Búsqueda de grupos de datos – Clustering con k-means

¿Alguna vez has pasado tiempo observando a una multitud? Si es así, es probable que hayas visto algunas personalidades recurrentes. Tal vez un cierto tipo de persona, identificada por un traje recién planchado y un maletín, viene a tipificar al ejecutivo de negocios "gordo". A un veinteañero que usa jeans ajustados, una camisa de franela y anteojos de sol, se le puede apodar como un "hipster", mientras que a una mujer que hace descender a los niños de una minivan se le puede etiquetar como una "mamá futbolera".

Por supuesto, este tipo de estereotipos es peligroso de aplicar a individuos, ya que no hay dos personas exactamente iguales. Sin embargo, entendidos como una forma de describir un colectivo, las etiquetas capturan algún aspecto subyacente de similitud compartido entre los individuos dentro del grupo.

Como pronto aprenderás, el acto de agrupar, o detectar patrones en los datos, no es muy diferente de detectar patrones en grupos de personas. En este documento se describen:

- Las formas en que las tareas de agrupamiento (clustering) difieren de las tareas de clasificación que examinamos anteriormente.
- Cómo el agrupamiento define un grupo y cómo se identifican dichos grupos mediante k-means, un algoritmo de agrupamiento clásico y fácil de entender.
- Los pasos necesarios para aplicar el agrupamiento a una tarea del mundo real de identificación de segmentos de marketing entre usuarios adolescentes de redes sociales.

Antes de pasar a la acción, comenzaremos por analizar en profundidad exactamente qué implica el agrupamiento.

Entender el agrupamiento

El agrupamiento o **clustering** es una tarea de aprendizaje automático no supervisada que divide automáticamente los datos en agrupamientos o grupos de elementos similares. Lo hace sin que se le haya dicho de antemano cómo deben verse los grupos. Como no le decimos a la máquina específicamente lo que estamos buscando, el agrupamiento se utiliza para el descubrimiento de conocimiento en lugar de la predicción. Proporciona una perspectiva de los agrupamientos naturales que se encuentran dentro de los datos.

Sin un conocimiento avanzado de lo que compone un agrupamiento, ¿cómo puede una computadora saber dónde termina un grupo y comienza otro? La respuesta es sencilla: la agrupación se guía por el principio de que los elementos dentro de un grupo deben ser muy similares entre sí, pero muy diferentes de los que están fuera. La definición de similitud puede variar según la aplicación, pero la idea básica es siempre la misma: agrupar los datos de manera que los elementos relacionados se coloquen juntos.

Los grupos resultantes se pueden utilizar para la acción. Por ejemplo, puedes encontrar métodos de agrupación en aplicaciones como:

- Segmentar a los clientes en grupos con características demográficas o patrones de compra similares para campañas de marketing dirigidas.
- Detectar comportamientos anómalos, como intrusiones no autorizadas en la red, mediante la identificación de patrones de uso que caen fuera de los grupos conocidos.
- Simplificar conjuntos de datos extremadamente "amplios" (aquellos con una gran cantidad de características) mediante la creación de una pequeña cantidad de categorías para describir filas con valores relativamente homogéneos de las características.

En general, la agrupación es útil siempre que se puedan ejemplificar datos diversos y variados mediante una cantidad mucho menor de grupos. Produce estructuras de datos significativas y procesables, que reducen la complejidad y brindan información sobre patrones de relaciones.

La agrupación en clústeres como tarea de aprendizaje automático

La agrupación en clústeres es algo diferente de las tareas de clasificación, predicción numérica y detección de patrones que hemos examinado hasta ahora. En cada una de estas tareas, el objetivo era crear un modelo que relacionara las características con un resultado, o relacionar algunas características con otras características. Cada una de estas tareas describe patrones existentes dentro de los datos. En cambio, el objetivo de la agrupación en clústeres es crear datos nuevos.

En la agrupación en clústeres, a los ejemplos no etiquetados se les asigna una nueva etiqueta de agrupación, que se ha inferido completamente a partir de las relaciones dentro de los datos. Por este motivo, a veces verás que una tarea de agrupación en clústeres se denomina **clasificación no supervisada** porque, en cierto sentido, clasifica ejemplos no etiquetados.

