Aprendizaje Automático

Modelo de Kohonen

2023-2Q

Redes de Kohonen

Teuvo Kohonen

Publicación: 1982

Descubre por sí misma regularidades en los datos de entrada

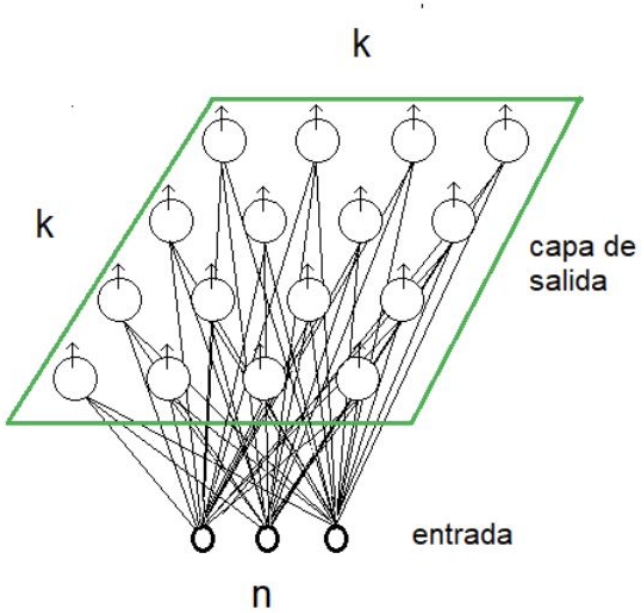
No existe maestro o supervisor externo

No Supervisado

Mapas Auto-Organizados
o
SOM: Self Organized Maps

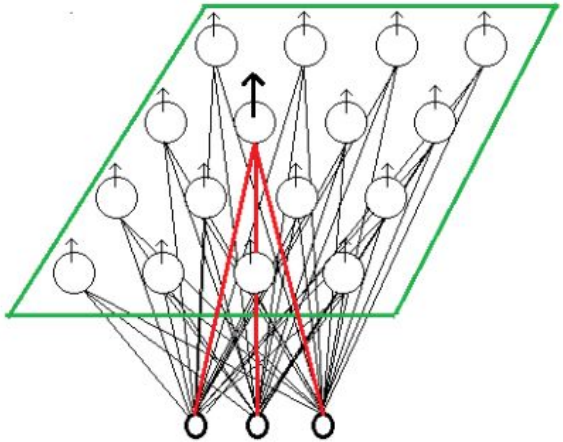
Estructura

- Única capa, llamada capa de salida con forma de grilla bidimensional (k x k)
- Cada neurona conectada con todas las entradas y dichas conexiones se representan con un vector de pesos **w** n-dimensional



Aprendizaje Competitivo

Cuando una entrada es presentada a la red, las neuronas de la capa de salida compiten unas con otras con el objetivo de que finalmente una de las neuronas sea la ganadora, y se active, y las demás sean forzadas a valores de respuesta mínimos.



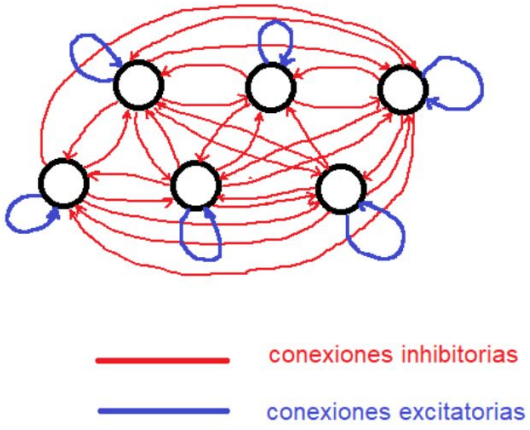
La neurona ganadora en Kohonen, es aquella que tenga el vector de pesos **w** más parecido a la entrada X

→

Entradas parecidas activarán la misma neurona

Conexiones

- Cada neurona tiene tiene conexiones inhibitorias con sus vecinas, y consigo misma una excitatoria, también conocida como retroalimentación.
- Esto produce que a lo largo del entrenamiento algunas unidades de salida tomen niveles de activación mayor mientras el nivel en las de´mas se anula.

