INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL

Laboratorio Inteligencia Artificial

¿Qué es la Inteligencia Artificial?

- La inteligencia artificial es una de las ramas de la ciencia que busca emular las capacidades del ser humano. Estas capacidades son las siguientes:
  - *Raciocinio (razonar, aprender, entender, tomar decisiones)*
  - *Comportamiento (movimiento)*
  - *Percepción*

Tipos de Inteligencia Artificial

- Sistemas que piensan como humanos
  - *Automatizan actividades como la toma de decisiones, la resolución de problemas y el aprendizaje. Un ejemplo son las redes neuronales artificiales.*
- Sistemas que actúan como humanos
  - *Se trata de computadoras que realizan tareas de forma similar a como lo hacen las personas. Es el caso de los robots.*
- Sistemas que piensan racionalmente
  - *Intentan emular el pensamiento lógico racional de los humanos, es decir, se investiga cómo lograr que las máquinas puedan percibir, razonar y actuar en consecuencia. Los sistemas expertos se engloban en este grupo.*
- Sistemas que actúan racionalmente
  - *idealmente, son aquellos que tratan de imitar de manera racional el comportamiento humano, como los agentes inteligentes.*

Ramas (tecnologías, disciplinas) de la Inteligencia Artificial

Ramas (tecnologías, disciplinas) de la Inteligencia Artificial

- Reconocimiento automático del habla (Speech recognition)
  - *El reconocimiento automático del habla es una disciplina perteneciente a la acústica y cuyo objetivo es el reconocimiento de fonemas en una señal de voz. Los sistemas de reconocimiento de voz procesan la señal recogida por un micrófono para identificar las palabras que el usuario pronunció.*
- Procesamiento del lenguaje natural NLP (PLN en español)
  - *Mientras que el reconocimiento del habla se centra en una conversión fiel de la voz a texto, el Procesamiento del lenguaje natural PLN (o NLP, Natural Language Processing, en inglés) es una disciplina que está más ligada al campo de la lingüística, y su objetivo es comprender qué intención tiene el usuario al lanzar un determinado comando, pregunta o afirmación (ya sea escrito o por voz) y qué espera obtener, así como analizar el estado anímico y encontrar patrones subjetivos en éstos. En resumen, es el campo que ayuda a la comunicación (principalmente sonora y escrita) hombre máquina, y viceversa.*

Ramas (tecnologías, disciplinas) de la Inteligencia Artificial

- Reconocimiento visual (Visual Recognition)
  - *El reconocimiento visual es la disciplina basada en el procesado de la señal de imagen o vídeo, con el objetivo de reconocer patrones, formas, y en el mejor de los casos, identificar fielmente los diferentes elementos en una imagen. Procesamiento del lenguaje natural NLP (PLN en español).*
- Reconocimiento de texto (Text Recognition)
  - *El reconocimiento de texto podría considerarse una parte del reconocimiento visual, ya que su principal objetivo es reconocer e identificar texto en formatos de imagen. Resulta común el uso de herramientas de OCR (Optical Character Recognition) para esta labor.*

Ramas (tecnologías, disciplinas) de la Inteligencia Artificial

- Big Data
  - *Big Data lo podemos considerar, sin entrar a tecnicismos, un gran volumen de datos. Big Data por sí solo no es una tecnología, pero tener a disposición una cantidad ingente de datos (preferiblemente estructurados) es una base vital en la consecución de objetivos tanto en la analítica de Business Intelligence como en la aplicación de determinados algoritmos de Machine Learning.Reconocimiento de texto (Text Recognition).*
- Sistemas expertos
  - *Los sistemas expertos son aquellos en los que se ha volcado todo el conocimiento humano posible acerca de una determinada rama. Un ejemplo clásico es el de los sistemas que juegan al ajedrez, que a partir de toda una colección de movimientos y estrategias que se les ha introducido en memoria, son capaces de determinar la mejor jugada (generalmente basándose en árboles de decisión) ante unas condiciones dadas.*

Ramas (tecnologías, disciplinas) de la Inteligencia Artificial

- Robótica
  - *La robótica (ya sea mecánica o robótica software, como el RPA) abarca un alto rango de dispositivos. Siempre que un sistema o robot muestre síntomas de inteligencia, por ejemplo, al ser capaz de tomar decisiones por muy básicas que sean éstas, estaremos hablando de Inteligencia Artificial. Recordemos que la IA no tiene por qué ser especialmente sofisticada, existe en todos los niveles, incluso los más básicos, y debe diferenciarse de la capacidad de aprender de las máquinas; es decir, el Machine Learning. Sistemas expertos.*
- Aprendizaje automático (Machine Learning)
  - *El aprendizaje automático o Machine Learning es la disciplina, dentro de la Inteligencia Artificial, que trata de conseguir que un sistema aprenda y relacione información del modo en que lo haría una persona. Para ello, usa algoritmos que son capaces de detectar patrones en los datos previos, pudiendo crear predicciones futuras, así como nuevas tendencias como el Deep Learning y sus algoritmos de redes neuronales.*

Ramas (tecnologías, disciplinas) de la Inteligencia Artificial

- Deep Learning
  - *El Deep Learning es una subdisciplina del Machine Learning. Es un sistema de aprendizaje que se inspira en el funcionamiento de las redes neuronales del cerebro humano para procesar la información, con una base matemática muy compleja detrás. Aunque sí se apoya en la experiencia (ya sean datos previos, generados por el entorno o autogenerados), no parte de indicaciones estrictas que determinen qué es correcto y qué no, de forma que el sistema pueda determinar conclusiones por sí sólo. Aprendizaje automático (Machine Learning).*
- Cognitive Intelligence & Cognitive Services
  - *La Inteligencia Cognitiva es una combinación de las tecnologías mencionadas previamente con el objetivo de crear servicios de Inteligencia Artificial capaces de tener comprensión humana. Es la unión de reconocimiento visual, sonoro, comprensión lectora, NLP y Machine Learning para crear sistemas capaces de comprender la información relacionada a la interacción humana y responder en consecuencia.*

CLASSROOM

- https://classroom.google.com/c/MjExNTAyNDMxODU4?cjc=l762yit
- Código: l762yit
- Llenar formulario de datos (solo los que llevan laboratorio)
- NOTA: este será el medio de comunicación por el momento, el oficial debería de ser UEDI

Datos de Contacto

- Correo: compi1.2019@gmail.com
- Asunto: [IA1] Asunto

Restricciones

- El lenguaje de programación para cualquier actividad de laboratorio será PYTHON 3.
- Todas las actividades serán realizadas de manera individual.

Entrega de Prácticas y Proyectos

- Crear un repositorio para su proyecto.
- En la calificación se clonará el repositorio para calificar de lo que tengan en él.
- Deben de darme acceso en todo momento a su repositorio, de lo contrario se invalidará la nota de su práctica o proyecto.
- En entregable será el enlace de su repositorio.

ALGORITMOS GENÉTICOS

Laboratorio Inteligencia Artificial

Algoritmos Genéticos

- Los Algoritmos Genéticos son métodos adaptativos que son utilizados para resolver problemas de búsqueda y optimización. Estos algoritmos están basados en el proceso genético de los seres vivos.
- Los Algoritmos Genéticos son capaces de ir creando soluciones para problemas del mundo real.
- Se busca que las soluciones dadas por el algoritmo evolucionen hacia valores óptimos del problema atacado.

Elementos dentro de un Algoritmo Genético

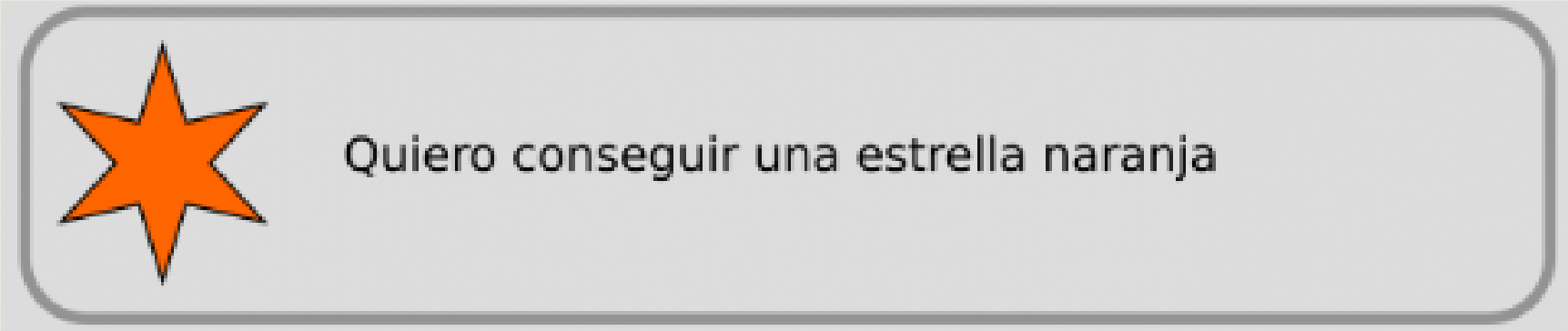
- Individuo: es una posible solución al problema. Estos están conformados por un conjunto de parámetros a los cuales vamos a llamar genes. Estos individuos van a estar representados como un arreglo dentro de nuestro código (un arreglo de genes).
- Población: es un conjunto de soluciones (individuos).

¿Cuál es el principal objetivo de los algoritmos genéticos?

OPTIMIZAR

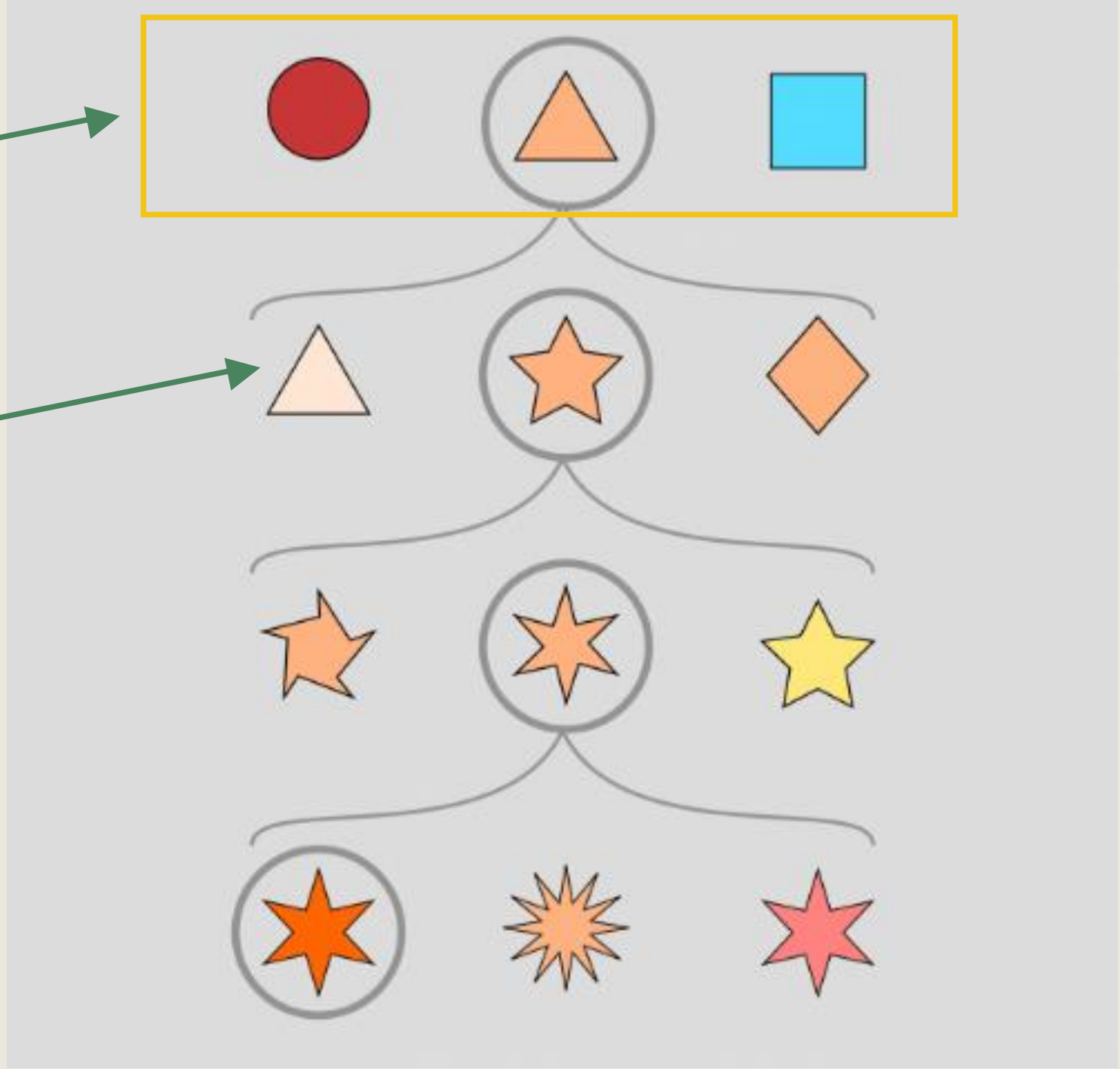
EJEMPLO

- Problema



Población

Individuo



Generación 1

Generación 2

Generación 3

Generación 4

Secuencia de un algoritmo genético

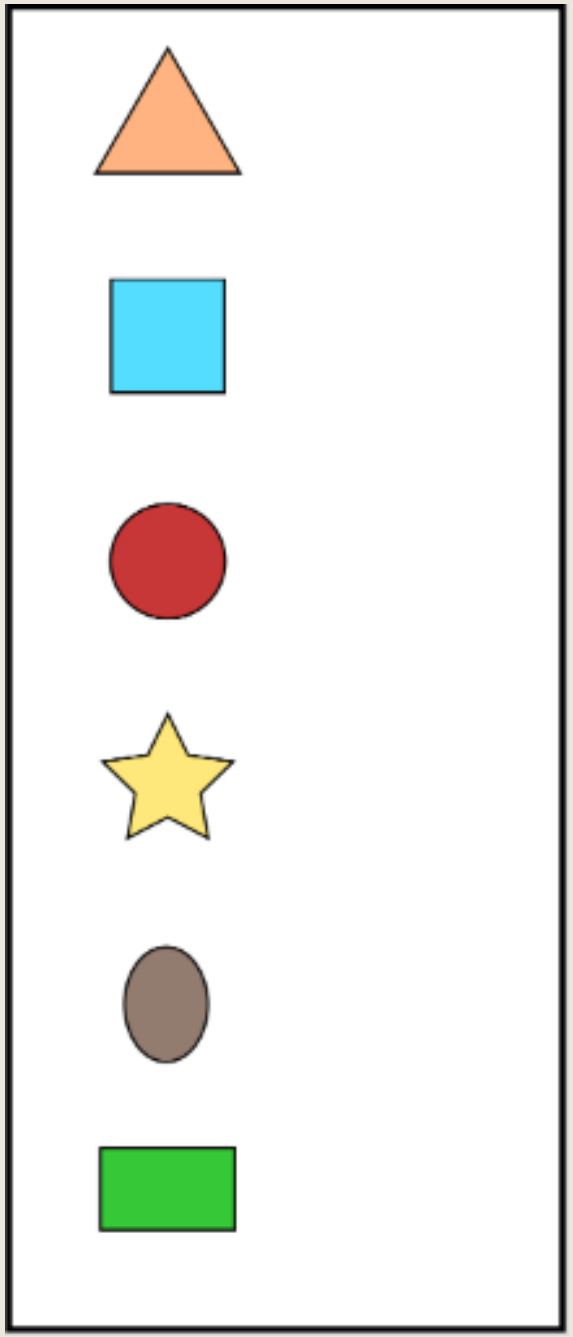
- Básicamente es un proceso iterativo que se realiza sobre una población de individuos. Se puede resumir en los siguientes pasos:

- Inicialización de la población
- Evaluación de un individuo
- Selección de padres (se toman solo ciertos padres de una generación)
- Emparejamiento (cómo se van a emparejar los padres para crear nuevos hijos)
- Cruzar (cómo se van a conformar los nuevos hijos, qué elementos de cada padre se van a utilizar)
- Mutación (qué alteraciones va a tener un nuevo hijo)

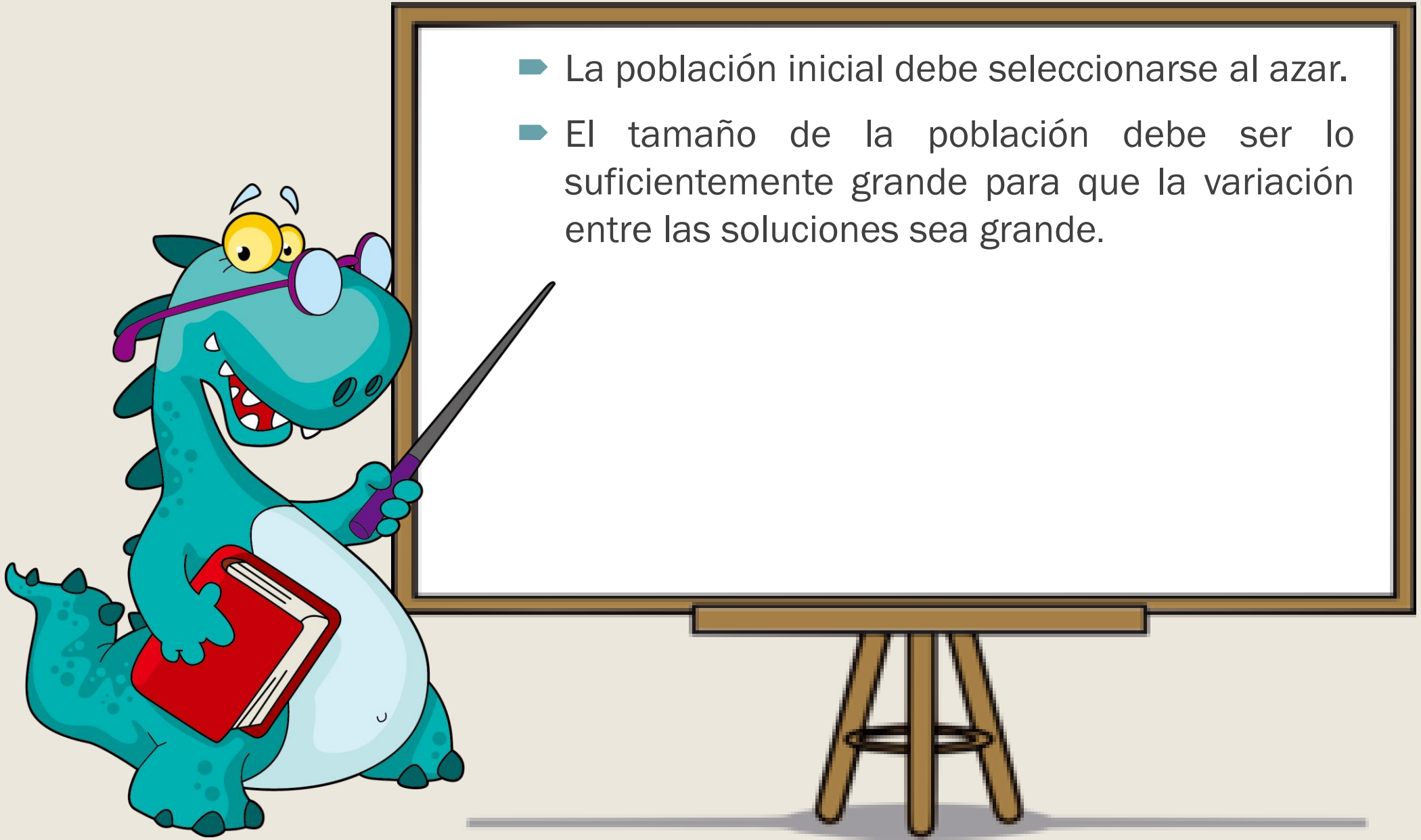
Inicialización

- Se crea la población inicial con individuos al azar.

P0 = [S1, S2,... SN]



Consejos

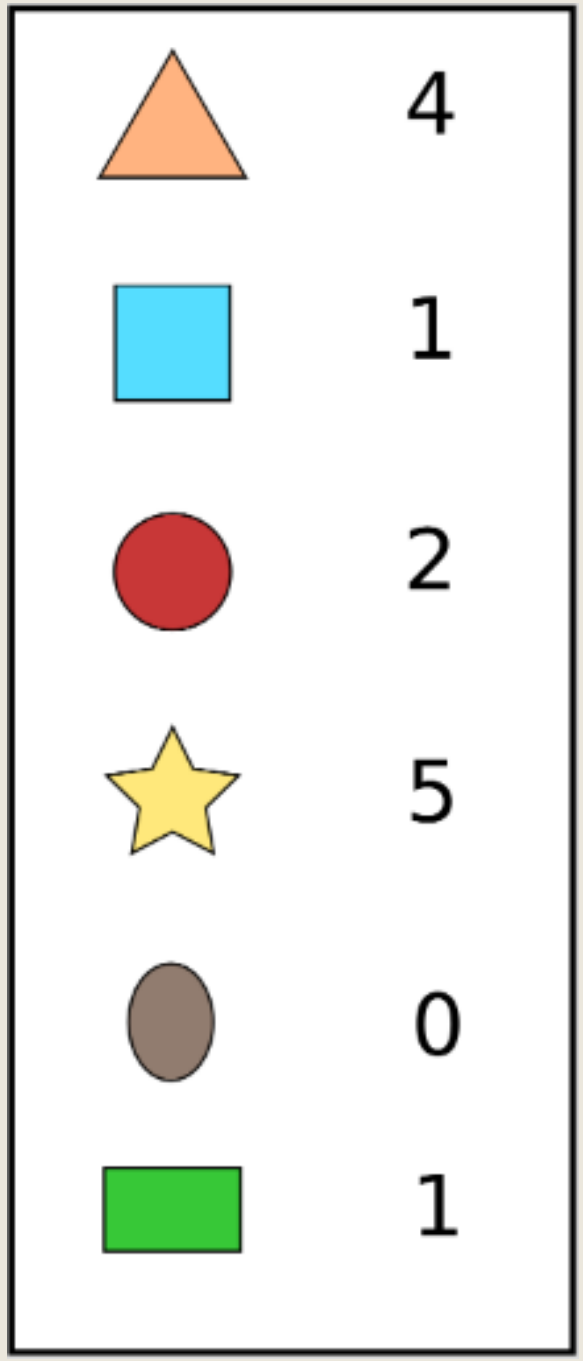


- La población inicial debe seleccionarse al azar.
- El tamaño de la población debe ser lo suficientemente grande para que la variación entre las soluciones sea grande.

Evaluación

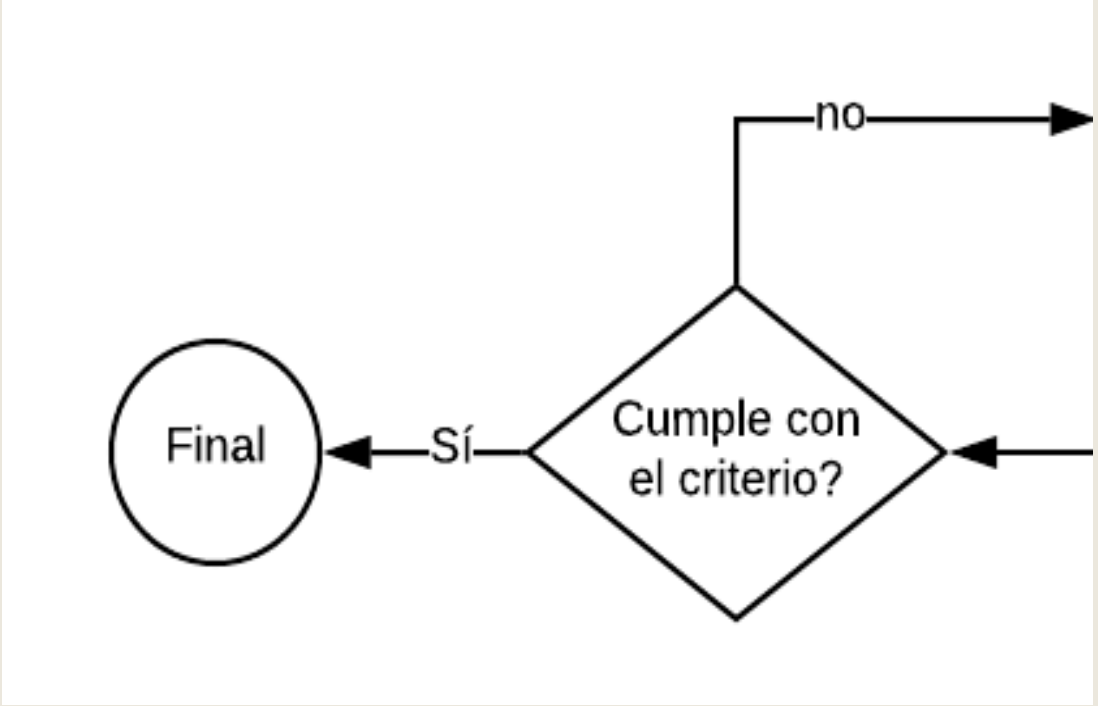
- Se evalúa la calidad de todos los individuos de la población, se les asigna un valor numérico llamado *fitness*.

```
E0 = [ ]
For every S in P0:
    temp = fitness(S)
    E0.push(temp)
```



Fin del algoritmo

- Después de la evaluación, si alguna de las soluciones cumple el criterio de aceptación el algoritmo termina y la mejor solución de la población pasa a ser la solución del algoritmo.

