Métodos Locales II (*Manifold Learning*)

Alan Reyes-Figueroa
Aprendizaje Estadístico (Aula 12) 19.febrero.2024

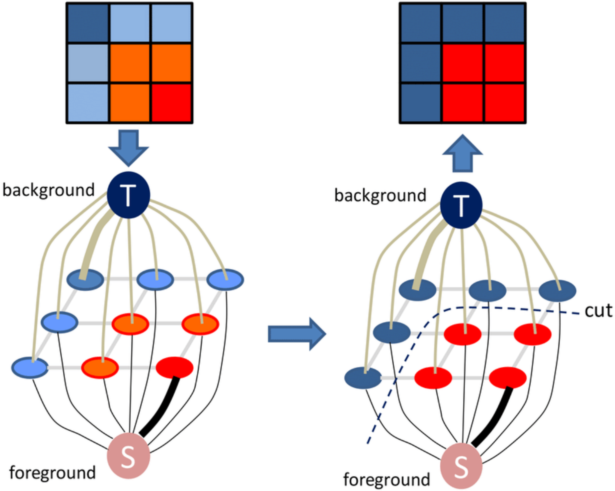
Spectral Embedding

Ref: Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. M. Belkin, P. Niyogi, Neural Computation, June 2003; 15(6) 1373-1396.

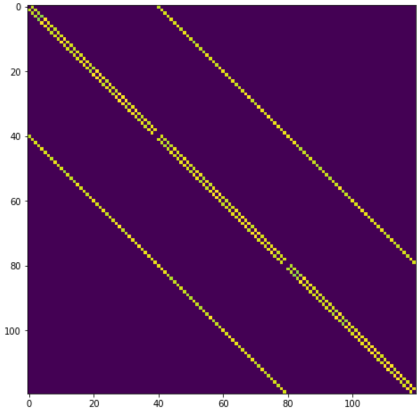
Idea: Se construye una matriz de adyacencia o de afinidad (similaridad) W entre una estructura de grafo entre los datos. Los elementos w_{ij} de W pesan o miden el grado de afinidad.

- Se construye la matriz laplaciana $L = D - W$ y la laplaciana normalizada $\mathcal{L} = D^{-1/2}(D - W)D^{-1/2}$.
- Se calculan la descomposición en autovalores de $\mathcal{L}$. Los autovectores describen las direcciones de proyección.

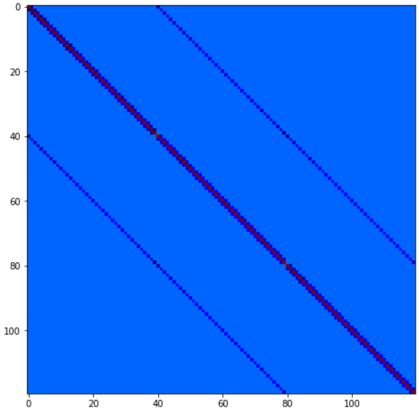
Spectral Embedding



Spectral Embedding



Matriz de afinidad W



Laplaciano normalizado $\mathcal{L}$

UNIVERSIDAD DEL VALLE DE GUATEMALA

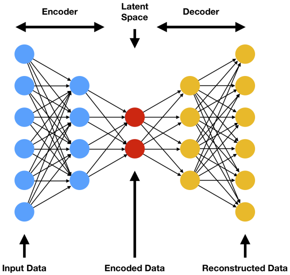
Autoencoders

Definir mapas lineales $\mathcal{E} : \mathbb{R}^d \to \mathbb{R}^p$ y $\mathcal{D} : \mathbb{R}^p \to \mathbb{R}^d$, con $p < d$.

