



# Inteligencia Artificial Aplicada para la Economía



## Profesores

### Profesor Magistral

Camilo Vega Barbosa

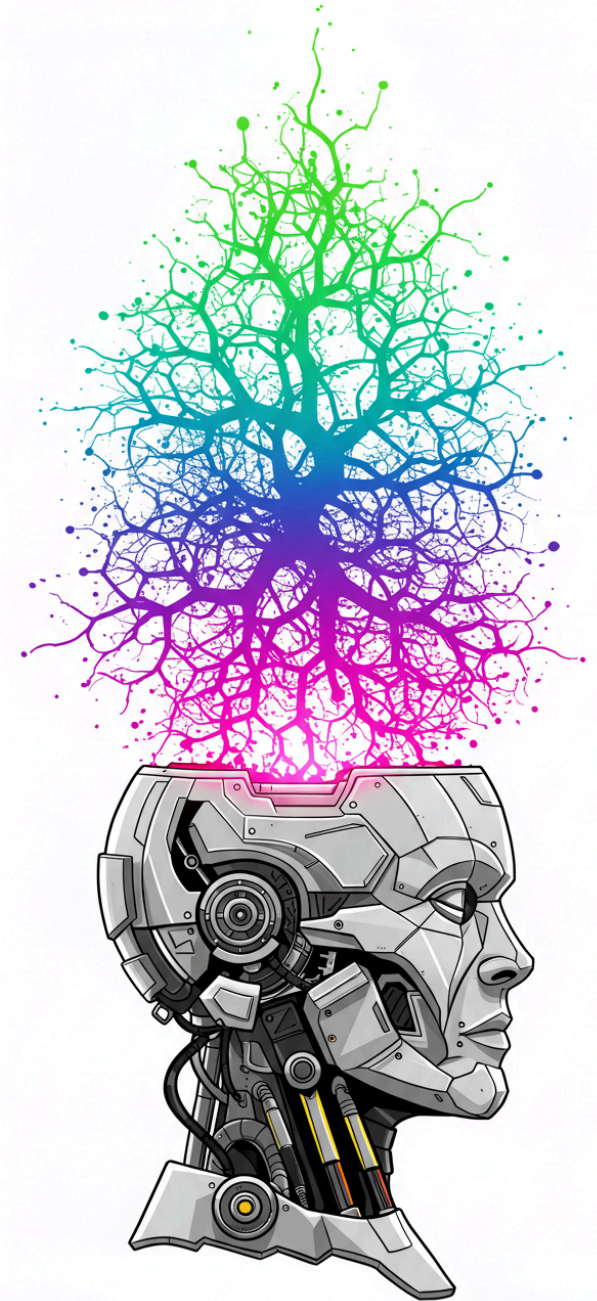
### Asistente de Docencia

Sergio Julian Zona Moreno



# Algoritmos basados en distancias

Del Concepto a la Implementación



# Algoritmos que Cubriremos

## Aprendizaje Supervisado

- **KNN**: Clasificación/regresión basada en vecinos más cercanos
- **SVM**: Clasificación mediante hiperplanos óptimos

## Aprendizaje No Supervisado

- **K-means**: Agrupamiento de datos similares
- **PCA** (*No algoritmo de clasificación*): Reducción de dimensionalidad y extracción de características (*en Supervisado y no Supervisado*)

## Elemento Común:




Los algoritmos se basan en el concepto de **distancia** o **similitud** entre puntos de datos



# Algoritmos Basados en Distancia

Los algoritmos basados en distancia son una familia de métodos que funcionan midiendo qué tan "cerca" o "lejos" están los datos entre sí, similar a cómo usamos un mapa para encontrar lugares cercanos. En el mundo de los datos, esta "distancia" puede representar cualquier tipo de similitud o diferencia entre observaciones.

## Aspectos Clave

-  Utilizan métricas como distancia euclidiana
-  Asumen que puntos cercanos son similares
-  Son intuitivos y fáciles de interpretar

## Consideración Principal




La "maldición de la dimensionalidad" afecta su rendimiento en espacios de alta dimensión. Pero hay soluciones que vamos a estudiar.

# K-Nearest Neighbors (KNN)

El algoritmo KNN es como tener un consejo de vecinos: cuando necesitas tomar una decisión, consultas con tus K vecinos más cercanos y sigues la opinión mayoritaria.