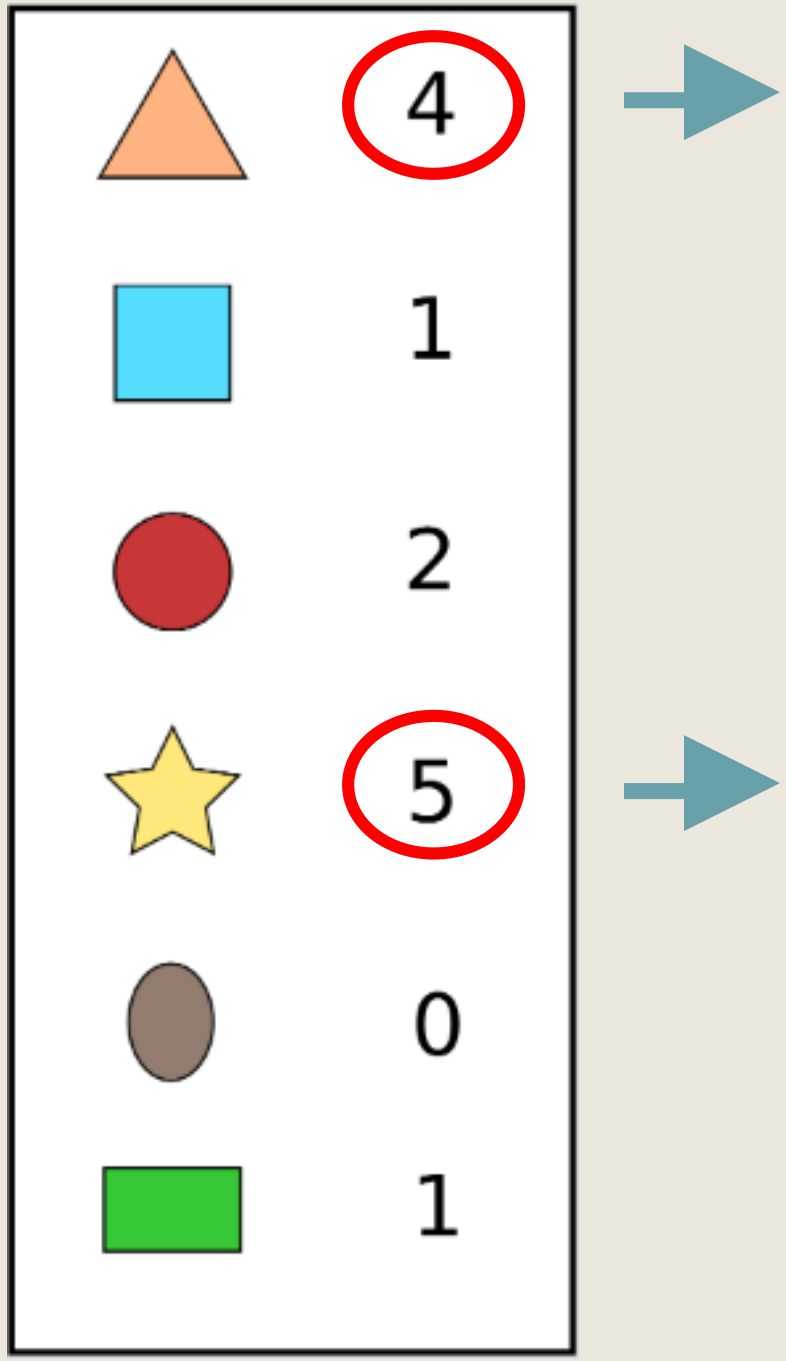


Selección

- Se eligen a los individuos con el valor **fitness** mas alto.

NP = [ ]

NP = max(P0, E0)



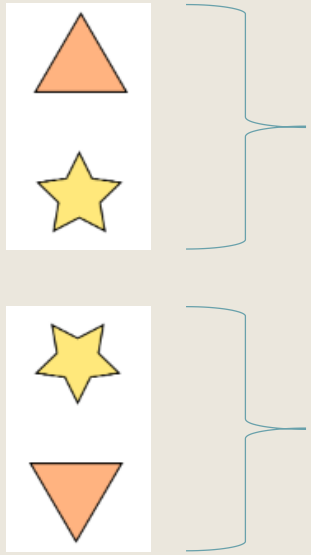
Emparejamiento

- Se emparejan los individuos seleccionados para la operación de reproducción (*Cruzar*).

NP = [S0, S1, S2,... SN]

Pair1 = random(NP,2)

Pair2 = random(NP,2)



Cruzar

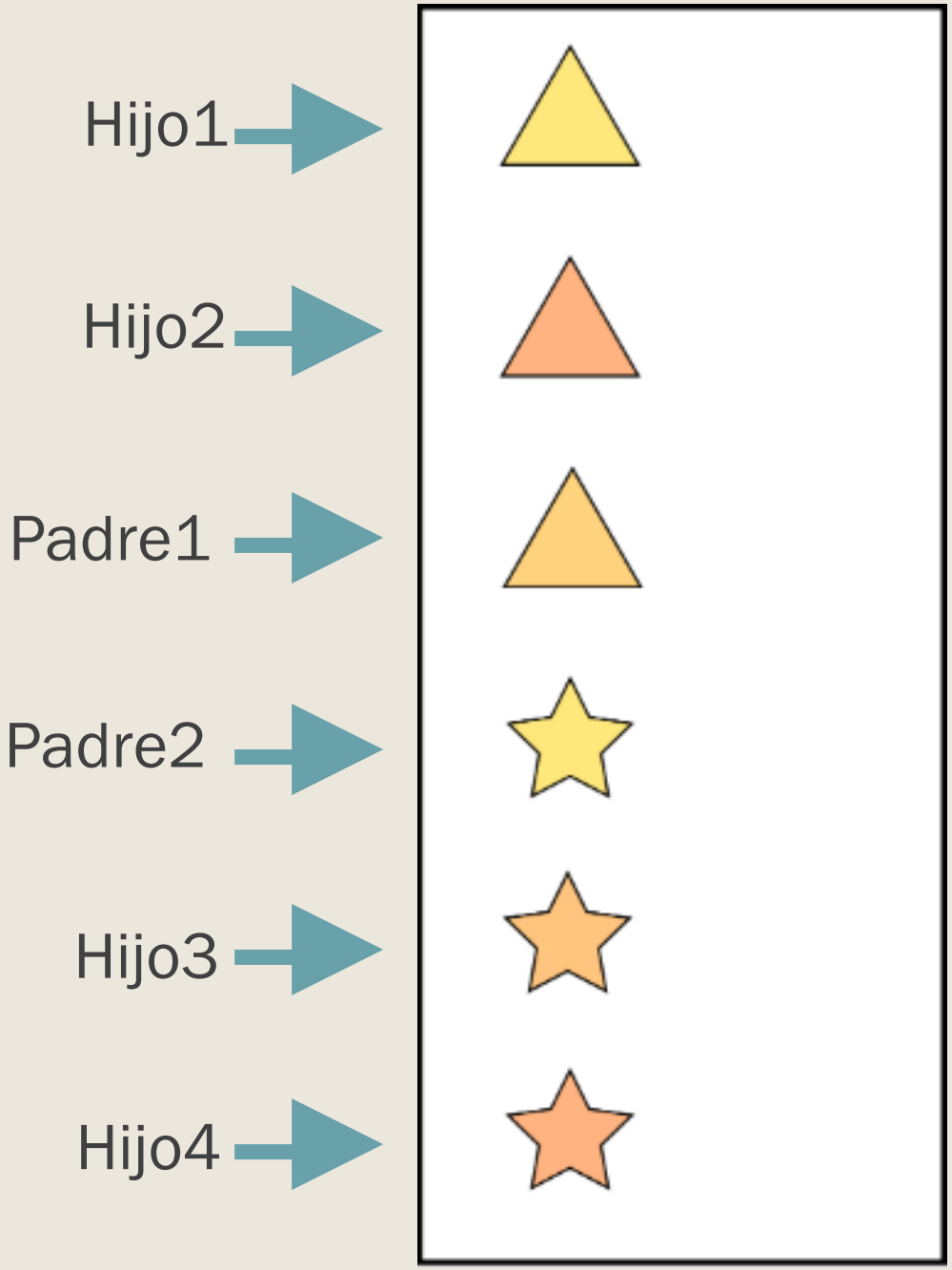
- Se mezclan las características de los padres seleccionados para generar hijos que sustituyan a los padres que no fueron seleccionados.

Hijo1 = merge(Pair1)

Hijo2 = merge(Pair2)

Hijo3 = merge(Pair3)

Hijo4 = merge(Pair4)



Mutación

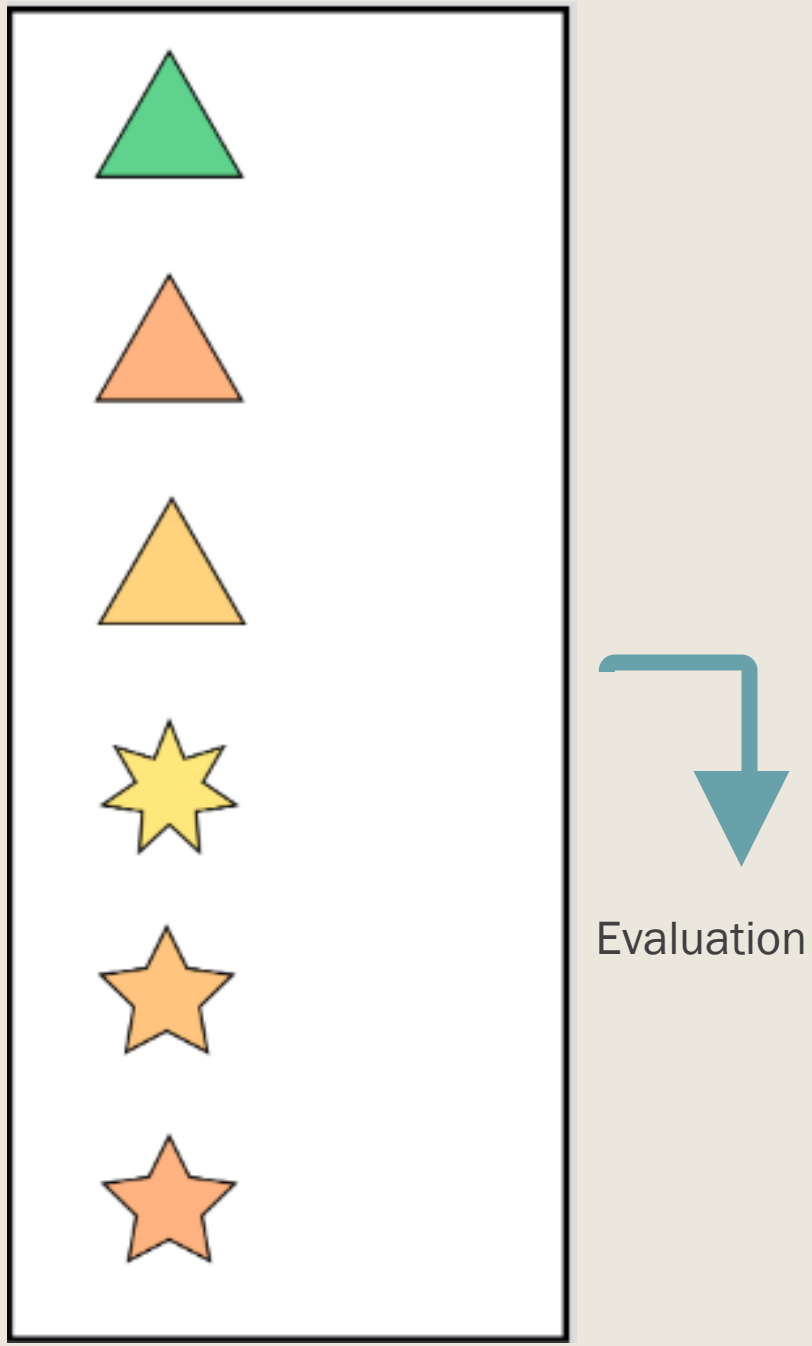
- Se agregan algunas variaciones en los hijos al azar.

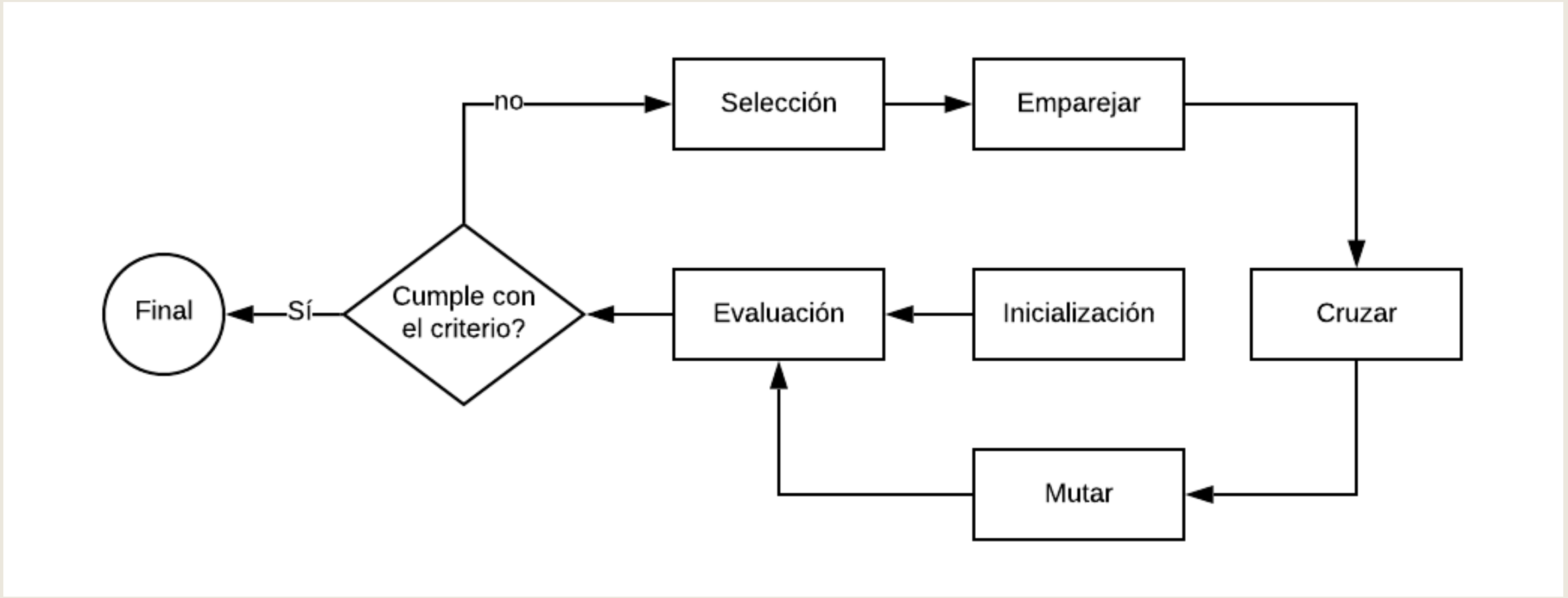
NP.add(mutation(Hijo1))

NP.add(mutation(Hijo2))

NP.add(mutation(Hijo3))

NP.add(mutation(Hijo4))




Selección
Emparejar
Cruzar
Mutar
Inicialización
Evaluación
Cumple con el criterio?
no
Sí
Final

Tipos de selección

- **Selección proporcional:** Cuanto mas alto sea el fitness de un individuo mas alta será la probabilidad que pase al a siguiente generación.

$$m_s = \frac{m_o - min(m_o)}{max(m_o) - min(m_o)}$$

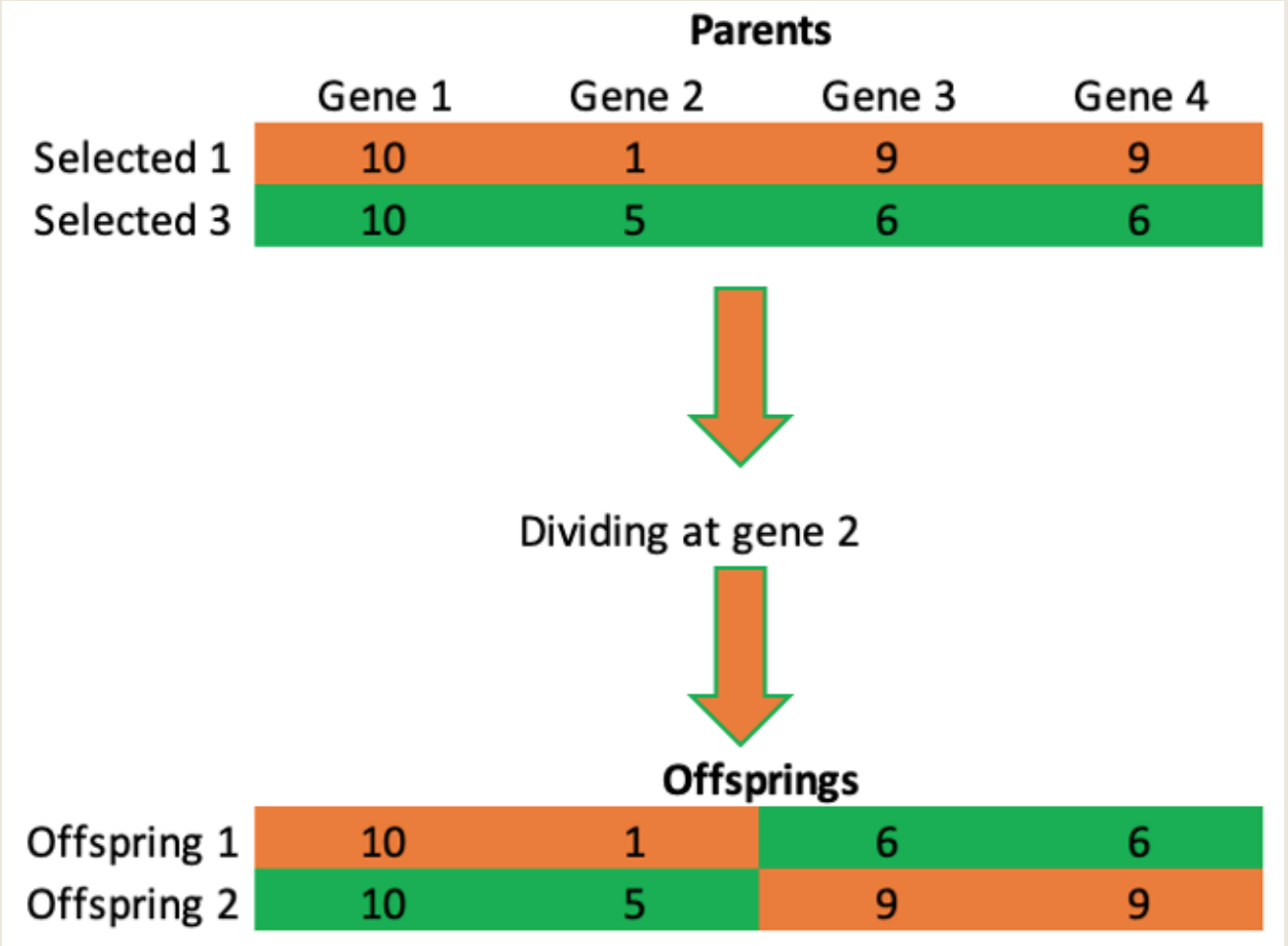
- **Selección por torneo:** Se eligen individuos al azar y el fitness mas alto del subgrupo es el que pasara a la siguiente generación.
- **Selección de la mejor mita:** La mitad de los mejores individuos pasan a la siguiente generación.
- **Selección al azar:** Los individuos son elegidos de forma aleatoria, con igual probabilidad para todos.

Tipos de emparejamiento

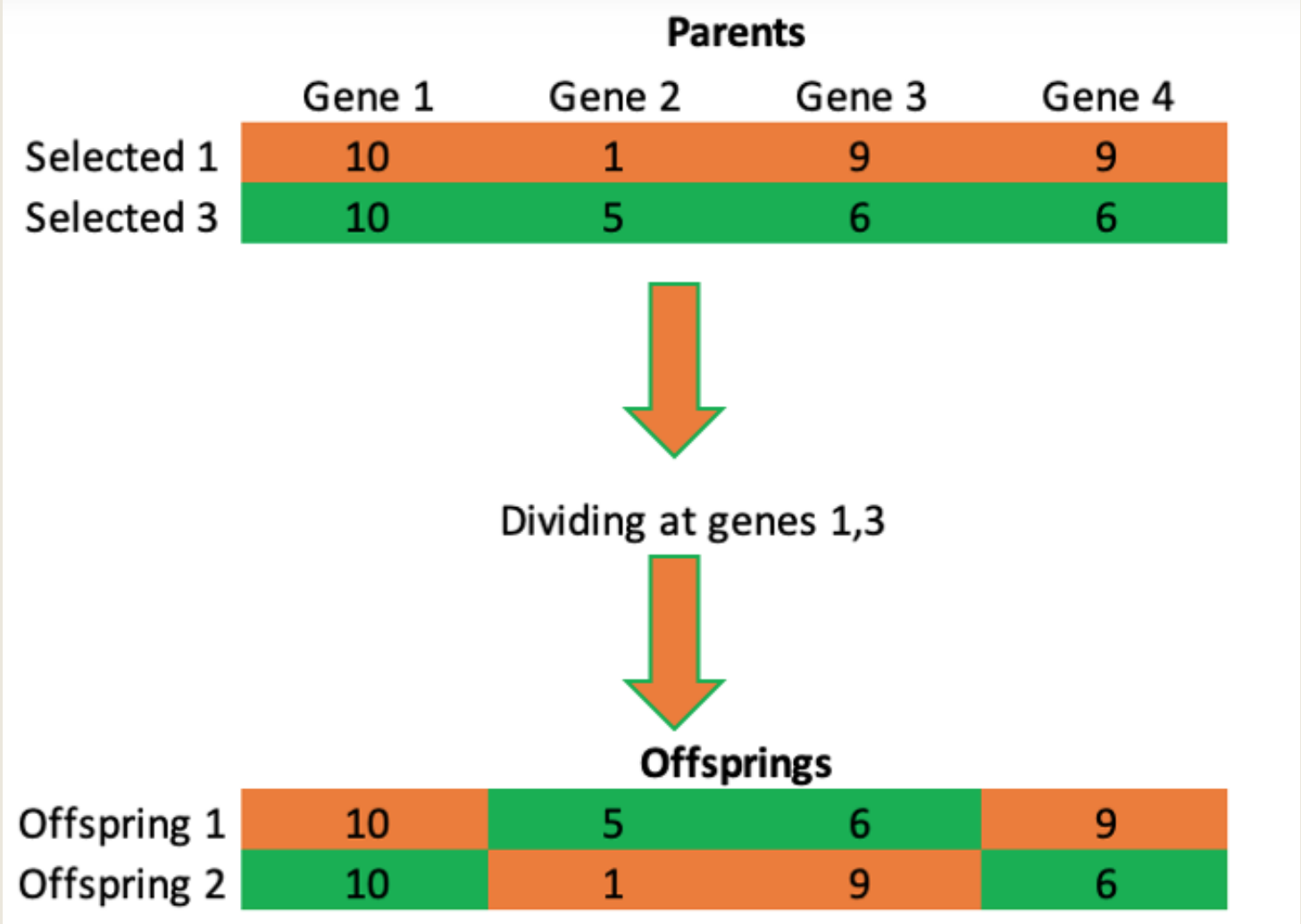
- **Mas apto:** Los mejores se emparejan con los mejores y los peores con los peores.
- **Aleatorio:** Las parejas se eligen de forma aleatoria.

