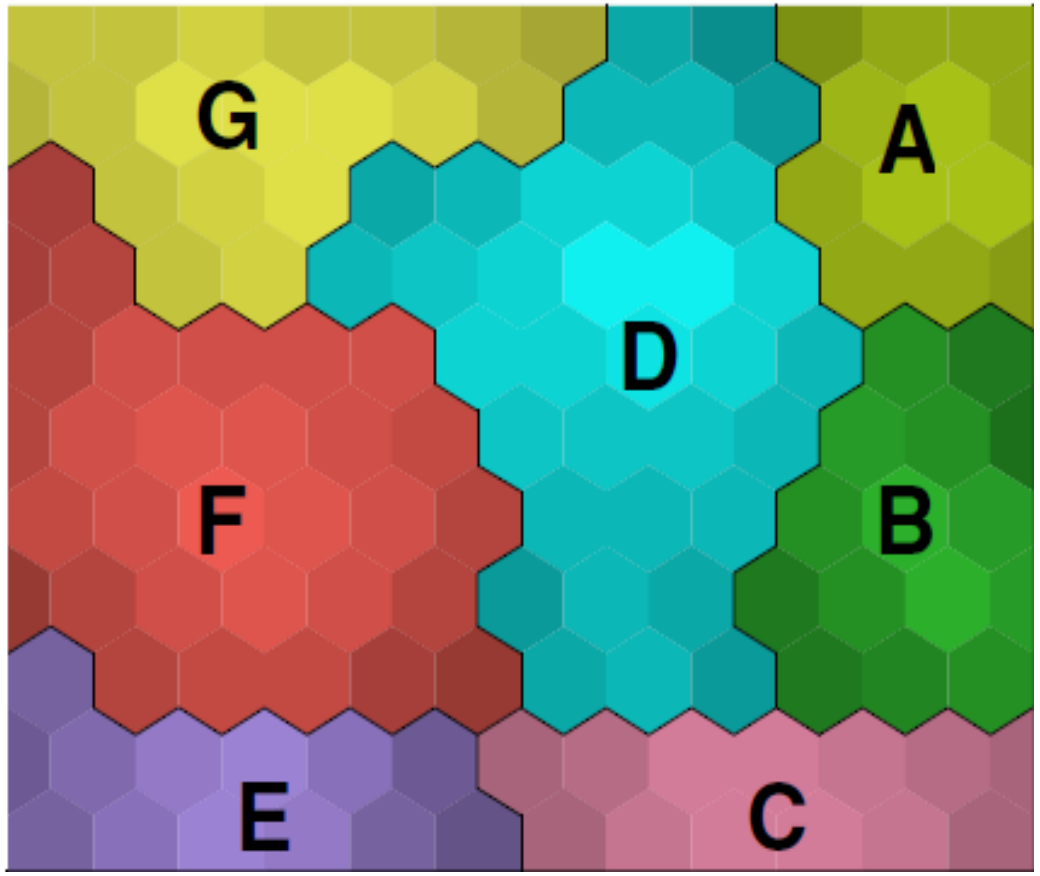
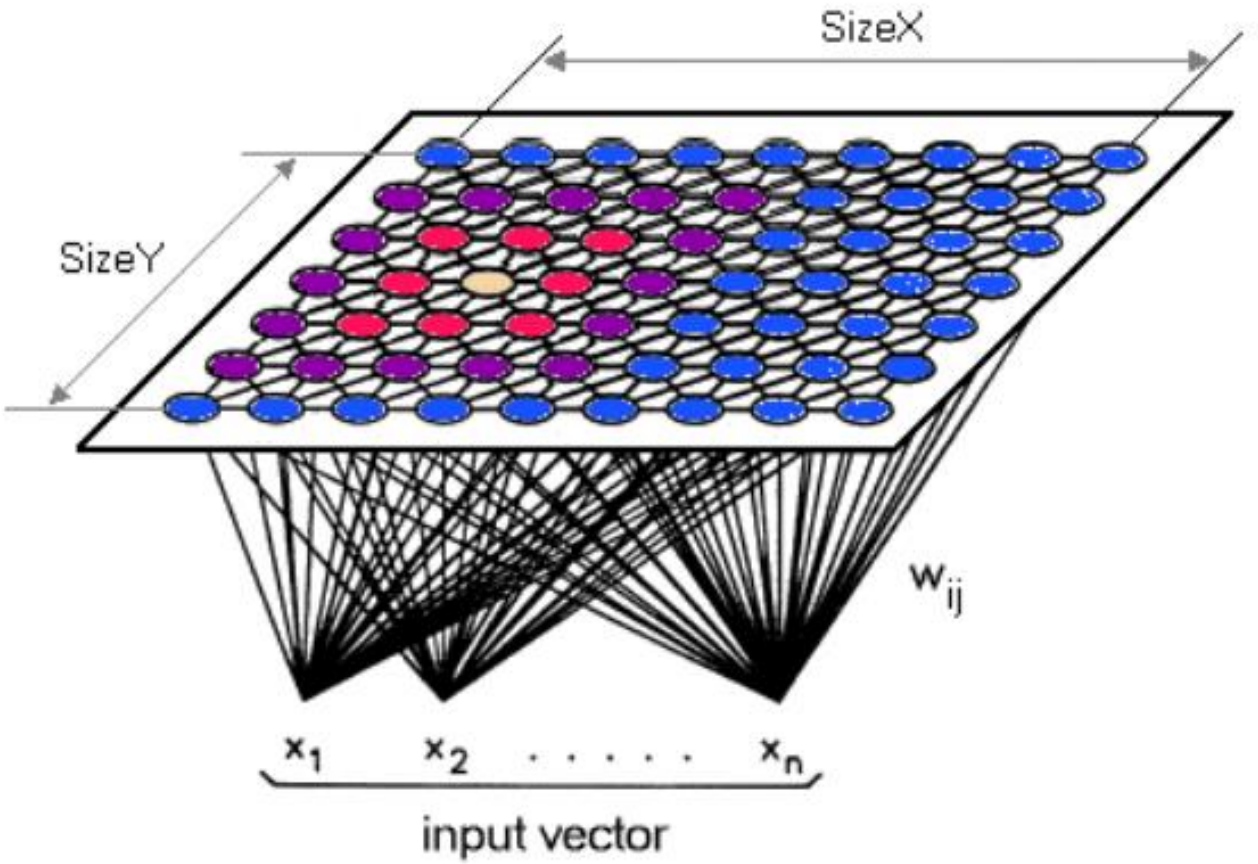
Self-Organizing Maps
(Metodología utilizada en Python & Matlab)