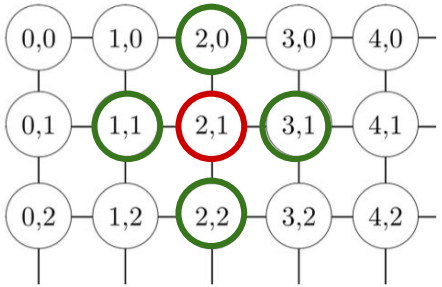


Vecindario

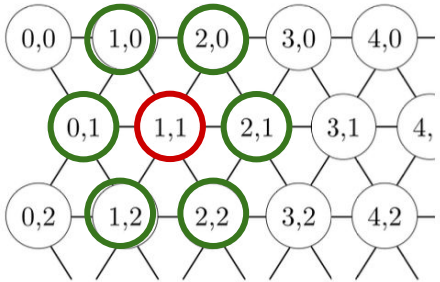
$1 \leq i \leq k$ y $1 \leq j \leq k$

- Si la grilla es de k x k, la neurona (i,j) es aquella en la fila i, columna j.
- Dada una neurona (i , j) se define su **vecindario** como el conjunto de neuronas que están a una distancia menor o igual a R (*radio del vecindario*)
-
- En una grilla hexagonal , si R=1, cada neurona tendrá 4 vecinos.
- Si, en cambio, R = √2 , tendrá 8 vecinos

Grilla rectangular



Grilla hexagonal



Con Radio =1

Entrenamiento

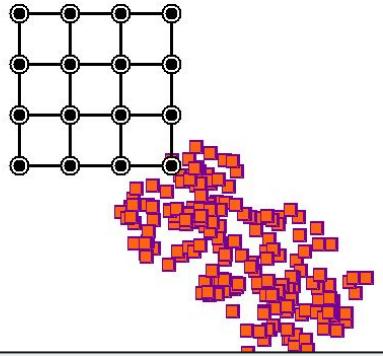
- Sea un conjunto de entrenamiento X que tiene P ejemplos, y cada ejemplo de dimensión n. p = 1, … ,P

$$X = \{x^1, \ldots, x^P\} \quad X^p = (x_1^p, x_2^p, \ldots, x_n^p)$$

- Sea una capa de salida formada por (k x k) neuronas

- Para cada neurona de la capa de salida (i, j), habrá un vector de pesos de dimensión n que representa sus conexiones con la entrada.

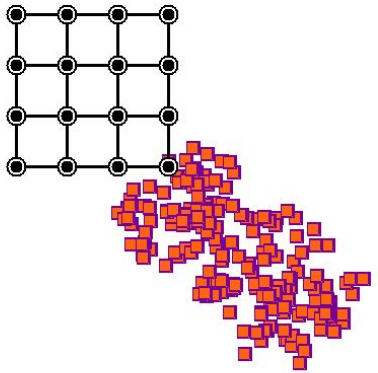
$$w^{i,j} = (w_1^{i,j}, w_2^{i,j}, \ldots, w_n^{i,j}) \qquad i = 1,\ldots,k \;\; j = 1,\ldots,k$$



- Al elegir un ejemplo de entrenamiento Xp, elegiremos a la unidad ganadora según la similitud del vector de pesos **w.**
- Luego, corregiremos el vector de pesos de la neurona ganadora, acercándose más al ejemplo Xp.
- En consecuencia, al presentar nuevamente la misma entrada, la neurona será más parecida

Entrenamiento - Vecinos

- No solo modificaremos los pesos de la ganadora, sino de sus vecinos.
- El radio de vecindad inicializará en un valor relativamente grande a la grilla, e irá decreciendo con el entrenamiento hasta llegar a 0.
- Esto provocará que no solo la ganadora, sino sus vecinas se acerquen al valor de entrada. Las neuronas vecinas se parecerán entre sí que con las demás.
- Este proceso se denomina **ordenamiento**



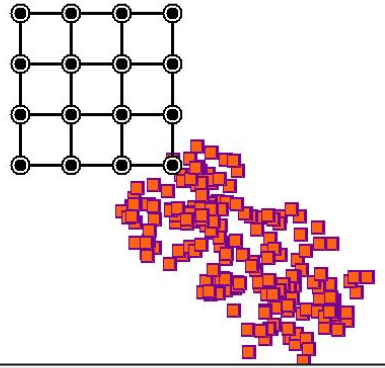
A medida que el radio R va disminuyendo, el ordenamiento va estabilizándose y el entrenamiento será gobernado por la convergencia.