Tipos de cruzamiento

- Unico Punto

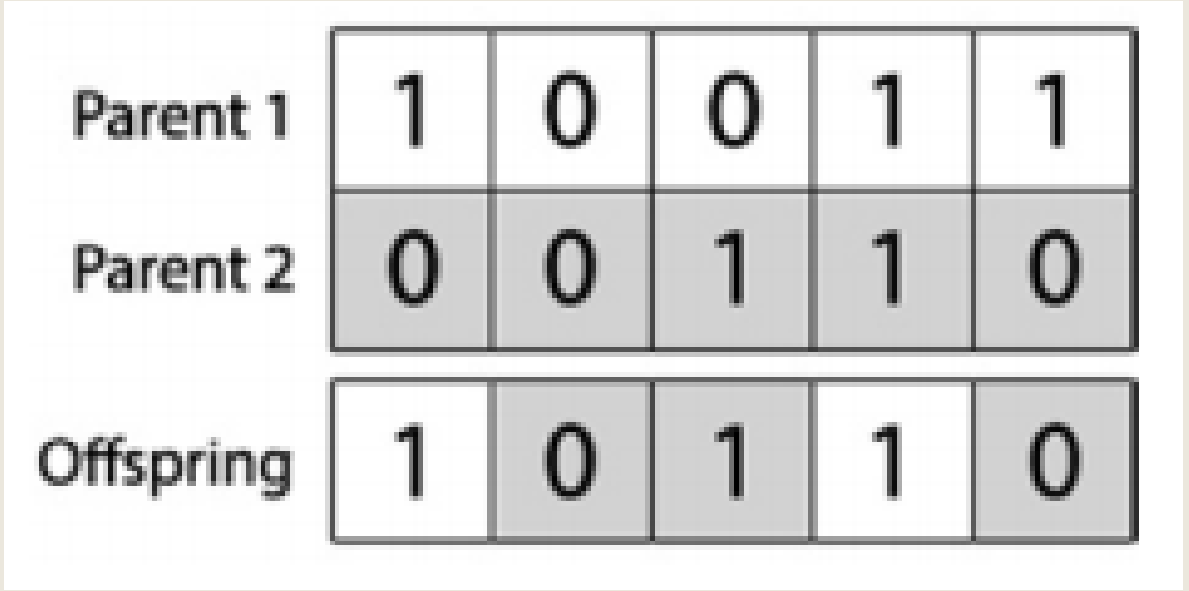


- Dos puntos


Parents
Gene 1
Gene 2
Gene 3
Gene 4
Selected 1
10
1
9
9
Selected 3
10
5
6
6
Dividing at genes 1,3
Offsprings
Offspring 1
10
5
6
9
Offspring 2
10
1
9
6

- Uniforme

Cada gen tiene la probabilidad del 50% de ser de un padre u otro.



Parent 1	1	0	0	1	1
Parent 2	0	0	1	1	0
Offspring	1	0	1	1	0

Criterio de finalización

- Máximo fitness
- Máximo fitness promedio
- Numero máximo de generaciones
- Máximo números similares de fitness

Nota: cuidado al elegir su criterio de finalización, puede que el algoritmo nunca termine.

Algoritmo General

```
generacion = 0
poblacion = inicializarPoblacion()
fin = verificarCriterio(poblacion, generacion)

while(fin == None):
    padres = seleccionarPadres(poblacion)
    poblacion = emparejar(padres)
    generacion += 1
    fin = verificarCriterio(poblacion, generacion)
```

MACHINE LEARNING

Laboratorio Inteligencia Artificial

Machine Learning

- Machine Learning o Aprendizaje Automático es una disciplina del campo de la Inteligencia Artificial que, a través de algoritmos, provee a las computadoras la capacidad de **identificar patrones** en datos masivos con el objetivo de hacer predicciones.
- Se generan sistemas que aprenden por sí solos.
- Tiene la capacidad de predecir situaciones que todavía no suceden basándose en datos que se la han dado previamente.

Aplicaciones del Machine Learning

- Detección de rostro
- Anti spam
- Anti virus
- Comprensión de textos
- Vehículos autónomos y robots
- Análisis de imágenes de alta calidad

Aprendizaje

- Los algoritmos de Machine Learning aprenden del mismo modo como aprendemos los humanos.

Niño aprendiendo

Máquina aprendiendo

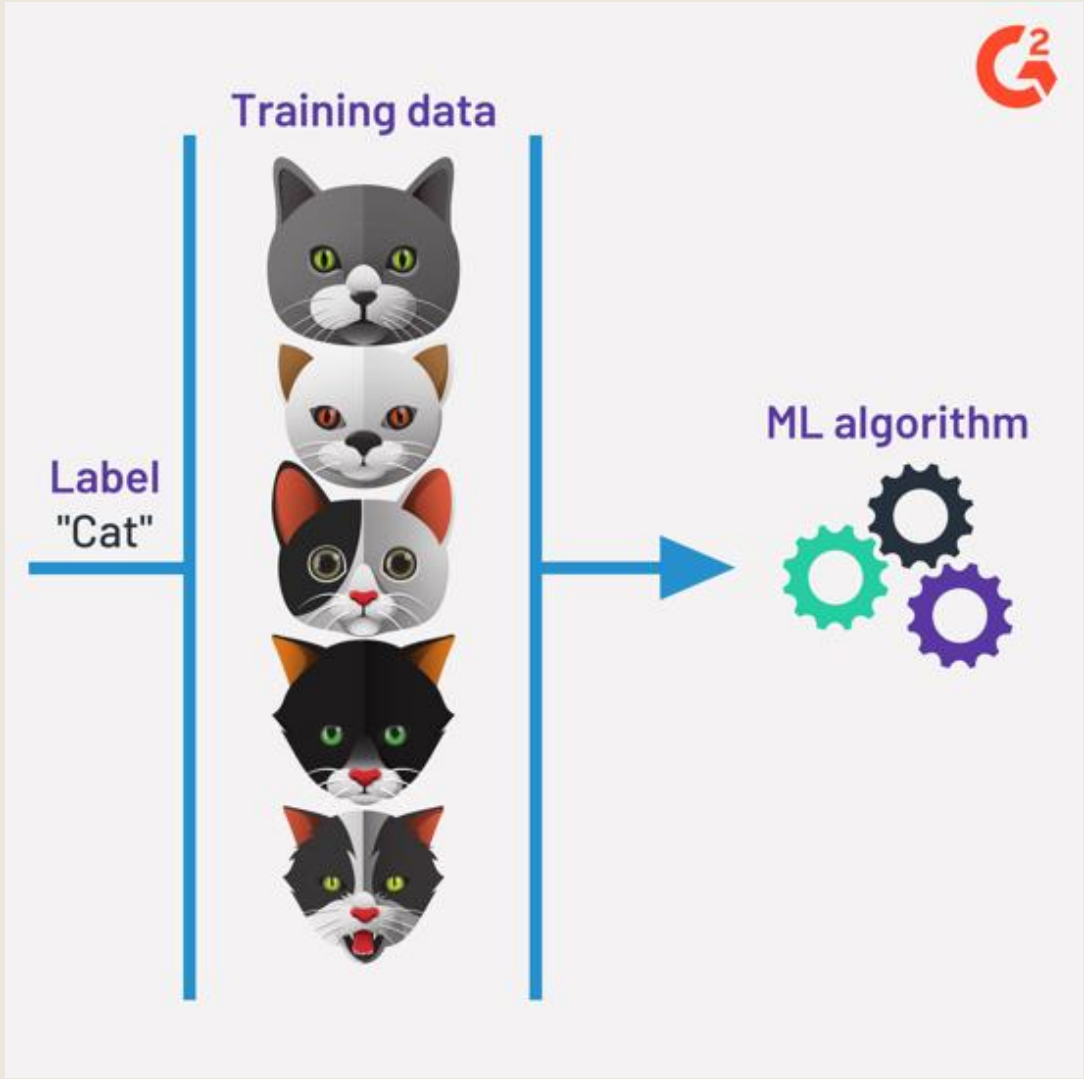
TIPOS DE MACHINE LEARNING

Aprendizaje Supervisado

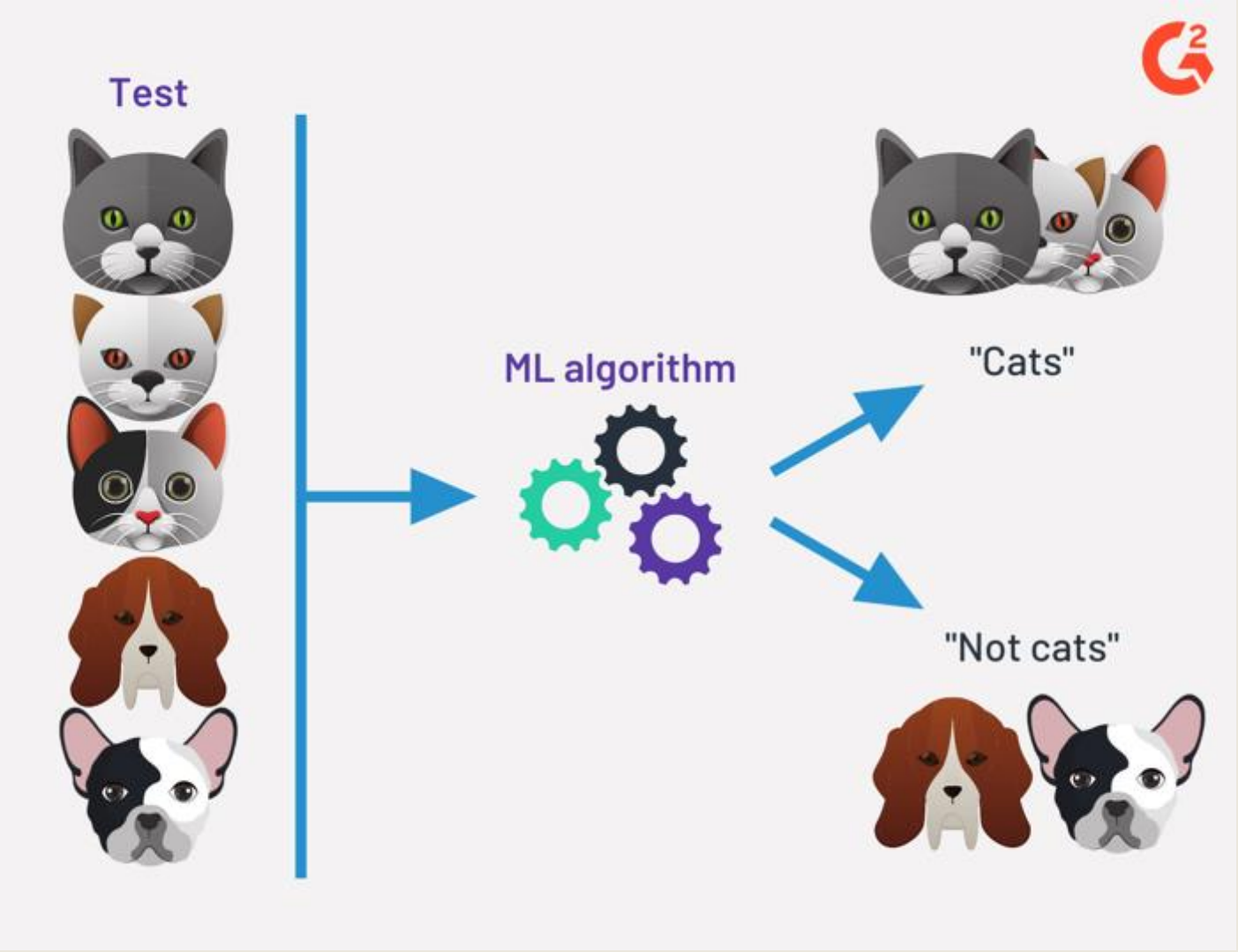
- El aprendizaje supervisado necesita datos previamente etiquetados para aprender a realizar su trabajo.
- En otras palabras el aprendizaje supervisado se caracteriza por contener las respuestas correctas al problema que se está resolviendo. Es decir, tenemos una gran cantidad de datos con las preguntas y las respuestas que esperamos encontrar.

Entrada	Salida
X1	Y1
X2	Y2
X3	Y3
X4	Y4

¿Cómo funciona el aprendizaje supervisado?



¿Cómo funciona el aprendizaje supervisado?

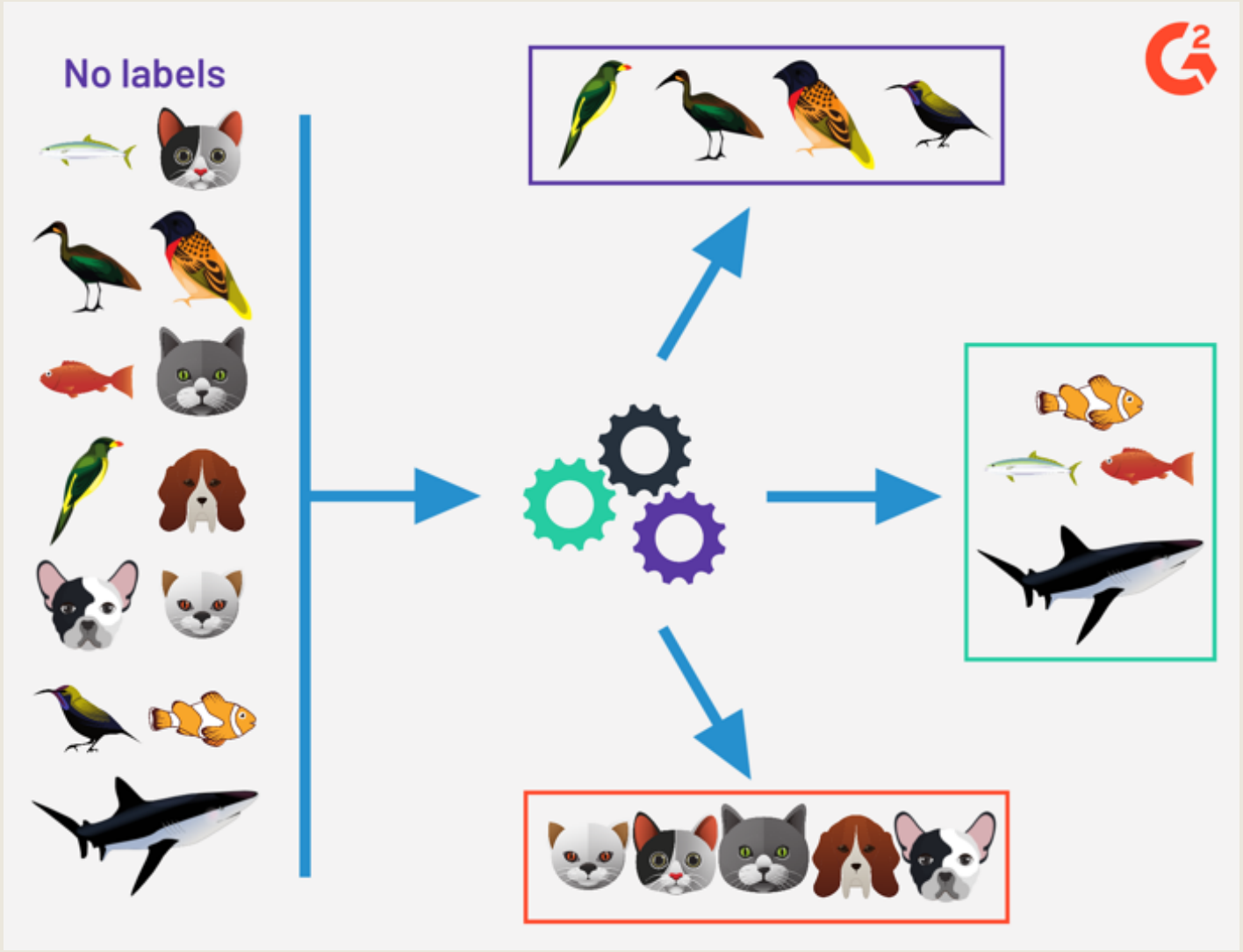


Aprendizaje No Supervisado

- El aprendizaje no supervisado no cuenta con un conocimiento previo.
- Se encuentra con un conjunto de datos con el objetivo de encontrar patrones que permitan organizarlos de alguna manera.
- En otras palabras, no se conocen las respuestas correctas. Es decir, nuestro conjunto de datos únicamente contienen las entradas pero no las salidas de lo que buscamos resolver.

Entrada
X1
X2
X3
X4

¿Cómo funciona el aprendizaje no supervisado?

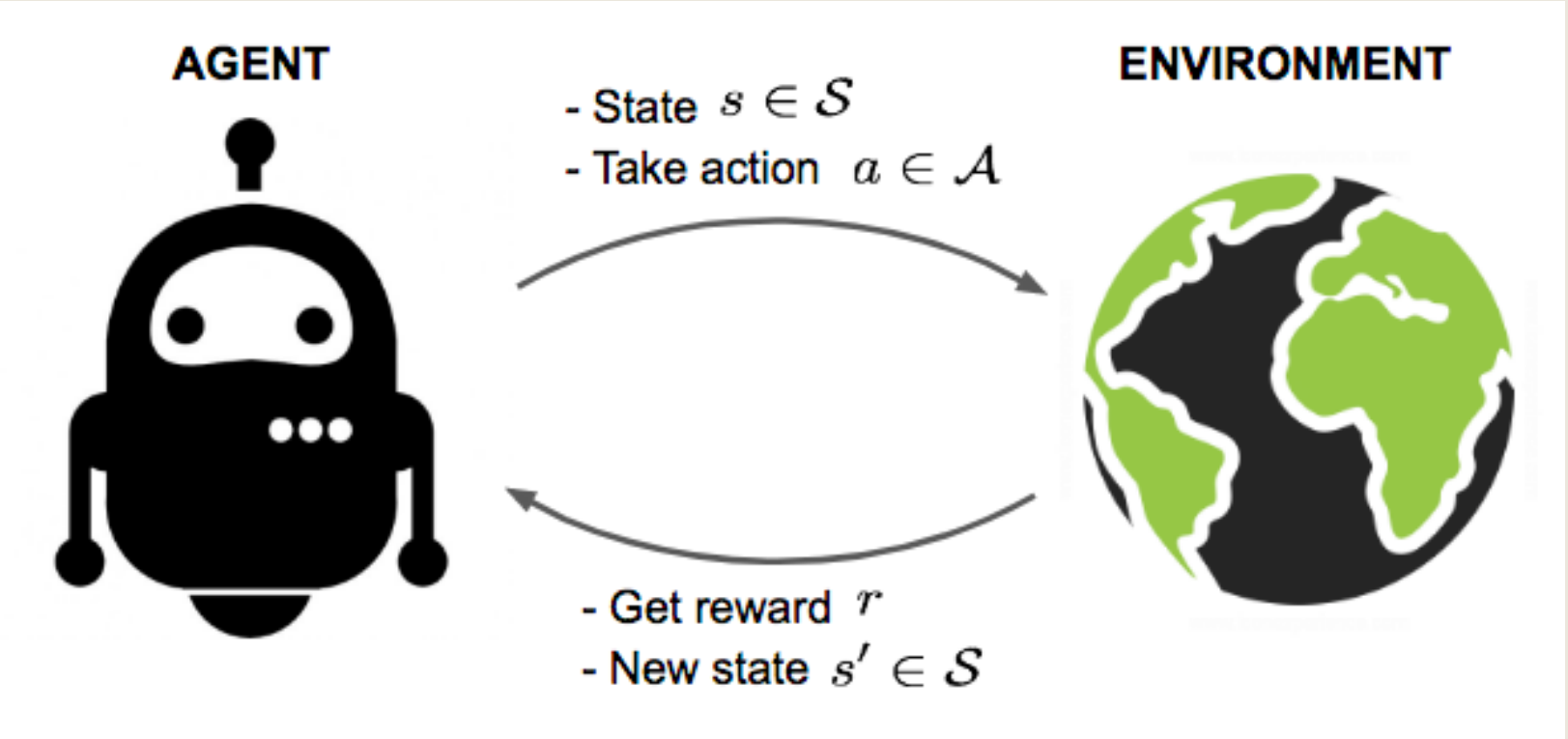


Aprendizaje Por Refuerzo

- El aprendizaje por refuerzo busca tomar la mejor decisión (dar la mejor respuesta) por medio de un proceso de retroalimentación. En otras palabras, no se tienen las respuestas correctas pero se le puede decir al algoritmo que tan buena es su respuesta. El sistema aprende a base de prueba y error.

Entrada	Recompensa
X1	R1
X2	R2
X3	R3
X4	R4

¿Cómo funciona el aprendizaje por refuerzo?



ANÁLISIS DE REGRESIÓN

Análisis de Regresión

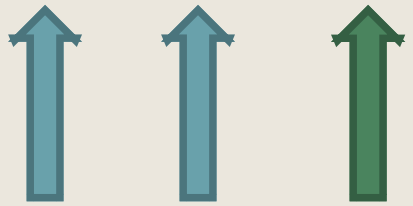
- El análisis de Regresión es un subcampo de Machine Learning Supervisado. Su propósito es establecer un modelo que relacione un cierto número de características con una variable objetivo.
- En los problemas de regresión perseguimos obtener una respuesta cuantitativa, como por ejemplo, predicciones sobre precios de inmuebles o el número de segundos que alguien dedicará a visualizar un vídeo.

Regresión Lineal

- Es considerado el algoritmo de Machine Learning mas sencillo. Su objetivo es encontrar la relación entre **una variable independiente y una dependiente** (regresión lineal simple), o la relación entre **un conjunto de variables independientes y una dependiente** (regresión lineal multi variable).

X	Y
X0	Y1
X1	Y2
x2	Y3

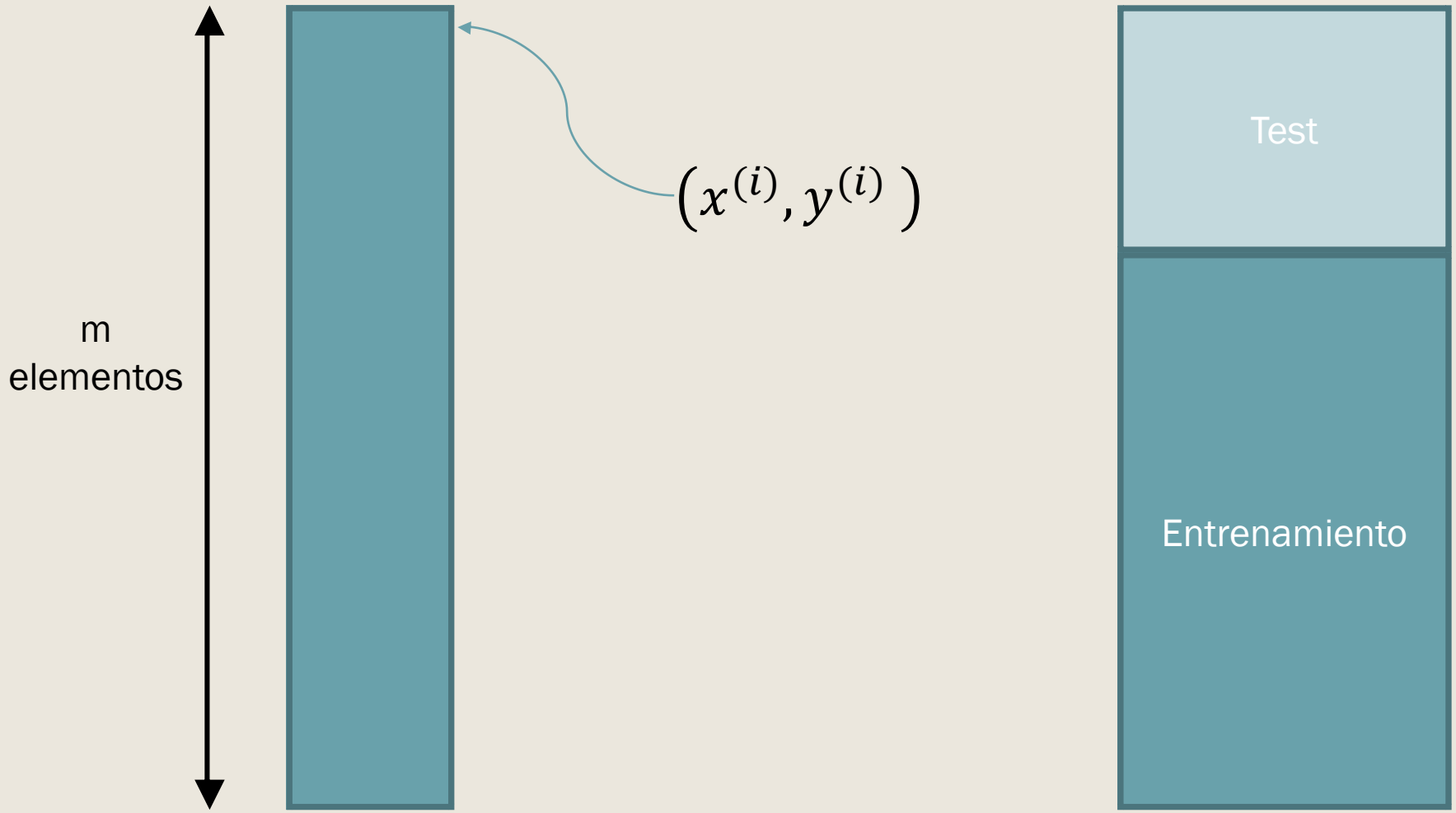
$$Y = \beta_0 + \beta_1 X + \varepsilon$$



Coeficientes

Error

Datos del Modelo



Función de Costo

- La función de costo va a indicar qué tan buenos son los coeficientes actuales del modelo $(\beta_0, \beta_1, \ldots, \beta_n)$. Mide la diferencia entre la salida conocida y la salida generada por el modelo.

