

# Análisis espacial de datos y sus aplicaciones en Python

**Profesor:** Germán González

**Sesión 6:** Agrupación y optimización

# Índice

Identificación de rutas y tiempo de recorrido

K-medias

K-prototipos

DBSCAN

# Índice

Identificación de rutas y tiempo de recorrido

K-medias

K-prototipos

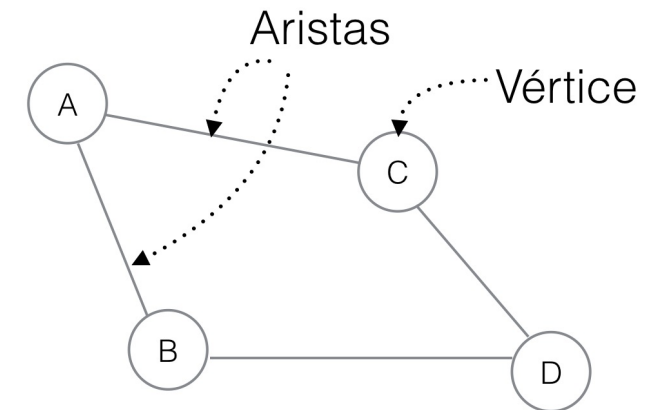
DBSCAN

# Grafos

Composición de un conjunto de objetos conocidos como nodos que se relacionan con otros nodos a través de un conjunto de conexiones conocidas como aristas.

Sea  $G$  un grafo, cuya relación es  $G=(N,A)$  donde  $n$  es una colección de puntos llamados nodos (o vértices) unidos por líneas llamadas arista. Cada arista une dos vértices.

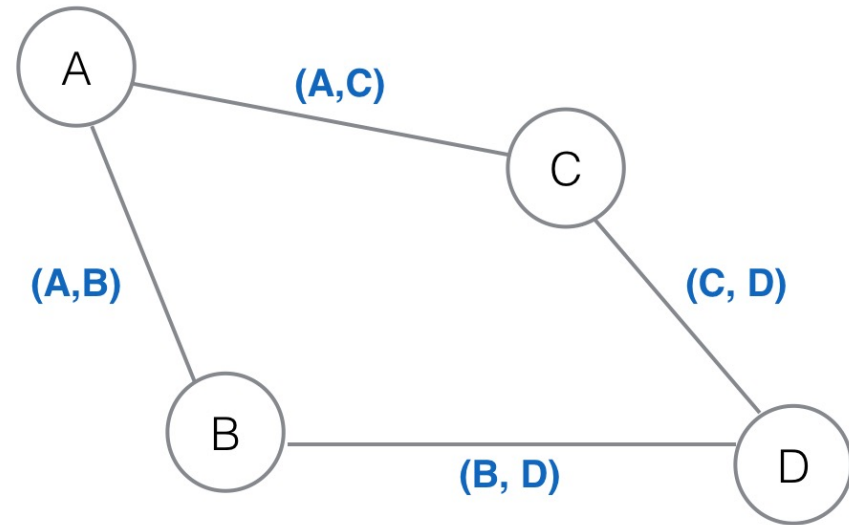
Los grafos permiten estudiar las relaciones que existen entre unidades que interactúan con otras.



Vértices (Nodos): A, B, C y D

## Grafos no ordenados

El orden de los vértices no define nada de información, pero sí indica qué vértices están conectados entre sí.



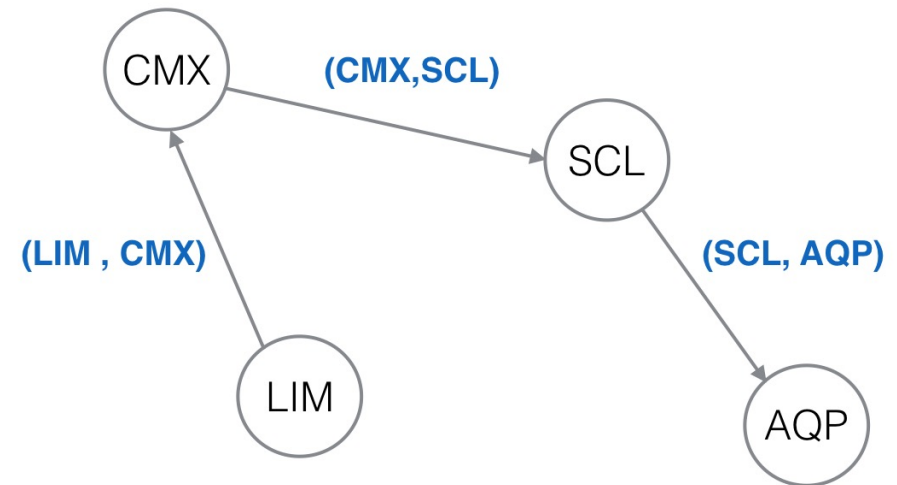
Vértices (Nodos): A, B, C y D  
Aristas: (A,B), (A,C), (C,D), (B,D)

## Grafos dirigido

Los vértices tienen un orden y su vez indica qué vértices están conectados entre si. El orden se representa con flechas en las aristas.

**Ejemplo:** Trayecto de un avión de carga:

Lima (LIM) – Ciudad de México (CMX) –  
Santiago (SCL) – Arequipa (AQP)



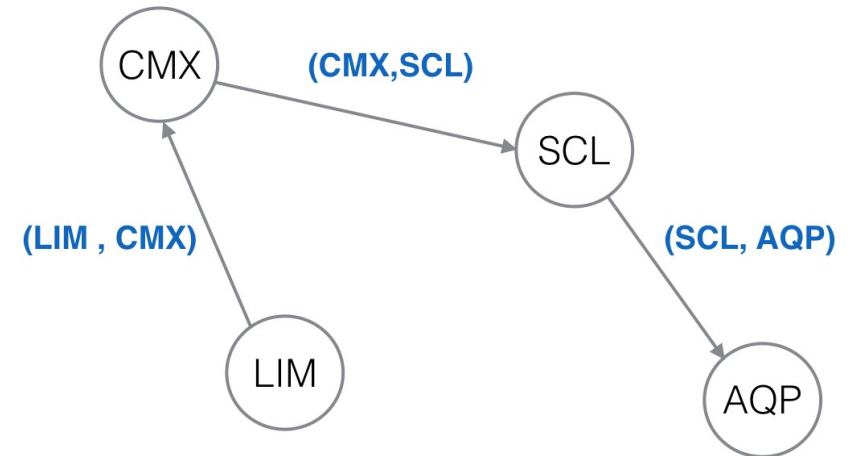
Vértices (Nodos): LIM, CMX, SCL y AQP  
Aristas: (LIM,CMX),(CMX,SCL),(SCL,AQP)

## Camino (Path)

En un Grafo dirigido se llama camino a la secuencia de vértices que conectan las aristas del grafo.

En el ejemplo el camino es [LIM, CMX, SCL, AQP]. Este camino es usando 3 aristas para conectar los vértices.

La cantidad de aristas que usa un camino se le llama el ***largo (length)*** del camino

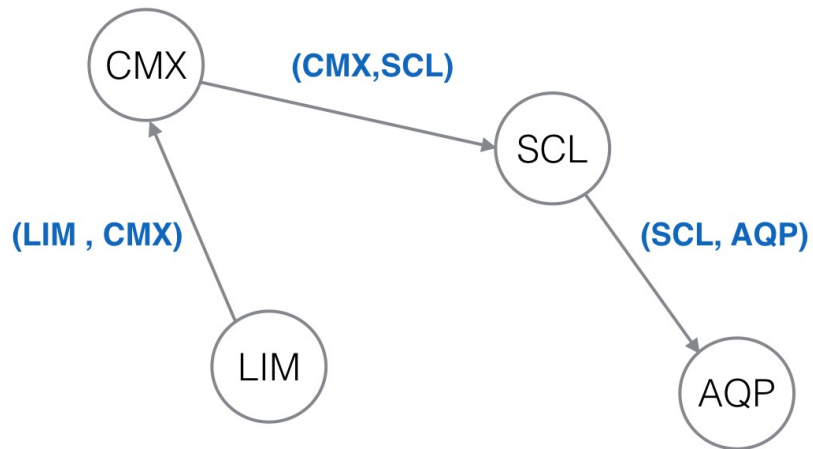


Vértices (Nodos): LIM, CMX, SCL y AQP

Aristas: (LIM,CMX),(CMX,SCL),(SCL,AQP))

# Matriz de adyacencia

La matriz de adyacencia es una matriz cuadrada que se utiliza como una forma de representar relaciones binarias.



