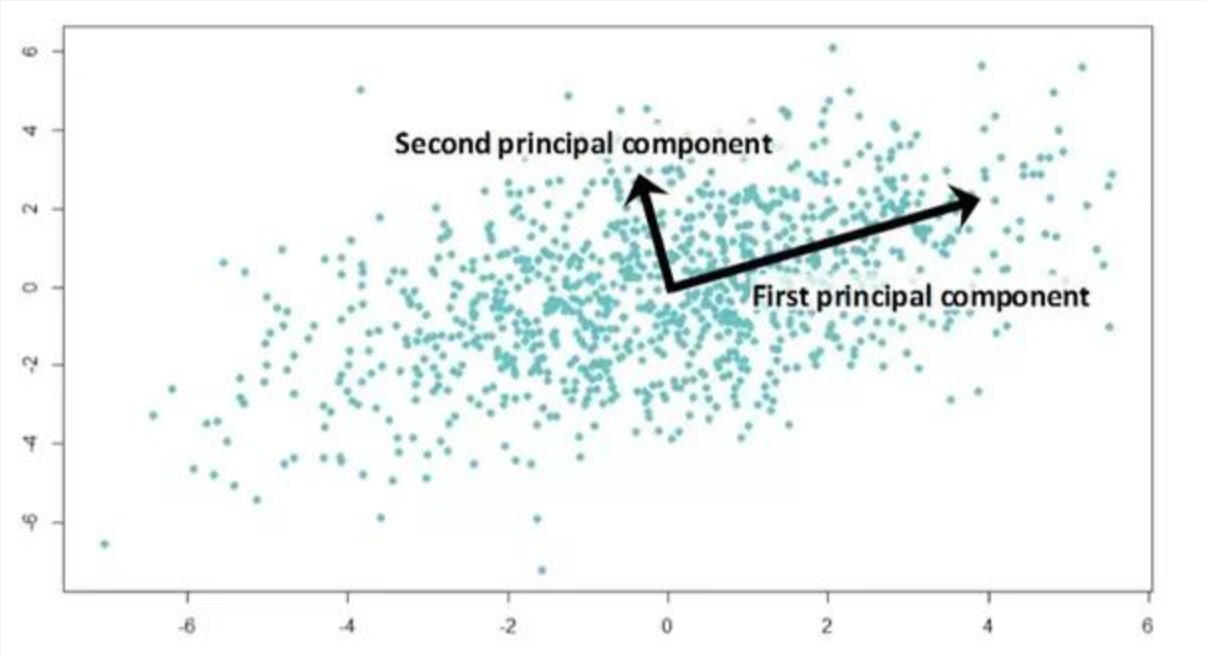


DigitalHouse > Coding School

DATA SCIENCE

Módulo 2

Técnicas de Reducción de la Dimensionalidad con feature extraction: PCA y Manifold Learning

1. **MOTIVACIÓN PARA LA REDUCCIÓN DE DIMENSIONALIDAD**
   Analizar las causas que justifican la aplicación de estas técnicas

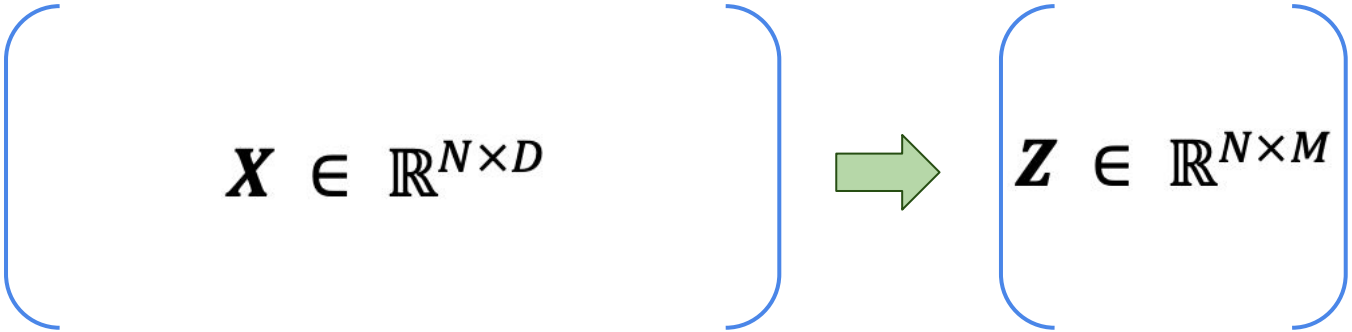
2. **ANÁLISIS DE LAS TÉCNICAS**
   En particular, entender cómo se calculan y qué problema resuelven los componentes principales, interpretación geométrica y algebraica.

3. **SOBRE LOS RESULTADOS**
   Interpretar los resultados de los algoritmos y determinar la cantidad óptima de componentes.

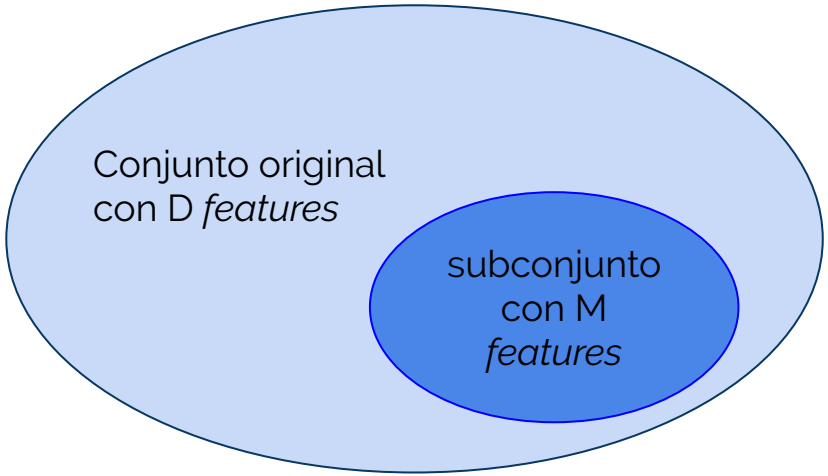
4. **APLICACIONES**
   Conocer y experimentar con las distintas aplicaciones prácticas de estas técnicas

¿Qué significa y por qué puede ser útil reducir la dimensionalidad de un problema?

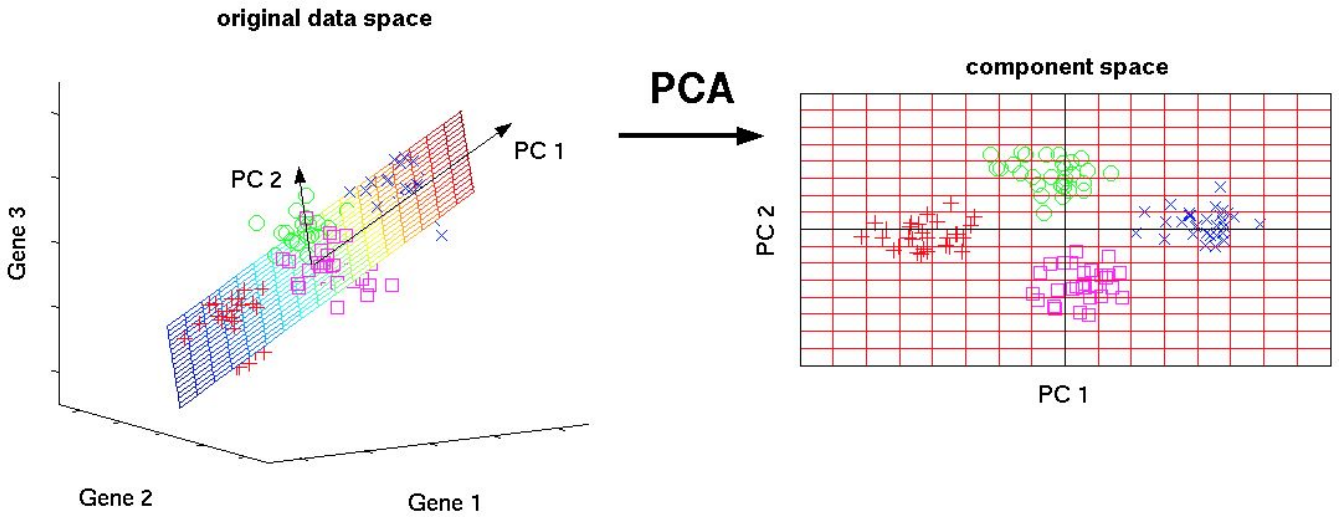
Por **reducción de la dimensionalidad** de un problema entendemos reducir la cantidad de variables explicativas o *features* del mismo. Es decir pasar de un problema con D *features*, donde D puede ser muy grande, a otro con M, donde M << D.



Podemos **reducir la dimensionalidad** del problema simplemente **seleccionando** el subconjunto de variables que más contribuyen a nuestro problema. A este proceso se lo llama selección de atributos o ***feature selection***.



Por otro lado, podemos intentar **transformar los datos** que tenemos para representar la misma información con menos dimensiones, buscando **minimizar la pérdida de información** en el proceso. Este tipo de técnicas son denominadas ***feature extraction*** y son las que vamos a ver hoy.



Pero antes, hagámonos una pregunta.