Algoritmo

1. Inicializar los pesos
   Inicializar el radio de vecindad
   Inicializar el factor de aprendizaje
2. Inicializar épocas en 0
3. Mientras épocas < max_ctd_épocas
   - Para cada ejemplo tomado al azar del conjunto de entrenamiento
     - Presentar el ejemplo en la capa de entrada
     - Obtener la neurona ganadora
     - Actualizar los pesos de las conexiones de la neurona ganadora
     - Actualizar los pesos de las conexiones de las neuronas vecinas
   - Incrementar épocas
   - Actualizar el radio de vecindad
   - Actualizar el factor de aprendizaje
4. Fin

Inicialización de pesos:

- valores aleatorios con distribución uniforme. Ejemplo: U(-1,1)
- Asignando aleatoriamente un dato de entrenamiento



Otras inicializaciones:

- **max_cts_épocas**: se puede poner constante x n, siendo la cant. de entradas. (ej. cte = 500)
- **radio de vecindad R**: R = k, permite que al inicio el vecindario abarque prácticamente todas las unidades.

Formula actualización de pesos

Actualización neurona ganadora (i,j)

$$w_k^{i,j}(t+1) = w_k^{i,j}(t) + \Delta w_k^{i,j}$$

$$\Delta w_k^{i,j} = \eta * (x_k^p - w_k^{i,j})$$

El factor de aprendizaje η dependerá del tamaño del conjunto de entrenamiento y de las épocas. Usualmente se usa **0,1**

Actualización η de forma decreciente

Actualización neuronas vecinas (i`,j`)

$$w_k^{i',j'}(t+1) = w_k^{i',j'}(t) + \Delta w_k^{i',j'}$$

$$\Delta w_k^{i',j'} = V * \eta * (x_k^p - w_k^{i',j'})$$

$$V = e^{(-2d/R)}$$

d = distancia entre (i,j) e (i`,j`)

Actualización R de forma decreciente. Una forma:

$$R = (max\_ctd\_epocas - epoca) * R_{inicial} / max\_ctd\_epocas)$$

Convergencia

De acuerdo a la regla de actualización de $w^{i,j}$ y restando x^p en ambos miembros:

$$w^{i,j}(t+1) - x^p = w^{i,j}(t) + \eta(x^p - w^{i,j}(t)) - x^p$$

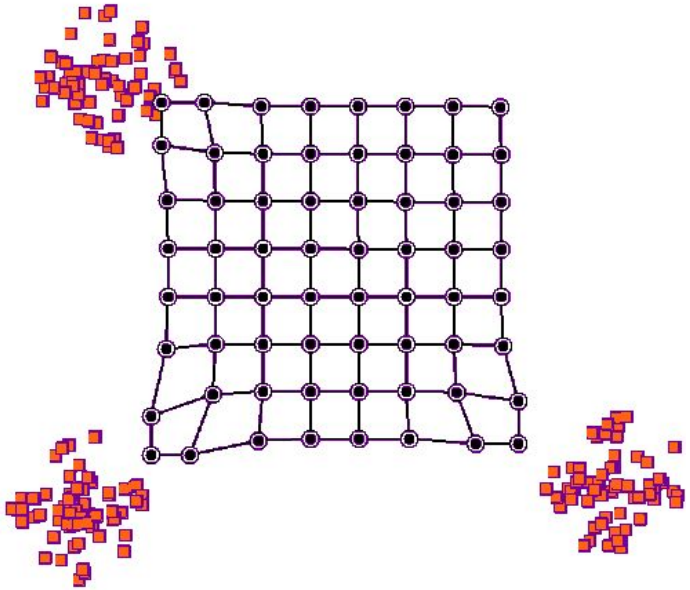
Agrupando y sacando factor común $(w^{i,j}(t) - x^p)$:

$w^{i,j}(t+1) - x^p = (1-\eta)(w^{i,j}(t) - x^p)$.

Así, en cada actualización resulta que:

$$\|w^{i,j}(t+1) - x^p\| \leq \|w^{i,j}(t) - x^p\|$$

con lo cual $w^{i,j}$ debería converger a x^p.



Ejemplo - Censo de Irlanda

Datos: Para cada ciudad se toman los datos de las siguientes variables:

- Cantidad de habitantes
- Promedio de edad
- Promedio de nivel de educación
- Número de autos registrados
- Número de personas desempleadas
- Número de personas subempleadas

para todas las ciudades irlandesas

Queremos saber

Qué ciudades son parecidas, tomando en cuenta todas las variables.