Jairo F. Gudiño R. (Julio – 2017)

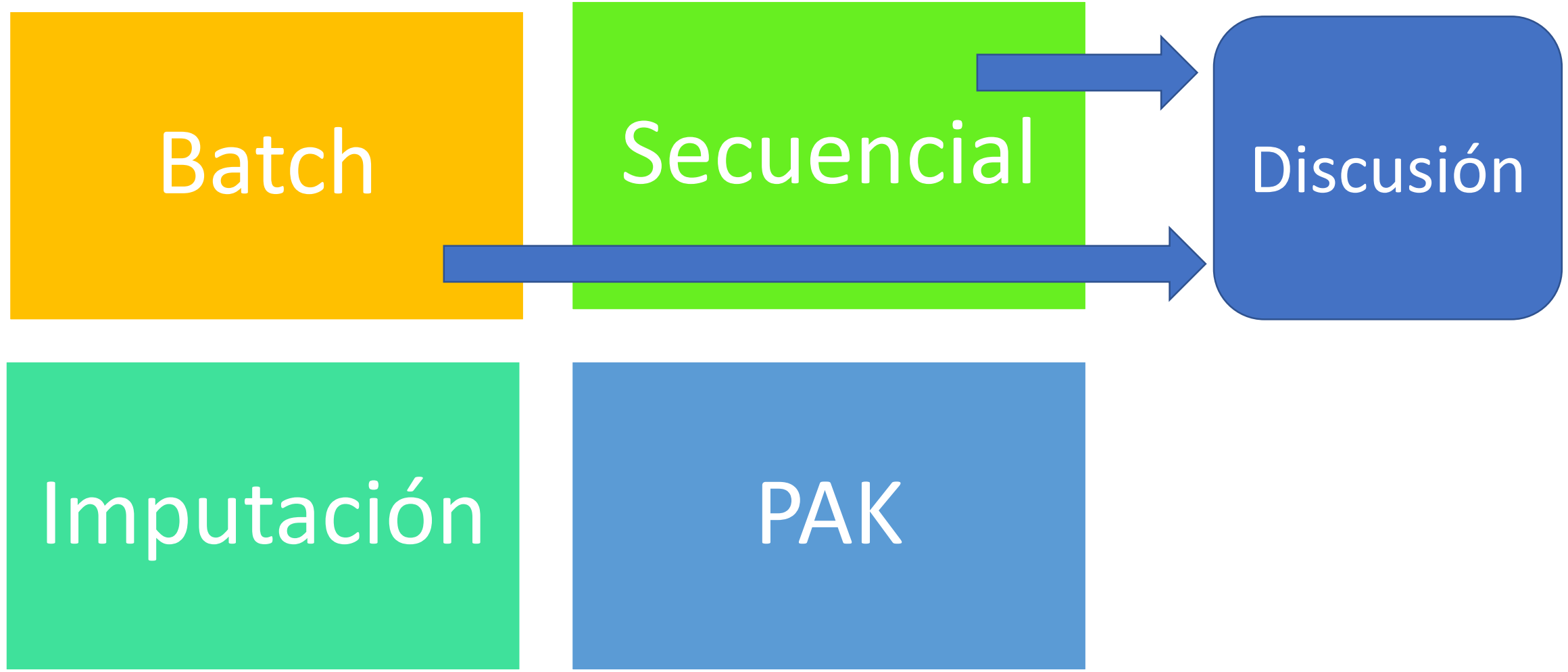




Ruta de trabajo

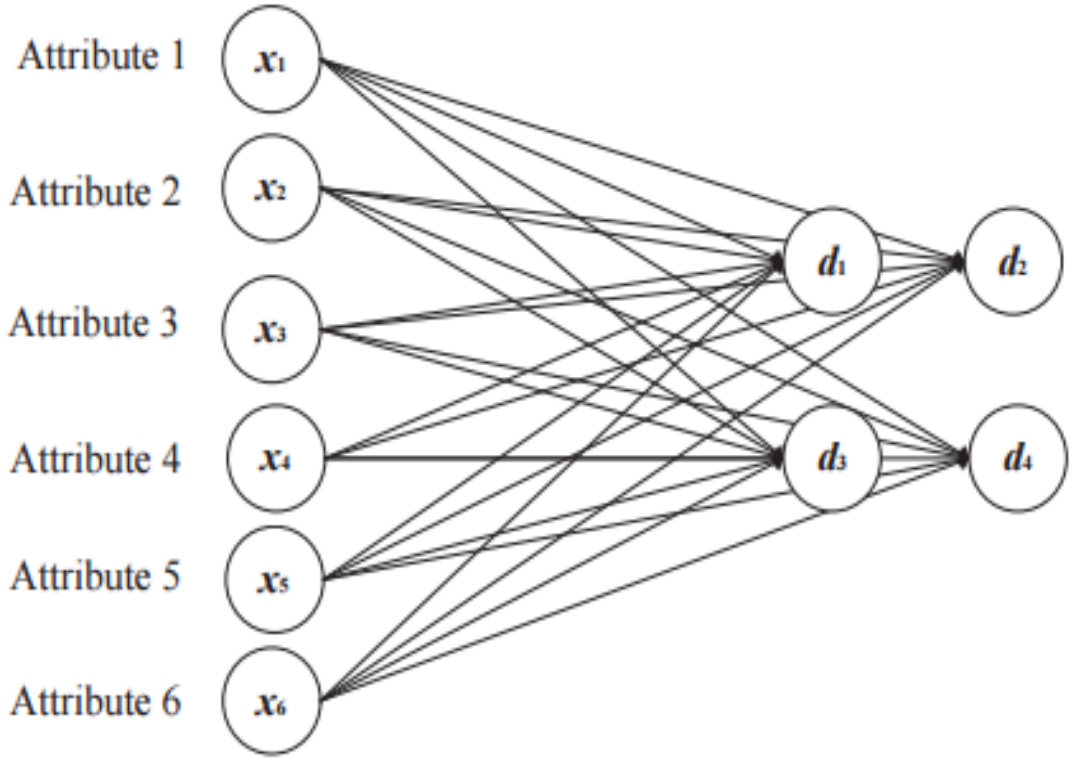
1. Explicación Técnica
2. Extensiones
3. Síntesis

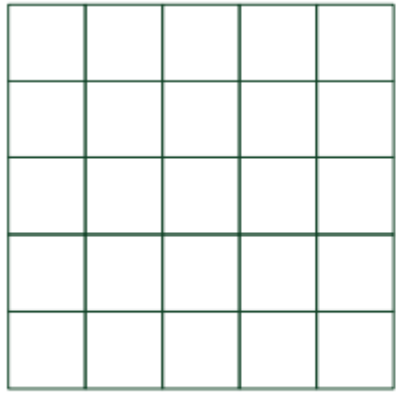
Tipos de Training



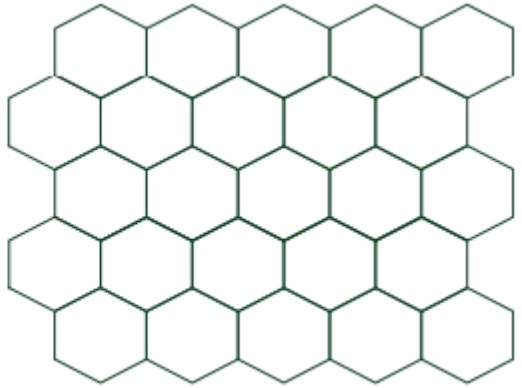
1. Self-Organizing Maps: Explicación técnica

SOM: Red con dos layers: *Input layer* (con *m* nodos/atributos) conectado de forma directa a un *Output* (o *Kohonen*) *layer* generalmente de dos dimensiones con un grid hexagonal. Topológicamente, los valores extremos localizados en bordes del output layer son incorporados de manera más satisfactoria mediante interacciones con valores vecinos al ser menos centrales (Asan & Ercan, 2012).





