# Machine Learning aplicado a las Ciencias Sociales

## Clase 1. Introducción y análisis no supervisado 1 (PCA)

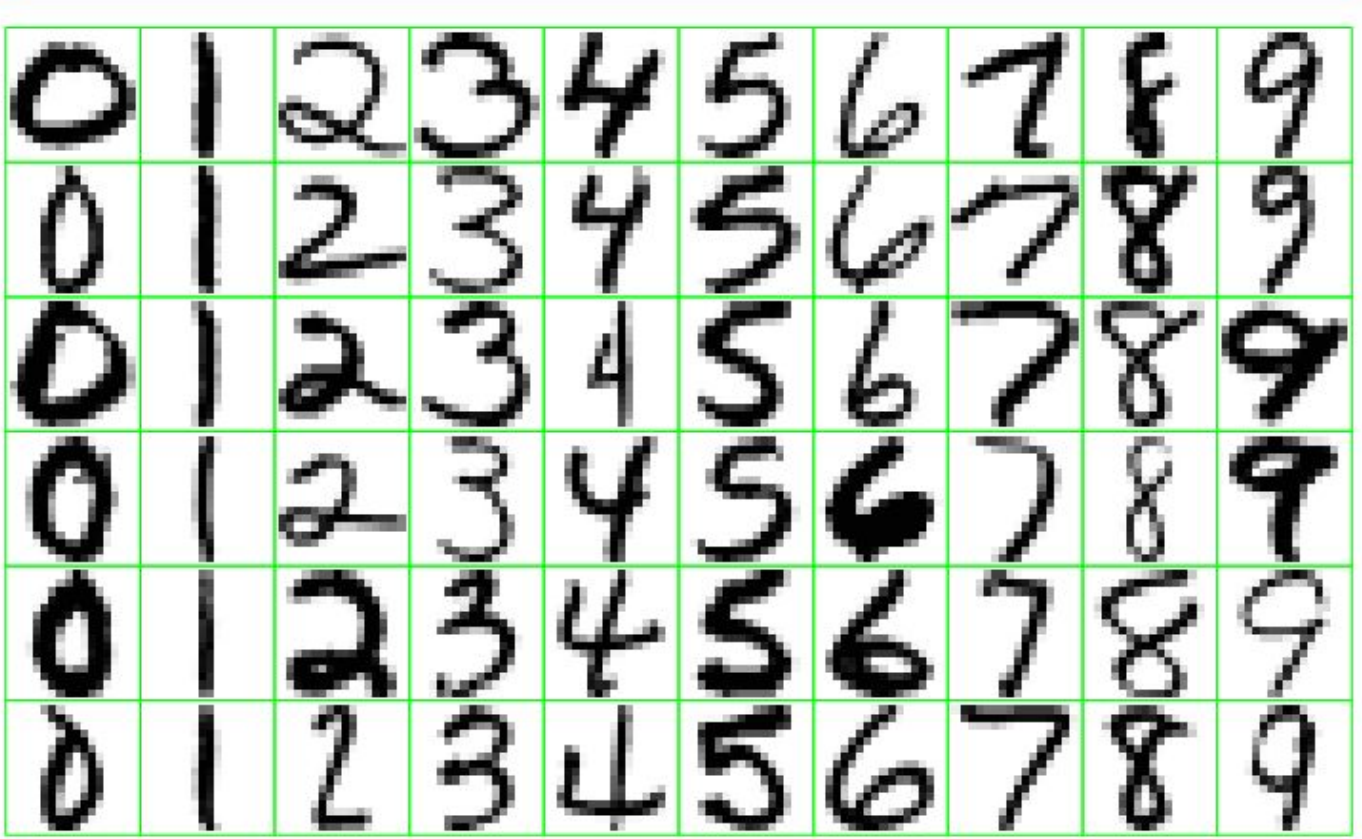
factor~data
EIDAES_UNSAM

# Machine Learning

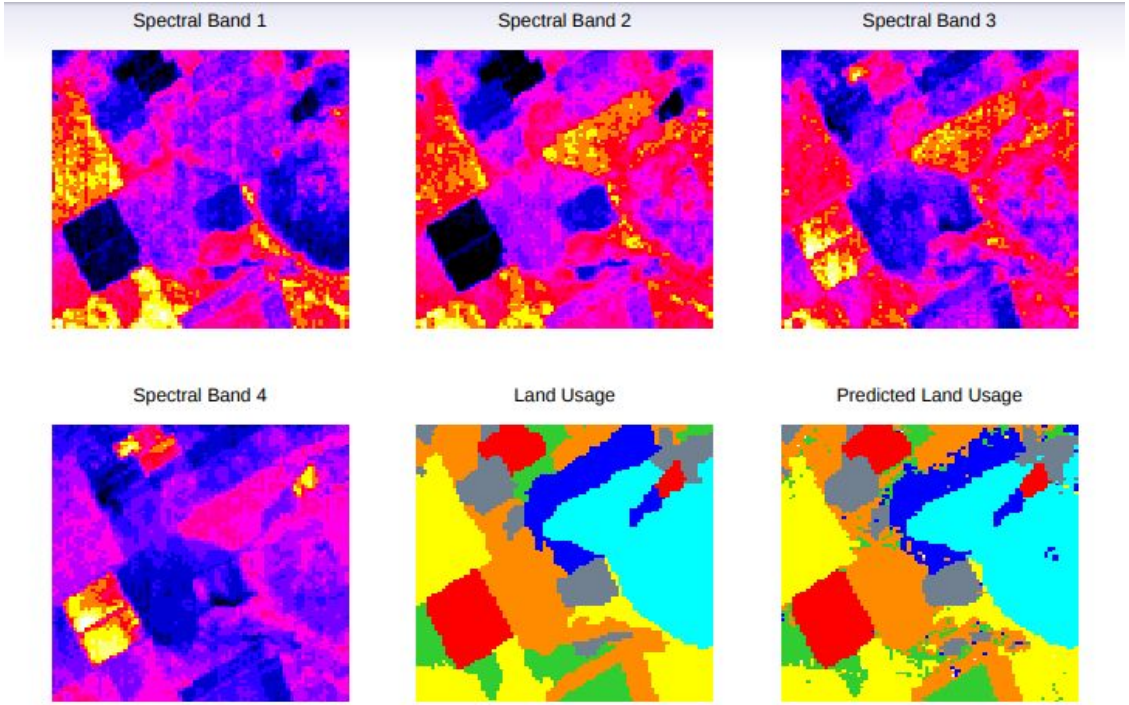
- ¿Podría una computadora ir más allá de “lo que sea que sepamos decirle que haga” y realmente “aprender” por su cuenta como realizar una determinada tarea?