El problema es que las etiquetas de clase obtenidas de un clasificador no supervisado no tienen un significado intrínseco. La agrupación en clústeres te indicará qué grupos de

ejemplos están estrechamente relacionados (por ejemplo, podría devolver los grupos A, B y C), pero depende de ti aplicar una etiqueta procesable y significativa, y contar la historia de lo que hace que una "A" sea diferente de una "B". Para ver cómo esto afecta la tarea de agrupamiento, consideremos un ejemplo hipotético simple.

Supongamos que estás organizando una conferencia sobre el tema de ciencia de datos. Para facilitar la creación de redes y la colaboración profesional, planeas sentar a las personas en una de tres mesas según sus especialidades de investigación. Desafortunadamente, después de enviar las invitaciones a la conferencia, te das cuenta de que olvidaste incluir una encuesta para preguntar en qué disciplina preferiría sentarse el asistente.

En un golpe de brillantez, te das cuenta de que podrías inferir la especialidad de investigación de cada académico examinando su historial de publicaciones. Para ello, comienzas a recopilar datos sobre la cantidad de artículos que cada asistente ha publicado en revistas relacionadas con la computación y la cantidad de artículos publicados en revistas relacionadas con las matemáticas o las estadísticas. Con los datos recopilados para los académicos, creas un diagrama de dispersión:

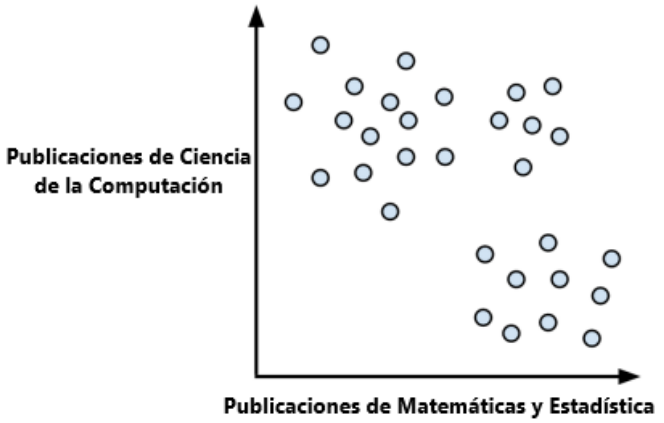

Figura 1: Visualización de académicos según sus datos de publicación en matemáticas y ciencias de la computación.

Como se esperaba, parece haber un patrón. Podríamos suponer que la esquina superior izquierda, que representa a las personas con muchas publicaciones en computación pero pocos artículos sobre matemáticas, es un grupo de científicos computacionales.

Siguiendo esta lógica, la esquina inferior derecha podría ser un grupo de matemáticos o estadísticos. De manera similar, la esquina superior derecha, aquellos con experiencia tanto en matemáticas como en computación, pueden ser expertos en aprendizaje automático.

La aplicación de estas etiquetas da como resultado la siguiente visualización:

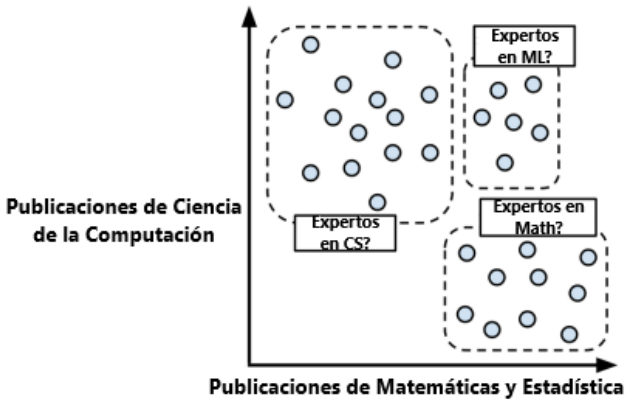

Figura 2: Los grupos se pueden identificar en función de presunciones sobre los académicos de cada grupo.

Nuestras agrupaciones se formaron visualmente; simplemente identificamos los grupos como puntos de datos agrupados de manera cercana.
Sin embargo, a pesar de las agrupaciones aparentemente obvias, sin preguntar personalmente a cada académico sobre su especialidad académica, no tenemos forma de saber si los grupos son realmente homogéneos. Las etiquetas son juicios cualitativos y presuntivos sobre los tipos de personas en cada grupo, basados en un conjunto limitado de datos cuantitativos.