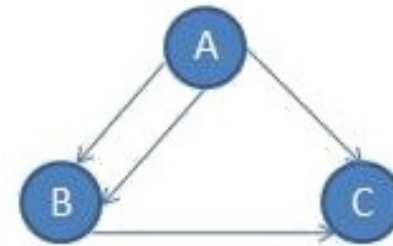
	LIM	CMX	SCL	AQP
LIM	0	1	0	0
CMX	1	0	1	0
SCL	0	1	0	1
AQP	0	0	1	0



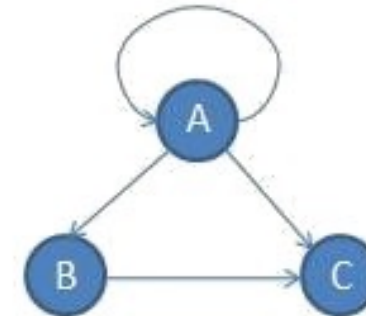
# Multigrafo

Este tipo de grafos tienen la capacidad de tener múltiples aristas que relacionan los mismos nodos. De esta forma, dos nodos pueden estar conectados por más de una arista.

Sea  $G$  un multigrafo,  $G=(N,A)$  donde  $n$  es una colección de puntos llamados nodos (o vértices) , y  $A$  es un multiconjunto de pares no ordenados de nodos.



*Multigrafo*



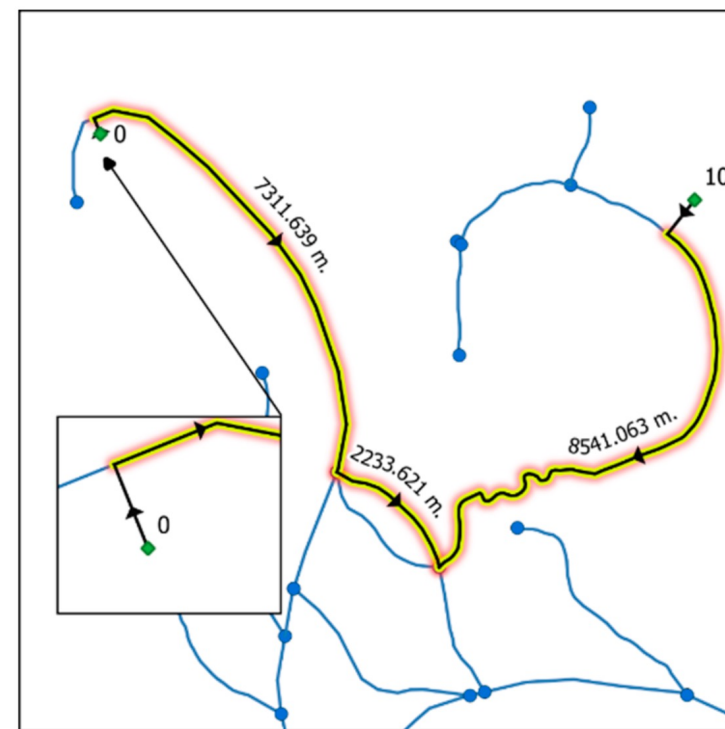
*Multigrafo*



# OSMNX

- OSMnx es un paquete de Python que le permite descargar geometrías espaciales y modelar, proyectar, visualizar y analizar redes de calles del mundo real desde las API de OpenStreetMap.

## Calles - Aristas



# Dijkstra's

Objetivo: Encontrar la ruta más corta (camino) que conecta a dos puntos en un grafo. Puede utilizar grafos dirigidos y no dirigidos.

**Paso 1:** Establecer vértices objetivo: (Punto de llegada y punto de salida)

**Paso 2:** Se elige el nodo de inicio y se evalúa el costo a los vértices adyacentes

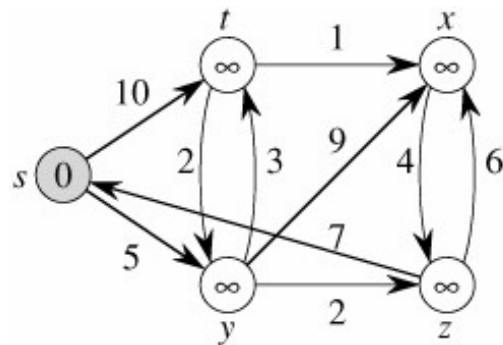
**Paso 3:** Se selecciona el nodo adyacente con costo mínimo y se fija como nodo de inicio.

**Paso 4:** Los pasos 2 y 3 deben repetirse teniendo en cuenta que el costo se va acumulando a medida que se va visitando cada nodo.

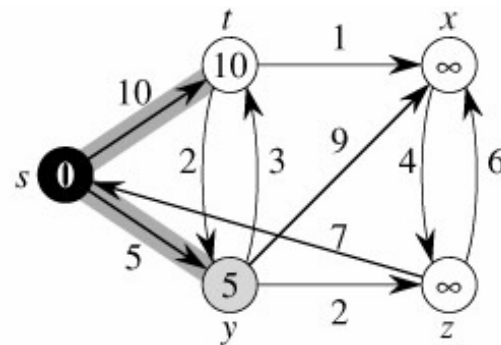
Se debe tener en cuenta si al intentar calcular los pesos para los nodos adyacentes a un nodo que está siendo visitado, uno de estos ya tiene un peso asignado, deben calcularse los demás pesos cuantas veces sean necesario, y siempre se tomara el peso mínimo calculado. (Los nodos pueden ser visitados una sola vez)



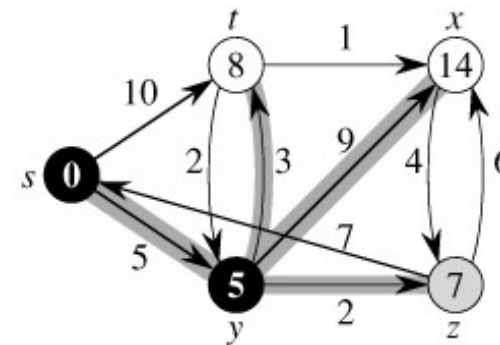
# Dijkstra: S (inicial) $\rightarrow$ x (final)



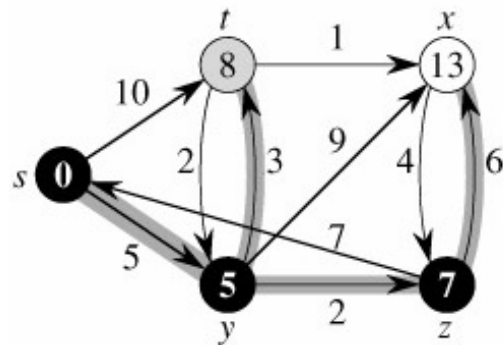
(a)



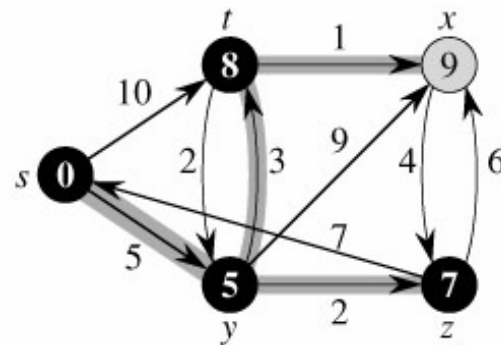
(b)



