Módulo 5: Aprendizaje no supervisado - 2do encuentro

Diplomatura Cs. de Datos - FaCENA-UNNE

Docentes: Magdalena Lucini, Luis Duarte, Griselda Bóbeda

Aprendizaje no supervisado: resumen 1^{er} clase

Reducción de dimensionalidad: PCA clásico

Reducción de dimensionalidad

Objetivos:

- Reducir la dimensionalidad del conjunto de datos cuando el número de características (variables, features) es elevado, o son redundantes, preservando la mayor cantidad de información posible.
- Proyectar los datos a otro espacio (quizás de menor dimensión) para poder separarlos

En este curso veremos:

- Análisis de Componentes Principales (ACP, PCA) ✓
- Análisis de Componentes Independientes (ICA)
- Embeddings: tSNE (t-distributed Stochastic Neighbor Embedding), UMAP (Uniform Manifold Approximation and Projection)

PCA clásico (clase 1): resumen

Problema

Encontrar un espacio de dimensión más reducida que represente adecuadamente los datos y brinde la mejor representación de la variabilidad y diversidad de los mismos.

Objetivos

- Encontrar una proyección lineal de los datos de manera tal que la varianza en el espacio proyectado sea máxima
- Reducir dimensionalidad describiendo las p variables de una matriz X por un subconjunto (pequeño) $r < p$ de combinaciones lineales de las variables originales.
- El error en la reconstrucción de los datos a partir de esas r combinaciones lineales (Componentes principales) es mínimo.

PCA

Dado un conjunto de datos $\mathrm{X} = [\mathrm{X}_1, \ldots, \mathrm{X}_p] \in \mathbb{R}^{n \times p}$, donde las $\{\mathrm{X}_i\}_{i=1}^{p}$ son las va

- Las CP son $\mathrm{Z}_j = \mathrm{X}_c \mathrm{u}_j,\ j = 1, \ldots, p$
- $\mathrm{Z}_j = u_{1j}\mathrm{X}_1 + \ldots + u_{pj}\mathrm{X}_p$ es combinación lineal de las p variables (features, características) originales $\mathrm{X}_1, \ldots, \mathrm{X}_p$.
  Los u_{ij} son las cargas (loadings) de la variable X_i sobre la componente principal Z_j.
- La proyección de X sobre el nuevo espacio generado por los r primeros autovectores es

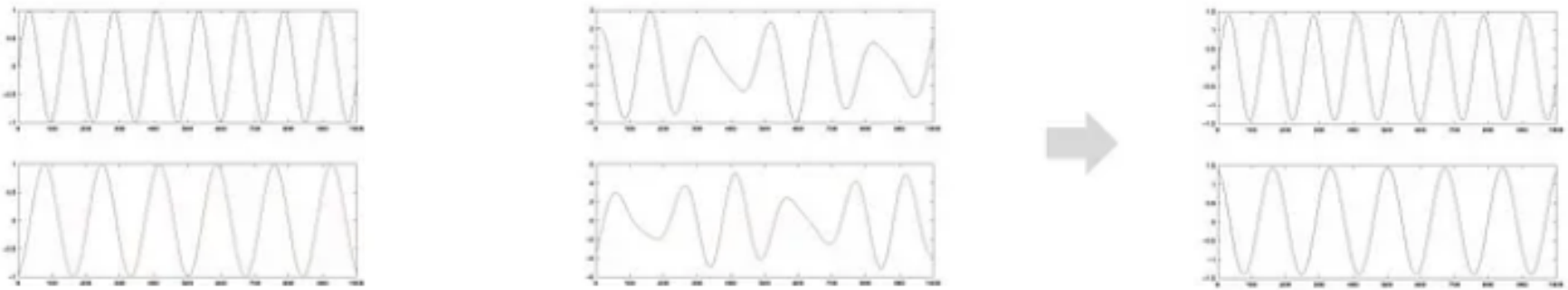
$$\mathrm{Z} = \mathrm{U}_r^t(\mathrm{X}_c)$$

  , con X_c datos centrados.

Situación 1: "Cocktail Party Problem"

Figura: Jonas Dieckman, towardsdatascience.com

Señales originales s_i **Mediciones** x_i ¿Cómo recuperar s_i?



Análisis de Componentes Independientes (ICA)

Si se asume que las señales observadas x_i, $i = 1, \ldots, p$ son una "mezcla" (combinación lineal) de señales independientes (latentes) s_i, $i = 1, \ldots, n$, ¿cómo recuperar esas señales originales (y desconocidas) s_i?

Independent Component Analysis (ICA):

- Técnica multivariada NO supervisada, que asume que las variables observadas es la combinación lineal de múltiples señales (fuentes, variables) latentes independientes.
- Objetivo: Encontrar las señales independientes originales (o lo más independientes que se pueda).
- Supuestos:
  - Las señales originales son No Gaussianas
  - Las señales originales son independientes

Situación 1: Cocktail Party Problem

Observaciones:

$\mathrm{x}_1(t) = a_{11}\mathrm{s}_1(t) + a_{12}\mathrm{s}_2(t)$

$\mathrm{x}_2(t) = a_{21}\mathrm{s}_1(t) + a_{22}\mathrm{s}_2(t)$

- $\mathrm{s}_i, i = 1, 2$ independientes
- $\mathrm{s}_i, i = 1, 2$ no gaussianas

Marco general:

$$\mathbf{x} = \mathrm{A}\mathbf{s}$$

- $\mathbf{x}$: observaciones
- $\mathrm{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ matriz de mezcla (NO observable), a_{ij} coeficientes de mezclado.
- $\mathbf{s}$ Variables (señales, fuentes) originales (NO observables)

