}\frac{dz_2^{[2]}}{dw_{2,1}^{[2]}}$$

$$= -0.17044 * 0.17184 * 0.60108$$

$$= \boxed{-0.01760}$$



Perceptrones Multicapa y Retropropagación

Ejemplo

Propagación
Hacia atrás

Actualizamos $w_{2,1}^{[2]}$

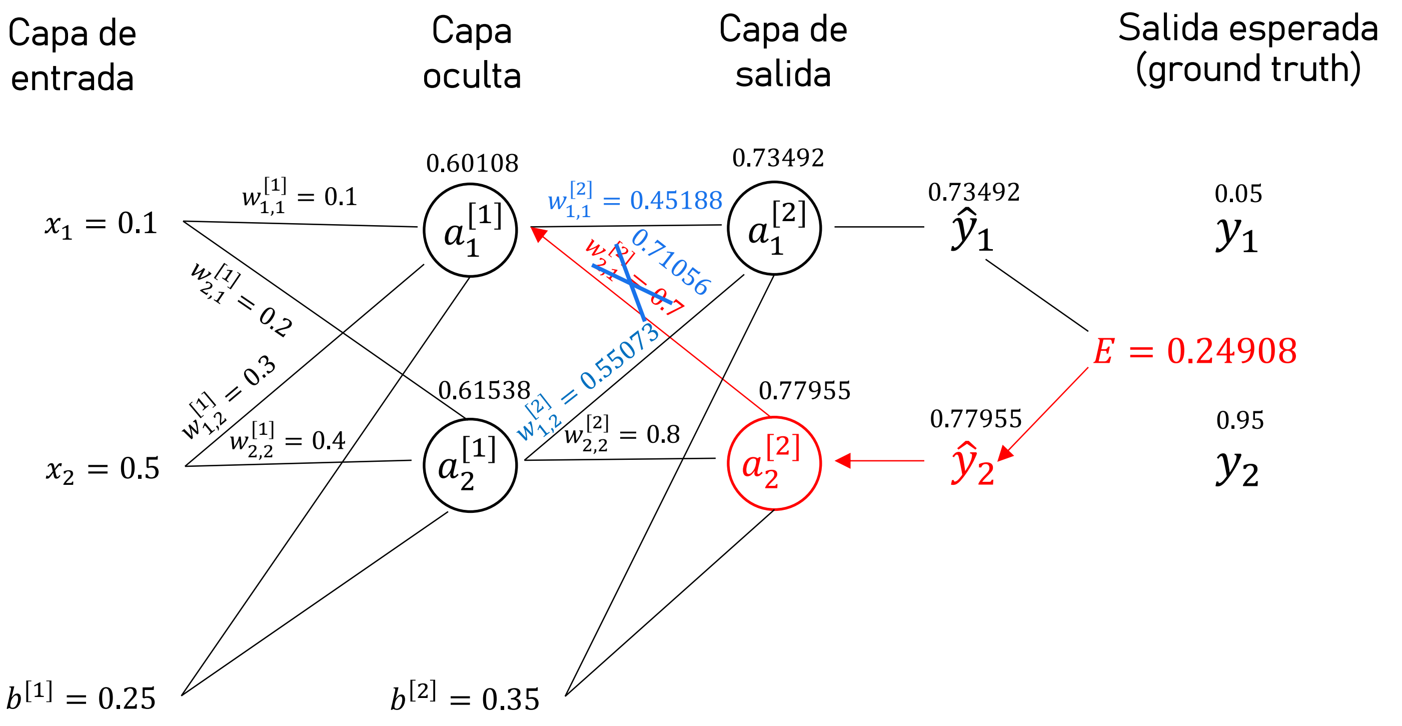
$\boldsymbol{\eta} = 0.6$

$$\boxed{w_{2,1}^{[2]}} = w_{2,1}^{[2]} - \boldsymbol{\eta}\left(\frac{dE}{dw_{2,1}^{[2]}}\right)$$

$$= 0.7 - 0.6(-0.01760)$$

$$= 0.7 + 0.01056$$

$$= \boxed{0.71056}$$



Perceptrones Multicapa y Retropropagación

Actualizar los pesos $w_{1,1}^{[1]}, w_{2,1}^{[1]}, w_{1,2}^{[1]}$ y $w_{2,2}^{[1]}$

Enviarlos por email a weam@turing.iimas.unam.mx