

Regresión Lineal

Método causal en el que una variable conocida como dependiente está relacionada con una o más variables independientes por medio de una ecuación lineal.

Dependiente	Independiente
Variable que se desea pronosticar (su comportamiento depende de las variables independientes).	Variable que influye en la dependiente y por ende son causa de los resultados obtenidos en el pasado.

- Regresión lineal:

$$\hat{y} = ax + b$$

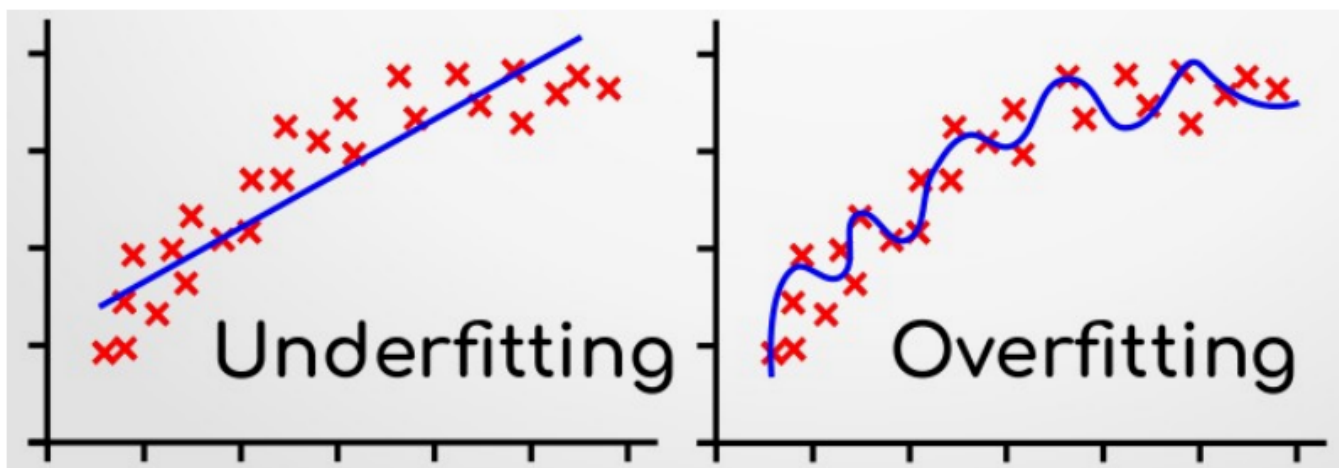
Es la estimación de «y» según el modelo encontrado

Intercepción u ordenada al origen

Pendiente

$$h_{\theta}(x) = \theta_1 x + \theta_0$$

Salida Continua



- **Underfitting:** El modelo de datos no es capaz de capturar de forma precisa la relación entre las variables de entrada y salida, de modo que se genera un alto índice de errores en el conjunto de entrenamiento y en los datos no vistos.
- **Overfitting:** Se ajusta exactamente a los datos de entrenamiento. El modelo no puede hacer predicciones o conclusiones precisas a partir de ningún otro dato que no sea el de entrenamiento.

Regresión Lineal - Algoritmo

Se basa en minimizar el Error Cuadrático Medio (MSE) siguiendo el principio del algoritmo de descenso por el gradiente, el cual busca minimizar la función coste llamada.

$$MSE = J(\theta) = \frac{1}{2n} \sum_{i=1}^n (\hat{y}^{(i)} - y^{(i)})^2$$

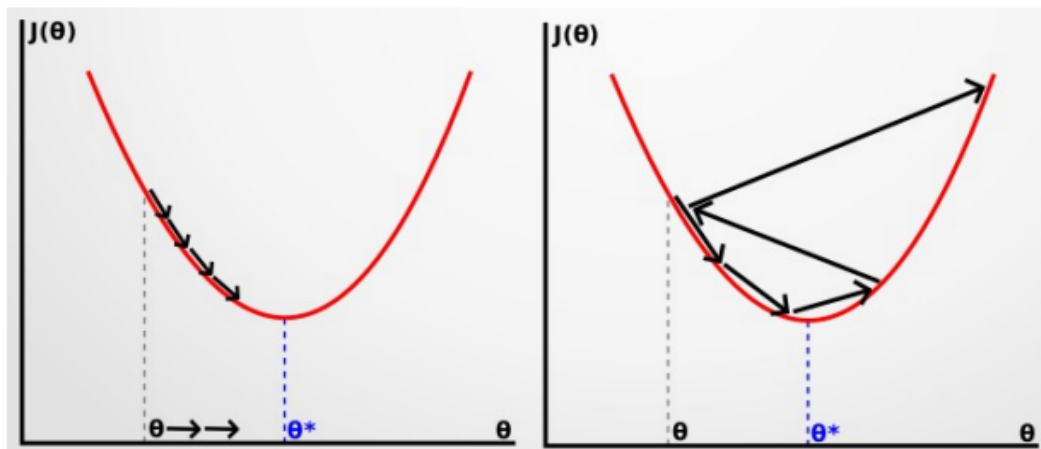
Coste Llamada

$$\min_{\theta} J(\theta) = \min_{\theta} \left[\frac{1}{2n} \sum_{i=1}^n (h_{\theta}(x)^{(i)} - y^{(i)})^2 \right]$$

Repetir hasta converger

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

- α pequeña: desciende lento, tarda más pero es preciso.
- α grande: desciende rápido, tarda poco pero se podría pasar incluso divergir.



Mínimo alcanzado si

$$\frac{\partial}{\partial \theta_j} J(\theta) = 0$$

Derivadas Parciales

$$j=0 \rightarrow \frac{\partial}{\partial \theta_0} J(\theta) = \frac{1}{n} \sum_{i=1}^n [h_{\theta}(x^{(i)}) - y^{(i)}]$$

$$j=1 \rightarrow \frac{\partial}{\partial \theta_1} J(\theta) = \frac{1}{n} \sum_{i=1}^n [h_{\theta}(x^{(i)}) - y^{(i)}] \cdot x^{(i)}$$

$$\theta_0 \leftarrow \theta_0 - \alpha \cdot \frac{\partial}{\partial \theta_0} J(\theta)$$

$$\theta_1 \leftarrow \theta_1 - \alpha \cdot \frac{\partial}{\partial \theta_1} J(\theta)$$

Coeficiente de Correlación

Es la medida que indica el nivel de asociación entre las variables dependiente e independiente.

$$R = \frac{\sum \hat{y}^{(i)} - \bar{y}}{\sum y^{(i)} - \bar{y}}$$

- $R = 0$: Son variables independientes.
- $R = 1$: Relación lineal exacta.
- $R > 0$: Relación directa. Si aumenta X también lo hace Y.
- $R < 0$: Relación inversa: Si aumenta X disminuye Y.