- ¿Podría ser posible el aprendizaje automático de estas reglas a partir de los datos?

# Reconocimiento de dígitos



# Clasificación de imágenes satelitales




$Usage \in \{red\ soil,\ cotton,\ vegetation\ stubble,\ mixture,\ gray\ soil,\ damp\ gray\ soil\}$

# Machine Learning

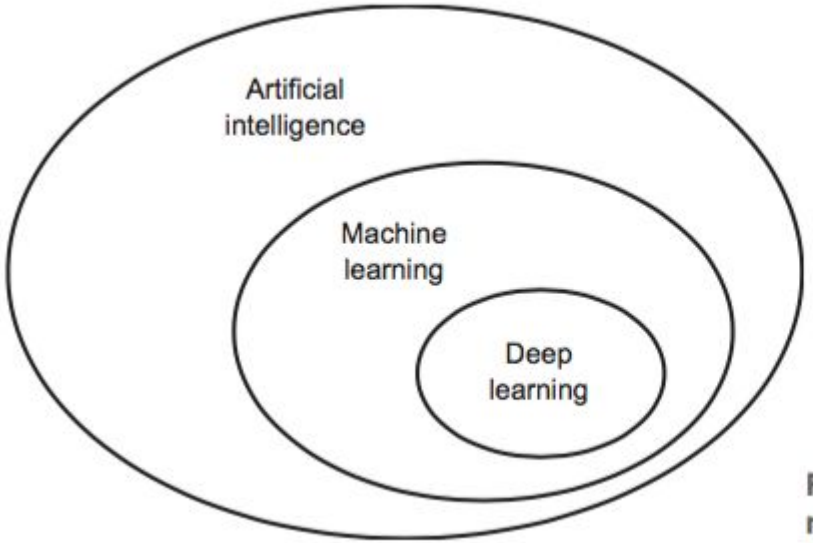



**Figure 1.1 Artificial intelligence, machine learning, and deep learning**

[Chollet, 2017]

# Un poco de epistemología… ¿qué es un modelo?

- Básicamente: una manera de proponer hipótesis sobre la forma en que se combinan variables
- En general vamos a estar tratando de general modelos de la forma Y = f(x) + e
- Todo el problema es estimar f (X), es decir, de qué forma(s) se combinan las X para generar un output
- Una posibilidad es suponer que Y es una combinación lineal de las X

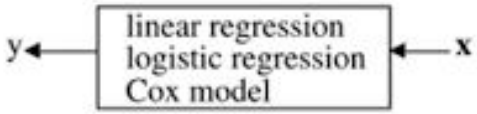
# Las dos culturas… (Breiman, 2001)

“Todos los modelos son equivocados, algunos son útiles” George Box


- El mundo como productor de outputs -y - en base a features -X -
- Problemas: ¿cuál es la manera en que el mundo produce resultados?
- Una forma común es asumir que los datos son generados por
- extracciones independientes de output = f (predictores, ruido, parámetros)

# Las dos culturas… (Breiman, 2001)

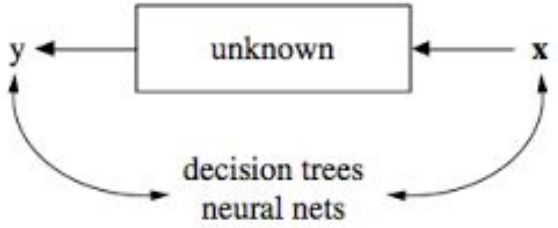
## Modelado estadístico




- Énfasis en f (x). El modelo se postula en base a supuestos sobre f (x)
- Conocimiento acumulado, teoría, diseño de experimentos
- Los parámetros son estimados con los datos y luego se realizan las
- predicciones.
- Evaluación del modelo: estimadores insesgados, robustos, mínima varianza

# Las dos culturas… (Breiman, 2001)

## Modelado algorítmico




- Énfasis en ŷ
- El enfoque es encontrar una función f (x) -un algoritmo- que opera sobre las x para predecir las y .
- El modelo se “aprende”de los datos
- Evaluación del modelo: performance predictiva