En otras palabras, poder agrupar ciudades según su parecido en todas las variables.

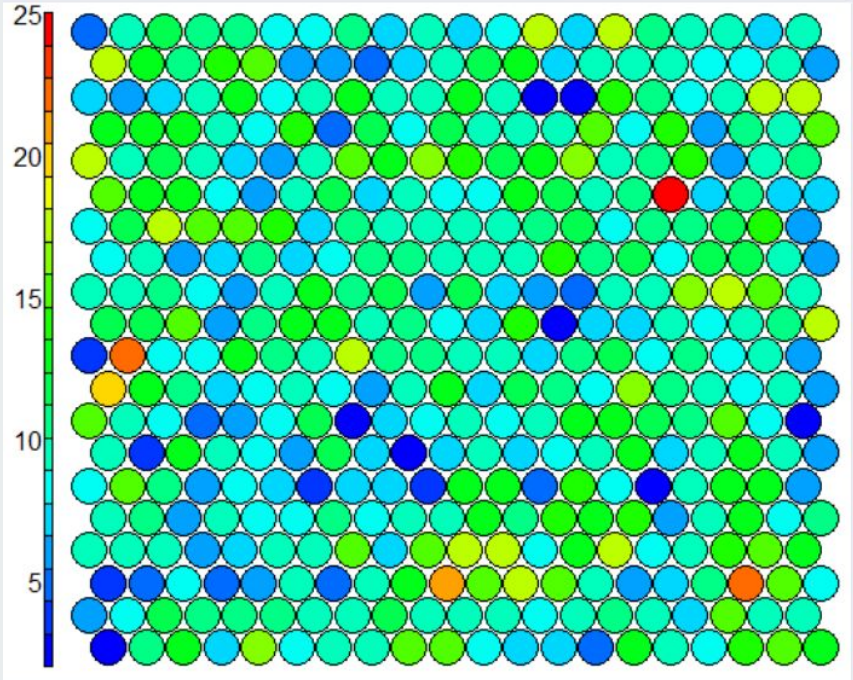
Utilizando una grilla de 20 x 20

Ejemplo - Censo de Irlanda

Mapa de Activación

Número de registros que activan cada neurona

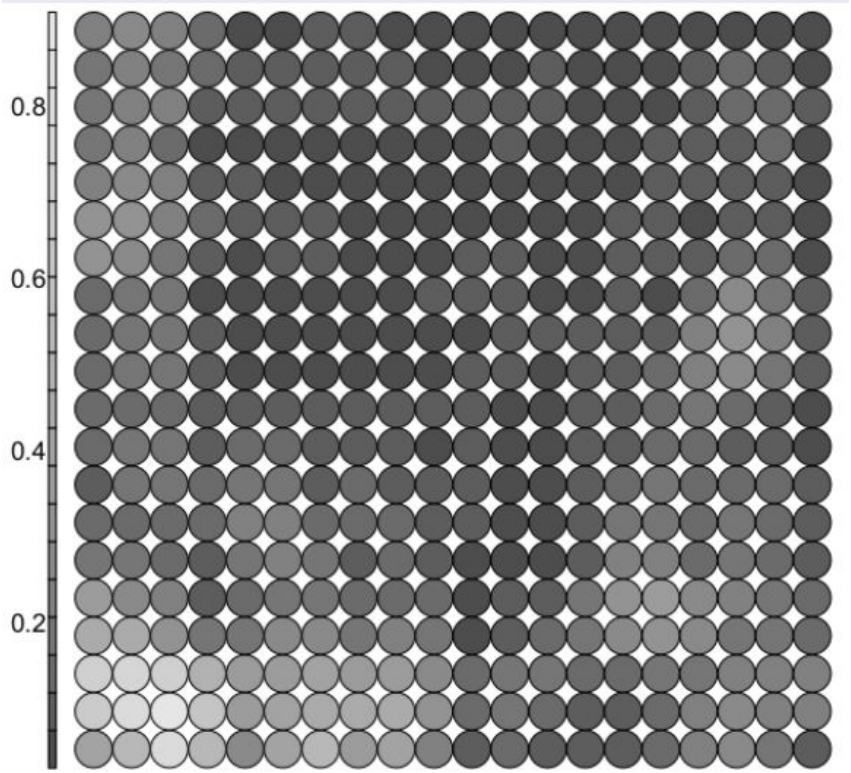
Visualización en 2 dimensiones



Ejemplo - Censo de Irlanda

Matriz U

La matriz U tiene, para cada nodo el promedio de la distancia euclídea entre el vector de pesos del nodo y el vector de pesos de los nodos vecinos.