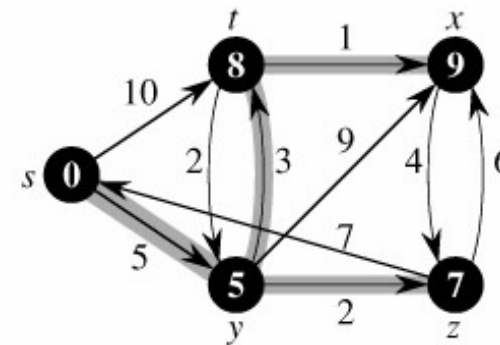
(c)



(d)

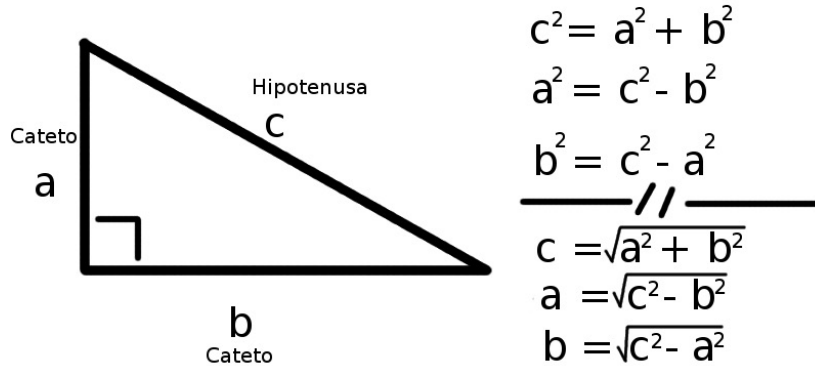


(e)

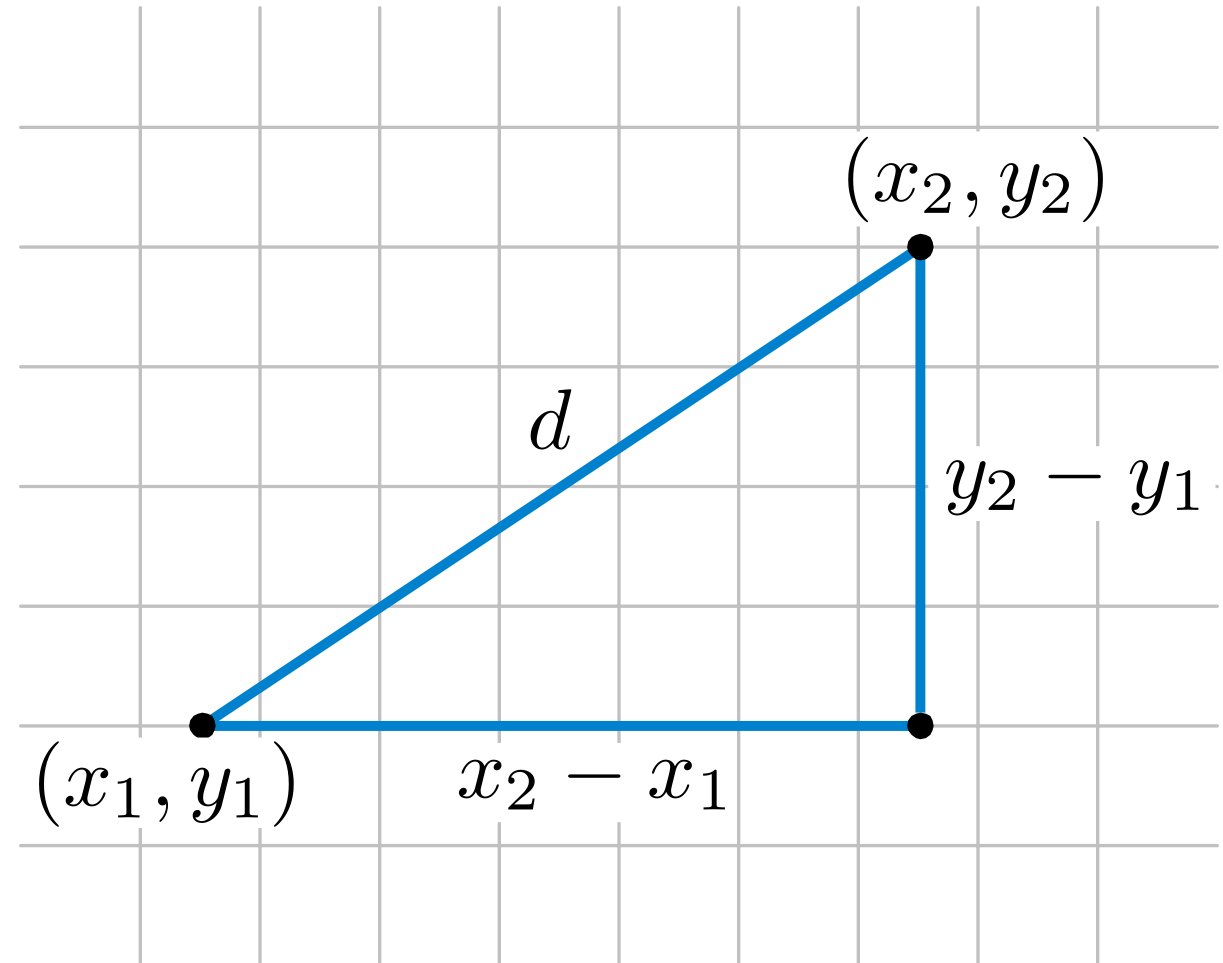


(f)

# Distancia Euclidiana




$$d_E(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



# Análisis de movilidad



 Developer

ProductsDocumentationPricingResourcesHelpLoginSign up

Routing API v7IntroductionQuick StartGuideAPI ReferenceCoverage InformationRelease NotesExamples

Resources and Parameters  
Calculate Route  
Get Route  
Get Routing Zones  
Calculate Isoline  
Calculate Matrix  
Common Parameter Types  
Response Data Types  
CalculateRouteResponseType  
GetRouteResponseType  
GetRoutingZonesResponseType  
CalculateIsolineResponseType  
CalculateMatrixResponseType  
Routing Data Types  
Base Data Types