Coeficiente de Determinación

Es una medida que indica porcentualmente el cambio de la variable dependiente respecto a la independiente. Da valores entre 0 y 1. Mientras más cerca de 1 mejor.

$$R^2 = \frac{\sum (\hat{y}^{(i)} - \bar{y})^2}{\sum (y^{(i)} - \bar{y})^2} \times 100$$

Es la estimación de «y» según el modelo encontrado

Regresión Lineal - Medidas a tener en cuenta

Error Absoluto Medio (EAM)

Es un promedio de la suma de los errores de los valores predichos ante los valores realmente observados.

$$EAM = \frac{\sum |\hat{y}^{(i)} - y^{(i)}|}{n}$$

Raíz del Error Cuadrático Medio (RECM)

Es la raíz cuadrada del promedio de la suma de los errores de los valores predichos ante los valores realmente observados elevados al cuadrado. Un valor igual a cero significa un ajuste perfecto.

$$R E C M = \sqrt{\frac{\sum(\hat{y}^{(i)} - y^{(i)})^2}{n}}$$

Error Relativo Absoluto (ERA)

Es un porcentaje que relaciona las diferencias de la sumatoria de los errores predichos y los valores reales con respecto a la sumatoria de la diferencia entre la media y los valores realmente observados.

$$E R A = \frac{\sum|\hat{y}^{(i)} - y^{(i)}|}{\sum|\bar{y} - y^{(i)}|} \times 100$$

Raíz del Error Cuadrático Relativo (RECR)

Es un porcentaje que se calcula tomando la raíz cuadrada que relaciona las diferencias de la sumatoria de los errores predichos y los valores reales con respecto a la sumatoria de la diferencia entre la media y los valores realmente observados.

$$R E C R = \sqrt{\frac{\sum(\hat{y}^{(i)} - y^{(i)})^2}{\sum(\bar{y} - y^{(i)})^2}} \times 100$$

Suma de Cuadrados de Error

Es la parte de la variabilidad de la variable dependiente que no conseguimos explicar con el modelo.

$$S_r = \sum_{i=1}^n (\hat{y}^{(i)} - y^{(i)})^2$$

Error Estándar

Es el valor que cuantifica cuanto se apartan los valores de la media de la población.

$$\hat{\sigma} = \sqrt{S_r \cdot \frac{1}{n - 1}}$$

Ejemplo: Pág. 33

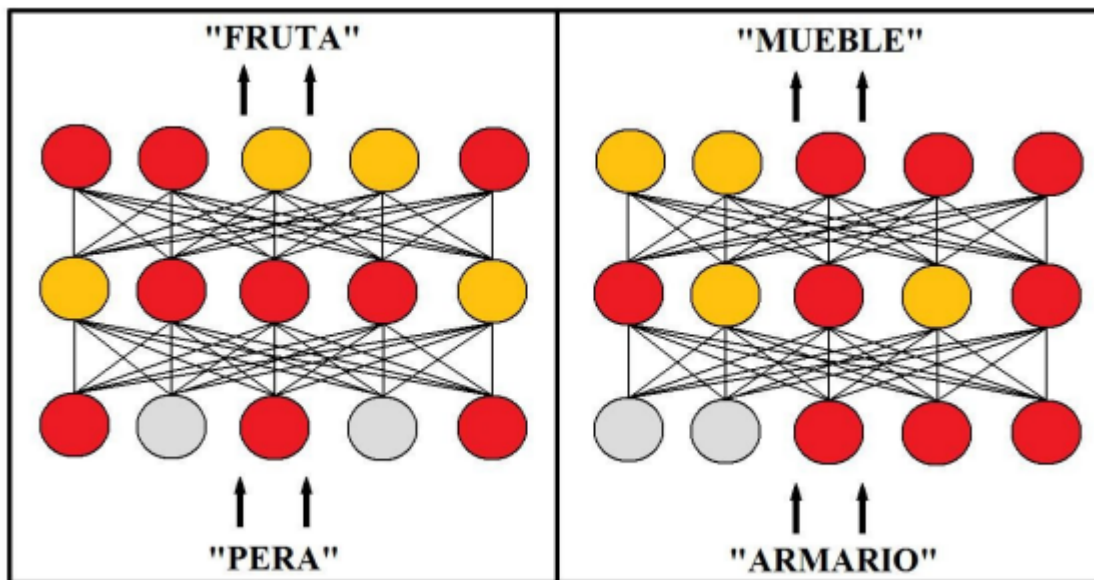
#5ta-clase

Mecanismos de Redes Neuronales

Modelo Conexionista o Redes Neuronales Artificiales

Esta rama de la IA toman como modelo a la mente humana, intentando simular por medio de una computadora el funcionamiento de nuestro cerebro.

Inspiración: Neurona y sus conexiones.



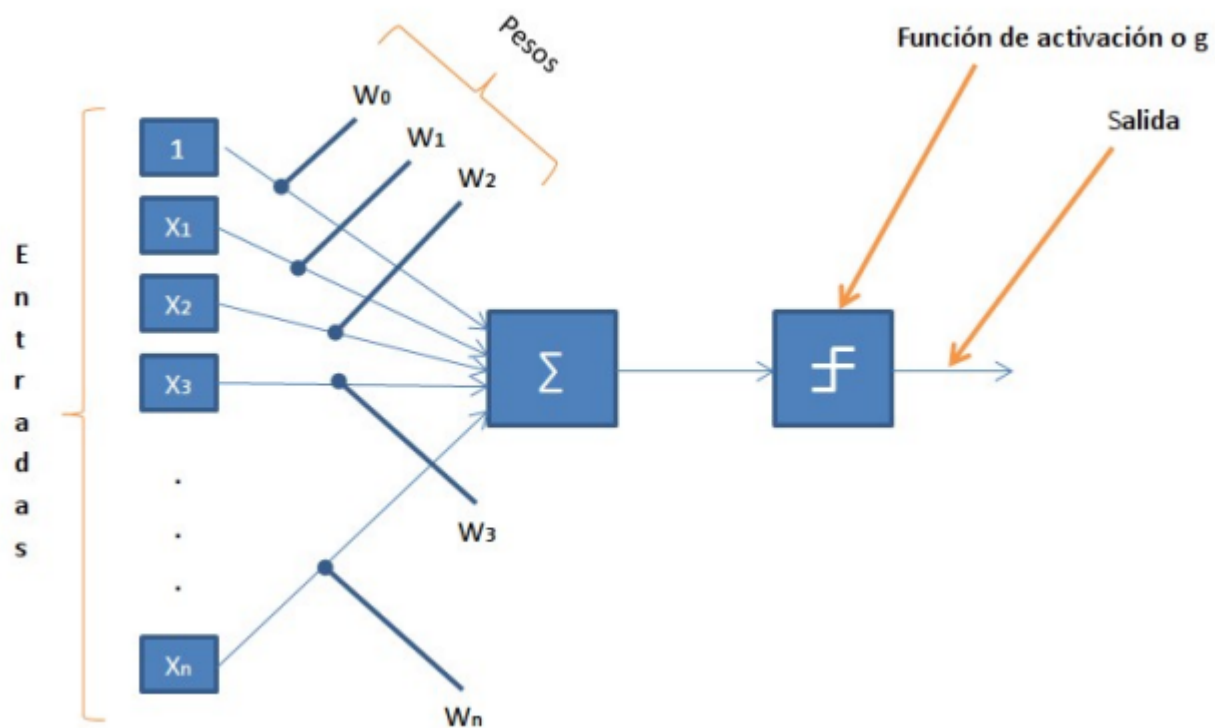
Elementos de una Red Neuronal Artificial

- Gran número de elementos muy simples que procesan de modo similar a las neuronas.
- Un gran número de conexiones con "pesos" entre los elementos (los pesos de las conexiones codifican el conocimiento de la red).