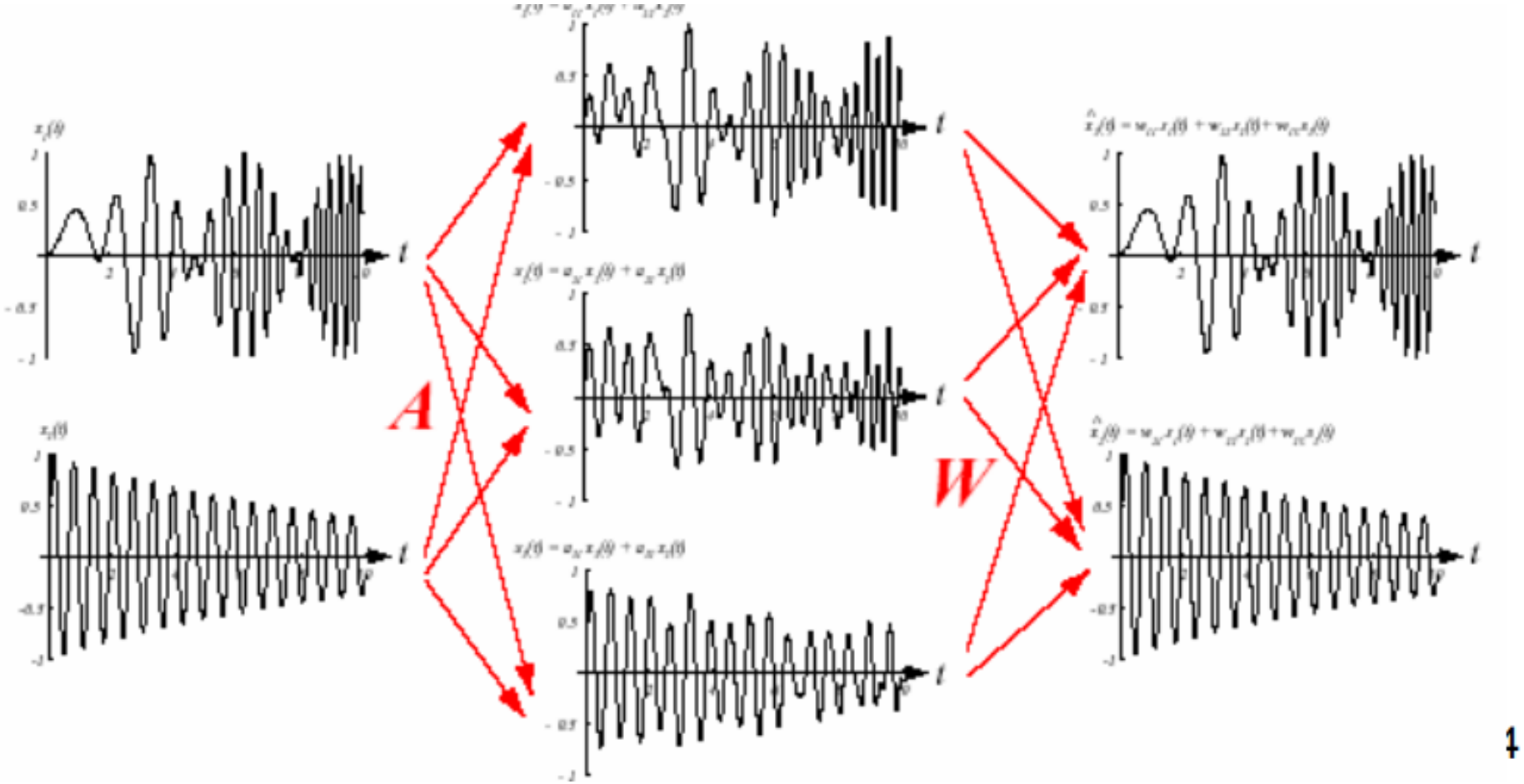
Objetivo

recuperar las conversaciones originales s_i.

Si conociéramos $\mathrm{A} \Rightarrow \mathbf{s} = \mathrm{A}^{-1}\mathbf{x}$ (siempre y cuando exista A^{-1}).

A es no observable

ICA: Idea intuitiva



A matriz de mezcla, W matriz de separación

ICA: Marco teórico

ICA:Técnica enfocada en la separación no supervisada (ciega) de fuentes (variables, señales) que han sufrido un proceso de mezclado (lineal).

$$x_i(t) = \sum_{j=1}^{N} a_{ij} s_j(t)$$

- $x_i(t), i = 1, \ldots, M$ son M mezclas
- $s_i(t), i = 1, \ldots, N$ son N fuentes originales.
- a_{ij} coeficientes de mezclado, t índice de tiempo (posición).

Matricialmente:

$\mathbf{x}(t) = \mathrm{A}\mathbf{s}(t)$

- $\mathbf{x}(t) = [x_1(t), \ldots, x_M(t)]^t$
- $\mathbf{s}(t) = [s_1(t), \ldots, s_N(t)]^t$
- A matriz de mezclado $\in \mathbb{R}^{\mathbb{M}\times\mathbb{N}}$ formada por los a_{ij}

A y $\mathbf{s}$ desconocidos. Sólo se tiene acceso a las mezclas $\mathbf{x}(t)$

ICA: Marco teórico

Supuestos ICA para recuperar fuentes **s**

- En general, se asume $M = N$
- Fuentes s_i independientes entre sí.
- Fuentes s_i NO gaussianas.

ICA: recuperación de fuentes $s_i(t)$

Se propone el modelo

$$\mathbf{y}(t) = W\mathbf{x}(t)$$

- $\mathbf{y}(t) = [y_1(t), \ldots . y_N(t)]$ son las fuentes originales estimadas = COMPONENTES INDEPENDIENTES
- W matriz de recuperación o desmezclado.

En caso de perfecta recuperación $W = A^{-1}$

ICA: Marco teórico

Supuestos:

1. Las CI $y_i(t)$ se suponen estadísticamente independientes.
2. $y_i(t)$ son no gaussianas
3. La matriz de mezclado A es cuadrada.

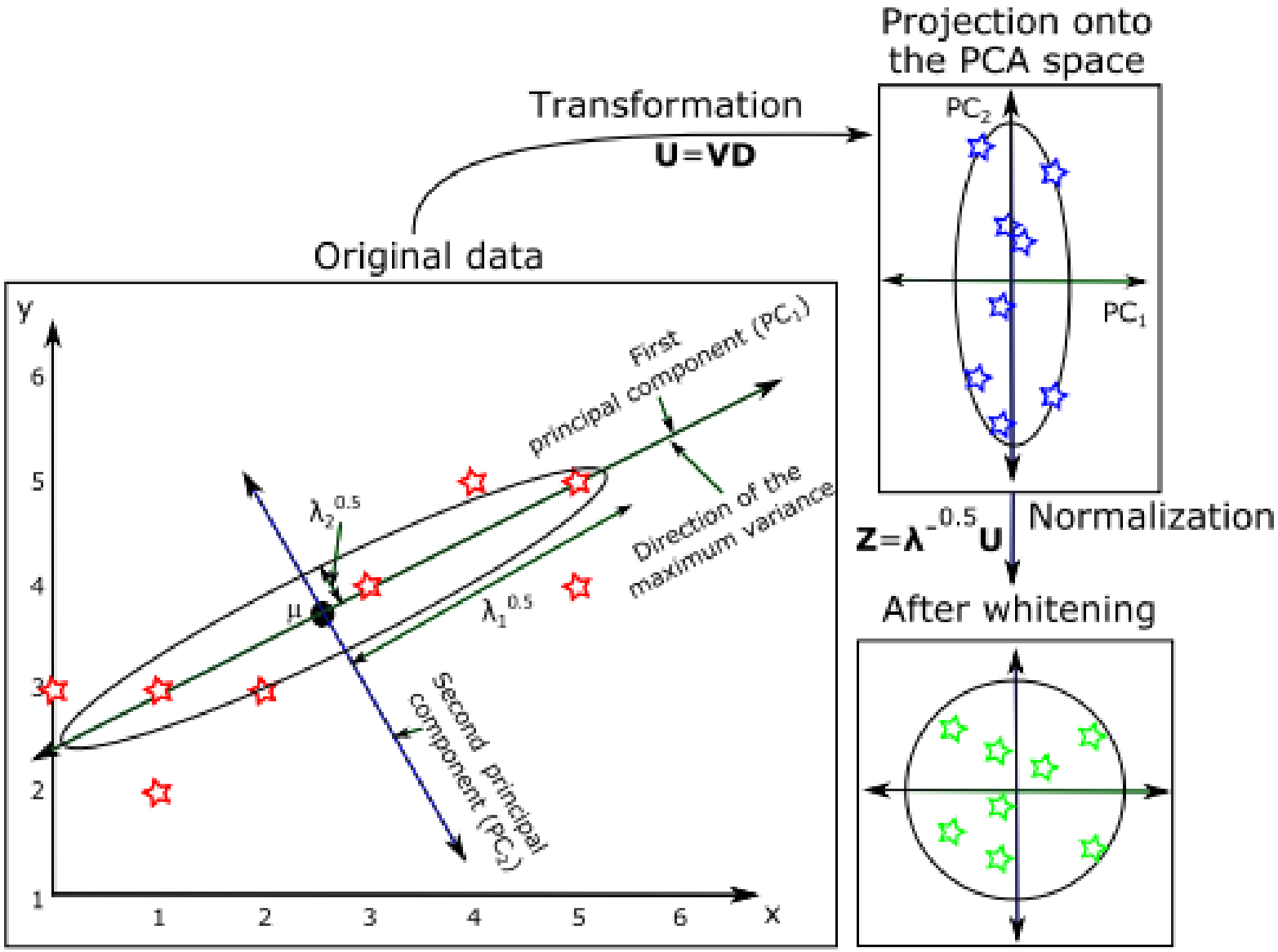
Preprocesamiento:

1. Centrado:Si $\bar{x}$ es el vector de medias, los datos centrados son

$$\mathbf{x}_c(t) = \begin{pmatrix} x_1(t) - \bar{x}(t) \\ \vdots \\ x_N(t) - \bar{x}(t) \end{pmatrix}$$

2. Blanqueado:Transformar las señales (centradas) en señales no correlacionadas y luego rescalar para que tengan varianza unitaria:
   1. Decorrelación : PCA, $U = V\mathbf{x}_c$
   2. Escalado: cada vector en U se rescala para que tenga varianza 1.

Preprocesamiento



ICA: Estimación de W

Objetivo: Encontrar la matriz de desmezclado W. Luego proyectar los datos blanqueados sobre esa matriz para luego extraer las componentes independientes $\mathbf{y} = W\mathbf{x}_c$.

Métodos para estimar W

- Medidas de no-Gaussianidad: Encontrar CI que maximizan la No-Gaussianidad (usando como medidas de no -gaussianidad la kurtosis o la **negentropía**)
- Minimización de la Información Mutua: La "mutual information" es una medida de la dependencia entre variables- Se busca minimizar la información mutua entre componentes, minimizando la divergencia de Kullback-Leibler entre densidades conjuntas y productos de marginales.
- Máxima Verosimilitud: Se usa el método de MV para encontrar la matriz W.

Algunas consideraciones

- La etapa de preprocesamiento se hace directamente sobre los datos, pero W se calcula numéricamente con algún método de optimización.
- Los métodos propuestos para la estimación de W resultan en matrices levemente distintas.
- Estos métodos están relacionados, y los resultados difieren en el signo o en un factor constante.
- Ambiguedades del ICA:
  - Orden de las CI: De acuerdo al método elegido en la estimación de W sus componentes serán rotadas y actualizadas iterativamente. Se recuperan las fuentes, pero no en un orden específico.
  - Signo de las CI: el signo de las CI no afecta el modelo.

Algoritmos para ICA

Uno de los más usados es FastICA [1], implementado en `sklearn.decomposition`

sklearn.decomposition.FastICA

```
class sklearn.decomposition.FastICA(n_components=None, *,
algorithm='parallel', whiten='unit-variance',
fun='logcosh', fun_args=None, max_ iter=200, tol=0.0001,
w_init=None, whiten_solver='svd', random_state=None)
```

[1] A. Hyvarinen and E. Oja, Independent Component Analysis: Algorithms and Applications, Neural Networks, 13(4-5), 2000, pp. 411-430

Ejemplos: Cuaderno Collab Práctico ICA

Situación 2: Zeizel Brain Dataset

- Base de datos que contiene tipos celulares en el cerebro de ratón (neuronas, oligodendrocitos, microglia, etc) [*]
- 3005 muestras (datos, observaciones), sobre las que se registraron 19972 "features" (expresiones de rna)

Objetivos: reducir dimensionalidad, visualizar la estructura en un espacio bi o tridimensional.

[*]Zeisel, A. et al. Brain structure. Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. Science 347, 1138–1142 (2015)

t-SNE

t-SNE: t distributed Stochastic Neighbor Embbeding [2]
Método no lineal para reducción de la dimensionalidad, usado principalmente para visualizar datos de alta dimensión proyectándolos en un espacio de dos o tres dimensiones preservando las relaciones locales entre puntos.

- Embedding:proyecciones, inscrustaciones.
- Neighbor Embedding: hace referencia a que observaciones"similares" (en un sentido probabilistico) en el espacio de alta dimensión son "similares" (probabilísticamente) al ser proyectadas al espacio de menor dimensión.
- Stochastic: (aleatorio) Hace referencia al proceso aleatorio por el cual el algoritmo proyecta las observaciones en un espacio de baja dimensión.
- t-distributed: distribución usada para determinar similaridades entre puntos en el espacio de baja dimensión

[2] Van DerMaaten and Hinton,Journal of Machine Learning Research 9 (2008) 2579-2605

Pasos de t-SNE

Principio de t-SNE: Se construye una distribución de probabilidad que represente las "similitudes" entre observaciones vecinas en un espacio de gran dimensión, y en un espacio de baja dimensión.
Sea $X = \{x_1, \ldots, x_n\}$ dataset de observaciones p-dimensionales.

Paso 1: Similitudes en espacio de alta dimensión

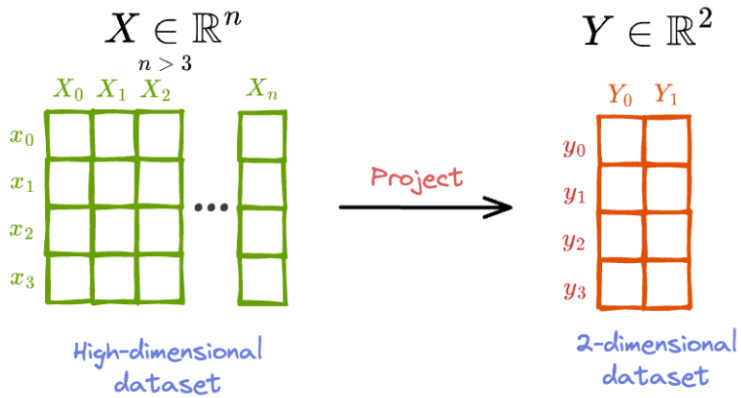
- Para cada punto x_i se propone una distribución gaussiana centrada en ese punto
- Para todo otro punto $x_j, j \neq i$ se calculan las probabilidades condicionales

$$p_{j/i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)}$$

- Se "simetrizan" las probabilidades definiendo $p_{ij} = \frac{p_{j/i} + p_{i/j}}{2n}$

