

MÉTODOS LOCALES II

ALAN REYES-FIGUEROA

INTRODUCCIÓN A LA CIENCIA DE DATOS

(AULA 18) 14.MARZO.2022

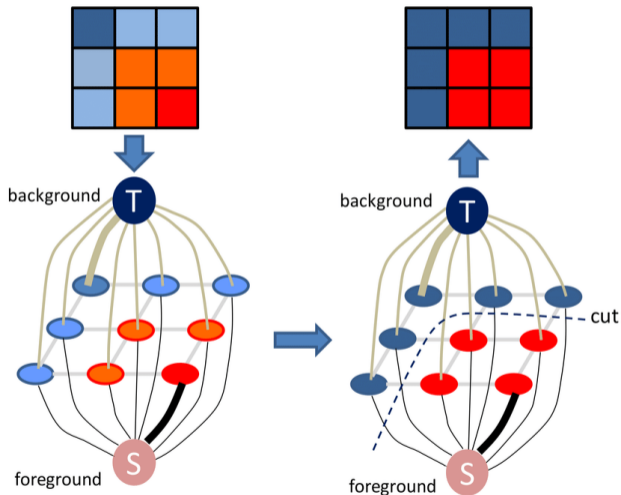
Spectral Embedding

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

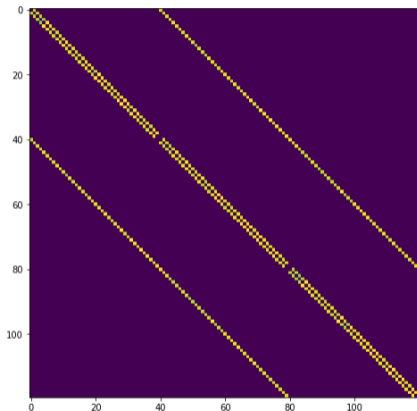
Idea: Se construye una matriz de adyacencia o de afinidad (similaridad) W entre una estructura de grafo entre los datos. Los elementos w_{ij} de W pesan o miden el grado de afinidad.

- 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.

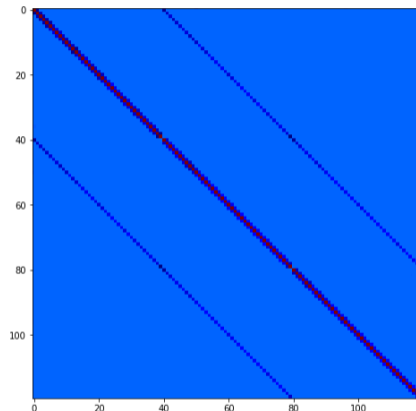
Spectral Embedding



Spectral Embedding



Matriz de afinidad W



Laplaciano normalizado \mathcal{L}

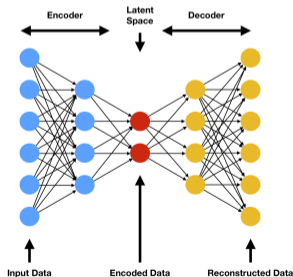
Autoencoders

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

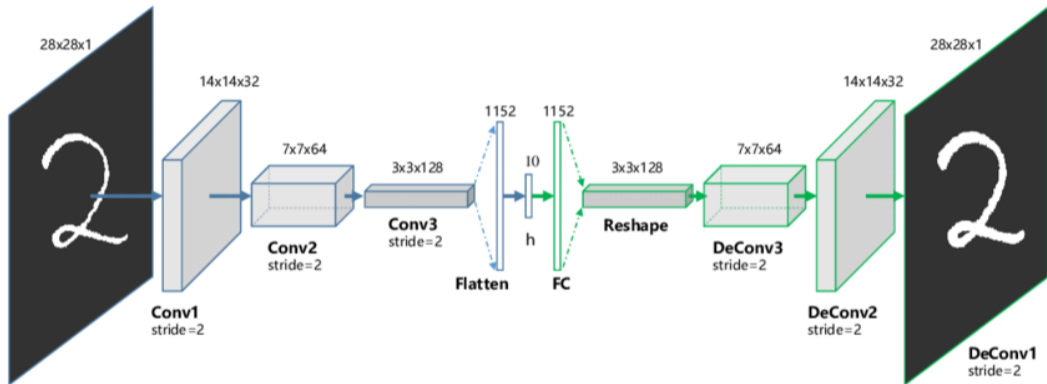
\mathcal{E} se llama el *encoder*, y \mathcal{D} el *decoder*. El objetivo es resolver

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

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.