Tipos de Neuronas

De entrada	Ocultas	De salida
Reciben los estímulos externos (relacionada con el aparato sensorial).	Generan el procesado y la representación interna de la información.	Estas unidades dan la respuesta del sistema.

Neurona Artificial (Perceptrón)



(j = neurona observada; i = neurona conectada)

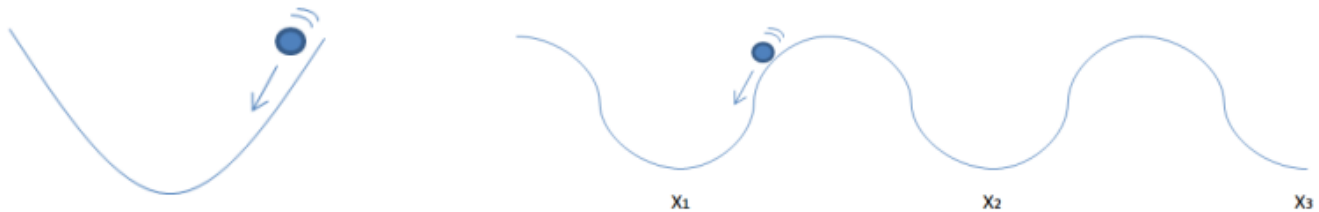
- Posee entradas que permite propagar la activación de la unidad i a la j.
- Cada conexión tiene un peso numérico asociado, que determina la fuerza y el signo de la conexión (Si $W_{ij} > 0$ -> Excitadora; $W_{ij} < 0$ -> Inhibidora; $W_{ij} = 0$ -> No hay conexión).
- Cada unidad i primero calcula la suma ponderada de sus entradas (con i = 0 hasta n).
- Luego se aplica la función de activación g a la suma para lograr la salida (con i = 0 hasta n).
- La función de activación habilita (cerca de +1) la unidad cuando se dan las entradas adecuadas y la desactiva en caso contrario.

Redes Monocapa

Modelo de Hopfield

- Memoria asociativa.

- Analogía física de la memoria:



Características

- Red neuronal monocapa.
- Suele manejar valores de $[1, -1]$.
- La salida de cada nodo se conecta a todos los demás.
- Conexiones multidireccionales.
- No se permiten conexiones de cada neurona consigo misma.
- Siempre converge a la solución de uno de los patrones aprendidos.
- Usa función de activación del tipo escalón para el modelo discreto.
- Utiliza un mecanismo de aprendizaje no supervisado del tipo Hebbiano: $\Delta w_{ij} = y_i \cdot y_j$.
- Muy buena para reconocer patrones.
- Se estima empíricamente que cada 7 neuronas se puede almacenar un patrón. Cantidad de patrones: $0,14 \cdot N$ (donde N es la cantidad de neuronas).

Funcionamiento

- Consiste en memorizar varios patrones como si de una memoria se tratase.
- Si se presenta a la entrada alguna de las informaciones almacenadas, la red evoluciona hasta estabilizarse.
- La red puede evolucionar a la salida más parecida pese a poseer entradas parciales o incompletas.
- La información de entrada debe ser codificada como vector.
- Cada neurona recibe parte de la información, es decir, un elemento del vector de entrada.

Aprendizaje

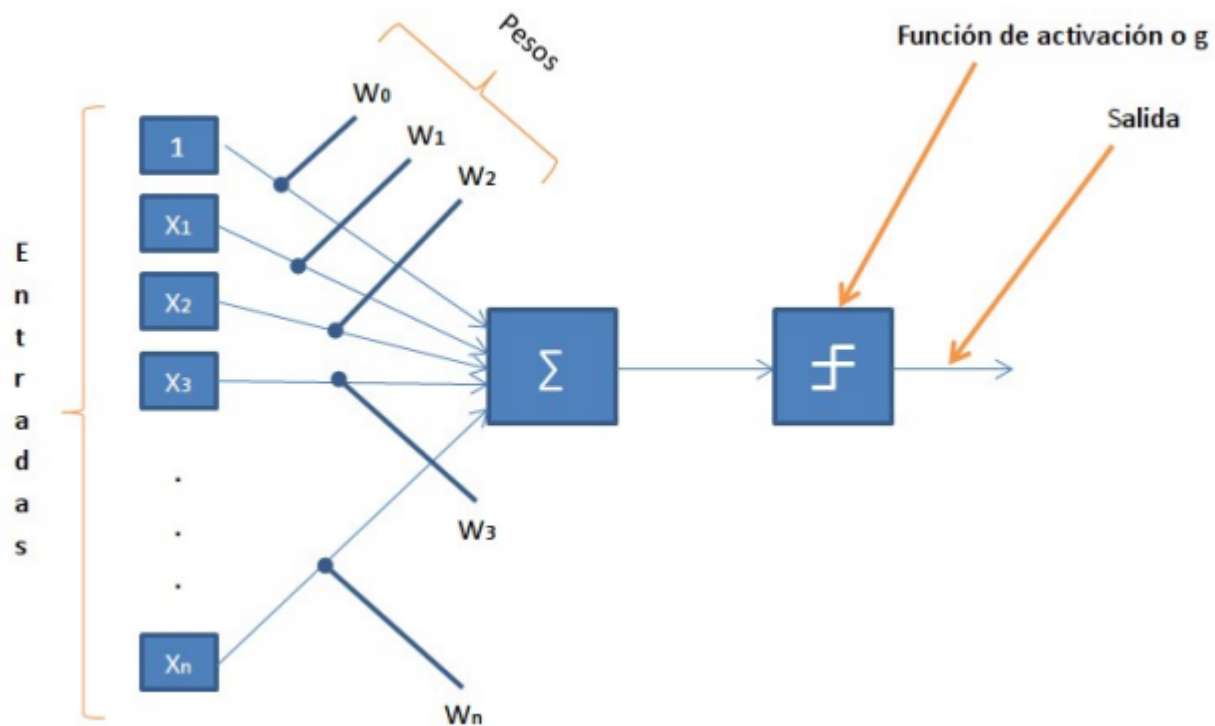
- Es no supervisado o offline.
- La red calculará la matriz de pesos W con un conjunto de patrones de ejemplo (que se pretende que memorice) y luego aplicará lo aprendido durante el funcionamiento para poder reconocer alguno de esos patrones.
- Una vez establecidos estos pesos la red entra en funcionamiento.

$$W = \sum_{i=1}^n (E_i x E_i^T) - I$$

- Cálculo de la matriz: (se resta la identidad porque no pueden conectarse consigo mismas).