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

$$\hat{y}$$

$$Y = \hat{y} + \varepsilon$$

Error

Error

Valor real

Valor calculado

$$\varepsilon = Y - \hat{y}$$

Valor calculado

Error

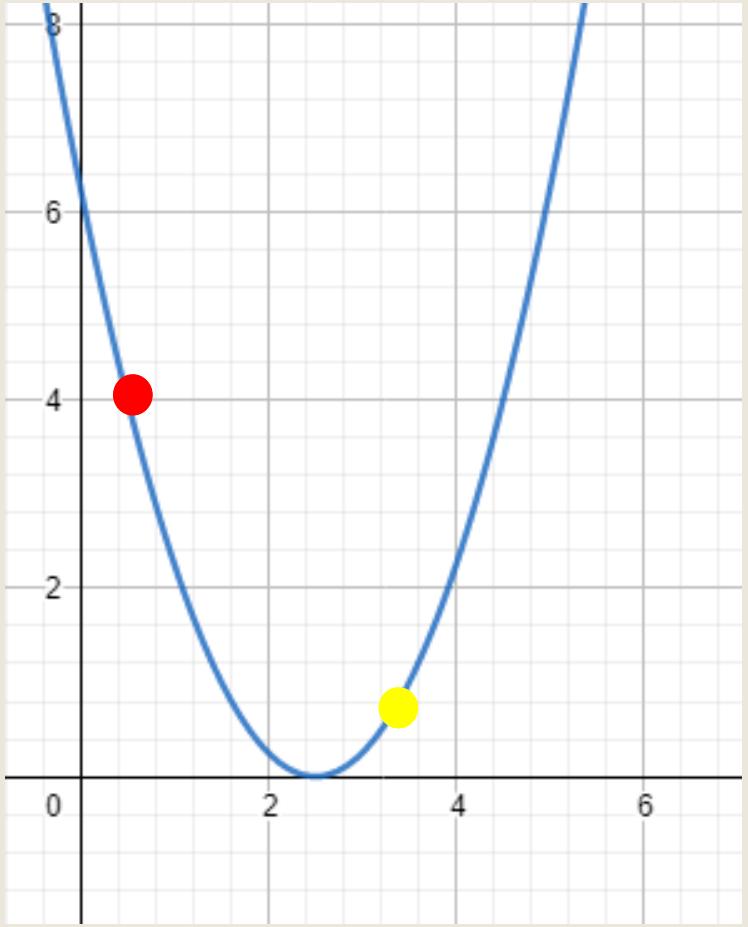
Valor real

Función de Costo

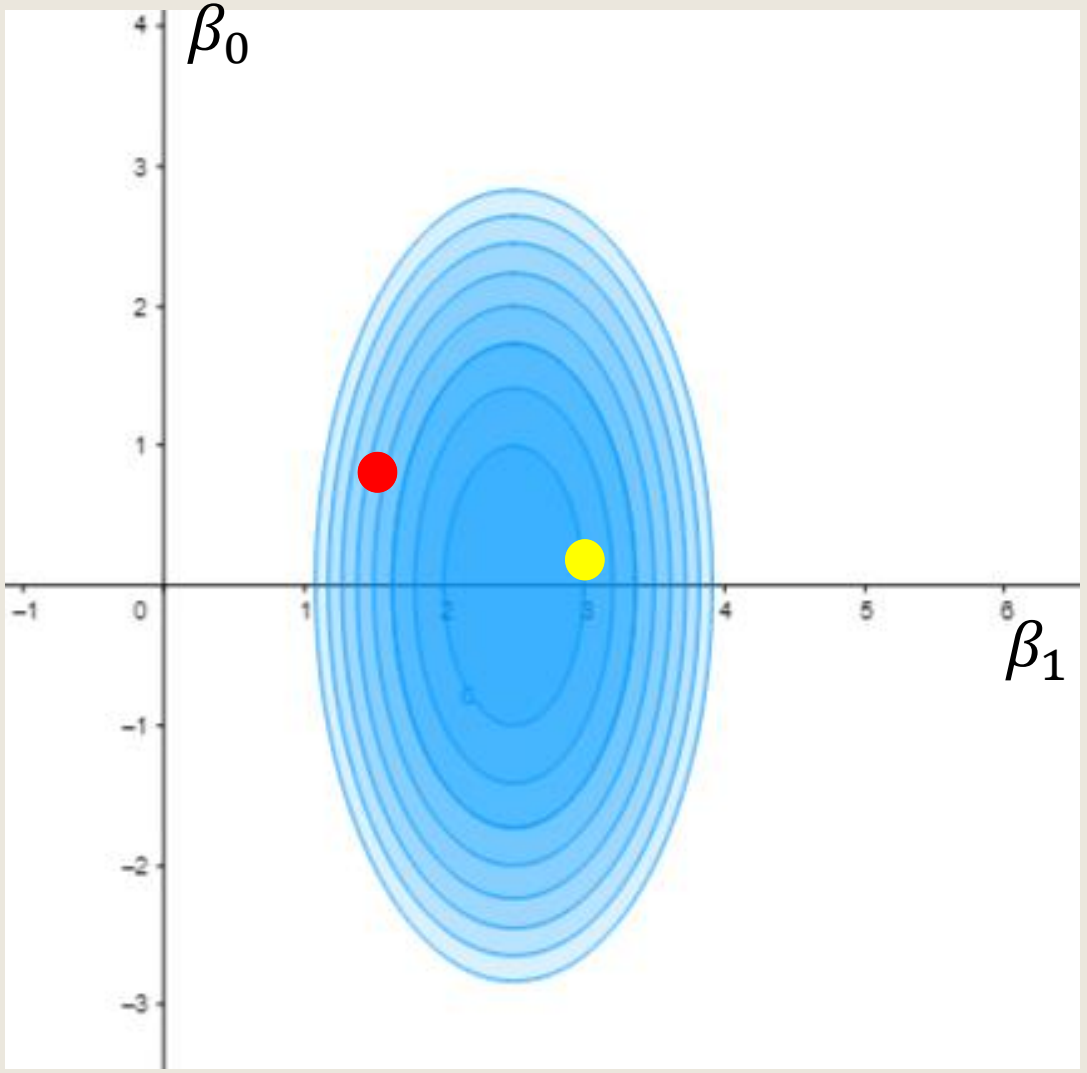
- Por lo tanto la función de costo (denotada como J) seria la siguiente:

$$\mathrm{J}(\beta_0, \beta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(Y^{(i)} - \hat{y}^{(i)}\right)^2 = \frac{1}{2m}\sum_{i=1}^{m}\left(Y^{(i)} - (\beta_0 + \beta_1 X^{(i)})\right)^2$$

J



β_1



Gradiente

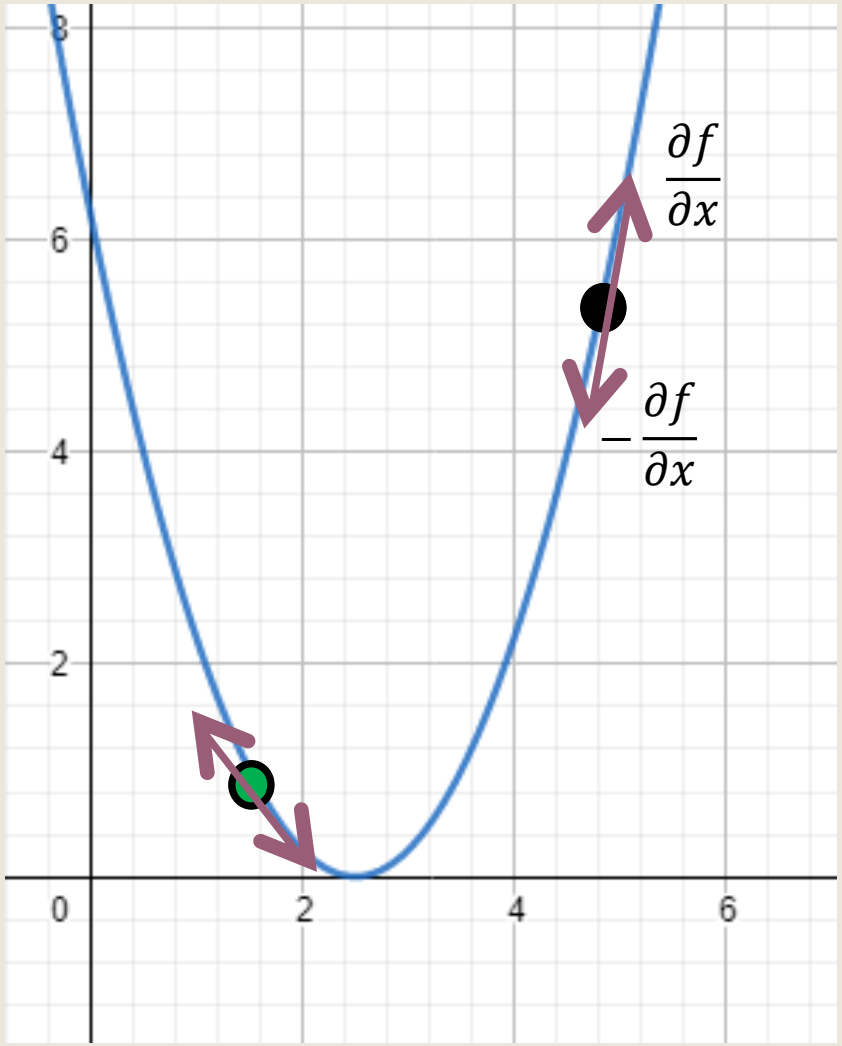
- El gradiente de una función es un vector cuyas componentes son las derivadas parciales de la función, su magnitud y dirección denotan la dirección en la que se produce el mayor cambio posible en la función (para dónde crece).

$$f(x, y, z)$$

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}\right]$$

Gradiente

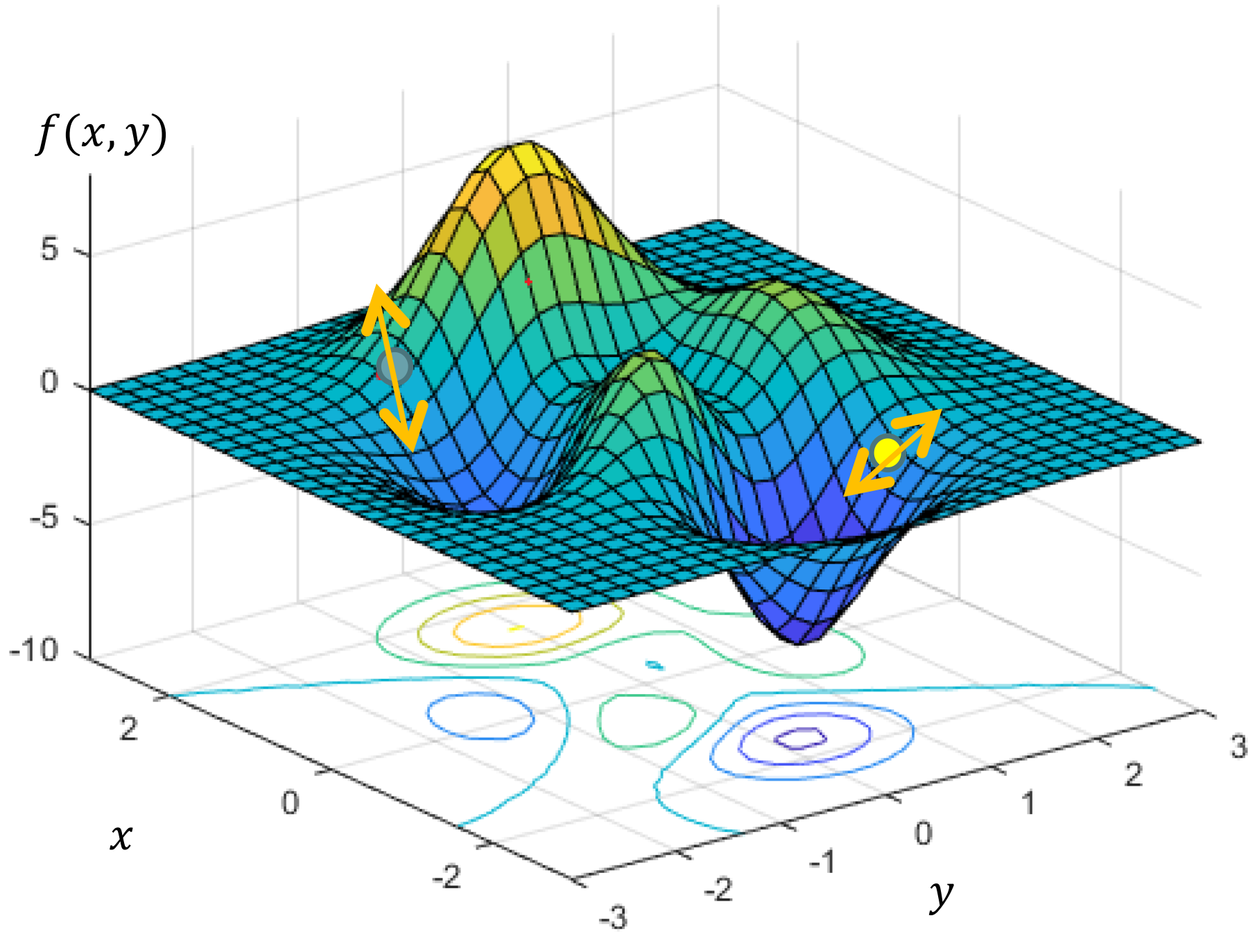
$f(x)$



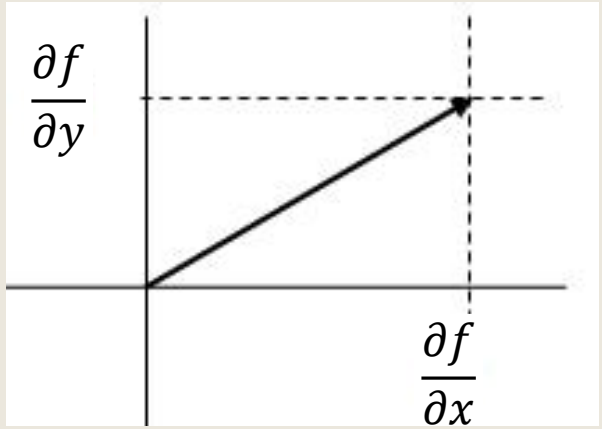
x

$$\nabla f = \frac{\partial f}{\partial x}$$

- En este caso como la función tiene únicamente una variable independiente, el gradiente solamente posee una componente.
- Dependiendo la dirección que tenga el gradiente indicara el máximo crecimiento o el máximo decrecimiento.



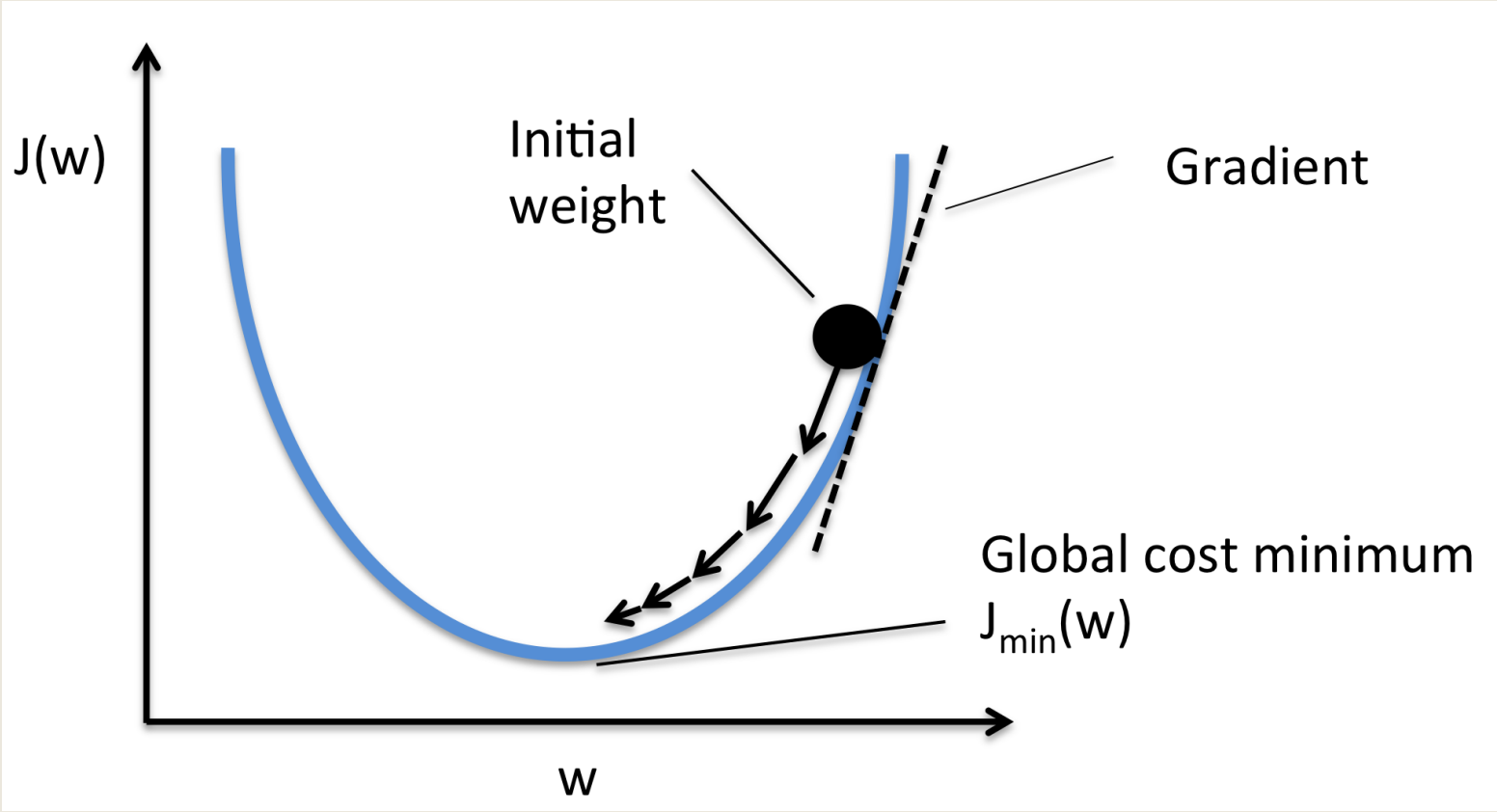
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$



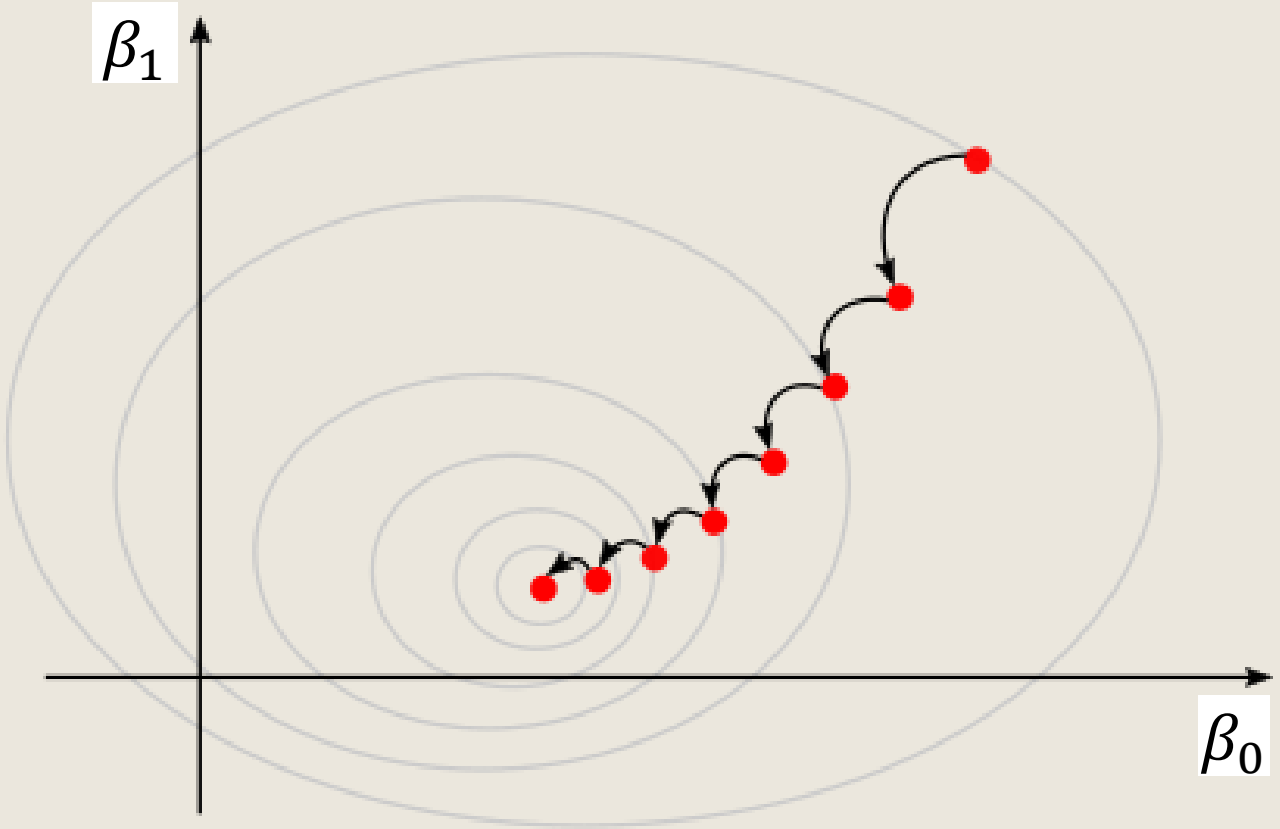
Descenso por Gradiente

- Para minimizar el error, se utilizará el método de descenso en gradiente, el cual calcula las diferentes direcciones en las que la recta se puede desplazar para reducir el error, y actúa de forma que reduce el error al mínimo.
- Consiste en obtener sucesivamente el gradiente de una función hasta encontrar el mínimo de esta.
- Se finaliza hasta que se cumple una condición establecida.
- Se buscan los valores de los coeficientes $(\beta_0, \beta_1, \ldots, \beta_n)$ que minimizan la función.
- El descenso por gradiente actuará en la dirección del gradiente negativo porque se quiere buscar el valor mínimo de la función.