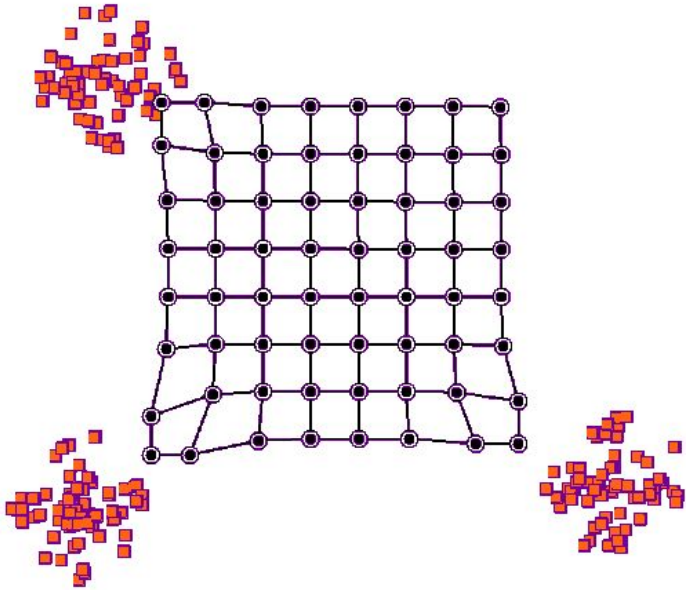


Características

La red de Kohonen permite mapear un espacio cuya dimensión puede ser grande, en un espacio **bi-dimensional**.

Una vez concluido el aprendizaje, la **distribución** de los pesos de las neuronas de la capa de salida es la misma que la distribución de los ejemplos de entrenamiento. Por ejemplo, si la distribución de los ejemplos de entrada es gaussiana, la distribución de los vectores de pesos de las neuronas de la capa de salida es gaussiana.

La red de Kohonen permite **agrupar sub-conjuntos** de ejemplos en clases (o grupos).
Esto es, los ejemplos que activan neuronas cercanas en la capa de salida, ser´an ejemplos que pertenecerán a una misma clase.



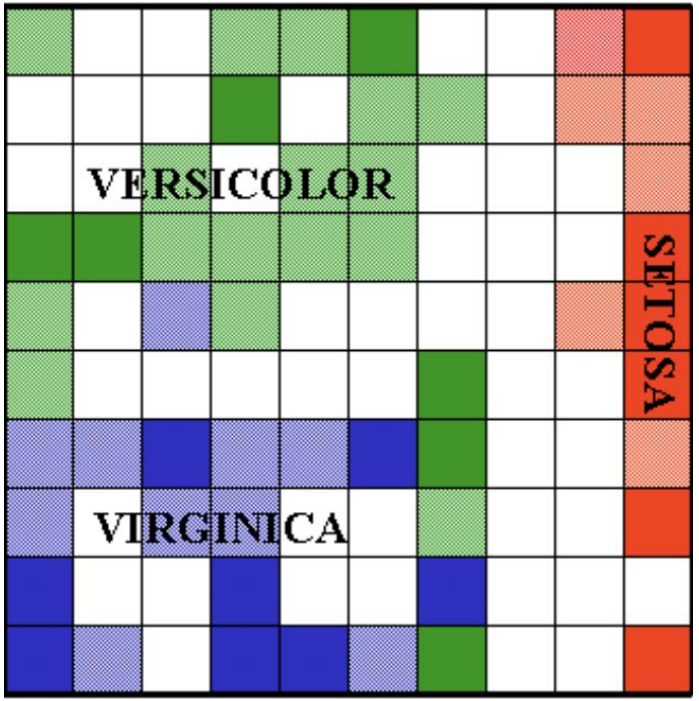
Ejemplo - Iris

Grilla de ….. x ……

Color indica ….

Dimensión de cada neurona …

Intensidad del color indica ….



¡Fin! ¿Alguna pregunta?