En lugar de definir los límites de los grupos de manera subjetiva, sería bueno utilizar el aprendizaje automático para definirlos de manera objetiva. Dadas las divisiones paralelas a los ejes en la figura anterior, nuestro problema parece una aplicación obvia para los árboles de decisión, como se describe en el tema, Divide y vencerás: Clasificación mediante árboles de decisión y reglas. Esto nos proporcionaría una regla clara como “si un académico tiene pocas publicaciones de matemáticas, entonces es un experto en computación”.

Desafortunadamente, hay un problema con este plan. Sin datos sobre el valor de clase real para cada punto, un algoritmo de aprendizaje supervisado no tendría la capacidad de aprender dicho patrón, ya que no tendría forma de saber qué divisiones darían como resultado grupos homogéneos.

A diferencia del aprendizaje supervisado, los algoritmos de agrupamiento utilizan un proceso muy similar al que hicimos al inspeccionar visualmente el diagrama de dispersión. Al utilizar una medida de cuán estrechamente están relacionados los ejemplos, se pueden identificar grupos homogéneos. En la siguiente sección, comenzaremos a ver cómo se implementan los algoritmos de agrupamiento.

Este ejemplo destaca una aplicación interesante del agrupamiento. Si comienzas con datos sin etiquetar, puedes utilizar la agrupación para crear etiquetas de clase. A partir de ahí, puedes aplicar un aprendizaje supervisado, como árboles de decisión, para encontrar los

predictores más importantes de estas clases. Este es un ejemplo de aprendizaje semi-supervisado.

Agrupamientos de algoritmos de clustering

Así como existen muchos enfoques para construir un modelo predictivo, existen múltiples enfoques para realizar la tarea descriptiva de agrupación. Muchos de estos métodos se enumeran en el siguiente sitio, la vista de tareas de CRAN para agrupación: https://cran.r-project.org/view=Cluster. Aquí, encontrarás numerosos paquetes R utilizados para descubrir agrupaciones naturales en datos. Los diversos algoritmos se distinguen principalmente por dos características:

- La **métrica de similitud**, que proporciona la medida cuantitativa de cuán estrechamente están relacionados dos ejemplos.
- La **función de aglomeración**, que rige el proceso de asignación de ejemplos a grupos en función de su similitud (*similaridad*).

Aunque puede haber diferencias sutiles entre los enfoques, por supuesto, se pueden agrupar de varias maneras. Existen múltiples tipologías de este tipo, pero un framework simple de tres partes ayuda a comprender las principales distinciones. Con este enfoque, los tres grupos principales de algoritmos de agrupamiento, enumerados desde el más simple hasta el más sofisticado, son los siguientes:

- **Métodos jerárquicos**, que crean una jerarquía de estilo de árbol genealógico que ubica los ejemplos más similares más cerca en la estructura del gráfico.

- **Métodos basados en particiones**, que tratan los ejemplos como puntos en un espacio multidimensional e intentan encontrar límites en este espacio que conduzcan a grupos relativamente homogéneos.

- **Métodos basados en modelos o densidad**, que se basan en principios estadísticos o en la densidad de puntos para descubrir límites difusos entre los grupos; en algunos casos, los ejemplos pueden asignarse parcialmente a varios grupos, o incluso a ningún grupo en absoluto.

A pesar de que el **agrupamiento jerárquico** es el más simple de los métodos, no deja de tener un par de ventajas interesantes. En primer lugar, da como resultado una visualización de gráfico jerárquico llamada **dendrograma**, que representa las asociaciones entre ejemplos de modo que los ejemplos más similares se ubiquen más cerca en la jerarquía.

Esta puede ser una herramienta útil para entender qué ejemplos y subconjuntos de ejemplos están agrupados de forma más estrecha. En segundo lugar, la agrupación jerárquica no requiere una expectativa predefinida de cuántos clústeres existen en el conjunto de datos. En cambio, implementa un proceso en el que, en un extremo, cada ejemplo se incluye en un único clúster gigante con todos los demás ejemplos; en el otro extremo, cada ejemplo se encuentra en un pequeño clúster que solo lo contiene a él mismo; y en el medio, los ejemplos pueden incluirse en otros clústeres de distintos tamaños.