σ_i se define por un valor llamado perplejidad que corresponde al número de vecinos alrededor de ese punto

Interpretación geométrica Paso 1:



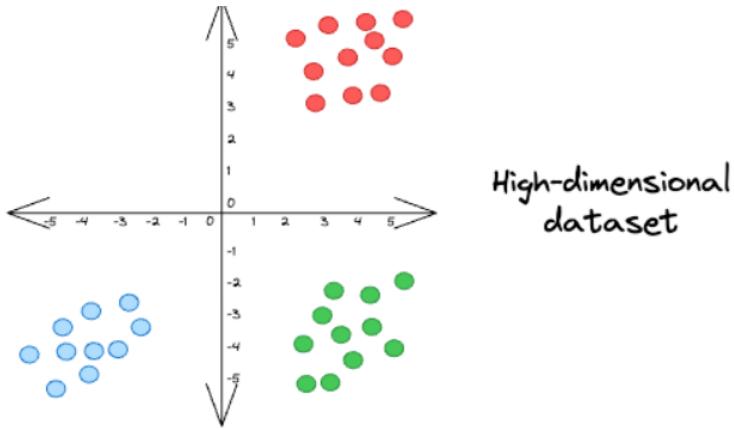


Figura: en espacio menor dimension puntos similares deberían ser similares

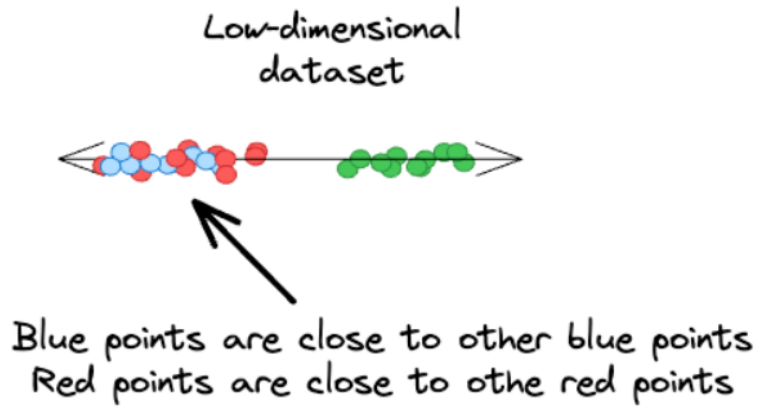


Figura: Asi no!!: las proyecciones en baja dimensión deben capturar la estructura global

Interpretación geométrica Paso 1:

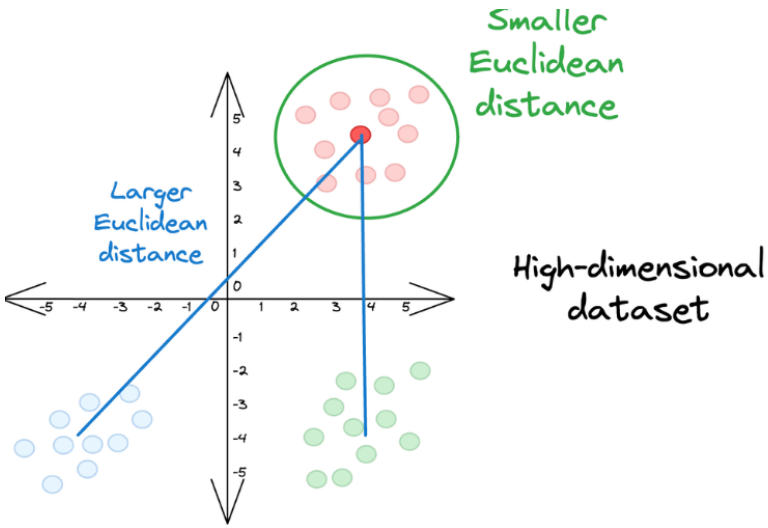


Figura: Se definen las $p_{j/i}$

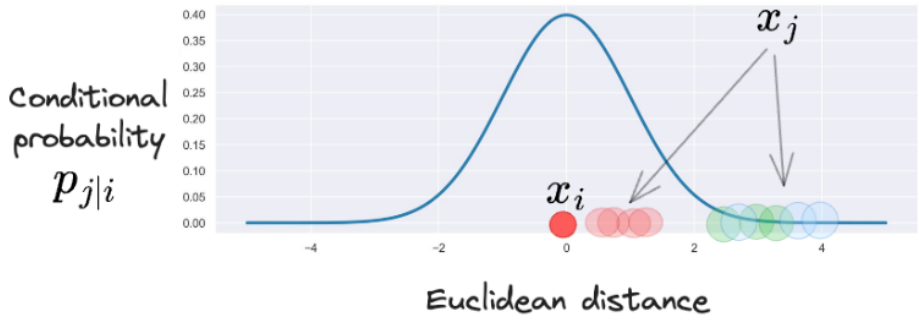


Figura: Si $p_{j/i}$ es alta, entonces punto i selecciona al punto j como vecino

- $p_{j/i}$ será relativamente alta para puntos cercanos a x_i
- $p_{j/i}$ será pequeña para puntos lejanos a x_i

t-SNE Paso 2: Similaridades en espacio de baja dimensión

Se debe crear un espacio de muy baja dimensión donde representar los datos, preservando estructuras locales. En este nuevo espacio , $Y = \{y_1, \ldots, y_n\}$