## La Intuición

Imagina una biblioteca donde los libros similares están cerca entre sí. Si encuentras un nuevo libro:

-  Miras los 3 libros más cercanos
-  Observas sus géneros
-  Asignas el género más común

# KNN: Matemática y Aplicaciones

## Matemática del KNN

Para dos puntos  $p$  y  $q$ :




$$d(p, q) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}$$

### Proceso

1. Calcular distancias
2. Seleccionar  $K$  vecinos
3. Votar/promediar resultado

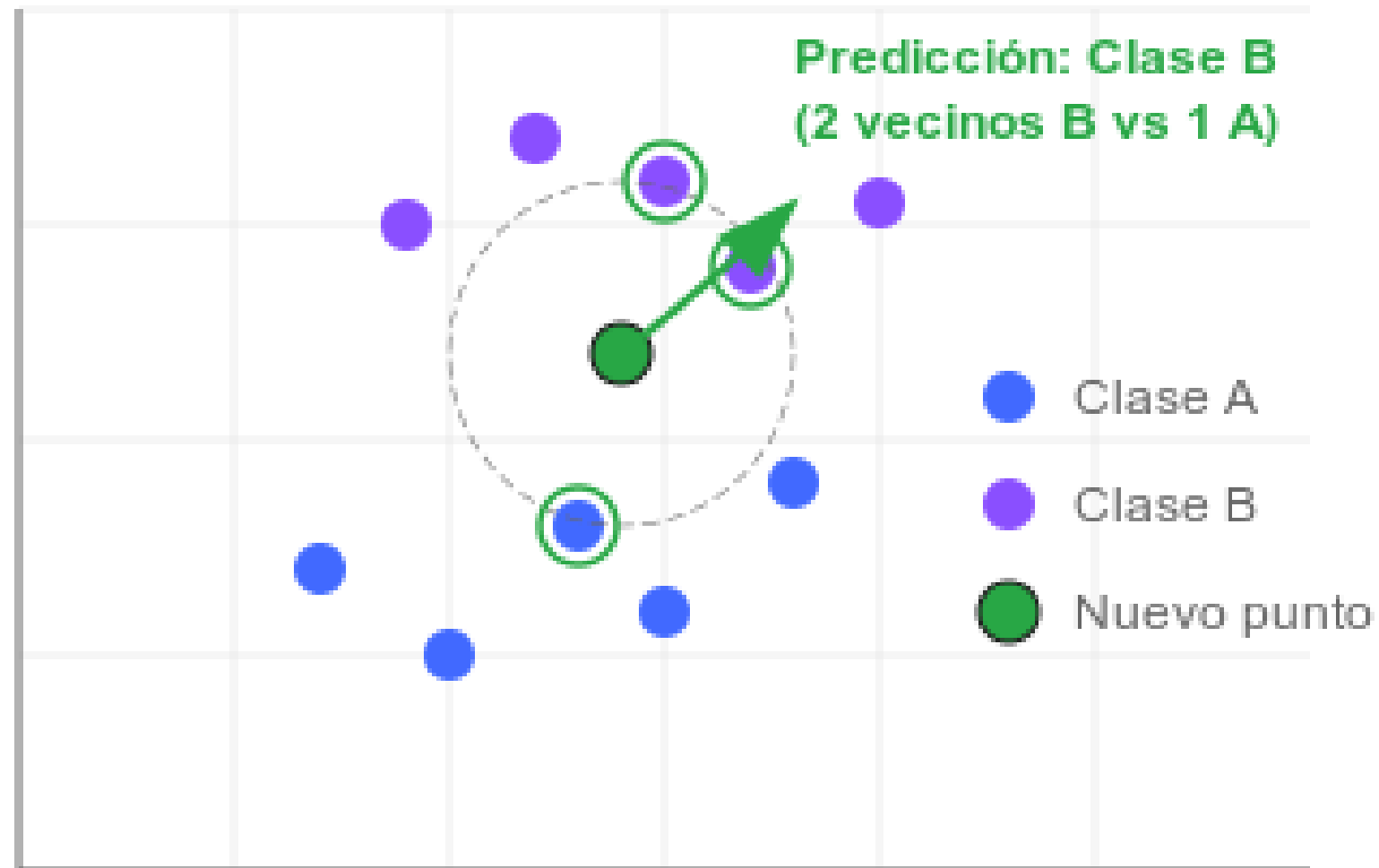
## Aplicaciones y Consideraciones

### Casos en Economía

-  Clasificación de riesgo crediticio
-  Predicción de mercados
-  Segmentación de clientes

### Aspectos Clave

- $K$  pequeño → Sensible a ruido
- $K$  grande → Más estable
- Ideal: datasets pequeños



## El Arte de Elegir K

La elección del número de vecinos (K) es crucial en KNN. Es un balance entre estabilidad y precisión: un K muy pequeño hace el modelo sensible al ruido, mientras que uno muy grande puede perder patrones locales importantes.