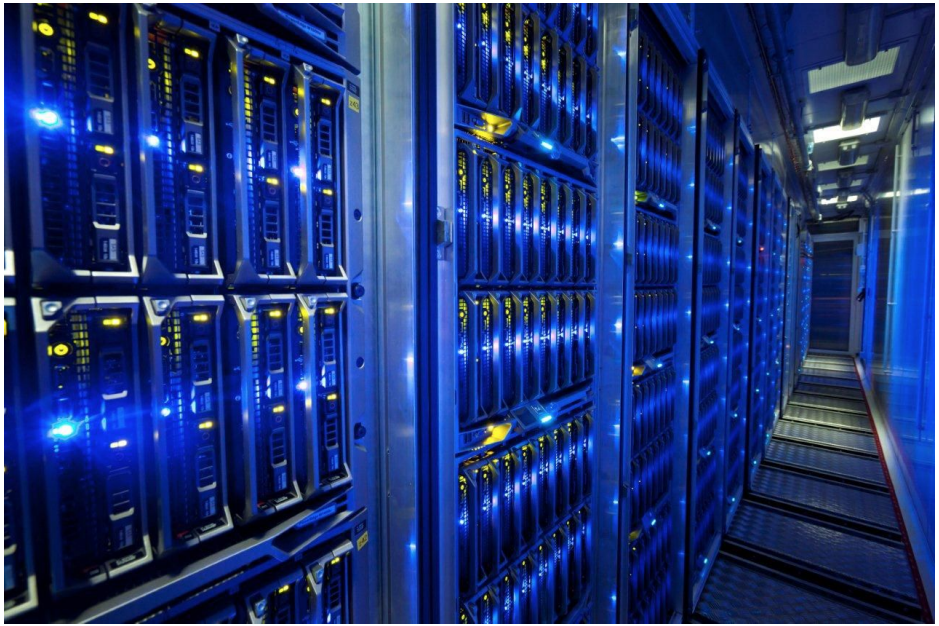
Si supuestamente, cuantos más datos tengamos mejor, **¿por qué podríamos querer reducir el número de nuestras *features?***

Pero antes, hagámonos una pregunta.

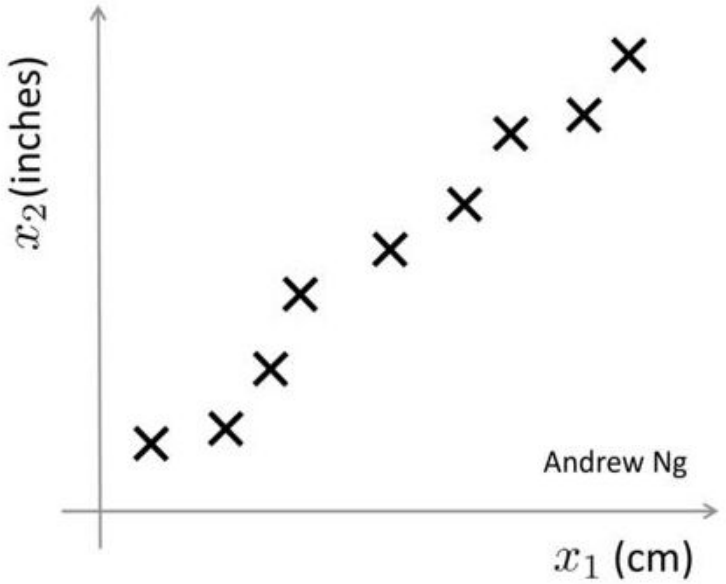
Si supuestamente, cuantos más datos tengamos mejor, **¿por qué podríamos querer reducir el número de nuestras *features?***

Básicamente por 2 grandes motivos: ***comprimir los datos y poder visualizarlos.***

Compresión de los datos: reducir los **requerimientos de memoria** y de **poder de cómputo** para entrenar el modelo.



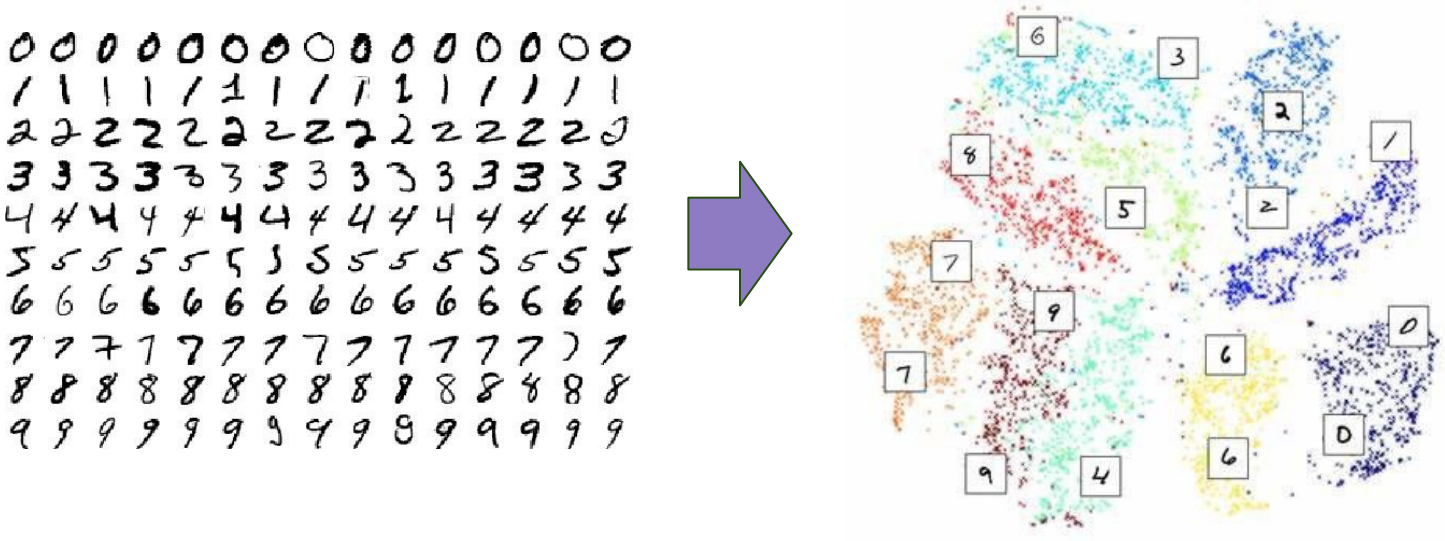
Compresión de los datos: eliminar la presencia de **variables redundantes.**



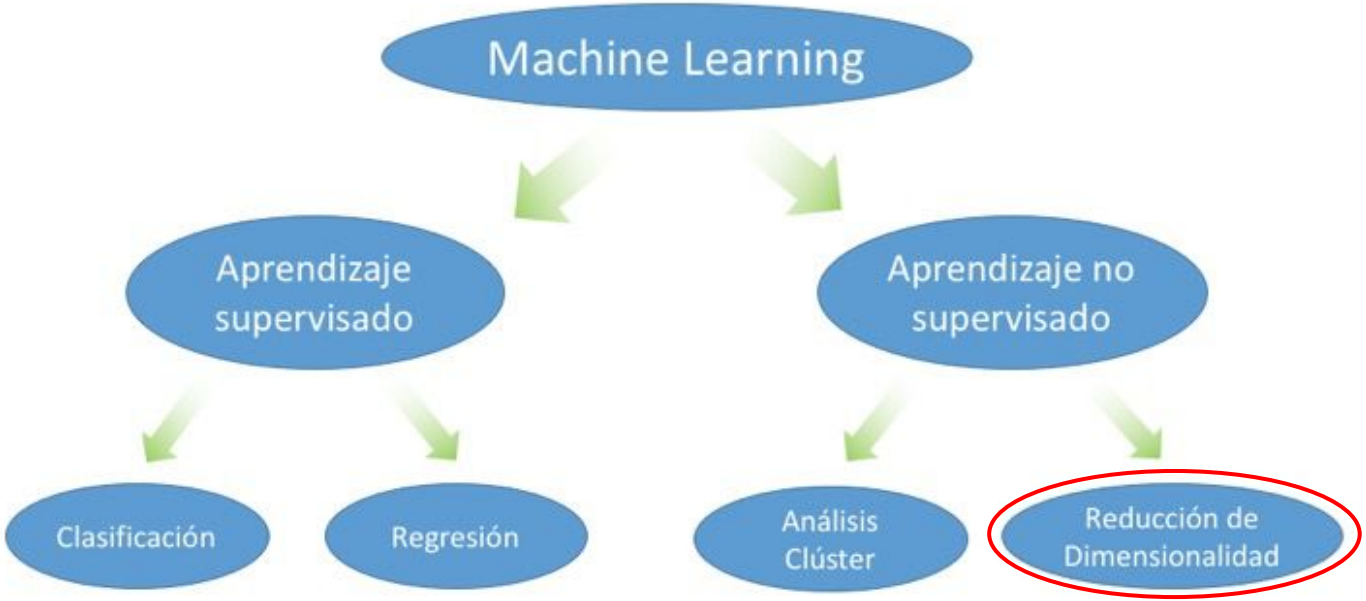
Compresión de los datos: reducir el **ruido** del dataset y mejorar la **capacidad de generalización** del modelo.

Modelos muy complejos (es decir con muchas *features*) tienden a tener **problemas para generalizar** ya que suelen ajustar en exceso al set de entrenamiento (es decir que aprenden el ruido presente en los datos) y por lo tanto arrojan resultados pobres al predecir observaciones nuevas. Este problema se llama **overfitting** y es uno de los problemas más importantes principales en machine learning.

Visualización de los datos: representar nuestro problema en 2 o 3 dimensiones nos permite **realizar visualizaciones** útiles para **explorar** nuestro dataset o para **comunicar** sus características de forma efectiva e impactante.