#6ta-clase

Perceptrón (Rosenblatt)



Características

- Fue uno de los primeros modelos neuronales.
- Imita a la neurona tomando la suma ponderada de sus entradas y enviando a la salida un 1 si la suma es más grande que algún valor umbral ajustable (si ocurre de otro modo devuelve 0).
- Poseen conexiones unidireccionales.
- Ideal para problemas linealmente separables.

Modo de Cálculo

Función Suma Ponderada

$$g(x) = w_0 + w_1 \cdot x_1 + w_2 \cdot x_2$$

Función Salida

Unidaria.

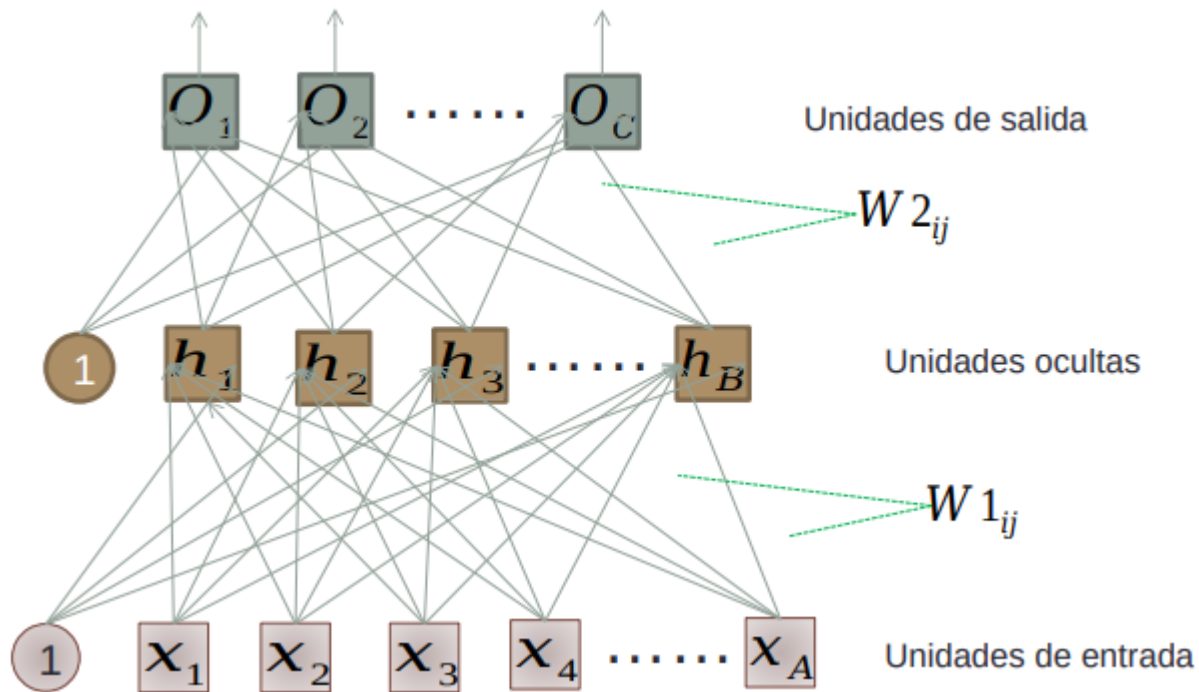
$$o(x) = \begin{cases} 0 & \text{si } g(x) < 0 \\ 1 & \text{si } g(x) > 0 \end{cases}$$

Ecuación de la Recta

$$g(x) = w_0 + w_1 \cdot x_1 + w_2 \cdot x_2$$
$$x_2 = -\frac{w_1}{w_2} \cdot x_1 - \frac{w_0}{w_2} \quad \left. \vphantom{x_2} \right\} \text{ Recta / superficie de decisión}$$

Redes Multicapa

Red Backpropagation



Características

- W_{1ij} : Pesos de las conexiones entre las capas de entrada y oculta.
- W_{2ij} : Pesos de las conexiones entre las capas de salida y oculta.
- Las activaciones "saltan" desde la capa de entrada hacia la oculta y de esta hacia la de salida.
- El conocimiento de la red está codificado en los pesos de las conexiones entre unidades.

- Los niveles de activación en la capa de salida determinan la salida de la red.

Función de Activación

- Aun hay una unidad de propagación hacia atrás que suma sus entradas ponderadas.
- Produce un valor entre [0, 1] basado en la función sigmoide continua y diferenciable.

$$salida = \frac{1}{1 + e^{-suma}}$$

- La ecuación de dicha función es:
- Si suma = 0 -> la salida se hace 0,5.
- A medida que la suma aumenta, la salida se aproxima a 1.
- A medida que la suma disminuye, la salida se aproxima a 0.

Funcionamiento Simplificado del Algoritmo

- Comienza con un conjunto de pesos aleatorios.
- La red ajusta sus pesos cada vez que ve un par entrada/salida.
- Cada par requiere dos etapas:
 - Paso hacia adelante: Se presenta un ejemplo de entrada y permite que las activaciones continúen hasta que alcancen la capa de salida.
 - Paso hacia atrás:
 - La salida que se tiene de la red en ese momento se compara con la salida objetivo y se calcula el error estimado de las unidades de salida.
 - Se ajustan los pesos asociados a las unidades de salida para reducir dichos errores.
 - Se deriva el error estimado de las capas de salida a las ocultas.
 - Por último, los errores se propagan hacia atrás a las conexiones procedentes de las unidades de entrada.

Calculo de Nodos Ocultos

La cantidad de neuronas que debe contener cada capa oculta es difícil de determinar ya que no existe una regla que permita calcularlas y muchas veces se hace de manera empírica.

Una idea es mantener el número de neuronas por capa lo más bajo posible ya que cada neurona es un costo de procesamiento extra.

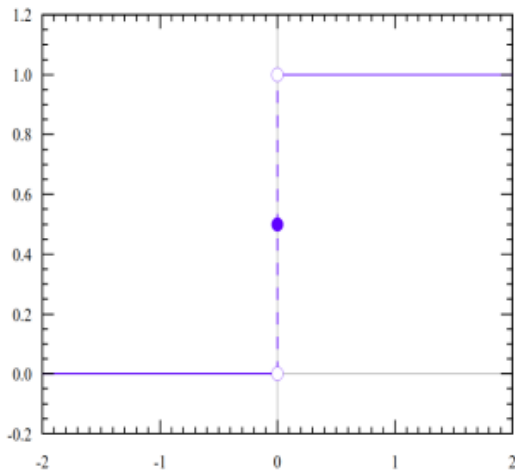
Regla Baum-Haussler

$$N_{oculta} \leq \frac{N_{entren} \cdot E_{tolerable}}{N_{entrada} \cdot N_{salida}}$$