# Los datos, los algoritmos y la ciencia social

CHRIS ANDERSON SCIENCE 06.23.08 12:00 PM

## THE END OF THEORY: THE DATA DELUGE MAKES THE SCIENTIFIC METHOD OBSOLETE

*"Olvídense de la teoría del comportamiento humano, desde la lingüística hasta la sociología. Olvídense de las taxonomías, de la ontología y de la psicología. ¿Quién sabe por qué la gente hace lo que hace?. El punto es que lo hacen y que podemos trackear y medir eso que hacen con una precisión sin precedentes. Con suficientes datos, los números hablan por sí mismos."*

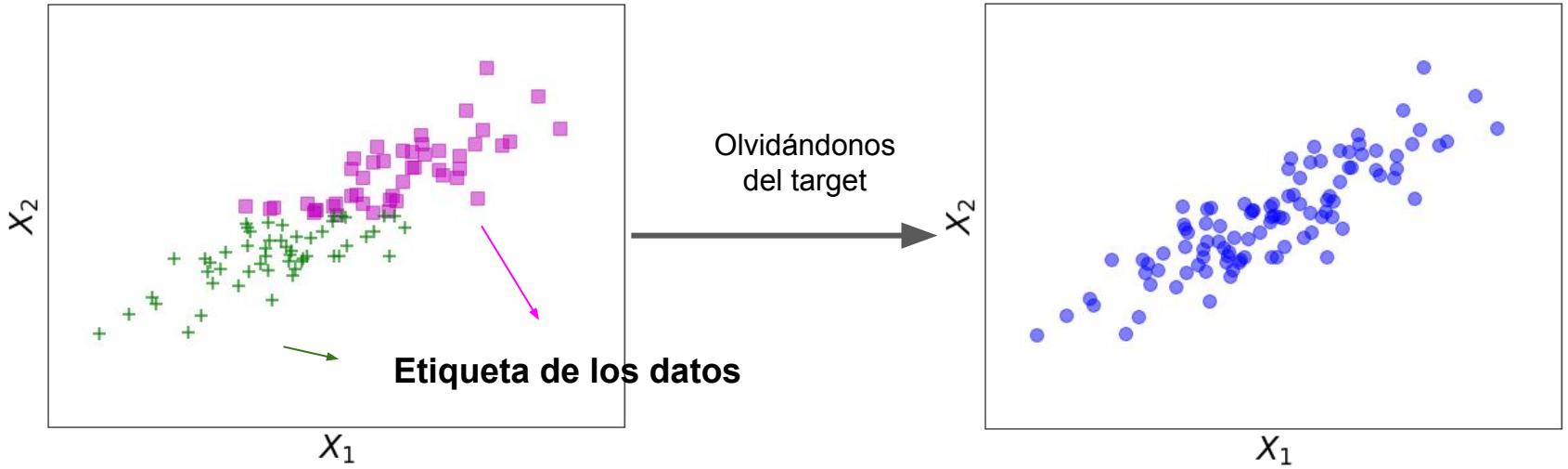
# Tipos de problemas

- Aprendizaje Supervisado
  - Variable dependiente => Y, resultado, target
  - Matriz de predictores (p), X, features, variables independientes, etc.
  - Problemas de regresión: Y es cuantitativa
  - Problemas de clasificación: Y es cualitativa
  - Tenemos datos de entrenamiento (conjuntos de Xi,Yi), observaciones
  - Podemos definir una (o varias) métricas para evaluar los modelos

- **Aprendizaje no Supervisado**
  - **No hay variable target (Y)**
  - **Solo hay X**
  - **Es más difícil evaluar qué tan bien funciona el modelo**

# ¿Qué es el aprendizaje no supervisado?

- El estudio o la exploración de cómo están representados datos y qué conclusiones podemos sacar de dicha representación.




**Aprendizaje no supervisado** es todo lo que podemos hacer con solo la representación de los datos en el espacio de features.

# ¿Qué podemos hacer solo con las features?

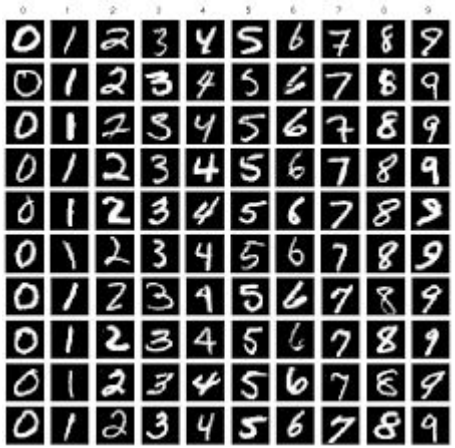
- Clustering (todavía no…): agrupar casos “parecidos”, que tengan una combinación de valores **similar** en las variables independientes (espacio de features) (próxima clase). Buscamos subconjuntos de datos muy parecidos entre sí.