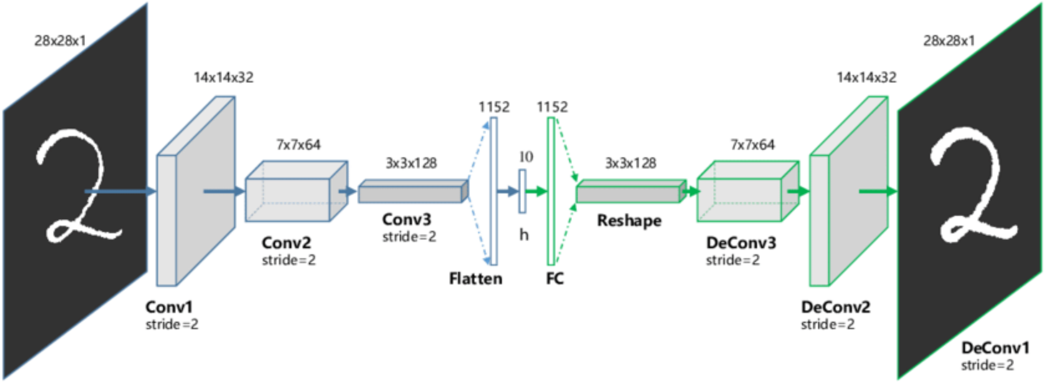
$\mathcal{E}$ se llama el *encoder*, y $\mathcal{D}$ el *decoder*. El objetivo es resolver

$$\min_{\mathcal{E},\mathcal{D}} \sum_i ||\mathbf{x}_i - (\mathcal{D} \circ \mathcal{E})(\mathbf{x}_i)||^2.$$

Se usa $\mathbf{x}_i^* = \mathcal{E}(\mathbf{x}_i)$ como representación de $\mathbf{x}_i$. La elección popular para $\mathcal{E}$ y $\mathcal{D}$: redes neuronales.



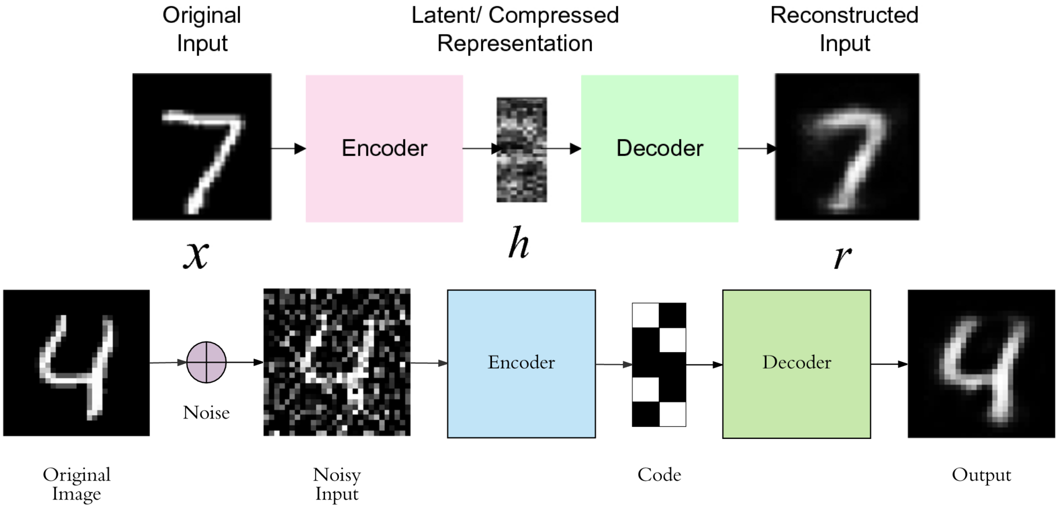
Autoencoders



Red neuronal covolucional profunda (CNN).

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

Autoencoders



LLE

LLE: *Local Linear Embedding*

Refs: Roweis ST, Lawrence LK (2000) Nonlinear Dimensionality Reduction by Locally Linear Embedding, Science 290(5500): 2323-2326.
`https://cs.nyu.edu/ roweis/lle/publications.html`

Idea: Caracterizar la estructura local en el espacio original, y tratamos de conservar esta estructura local en el nuevo espacio.

