El algoritmo K-means agrupa los datos tratando de separar las muestras en n grupos de varianza igual, minimizando un criterio conocido como *inercia* o suma de cuadrados dentro del grupo (ver más abajo). Este algoritmo requiere que se especifique el número de clústeres. Se adapta bien a una gran cantidad de muestras y se ha utilizado en una amplia gama de áreas de aplicación en muchos campos diferentes.

El algoritmo de k-medias divide un conjunto de N muestras X dentro K clústeres disjuntos C, cada uno descrito por la media μj de las muestras en el conglomerado. Los medios se denominan comúnmente "centroides" del grupo; tenga en cuenta que no son, en general, puntos de X, aunque viven en el mismo espacio.

El algoritmo de K-medias tiene como objetivo elegir centroides que minimicen la inercia , o el criterio de suma de cuadrados dentro del clúster :

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2)$$

La inercia se puede reconocer como una medida de cuán coherentes son los clústeres internamente.

En términos básicos, el algoritmo tiene tres pasos. El primer paso elige los centroides iniciales, siendo el método más básico elegir k muestras del conjunto de datos X. Después de la inicialización, K-means consiste en recorrer los otros dos pasos. El primer paso asigna cada muestra a su centroide más cercano. El segundo paso crea nuevos centroides tomando el valor medio de todas las muestras asignadas a cada centroide anterior. Se calcula la diferencia entre el antiguo y el nuevo centroide y el algoritmo repite estos dos últimos pasos hasta que este valor es menor que un umbral. En otras palabras, se repite hasta que los centroides no se mueven significativamente.

Parámetros

n_clusters *int, predeterminado = 8*
El número de conglomerados que se formarán, así como el número de centroides que se generarán.

init *{'k-means ++', 'random'}, invocable o similar a una matriz de forma (n_clusters, n_features), predeterminado = 'k-means ++'*
Método de inicialización:

'k-means ++': selecciona los centros de clústeres iniciales para el clustering de k-mean de una manera inteligente para acelerar la convergencia. Consulte la sección Notas en k_init para obtener más detalles.

'aleatorio': elija n_clusters observaciones (filas) al azar de los datos para los centroides iniciales.

Si se pasa una matriz, debería tener la forma (n_clusters, n_features) y proporcionar los centros iniciales.

Si se pasa un invocable, debe tomar los argumentos X, n_clusters y un estado aleatorio y devolver una inicialización.

n_init *int, predeterminado = 10*
Número de veces que se ejecutará el algoritmo de k-medias con diferentes semillas de centroide. Los resultados finales serán la mejor salida de n_init corridas consecutivas en términos de inercia.

max_iter *int, predeterminado = 300*
Número máximo de iteraciones del algoritmo k-means para una sola ejecución.

random_state *int, instancia de RandomState o None, predeterminado = None*
Determina la generación de números aleatorios para la inicialización del centroide. Utilice un int para hacer que la aleatoriedad sea determinista.

Método del codo

El método del codo ayuda a elegir el valor óptimo de 'k' (número de grupos) ajustando el modelo con un rango de valores de 'k'.

Este método utiliza los valores de la inercia obtenidos tras aplicar el K-means a diferente número de Clusters (desde 1 a N Clusters), siendo la inercia la suma de las distancias al cuadrado de cada objeto del Cluster a su centroide:

$$Inercia = \sum_{i=0}^{N} \|x_i - \mu\|^2$$

Una vez obtenidos los valores de la inercia tras aplicar el K-means de 1 a N Clusters, representamos en una gráfica lineal la inercia respecto del número de Clusters. En esta gráfica se debería de apreciar un cambio brusco en la evolución de la inercia, teniendo la línea representada una forma similar a la de un brazo y su codo. El punto en el que se observa ese cambio brusco en la inercia nos dirá el número óptimo de Clusters a seleccionar para ese dataset; o, dicho de otra manera: el punto que representaría al codo del brazo será el número óptimo de Clusters para ese data set.

Métodos

fit(*X* , *y = Ninguno* , *sample_weight = Ninguno*)

Parámetros

X *{matriz dispersa en forma de matriz} de forma (n_samples, n_features)*
Capacitación de instancias para agrupar. Debe tenerse en cuenta que los datos se convertirán al orden C, lo que provocará una copia de la memoria si los datos dados no son C-contiguos. Si se pasa una matriz dispersa, se hará una copia si no está en formato CSR.

y *Ignorado*
No utilizado, presente aquí para la coherencia de API por convención.

sample_weight tipo *matriz de forma (n_samples,), predeterminado = Ninguno*
Los pesos de cada observación en X. Si Ninguno, a todas las observaciones se les asigna el mismo peso.

.

Devoluciones

uno mismo
Estimador ajustado.

fit_predict(*X* , *y* = *Ninguno* , *sample_weight* = *Ninguno*)

Parámetros

X ***{matriz dispersa en forma de matriz} de forma (n_samples, n_features)***
Nuevos datos para transformar.

Y ***Ignorado***
No utilizado, presente aquí para la coherencia de la API por convención.

sample_weight similar a una ***matriz de forma (n_samples,), predeterminado = Ninguno***
Los pesos para cada observación en X. Si Ninguno, a todas las observaciones se les asigna el mismo peso.

Devoluciones

etiquetas ***ndarray de forma (n_samples,)***
Índice del conglomerado al que pertenece cada muestra.

Más información en scikit-learn