### Reglas Matemáticas Sugeridas



- Regla de la Raíz:  $K = \sqrt{n}$  donde n es el tamaño del dataset
- Regla Impar: Si K es par, sumar 1 para evitar empates



# Support Vector Machines (SVM)

Las Máquinas de Vectores de Soporte (SVM) son algoritmos versátiles de aprendizaje supervisado que pueden aplicarse tanto a **clasificación** como a **regresión**. Su principio fundamental cambia según la tarea: para clasificación busca el hiperplano óptimo que separa clases, mientras que para regresión busca una función que capture la relación entre variables.

## La Intuición

-  **Para Clasificación:** Buscar la mejor "frontera" que separe categorías
-  **Para Regresión:** Encontrar la línea que mejor ajuste los datos con margen de tolerancia

# SVM para Clasificación vs Regresión

## SVM Clasificación (SVC)

**Objetivo:** Separar clases con máximo margen

$$f(x) = \mathbf{w} \cdot \mathbf{x} + b = 0$$

**Proceso**

1. Maximizar margen:  $\frac{2}{\|\mathbf{w}\|}$
2. Sujeto a:  $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1$
3. Decisión:  $\text{sign}(f(x))$

## SVM Regresión (SVR)

**Objetivo:** Predecir valores continuos con margen de tolerancia ( $\varepsilon$ )

$$f(x) = \mathbf{w} \cdot \mathbf{x} + b$$



**Proceso**

1. Minimizar:  $\frac{1}{2} \|\mathbf{w}\|^2 + C \sum \xi_i$
2. Sujeto a:  $|y_i - f(x_i)| \leq \varepsilon + \xi_i$
3. Predicción: valor numérico directo



# Aplicaciones de SVM: Clasificación vs Regresión



## SVM Clasificación - Casos en Economía

-  Predicción de crisis financieras (Crisis/No Crisis)
-  Evaluación de riesgo crediticio (Alto/Bajo Riesgo)

### Aspectos Clave

- Salida: Clases discretas
- Métricas: Accuracy, Precision, Recall, F1-Score

## SVM Regresión - Casos en Economía

-  Predicción de precios (valores continuos)
-  Estimación de demanda (cantidades)

### Aspectos Clave

- Salida: Valores continuos
- Métricas: MSE, RMSE, MAE,  $R^2$



## El Truco del Kernel

El kernel es una función matemática que permite tanto a SVM de clasificación como de regresión operar en espacios de mayor dimensión sin calcular explícitamente las coordenadas en ese espacio.

## Kernels Principales y sus Ecuaciones

- Kernel Lineal:

$$K(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$$

- Kernel Polinomial:


$$K(\mathbf{x}_1, \mathbf{x}_2) = (\gamma \mathbf{x}_1^\top \mathbf{x}_2 + c)^d$$

- Kernel RBF (Gaussiano):

$$K(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\gamma \|\mathbf{x}_1 - \mathbf{x}_2\|^2)$$