- Si conozco los k-vecinos más cercanos a $\mathbf{x}_i$, denotados por $\{\mathbf{x}_j : j \in vec(i)\}$.
  Vamos a tratar de escribir $\mathbf{x}_i$ como combinación lineal de sus k-vecinos más cercanos ($k < d$)

$$\mathbf{x}_i = \sum_{j \in vec(i)} w_{ij}\mathbf{x}_j, \quad \text{com} \sum_{j \in vec(i)} w_{ij} = 1.$$

LLE

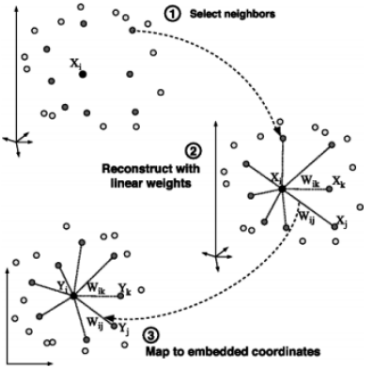
- Para cada $\mathbf{x}_i$ buscamos los k-vecinos más cercanos $\{\mathbf{x}_j : j \in vec(i)\}$.
- Resolvemos

$$\min_{w_{ij}} \sum_i ||\mathbf{x}_i - \sum_{j \in vec(i)} w_{ij}\mathbf{x}_j||^2,$$

  sujeto a $\sum_j w_{ij} = 1$.
- Resolvemos

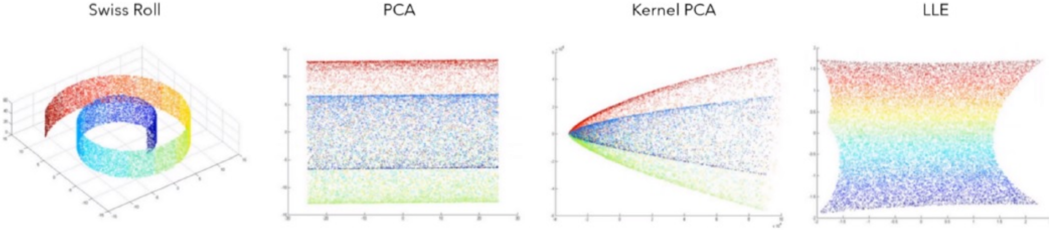
$$\min_{\mathbf{x}_j^*} \sum_i ||\mathbf{x}_i^* - \sum_{j \in vec(i)} w_{ij}\mathbf{x}_j^*||^2,$$

  sujeto a restricciones de norma y promedio de $\mathbf{x}_i^*$.