### Calculate Route

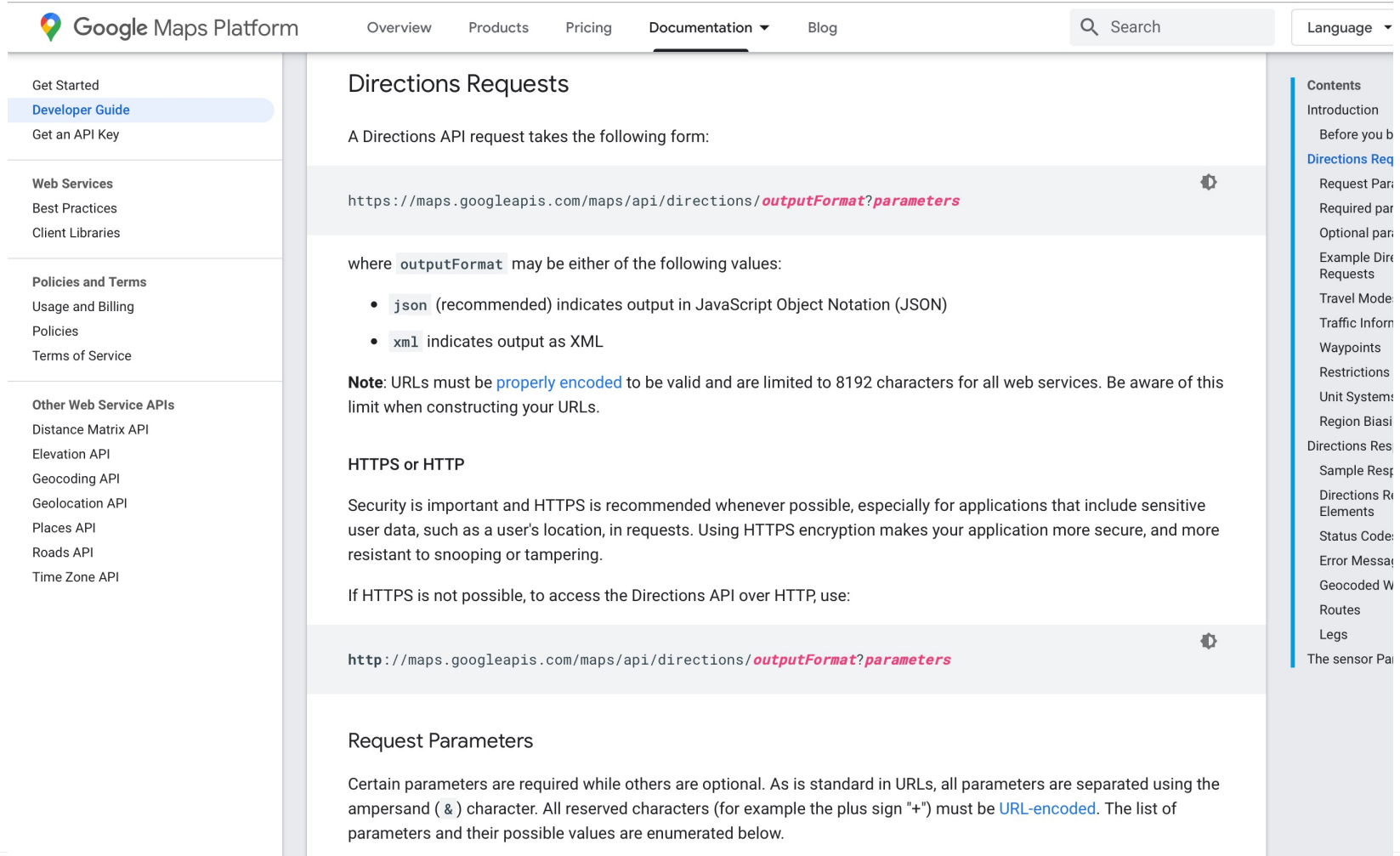
Use the `calculateroute` resource to return a route between two waypoints. The required parameters for this resource are `app_id` and `app_code`, two or more waypoints (`waypoint0` and `waypoint1`, to `waypointN`) and `mode` (specifying how to calculate the route, and for what mode of transport). For some modes `departure` or `arrival` (if applicable) is required. This includes `publicTransportTimeTable`, `publicTransport` and all modes with enabled traffic. Other parameters can be left unspecified.

```
../routing/7.2/calculateroute.{format}?<parameter>=<value>...
```

Parameter	Description
<code>apiKey</code>	A 43-byte Base64 URL-safe encoded string used for the authentication of the client application. As a logged in user, you can generate it at <a href="https://developer.here.com/projects">https://developer.here.com/projects</a> . API Keys never expire but you can invalidate your API Keys at any time. You cannot have more than two API Keys for one app at the same time. You must include an <code>apiKey</code> with every request. For further details, see <a href="#">Acquiring Credentials</a> .
<code>app_id</code>	A 20 bytes Base64 URL-safe encoded string used for the authentication of the client application. If you use the app ID/app code option, you need to include an <code>app_id</code> and



# Google



The screenshot shows the Google Maps Platform documentation page for "Directions Requests". The page has a navigation bar at the top with links for Overview, Products, Pricing, Documentation (selected), and Blog. A search bar and a language dropdown are also present. On the left, a sidebar lists various developer resources. The main content area explains the format of a Directions API request, showing a URL template with `outputFormat` and `parameters` as placeholders. It lists two possible values for `outputFormat`: `json` (recommended) and `xml`. A note specifies that URLs must be properly encoded and limited to 8192 characters. It also discusses the use of HTTPS or HTTP. At the bottom, a section titled "Request Parameters" begins to describe required and optional parameters.

Google Maps Platform

Overview Products Pricing Documentation Blog

Search

Language

Get Started

Developer Guide

Get an API Key

Web Services

Best Practices

Client Libraries

Policies and Terms

Usage and Billing

Policies

Terms of Service

Other Web Service APIs

Distance Matrix API

Elevation API

Geocoding API

Geolocation API

Places API

Roads API

Time Zone API

## Directions Requests

A Directions API request takes the following form:

```
https://maps.googleapis.com/maps/api/directions/outputFormat?parameters
```

where `outputFormat` may be either of the following values:

- `json` (recommended) indicates output in JavaScript Object Notation (JSON)
- `xml` indicates output as XML

**Note:** URLs must be properly encoded to be valid and are limited to 8192 characters for all web services. Be aware of this limit when constructing your URLs.

### HTTPS or HTTP

Security is important and HTTPS is recommended whenever possible, especially for applications that include sensitive user data, such as a user's location, in requests. Using HTTPS encryption makes your application more secure, and more resistant to snooping or tampering.

If HTTPS is not possible, to access the Directions API over HTTP, use:

```
http://maps.googleapis.com/maps/api/directions/outputFormat?parameters
```

### Request Parameters

Certain parameters are required while others are optional. As is standard in URLs, all parameters are separated using the ampersand (&) character. All reserved characters (for example the plus sign "+") must be URL-encoded. The list of parameters and their possible values are enumerated below.

Contents

- Introduction
- Before you b
- Directions Req
- Request Par
- Required par
- Optional par
- Example Dir
- Requests
- Travel Mode
- Traffic Inform
- Waypoints
- Restrictions
- Unit Systems
- Region Biasi
- Directions Res
- Sample Resp
- Directions R
- Elements
- Status Code
- Error Messa
- Geocoded W
- Routes
- Legs
- The sensor Pa

# GTFS - Transporte Público

## Objetivo

El formato común para los horarios de transporte público y la información geográfica relacionada. Los "Feeds" GTFS permiten que las empresas de transporte público publiquen sus datos y que los desarrolladores programen aplicaciones que consuman esos datos de transporte público de manera interoperable.

- [Descripción](#)
- [Referencias](#)
- [Tiempo real](#)
- [GTFS por país](#)
- [Colombia](#)

Google™ | Transit



# Índice

**Identificación de rutas y tiempo de recorrido**

**K-medias**

**K-prototipos**

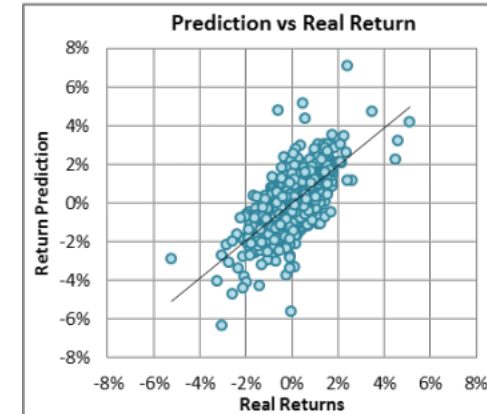
**DBSCAN**

# Machine Learning

## Modelos supervisados:

El aprendizaje supervisado utiliza como entrenamiento un set de datos que contiene una marca o etiqueta en los datos. En este tipo de aprendizaje es claro la distinción entre las variables independientes y la variable dependiente.