Descenso por Gradiente



$$J(\beta_0, \beta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(\beta_0 + \beta_1 X^{(i)} - Y^{(i)}\right)^2$$



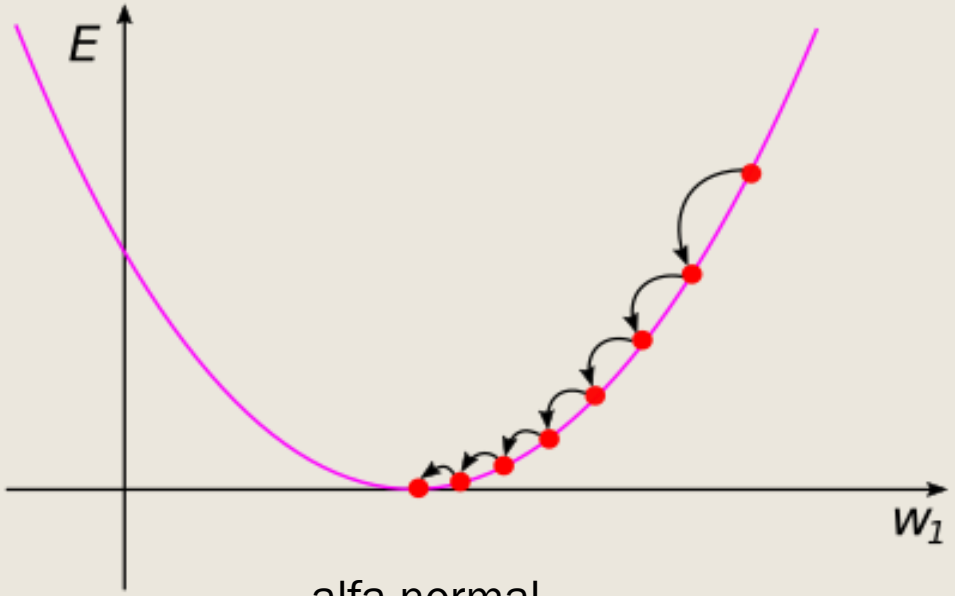
- En cada paso

$$\beta_0 = \beta_0 - \alpha \frac{\partial f}{\partial \beta_0}$$

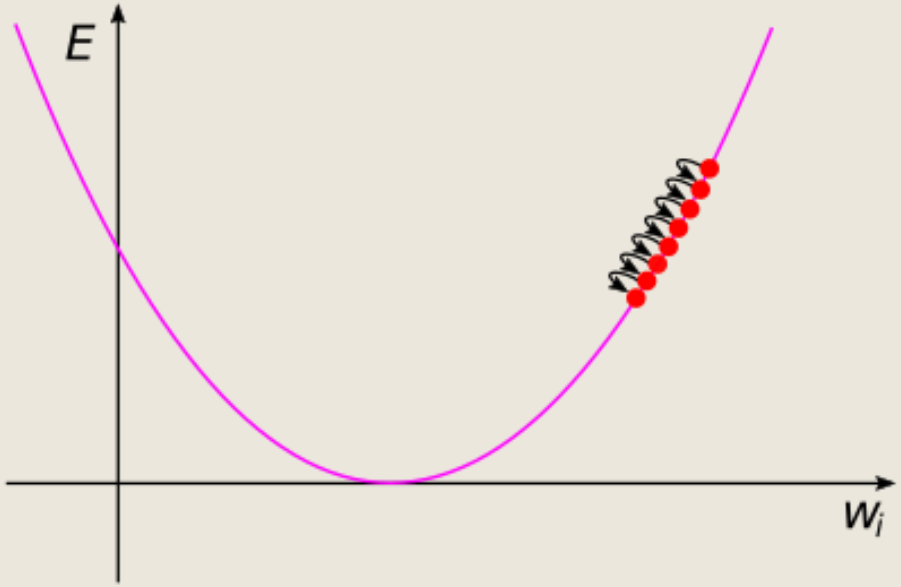
$$\beta_1 = \beta_1 - \alpha \frac{\partial f}{\partial \beta_1}$$

Tasa de Aprendizaje

- Con este parámetro podemos definir el tamaño que tendrá cada desplazamiento al descender por la pendiente.
- Está definida como la variable alfa

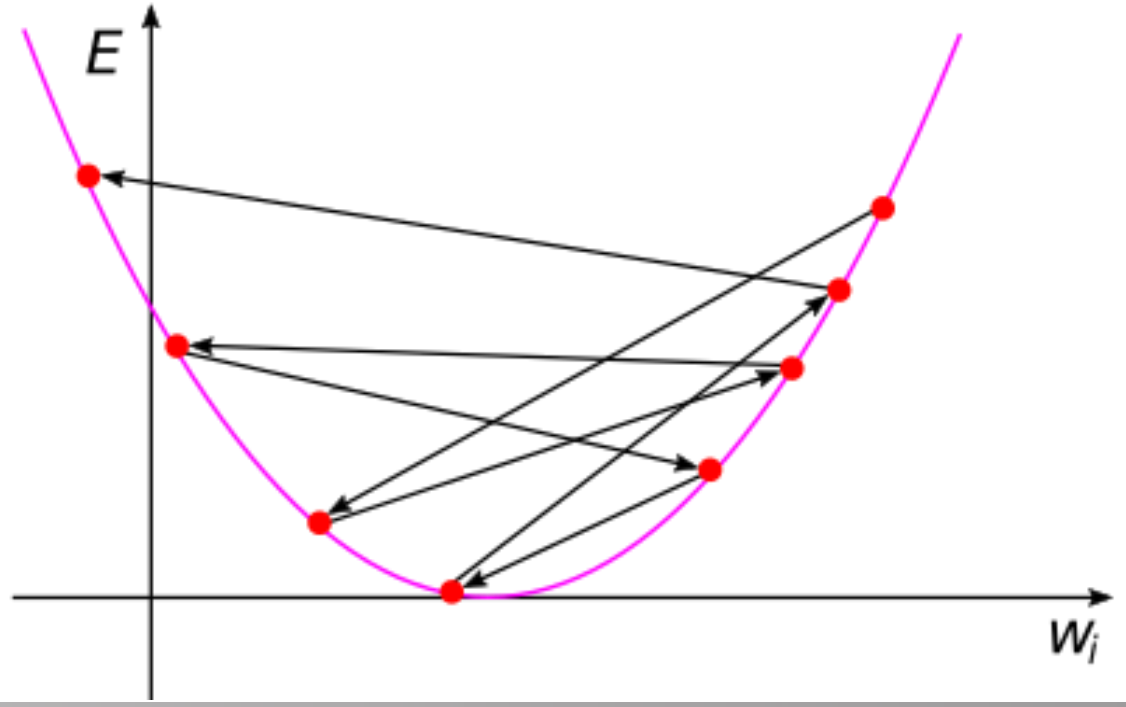


alfa normal



alfa pequeño

- Si la tasa es demasiado grande puede hacer al algoritmo no converger.



Derivadas Parciales

$$J(\beta_0, \beta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(\beta_0 + \beta_1 X^{(i)} - Y^{(i)}\right)^2$$

- Aplicando la regla de la cadena.

$$\frac{\partial J}{\partial \beta_1} = \cancel{2}\frac{1}{\cancel{2}m}\sum_{i=1}^{m}\left(\beta_0 + \beta_1 X^{(i)} - Y^{(i)}\right)X^{(i)}$$

$$\frac{\partial J}{\partial \beta_0} = \cancel{2}\frac{1}{\cancel{2}m}\sum_{i=1}^{m}\left(\beta_0 + \beta_1 X^{(i)} - Y^{(i)}\right)$$

$$\frac{\partial J}{\partial \beta_1} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - Y^{(i)}\right)X^{(i)}$$

$$\frac{\partial J}{\partial \beta_0} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - Y^{(i)}\right)$$

Pseudocódigo del algoritmo

```
1   Inicializar modelo
2   Evaluar modelo
3   Mientras no cumpla el criterio
4       Calcular gradientes
5       Actualizar modelo
6       Evaluar modelo
7   Mostrar modelo resultado
```

MACHINE LEARNING

Laboratorio de Inteligencia Artificial

Escalamiento de variables

- Podemos optimizar el funcionamiento del algoritmo de descenso modificando un poco las variables de entrada. Esto es porque los coeficientes β pueden descender mas rápidamente en rangos pequeños.
- Por lo tanto vamos a modificar el rango de cada una de las variables de entrada con alguno de los siguientes procedimientos:

Normalización Promedio

$$X_n{}^{(i)} = \frac{X_n{}^{(i)} - \mu_n}{\max(X_n) - \min(X_n)}$$

Donde:

μ_n *es el promedio de todas las variables* X_n

X_1
150
300
267
120

$$\mu_n = 209.25$$
$$Max - Min = 209.25$$

X_1
-0.33
0.50
0.32
-0.50

Normalización Min-Max

$$X_n{}^{(i)} = \frac{X_n{}^{(i)} - \min(X_n)}{\max(X_n) - \min(X_n)}$$

Escalamiento

$$X_n{}^{(i)} = \frac{X_n{}^{(i)}}{\max(X_n)}$$

Estandarización

$$X_n{}^{(i)} = \frac{X_n{}^{(i)} - \mu_n}{\sigma(X_n)}$$

Donde:

$\sigma(X_n)$ *es la derivacion estandar de todas las variables* X_n

- Otra ventaja de modificar las entradas es para evitar que las variables de mayor magnitud dominen a las otras en el aprendizaje del algoritmo.

$$\hat{y} = \beta_0 + \beta_1 X + \beta_1 X$$

Genero [0, 1]

Altura [50, 200]

La variable de altura al ser mayor va dominar a la de genero en el entrenamiento.

Parámetros vs Híper parámetros

- Los parámetros son la parte mas importante de machine learning. Es la parte que el algoritmo aprende de los datos. La clave es estimarlos correctamente.
- Los híper parámetros son valores que se configuran antes del entrenamiento del modelo. Podría decirse que son configuraciones que afectan el comportamiento del modelo y no provienen de los datos.

α β_0 β_2 λ β_1

El problema de la clasificación

- La diferencia entre los problemas de regresión(visto anteriormente) y clasificación es que para esta ultima la variable de salida es un valor **cualitativo**. El objetivo es que el sistema aprenda a clasificar una entrada entre un **conjunto finito** de opciones.

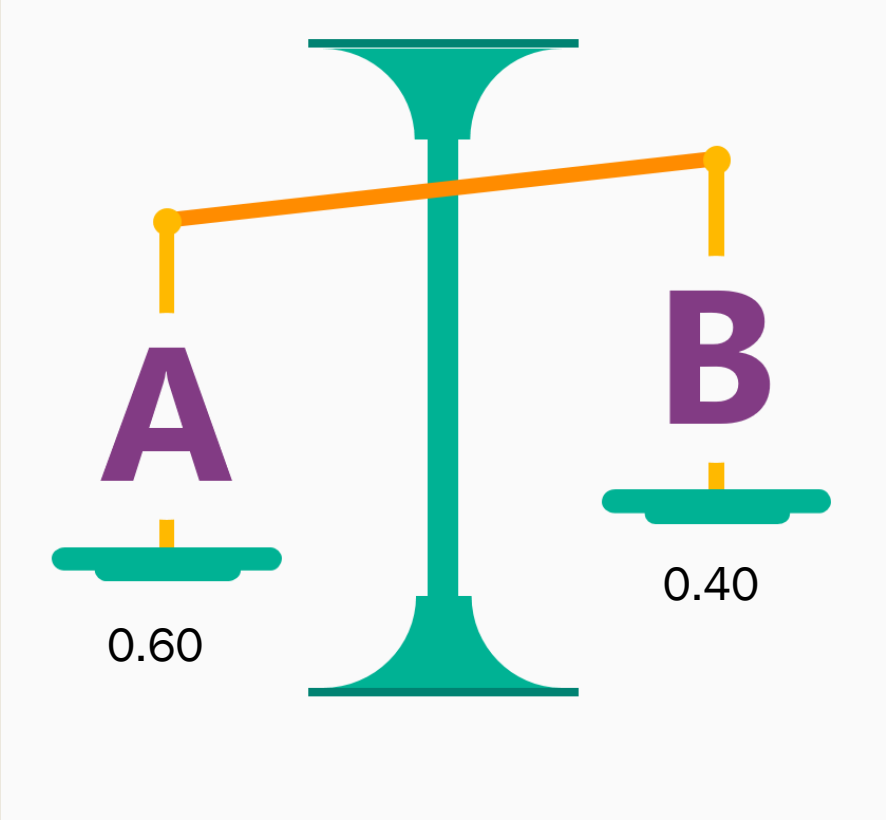
Ejemplos de su utilización

- ¿Es este un buen cliente?
- ¿Este correo es spam?
- ?Esta transacción es fraudulenta¿
- ¿Comprara el cliente este producto?

¿Cómo funciona?

Probabilidades

¿Es un A o B?



Es A

Regresión Logística

- Es una de las técnicas básicas del machine learning para los problemas de clasificación. Esta conformada por dos partes que son:
- Una combinación lineal
- Una operación logística

X1	X2	...Xn	Y
X1	X2	Xn	Y
X1	X2	Xn	Y
X1	X2	Xn	Y

$Y \in [0,1]$

$$z = \beta_0 + \beta_1 X_1 + \cdots \beta_n X_n$$

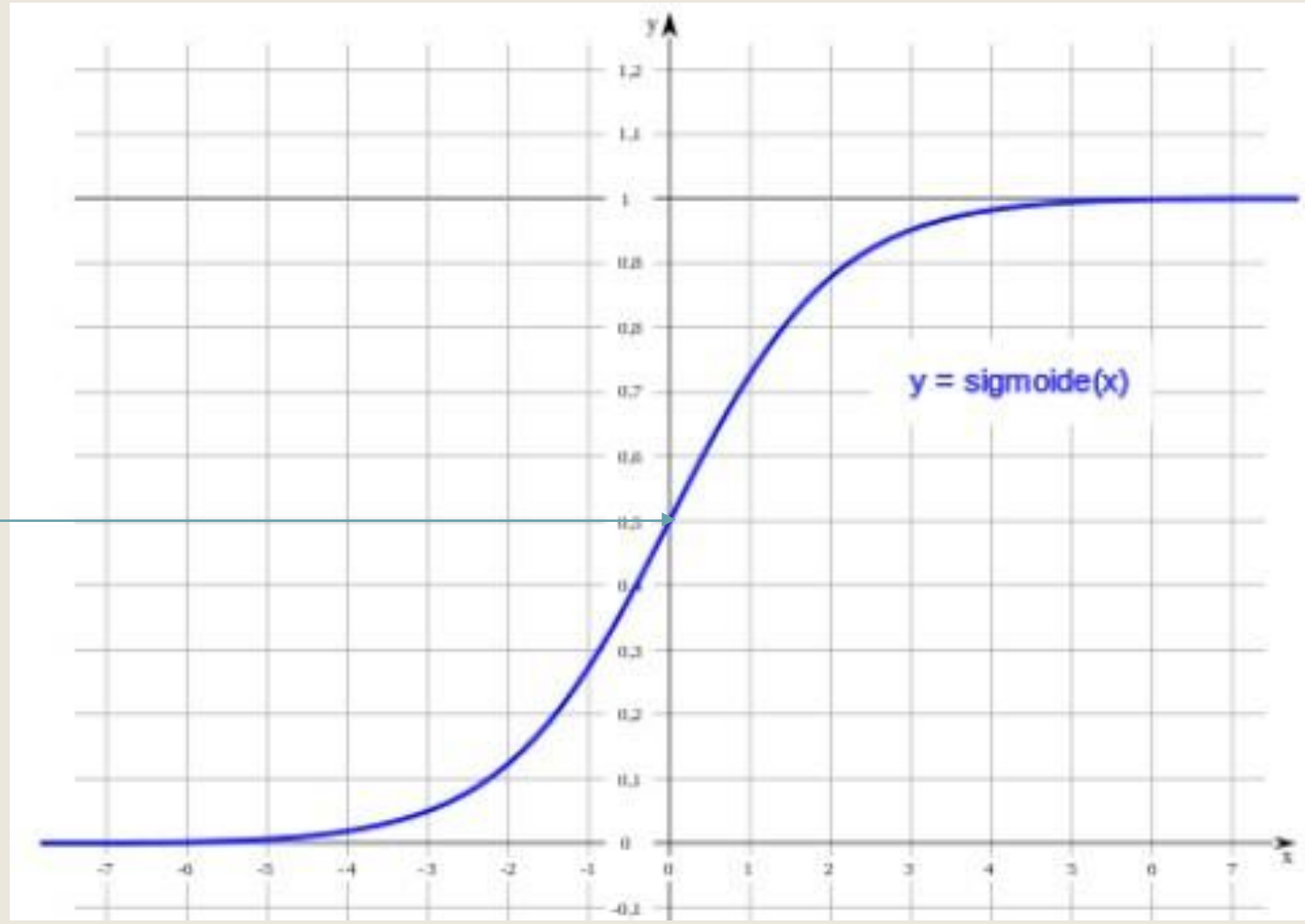
Coeficientes

$$\hat{y} = \sigma(z)$$

Función Logística

Función logística o sigmoide

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

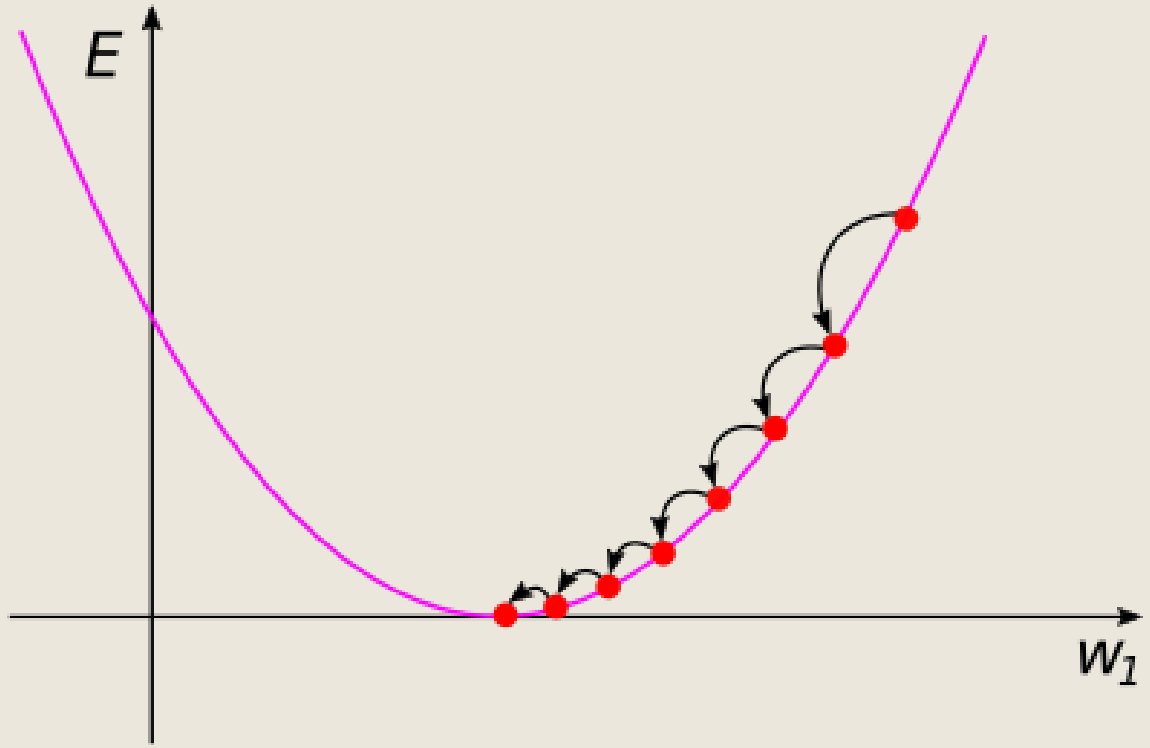


Limite de decisión

Es el punto en el cual decimos si es o no es

Descenso por gradiente

- El algoritmo se aplica de la misma manera que en la regresión lineal. Se calcula sucesivamente el gradiente hasta llegar a la región con el menor error posible.



- En cada paso

$$\beta_0 = \beta_0 - \alpha \frac{\partial J}{\partial \beta_0}$$

$$\beta_1 = \beta_1 - \alpha \frac{\partial J}{\partial \beta_1}$$

...

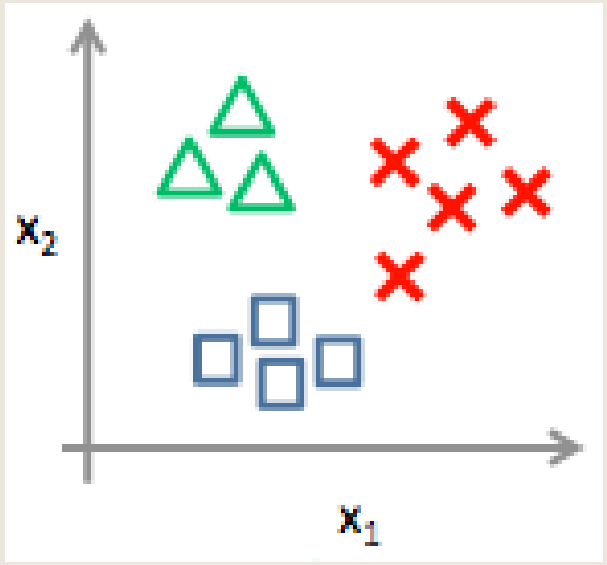
$$\beta_n = \beta_n - \alpha \frac{\partial J}{\partial \beta_n}$$

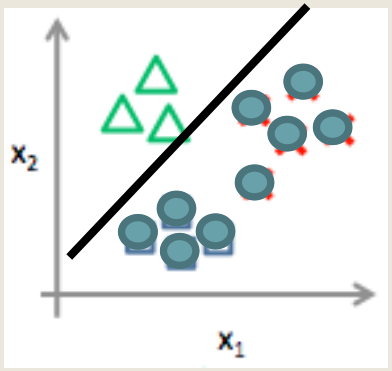
Seudocódigo

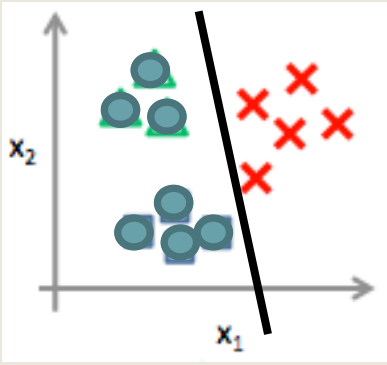
```
1  Inicializar modelo
2  Evaluar modelo
3  Mientras No cumpla el criterio
4      Calcular gradientes
5      Actualizar modelo
6      Evaluar modelo
7  Mostrar el modelo
8
```

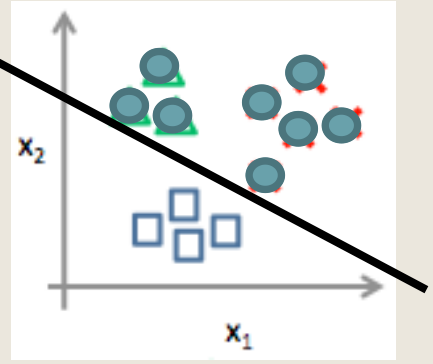
Multiclasificación

- Muchas veces queremos que el algoritmo logre clasificar entre mas de dos clases.





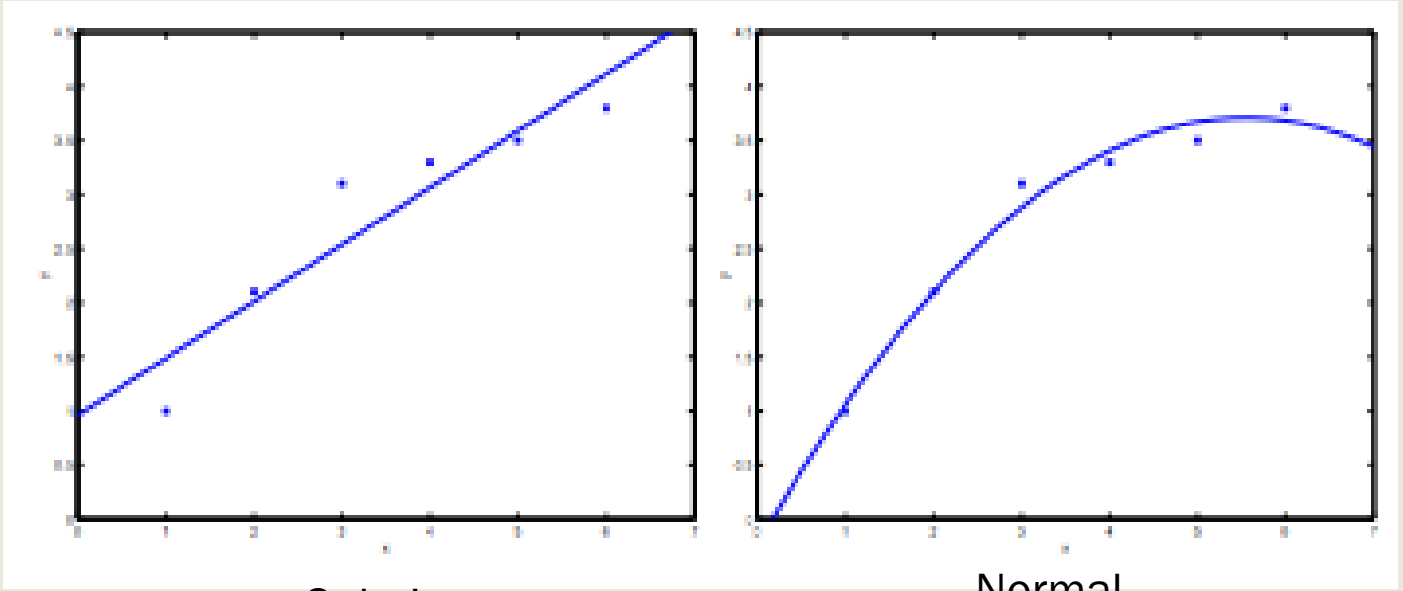




Se usa un clasificador por cada clase y se elige el que tenga la mayor probabilidad.