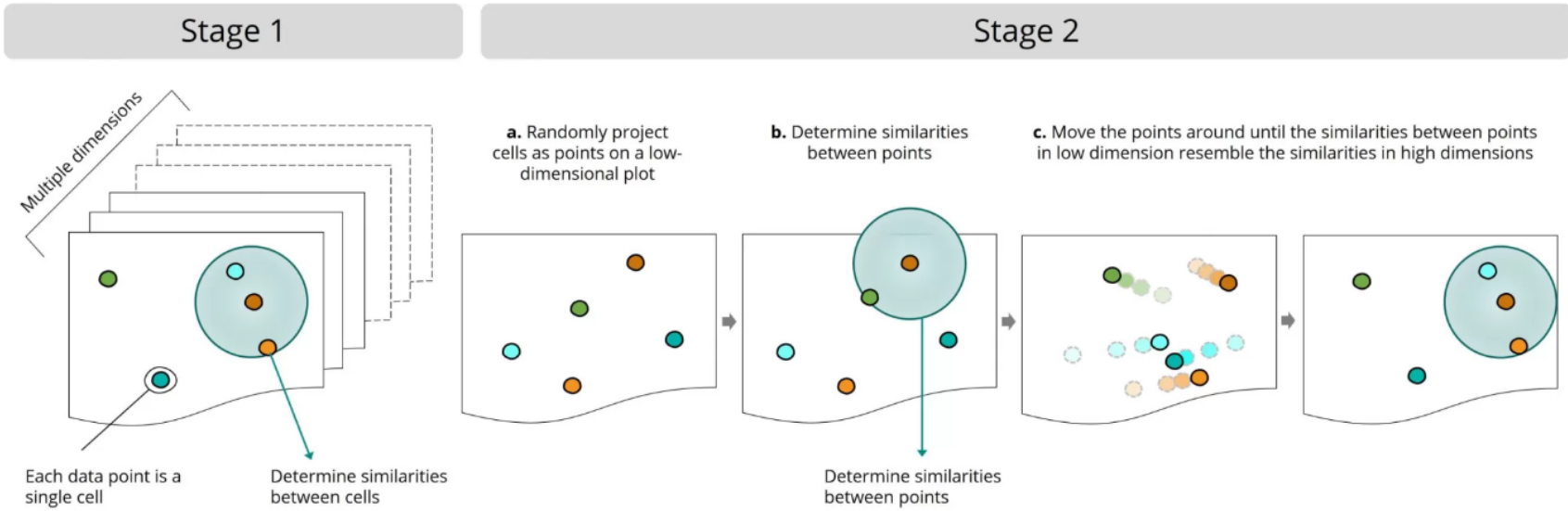
- Inicialmente se distribuyen los puntos aleatoriamente en este espacio, pues no se conocen coordenadas ideales.
- Se calculan las "similitudes" entre los puntos del nuevo espacio de manera análoga a lo hecho en Paso 1, pero usando una distribución t-Student.
- Se obtiene una nueva lista de probabilidades: $q_{j/i} = \frac{(1+||y_i - y_j||^2)^{-1}}{\sum_{k \neq i}(1+||y_i - y_k||^2)^{-1}}$

Objetivo: Obtener probabilidades $q_{j/i}$ (espacio baja dimensión), idénticas a $p_{j/i}$ (espacio alta dimensión) (si las proyecciones y_i e y_j modelaran correctamente la similitud entre los puntos x_i y x_j entonces $q_{j/i}$ y $p_{j/i}$ deberian ser casi iguales.

t-SNE Paso 3: representación en el espacio de menor dimensión

- Se comparan las similitudes de los puntos en los dos espacios usando la divergencia de Kullback-Leibler (KL)
- Se minimiza esa medida usando un método de gradiente descendiente para encontrar el "mejor" y_i en el espacio de menor dimensión. Esto se optimiza hasta alcanzar un estado estable o un número máximo de iteraciones.
- Este proceso permite la creación de conglomerados de puntos de datos similares en el espacio de baja dimensión.
- Se visualizan estos puntos para comprender la estructura y la relación de los datos en el espacio de alta dimensión.

t-SNE: Esquema general



Situación 2:t-SNE aplicado a dataset Zeizel

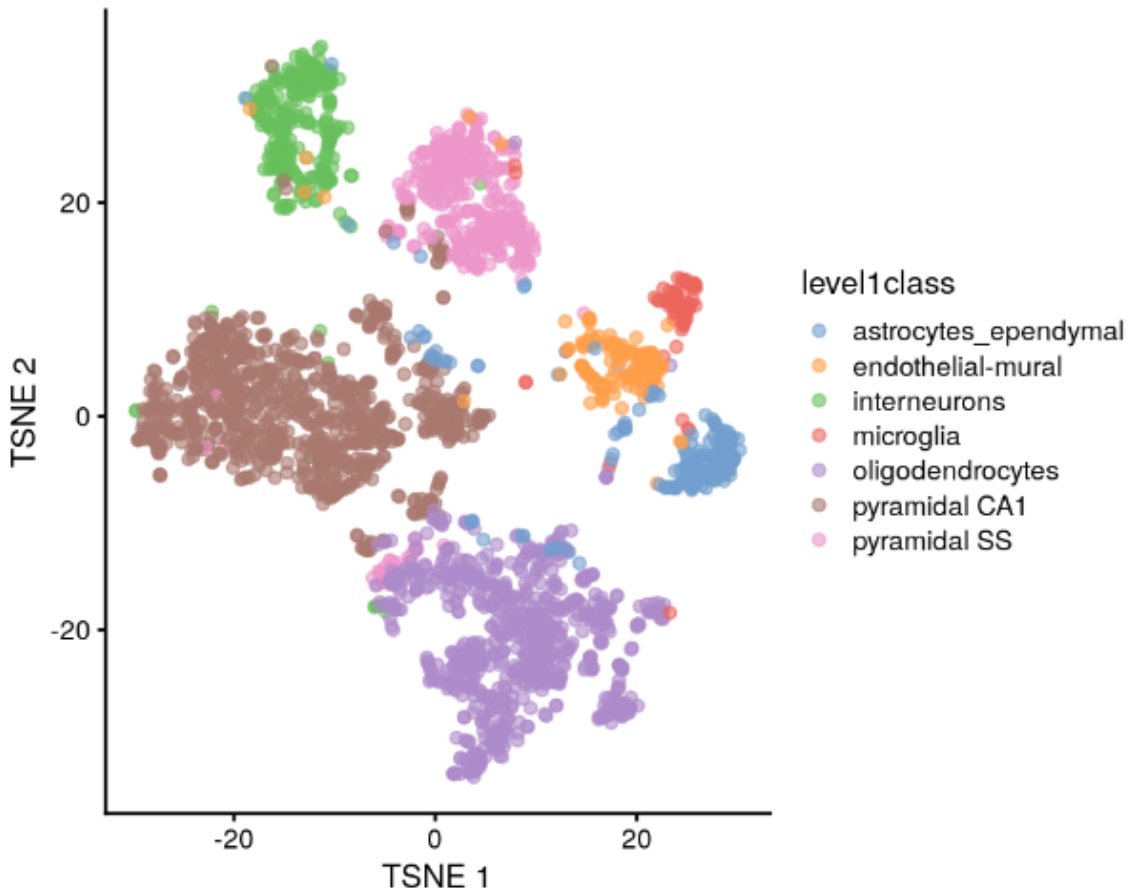


Figura: Perplejidad = 30

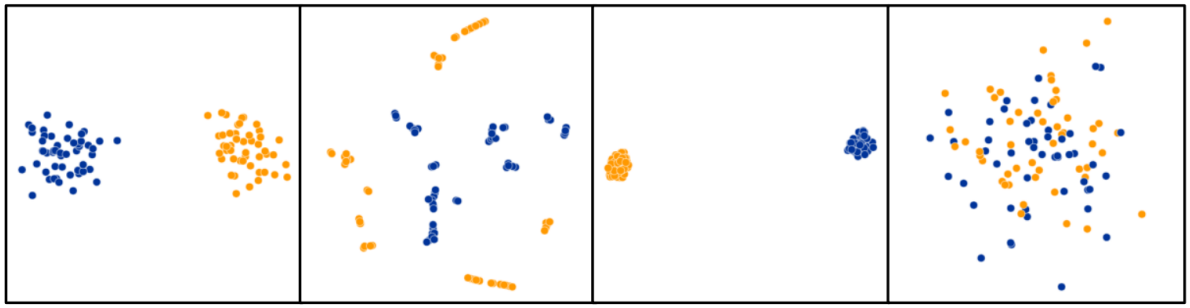
tSNE

tSNE en sklearn

```
 class sklearn.manifold.TSNE(n_components=2, *,
perplexity=30.0, early_exaggeration=12.0,
learning_rate='auto', max_iter=None,
n_iter_without_progress=300, min_grad_norm=1e-07,
metric='euclidean', metric_params=None, init='pca',
verbose=0, random_state=None, method='barnes_hut',
angle=0.5, n_jobs=None, n_iter='deprecated')
```