La figura 3 ilustra un dendrograma hipotético para un conjunto de datos simple que contiene ocho ejemplos, etiquetados de la A a la H. Observa que los ejemplos más estrechamente relacionados (representados por proximidad en el eje *x*) están vinculados más estrechamente como hermanos en el diagrama. Por ejemplo, los ejemplos D y E son los más similares y, por lo tanto, son los primeros en agruparse. Sin embargo, los ocho ejemplos finalmente se vinculan a un gran clúster, o pueden incluirse en cualquier número de clústeres intermedios. Al cortar el dendrograma horizontalmente en diferentes posiciones se crean cantidades variables de grupos, como se muestra para tres y cinco grupos:

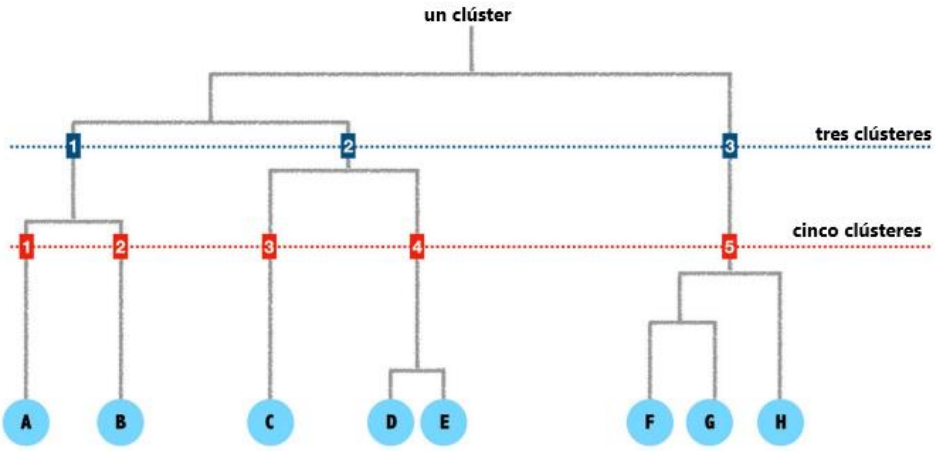

Figura 3: La agrupación jerárquica produce un dendrograma que representa las agrupaciones naturales para la cantidad deseada de grupos.

El dendrograma para la agrupación jerárquica se puede desarrollar utilizando enfoques de "abajo hacia arriba (bottom-up)" o "arriba hacia abajo (top-down)". El primero se denomina **agrupación aglomerativa** y comienza con cada ejemplo en su propio grupo, luego conecta los ejemplos más similares primero hasta que todos los ejemplos estén conectados en un solo grupo. El último se denomina **agrupación divisiva** y comienza con un solo grupo grande y termina con todos los ejemplos en sus propios grupos individuales.

Al conectar ejemplos a grupos de ejemplos, se pueden utilizar diferentes métricas, como la similitud del ejemplo con el miembro más similar, menos similar o promedio del grupo. Una métrica más compleja conocida como el método de Ward no utiliza la similitud entre

ejemplos, sino que considera una medida de homogeneidad de los grupos para construir los vínculos. En cualquier caso, el resultado es una jerarquía que pretende agrupar los ejemplos más similares en cualquier número de subgrupos.

La flexibilidad de la técnica de agrupamiento jerárquico tiene un costo, que es la complejidad computacional debido a la necesidad de calcular la similitud entre cada ejemplo y todos los demás.
A medida que aumenta el número de ejemplos (N), aumenta el número de cálculos como $N*N = N^2$, al igual que la memoria necesaria para la matriz de similitud que almacena el resultado. Por este motivo, el agrupamiento jerárquico se utiliza solo en conjuntos de datos muy pequeños y no se demuestra en este documento. Sin embargo, la función hclust() incluida en el paquete stats de R proporciona una implementación sencilla, que se instala con R de forma predeterminada.

Las implementaciones inteligentes del agrupamiento divisivo tienen el potencial de ser ligeramente más eficientes computacionalmente que el agrupamiento aglomerativo, ya que el algoritmo puede detenerse antes de tiempo si no es necesario crear un mayor número de agrupamientos. Dicho esto, tanto la agrupación aglomerativa como la divisiva son ejemplos de algoritmos "codiciosos (*greedy*)" según se define en el tema, Divide y vencerás: Clasificación mediante árboles de decisión y reglas, porque utilizan los datos por orden de llegada y, por lo tanto, no se garantiza que produzcan el conjunto general óptimo de agrupaciones para un conjunto de datos determinado.

Los métodos de **agrupamiento basados en particiones** tienen una clara ventaja de eficiencia sobre el agrupamiento jerárquico, ya que aplican métodos heurísticos para dividir los datos en grupos sin la necesidad de evaluar la similitud entre cada par de ejemplos. Exploraremos en mayor detalle un método basado en particiones ampliamente utilizado en breve, pero por ahora basta con entender que este método se ocupa de encontrar límites entre grupos en lugar de conectar ejemplos entre sí, un enfoque que requiere muchas menos comparaciones entre ejemplos.