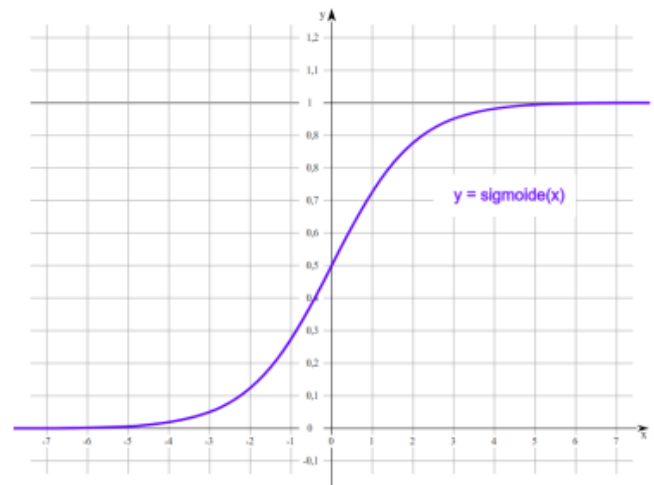
- N_{entren} = Nro patrones entrenamiento
- $E_{tolerable}$ = Error deseado de la red
- $N_{entrada}$ = Nro de nodos de entrada
- N_{salida} = Nro de nodos de salida

Perceptrón vs Backpropagation

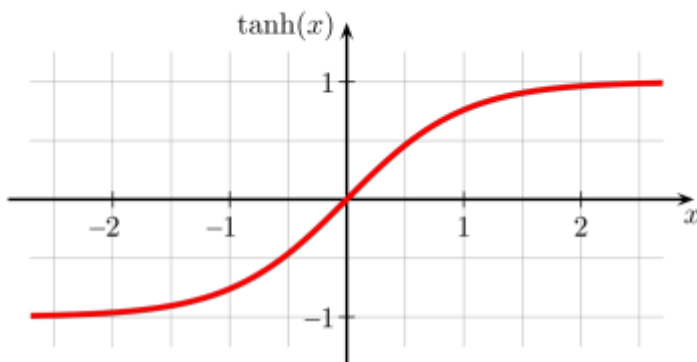
Función de activación: (escalón)
Perceptrón



Función de activación: (sigmoide)
Backpropagation

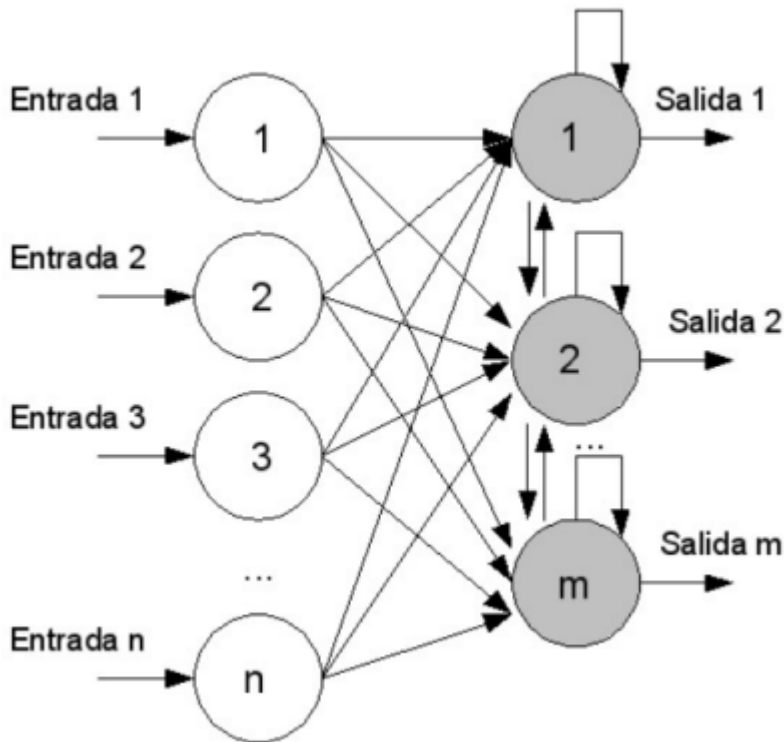


Función de activación: (Tangente hiperbólica)
Backpropagation



Redes Competitivas

Red Kohonen



Características

- Basadas en el funcionamiento del cerebro que indica que hay neuronas que se organizan en zonas (mapas bidimensionales).
- Una gran parte de esta organización es genéticamente heredada pero gran parte de ella se origina mediante el aprendizaje.
- El modelo de Kohonen tiene la capacidad de formar mapas de características de manera similar a como ocurre en el cerebro humano.
- Se aplica al reconocimiento de voz y al texto manuscrito.
- El modelo tiene 2 variantes:
 - LVQ (Learning Vector Quantification): Para vectores de entrada de una sola dimensión.
 - TPM (Topology Preserving Map) o SOM (Self Organization Map): Para vectores de entrada bidimensionales o incluso tridimensionales en TPM.

Estructura

- Es una red de 2 capas con N neuronas de entrada y M de salida.
- Cada una de las N entradas se conecta con las M salidas a través de conexiones feedforward.

- Entre las neuronas de salida existen conexiones laterales de inhibición (peso negativo) indicando que cada neurona va a tener cierta influencia sobre sus vecinas.
- El valor que se le vaya a asignar a los pesos de las conexiones feedforward entre la capa de entrada y salida (W_{ij}), durante el aprendizaje, dependerá de dichas interacciones laterales.
- Influencia de cada neurona de salida sobre las demás:



Funcionamiento Simplificado

- Se presenta una entrada: $E_k = (e_1^{(k)} e_2^{(k)} \dots e_N^{(k)})$
- Cada una de las neuronas de la capa de salida recibe una parte de esa entrada por medio de las conexiones feedforward con pesos W_{ij} .
- También cada neurona de salida recibe las entradas de las interacciones laterales.
- Salida generada por una neurona de salida S_j ante un vector de entrada E_k :

$$S_j(t+1) = f(\sum_{i=1}^N w_{ij} e_i^{(k)} + \sum_{p=1}^M Int_{pj} S_p(t))$$

- Int_{pj} es una función de tipo sombrero mejicano que representa la influencia lateral de la neurona p sobre la neurona j .
- La función de activación de las neuronas de salida será de tipo continua, lineal o sigmoideal.
- Esta red trabaja con valores reales.
- La red evoluciona ante la entrada E_k hasta alcanzar un estado estable en el que solo hay una neurona activa (la ganadora).

$$S_j = \begin{cases} 1 & \rightarrow \text{MIN } \|E_k - W_j\| = \text{MIN}(\sqrt{\sum_{i=1}^N (e_i^{(k)} - w_{ij})^2}) \\ 0 & \rightarrow \text{rest} \end{cases}$$

- Funcionamiento simplificado:

- $\|E_k - W_j\|$ es una medida de la diferencia entre el vector de entrada y el vector de pesos de las conexiones que llegan a la neurona j desde la entrada.
- Objetivo: Se pretende encontrar el dato aprendido más parecido al de entrada para averiguar que neurona se activará.