- Reducción de la dimensionalidad (clase de hoy): encontrar combinaciones de features que reemplacen a los originales para reducir la dimensión del problema. “Reescribir” los datos en una menor cantidad de variables (o dimensiones)

# ¿Por qué reducir la dimensionalidad del problema?

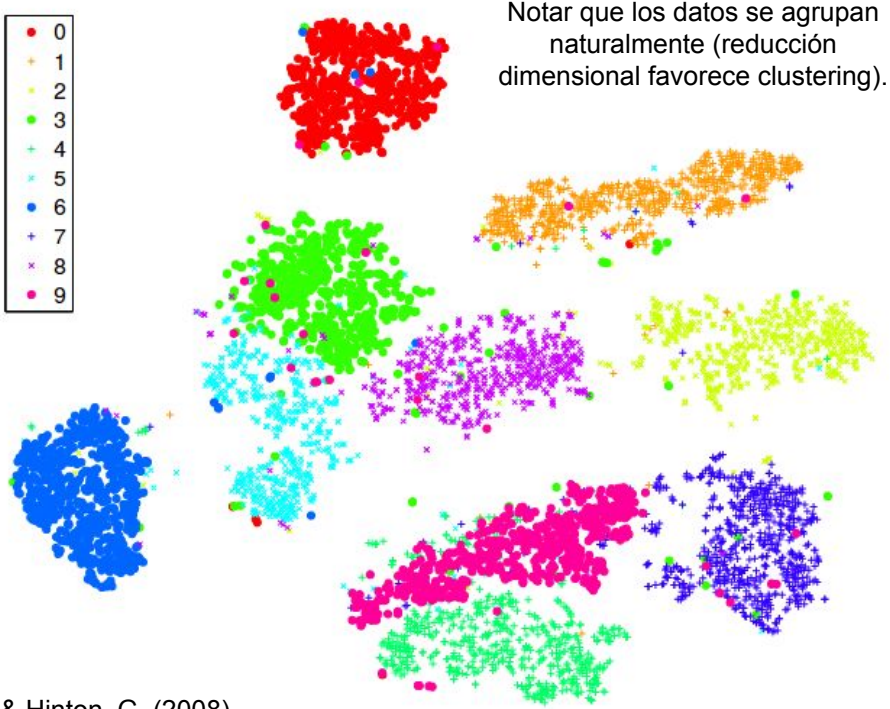
- ¿Todos los features aportan información relevante? ¿Es necesario trabajar con todos?
- Reduciendo la dimensión podemos:
  - Visualizar los datos en el espacio de dimensión reducida, más fácil de interpretar
  - Comprimir la información: nos permite separar la señal del ruido (abstracción)
  - Insumo para clustering: instancias parecidas en un espacio multidimensional son más parecidas en un espacio reducido (emergencia de estructuras).

# ¿Por qué reducir la dimensionalidad del problema?



**MNIST dataset**
Datos representados en un espacio de **748 píxeles**.

Visualización de datos multidimensionales en 2D




Notar que los datos se agrupan naturalmente (reducción dimensional favorece clustering).

(a) Visualization by t-SNE.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, *9*(11).

# ¿Por qué reducir la dimensionalidad del problema?

Compresión (con pérdida) de la información: separación de la señal del ruido.

Con 100 dimensiones (<< 4096) ya logramos una reproducción fiel de la imagen original.

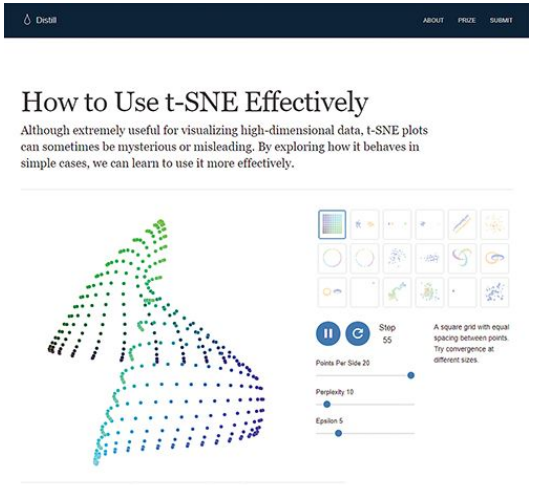

Datos representados en un espacio de **4096 píxeles**.