Esta heurística puede ser bastante eficiente desde el punto de vista computacional, pero una salvedad es que es algo rígida o incluso arbitraria cuando se trata de asignaciones de grupos. Por ejemplo, si se solicitan cinco grupos, dividirá los ejemplos en los cinco grupos; si algunos ejemplos caen en el límite entre dos grupos, se ubicarán de manera algo arbitraria pero firme en un grupo u otro. De manera similar, si cuatro o seis grupos podrían haber dividido mejor los datos, esto no sería tan evidente como lo sería con el dendrograma del agrupamiento jerárquico.

Los métodos de **agrupamiento basados en modelos y en densidades** más sofisticados abordan algunos de estos problemas de inflexibilidad al estimar la probabilidad de que un ejemplo pertenezca a cada grupo, en lugar de simplemente asignarlo a un solo grupo. Algunos

de ellos pueden permitir que los límites de los grupos sigan los patrones naturales identificados en los datos en lugar de forzar una división estricta entre los grupos. Los enfoques basados en modelos a menudo suponen una distribución estadística de la que se cree que se han extraído los ejemplos.

Uno de estos enfoques, conocido como **modelado de mezclas**, intenta desenredar conjuntos de datos compuestos por ejemplos extraídos de una mezcla de distribuciones estadísticas, típicamente gaussianas (la curva de campana normal). Por ejemplo, imagina que tienes un conjunto de datos compuesto por datos de voz de una mezcla de registros vocales masculinos y femeninos, como se muestra en la figura 4 (ten en cuenta que las distribuciones son hipotéticas y no se basan en datos del mundo real). Aunque existe cierta superposición entre los dos, el hombre promedio tiende a tener un registro más bajo que la mujer promedio.

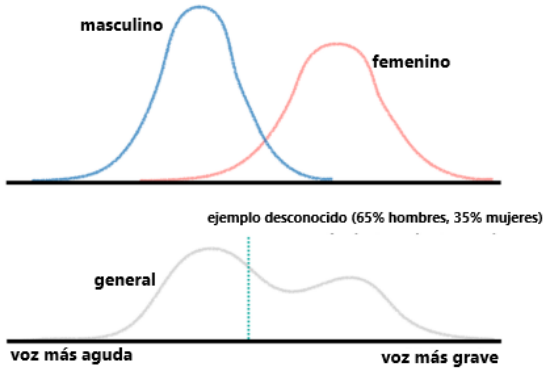

Figura 4: El modelado de mezclas asigna a cada ejemplo una probabilidad de pertenecer a una de las distribuciones subyacentes.

Dada la distribución general sin etiquetar (la parte inferior de la figura), un modelo de mezcla sería capaz de asignar una probabilidad de que cualquier ejemplo dado pertenezca al grupo de hombres o al grupo de mujeres, ¡increíblemente, sin haber sido entrenado por separado con voces masculinas o femeninas en la parte superior de la figura! Esto es posible al descubrir los parámetros estadísticos como la media y la desviación estándar que tienen más probabilidades de haber generado la distribución general observada, bajo el supuesto de que estuvieran involucradas una cantidad específica de distribuciones distintas, en este caso, dos distribuciones gaussianas.

Como método no supervisado, el modelo de mezcla no tendría forma de saber que la distribución de la izquierda es de hombres y la de la derecha de mujeres, pero esto sería fácilmente evidente para un observador humano que comparara los registros con una alta probabilidad de que los hombres estén en el grupo izquierdo frente al derecho. La desventaja de esta técnica es que no solo requiere el conocimiento de cuántas distribuciones están involucradas, sino que también requiere una suposición de los tipos de distribuciones. Esto puede ser demasiado rígido para muchas tareas de agrupamiento del mundo real.

Otra técnica de agrupamiento potente, llamada DBSCAN, recibe su nombre del enfoque de "agrupamiento espacial basado en la densidad de aplicaciones con ruido" que utiliza para identificar agrupamientos naturales en los datos. Esta galardonada técnica es increíblemente flexible y se desempeña bien con muchos de los desafíos del agrupamiento, como adaptarse al número natural de agrupamientos del conjunto de datos, ser flexible con respecto a los límites entre agrupamientos y no asumir una distribución estadística particular para los datos.