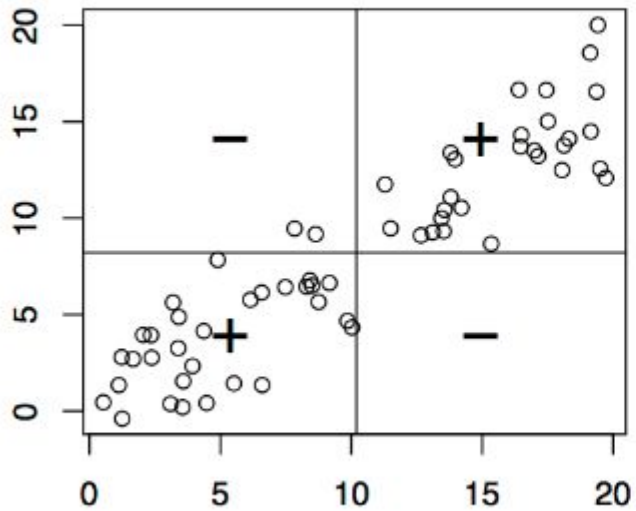
La reducción de la dimensionalidad por *feature extraction* es un problema de **aprendizaje no supervisado**.



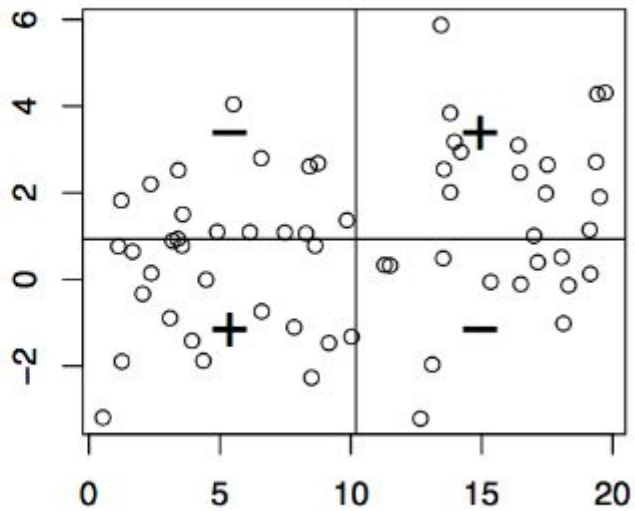
Repaso: covarianza y correlación

Decimos que dos variables X e Y, tienen **covarianza** positiva cuando se encuentran por encima de su media al mismo tiempo y tienen covarianza negativa cuando al mismo tiempo, una está por debajo y otra por encima.

Covarianza positiva



Covarianza cercana a cero.



La covarianza se mide como:

$$Cov_{xy} = \frac{\Sigma(x-\bar{x})(y-\bar{y})}{(n-1)}$$

La covarianza de un conjunto de datos con D variables se puede representar con una matriz de D x D llamada **matriz de varianzas y covarianzas**:

	^GSPC	^IXIC	XOM	C	GE	MSFT	K	GM
^GSPC	0.633	0.929	0.505	0.495	0.448	0.258	0.261	1.226
^IXIC	0.929	1.737	0.340	0.584	0.507	0.482	0.211	1.842
XOM	0.505	0.340	3.253	-0.421	-0.017	0.268	0.318	2.197
C	0.495	0.584	-0.421	1.923	0.688	0.176	0.277	-0.242
GE	0.448	0.507	-0.017	0.688	1.834	0.761	0.232	0.049
MSFT	0.258	0.482	0.268	0.176	0.761	1.945	0.181	1.315
K	0.261	0.211	0.318	0.277	0.232	0.181	1.045	0.688
GM	1.226	1.842	2.197	-0.242	0.049	1.315	0.688	9.429

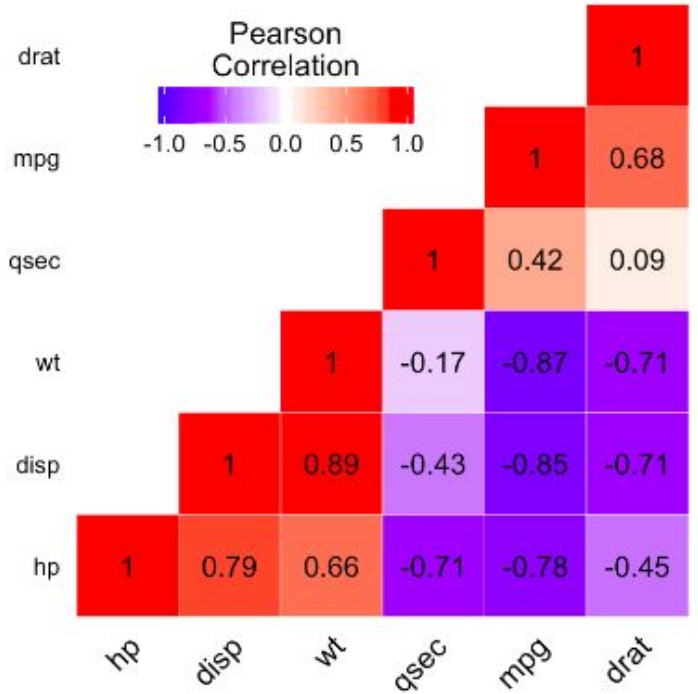
* En la diagonal se encuentra la varianza de cada feature

* En el resto de la matriz se encuentran las covarianzas

La **correlación es una versión estandarizada** (dividida por los desvíos estándar) de la covarianza:

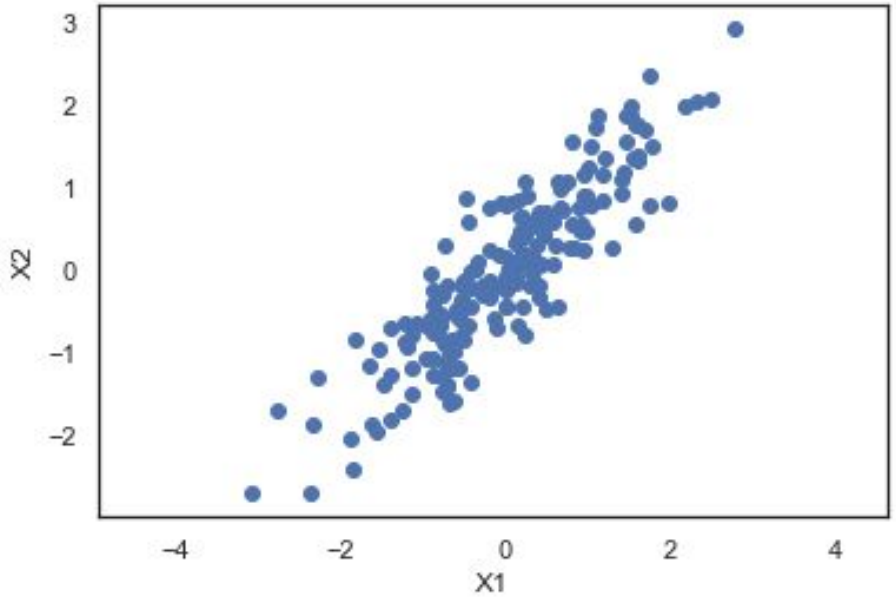
$$r_{xy} = \frac{1}{n-1} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right)$$

* La correlación está acotada entre 1 y -1.
* Siempre que la covarianza es positiva, la correlación es positiva y viceversa.
* Mientras que la correlación no tiene unidades físicas, la covarianza sí.



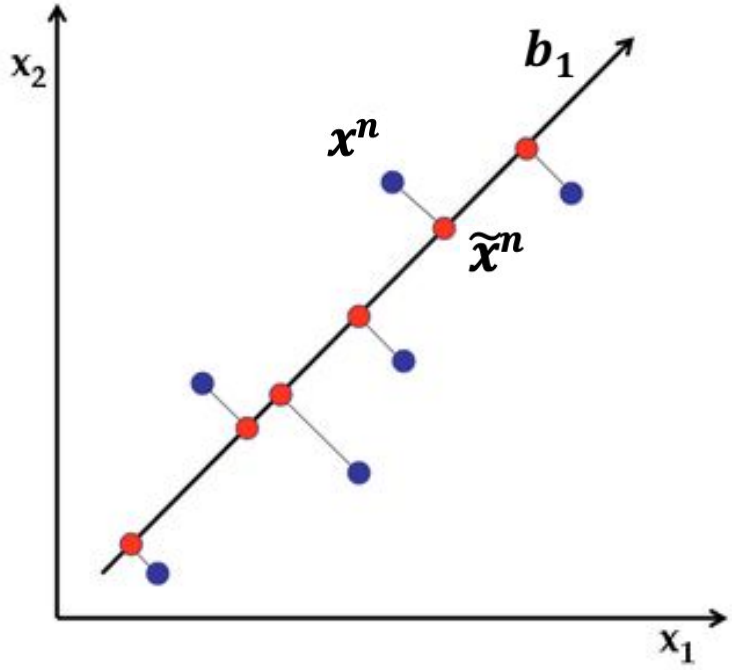
Análisis de Componentes Principales

Imaginemos que tenemos 2 variables, x1 y x2. Las dos están **fuertemente correlacionadas**. Esto quiere decir que hay información que ambas variables comparten.