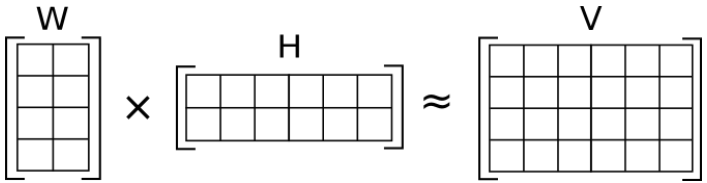
\+ dimensiones del espacio reducido


Imagen reconstruida de un espacio de dimensión reducida

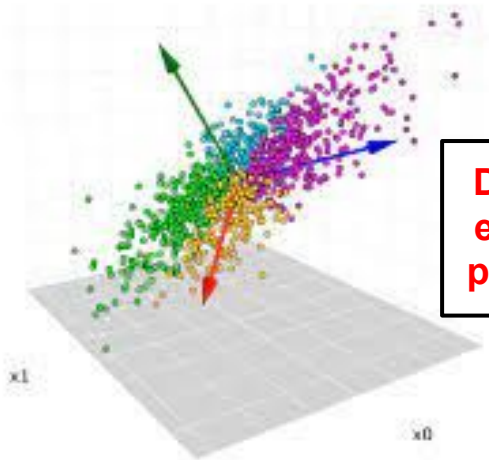
factor~data
EIDAES_UNSAM

# Algunos métodos de reducción dimensional

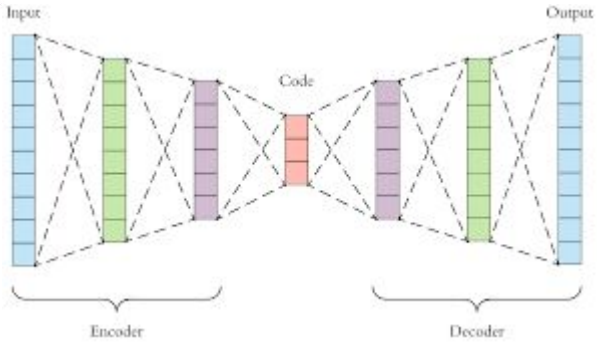



**Non-negative matrix factorization (NMF)**




**Descomposición en componentes principales (PCA)**

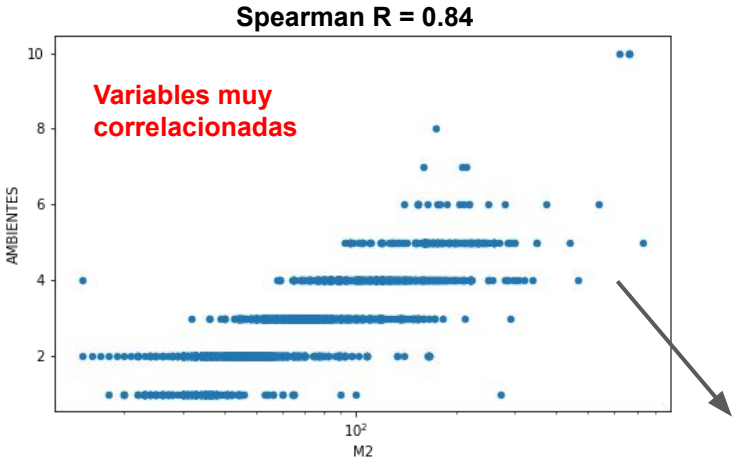


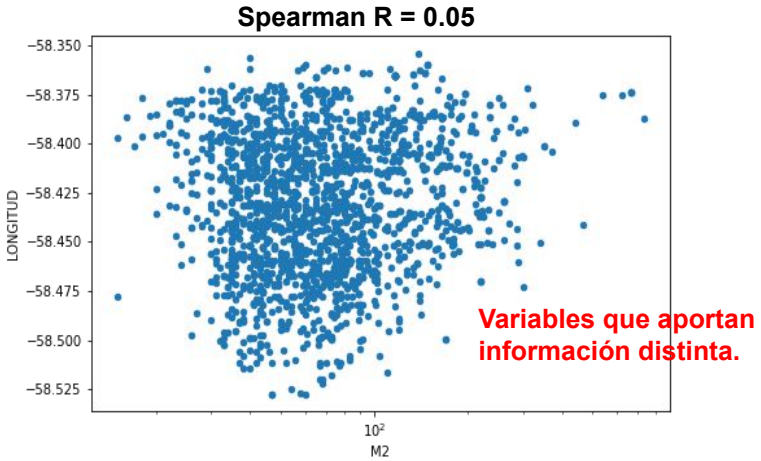

**Autoencoders**

factor~data
EIDAES_UNSAM

# Análisis de componentes principales (PCA)

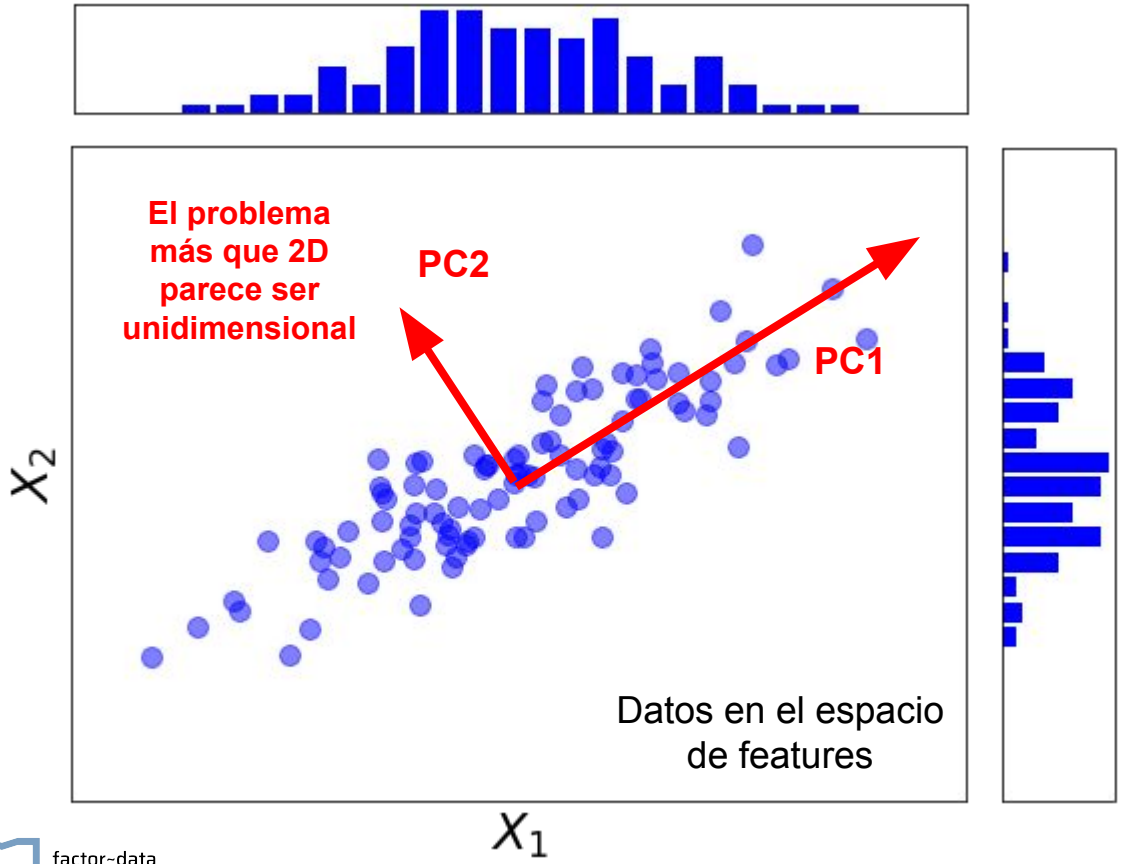
Problema: muchas veces hay variables que contienen prácticamente la misma información que otras (están muy correlacionadas entre sí), por lo que agregan una dimensión más al problema sin aportar muchas más información.