Si bien los detalles de implementación están fuera del alcance de este curso, el algoritmo DBSCAN se puede entender intuitivamente como un proceso de creación de vecindarios de ejemplos que se encuentran todos dentro de un radio determinado de otros ejemplos en el agrupamiento. Una cantidad predefinida de puntos centrales dentro de un radio especificado forma el núcleo del agrupamiento inicial, y los puntos que se encuentran dentro de un radio especificado de cualquiera de los puntos centrales se agregan luego al agrupamiento y comprenden el límite más externo del agrupamiento. A diferencia de muchos otros algoritmos de agrupamiento, a algunos ejemplos no se les asignará ningún agrupamiento, ya que cualquier punto que no esté lo suficientemente cerca de un punto central se tratará como ruido.

Aunque DBSCAN es potente y flexible, puede requerir experimentación para optimizar los parámetros para que se ajusten a los datos, como la cantidad de puntos que comprenden el núcleo o el radio permitido entre puntos, lo que agrega complejidad temporal al proyecto de aprendizaje automático. Por supuesto, el hecho de que los métodos basados en modelos sean más sofisticados no implica que sean los más adecuados para todos los proyectos de agrupamiento. Como veremos a lo largo del resto de este tema, un método basado en particiones más simple puede funcionar sorprendentemente bien en una tarea de agrupamiento desafiante del mundo real.

Aunque el modelado de mezclas y DBSCAN no se demuestran en este documento, existen paquetes R que se pueden usar para aplicar estos métodos a tus propios datos. El paquete mclust ajusta un modelo a mezclas de distribuciones gaussianas, y el paquete dbscan proporciona una implementación rápida del algoritmo DBSCAN.

El algoritmo de agrupamiento k-means

El **algoritmo k-means** es quizás el método de agrupamiento más utilizado y es un ejemplo de un enfoque de agrupamiento basado en particiones. Habiendo sido estudiado durante varias décadas, sirve como base para muchas técnicas de agrupamiento más sofisticadas. Si comprendes los principios simples que utiliza, tendrás el conocimiento necesario para comprender casi cualquier algoritmo de agrupamiento que se utilice en la actualidad.

A medida que k-means ha evolucionado con el tiempo, existen muchas implementaciones del algoritmo. Un enfoque temprano se describe en A k-means clustering algorithm, Hartigan, J.A., Wong, M.A., Applied Statistics, 1979, Vol. 28, pp. 100-108.

Aunque los métodos de agrupamiento han evolucionado desde el inicio de k-means, esto no implica que k-means esté obsoleto. De hecho, el método puede ser más popular ahora que nunca. La siguiente tabla enumera algunas razones por las que el algoritmo k-means todavía se utiliza ampliamente:

Fortalezas	**Debilidades**
• Utiliza principios simples que se pueden explicar en términos no estadísticos • Es muy flexible y se puede adaptar con ajustes simples para abordar muchas de sus deficiencias • Funciona bastante bien en muchos casos de uso del mundo real	• No es tan sofisticado como los algoritmos de agrupamiento más modernos • Debido a que utiliza un elemento de azar, no se garantiza que encuentre el conjunto óptimo de clústeres • Requiere una estimación razonable de cuántos clústeres existen naturalmente en los datos • No es ideal para clústeres no esféricos o clústeres de densidad muy variable

Si el nombre k-means te suena familiar, es posible que recuerdes el algoritmo de k-vecinos más cercanos (k-NN, *k-nearest neighbors*) presentado en el documento, Aprendizaje perezoso: Clasificación utilizando vecinos más cercanos. Como verás pronto, k-means tiene más en común con k-NN que solo la letra k.

El algoritmo k-means asigna cada uno de los *n* ejemplos a uno de los *k* grupos, donde *k* es un número que se ha determinado de antemano. El objetivo es minimizar las diferencias en los valores de las características de los ejemplos dentro de cada grupo y maximizar las diferencias entre los grupos.

A menos que *k* y *n* sean extremadamente pequeños, no es posible calcular los grupos óptimos en todas las combinaciones posibles de ejemplos. En cambio, el algoritmo utiliza un proceso heurístico que encuentra soluciones óptimas a nivel local. En pocas palabras, esto significa que comienza con una estimación inicial de las asignaciones de los grupos y luego modifica las asignaciones ligeramente para ver si los cambios mejoran la homogeneidad dentro de los grupos (clústeres).