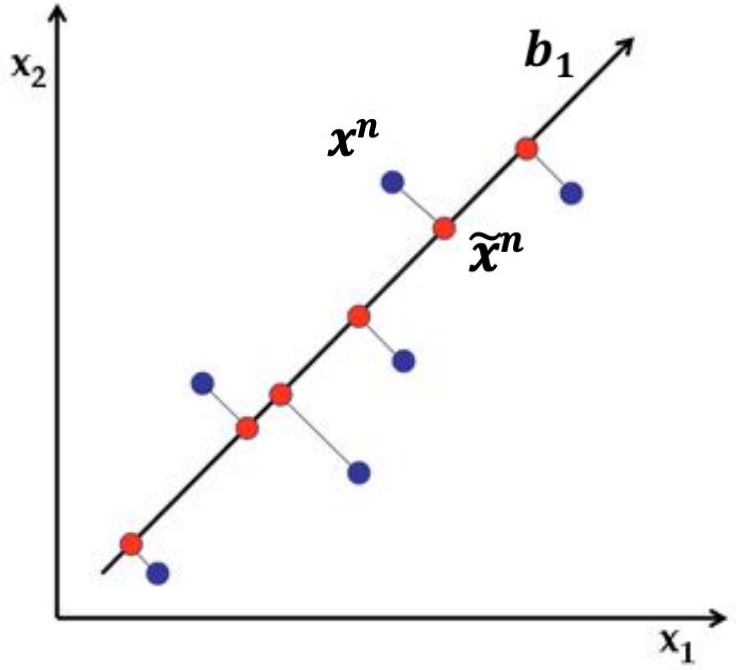
El problema que se plantea **PCA** es encontrar la forma de proyectar estos datos a un **subespacio de menor dimensionalidad**, preservando la mayor cantidad posible de información.

El objetivo de **PCA** es entonces encontrar la **dirección b1 que preserva la mayor cantidad posible de información**.



De esta manera, podríamos **proyectar** nuestros datos sobre el vector **b1** y representarlos **usando solamente 1 coordenada**, la ubicación en el subespacio de 1 dimensión generado por **b1**, en lugar de 2 coordenadas, el espacio de 2 dimensiones generado por **x1** y **x2**.

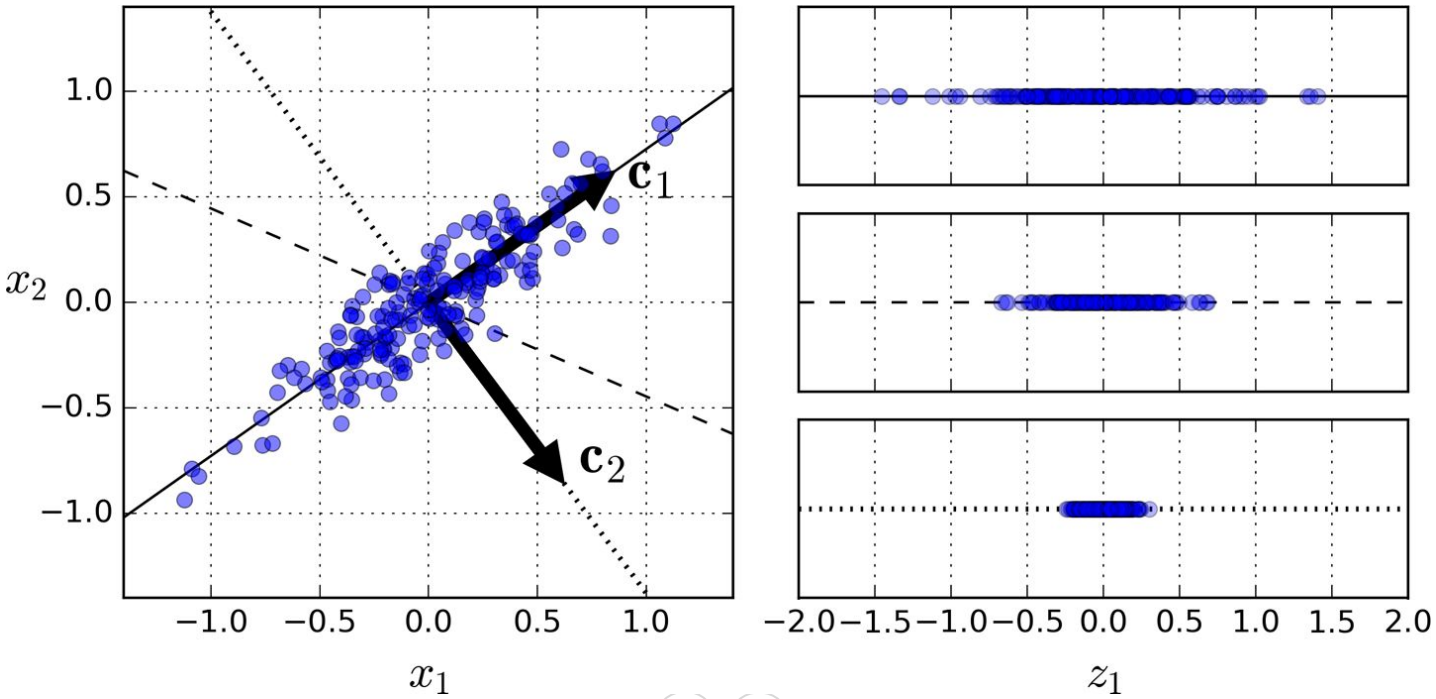
El objetivo de **PCA** es entonces encontrar la **dirección b1 que preserva la mayor cantidad posible de información**.



Es decir que queremos encontrar la proyección que minimice el ***error cuadrático medio de reconstrucción***:

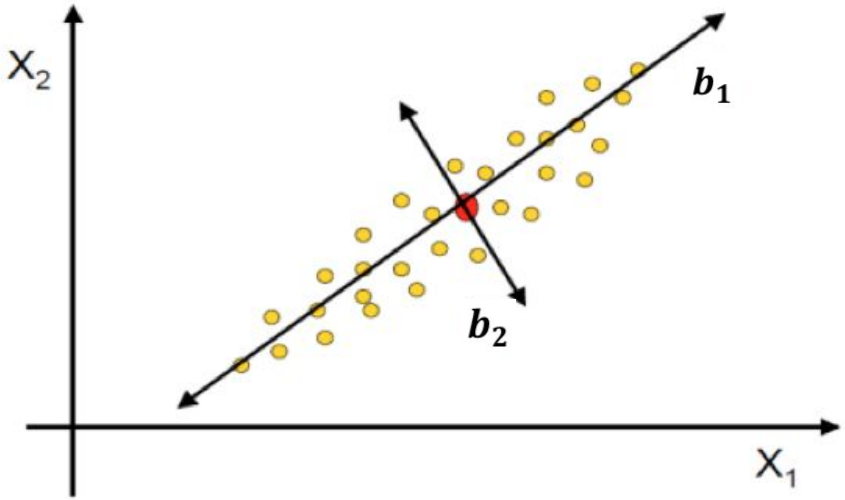
$$J = \frac{1}{N}\sum_{n=1}^{N}\|x^n - \tilde{x}^n\|^2$$

Más adelante, vamos a ver que buscar las direcciones que minimizan el *Error Cuadrático Medio de Reconstrucción* es equivalente a buscar las **direcciones que maximizan la varianza de nuestros datos**.

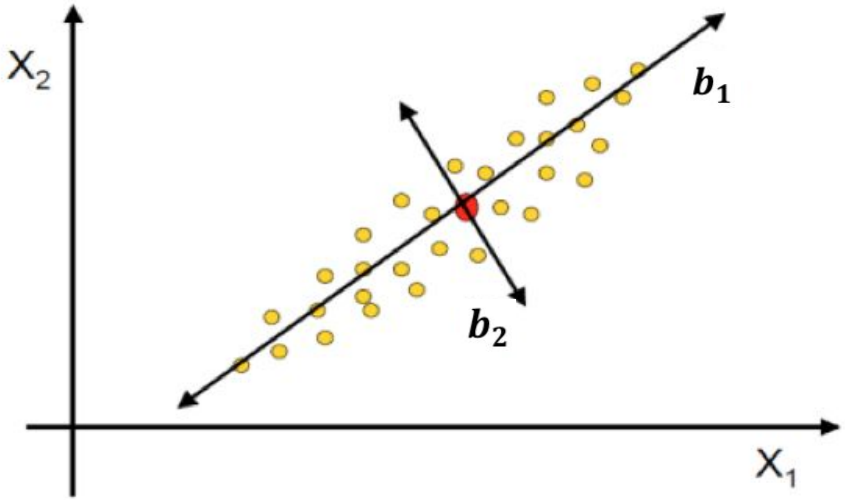


Una vez encontrado el vector **b1**, podemos definir un segundo vector **b2**, ortogonal a **b1**. Estos dos vectores forman una base de $\mathbb{R}^2$ al igual que **x1** y **x2.**

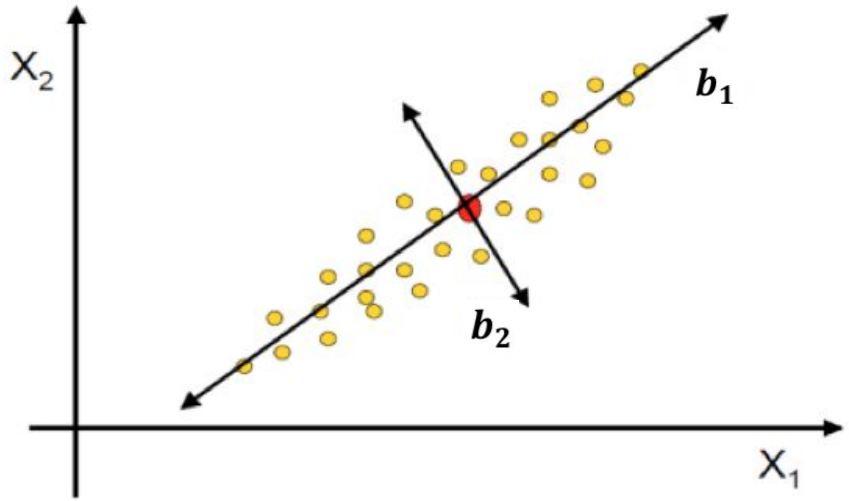
De esta manera, vamos a haber definido **un nuevo sistema de coordenadas** que reemplaza al sistema definido por el espacio de las *features* originales.