Hiperparámetros tSNE

- Perplejidad: Es una medida del número efectivo de vecinos para cada punto. Su valor afecta el balance entre la estructura local y global de los datos (muy pequeño enfatiza estructura local, muy grande involucra estructura global).
  Valores típicos varían entre 5 y 50 , dependiendo del dataset. Se sugiere probar varios y ver cómo afecta a los resultados.
- Learning rate: Tamaño del paso en cada iteración cuando se busca minimizar cierta función de costo. Si es muy grande el algoritmo oscilará y quizas no encuentre mínimo global, si es muy pequeño puede quedarse en un mínimo local. Valores sugeridos entre 10 y 1000.
- Número iteraciones: Cantidad de iteraciones debe hacer el algoritmo.



https://distill.pub/2016/misread-tsne/

Figura: Efecto perplejidad

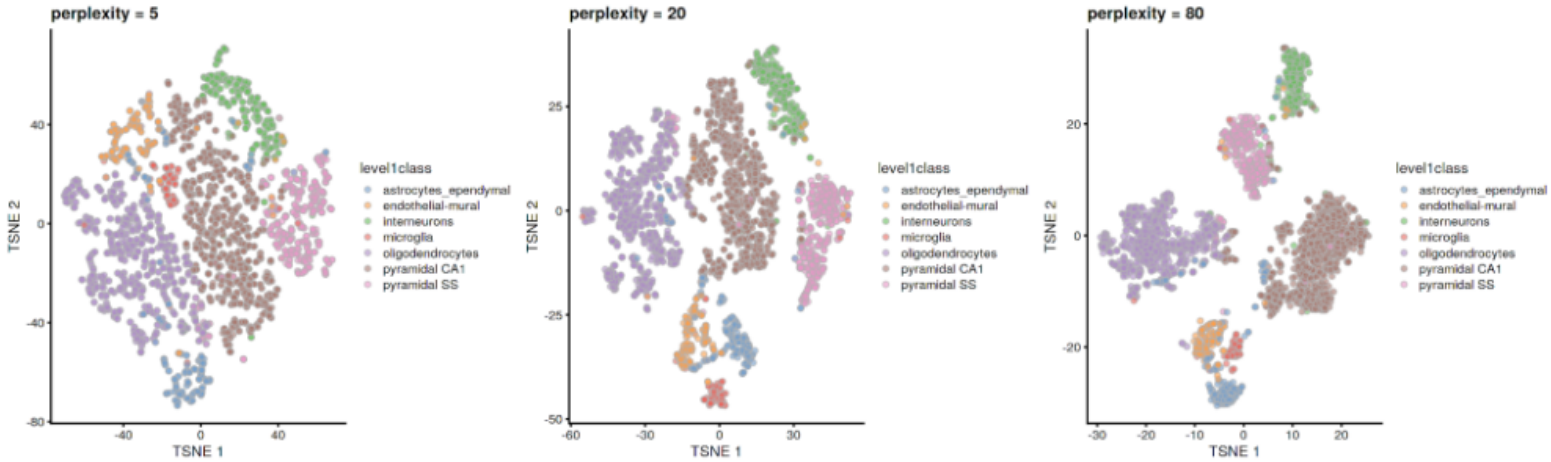


Figura: Efecto perplejidad en dataset Zeizel

Consideraciones

- Ejes: Los ejes no tienen sentido, no deben interpretarse. tSNE se usa para visualización
- reproducibilidad: Como se inicia aleatoriamente, los resultados serán distintos cada vez que se ejecute.
- Normalización: El preprocesamiento de los datos por escalado o normalización, especialmente si las variables se miden en escalas diferentes, tiene un impacto significante en los resultados.
- Dimensionalidad: Si el número de variables (features) es muy grande, se sugiere hacer un PCA previo al tSNE
- Desventajas: Computacionalmente caro, no es bueno para preservar estructurs globales, sensible a los hiperparámetros, puede quedarse oscilando en un mínimo local.

Ejemplos: Cuaderno Collab práctico

UMAP

UMAP:Uniform Manifold Approximation and Projection [3]

- Uniform: Asume que los datos se distribuyen uniformemente sobre una variedad, esto es, datos uniformemente distribuidos, pero que la distancia entre puntos varía sobre esa variedad. Usa distancias para calcular similaritud entre datos.
- Manifold: la "forma" de los datos. UMAP la aprende para determinar las similitudes entre puntos.
- Approximation: el algoritmo debe aproximar la "forma" de los datos para determinar similitudes entre ellos.
- Projection:una vez determinadas las similitudes, UMAP proyecta los datos como puntos en un espacio 2D o 3D. Los mueve hasta que captura las similitudes existentes en el espacio de alta dimensión

[3]McInnes, L, Healy, J, UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, ArXiv e-prints 1802.03426, 2018

UMAP

UMAP:Uniform Manifold Approximation and Projection

- Algoritmo no supervisado para reducción no lineal de dimensionalidad (visualización)
- Busca aprender la variedad estructural de los datos (en el sentido topológico) en el espacio de alta dimensión para luego encontrar una proyección (incrustación) en un espacio de baja dimensión , preservando la estructura topológica esencial de la variedad original.
- Marco teórico basado en geometría riemaniana y topología algebraica.
- Busca aprender la variedad estructural de los datos (en el sentido topológico) en el espacio de alta dimensión para luego encontrar una proyección (incrustación) en un espacio de baja dimensión, preservando la estructura topológica esencial de la variedad original.
- Preserva estructuras globales y locales.

Pasos UMAP

- Paso 1: Construción de un grafo de los puntos $x_1, \ldots, x_n$ en el espacio de alta dimensión.
- Paso 2: Mapea de manera optima (método de gradiente descendiente estocástico + Cross Entropy entre distribuciones datos en espacio alta dimension y baja dimensión) ese grafo en un espacio de menor dimensión, su estructura debe ser tan similar como sea posible a la del grafo en el espacio de alta dimensión.