Subajuste (underfitting)

- El subajuste ocurre cuando nuestro modelo no hay podido aprender correctamente la tendencia de los datos. Por lo que no hace buenas predicciones en el conjunto de entrenamiento ni en el de prueba. Para eliminarse se puede:
- Utilizar mas variables de entrada
- Mas tiempo de entrenamiento

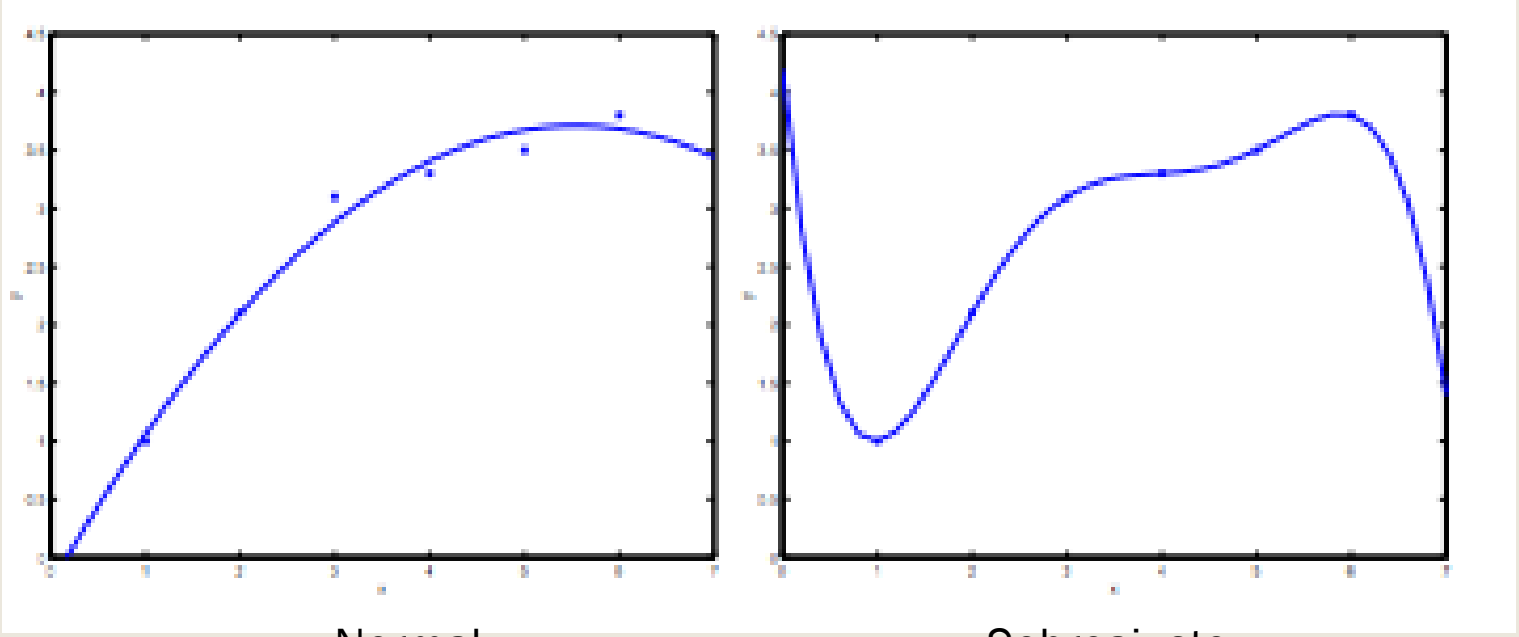


Subajuste

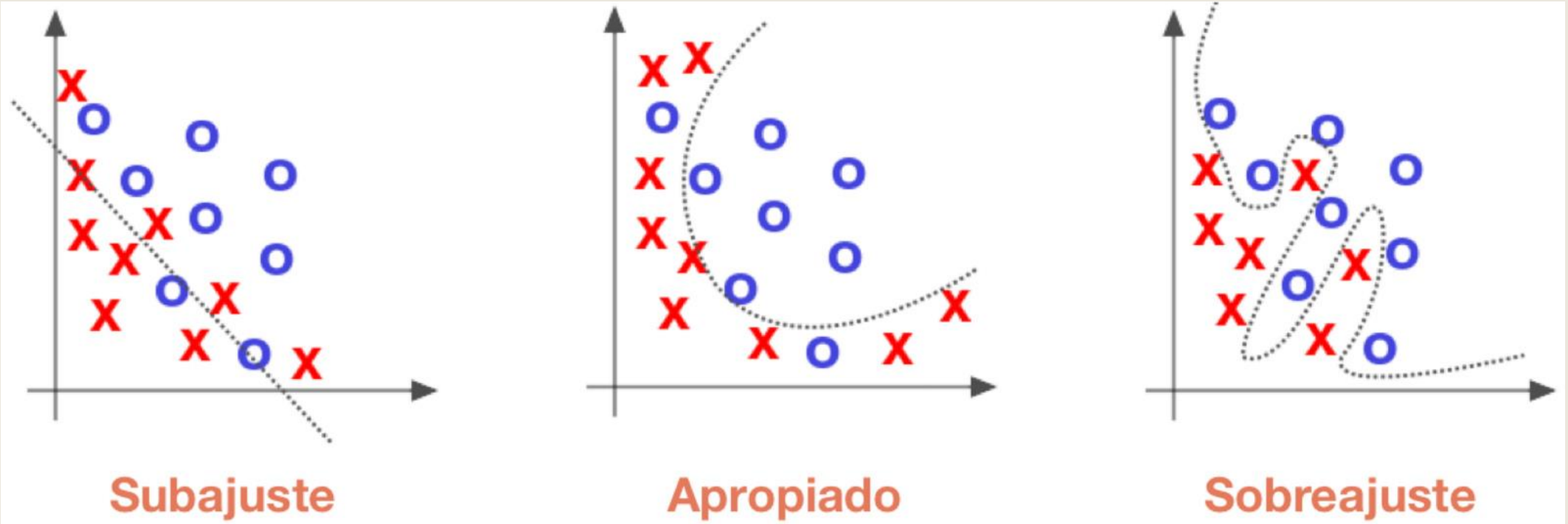
Normal

Sobreajuste (overfitting)

- El sobreajuste ocurre en el extremo contrario, cuando el modelo se ha acoplado tan bien a los datos de entrenamiento que no puede generalizar ese conocimiento a nuevos datos. Para eliminarlo se puede:
- Quitar algunas variables de entrada
- Utilizar regularizacion



Normal Sobreajuste


Subajuste
Apropiado
Sobreajuste

Regularización

- Es una técnica que consiste en agregar un termino extra a la función de costo para reducir la complejidad del modelo.
- Se basa en la premisa de que **entre mas grandes son los coeficientes de un modelo mas complejo se vuelve este**.

$$J = \frac{1}{m}\sum_{i=0}^{m} Cos\left(X^{(i)}\beta, Y^{(i)}\right) + \frac{\lambda}{2m}\sum_{j=0}^{n}\left|\beta_j\right|^2$$

Termino de perdida

Mide que tan bien se ajusta el modelo a los datos.

Termino de regularización

Mide que tan complejo es el modelo.

Tasa de regularización

- Con este parámetro se puede controlar el impacto que tendrá la regularización en el entrenamiento del modelo. Al elegir el valor de lambda, el objetivo es lograr un balance adecuado entre la simpleza y el ajuste de los datos.

Si es demasiado grande

El modelo será simple, pero se corre el riesgo de caer en el Subajuste ya que el modelo no aprenderá lo suficiente.

Si es demasiado pequeño

El modelo puede llegar a ser mas complejo, literalmente es como si no usaras regularización por lo que puedes caer en el sobreajuste.

Como cambio la función de costo, ¿También cambió la expresión del gradiente?

$$\nabla \mathrm{J} = \left[\frac{\partial J}{\partial \beta_0}, \frac{\partial J}{\partial \beta_1}, \ldots . \frac{\partial J}{\partial \beta_n}\right]$$

Ahora si

Sin regularización

$$\frac{\partial J}{\partial \beta_0} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - Y^{(i)}\right)$$

$$\frac{\partial J}{\partial \beta_1} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - Y^{(i)}\right)X_1$$

...

$$\frac{\partial J}{\partial \beta_n} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - Y^{(i)}\right)X_n$$

Con regularización

$$\frac{\partial J}{\partial \beta_0} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - Y^{(i)}\right) + \frac{\lambda}{m}\beta_0$$

$$\frac{\partial J}{\partial \beta_1} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - Y^{(i)}\right)X_1 + \frac{\lambda}{m}\beta_1$$

...

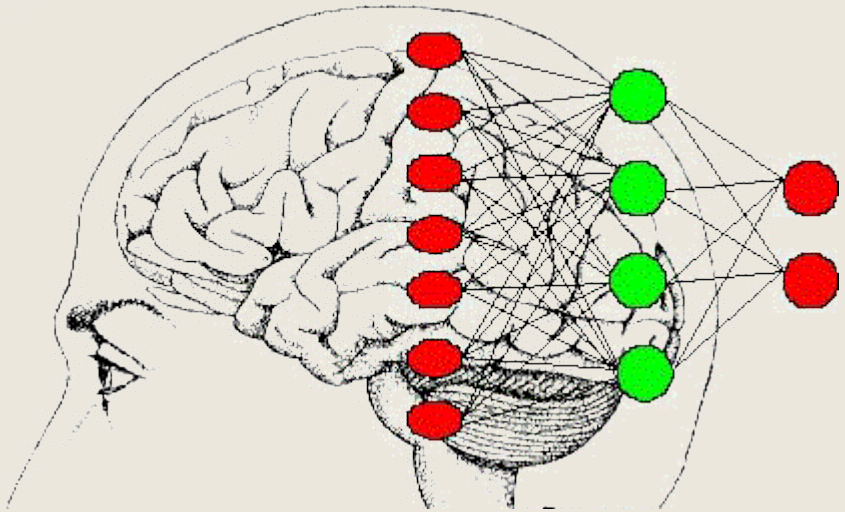
$$\frac{\partial J}{\partial \beta_n} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}^{(i)} - Y^{(i)}\right)X_n + \frac{\lambda}{m}\beta_n$$

REDES NEURONALES

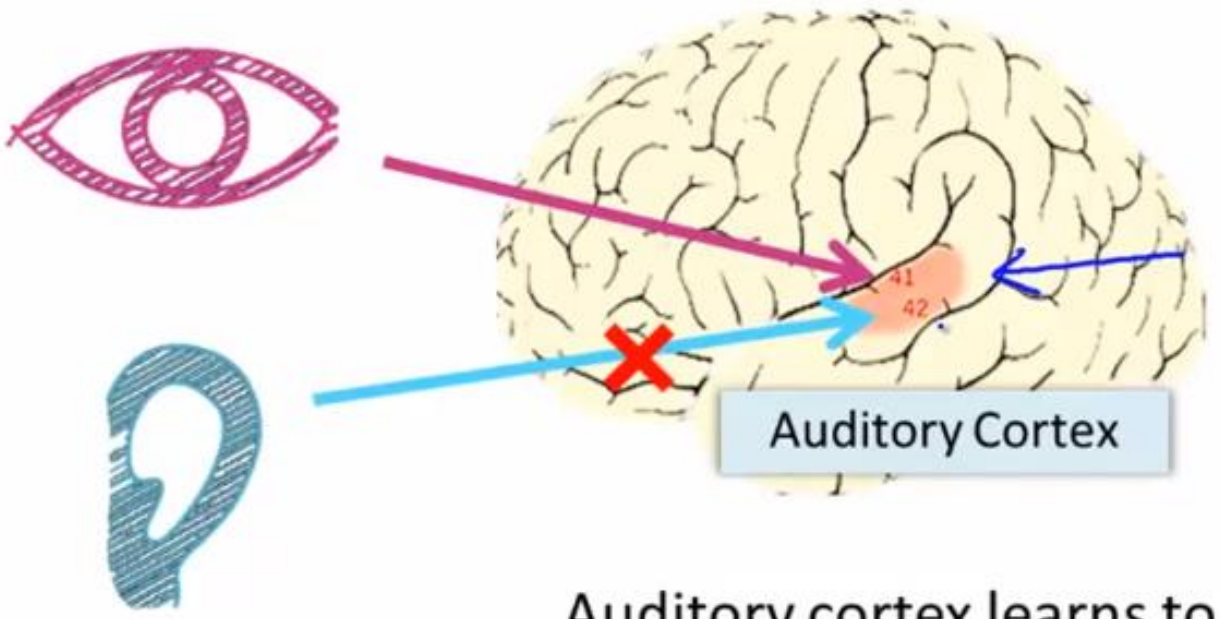
Laboratorio de Inteligencia Artificial

¿Qué son las redes neuronales?

- Es un modelo matemático simplificado que emula el modo en el cual nuestro cerebro procesa la información. Es decir, que esta inspirado en el comportamiento biológico de las neuronas y la estructura del cerebro.



The “one learning algorithm” hypothesis



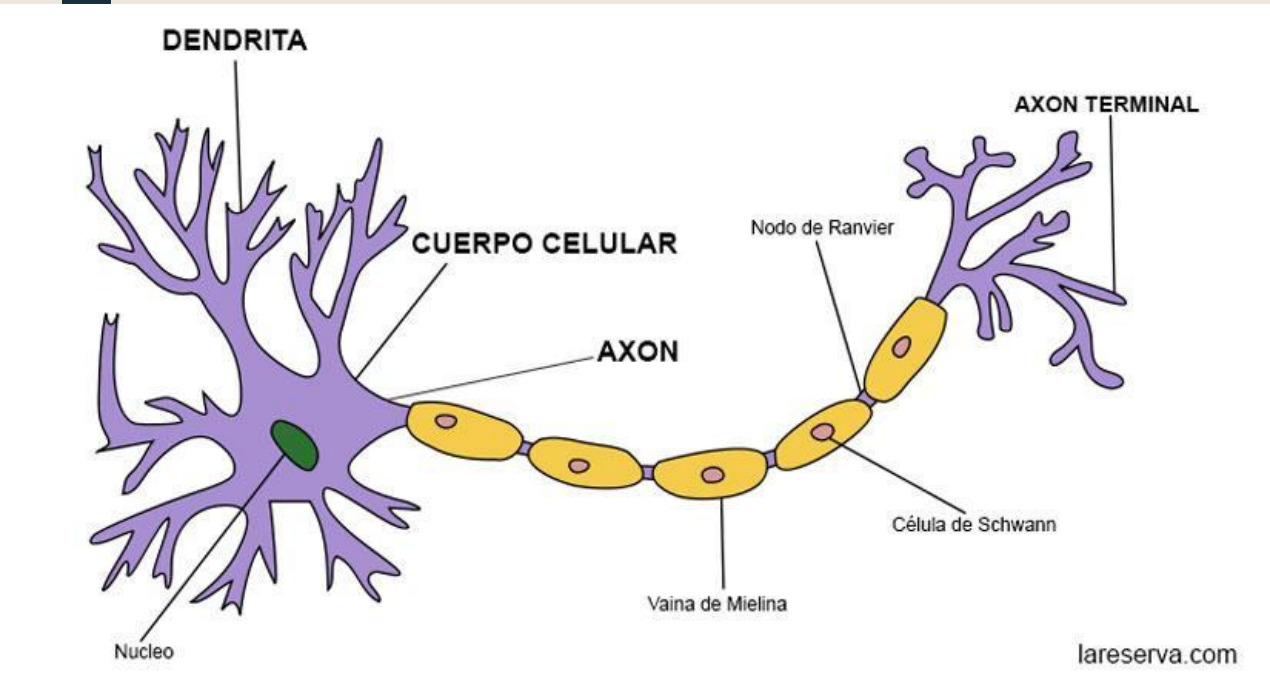
Auditory cortex learns to see

See with your tongue

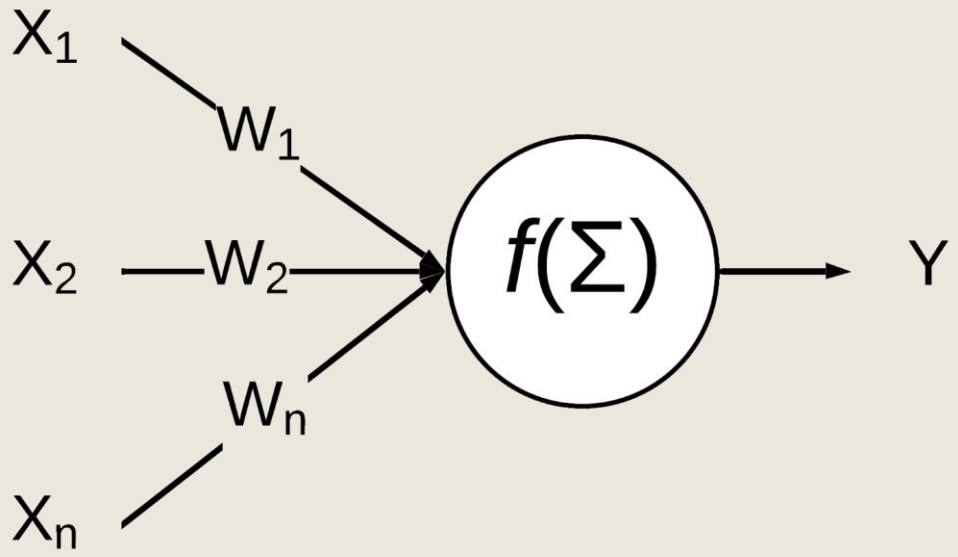
¿Cuál es la finalidad de una red neuronal?

PREDECIR

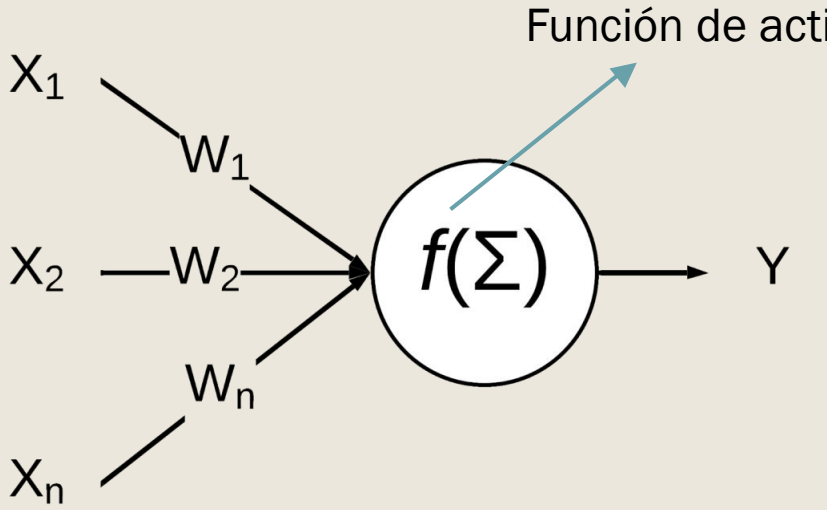
El concepto de neurona



Real



Modelo



$$Y = f(x_1 w_1 + x_2 w_2 + \cdots x_n w_n) \rightarrow f(XW)$$

¿Nos parece conocido?

- Cuando la función de activación posee las siguientes formas:

$$f(z) = \frac{1}{1 + e^{-z}}$$

$$\mathrm{Y} = \frac{1}{1 + e^{-XW}}$$

Es una regresión logística!!!

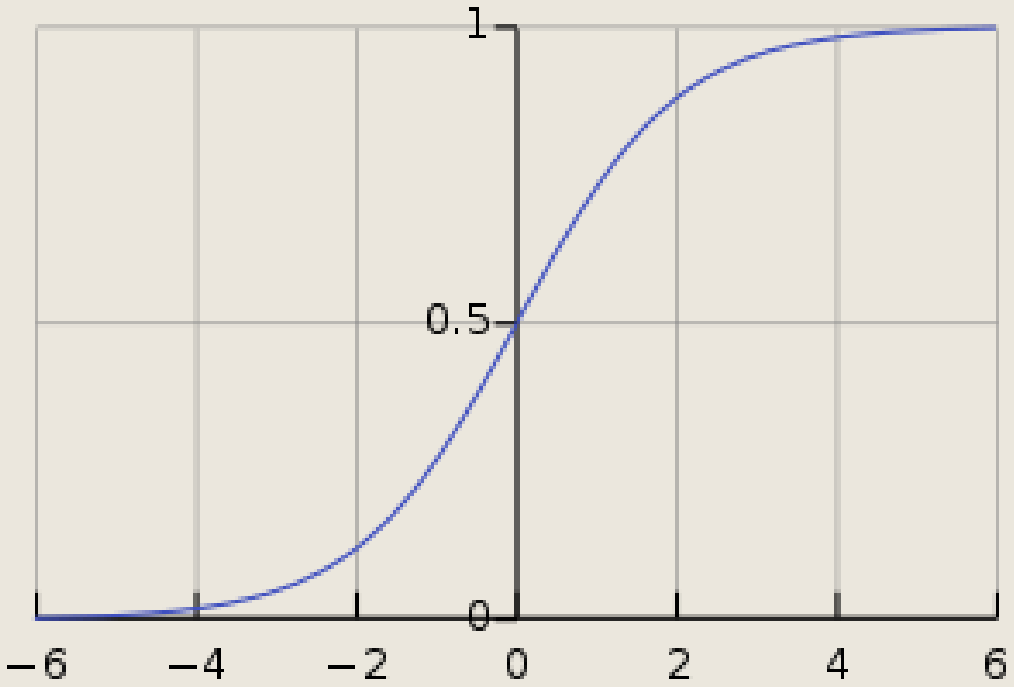
$$f(z) = z$$

$$\mathrm{Y} = XW$$

Es una regresión lineal!!!