Este **nuevo sistema de coordenadas** es el de los **Componentes Principales**, donde **b1** y **b2** son, correspondientemente, el **primer** y **segundo** componentes principales.

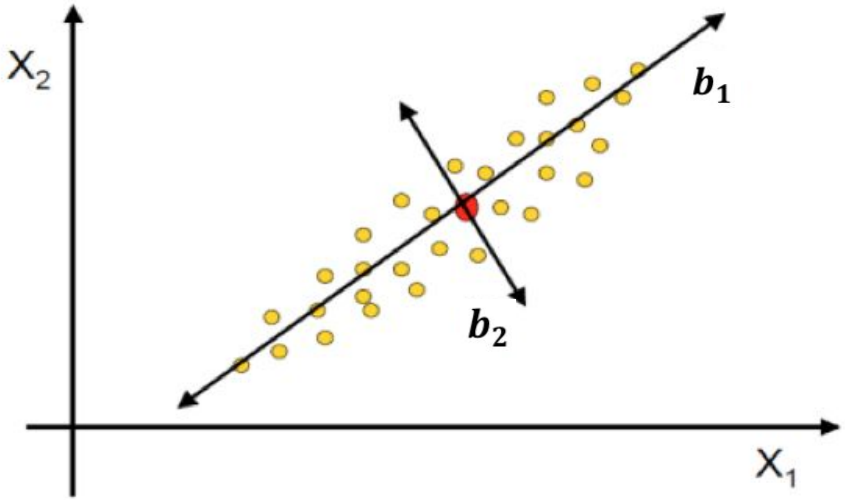


Por diseño de **PCA**, cada componente se define como **la dirección que más varianza de los datos (es decir información) captura** y sea ortogonal a todos los otros componentes principales. Es decir que los componentes van a estar ordenados de acuerdo a la cantidad de varianza que capturen. Cómo además vamos a exigir que sean de norma igual a 1, los CP van a formar una **base ortonormal** de igual dimensión del espacio de las *features* originales.

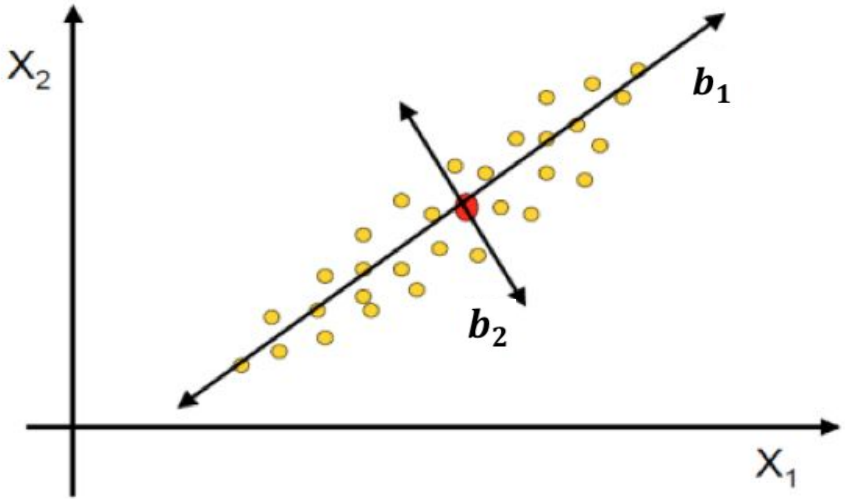


La **ortogonalidad** (que podría interpretarse como perpendicularidad) implica que las nuevas coordenadas van a tener **correlación cero** entre ellas.

Así, **descartando los Componentes Principales que menos varianza capturen**, podemos reducir la dimensionalidad del problema, conservando la mayor cantidad posible de información. Si no descartamos ningún Componente Principal, no vamos a reducir la dimensionalidad del problema.

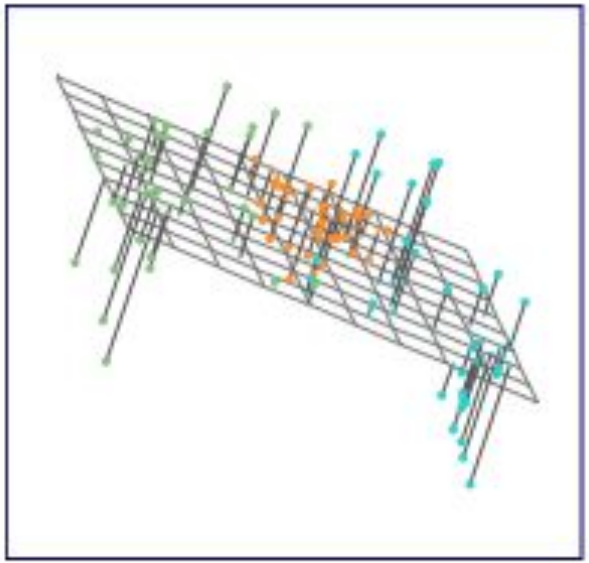


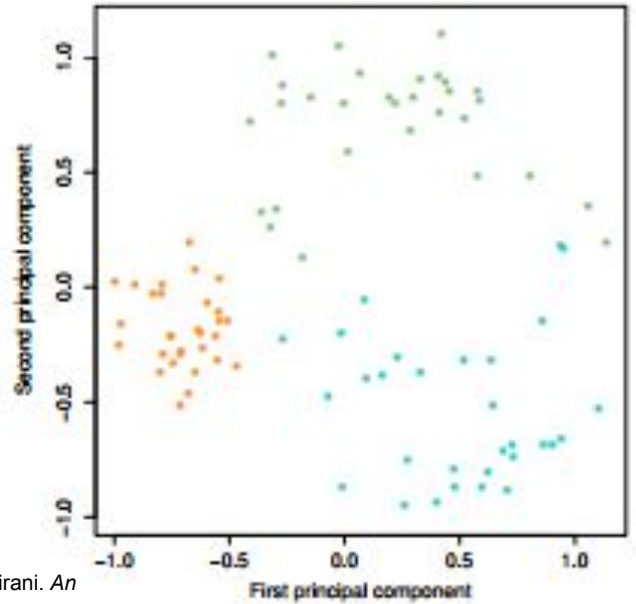
Así, **descartando los Componentes Principales que menos varianza capturen**, podemos reducir la dimensionalidad del problema, conservando la mayor cantidad posible de información. Si no descartamos ningún Componente Principal, no vamos a reducir la dimensionalidad del problema.



En este caso, descartaríamos a **b2**, proyectando los datos a **b1** y quedándonos con una sola dimensión.

En el caso de un problema con **3 dimensiones** (3 *features*), podríamos quedarnos con el primer y segundo componentes principales, que definen **el plano que más se acerca a los datos**.





Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. *An Introduction to Statistical Learning - with Applications in R*

Teniendo ya una intuición geométrica, profundicemos un poco en los aspectos algebraicos de **PCA** para tener una mejor comprensión del modelo.

N: cantidad de observaciones (registros)
D: cantidad de features

Tenemos el siguiente **dataset**:

$$\boldsymbol{X} = \{\boldsymbol{x^1}, \ldots, \boldsymbol{x^N}\}, \; \boldsymbol{x^n} \in \mathbb{R}^D$$

Tenemos el siguiente **dataset**: $\boldsymbol{X = \{x^1, \ldots, x^N\},\ x^n \in \mathbb{R}^D}$

Los datos están expresados en el espacio de las *features*.

Podemos plantear una base $\boldsymbol{B = (b_1, \ldots, b_D)}$, a la que le vamos a exigir que sea ortonormal, tal que:

$$\boldsymbol{x^n = \sum_{j=1}^{D} \beta_j^n \, b_j \in \mathbb{R}^D}$$

Suponiendo que queremos conservar M dimensiones, podemos expresar a $\boldsymbol{x^n}$ de la siguiente manera:

$$\boldsymbol{x^n} = \sum_{j=1}^{D} \beta_j^n \, \boldsymbol{b_j} = \sum_{j=1}^{M} \beta_j^n \, \boldsymbol{b_j} + \sum_{j=M+1}^{D} \beta_j^n \, \boldsymbol{b_j}$$

Si descartamos el subespacio generado por los vectores $\boldsymbol{b_{M+1}, \ldots, b_D}$ y nos quedamos con las primeras M dimensiones, obtenemos:

$$\tilde{\boldsymbol{x}}^{\boldsymbol{n}} = \sum_{j=1}^{M} \beta_j^n \, \boldsymbol{b_j} \in \mathbb{R}^D$$

El problema de optimización que plantea **PCA:**

Encontrar los vectores $\boldsymbol{b_1, \dots, b_M}$ y las coordenadas $\boldsymbol{\beta_1^n, \dots, \beta_M^n}$ que minimicen el **error cuadrático medio de reconstrucción**:

$$J = \frac{1}{N} \sum_{n=1}^{N} \|\boldsymbol{x^n} - \tilde{\boldsymbol{x}}^{\boldsymbol{n}}\|^2$$

Y volvemos al problema al que habíamos llegado con la intuición geométrica.