(a) Rectangular topology



(b) Hexagonal topology

1. Self-Organizing Maps: Explicación técnica

1. Determinación del tamaño del *Output layer*:

Con un input de M observaciones y N features, se calcula el producto punto entre cada uno de los N *features* y al vector resultante se le calcula un valor promedio, hasta completar una matriz cuadrada A (NxN). Obtener los eigenvalores asociados y calcular un ratio R, utilizando el número de unidades (Vesanto et. al., 2000) – default: 100 ó $5^*\sqrt{N.Epochs}$:

$$R = \sqrt{\frac{Eigenvalor\ mayor}{Segundo\ eigenvalor\ mayor}}$$

$$N.Columnas = Mín\ \{N.unidades, \sqrt{\frac{N.unidades}{R}}\} \qquad N.Filas = \frac{N.unidades}{N.Columnas}$$

2. Creación de valores aleatorios:

Se crea una matriz de dimensión (N. Filas x N. Columnas) x N *features* con valores aleatorios. Cada una de las filas son pesos iniciales de los nodos (Asan y Ercan, 2012).

1. Self-Organizing Maps: Explicación técnica

3. Similarity Matching:

Se encuentra para cada una de las observaciones el Best Matching Unit (BMU): El nodo que es más cercano a la observación por medio de distancias euclidianas (Kohonen, 2013).

$$c(t) = \arg\min_i \{\|x(t) - w_i(t)\|\}$$

Del procedimiento de optimización se obtienen dos medidas de interés: (i) El BMU para cada una de las observaciones; (ii) La distancia entre la observación y su BMU correspondiente.

4. Cálculo del error de cuantificación

Sumar para cada observación las distancias mínimas respecto de su respectivo BMU a la norma del vector de *features* asociado. Al valor resultante calcular la raíz cuadrada. Finalmente, promediar el total de estas raíces al cuadrado → Quantization Error (QE): medida sujeta a minimización.

1. Self-Organizing Maps: Explicación técnica

5. Determinación de valores de inicialización de radios de vecindario y número de epochs.

El **número de *epochs* (t)** hace referencia al número de repeticiones necesarias tal que el proceso de ajuste del output layer a las observaciones es máximo, y su número óptimo se puede encontrar en el punto donde el QE es mínimo.

Este número determina la longitud del **radio de vecindario (σ)**, que se define como vector de distancias entre un BMU determinado y el resto de nodos del output layer. Es igual para todos los nodos, y sus valores iniciales y finales siguen reglas determinísticas:

$$Radio\ Inicial\ (Default) = Máx\ \{1, \frac{Máx\{N.Filas, N.Columnas\}}{8}\}$$

$$Radio\ Final(Default) = Máx\ \{1, \frac{Radio\ Inicial}{4}\}$$

$$Número\ de\ Epochs(Default) = Máx\{1, 50\ .\ (\frac{N.Filas\ x\ N.Columnas}{N})\}$$

1. Self-Organizing Maps: Explicación técnica

6. Cálculo de distancias entre los nodos del Output Layer

Siendo ya definido el tamaño del output layer, se calculan las coordenadas entre cada uno de los nodos.

0	0
0,5	1
0	2
0,5	3
0	4
0,5	5
0	0
1,5	1

En la primera columna, se intercalan valores 0 y un número con decimal en 5. Luego, se multiplican los valores por $\sqrt{0.75}$.
En la segunda columna, se repiten valores desde 0 hasta el número de features un número de veces igual al número de columnas calculado anteriormente. Luego, en las posiciones pares, se suma 0.5.

Dado el cálculo de las coordenadas, se utiliza la fórmula de la distancia euclidiana para obtenerse una matriz de distancias **d** (Vesanto et. al. 2000).

1. Self-Organizing Maps: Explicación técnica

7. Cálculo de funciones de vecindario (h) para cada epoch

En forma de loop, con la matriz de distancias ($\boldsymbol{d_{ic}}$) entre nodos *c* e *i* y un radio de vecindario se calcula una *función de vecindario*, de manera que los nodos más cercanos a cierto BMU tenderán a altas similaridades entre sí (Kohonen, 2013).

$$h_{ci}(t) = \mathbf{1}(\sigma(t) - d_{ci})$$ **Bubble**

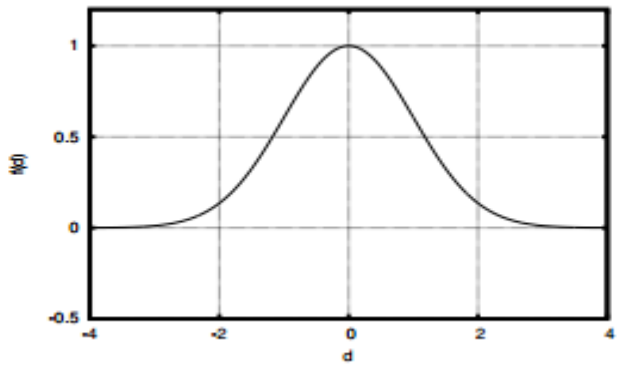
$$h_{ci}(t) = \exp\left(-\frac{d_{ci}^2}{2\sigma^2(t)}\right)\mathbf{1}(\sigma(t) - d_{ci})$$ **Cutgauss**

$$h_{ci}(t) = \max\left\{0,\ 1 - (\sigma(t) - d_{ci})^2\right\}$$ **Epanechnikov**