Funciones de activación

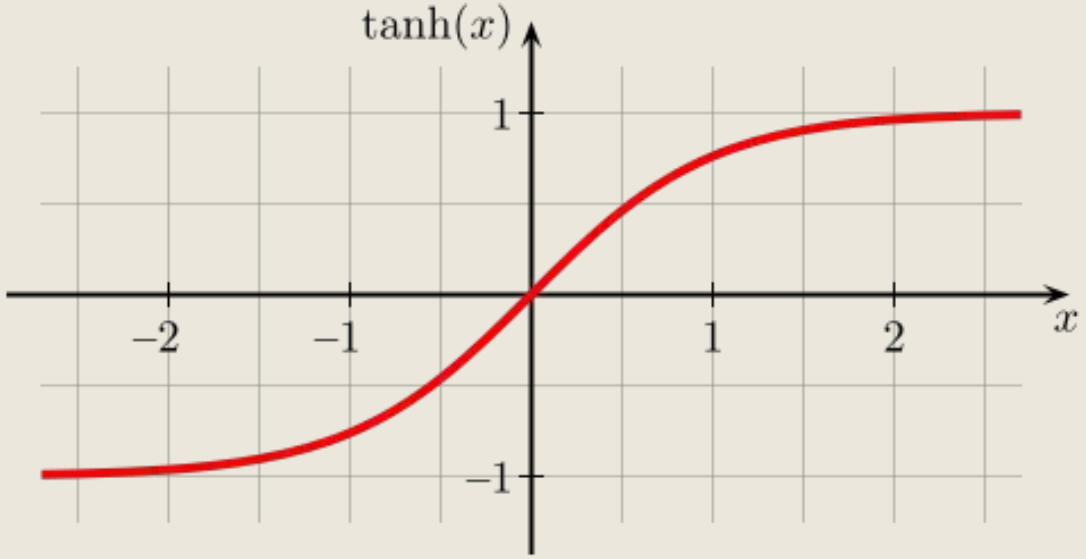
Sigmoide



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

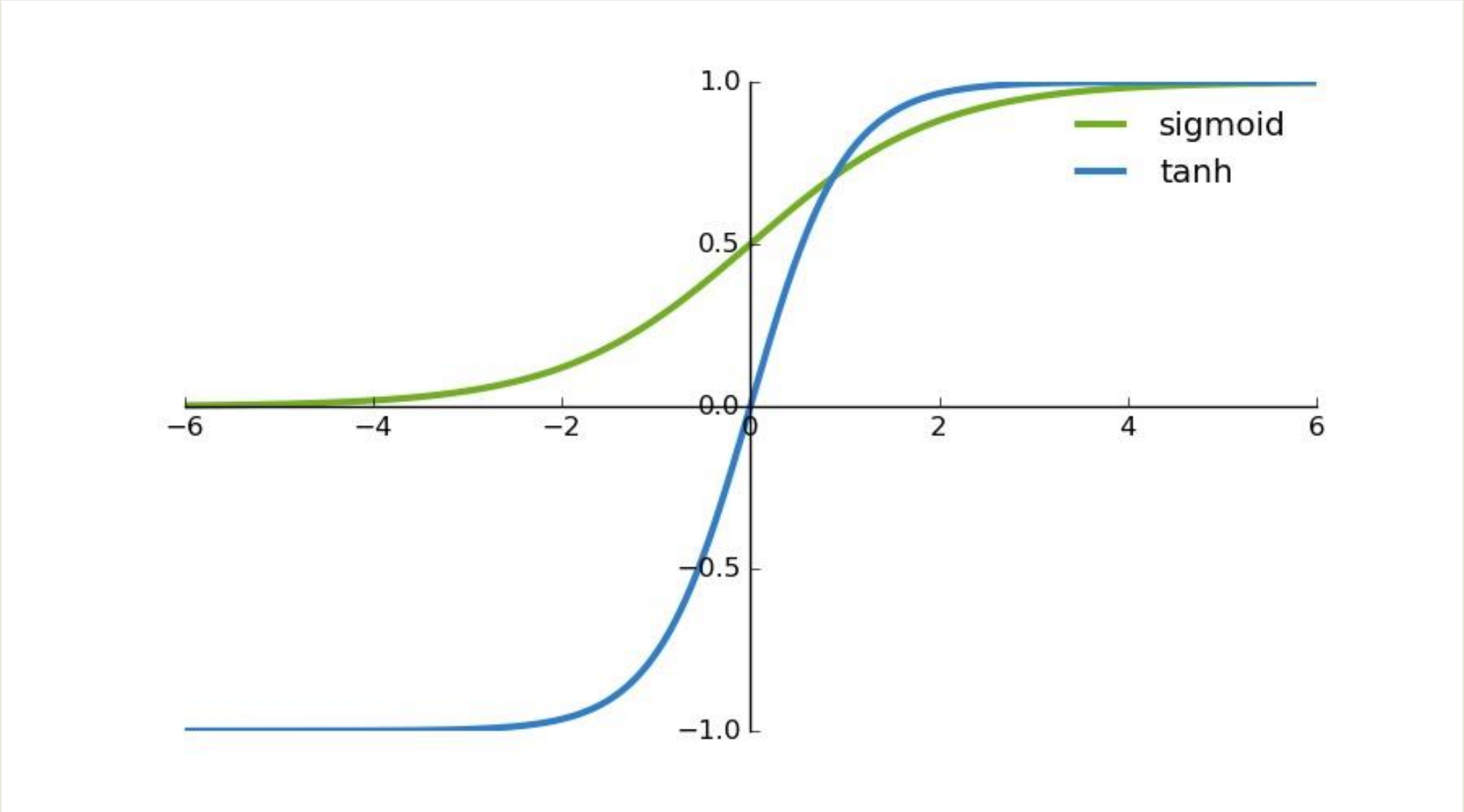
$$f'(x) = f(x)[1 - f(x)]$$

Tanh

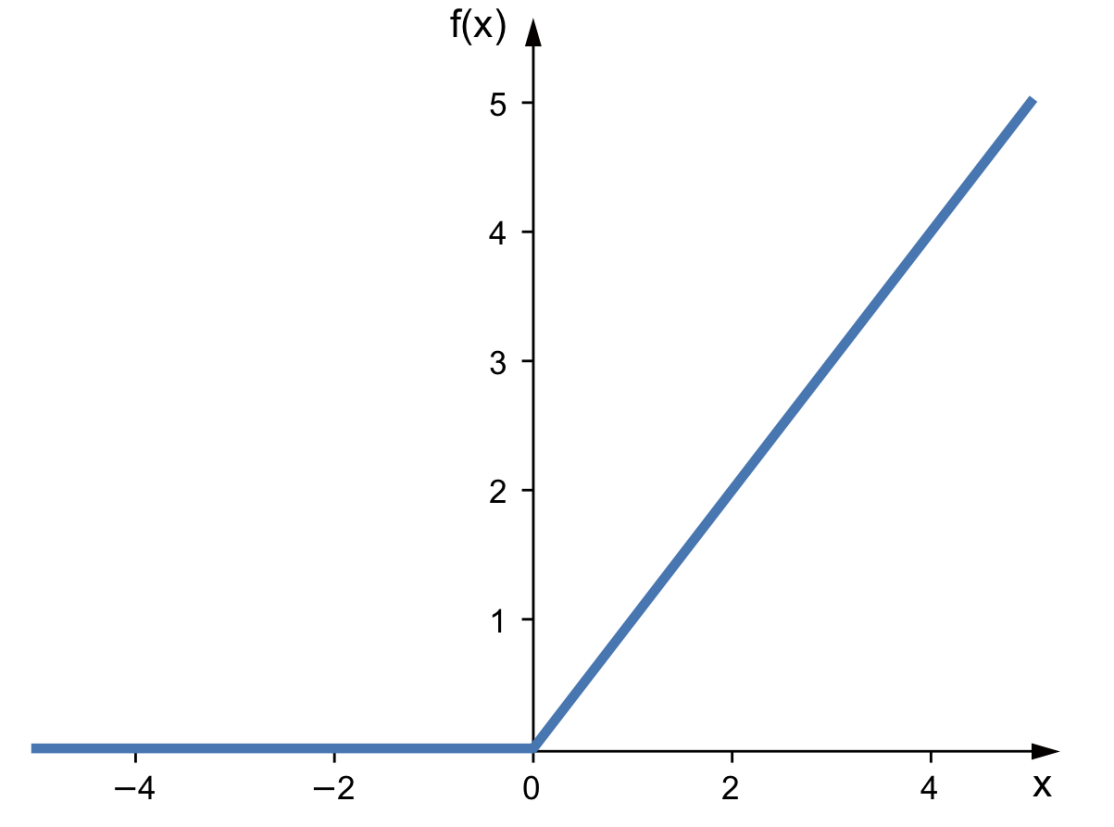


$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

$$f'(x) = 1 - [f(x)]^{2}$$


1.0
0.5
0.0
−0.5
−1.0
−6
−4
−2
0
2
4
6
sigmoid
tanh

Relu

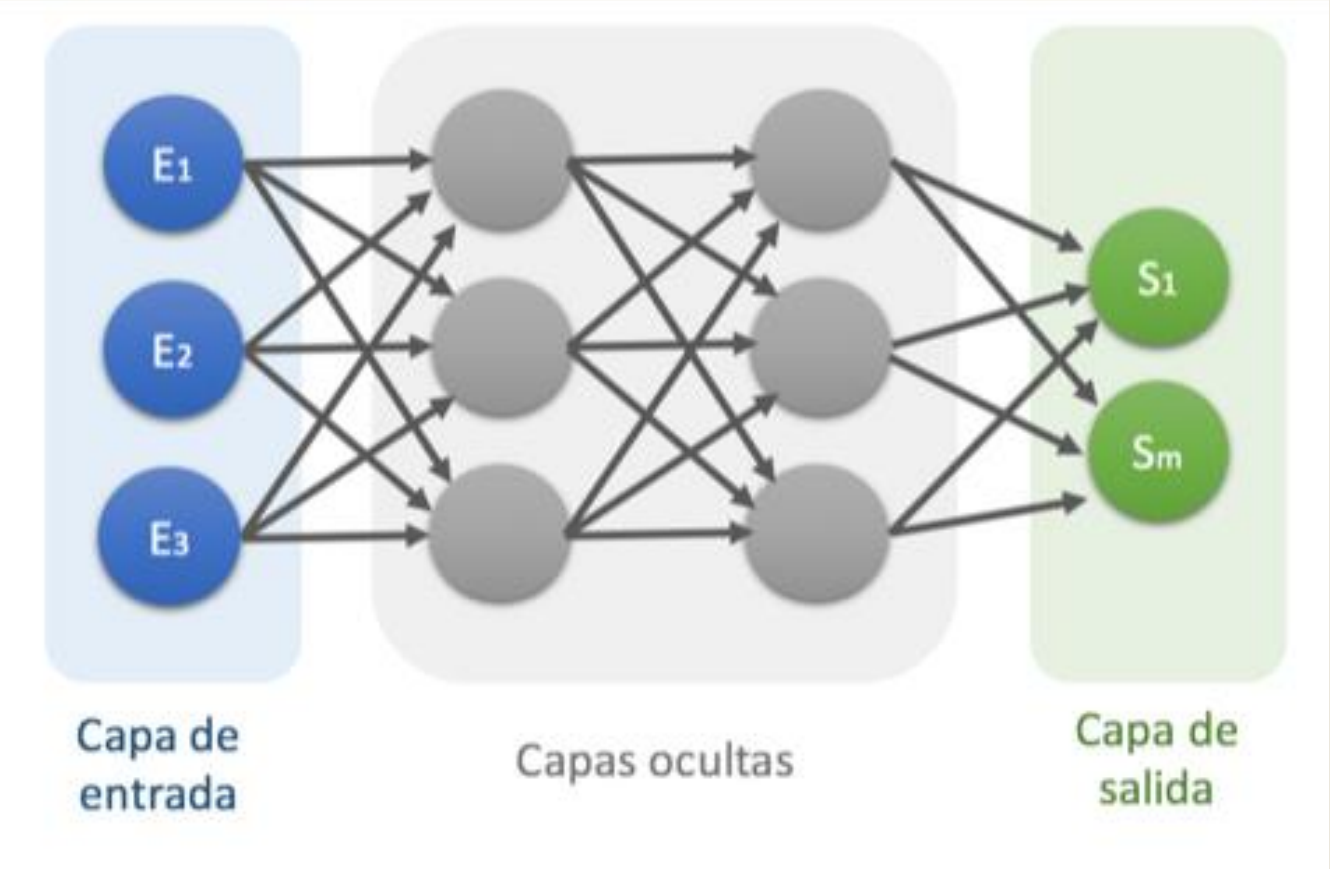


$$f(x) = \max(0, x)$$

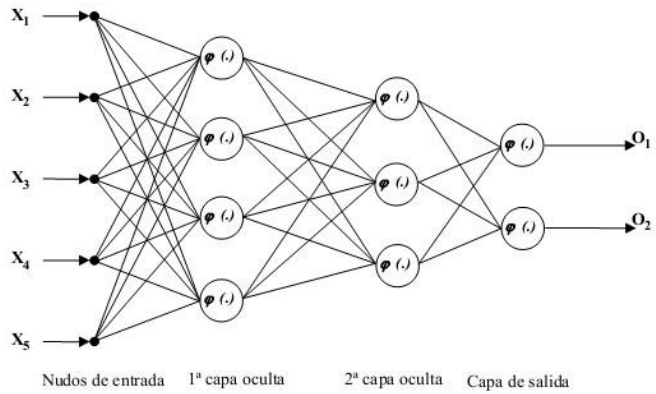
$$f'(x) = ?$$

Red Neuronal

- Es un conjunto de neuronas que trabajan conjuntamente para resolver un problema en especifico.



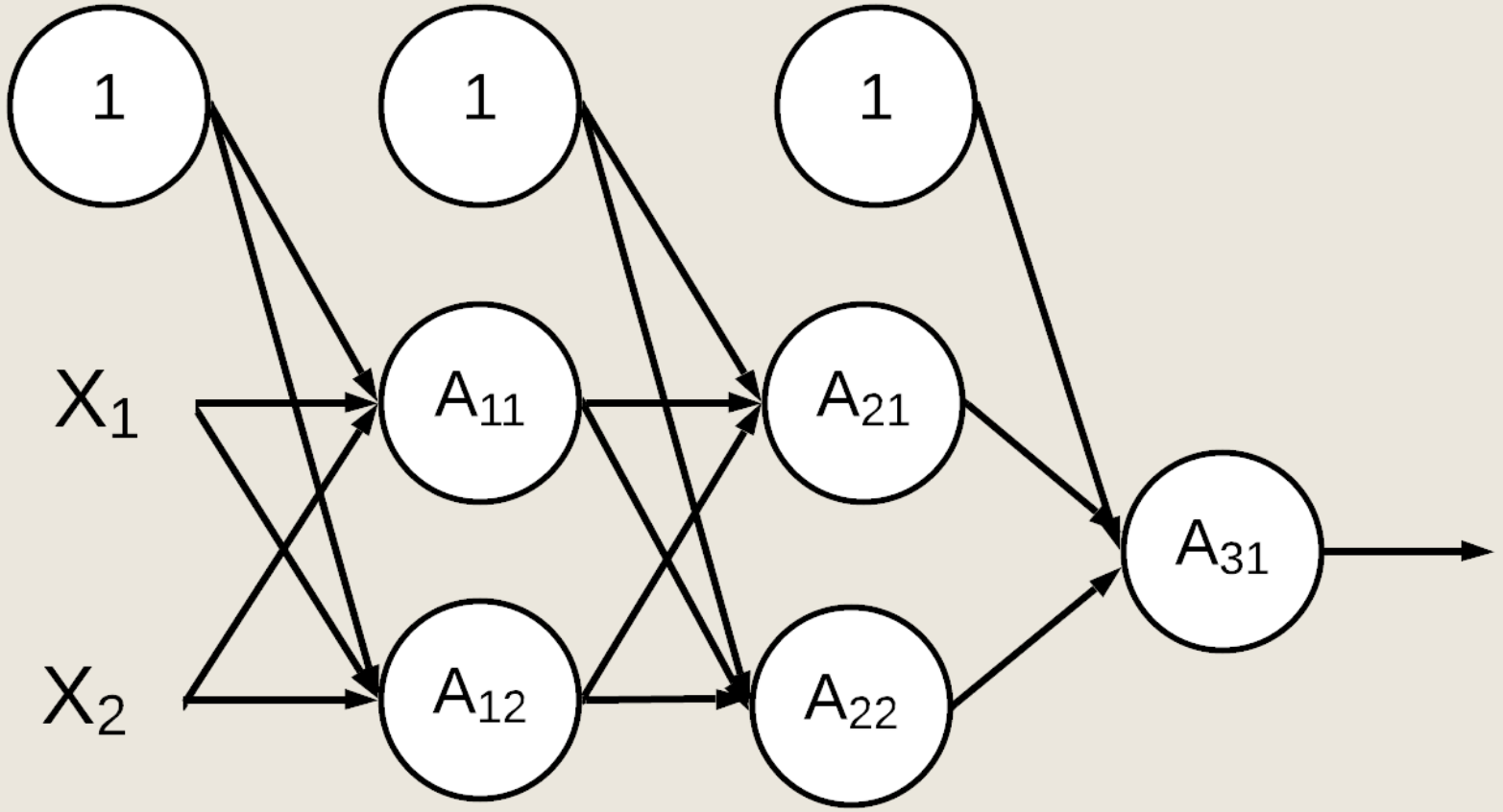
- Entre mas capas ocultas tenga una red neuronal podrá aprender comportamientos mas complejos pero a la vez incrementara el proceso de entrenamiento del modelo.
- Cuando se cuentan las capas de una red no se toma en cuenta a la capa de entradas.



Red de 3 capas

Entradas Bias

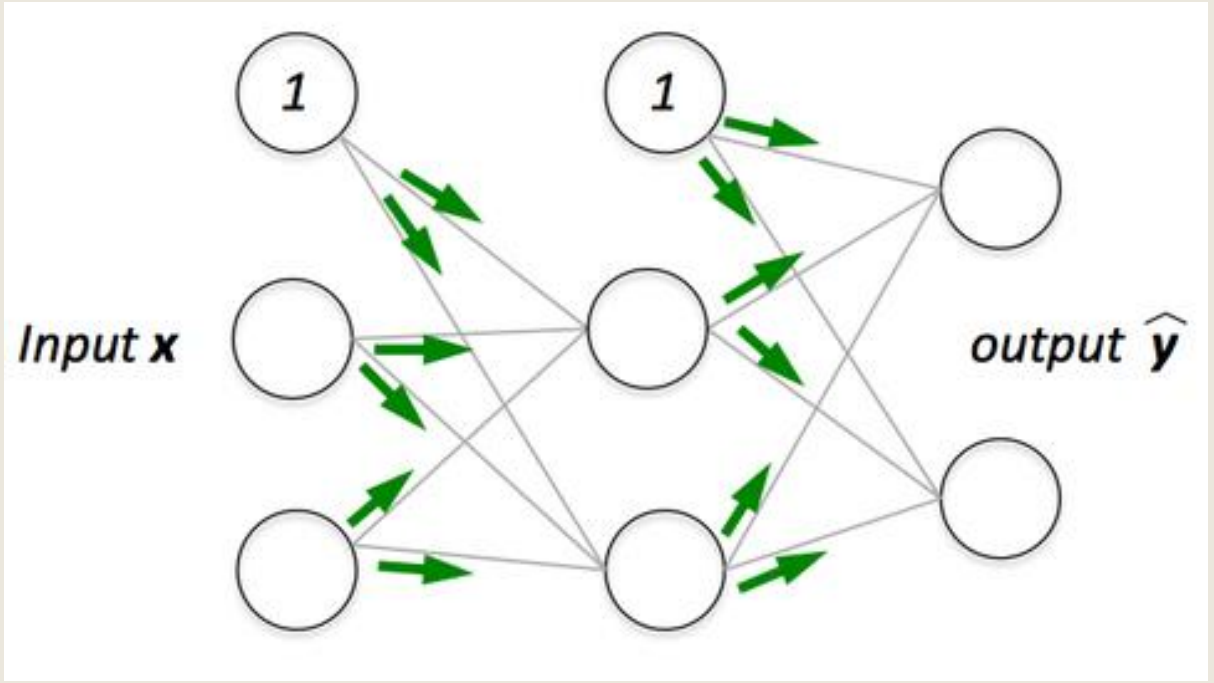
- Es un valor constante que ayuda al modelo de una manera que pueda ajustarse mejor a los datos de entrenamiento.



Funcionamiento

Propagación hacia delante (Forward propagation)

- Es el proceso por el cual todas las capas de una red neuronal procesan los datos de la entrada. Se le llama así porque los cálculos fluyen de la primera capa hasta la ultima.



IS THIS A

CAT or DOG?

CAT DOG

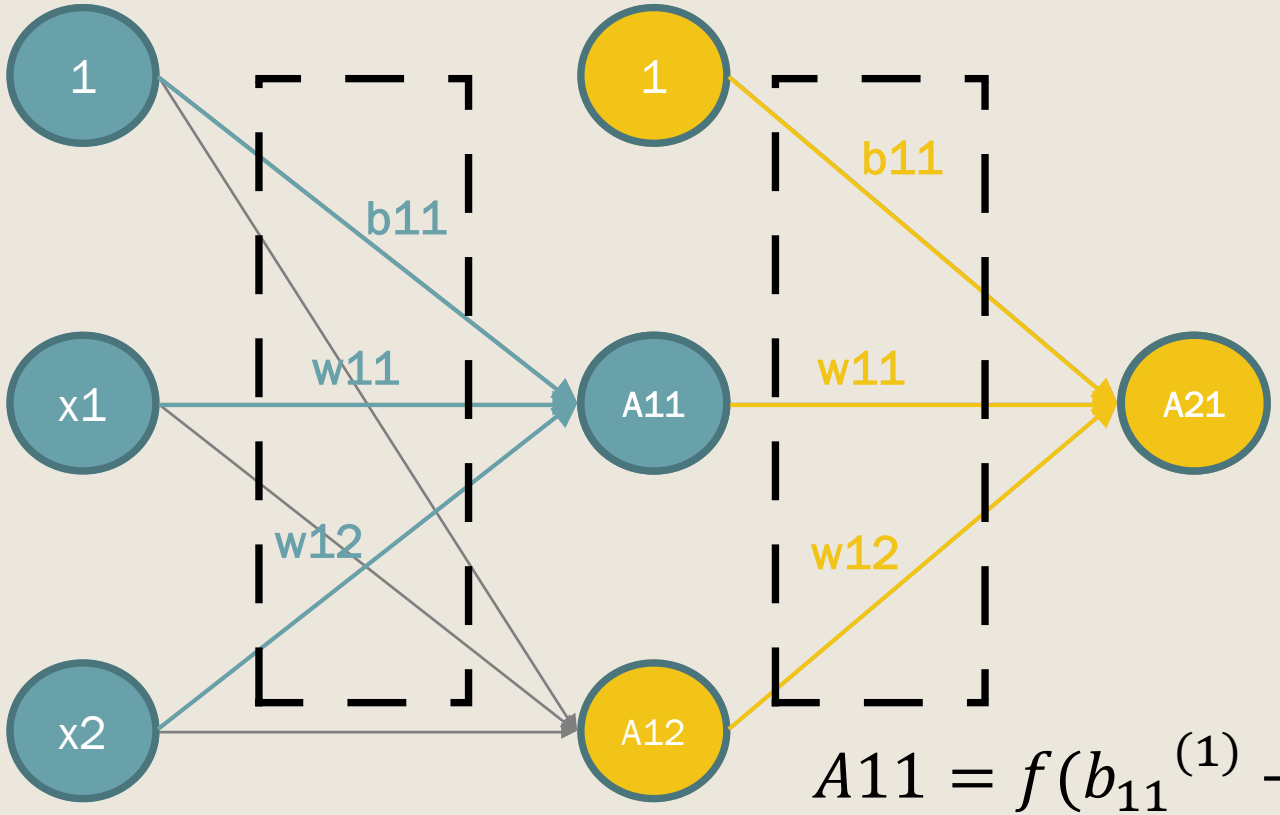
OUTPUT LAYER

ACTIVATED NEURONS

INPUT LAYER

DEEP *neural* NETWORK

¿Quien es A21?



$$A_{21} = \hat{y}$$

$$A11 = f(b_{11}{}^{(1)} + x_1 w_{11}{}^{(1)} + x_2 w_{12}{}^{(1)})$$

$$A21 = f(b_{11}{}^{(2)} + A_{11} w_{11}{}^{(2)} + A_{12} w_{12}{}^{(2)})$$

Función de costo

- Indica que tan bien se acercan las predicciones al conjunto de entrenamiento.

$$\mathrm{J} = \frac{1}{m}\sum_{i=1}^{m} Cost(\hat{y}^{(i)}, y^{(i)})$$

Problemas de regresión

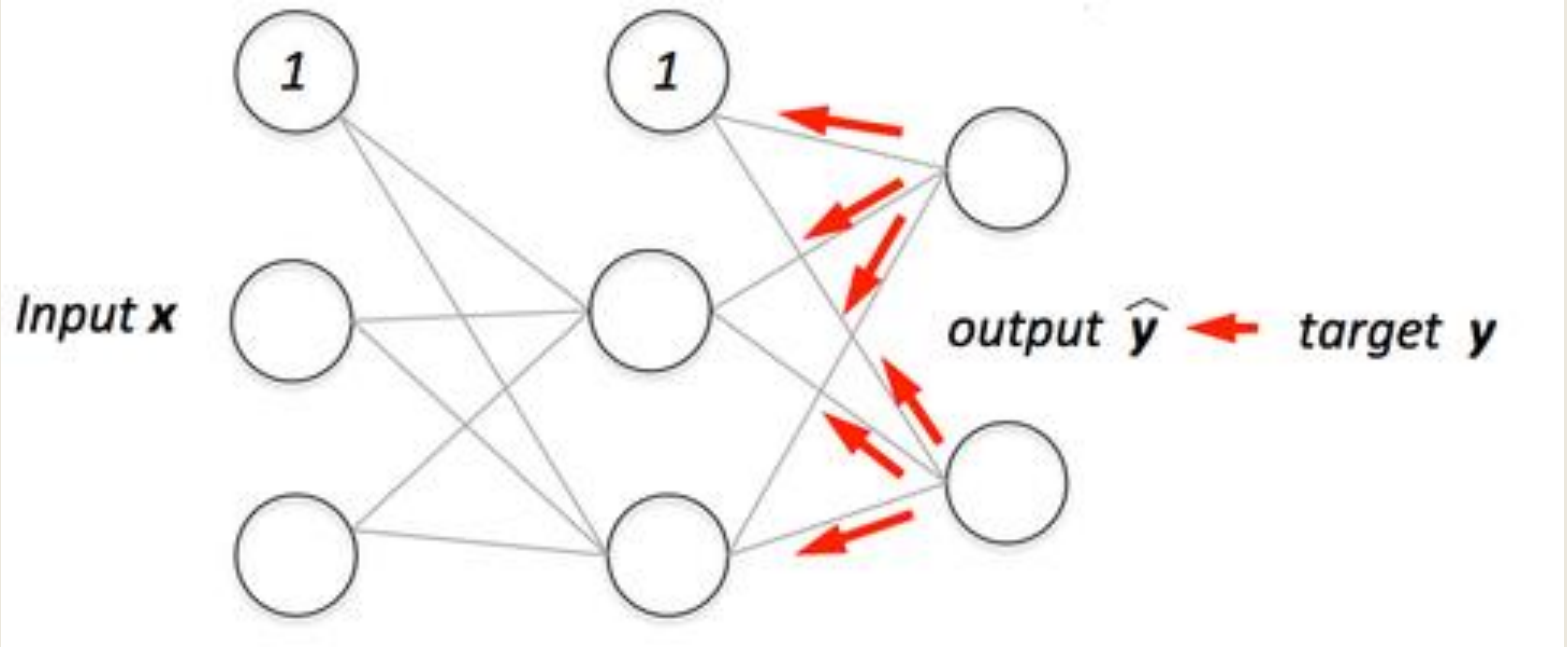
$$Cost(X, Y) = \frac{1}{2}(\hat{y} - Y)^2$$

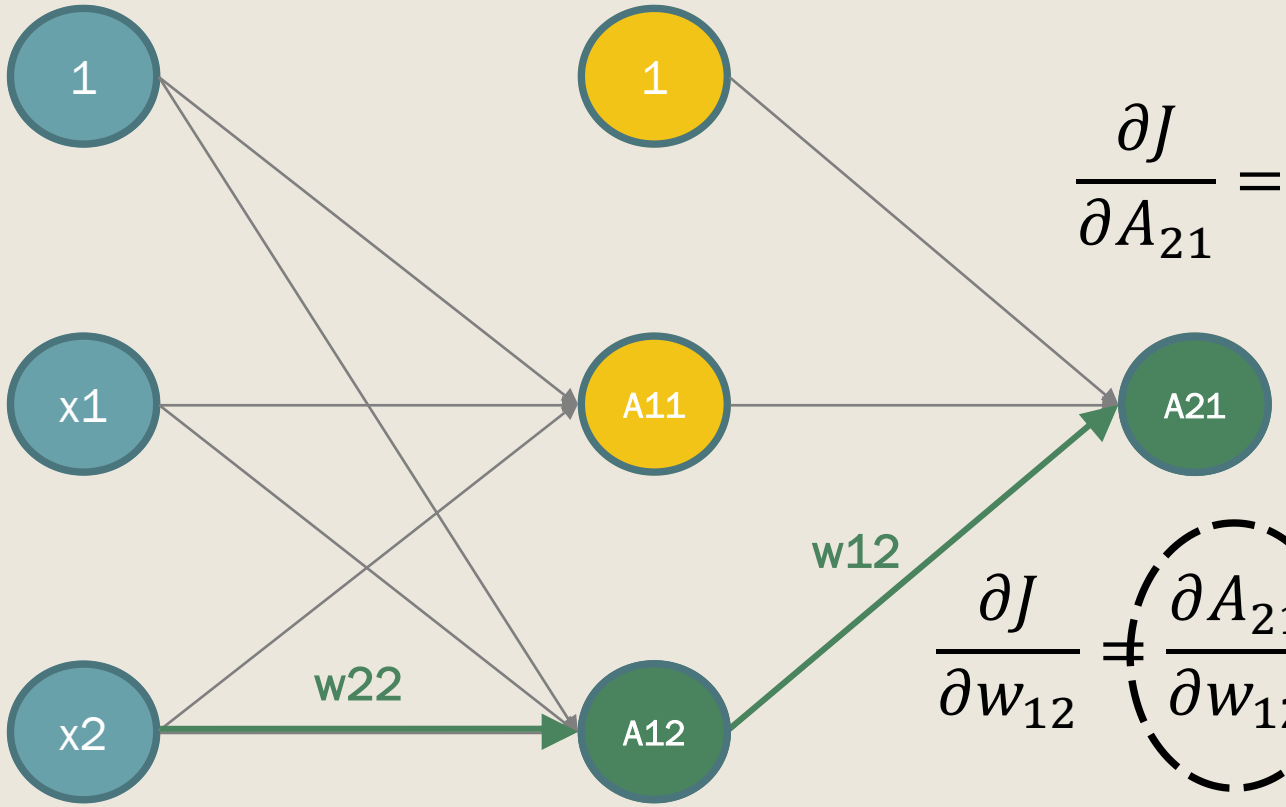
Problemas de clasificación

$$Cost(\hat{y}, Y) = -Y * Log(\hat{y}) - (1 - Y) * Log(1 - \hat{y})$$

Propagación hacia atrás (Back propagation)