Analicemos la expresión para $\tilde{\boldsymbol{x}}^{\boldsymbol{n}}$ más en detalle:

$$\tilde{\boldsymbol{x}}^{\boldsymbol{n}} = \sum_{j=1}^{M} \beta_j^n \boldsymbol{b_j} \in \mathbb{R}^D$$

- $\tilde{\boldsymbol{x}}^{\boldsymbol{n}}$ vive en $\mathbb{R}^D$ pero en el subespacio de $\mathbb{R}^D$ generado por los vectores $\boldsymbol{b_1}, \ldots, \boldsymbol{b_M}$
- Este subespacio es el complemento ortogonal del subespacio generado por las D-M dimensiones que descartamos y se denomina **subespacio principal.**
- Para representar a $\tilde{\boldsymbol{x}}^{\boldsymbol{n}}$ en el espacio de los **componentes principales** necesitamos solamente M coordenadas: $\beta_1^n, \ldots, \beta_M^n$.

Analicemos la expresión para $\tilde{\boldsymbol{x}}^{\boldsymbol{n}}$ más en detalle:

$$\tilde{\boldsymbol{x}}^{\boldsymbol{n}} = \sum_{j=1}^{M} \beta_j^n \boldsymbol{b_j} \in \mathbb{R}^D$$

- Los vectores $\boldsymbol{b_1, \ldots, b_M}$ son los **componentes principales** y ellos mismos viven en $\mathbb{R}^D$ y están expresados en términos de las *features* originales, lo que significa que son **combinaciones lineales de las *features***.

- Los coeficientes de los **componentes principales** se llaman **loadings** y determinan el peso de cada variable original en los nuevos componentes. Estos pesos son importantes para **interpretar a los componentes principales como variables latentes.**

Antes de avanzar precisemos los **supuestos de PCA**:

1) El dataset $\boldsymbol{X = \{x^1, \dots, x^N\},\ x^n \in \mathbb{R}^D}$ está normalizado, es decir todas las variables tienen **media igual a 0 y varianza igual a 1.**

2) La base $\boldsymbol{B = (b_1, \dots, b_D)}$ forma una base ortonormal de $\mathbb{R}^D$. Recordemos que la base está expresada en nuestro sistema de coordenadas original, es decir el de las *features*.

Volviendo al problema de optimización, haciendo uso de los supuestos, es posible demostrar que podemos reformular al error cuadrático medio de reconstrucción de la siguiente manera:

$$J = \frac{1}{N}\sum_{n=1}^{N}\|\boldsymbol{x^n} - \tilde{\boldsymbol{x}}^{\boldsymbol{n}}\|^2 = \sum_{\boldsymbol{j=M+1}}^{\boldsymbol{D}} \boldsymbol{b_j^T}\left(\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x^n x^{nT}}\right)\boldsymbol{b_j}$$

Donde el supuesto de media igual a cero nos permite afirmar que:

$$\boldsymbol{S} = \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x^n x^{nT}}$$

Donde **S** es la **matriz de covarianzas** de nuestro dataset.

Obtenemos entonces la siguiente expresión:

$$J = \sum_{j=M+1}^{D} b_j^T S\, b_j = traza\left(\left(\sum_{j=M+1}^{D} b_j\, b_j^T\right) S\right)$$

La función de pérdida expresada así puede ser interpretada como la **varianza del dataset proyectada sobre el complemento ortogonal del subespacio principal.**

Esto implica que **minimizar** J es equivalente a minimizar **la varianza de los datos que viven en el subespacio que ignoramos**.

En otras palabras, queremos realizar la **proyección que retenga la mayor cantidad de varianza posible**.

La expresión reformulada de nuestra función de pérdida definida por

$$J = \sum_{j=M+1}^{D} b_j^T S\, b_j$$

se minimiza en b_j con la restricción de ortonormalidad de los vectores.

Para minimizar *J* hay que plantear la función lagrangiana donde λ_j son los **multiplicadores de Lagrange**. Resolviendo el problema de optimización, obtenemos:

$$J\ min \leftrightarrow Sb_j = \lambda_j b_j\ ,\quad \mathrm{j} = \mathrm{M}+1\ ,\ldots,\mathrm{D}$$

Obtuvimos entonces:

$$J\ min \leftrightarrow \boldsymbol{Sb_j} = \boldsymbol{\lambda_j b_j}\ ,\quad \mathrm{j} = \mathrm{M}+1\ ,\ldots,\mathrm{D}$$

Donde los vectores $\boldsymbol{b_j}$ son **autovectores** de la matriz de covarianza de **S** y los multiplicadores de Lagrange λ_j son sus **autovalores**.

DigitalHouse > Coding School

¡Repaso express de álgebra!

Definición:

Sea $A \in \mathbb{R}^{n \times n}$, $\lambda \in \mathbb{R}$ es autovalor de A si y sólo si existe un vector $v \in \mathbb{R}^{n \times 1}$ no nulo tal que:

$$A.v = \lambda.v \quad , \quad v \neq 0_V$$

v se llama autovector asociado a λ.

Es decir que los **autovectores** de A son los vectores que **conservan la dirección** después de la transformación. Los autovectores pueden dilatarse por el factor λ e incluso cambiar de sentido, pero no de dirección.

¡Repaso express de álgebra!

2 propiedades importantes de los autovectores:

1) Los autovectores asociados a autovalores distintos son **linealmente independientes**.
2) Si **A** es una **matriz simétrica**, los autovalores son reales y autovalores distintos tienen asociados **autovectores ortogonales**.

Obtuvimos entonces:

$$J\ min \leftrightarrow \boldsymbol{S}\boldsymbol{b}_j = \boldsymbol{\lambda}_j\boldsymbol{b}_j\ ,\quad \mathrm{j} = \mathrm{M}+1\ ,\ldots,\mathrm{D}$$

Donde los vectores $\boldsymbol{b}_j$ son **autovectores** de la matriz de covarianza de **S** y los multiplicadores de Lagrange λ_j son sus **autovalores**.

Resulta entonces que el **valor mínimo** del error cuadrático medio de reconstrucción es:

$$J = \sum_{j=M+1}^{D} \boldsymbol{\lambda}_j$$

Resulta entonces que el **valor mínimo** del error cuadrático medio de reconstrucción es:

$$J = \sum_{j=M+1}^{D} \lambda_j$$

Es decir que para minimizar la función de pérdida, tenemos que elegir como base del subespacio que vamos a ignorar a los D-M **autovectores de la matriz de covarianza asociados a los autovalores más chicos**.

Esto equivale a decir que los **componentes principales son los autovectores de la matriz de covarianza asociados a los autovalores más grandes** y estos **autovalores corresponden a la varianza de la dirección del autovector**.

Descomposición por autovectores y autovalores

Como **S** es una matriz simétrica, la podemos descomponer de la siguiente manera:

$$\boldsymbol{S = B\Lambda B^{-1}} = \begin{pmatrix} b_{11} & \cdots & b_{1D} \\ \vdots & \ddots & \vdots \\ b_{D1} & \cdots & b_{DD} \end{pmatrix} \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_D \end{pmatrix} \begin{pmatrix} b_{11} & \cdots & b_{1D} \\ \vdots & \ddots & \vdots \\ b_{D1} & \cdots & b_{DD} \end{pmatrix}^{-1}$$

Donde $\boldsymbol{B}$ es una matriz cuyas columnas son los autovectores de **S** y $\boldsymbol{\Lambda}$ es una matriz diagonal cuyos elementos son los autovalores de **S**, ordenados de mayor a menor.

Descomposición por autovectores y autovalores

Como **S** es una matriz simétrica, la podemos descomponer de la siguiente manera:

$$\boldsymbol{S = B\Lambda B^{-1}} = \begin{pmatrix} b_{11} & \cdots & b_{1D} \\ \vdots & \ddots & \vdots \\ b_{D1} & \cdots & b_{DD} \end{pmatrix} \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_D \end{pmatrix} \begin{pmatrix} b_{11} & \cdots & b_{1D} \\ \vdots & \ddots & \vdots \\ b_{D1} & \cdots & b_{DD} \end{pmatrix}^{-1}$$

Reducir la dimensionalidad de nuestro dataset con PCA, consiste entonces en **tomar los primeros M vectores de B** como los componentes principales.

Descomposición por autovectores y autovalores

Una vez que nos quedamos con las primeras M columnas de $\boldsymbol{B}$ quedándonos con la matriz reducida $\boldsymbol{B_{red}}$ de dimensiones DxM, podemos **pasar del espacio de las variables originales al espacio de los componentes principales** de la siguiente manera:

$$\boldsymbol{z^n = B_{red}^T x^n}$$

Donde el vector resultante es de dimensión M y sus elementos son $\boldsymbol{\beta_1^n, \ldots, \beta_M^n}$

Varianza explicada y selección de número de Componentes Principales

Vimos que los **autovalores** λ_j corresponden a la **varianza que captura cada componente principal**.