Una opción para solucionar el problema sería tirar una de las dimensiones por redundante, sin embargo ¿estamos seguros que toda la información que tiramos no es necesaria? ¿Hay alguna otra forma más inteligente de tirar dimensiones?

## Esquema general

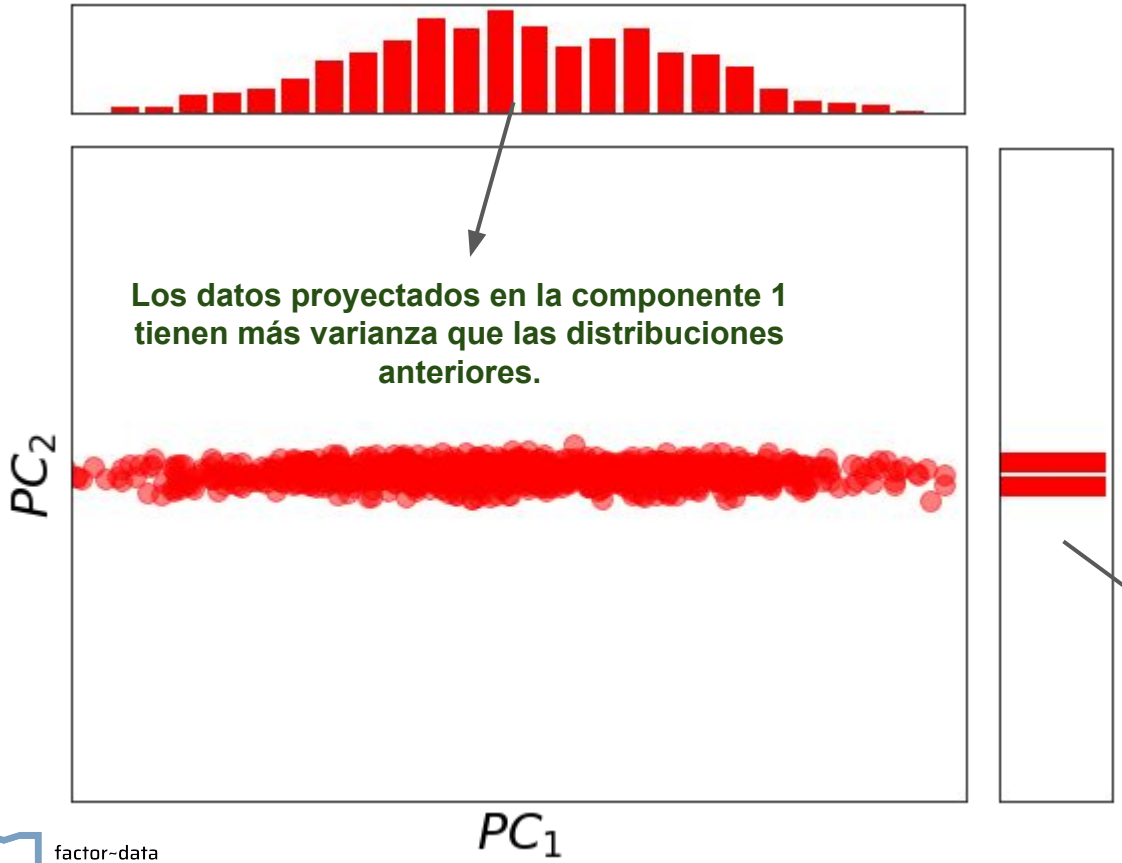



Podemos ver cómo se distribuyen los datos proyectando sobre cada uno de los features.

De las distribuciones podemos calcular su varianza, como una medida de qué tan dispersos están los datos en esa dirección.

Sin embargo, notamos direcciones (combinación lineal de features) donde los datos parecen variar más.