$\mathbf{1}(x)$ is the step function: $\mathbf{1}(x) = 0$ if $x < 0$ and $\mathbf{1}(x) = 1$ if $x \geqslant 0$.

$$h_{ci}(t) = \exp\left(-\frac{d_{ci}^2}{2\sigma^2(t)}\right)$$

Función gaussiana



1. Self-Organizing Maps: Explicación técnica

8. Reactualización de pesos de cada uno de los nodos en cada *epoch*

A partir del cálculo de funciones de vecindario, se reemplazan los pesos aleatorios asignados a cada nodo por el promedio de los pesos de sus nodos vecinos en cada *feature*. La contribución o activación de los nodos vecinos está determinada por el radio de la función de vecindario escogida (Vesanto et. al., 2000).

El proceso de reactualización termina en tanto se calcula el QE del último valor del *epoch* escogido.

Self-Organizing Maps: Sequential Training

Determinación del Radio de Vecindad [σ(t)]

- Se determina la longitud radial mediante la fórmula

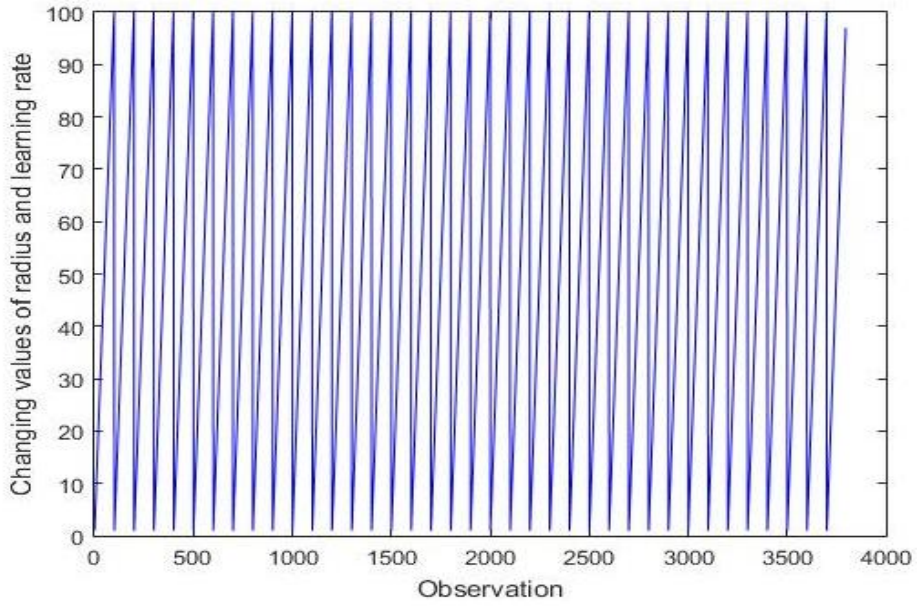
$$L_R = \frac{\sigma_{Final} - \sigma_{Inicial}}{N.de\ Observaciones - 1}$$

- Se determina el radio de vecindad (por default lineal):

$$\sigma(t) = \{\sigma_{Inicial} + [0{:}99]' * L_R\}^2$$

Determinación de la Tasa de Aprendizaje [α(t)]

Función de Aprendizaje	Fórmula
Función Lineal	$\frac{0{:}-1{:}-99}{N.de\ Observaciones} * \alpha_{Inicial}$
Función Inversa	$\frac{\alpha}{(\beta + [0{:}99]')}$
Función de Poder	$\alpha_{Inicial} * \{\frac{0.005}{\alpha_{Inicial}}\}^{\frac{[0:99]'}{N.de\ Observaciones}}$