En breve abordaremos el proceso en profundidad, pero el algoritmo implica esencialmente dos fases. En primer lugar, asigna ejemplos a un conjunto inicial de *k* clústeres. A continuación, actualiza las asignaciones ajustando los límites de los clústeres según los ejemplos que actualmente pertenecen al clúster. El proceso de actualización y asignación se produce varias veces hasta que los cambios ya no mejoran el ajuste del clúster.
En este punto, el proceso se detiene y los clústeres se finalizan.

Debido a la naturaleza heurística de k-means, es posible que obtengas resultados algo diferentes si realizas solo cambios leves en las condiciones iniciales. Si los resultados varían drásticamente, esto podría indicar un problema. Por ejemplo, es posible que los datos no tengan agrupaciones naturales o que el valor de *k* se haya elegido mal. Con esto en mente, es una buena idea intentar un análisis de clústeres más de una vez para probar la solidez de tus hallazgos.

Para ver cómo funciona el proceso de asignación y actualización en la práctica, revisemos nuevamente el caso de la conferencia hipotética de ciencia de datos. Aunque este es un ejemplo simple, ilustrará los conceptos básicos de cómo funciona k-means en profundidad.

Uso de la distancia para asignar y actualizar clústeres

Al igual que con k-NN, k-means trata los valores de las características como coordenadas en un espacio de características multidimensional.

Para los datos de la conferencia, solo hay dos características, por lo que podemos representar el espacio de características como un diagrama de dispersión bidimensional como se describió anteriormente.

El algoritmo k-means comienza eligiendo *k* puntos en el espacio de características para que sirvan como centros de los clústeres. Estos centros son el catalizador que impulsa a los ejemplos restantes a ubicarse en su lugar. A menudo, los puntos se eligen seleccionando *k* ejemplos aleatorios del conjunto de datos de entrenamiento. Debido a que esperamos identificar tres clústeres, utilizando este método, se seleccionarán *k* = 3 puntos al azar.

Estos puntos se indican con la estrella, el triángulo y el diamante en la figura 5:

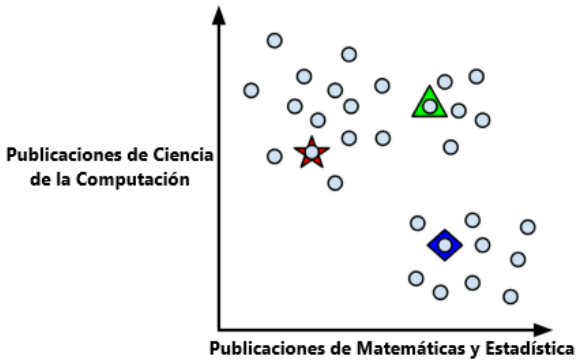

Figura 5: La agrupación con k-means comienza seleccionando *k* centros de clúster aleatorios.

Vale la pena señalar que, aunque los tres centros de los clústeres del diagrama anterior están muy espaciados entre sí, no siempre será así necesariamente. Debido a que los puntos de partida se seleccionan al azar, los tres centros podrían haber sido fácilmente tres puntos

adyacentes. Combinado con el hecho de que el algoritmo k-means es muy sensible a la posición inicial de los centros de los clústeres, un buen o mal conjunto de centros iniciales de los clústeres puede tener un impacto sustancial en el conjunto final de clústeres.

Para abordar este problema, se puede modificar k-means para utilizar diferentes métodos para elegir los centros iniciales. Por ejemplo, una variante elige valores aleatorios que aparecen en cualquier parte del espacio de características en lugar de seleccionar solo entre los valores observados en los datos. Otra opción es omitir este paso por completo; al asignar aleatoriamente cada ejemplo a un clúster, el algoritmo puede pasar inmediatamente a la fase de actualización. Cada uno de estos enfoques agrega un sesgo particular al conjunto final de clústeres, que puede utilizar para mejorar sus resultados.

En 2007, se introdujo un algoritmo llamado k-means++, que propone un método alternativo para seleccionar los centros de los clústeres iniciales. Pretende ser una forma eficiente de acercarse mucho más a la solución de agrupamiento óptima y, al mismo tiempo, reducir el impacto del azar. Para obtener más información, consulta K-means++: The advantage of concerned seeding, Arthur, D, Vassilvitskii, S, Proceedings of the eightth annual ACM-SIAM symposium on discrete algorithms, 2007, pp. 1027–1035.