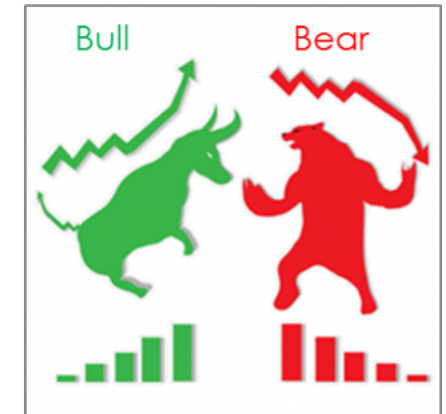
### Regression



**Regresión:**  
Estimación continua

vs

### Classification



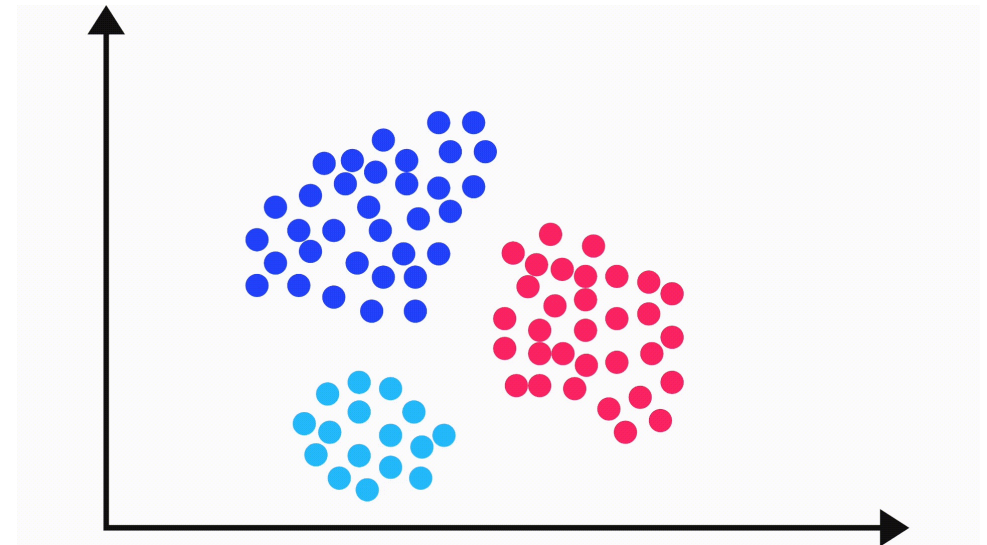
**Clasificación:**  
Estimación discreta

# No-supervisado

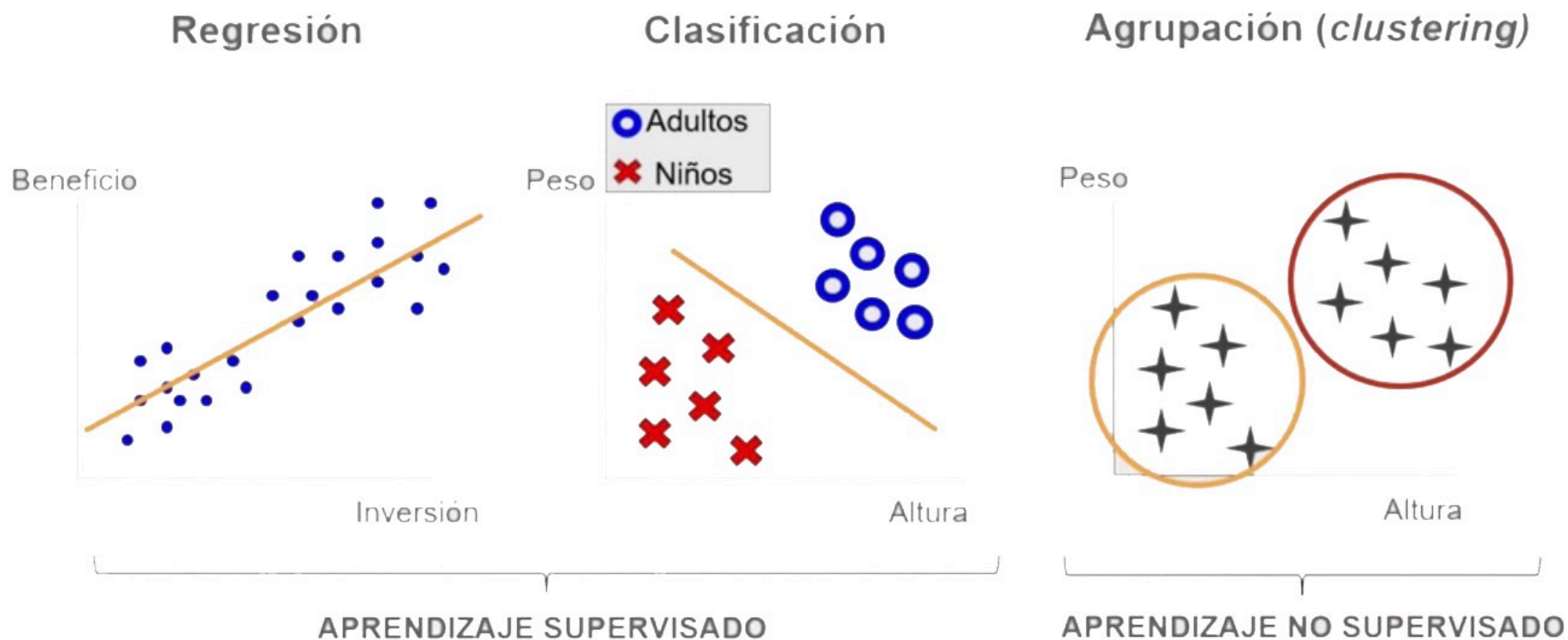
Sin conocer una etiqueta previa de los datos, se busca entender la distribución intrínseca de nuestros datos.

## Aplicaciones:

- Métodos de agrupamiento
- Reducción de dimensionalidad.
- Detección de anomalías.
- Estimación de distribución empírica.



# Machine Learning

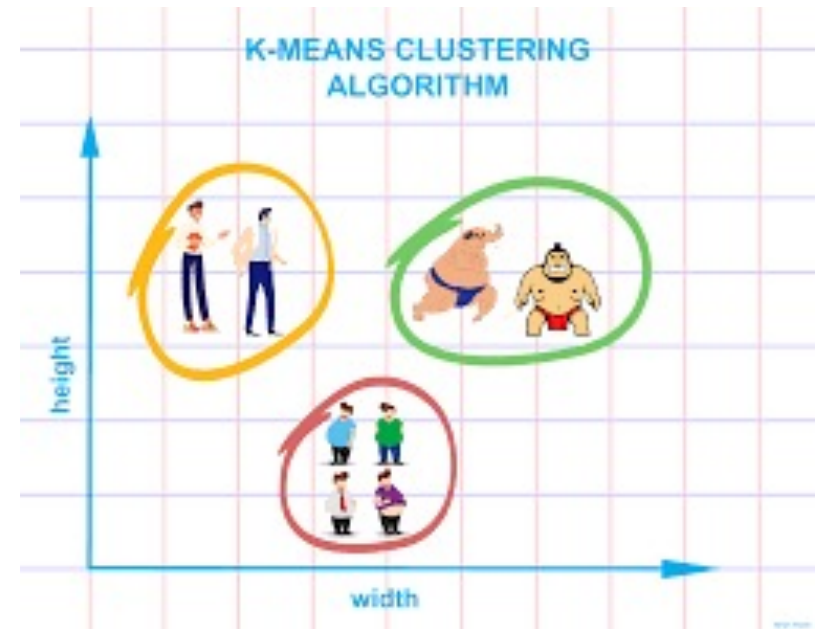


# K-medias

Los métodos de agrupamiento buscan subgrupos homogéneos, coherentes e interesantes de observaciones, para separar los datos entre tales subconjuntos.

Queremos dividir la base de datos en diferentes grupos tales que las observaciones en un mismo grupo sean similares entre sí, y observaciones en grupos distintos sean diferentes entre ellas.