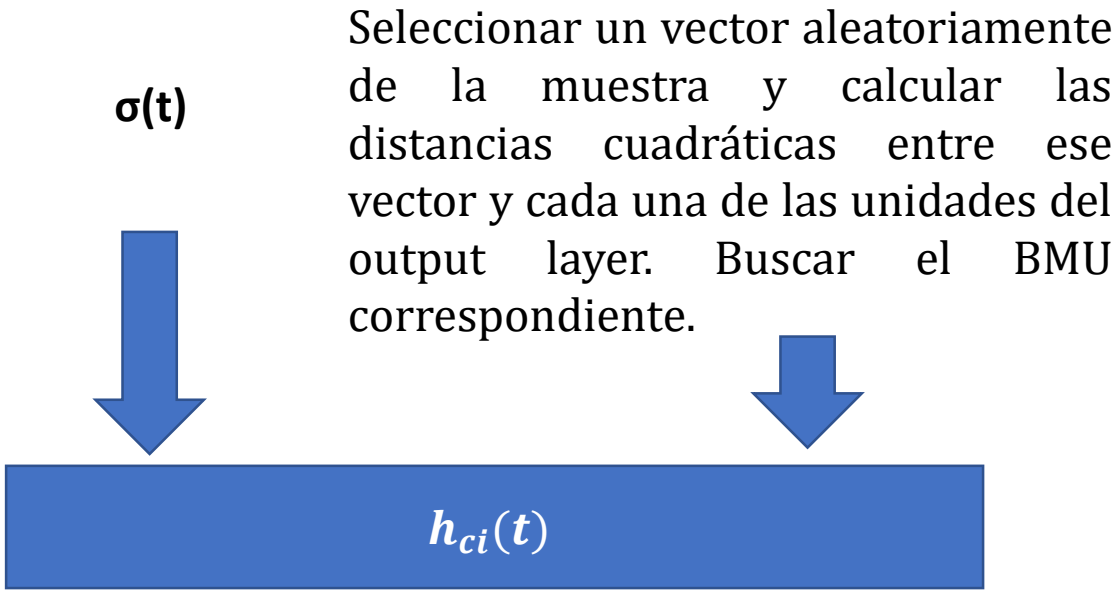


Valores cíclicos

$$\beta = \frac{N.de\ Observaciones - 1}{99};$$

$$\alpha = \alpha_{Inicial} * \beta$$

Self-Organizing Maps: Sequential Training



La reactualización de los pesos de los nodos sigue un proceso con una tasa de aprendizaje α para cada muestra que aparece (Kohonen, 2013):

$$w_i(t+1) = w_i(t) + \alpha(t)[x(t) - w_i(t)]$$, para un BMU *i*

$$w_i(t+1) = w_i(t) + \alpha(t)h_{ci}(t)[x(t) - w_i(t)]$$, para un nodo vecino *i*

En estas ecuaciones, los pesos del output layer son reactualizados buscando a cada observación su BMU y luego alterando los pesos de los nodos vecinos dependiendo del radio de vecindario calculado.

2. Síntesis

Competencia:

Los *output nodes* compiten entre sí por representar mejor cada una de las observaciones. El BMU se determina mediante distancias euclidianas.

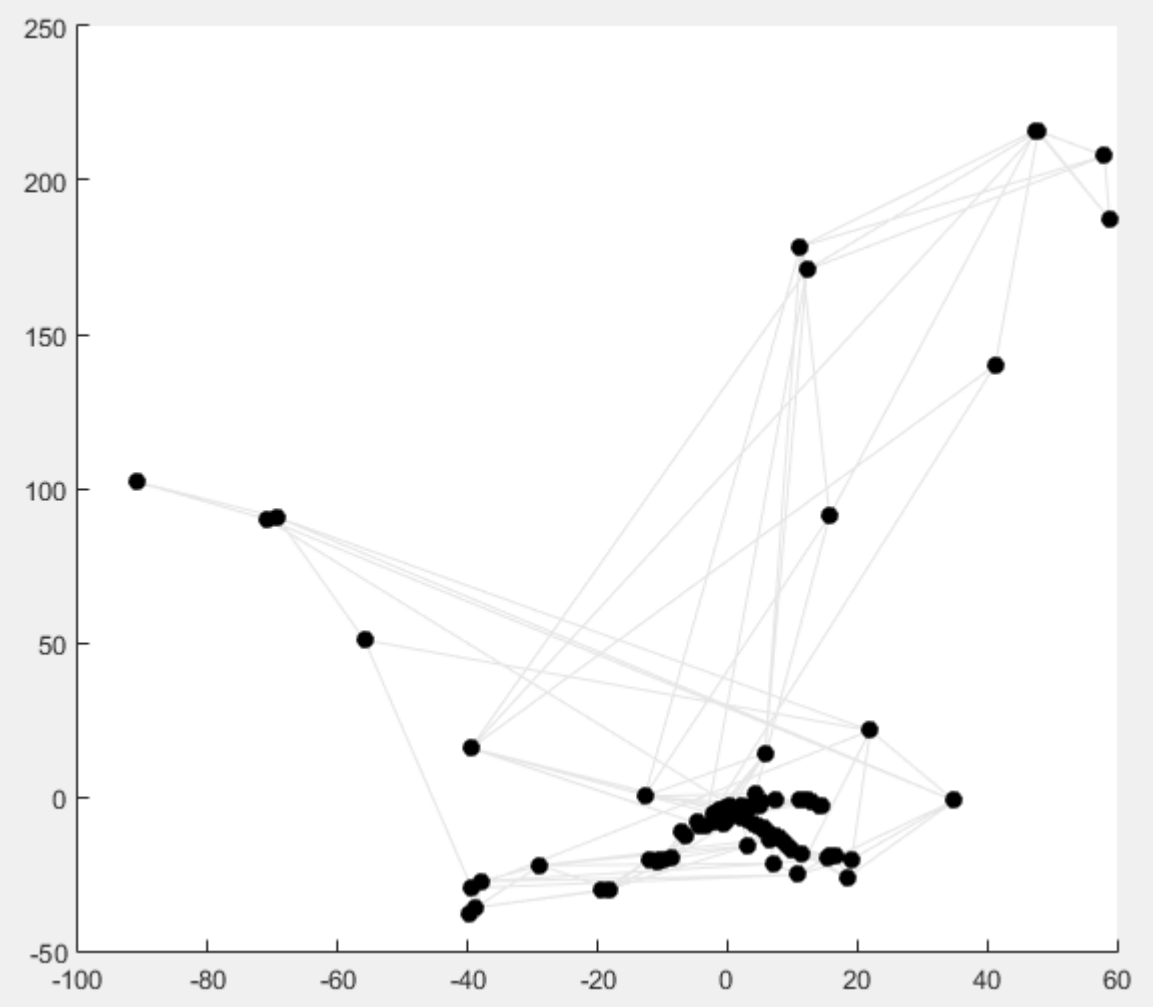
Cooperación:

El SOM es una organización topográfica donde los BMU determinan la localización espacial de un vecindario de nodos cooperativos.

Adaptación:

Los pesos de cada uno de los *output nodes* son ajustados a través de un proceso de aprendizaje a partir de funciones de vecindario.

3. Extensiones



A partir de la matriz de distancias entre nodos del *output layer*, es posible utilizar el algoritmo de Sammon (1969) para reducir la dimensionalidad de los datos, de 25 a 2.

Let $E(m)$ be defined as the mapping error after the mth iteration, i.e.,

$$E(m) \equiv \frac{1}{c} \sum_{i<j}^{N} [d_{ij}^* - d_{ij}(m)]^2 / d_{ij}^*$$

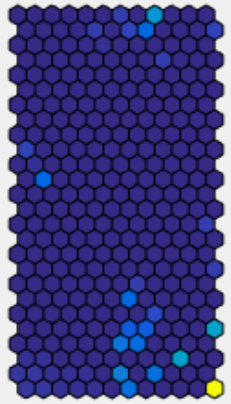
where

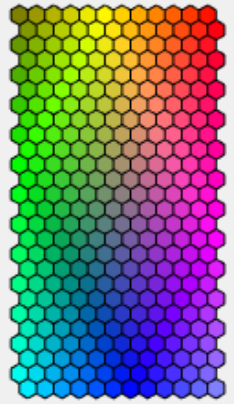
$$c = \sum_{i<j}^{N} [d_{ij}^*]$$

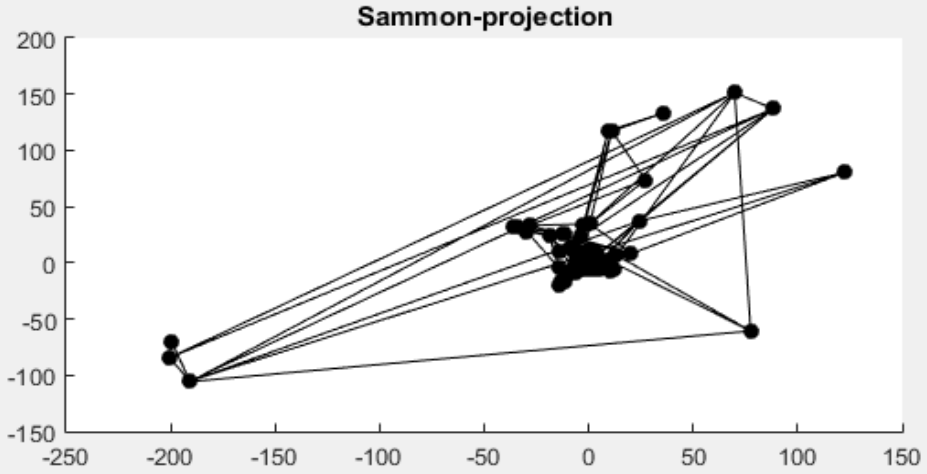
The new d-space configuration at time $m+1$ is given by

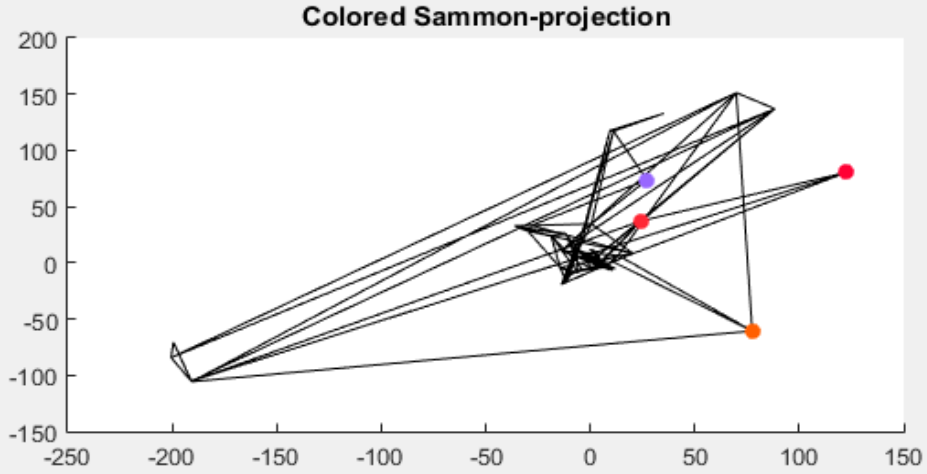
$$y_{pq}(m+1) = y_{pq}(m) - (MF) \cdot \Delta_{pq}(m)$$