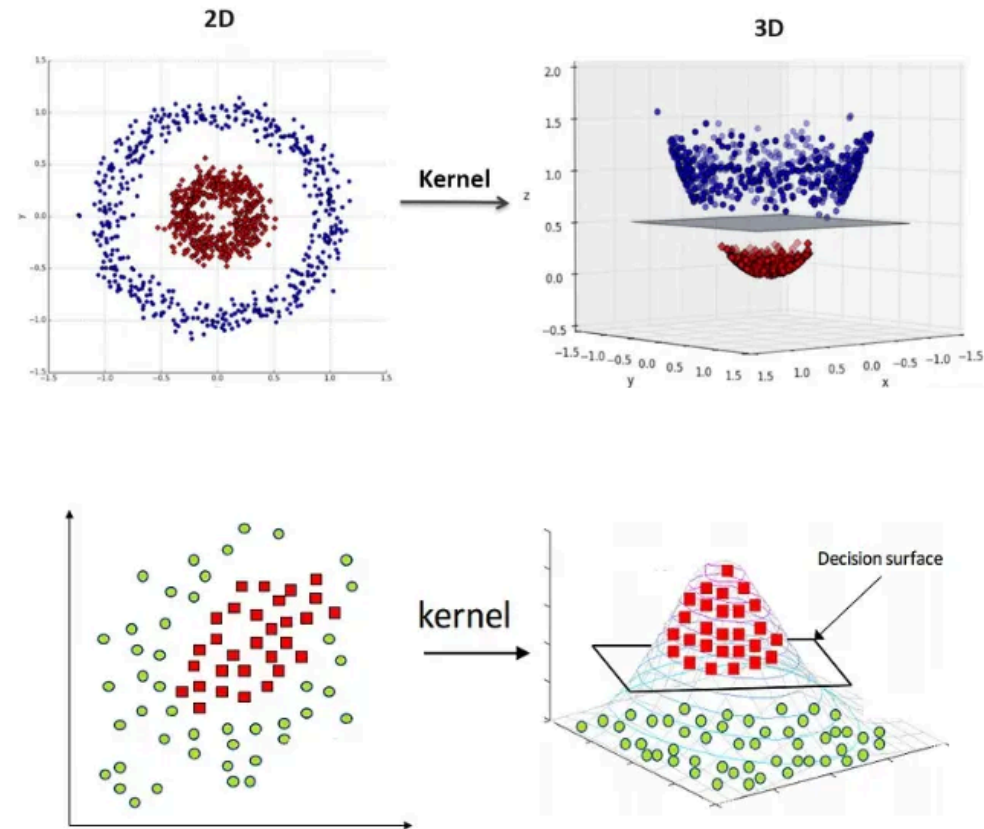
- Kernel Sigmoid:

$$K(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\gamma \mathbf{x}_1^\top \mathbf{x}_2 + c)$$

 **Nota:** Los kernels funcionan igual para clasificación y regresión, la diferencia está en la función objetivo y la interpretación de resultados.



# Visualización Kernel



Tomado de: <https://medium.com/@abhishekjainindore24/svm-kernels-and-its-type-dfc3d5f2dcd8>



# Aprendizaje No Supervisado





El aprendizaje no supervisado representa una rama fascinante del machine learning donde los algoritmos exploran patrones y estructuras ocultas en **datos sin etiquetas**. A diferencia del aprendizaje supervisado, aquí no hay "respuestas correctas" que guíen al modelo.

## Algoritmos Principales

- **K-means**: Para descubrir grupos naturales en los datos
- **PCA**: Para reducir la dimensionalidad y encontrar las características más importantes

# K-means: Proceso y Características

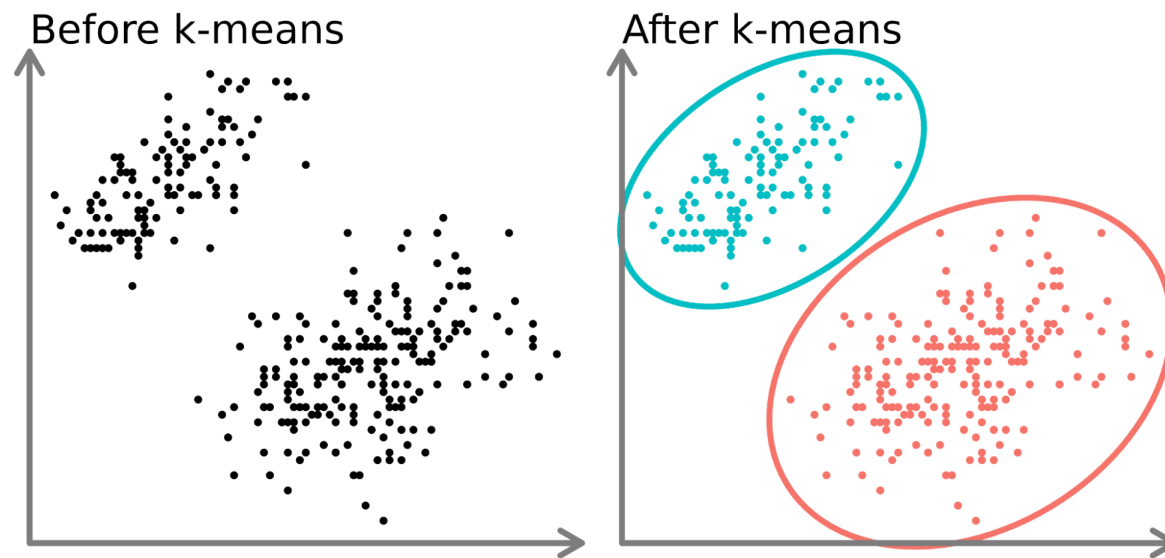
## El Algoritmo

1.  Inicialización: K centroides aleatorios
2.  Asignación a centroides cercanos
3.  Actualización de centroides
4.  Repetir hasta convergencia