**¿Cómo definimos que dos datos como similares o diferentes?**



# K-medias

¿Cómo definimos que dos datos como similares o diferentes?

Este es un método de aprendizaje no supervisado porque intenta rescatar información y estructura (distintos subgrupos) de la base de datos.

No se conoce la cantidad ni la forma que tienen estos clusters, ni tampoco a qué grupo pertenece cada observación:

¿Cómo evaluamos qué tan buena es nuestra separación de los datos?

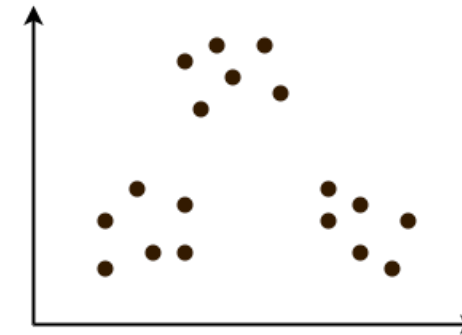


# K-medias

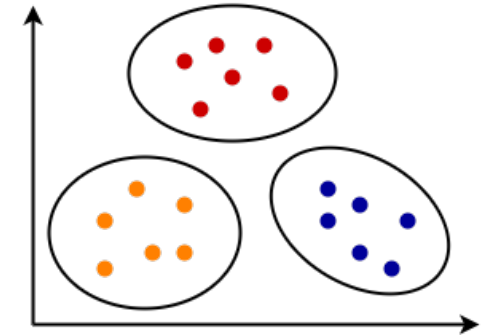
Si tenemos un conjunto de  $N$  observaciones  $\{x_1, \dots, x_N\}$  ( $x_i$  es el vector de características del registro  $i$ ), queremos encontrar conjuntos  $C_1, \dots, C_K$  que agrupen los datos, y que cumplan dos condiciones:

- $C_1 \cup C_2 \cup \dots \cup C_K = \{x_1, \dots, x_M\}$ : Cada observación pertenece al menos a uno de los  $K$  grupos.
- $C_k \cap C_{k'} = \emptyset$  para todo  $k \neq k'$ : Los grupos no se sobrelapan, i.e. ninguna observación pertenece a mas de un grupo.

Estas dos condiciones combinadas resultan en que cada una de las observaciones  $x_i$  pertenece a uno, y solo uno, de los clusters  $C_k$ .



Before K-Means

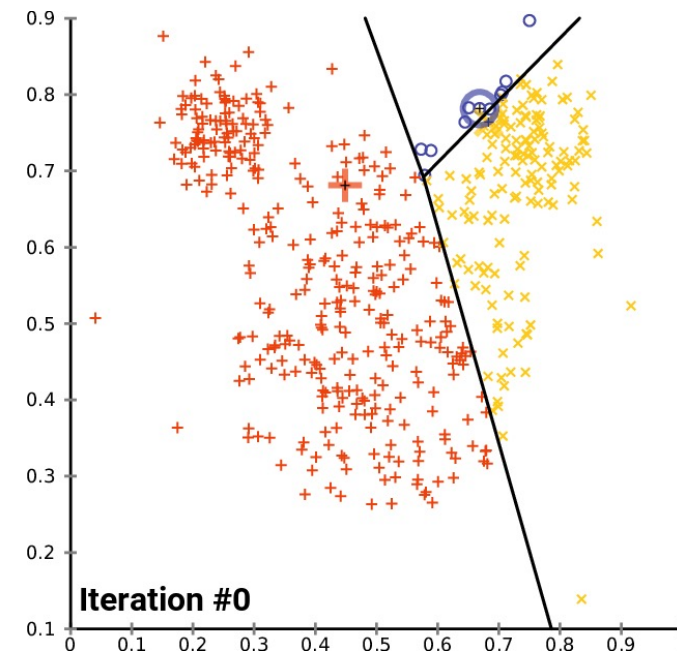


After K-Means

# K-medias

- Intuitivo y fácil de usar.
- Eficiente computacionalmente.
- Muy popular y conocido.

La idea detrás de este método de agrupamiento es que una buena manera de segmentar los datos es haciendo que la varianza dentro de cada grupo sea tan pequeña como sea posible.



# K-medias

Dado un subconjunto de los datos  $C_k$ , se calcula su media interior para la característica  $j$  como:

$$\bar{x}_1^k = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_{i,j}$$

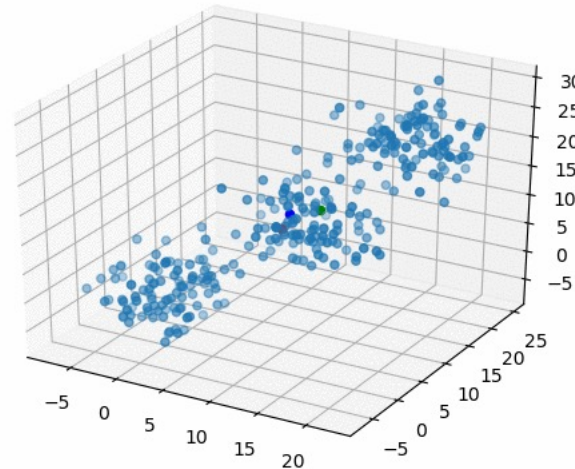
En Donde

- $|C_k|$  el número de elementos en el clúster
- $\sum_{x_i \in C_k} x_{i,j}$  la suma solo se tienen en cuenta aquellas observaciones pertenecientes al grupo.

# K-medias

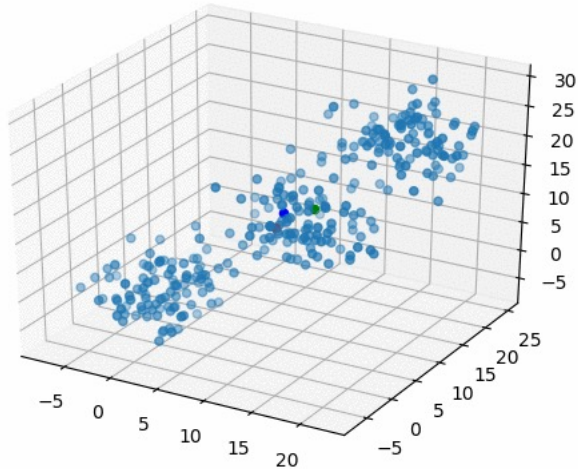
Al vector que recoge la media de cada variable al interior de un cluster se le conoce como centroide del grupo, y se denota con la letra griega  $\mu$ .

$\mu_k = (\bar{x}_1^k, \bar{x}_2^k, \dots, \bar{x}_p^k)$  es el centroide del cluster  $C_k$  con  $p$  características



# K-medias

Se calcula entonces la varianza dentro del grupo (within cluster variation)  $C_k$ :



$$Var(C_k) = \sum_{j=1}^p Var(X_j^k) = \sum_{j=1}^p \left( \frac{1}{|C_k| - 1} \sum_{x_i \in C_k} (x_{i,j} - \bar{X}_j^k)^2 \right).$$

Queremos que esta varianza sea lo más pequeña posible, en otras palabras, que los datos dentro del cluster sean lo más parecidos.

# K-medias

De esta manera, queremos encontrar subgrupos de la base de datos  $C_1, \dots, C_k$ , tales que la suma de sus respectivas varianzas sea lo más pequeña posible.