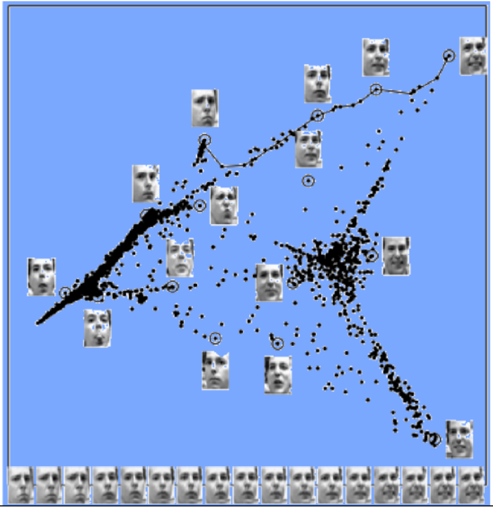
UNIVERSIDAD DEL VALLE DE GUATEMALA

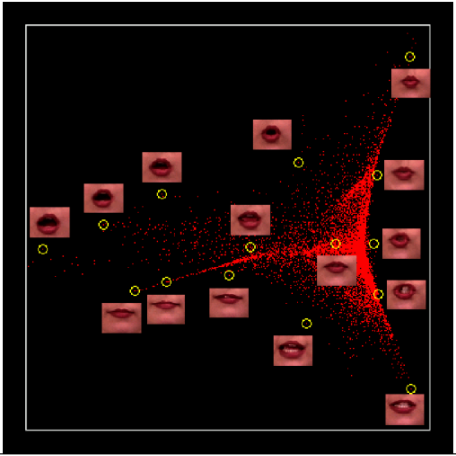
LLE



Source: Jennifer Chu. Image free to share

LLE

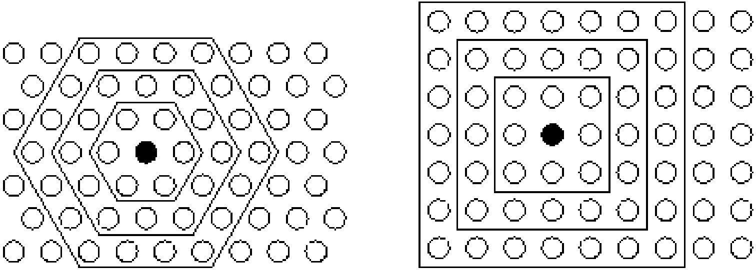




SOM: *Self organizing maps*

Ref: Kohonen, Teuvo (1982). Self-Organized Formation of Topologically Correct Feature Maps. Biological Cybernetics 43 (1): 59-69.

Idea: Colocar cada dato $\mathbf{x}_i$ en una celda $c_{\ell(i)}$ de una retícula o *grid*. Asociamos con cada celda c_ℓ un representante $\mathbf{m}_\ell \in \mathbb{R}^d$.



Imponemos que

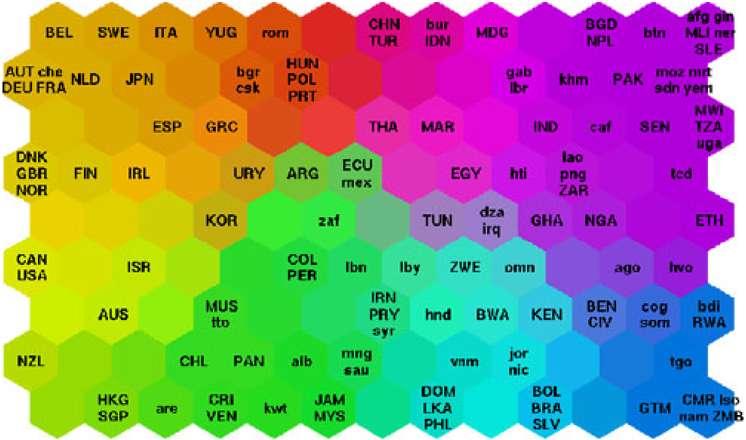
- los representantes a celdas cercanas sean similares,
- los datos son similares al representante de su celda.

Repetir para cada $\mathbf{x}_i$:

1. Buscar el repreentante más cercano a $\mathbf{x}_i$, denotado como $\mathbf{m}_{\ell(i)}$
2. Para todas las celdas c_k,, actualizamos
$$\mathbf{m}_k = \mathbf{m}_k + \alpha h\big(d(c_k, c_\ell(i))^2\big)||\mathbf{x}_i - \mathbf{m}_k||^2.$$
(*h* es positiva y decreciente, d es la distancia en el grid, α es un tamaño de paso decreciente en el tiempo.)

El método minimiza la función de costo
$$J(\{\mathbf{m}_k\}, \{\ell(i)\}) = \sum_\ell \sum_k h\big(d(c_k, c_\ell(i))^2\big)||\mathbf{x}_i - \mathbf{m}_k||^2.$$

SOM



SOM de países sobre 39 indicadores: salud, educación, economía, servicios, ... (Kohonen)