Por lo tanto, la varianza que vamos a poder retener con los M Componentes Principales será:

$$Varianza\ Explicada = \sum_{j=1}^{M} \lambda_j$$

Varianza explicada y selección de número de Componentes Principales

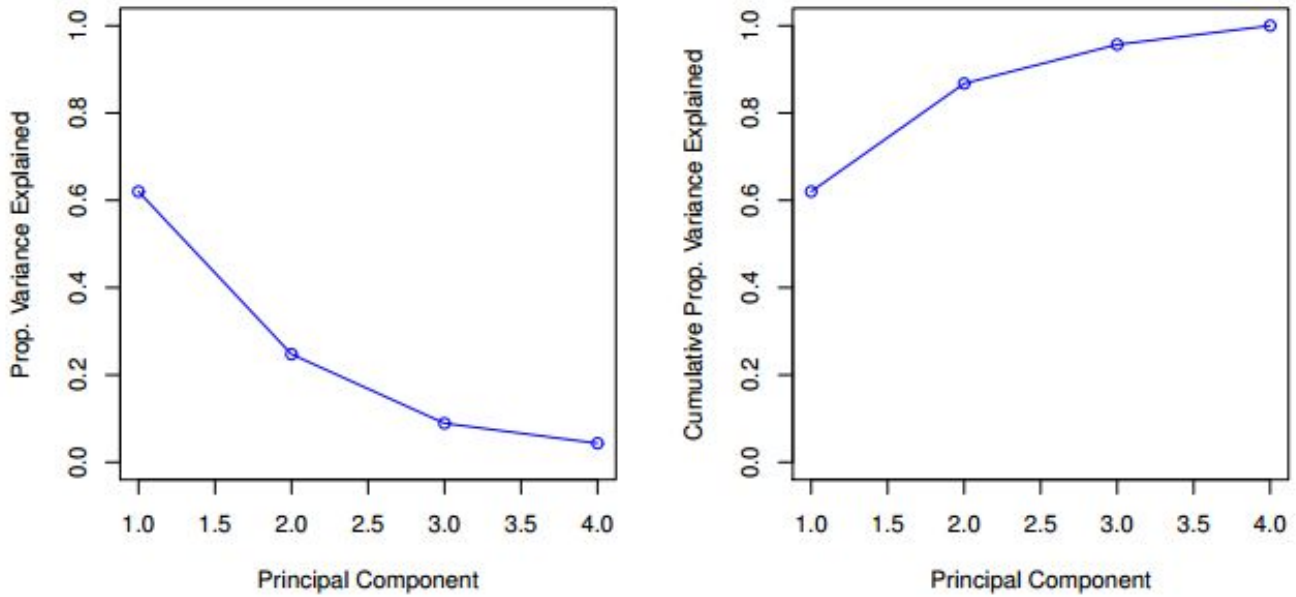
$$Varianza\ Explicada = \sum_{j=1}^{M} \lambda_j$$

Como nos interesa conocer la varianza explicada en relación a la varianza total de nuestro dataset original, normalmente vamos a prestarle atención a:

$$Porcentaje\ Varianza\ Explicada = \frac{\sum_{j=1}^{M} \lambda_j}{\sum_{j=1}^{D} \lambda_j}$$

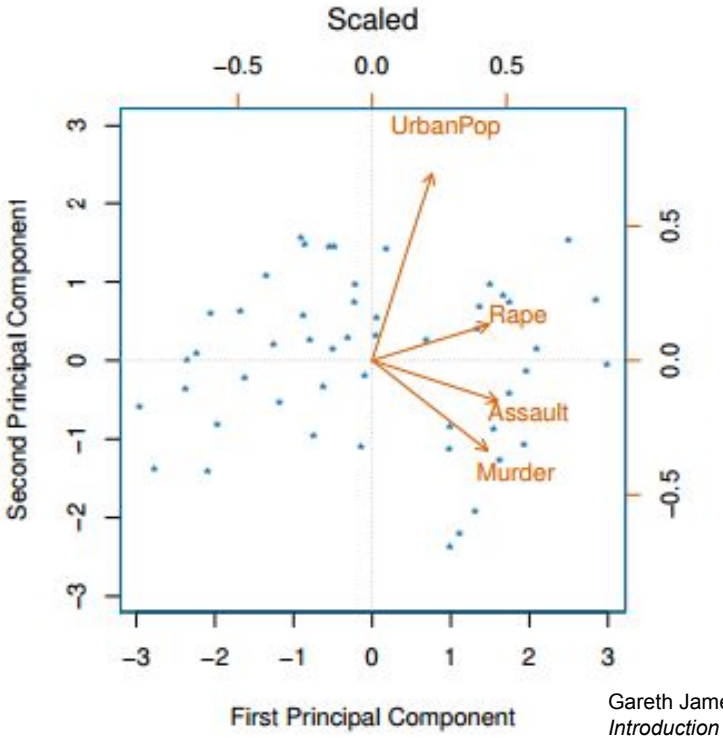
De esta forma, vamos podemos definir M tal que podamos retener un porcentaje aceptable de varianza.

Aporte individual y acumulado de varianza explicada a medida que agregamos componentes principales:



Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. *An Introduction to Statistical Learning - with Applications in R*

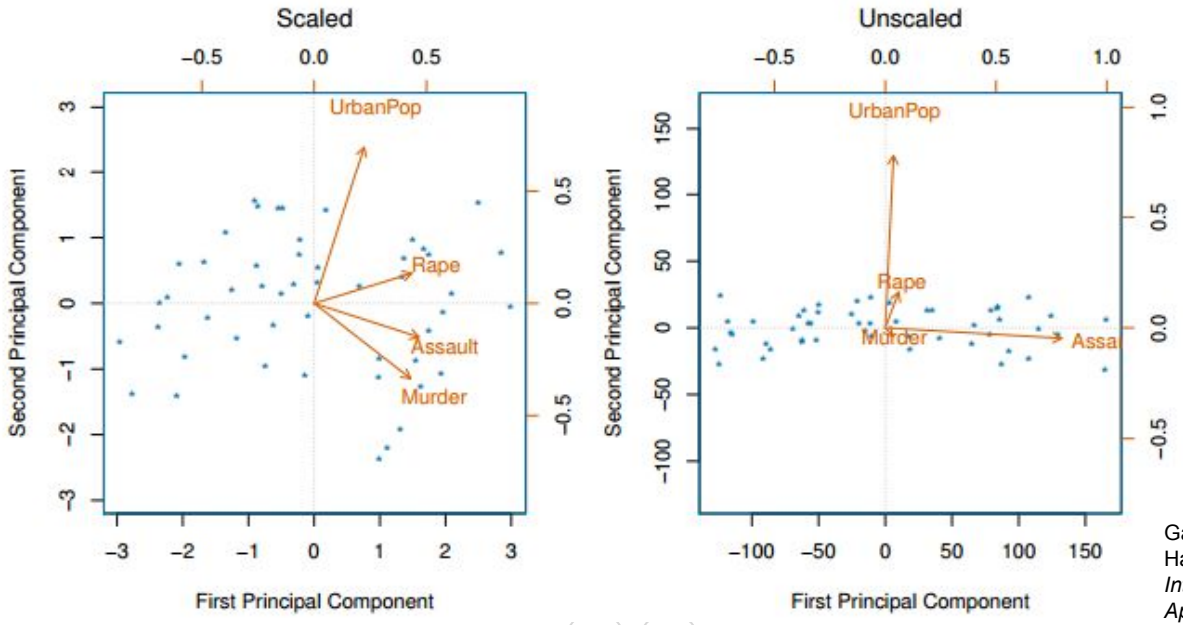
Los componentes principales, pueden ser la expresión de **variables latentes**.



Por ejemplo: en este caso un Estado con alto valor en el primer componente es un Estado inseguro y uno con alto valor en el segundo componente es un Estado predominantemente urbano.

Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. *An Introduction to Statistical Learning - with Applications in R*

Las variables en PCA son **sensibles a la escala. Deben estar estandarizadas para que todas tengan la misma varianza y estén en la misma escala.** Si una variable está en una escala que le da mayor representación que a las demás puede dominar la formación de los componentes principales. Lo mismo sucede si las varianzas son distintas.

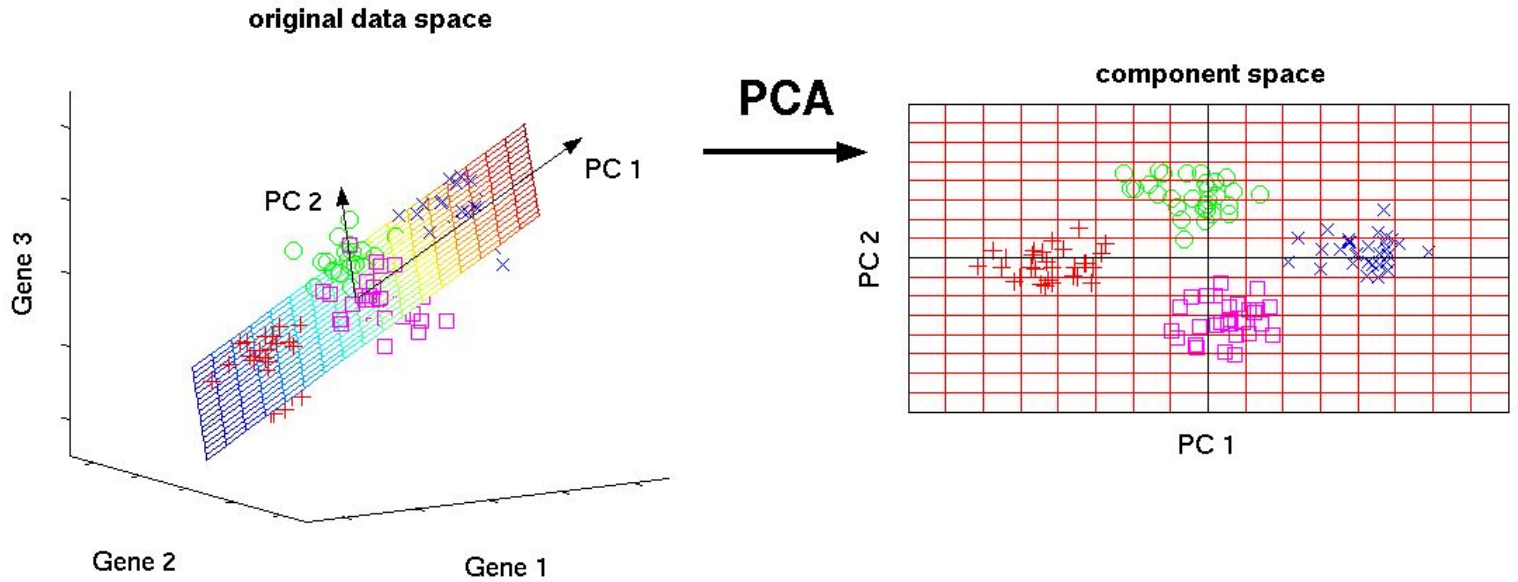


Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. *An Introduction to Statistical Learning - with Applications in R*

