IIC2433 — Minería de Datos

# Apuntes: Dos algoritmos para clusterizar

Nota: ambos algoritmos para clusterizar operan sobre vectores numéricos. El input, enconces, siempre se asume que es un conjunto $X$ de vectores de una cierta dimensión $D$, y cada vez que nos referimos a la distancia $d(x_1, x_2)$ entre dos elementos de $X$, nos referimos a la distancia euclideana entre dos vectores $D$-dimensionales. Adicionalmente los algoritmos podrían recibir otros parámetros, ya lo discutiremos.

## 1. DBSCAN

El algoritmo de Density-based Spatial Clustering of Applications with Noise es un algoritmo que presenta algunas diferencias bien claras con Kmeans:

- No busca clasificar todo el espacio, algunos vectores son clasificados como ruido.
- Se basa en encontrar regiones densamente pobladas en el espacio, que no necesariamente son círculos o elipses centradas en un punto.
- En vez de trabajar con un número de clústeres dado, DBSCAN requiere dos inputs relacionados con cuando se considera densa un área en particular.

### 1.1. Definiciones

Considera un número real $\epsilon \geq 0$, que va a representar la distancia bajo la cual consideramos que dos puntos están *cerca*, y un natural $mP$ que va a representar la cantidad minima de puntos que debe tener un *espacio denso*.

La idea de DBSCAN es encontrar los clústeres que cumplan con las siguiente propiedades. Primero unas definiciones. Dos puntos centrales $x$ e $y$ están *conectados* si $d(x, y) \leq \epsilon$. Un *punto central* es un punto que está conectado con al menos $mP$ nodos. Entonces:

- Sea $C_1, \ldots, C_k$ las $k$ componentes conexas de la relación *conectados* (es decir, la componente conexa del grafo cuyos vértices son los puntos centrales y hay una arista entre dos puntos centrales si están conectados). Cada una de esas componentes conexas es un clúster.
- Para cada punto no central, hay dos posibilidades. Primero, si está a una distancia menor a $\epsilon$ de un punto central correspondiente al cluster $C_i$, entonces el punto no central se asigna también a $C_i$. De lo contrario (si todos los puntos centrales están a distancia de más de $\epsilon$), el punto es catalogado como *ruido*.

## 1.2. Cálculo

El algoritmo usualmente se implementa pasando una y otra vez por cada punto, definiendo puntos centrales y asignando puntos a clústeres a medida que crecen las componentes conexas. Sin embargo, es recomendable la ayuda de estructuras o índices específicos para poder computar todos los nodos *cercanos* a otros de forma eficiente.

# 2. GMM

Los Gaussian Mixture Models son un modelo probabilístico que se usa comúnmente para clusterizar vectores. Sea $K$ una cantidad fja de clústeres. El modelo functiona bajo las siguientes suposiciones:

1. Cada cluster $k$ distribuye de acuerdo a una gaussiana (multivariada) con parámetros $(\mu_k, \Sigma_k)$ (recuerda que en $D$ dimensiones la media es un vector y la varianza está dada en forma de matriz de covarianzas). En particular, cada cluster usa potencialmente una gaussiana distinta. *Cada entidad tiene asociada una probabilidad $\pi_i$ de pertenecer a cada uno de estos clústeres*

De esta forma, para cada $1 \leq i \leq k$, el modelo estima el trío de parámetros $(\mu_k, \sigma_k, \pi_i)$.

## 2.1. Estimación

La estimación nuevamente se hace maximizando la log-verosimilitud de cada punto (cada punto finalmente distribuye de acuerdo a una suma ponderada de gaussianas). Aunque escapa de este curso, sugerimos a los interesados en revisar la bibliografía para ver cómo encontrar parametros que maximizen (localmente) a la verosimlitud.

## 2.2. Clusterización

Para clusterizar, simplemente asignar cada $x \in X$ al cluster $C_i$ tal que $i$ maximiza $\pi_i(x)$. Lo interesante de GMM es que al estar basado en distribuciones gaussianas, las áreas determinadas por los clusters pueden tomar formas elípticas en el plano, no necesariamente son regiones esféricas como en el caso de Kmeans.