SOM

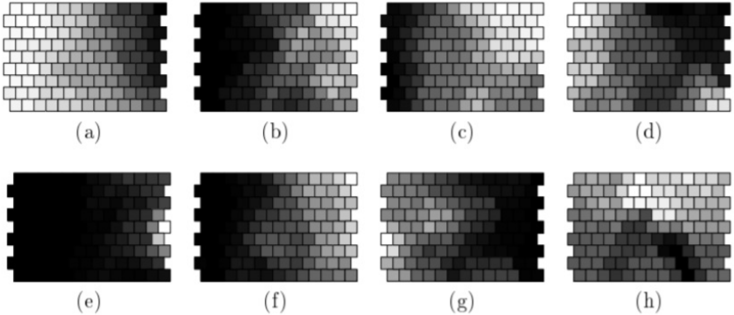
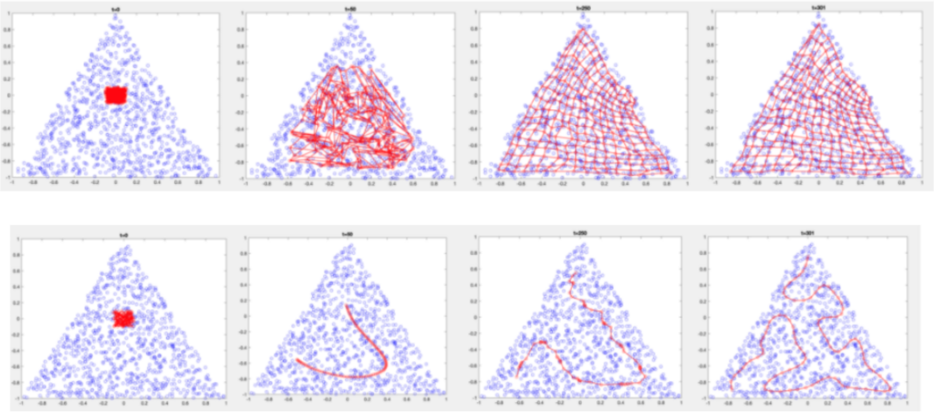


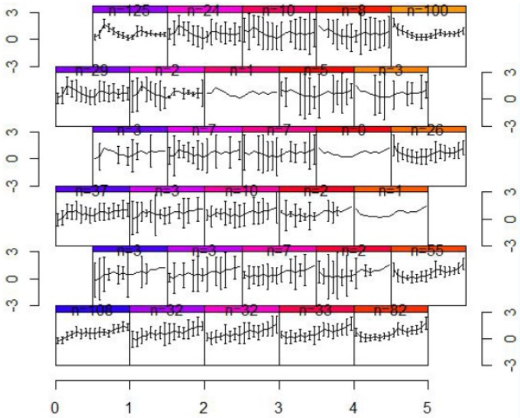
Figure 2: The values of some of the indicators visualized on the SOM groundwork: (a) Life expectancy at birth (years); (b) Adult illiteracy (%); (c) Share of food in household consumption (%); (d) Share of medical care in household consumption (%); (e) Population per physician; (f) Infant mortality rate (per thousand live births); (g) Tertiary education enrollment (% of age group); and (h) Share of the lowest-earning 20 percent in the total household income. In each display, white indicates the largest value and black the smallest, respectively.

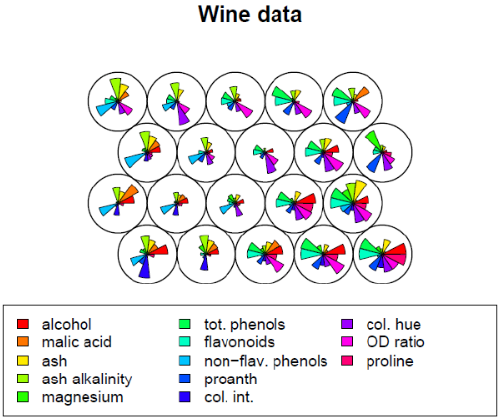
Ejemplo 2 https://towardsdatascience.com/how-to-implement-kohonens-self-organizing-maps-989c4da05f19



UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

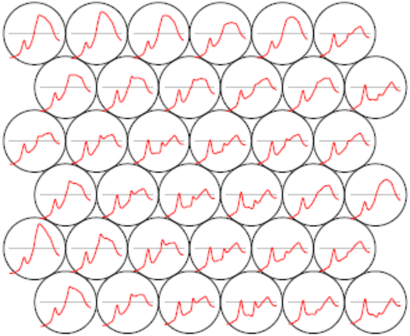
SOM



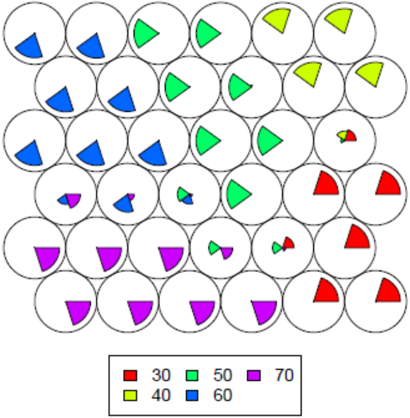

Wine data
alcohol
malic acid
ash
ash alkalinity
magnesium
tot. phenols
flavonoids
non−flav. phenols
proanth
col. int.
col. hue
OD ratio
proline