3. Extensiones

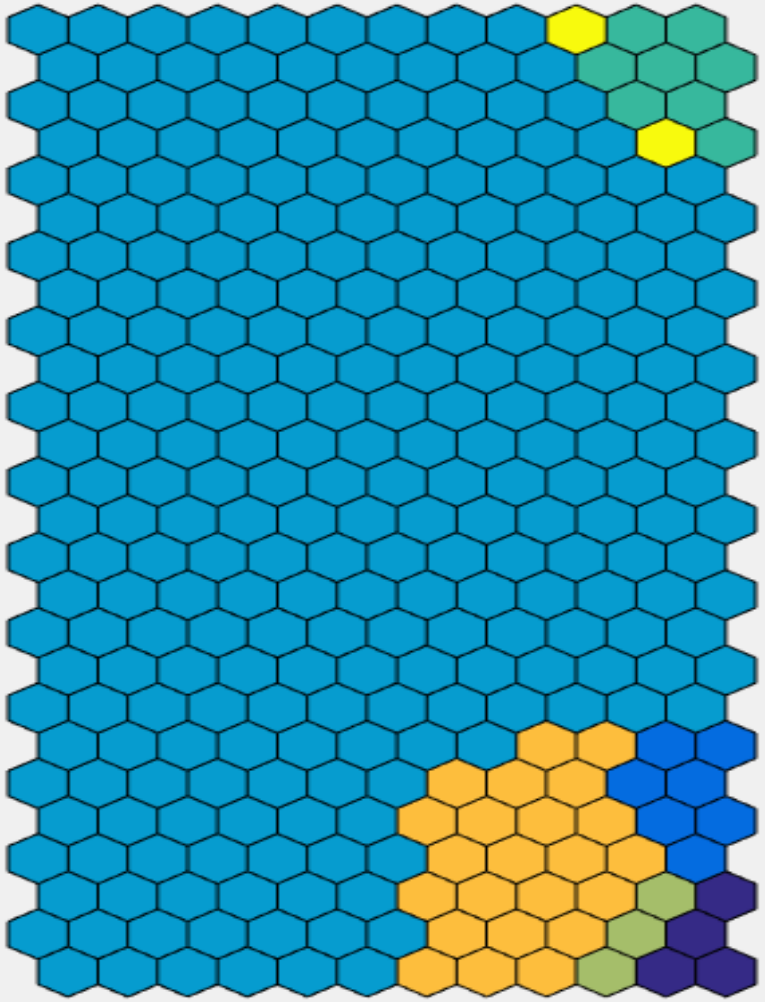








3. Extensiones



Predefiniendo 7 clusters, es posible utilizar el algoritmo de k-medias a partir de los pesos finales de cada uno de los nodos del output layer. Es posible utilizar diversas métricas para evaluar el ajuste de resultados.

Por último, es posible deducir la probabilidad de pertenencia a cada uno de los clusters a partir de modelos de mezclas gaussianas (GMM), siguiendo a Bishop (2006).

Referencias bibliográficas

Asan, U. & Ercan, S. (2012). "An Introduction to Self Organizing Maps". Computational Intelligence Systems in Industrial Engineering. Chapter 14. Atlantis Press.

Bishop, C. (2006). "Pattern Recognition and Machine Learning". Chapter 9. Springer Books.

Kohonen, T. (2013). "Essentials of the Self-Organizing Map", *Neural Networks*, Twenty-fifth Anniversary Commemorative Issue, 37, 52-65.

Kolari, J. & Sánz, P. (2016). "Systemic Risk Measurement in Banking using Self-Organizing Maps", *Journal of Banking Regulation*, 1-21.

León, C., Gutiérrez, J. & Cely, J. (2016). "Whose balance is this?". Borradores de Economía No. 959. Banco de la República, Bogotá.

Sammon, J. (1969). "A Non-linear mapping for Data Structure Analysis", *IEE Transactions on Computers*, Vol. C-18 (5), 401-409.

Vesanto, J., Himberg, J., Alhoniemi, E., & Parhankangas, J. (2000). "Self-Organizing Toolbox for Matlab". Helsinki University of Technology Working Papers. Report A57.