Matemáticamente,

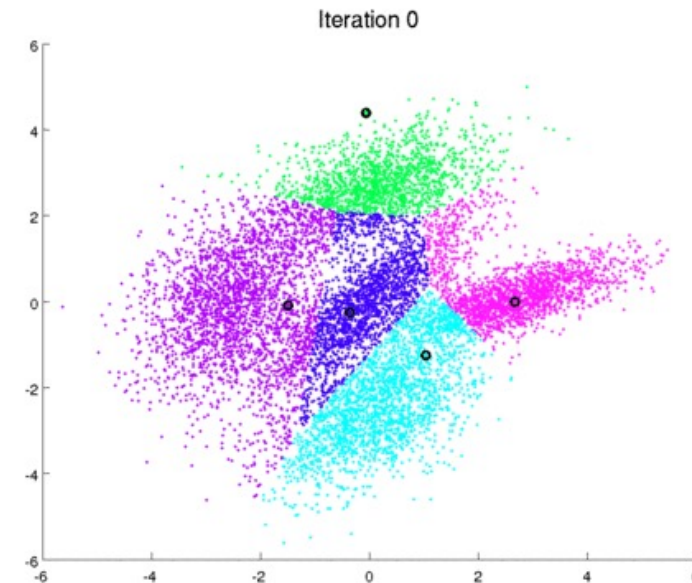
$$\min_{C_1, \dots, C_k} \sum_{k=1}^K \text{Var}(C_k) = \min_{C_1, \dots, C_k} \sum_{k=1}^K \left[ \sum_{j=1}^p \left( \frac{1}{|C_k| - 1} \sum_{x_i \in C_k} (x_{i,j} - \bar{X}_j^k)^2 \right) \right]$$

# K-medias

Se calcula entonces la varianza dentro del grupo (within cluster variation)  $C_k$ :

Dado el número  $K$  de grupos a encontrar:

1. Seleccione aleatoriamente  $K$  observaciones de su muestra, y supongamos que estos son los centroides de los grupos buscados.
2. **Repetir hasta que los grupos no varíen:**
  - a) Asignar a cada elemento el grupo del centroide mas cercano.
  - b) Actualizar los centroides con los nuevos grupos formados.

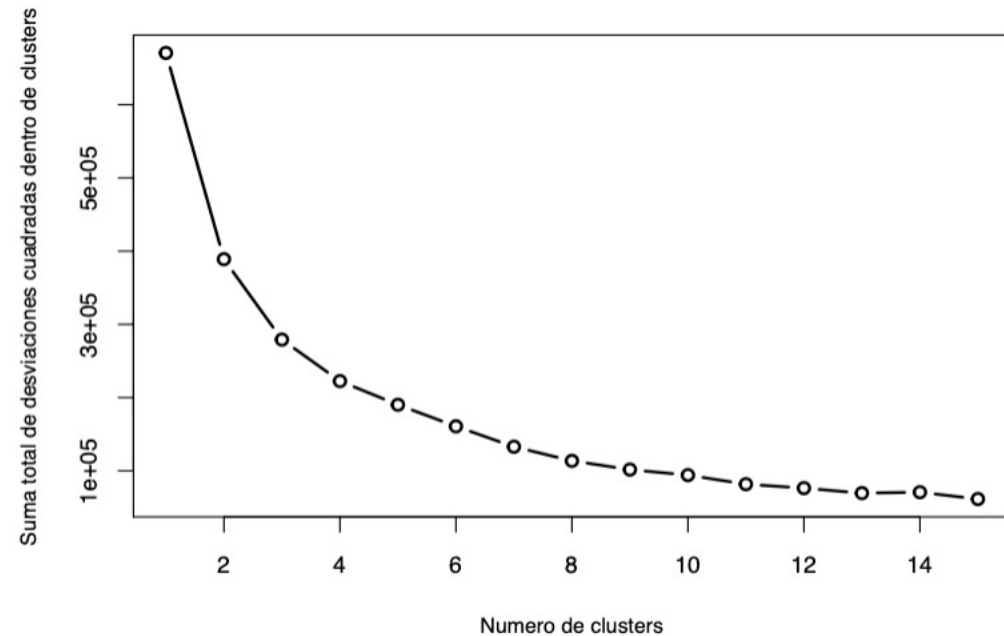


# K-medias

Se calcula entonces la varianza dentro del grupo (within cluster variation)  $C_k$ :

¿Cómo escogemos el número de grupos  $K$ ?

- Con información previa del problema.
- Regla del codo.
- Criterio de la silueta



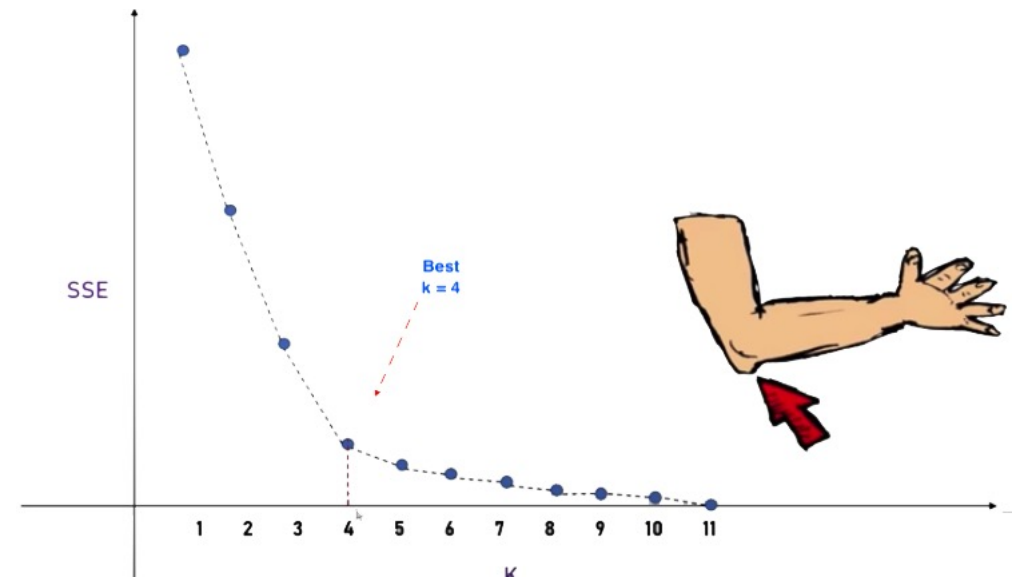


# Regla del codo

Este método utiliza como criterio la suma de los cuadrados de las desviaciones entre cada punto (individuo) y la media del cluster en el que se integra. Esto se calcula para diferentes número sde Clusters (desde 1 a N Clusters).

$$\sum_{i=1}^n (y_i - \hat{y}_l)^2$$

**Objetivo:** Identificar el número de grupos en dónde el cambio de la suma de los cuadrados de las desviaciones entre cada punto y el centroide del cluster sea más drástico.



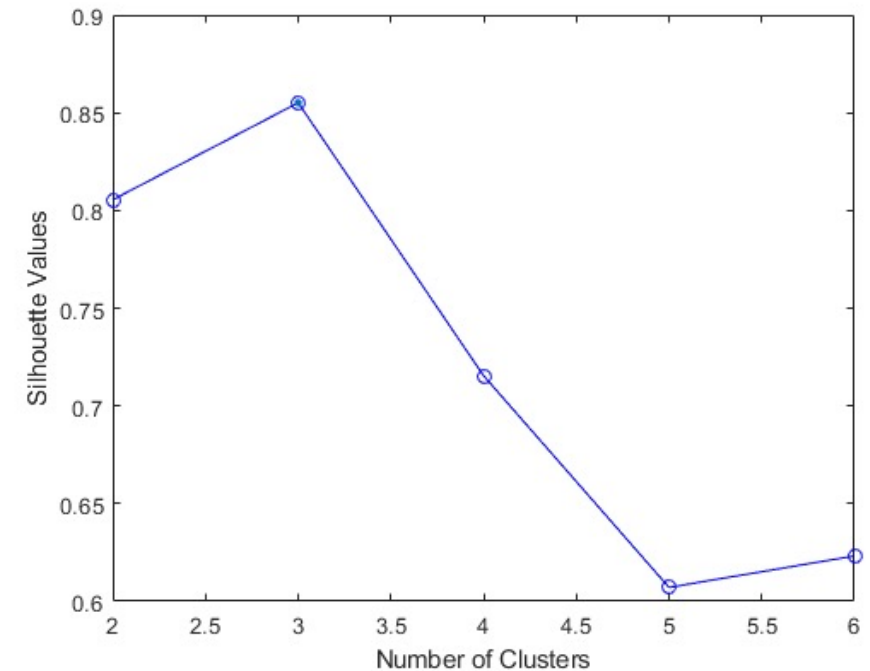
# Criterio de la silueta

El criterio de la silueta se define como la diferencia entre la **separación** y la **cohesión** sobre el máximo de la cohesión y la separación.