SOM

X



Y

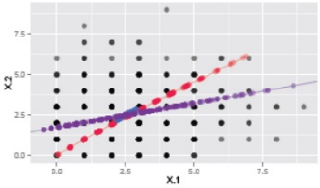


UNIVERSIDAD DEL VALLE DE GUATEMALA

Otros métodos

Probabilistic PCA:
Hacer PCA en el espacio de parámetros de la distribución. Considermaso $\mathbb{X} = [\theta_{ij}]$ ó $\mathbb{X} = [g(\theta_{ij})]$ asociados a una muestra $[X_{ij}$ de v.a. independientes con distribuciones cualquiera.
Hacer $[\theta_{ij}] = USV^T$.

Ejemplo Poisson PCA:



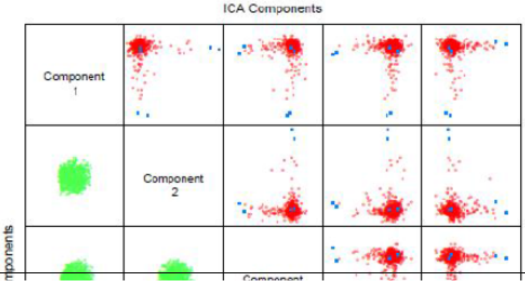
(e) $n = 500, \lambda \in (2.16, 2.90)$

Otros métodos

Projection Pursuit:

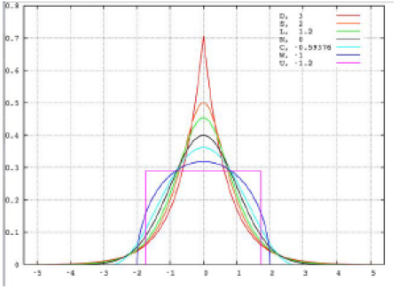
Similar a PCA. En lugar de buscar la dirección ℓ de máxima varianza, usamos otra medida de proyección óptima.

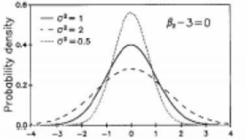
Buscar direcciones que maximicen la no gaussianidad (caracterizamos la gaussiana en términos de la entropía). Por ejemplo, buscamos ℓ tal que la negentropía de $\ell^T\mathbf{x}$ sea máxima. (Similar a ICA)



Camino alternativo: usar Kurtosis (peakedness)

$$Kurt_N(X) = \frac{E(X - EX)^4}{Var(X)^2} \qquad Kurt(X) = E(X - EX)^4 - 3Var(X)^2$$

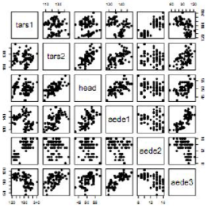


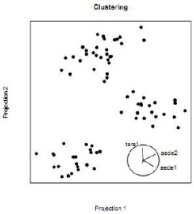


Otros métodos

<u>Métodos aleatorios y *grand tour*</u>:

Hacer una caminata aleatoria (película) con proyecciones que cambian suavemente.





UNIVERSIDAD DEL VALLE DE GUATEMALA

Recursos en Python

- `sklearn.decomposition`: Contiene los métodos de PCA, KernelPCA, NMF, FastICA, y contiene otros similares como LDA, FactorAnalysis, DictionaryLearning.
- `sklearn.manifold`: Contiene métodos de *manifold learning*: Isomap, t-SNE, Local Lineal Embedding (y variantes de LLE: modified LLE, Hessian LLE, LTSA LLE), MultiDimensionalScaling (MDS), SpectralEmbedding.
- `tensorflow` y `pytorch`: Librerías para redes neuronales, en particular auto-encoders.
- Software *ggobi*: Tiene importantes herramientas para visualización. Contiene el método de *grand tour*.
- `SimpSOM`, `MiniSom`, `SOMPy`, `kohonen`: Librerías con implementaciones de SOM. (pip install ...)