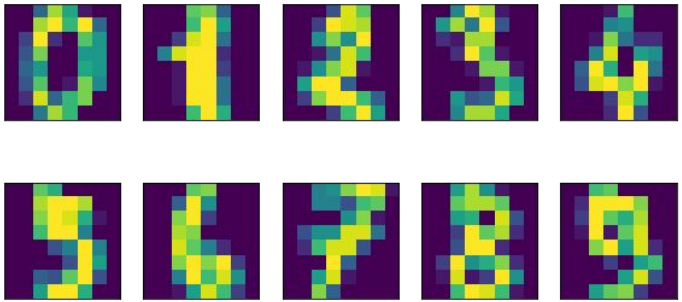
- PCA es una técnica no supervisada en la que ninguna variable juega el rol de target.
- Permite reducir la dimensionalidad de los datos descartando información redundante o ruido.
- Nos permite lograr una representación gráfica de la información multidimensional.
- Si las escalas y/o las varianzas son distintas entre las X, hay que estandarizar las variables.
- PCA se basa en la matriz de correlaciones. Por lo tanto sólo tiene sentido cuando trabajamos con **variables cuantitativas.** Existen otras técnicas para trabajar con datasets de variables categóricas o mixtas: análisis de correspondencias y categorical PCA.

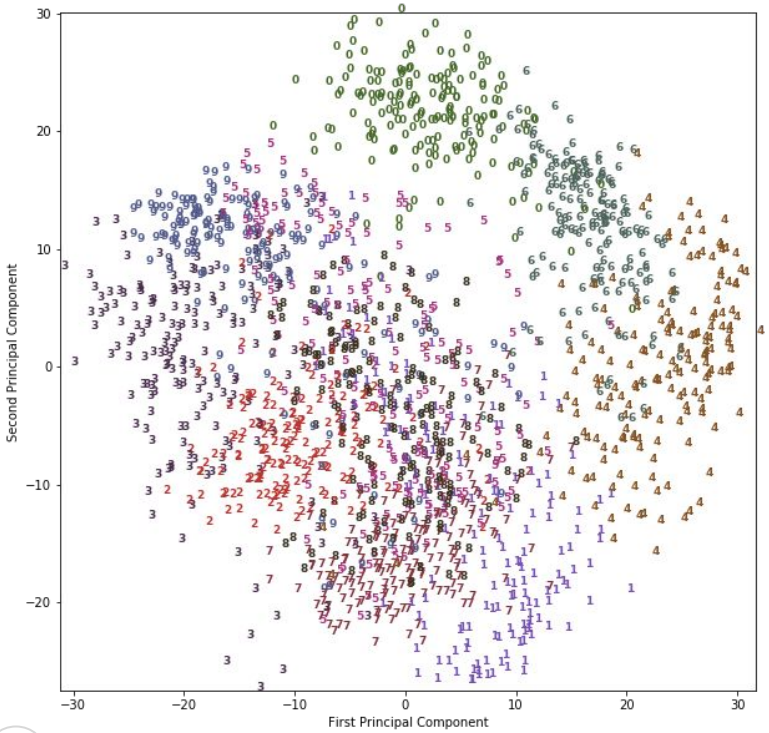
Manifold Learning: técnicas no lineales de reducción de la dimensionalidad

PCA es una técnica no supervisada para reducir la dimensionalidad de los datos basada en **transformaciones lineales**. Cambiamos el sistema de coordenadas y luego descartamos las dimensiones menos relevantes:

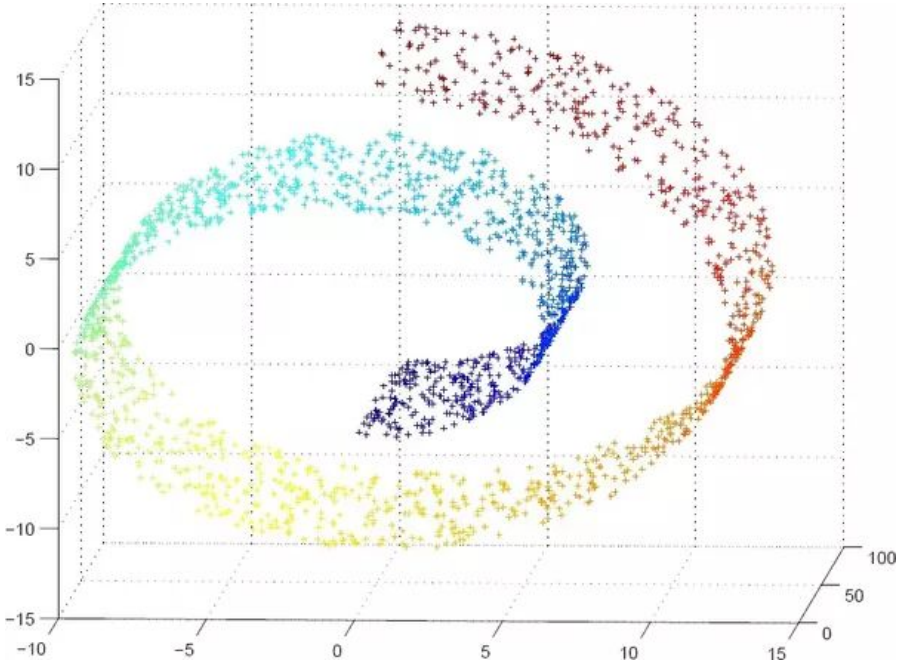


PCA no funciona demasiado bien si las relaciones entre los datos exhiben fuertes rasgos de **no linealidad.** Veamos su performance con el dataset de dígitos:

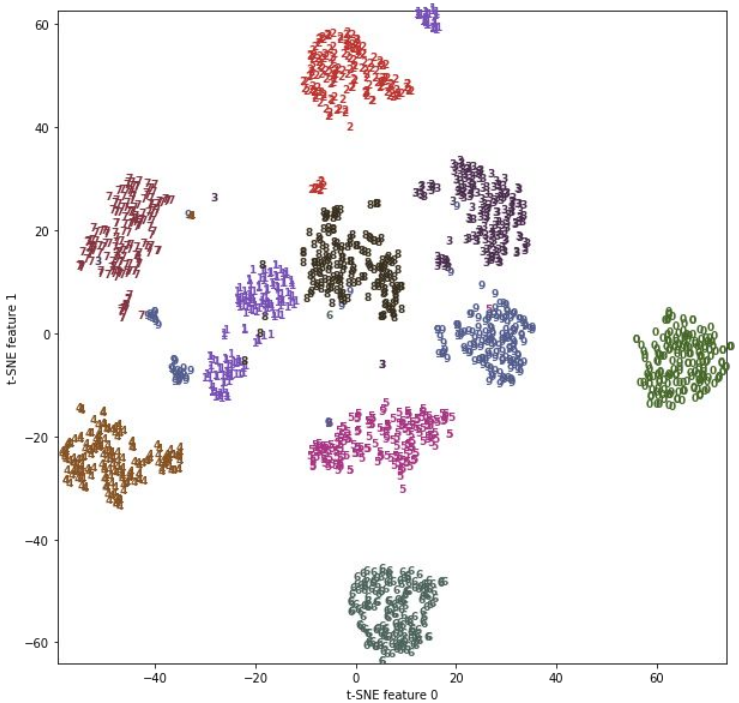




Técnicas de **Manifold Learning** para reducción de la dimensionalidad: buscan describir un dataset como un **geometría de baja dimensionalidad "embebida" en un espacio multidimensional no lineal**.



Vamos a ver cómo opera una técnica de **Manifold** con el dataset de dígitos:



La técnica que usamos se llama **T-SNE.** Busca generar una representación en 2D que **preserve la distancia entre los puntos**: los puntos que están cerca en el dataset original quedan cerca en el transformado y los que están lejos, quedan lejos.

Manifold Learning como reducción de la dimensionalidad

- Dado un set de datos en un espacio multidimensional busca una **representación de menor dimensionalidad** que preserve ciertas relaciones dentro de los datos.
- Son una familia de estimadores **no supervisados**.
- Hay diferentes técnicas de Manifold, cada una busca estimar o aproximar el espacio no lineal de diferentes maneras. Vamos a ver dos:
  - T-SNE
  - Isometric Mapping (IsoMap)

— Uso fundamental de estas técnicas: **visualización**

— Diferencias con PCA:

- La presencia de ruido en los datos puede alterar drásticamente los embeddings. PCA, por contraste, "filtra" los ruidos a través de los componentes principales.
- El óptimo global en cantidad de dimensiones suele ser difícil de determinar. PCA, en cambio, tiene un criterio: la cantidad de varianza que explican los componentes.
- Tienen complejidad cuadrática o cúbica.

— Ventaja fundamental: capacidad de preservar relaciones no lineales. Por esta razón, suele ser útil realizar exploraciones de datos con manifold learning, luego de haberlos explorado con PCA.