La intuición detrás, es que un buen agrupamiento se logra cuando se alcanza la mayor separación entre los grupos respetando la cohesión entre estos.

Este criterio toma valores de -1 y 1.

- -1 significa un mal agrupamiento
- 0 indiferente
- 1 un buen agrupamiento.



# Índice

Identificación de rutas y tiempo de recorrido

K-medias

K-prototipos

DBSCAN

# Otras alternativas

La distancia euclídeana base de K-medias no funciona para variables categóricas.

**K-Modas, K-Medianas, K-medoids, K-prototipos.**

**K-Medianas:** El centroide de cada grupo se construye con la mediana de cada variable. Limitado a datos continuos pero más robusto a datos atípicos.

**K-Modas:** En lugar de usar como centroide la media de cada variable, se utiliza la moda. Especialmente útil para variables categóricas, junto con distancias discretas.

# Otras alternativas

La distancia euclideana base de K-medias no funciona para variables categóricas.

**K-Medias:** El centroide de cada grupo se construye con la mediana de cada variable. Limitado a datos continuos pero más robusto a datos atípicos.

**K-Modas:** En lugar de usar como centroide la media de cada variable, se utiliza la moda. Especialmente útil para variables categóricas, junto con distancias discretas.

**K-Medoides:** El centroide de cada grupo es una de sus observaciones, y debe ser “central” al interior del grupo. Combina variables numéricas y categóricas. Costoso computacionalmente.

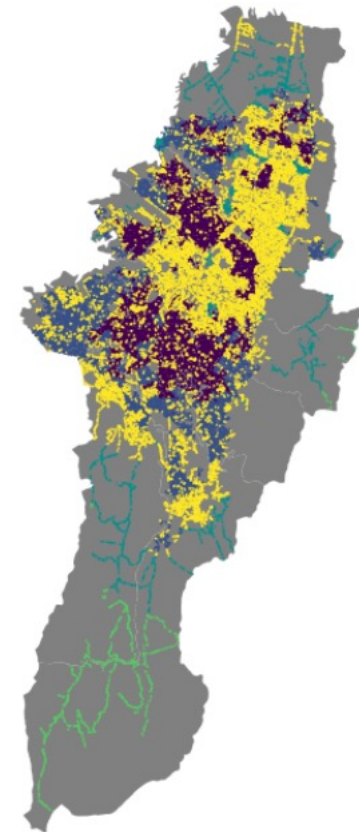
**K-Prototipos:** Combina medias de variables continuas con modas de variables categóricas, y distancias para cada tipo de variables.

# Ejemplo: K-protipos

Grupos espaciales Bogotá, 5-Prototipos

Se generaron grupos de lugares en la ciudad similares usando el estrato socioeconómico, si está en zona rural o urbana, y su distancia al CAI más cercano:

Metodología K – prototipos y escogencia de grupos por regla del codo.



# Índice

Identificación de rutas y tiempo de recorrido

K-medias

K-prototipos

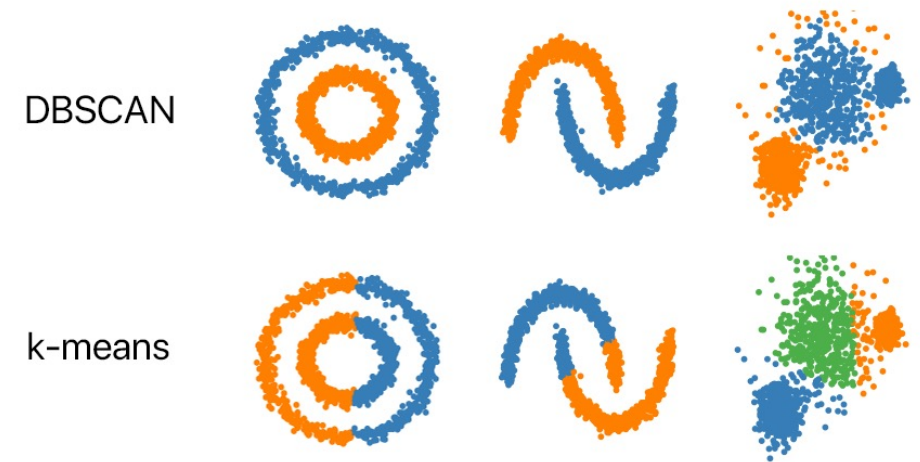
DBSCAN

# Clustering espacial basado en la densidad de aplicaciones con ruido (DBSCAN)

Se deben especificar dos parámetros:

**Radio  $\epsilon$ :** La máxima distancia (euclídeana) permitida para que un par de puntos sean considerados vecinos.

**Min Points:** Número mínimo de observaciones en cada grupo.





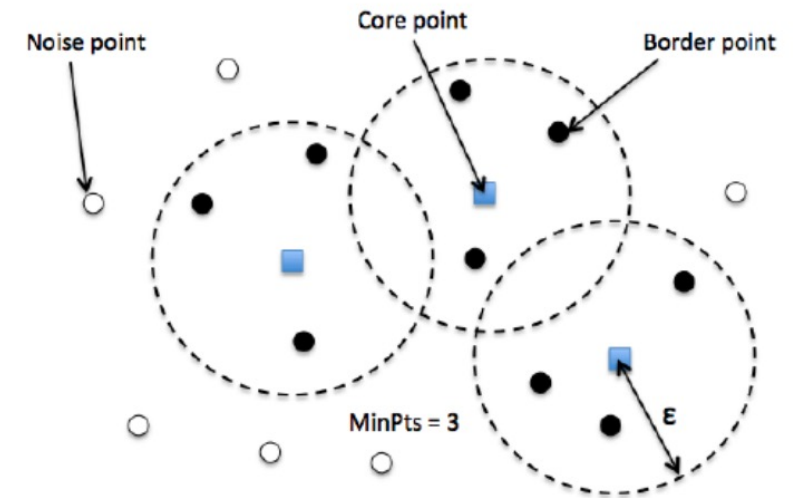
# DBSCAN

Cada observación se cataloga en uno de tres tipos:

**Punto central:** Si existen más de Min Points observaciones (incluyéndolo a él mismo) a una distancia menor que  $\epsilon$ .

**Punto frontera:** Observaciones que están a una distancia menor o igual que  $\epsilon$  de alguno de los puntos centrales identificados, y no son ellas mismas puntos centrales.

**Punto ruido:** Registros no clasificados ni como punto central ni como punto frontera. Se encuentran a una distancia mayor que  $\epsilon$  que todos los puntos centrales identificados, y se consideran observaciones atípicas.



# DBSCAN

## Ventajas:

- No se debe especificar el número de grupos.
- Flexibilidad en los tamaños y formas que toman los grupos identificados.
- Identifica simultáneamente observaciones ruido o atípicas.

## Desventajas

- No determinístico por puntos a distancia menor que  $\epsilon$  de varios puntos centrales.
- Escoger  $\epsilon$  es difícil especialmente en dimensiones altas.
- No funciona bien para grupos de densidad variable. Aunque agrupa las regiones de alta densidad y las separan de las de baja densidad, tiene muchas dificultades en el caso de los grupos de densidad similar.

# Gracias