Algoritmo de Aprendizaje

- Es de tipo off line.
- No supervisado de tipo competitivo.
- Las neuronas de la capa de salida compiten por activarse.
- Solo una de ellas permanecerá activa ante una determinada entrada.
- Pasos del algoritmo:
 - Inicializar los pesos W_{ij} con valores aleatorios pequeños y se fija la zona inicial de vecindad entre las neuronas de salida.
 - Presentar a la red una entrada cuyas componentes son valores continuos.

$$E_k = (e_1^{(k)} e_2^{(k)} \dots e_N^{(k)})$$

- Determinar la neurona vencedora a la salida. Esta será aquella j cuyo vector de pesos W_j sea el más parecido a la información de entrada E_k . Para ello se debe calcular las distancias entre ambos vectores, una para cada neurona de salida:

Donde: • $e_i^{(k)}$: Componente i -ésimo del vector k -ésimo de entrada.

• w_{ij} : Peso de la conexión entre las neuronas i (de entrada) y j (de salida).

- Una vez localizada la neurona vencedora j^* se actualizan los pesos de las conexiones feedforward que llegan a dicha neurona y a sus vecinas:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(t)[e_i^{(k)} - w_{ij^*}(t)] \quad j \in \text{Zona}_{j^*}(t)$$

Donde:

• $\text{Zona}_{j^*}(t)$ es la zona de vecindad de la neurona vencedora j^*

• $\alpha(t)$ es el coeficiente de aprendizaje entre 0 y 1 el cual decrece con cada iteración:

• Suele tener la siguiente expresión: $\alpha(t) = \frac{1}{t}$

- El proceso debe repetirse hasta lograr que los pesos se estabilicen y tiendan a un cierto valor de error (pequeño) o por lo menos iterar un mínimo de 500 veces.

#7ma-clase

Algoritmos Genéticos y Computación Evolutiva

Algoritmo Genético

¿Qué son?

Un sistema biológico eficiente desarrolla estrategias exitosas de adaptación para lograr la supervivencia.

Los AG tratan de simular la evolución de una población de individuos, mediante un proceso iterativo aplicado sobre un conjunto de estructuras.

Cada estructura está compuesta de características que definen la **aptitud** del individuo en su entorno.

Objetivo: Dependiendo el problema planteado, encontrar el máximo valor o el más cercano al valor máximo. Con f que es la función de aptitud asignada a cada individuo de la población.

Expresiones Genéticas

- **Cromosoma:** Individuo o estructura.
- **Gen:** Características o atributos.
- **Alelo:** Valor en una posición determinada.
- **Locus:** Posición en la estructura.
- **Genotipo:** Conjunto de genes o estructuras. (potencial solución)
- **Fenotipo:** Aptitud del individuo.

Características

- Se basa en el principio de la selección natural.
- Los individuos que mejor se adapten deben sobrevivir.
- En cada generación del AG debe heredarse la mejor configuración genética posible.
- Los rasgos hereditarios que favorecen el éxito deben estar más representados en la población.

Selección Natural de Darwin

1. Los individuos de la población tienen diferencias. (evitar la endogamia)
2. Las variaciones pueden heredarse.
3. Los organismos pueden tener más descendientes de los que pueden sobrevivir con los recursos disponibles.
4. Las variaciones que aumentan el éxito reproductivo tienen mayor oportunidad de transmitirse.

Aplicaciones

- Pueden resolver problemas complejos de forma rápida y eficiente.
- Utilizan muy poca información específica del problema.
- Son extensibles ya que es fácil modificar o incorporar conocimiento en los operadores genéticos.

- Muy buenos para problemas de optimización. (ej: optimización del tamaño de enlaces en una red, procesamiento de imágenes, reconocimiento de patrones)

Algoritmo Simple

- Generar población inicial creando a sus individuos aleatoriamente.
- Luego, cada generación será creada a partir de la anterior tras aplicar 3 operadores básicos:
 - Selección.
 - Cruza.
 - Mutación.
- El proceso iterativo prosigue hasta que se cumpla con alguna condición de corte.

nota: Todo individuo seleccionado se reemplaza por su sucesor después del cruzamiento y mutación. Los individuos no seleccionados mueren inmediatamente.

Métodos de Selección

Proporcionales

Eligen individuos teniendo en cuenta su aptitud en relación con el resto de la población.

Basados en el Orden

Confeccionan una tabla ordenada de los individuos en base a su aptitud.

Ruleta

A cada individuo se le asigna un tamaño de ranura proporcional a su aptitud.

Control del Número Esperado

Se define un valor C_i que determina la cantidad de copias a asignar a cada individuo.

$$C_i = \frac{f_i}{\bar{f}}$$

nota: Con Control del número esperado, es determinante el redondeo hacia arriba. Si los decimales son > 0.5 -> suma 1.

Nº	Individuo	Aptitud	C _i	Copias esperadas	Copias asignadas
1	XYXY	169	0,58	≥ 0	0 + 1
2	YYXX	576	1,97	≥ 1	1 + 1
3	XYXX	64	0,22	≥ 0	0 + 0
4	YXXY	361	1,23	≥ 1	1 + 0
Total		1170	4		4

Elitista

Preserva los mejores m individuos de la generación actual, incluyéndolos directamente en la siguiente.

nota: Con Elitista, siempre se usa en conjunto con otras variantes. (endogamia)

Por ranking

Cada individuo recibe una cantidad de copias que sólo depende de su ubicación en la tabla. Para esto, se ordena en forma descendente por la aptitud de cada individuo.

Cantidad de copias para el individuo en la posición i:

$$R_{min} + 2 \frac{(n-i)(1-R_{min})}{(n-1)}$$

- **R_min** = Número de copias que recibe el **PEOR** individuo.
- **(n - i)** = Inversión de la posición. (valor alto -> mejor)
- **(1 - R_min)** = Rango total disponible para distribuir.
- **(n - 1)** = Normalización para el tamaño de población.

Cantidad de copias

- Mejor individuo: 2-R_min.
- Peor individuo: R_min.

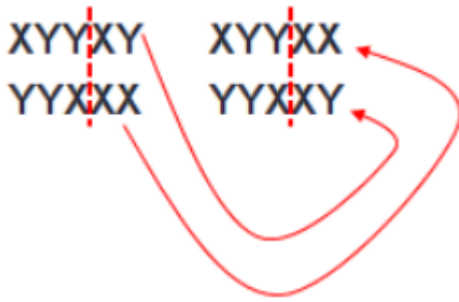
Restricciones para R_min

- 0 < R_min: El peor debe tener alguna chance.
- R_min < 1: si no, no habría diferencia.

Métodos de Cruza

Cruza Simple

Elige al azar uno de los posibles puntos de cruce.



Cruza Multipunto

En este caso el cromosoma es considerado un anillo y se eligen n puntos de cruce en forma aleatoria. La cruce simple es un caso especial con $n = 1$.

Cruza Binomial

Para generar un cromosoma hijo se define la probabilidad P_0 como la probabilidad de que el Alelo de cualquier posición del descendiente se herede del padre y $1-P_0$ como la probabilidad de que lo herede de la madre.

Métodos de Mutación

La mutación permite mantener la diversidad en la población disminuyendo el riesgo de convergencia prematura.