UMAP: Paso 1

- Se calculan las distancias entre todos los puntos de datos en el espacio de alta dimensión
- Se construye un grafo inicial en el que cada punto es un nodo y las aristas del grafo conectan puntos “cercanos “ entre sí.
- Similaridades : $v_{j/i} = e^{-\frac{d(x_i,x_j)-\rho_i}{\sigma_i}}$
- fuzzy: la verosimilitud que dos puntos estén conectados decrece a medida que el radio aumenta.
- Estipulando que cada punto debe estar conectado con al menos su vecino más cercano, se asegura de preservar de manera balanceada estructuras globales y locales.

UMAP: Paso 1:

- ρ_i= mínima distancia deseada entre puntos cercanos en el espacio de baja dimensión (hiperparámetro)
- σ_i es la solución de

$$\sum_{j=1}^{k} exp\left(-\frac{max(0, d(x_i, x_j)^2 - \rho_i)}{\sigma_i}\right) = log_2(k)$$

- k= número de vecinos cercanos (hiperparámetro)

UMAP Paso1

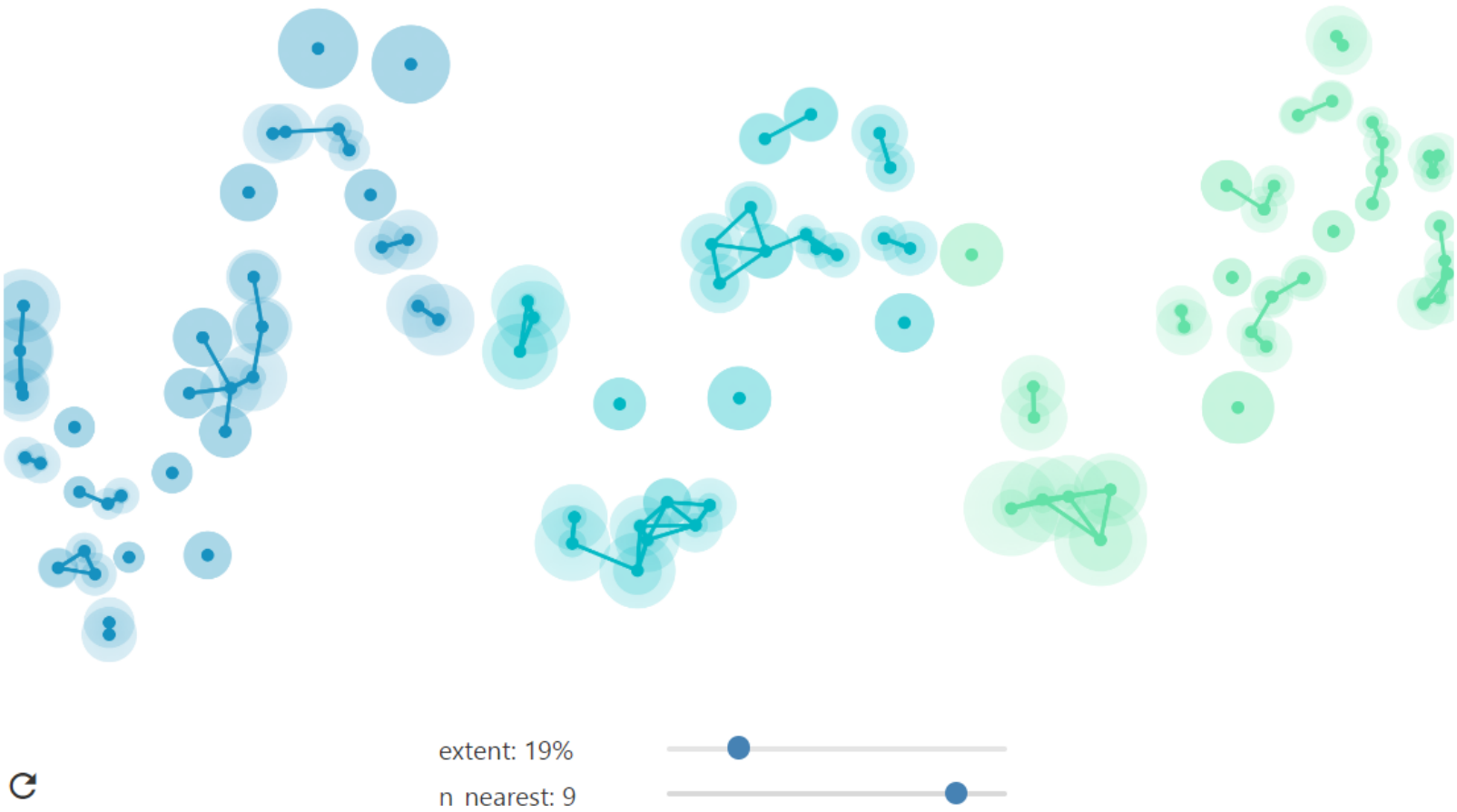


Figura: UMAP construcción grafo

UMAP: Paso 2

- Una vez construido el grafo en el espacio de alta dimensión, UMAP "optimiza" la construcción de un grafo "análogo" en el espacio de baja dimensión usando un método de gradiente descendiente estocástico .
- Similaridades en este espacio : $w_{ij} = (1 + a||y_i - y_j||^{2b})^{-1}$, a y b se eligen por un ajuste de mínimos cuadrados.