- Es el proceso por el cual todas las capas de una red neuronal se retroalimentan del error obtenido en la función de costo. Se le llama así porque los cálculos fluyen desde la ultima capa hasta las primeras.





$$\frac{\partial J}{\partial A_{21}} = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})$$

$$\frac{\partial J}{\partial w_{12}} = \frac{\partial A_{21}}{\partial w_{12}} * \frac{\partial J}{\partial A_{21}}$$

$$A21 = f(b_{11}^{(2)} + A_{11}w_{11}^{(2)} + A_{12}w_{12}^{(2)})$$

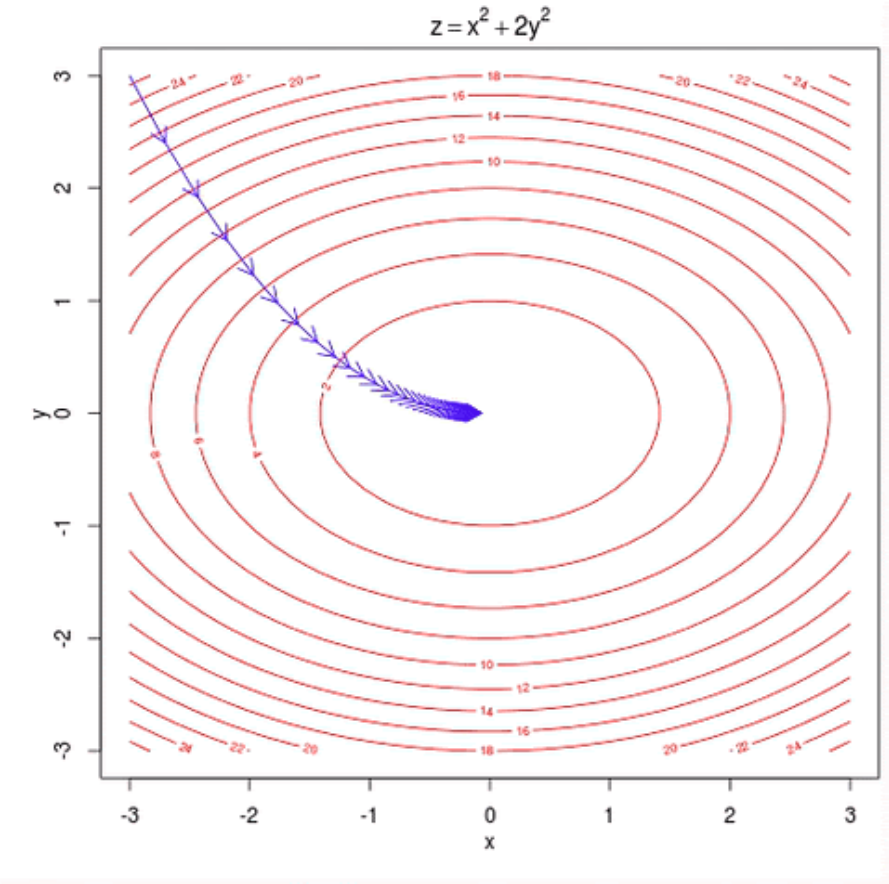
$$\frac{\partial J}{\partial w_{22}} = \frac{\partial A_{12}}{\partial w_{22}} * \frac{\partial J}{\partial A_{12}}$$

$$\frac{\partial A_{21}}{\partial w_{12}} = \frac{\partial f}{\partial w_{12}} * A_{12}$$

$$\frac{\partial J}{\partial A_{12}} = \frac{\partial A_{21}}{\partial A_{12}} * \frac{\partial J}{\partial A_{21}}$$

Descenso de gradiente

- Ya teniendo las derivadas parciales de cada coeficiente procedemos a actualizar los valores de los coeficientes.



$$W = W - \alpha \frac{\partial J}{\partial W}$$

$$B = B - \alpha \frac{\partial J}{\partial B}$$

En resumen

Loop{

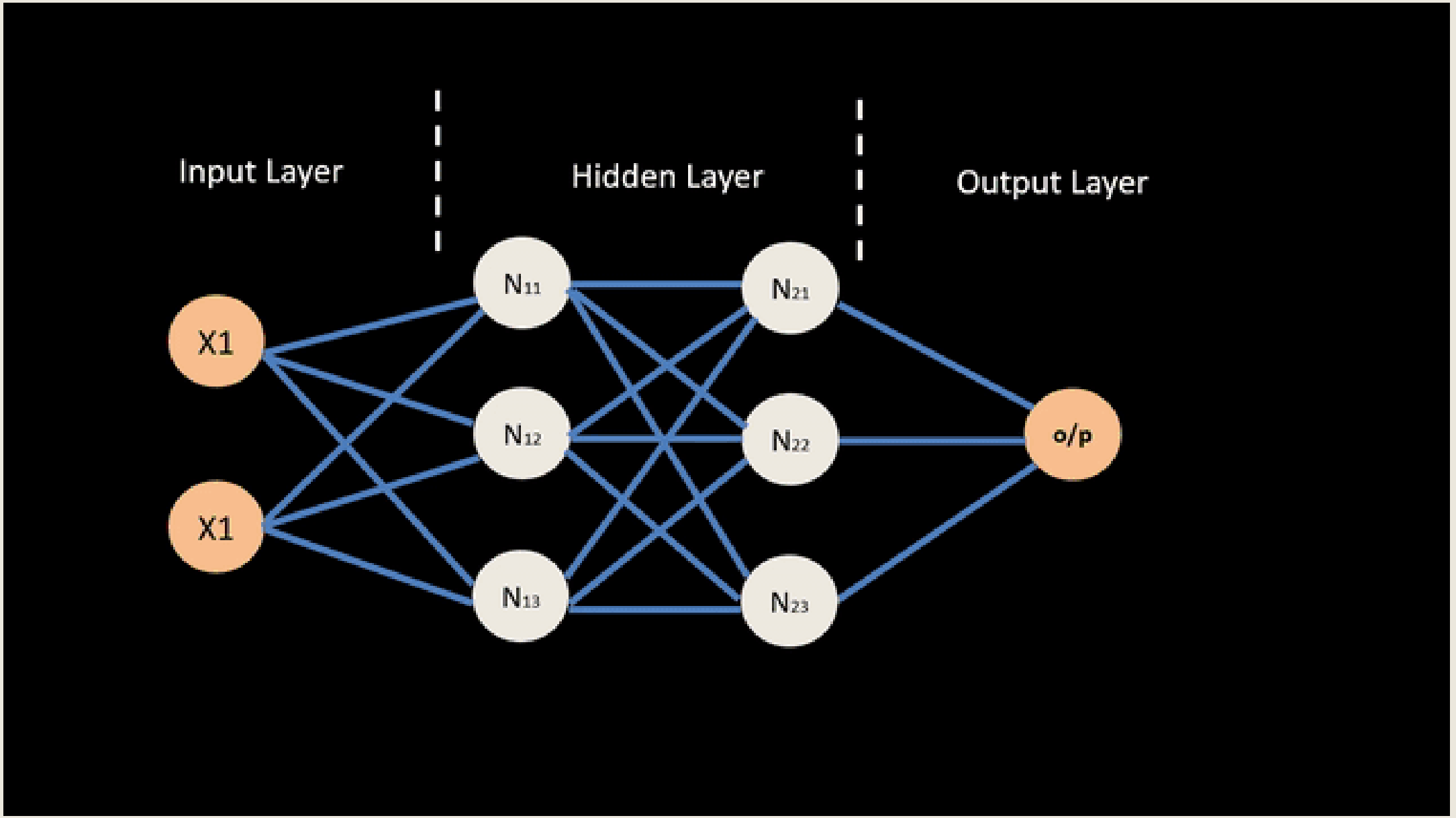
Propagación hacia adelante

Calcular el costo

Propagación hacia atrás

Actualizar coeficientes

}


Input Layer
Hidden Layer
Output Layer
X1
X1
N11
N12
N13
N21
N22
N23
o/p

Conjunto de datos

Para conjuntos pequeños (100 – 10,000)

70 % 15 % 15 %

Para conjuntos grandes (100,000)

98 % 1 % 1 %

Regularización

- Consiste en agregar un termino a la función de costo que penalice la complejidad del modelo. Se basa en la idea de que entre mas grandes son los valores de los coeficientes es mas complejo el modelo.

Regularización L2

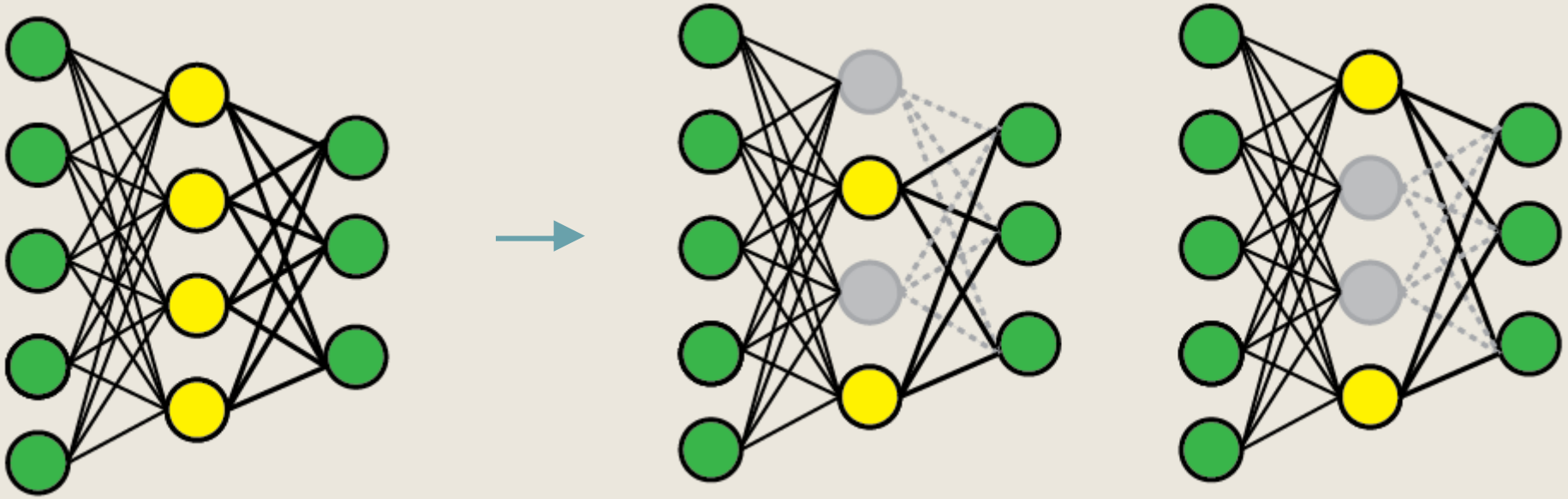
$$J = \frac{1}{m}\sum_{i=0}^{m} Cos\left(\hat{y}^{(i)}, Y^{(i)}\right) + \frac{\lambda}{2m}\sum_{l=0}^{L}\sum_{j=0}^{n}\left|w_j\right|^2$$

Regularización L1

$$J = \frac{1}{m}\sum_{i=0}^{m} Cos\left(\hat{y}^{(i)}, Y^{(i)}\right) + \frac{\lambda}{2m}\sum_{l=0}^{L}\sum_{j=0}^{n}\left|w_j\right|$$

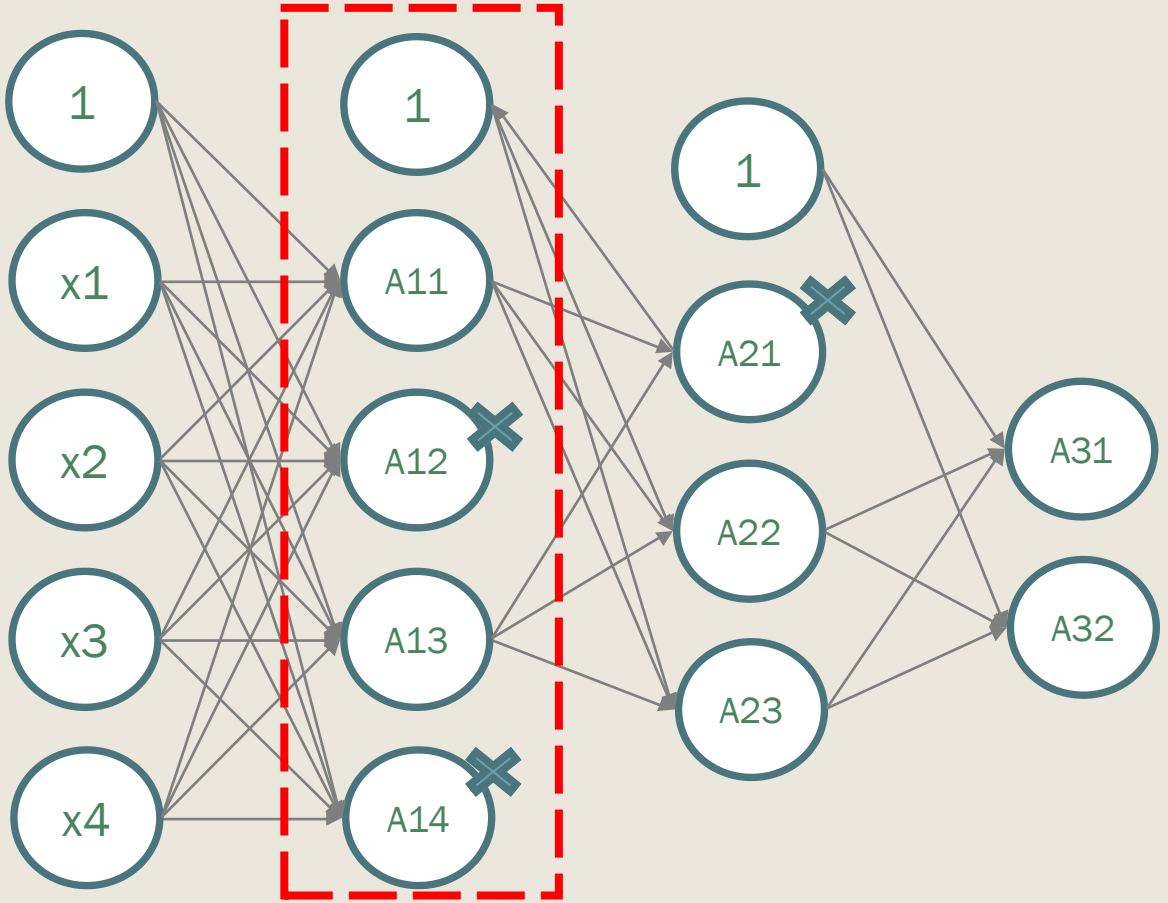
Dropout

- Esta técnica consiste en matar una cantidad de neuronas al azar durante cada iteración del **entrenamiento**, con el objetivo de tener menos parámetros que procesar. Por lo tanto al momento de utilizar los conjuntos de validación y prueba no se utiliza.



Dropout invertido

- Es una de tantas formas que existe para implementar el dropout. Consiste en tener una probabilidad **kp (keep_prob)** que nos indica cual es la probabilidad de que cada una de las neuronas desaparezca.



Kp = 0.7

$$A_1 = [A_{11}, A_{12}, A_{13}, A_{14}]$$

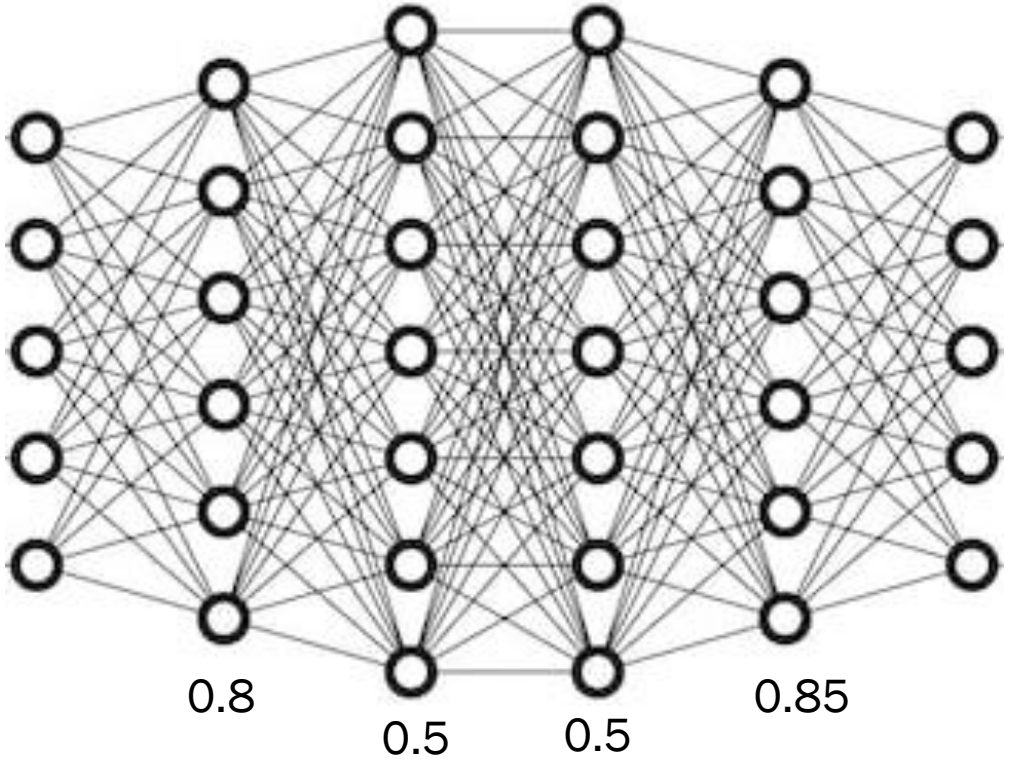
$$d_1 = [0.5, 0.8, 0.4, 0.91]$$

$$A_1 = A_1 * (d_1 < kp)$$

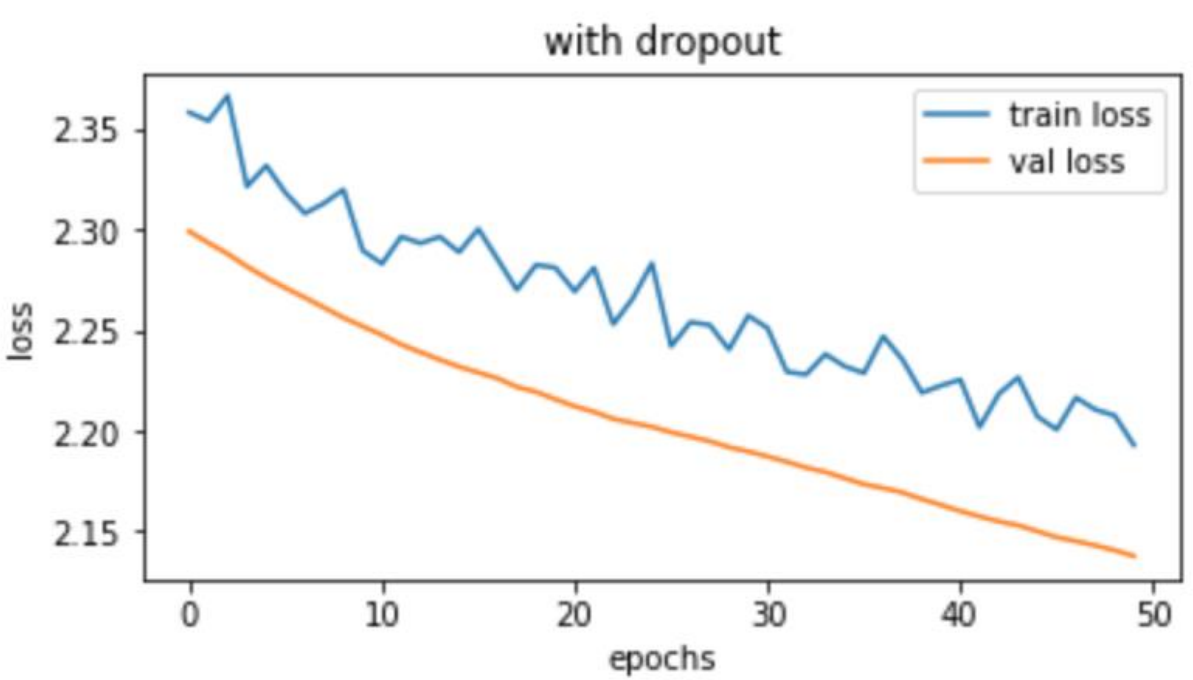
0, 1

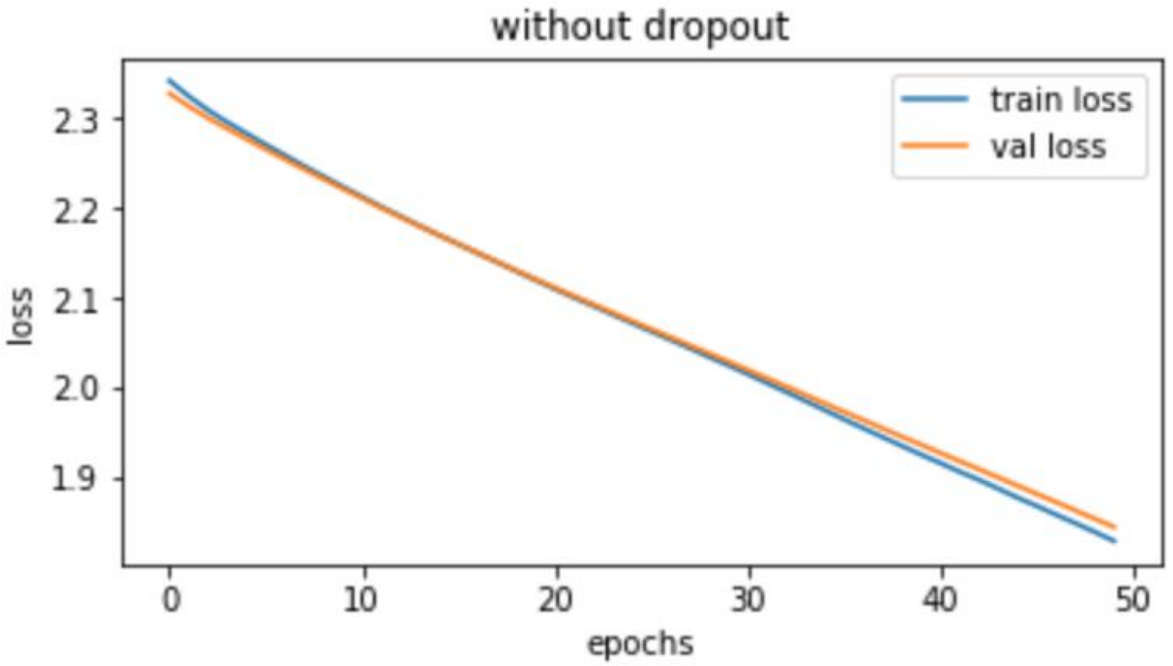
$$A_1 = \frac{A_1}{kp}$$

- Se puede tener un valor de kp distinto para cada capa de la red neuronal.



¿Cómo afecta el Dropout a la función de costo?


with dropout
train loss
val loss
2.35
2.30
2.25
2.20
2.15
loss
0
10
20
30
40
50
epochs


without dropout
train loss
val loss
2.3
2.2
2.1
2.0
1.9
loss
0
10
20
30
40
50
epochs