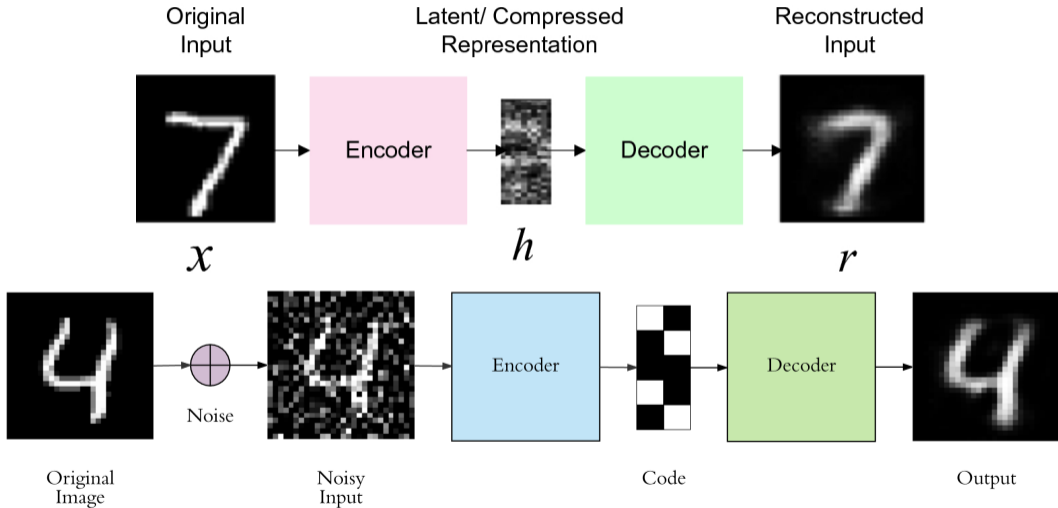


Autoencoders



Red neuronal convolucional profunda (CNN).

Autoencoders



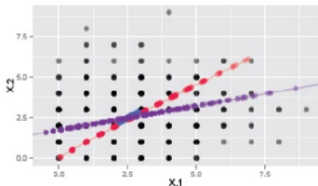
Otros métodos

Probabilistic PCA:

Hacer PCA en el espacio de parámetros de la distribución. Consideramos $\mathbb{X} = [\theta_{ij}]$ 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:



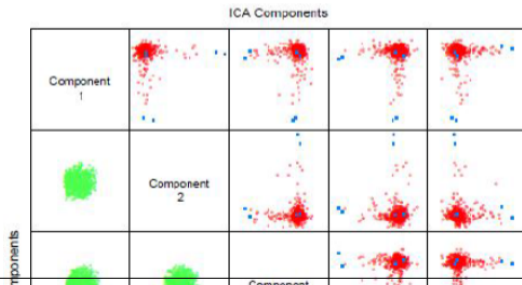
(e) $n = 500, \lambda \in (2.16, 2.90)$

Otros métodos

Projection Pursuit:

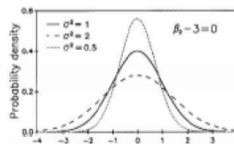
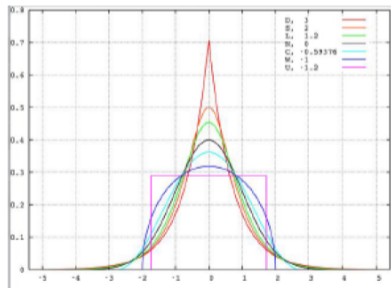
Similar a PCA. En lugar de buscar la dirección ℓ de máxima varianza, usamos otra medida de proyección óptima.

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)

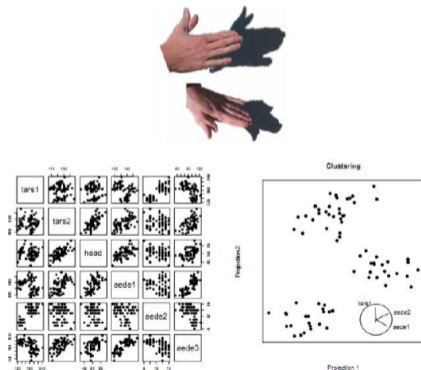
$$Kurt_N(X) = \frac{E(X - EX)^4}{Var(X)^2} \quad Kurt(X) = E(X - EX)^4 - 3Var(X)^2$$



Otros métodos

Métodos aleatorios y *grand tour*:

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



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 Linear 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, SOMPpy, kohonen: Librerías con implementaciones de SOM. (pip install ...)