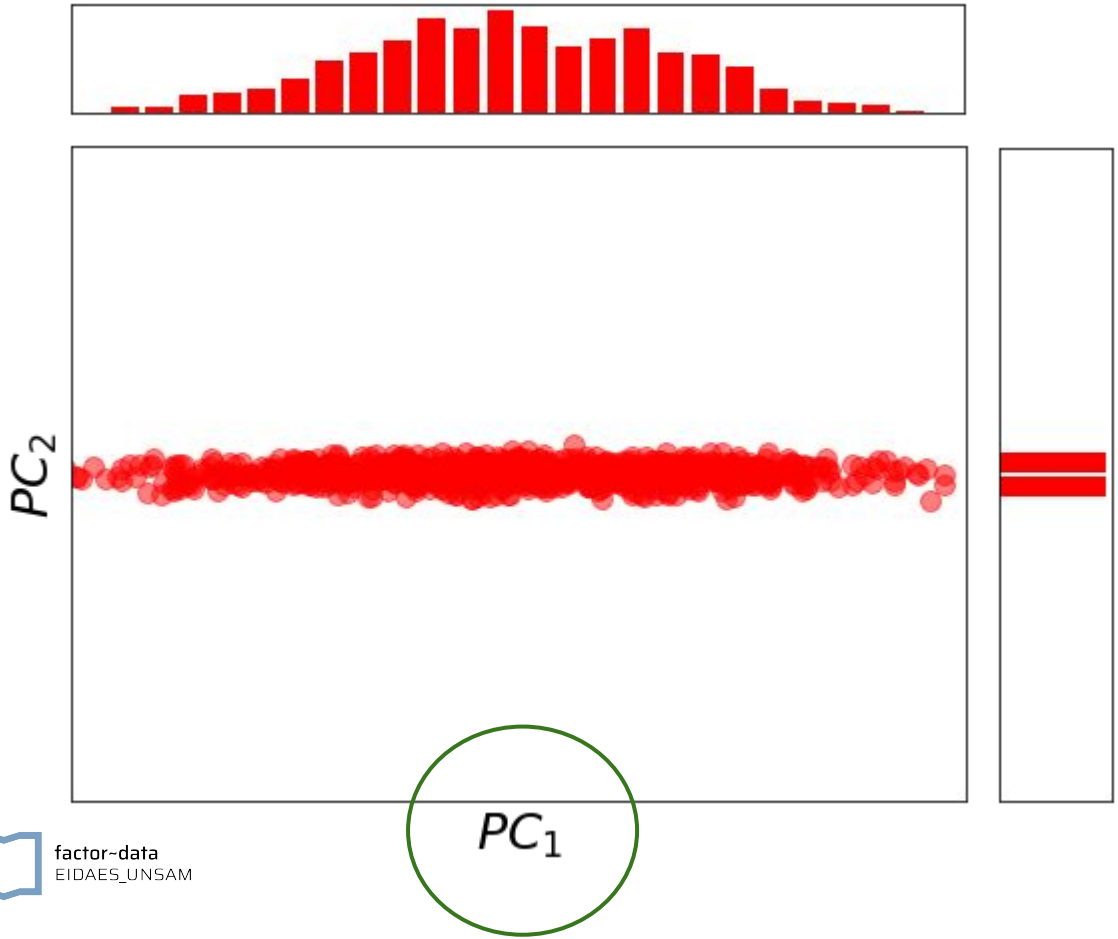
## Esquema general




Proyectando en esas nuevas direcciones (ortogonales) volvemos a ver la distribución de los datos proyectadas en las mismas.

**La componente 2 tiene mucha menos variación (parece ser más ruido que información importante)**

## Esquema general




Notar que si nos olvidamos de la componente 2, no perdemos tanta información como si hubiésemos tirado alguno de los features originales.

**Podemos reducir nuestro problema de 2 dimensiones a una sola.**

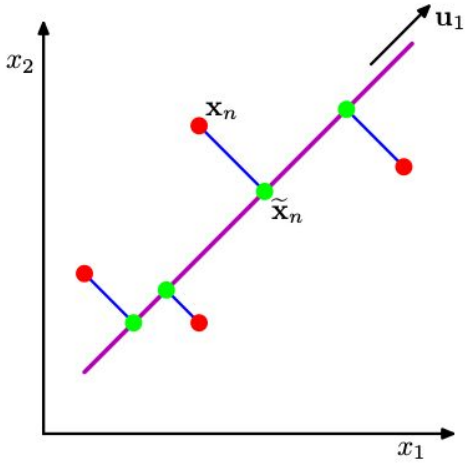
Las direcciones que se llevan las mayor cantidad de varianza de los datos se llaman **componentes principales**

# PCA: descripción matemática

Dado un conjunto {xn} de N datos en un espacio de dimensión D (cada xn es un vector en el espacio de D dimensiones).

Las M < D componentes principales son:

- las M direcciones que maximizan la varianza de las proyecciones en ese subespacio,
- o, equivalentemente, las M direcciones que minimizan el error en la proyección (figura).



# PCA: descripción matemática

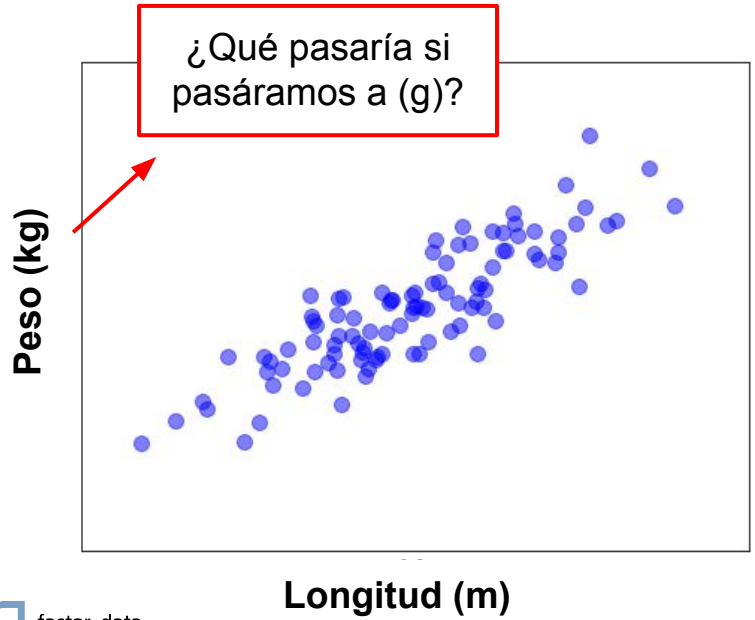
No nos vamos a detener en detalle en los diferentes algoritmos que se utilizan para estimar los componentes. Solo diremos que uno de ellos lleva a que podemos obtener los componentes principales a partir de la descomposición de la matriz de covarizanza **S**:

$$\bar{\bar{S}}\bar{u} = \lambda\bar{u}$$

- Las componentes principales son ortogonales.
- Ordenando las componentes desde el autovalor más grande al más chico, las primeras componentes llevan la mayor varianza de los datos (es lo que efectivamente hace PCA).