## Características Clave

- Convergencia garantizada
- Proceso intuitivo y simple
- Solución localmente óptima
- Eficiente computacionalmente

## Visualización de K-means



Tomado de: <https://www.datacamp.com/es/tutorial/k-means-clustering-python>

# K-means: Aspectos Técnicos

## Formulación Matemática

$$J = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - c_j\|^2$$

Donde:

- $x_i^{(j)}$ : puntos en cluster j
- $c_j$ : centroide del cluster j

## Consideraciones

- Inicialización afecta resultado
- Elección de K es crucial
- Usar métricas de validación
- Sensible a outliers

# Eligiendo el K Óptimo

La elección del número de clusters (K) es crucial para el rendimiento del algoritmo. Existen varios métodos para encontrar el K óptimo, pero el más común es el método del codo.

## Método del Codo (Elbow Method)

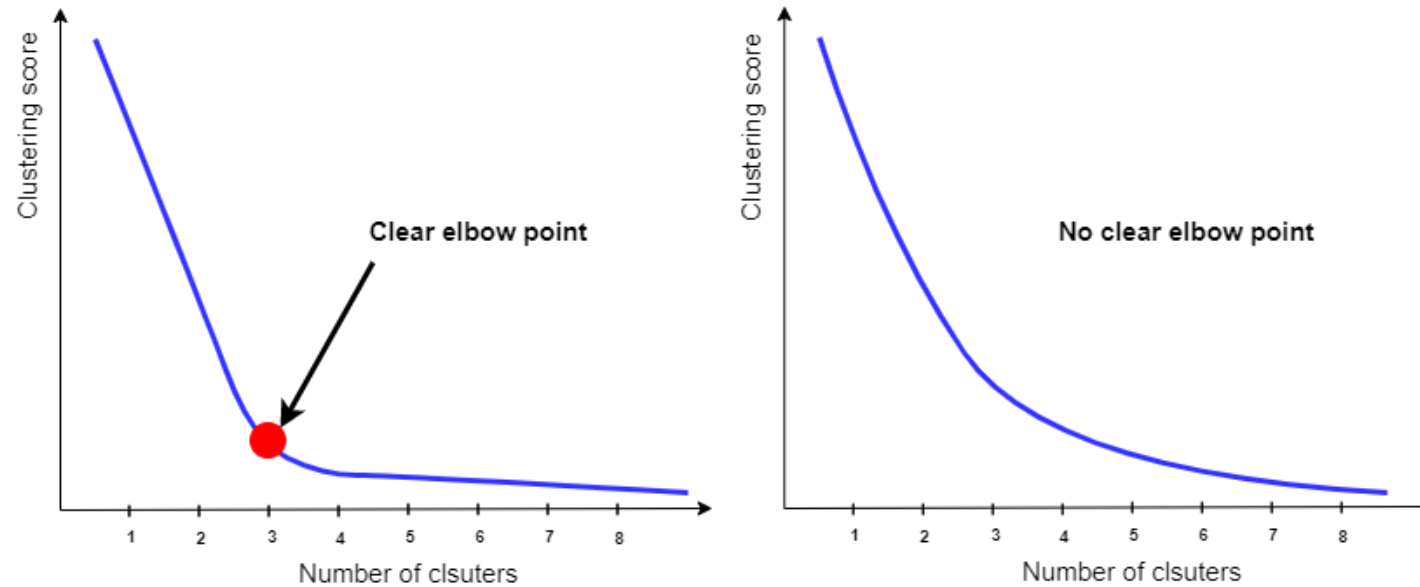
- Grafica la inercia vs número de clusters
- Busca el "codo" en la curva
- Inercia (*Clustering Score*) =  $\sum_{i=0}^n \min_{\mu_j \in C} (\|x_i - \mu_j\|^2)$

## Intuición del Clustering Score

Mide qué tan "compactos" son nuestros clusters - es la suma de las distancias de cada punto a su centro más cercano. Un score más bajo indica grupos más cohesionados.



# Visualización del Método del Codo



Tomado de: <https://www.kaggle.com/code/mafaisal007/k-means-clustering-tutorial>






# PCA: Análisis de Componentes Principales

El Análisis de Componentes Principales (PCA) es una técnica fundamental que nos permite reducir la dimensionalidad de nuestros datos mientras preservamos la máxima varianza posible. **No es un algoritmo** en el sentido que no clasifica, pero es una técnica que ayuda a transformar datos para luego aplicar un algoritmo **supervisado o no supervisado**.