UMAP en dataset Zeizel

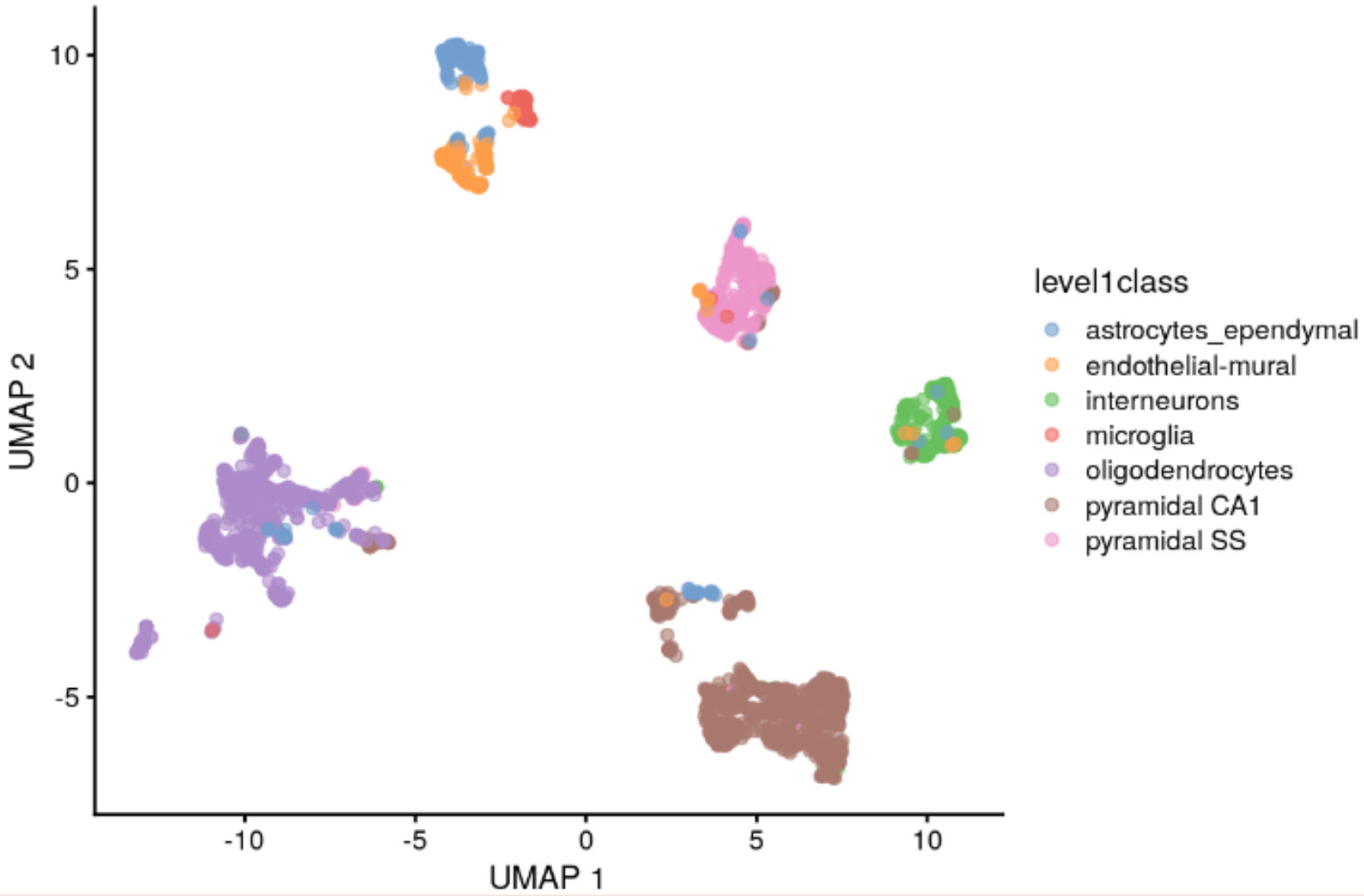


Figura: resultado de aplicar UMAP en el dataset Zeizel, n vecinos = 15 , min_dist = 0.1

Consideraciones

- UMAP ofrece una serie de ventajas sobre tSNE: es más veloz, preserva balance entre esructuras globales y locales.
- La elección de los hiperparámetros afecta a los resultados, por lo que debe ejecutarse con distintas configuraciones.
- Tamaño de grupos de puntos y distancias entre grupos distintos no tienen ningún significado

Ejemplos: Cuaderno Collab práctica.

Consideraciones

PCA

- Propósito:reducción de dimensionalidad
- Algoritmo: Determinístico
- Solución única?: si
- Proyección: lineal
- Interpretación: rotación de ejes.
- Bueno para datos linealmente separables

t-SNE

- Propósito: visualización
- Algoritmo: Estocástico
- Solución única?: no
- Proyección: no lineal
- Interpretación: subjetivo.
- Preserva estructuras locales

UMAP

- Propósito: visualización
- Algoritmo: Estocástico
- Solución única?: no
- Proyección: no lineal
- Interpretación: subjetivo.
- Preserva estructuras locales y globales.

Comparación PCA, tSNE, UMAP usando dataset Zeizel

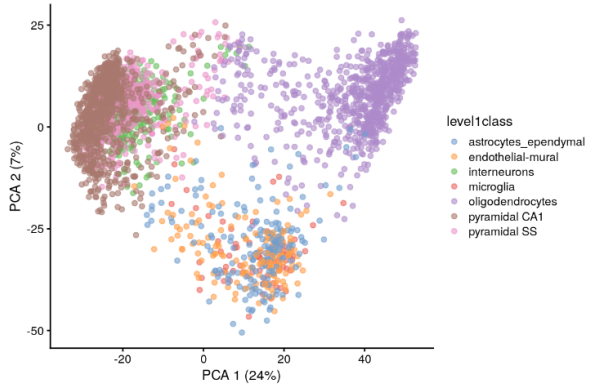


Figura: PCA

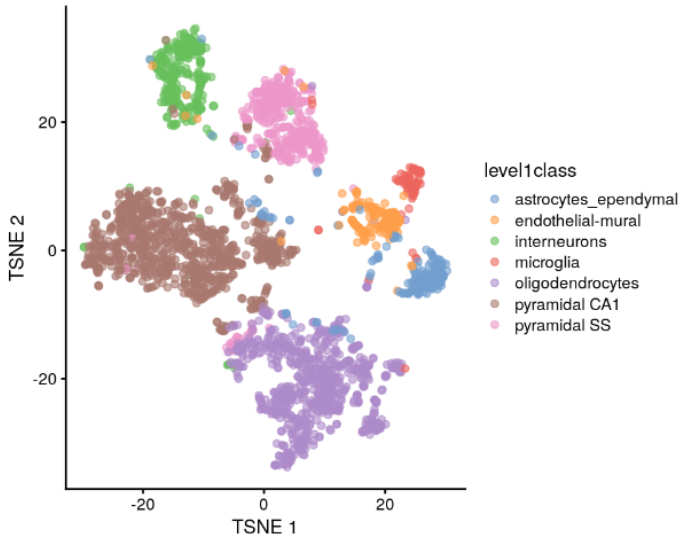


Figura: tSNE, p=30

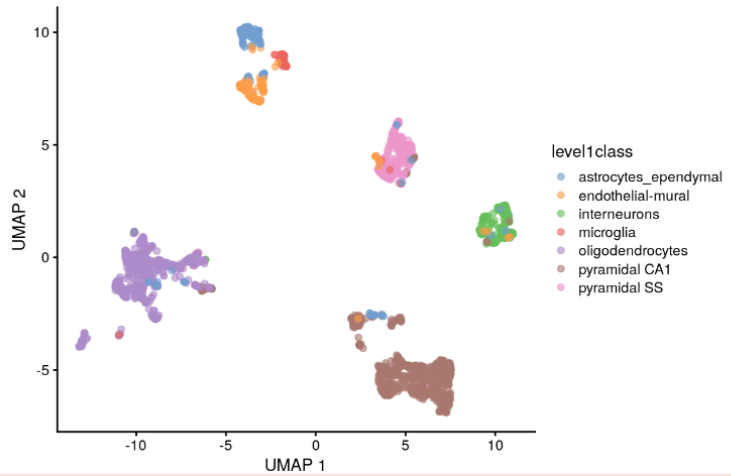


Figura: UMAP, n vecinos = 15, min_dist = 0.1

Bibliografía

[1] A. Hyvarinen and E. Oja, Independent Component Analysis: Algorithms and Applications, Neural Networks, 13(4-5), 2000, pp. 411-430

[2] Van DerMaaten and Hinton,Journal of Machine Learning Research 9 (2008) 2579-2605

[3]McInnes, L, Healy, J, UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, ArXiv e-prints 1802.03426, 2018

[4] https://umap-learn.readthedocs.io/en/latest/basic_usage.html