# Dependencia de las unidades y escaleo

¿Qué pasa con la variabilidad si cambiamos las unidades de alguna de las variables?

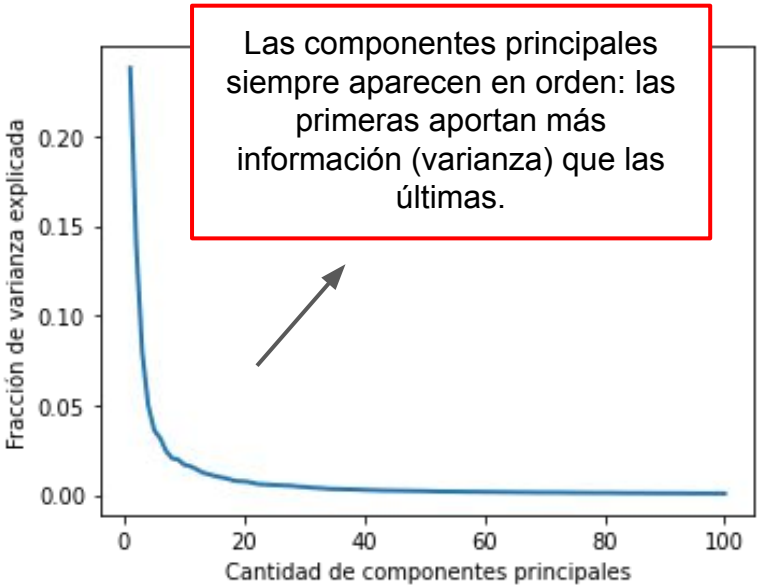



Para evitar introducir o sacar variabilidad por cambiar las unidades de alguna de las variables, una práctica habitual antes de hacer PCA es estandarizar las variables:

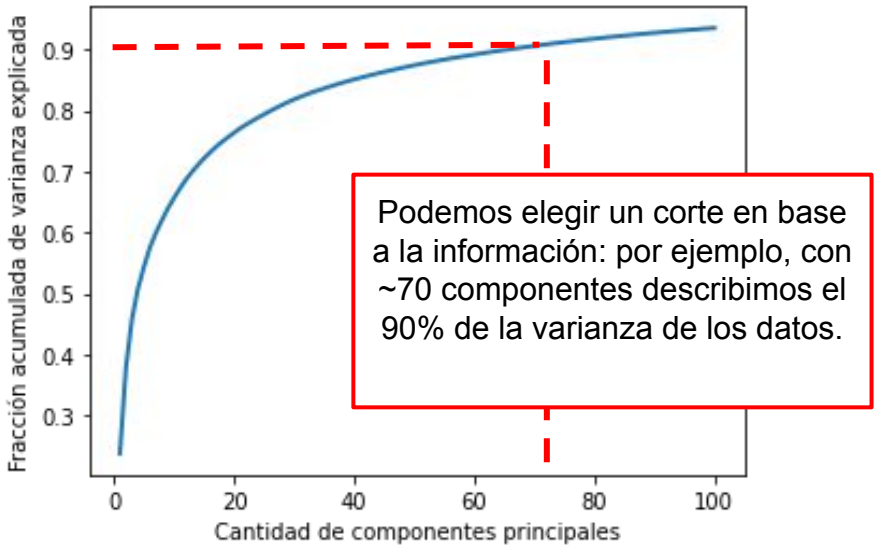
$$Z = \frac{X - \mu}{\sigma}$$

# ¿Cuántas componentes?

Al ser un problema de aprendizaje no-supervisado, no tenemos un conjunto de test para validar el número de componentes que elijamos. ¿Con cuántas nos quedamos?




**Fracción de varianza que aporta cada componente**




**Fracción de varianza acumulada**

# Reducción dimensional como input de otros modelos

Podemos usar las primeras componentes principales (Zm) como variables independientes en modelos de regresión o clasificación.

$$y \sim \beta_0 + \beta_1 Z_1 + \ldots + \beta_M Z_M$$

Ventajas:

- Resolvemos el problema de colinealidad: En PCA las variables son por defecto ortogonales y por lo tanto buenas candidatas.
- Podemos validar nuestro modelo y por lo tanto tomar la cantidad de componentes principales como un hiperparámetro.

# Resumen de PCA

- Las componentes principales son una combinación lineal de los features originales y se corresponden con los autovectores de la matriz de covarianza de los datos.
- Las componentes están ordenadas de mayor a menor, en el sentido de la información (varianza) que se llevan. Si tiramos las últimas componentes, estamos reduciendo la dimensión de nuestro problema.
- Una práctica usual es estandarizar las variables antes de aplicar PCA (fundamental si los features corresponden a distintas magnitudes).
- Podemos usar las componentes principales para: comprimir la información, visualizar datos multidimensionales en bajas dimensiones, y como input para modelos de aprendizaje supervisado.

# Vamos al Notebook