Algoritmos avanzados basados en densidad

Algoritmos basados en densidad topológica.

Los dos métodos que vamos a exponer en esta sección no necesitan una determinación inical del número de clusters.

Mean Shift

Es un algoritmo basado en ventanas deslizantes que intenta encontrar áreas densas de puntos de datos. Es un algoritmo basado en centroide que funciona actualizando a los candidatos para que los puntos centrales (centroides) sean la media de los puntos dentro de la ventana deslizante. Estas ventanas candidatas se filtran luego en una etapa de post-procesamiento para eliminar los duplicados cercanos, formando el conjunto final de puntos centrales y sus grupos correspondientes.

- Comenzamos con una ventana corredera circular centrada en un punto C (seleccionado aleatoriamente) y teniendo radio r como núcleo. El mean shift es un algoritmo de escalada implica desplazar este centroide de forma iterativa a una región de mayor densidad en cada paso hasta la convergencia.
- En cada iteración la ventana deslizante se desplaza hacia las regiones de mayor densidad, desplazando el punto central a la media de los puntos dentro de la ventana (de ahí el nombre). La densidad dentro de la ventana deslizante es proporcional al número de puntos dentro de ella. Naturalmente, al pasar a la media de los puntos en la ventana se moverá gradualmente hacia áreas de mayor densidad de puntos.
- Seguimos desplazando la ventana deslizante de acuerdo a la media hasta que no haya dirección en la que un turno pueda acomodar más puntos dentro del núcleo. Seguimos moviendo el círculo hasta que ya no aumentamos la densidad (es decir, el número de puntos en la ventana).
- Este proceso de los pasos 1 a 3 se realiza con muchas ventanas correderas hasta que todos los puntos se encuentran dentro de una ventana. Cuando varias ventanas correderas se superponen, se conserva la ventana que contiene la mayor parte de los puntos. Los puntos de datos se agrupan de acuerdo con la ventana deslizante en la que residen.

DB-SCAN

- DBSCAN comienza con un punto de datos de inicio arbitrario. El vecindario de este punto se extrae utilizando una distancia épsilon ε (todos los puntos que están dentro de la distancia ε son puntos de vecindad).
- Si hay un número suficiente de puntos (según el parámetro que establezcamos minPoints) dentro de este vecindario, entonces se inicia el proceso de agrupación en clústeres y el punto de datos actual se convierte en el primer punto en el nuevo clúster. De lo contrario, el punto se etiquetará como ruido (más adelante este punto ruidoso podría convertirse en la parte del clúster). En ambos casos ese punto se marca como "visitado".
- Para este primer punto en el nuevo cluster, los puntos dentro de su vecindad de la distancia ε también forman parte del mismo cluster. Este procedimiento de hacer que todos los puntos del vecindario ε pertenezcan al mismo clúster se repite para todos los nuevos puntos que se acaban de agregar al grupo de clústeres.
- Este proceso de los pasos 2 y 3 se repite hasta que se determinen todos los puntos en el cluster, es decir, todos los puntos dentro de la ε de los puntos del vecindario han sido visitados.
- Una vez que hayamos terminado con el clúster actual, se recupera y procesa un nuevo punto no visitado, lo que conduce al descubrimiento de otro clúster o ruido. Este proceso se repite hasta que todos los puntos se marcan como visitados. Puesto que al final de todo esto se han visitado todos los puntos, cada punto bien se han marcado como pertenecientes a un cluster o siendo ruido.

Como puede apreciarse, tiene la ventaja de que no obliga a todos los puntos a pertenecer a un cluster y puede permitir puntos anómalos o ruidos no pertenecientes a ninguno.

Algoritmos basados en densidad probabilística EM-GMM (Expectation Maximization using Gaussian Mixture Models.

Asumimos que los puntos de datos están normalmente distribuidos. De esta manera, tenemos dos parámetros para describir la forma de los conglomerados: la media y la desviación típica. Tomando un ejemplo en dos dimensiones, esto significa que los clústeres pueden tomar cualquier tipo de forma elíptica (ya que tenemos desviación estándar en las direcciones x e y). Por lo tanto, cada distribución gaussiana se asigna a un único clúster.

Para encontrar los parámetros de la distribución normal para cada cluster (media y desviación típica), utilizaremos un algoritmo de optimización llamado Expectation Maximization (EM).

Comenzamos seleccionando el número de clústeres (como K-medias) e inicializando aleatoriamente los parámetros de distribución Gaussianos para cada cluster (podemos intentar aproximar los parámetros iniciales a partir de una mirada rápida a los datos)

Dadas estas distribuciones Gaussianas para cada clúster, calculamos la probabilidad de que cada punto de datos pertenezca a un clúster determinado. Cuanto más se acerque un punto a la media, más probable es que pertenezca a ese grupo. Esto debería tener sentido intuitivo ya que con una distribución normal asumimos que la mayoría de los datos se encuentran más cerca del centro del clúster.

Basándose en estas probabilidades, calculamos un nuevo conjunto de parámetros para las distribuciones, de modo que maximizamos las probabilidades de los puntos de datos dentro de los clústeres. Calculamos estos nuevos parámetros utilizando una suma ponderada de las posiciones de punto de datos, donde los pesos son las probabilidades del punto de datos que pertenece a ese clúster en particular. Así, naturalmente, la media de la distribución se desplaza más cerca de aquellos puntos con más probabilidad. La desviación típica también cambia con el fin de maximizar la suma ponderada por las probabilidades.

Los pasos 2 y 3 se repiten de forma iterativa hasta la convergencia, donde las distribuciones no cambian tras cada iteración.