Mutación Simple

Se elige de forma aleatoria un gen y se lo muta con cierta probabilidad muy baja la cual se mantiene constante durante sucesivas generaciones.

Mutación Adaptativa por Convergencia

La probabilidad de mutación varía en base a la información proveniente de la búsqueda genética. Es decir, que la probabilidad de mutación aumenta cuando la población es muy homogénea y disminuye cuando hay demasiada diversidad.

Mutación Adaptativa por Temperatura

Este tipo no utiliza información genética de la población.

La probabilidad de mutación $P_m = P_m(t)$ y esta acotado por un máximo y un mínimo: $P_m^{\min} \geq P_m \geq P_m^{\max}$ partiendo de un valor inicial $P_m(0)$ la actualización se realiza hasta alcanzar un valor final de la siguiente manera:

$$P_m(t+1) = P_m(t) + \lambda$$

Dependiendo del signo de λ tenemos:

- Mutación adaptativa por temperatura ascendente
- Mutación adaptativa por temperatura descendente

Mutación Adaptativa por Temperatura Ascendente

La probabilidad de mutación aumenta con las sucesivas generaciones hasta alcanzar un valor máximo y a partir de allí debe mantenerse constante.

Objetivo: Mantener la diversidad de la población, que por lo general con el pasar del tiempo tiende a volverse homogénea.

Máximo: El aumento de la mutación debe tener una cota máxima para que ocurra la supervivencia de buenos individuos porque sino se corre el riesgo de que la población nunca converja.

$$P_m(t+1) = \begin{cases} P_m(t) + |\lambda| & \text{si } P_m < P_m^{\max} \\ P_m^{\max} & \text{en caso contrario} \end{cases}$$

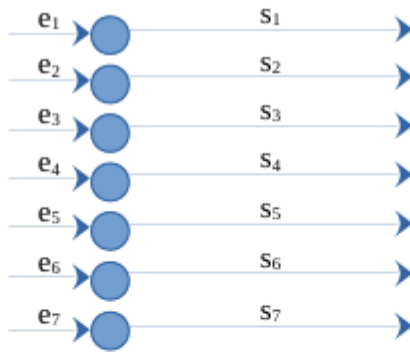
Mutación Adaptativa por Temperatura Descendente

La probabilidad de mutación disminuye con las sucesivas generaciones hasta alcanzar un valor mínimo y a partir de allí debe mantenerse constante.

Objetivo: Permitir una exploración alta en las primeras generaciones evitando la convergencia prematura.

Mínimo: Siempre debe ser mayor a 0, puesto que sin mutación se corre el riesgo de perder estructuras interesantes.

Respuestas Parcial 2



- a) ¿El grafico de la red es correcto?. Explique.
- b) ¿Cuántos patrones aproximadamente puede almacenar una red de Hopfield con la misma cantidad de nodos que la red del grafico? Explique.
- c) ¿Qué significa, en el algoritmo de aprendizaje de Hopfield, que la matriz de pesos W tenga todos los elementos de la diagonal principal igual a cero?

a. No es correcto. En la red de Hopfield todas las neuronas están conectadas con todas las demás, cosa que no pasa en el gráfico.

b. La red puede almacenar 1 patrón. Debido a que empíricamente se dice que cada 7 neuronas, la red puede almacenar 1 patrón, mientras que la regla general nos dice que la cantidad está dada por $0,14 \cdot N$ (con N cantidad de neuronas).

c. Eso representa una característica importante de la red. Donde no se permite la conexión de una neurona consigo misma, es por eso que en las posiciones de la matriz donde $i = j$ tiene que ser siempre 0.

Al aplicar un algoritmo de regresión lineal sobre un conjunto de datos, explique lo siguiente:

- a) ¿Cómo debe ser el dataset?
- b) ¿Cuál función se busca minimizar? Explique.
- c) ¿Cómo debe ser el coeficiente de determinación?

a. El dataset debe tener:

- Relación lineal entre variables.
- Suficientes datos (mínimo 30 observaciones).
- Variables continuas.
- Poca multicolinealidad.

b. Busca minimizar el Error Cuadrático Medio (MSE) siguiendo el principio del descenso por el gradiente, el cual busca minimizar la función coste llamada. Calculando las derivadas parciales

en cada punto hasta conseguir encontrar el mínimo valor de la función.

$$MSE = J(\theta) = \frac{1}{2n} \sum_{i=1}^n (\hat{y}^{(i)} - y^{(i)})^2$$

c. El coeficiente de determinación es el porcentaje de cambio que sufre la variable dependiente respecto a la independiente. Da valores entre 0 y 1. Mientras más cerca a 1 mejor.

$$R^2 = \frac{\sum(\hat{y}^{(i)} - \bar{y})^2}{\sum(y^{(i)} - \bar{y})^2} \times 100$$

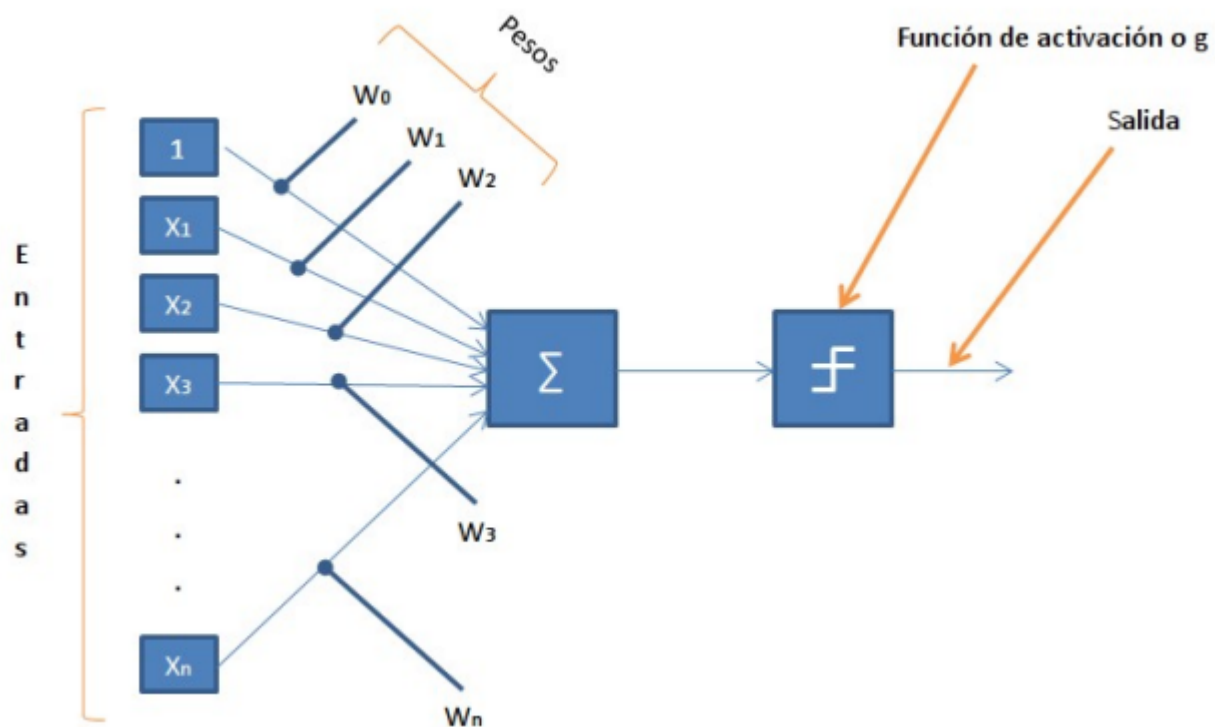
Es la estimación de «y» según el modelo encontrado