## La Intuición

Imagina una fotografía 3D de una hoja de papel:

-  Aunque tiene 3 dimensiones
-  Realmente solo necesitas 2 para describirla
-  PCA encuentra estas dimensiones importantes

# PCA: Matemática y Aplicaciones

## Matemática del PCA

La transformación PCA:




$$X_{pca} = XW$$

Donde  $W$  son los eigenvectores de:

$$\Sigma = \frac{1}{n-1} X^T X$$

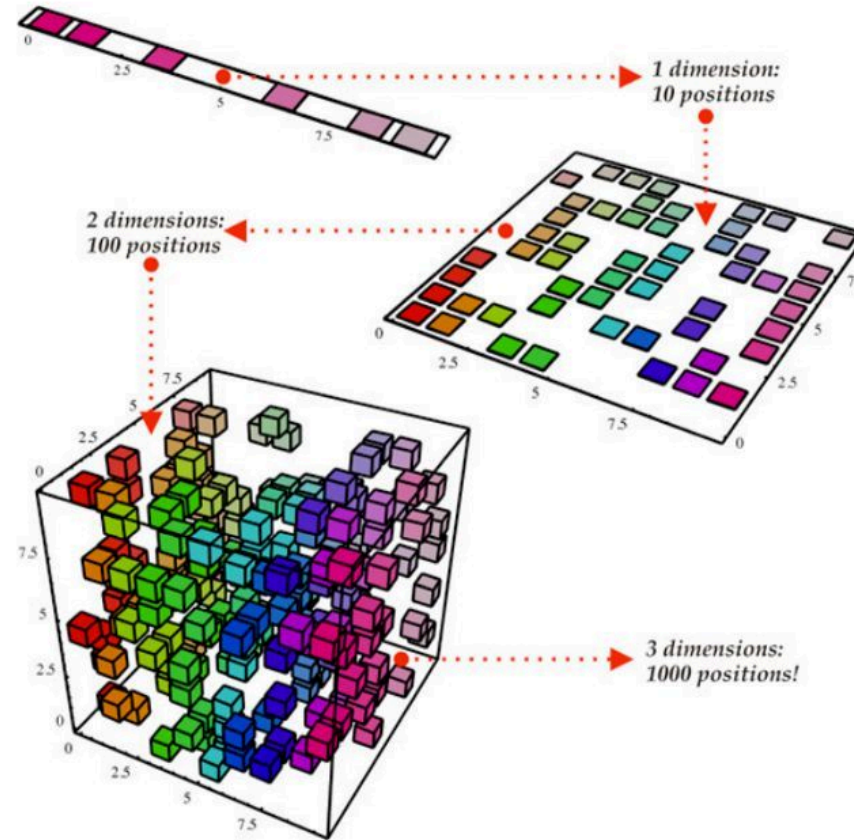
## Aplicaciones y Consideraciones

### Casos en Economía

-  Análisis de mercados financieros
-  Indicadores macroeconómicos
-  Reducción de variables en modelos



# Visualización de Reducción dimensionalidad



Tomado de: <https://co.pinterest.com/pin/440578776072738358/>

# Selección de Componentes en PCA: Intuición

El PCA ordena componentes por importancia. El desafío está en elegir cuántos mantener, balanceando simplicidad y preservación de información.

## Conceptos Fundamentales

Cada componente captura una dirección de máxima varianza, siendo independientes entre sí. Como en una orquesta, cada uno aporta su parte única a la variabilidad total.

## Una Analogía Visual

Una foto digital ilustra bien este concepto: aunque tiene millones de píxeles, podemos capturar su esencia con menos componentes. En PCA buscamos lo mismo: mantener lo esencial con el mínimo necesario.



# Métodos Cuantitativos de Selección



## Varianza Explicada Acumulada

Nuestra principal métrica de selección:

$$VE = \frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^n \lambda_i}$$

Buscamos explicar entre 80-90% de la varianza total.



## Criterio de Kaiser

Mantenemos componentes con *eigen valores* o valores propios ( $\lambda$ )  $> 1$ , que aportan más información que una variable original.

## Recursos del Curso

### Plataformas y Enlaces Principales

#### GitHub del curso

 [github.com/CamiloVga/IA\\_Aplicada](https://github.com/CamiloVga/IA_Aplicada)

#### Asistente IA para el curso

 [Google Notebook LLM](#)