- Explique las partes principales (de la neurona biológica) y describa brevemente su funcionamiento.
- Construya la neurona artificial equivalente. Describa cada una de sus partes y funcionamiento.

a. Partes principales:

- **Dendritas:** Reciben señales.
- **Soma:** Procesa las señales.
- **Axón:** Transmite el impulso.
- **Terminales sinápticas:** Liberan neurotransmisores.

b.



Dado el siguiente dataset:

X_1	X_2	X_3	Y_s
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

- a) ¿Es posible de resolver la tabla de datos mediante un Perceptrón simple? ¿Por qué?
- b) De ser posible de resolver mediante un Perceptrón grafique como sería la topología del mismo.

a. **NO** es posible resolver con un Perceptrón simple porque la función AND de 3 variables **no es linealmente separable**. El único caso con salida 1 es cuando todas las entradas son 1.

b. No se puede graficar un Perceptrón simple que resuelva esto, ya que **no existe** un hiperplano que separe perfectamente las clases en 3 dimensiones.

Explique con sus palabras y describa:

- a) El algoritmo de Backpropagation.
- b) ¿Cuales son las principales características de un Perceptrón multicapa que es entrenado mediante Backpropagation?
- c) ¿Cuales funciones de activación suelen usarse? ¿Por qué?

a. Algoritmo de backpropagation:

- Arranca con pesos aleatorios.
- La red ajusta los pesos cada vez que ve un par entrada/salida.
- Cada par requiere 2 etapas:
 - Paso adelante: Cuando hay una entrada se procesa a lo largo de la red por medio de las activaciones hasta llegar a la capa de salida.
 - Paso atrás: Una vez alcanzada la salida, se la compara con la salida objetivo y se calcula el error estimado, se ajustan los pesos para reducir los errores. El error estimado se deriva a las capas ocultas y por último, los errores se propagan a las conexiones de las unidades de entrada.

b. Características:

- El conocimiento de la red esta codificado en los pesos.
- Múltiples capas ocultas.

- No lineal. Resuelve problemas no lineales.
- Entrenamiento supervisado.

c. La sigmoide (salidas probabilísticas) y la tangente hiperbólica (salidas entre -1 y 1). Porque son continuas y diferenciables.

Respuestas Recuperatorio 2

Explique lo siguiente sobre una red de Hopfield:

a) ¿Como sería el grafico de la red si se desean almacenar 1,5 patrones? Explique.

b) ¿Qué significa, en el algoritmo de aprendizaje de Hopfield, la matriz de pesos W?

a. Por formula tenemos que Cant. patrones = $0,14 N \Rightarrow 1,5 = 0,14 N \Rightarrow 1,5/0,14 = N \Rightarrow 10,7$ neuronas para almacenar 1,5 patrones.

nota: No se podría almacenar porque debe ser un nro entero. Pero de ser así tendríamos que tener 11 neuronas en nuestra red para cubrir esa cantidad.

b. La matriz represente las conexiones sinápticas entre las neuronas, determinan la fuerza de esas conexiones.

Al aplicar un algoritmo de regresión lineal sobre un conjunto de datos, explique lo siguiente:

a) ¿Como debe ser el dataset?

b) ¿Cual función se busca minimizar? Explique.

c) ¿Como debe ser el coeficiente de correlación?

c. El coeficiente de correlación es un valor que representa el nivel de asociación entre las variables dependiente e independiente.

- $R = 0$: Son variables independientes.
- $R = 1$: Relación lineal exacta.
- $R > 0$: Relación directa. Si aumenta X también lo hace Y.
- $R < 0$: Relación inversa: Si aumenta X disminuye Y.

Explique lo siguiente sobre una red Perceptrón:

a) Describa brevemente el funcionamiento del algoritmo de entrenamiento del mismo.

b) Describa cada una de sus partes.

c) ¿Ante cuales problemas falla el Perceptrón? ¿Por qué?

a.

1. Inicializar pesos aleatorios.

2. Para cada patrón de entrada, calcular salida.
3. Si hay error, ajustar pesos: $\Delta w = \eta (d - y) x$.
4. Repetir hasta convergencia.

c. Es una red ideal para problemas linealmente separables. Cuando se presenta uno que no lo sea, la red falla (XOR).

Explique con sus palabras y describa:

- a) El algoritmo de Kohonen.
- b) ¿Cuáles son las principales características de una red de Kohonen?
- c) ¿Qué función cumple dentro de dicha red la onduleta tipo sombrero Mejicano (u ondícula de Ricker)?

a. Redes competitivas.

- Se inician con pesos aleatorios pequeños y se fija en la zona inicial de la vecindad entre las salidas.
- Se presenta una entrada cuyos componentes son valores continuos.
- Determina cual es la neurona vencedora de la salida. Será aquella j cuyo vector de pesos sea el más parecido a la información de entrada. Para ello se calcula la distancia entre ambos vectores, una para cada neurona de salida.
- Una vez obtenida la ganadora, se actualizan los pesos de las conexiones feedforward que llegan a dicha neurona y a sus vecinas.
- El proceso se repite hasta que los pesos se estabilicen y tiendan a un cierto valor de error (pequeño) o por lo menos iterar 500 veces.

b.

- Divide por vecindades.
- Las neuronas compiten para activarse.
- Es una red bicapa. Con entradas y salidas.
- Conexiones feedforward.
- Entre las neuronas de salida existen conexiones laterales de inhibición.
- Trabaja con valores reales.
- Aprendizaje de tipo no supervisado competitivo.

c. Dicha función representa la influencia de una neurona de salida sobre sus vecinas. Se mide en: Interacción lateral (y) y Distancia entre neuronas (x).

Algoritmos Genéticos y Computación Evolutiva

Cromosomas con aptitud (medible). Se escriben en binario. Es una solución del problema (pueden haber muchos).

Fenotipo: Es agarrar el cromosoma y pasarlo por la función de aptitud.

Algoritmo Simple

- Selección: El criterio es la aptitud.
- Cruza: Se define la altura del corte (aleatorio) y se intercambian los segmentos entre dos individuos.
- Mutación: Se selecciona el individuo aleatoriamente y se toma un gen aleatorio para cambiarlo (si es 0 -> 1 o 1 -> 0).

nota: No es necesario aplicar la mutación.

Cuando termina el algoritmo se selecciona el mejor cromosoma por medio de la función aptitud.