Después de elegir los centros de los clústeres iniciales, los demás ejemplos se asignan al centro de clúster más cercano según una función de distancia, que se utiliza como medida de similitud. Tal vez recuerdes que utilizamos funciones de distancia como medidas de similitud mientras aprendíamos sobre el algoritmo de aprendizaje supervisado k-NN. Al igual que k-NN, k-means tradicionalmente utiliza la distancia euclidiana, pero se pueden utilizar otras funciones de distancia si se desea.

Curiosamente, cualquier función que devuelva una medida numérica de similitud podría utilizarse en lugar de una función de distancia tradicional. De hecho, k-means podría incluso adaptarse para agrupar imágenes o documentos de texto mediante una función que mida la similitud de pares de imágenes o textos.

Para aplicar la función de distancia, recuerda que si *n* indica el número de características, la fórmula para la distancia euclidiana entre el ejemplo *x* y el ejemplo *y* es la siguiente:

$$dist(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Por ejemplo, para comparar un invitado con cinco publicaciones de computación y una publicación de matemáticas con un invitado con cero artículos de computación y dos artículos de matemáticas, podríamos calcular esto en R como:

```
> sqrt((5 - 0)^2 + (1 - 2)^2)
[1] 5.09902
```

Usando la función de distancia de esta manera, encontramos la distancia entre cada ejemplo y cada centro de clúster. Luego, cada ejemplo se asigna al centro de clúster más cercano.

> Ten en cuenta que, dado que estamos utilizando cálculos de distancia, todas las características deben ser numéricas y los valores deben normalizarse a un rango estándar con anticipación. Los métodos presentados en el tema de k-NN, resultarán útiles para esta tarea.

Como se muestra en la siguiente figura, los tres centros de los clústeres dividen los ejemplos en tres particiones denominadas Clúster A, Clúster B y Clúster C. Las líneas discontinuas indican los límites del **diagrama de Voronoi** creado por los centros de los clústeres. El diagrama de Voronoi indica las áreas que están más cerca de un centro de clúster que de cualquier otro; el vértice donde se encuentran los tres límites es la distancia máxima desde los tres centros de clústeres.

Usando estos límites, podemos ver fácilmente las regiones reclamadas por cada una de las semillas iniciales de k-means:

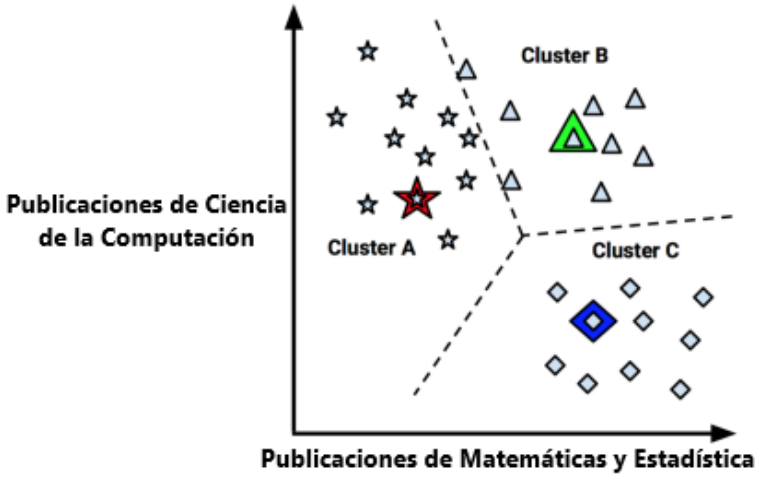

Figura 6: Los centros de clústeres iniciales crean tres grupos de puntos “más cercanos”.

Ahora que se ha completado la fase de asignación inicial, el algoritmo de k-means procede a la fase de actualización. El primer paso para actualizar los clústeres implica desplazar los centros iniciales a una nueva ubicación, conocida como centroide, que se calcula como la posición promedio de los puntos actualmente asignados a ese clúster. La figura 7 ilustra cómo, a medida que los centros de los grupos se desplazan hacia los nuevos centroides, los límites en el diagrama de Voronoi también se desplazan y un punto que alguna vez estuvo en el grupo B (indicado por una flecha) se agrega al grupo A:

Como resultado de esta reasignación, el algoritmo k-means continuará a través de otra fase de actualización. Después de cambiar los centroides de los clústeres, actualizar los límites de los clústeres y reasignar puntos a nuevos clústeres (como lo indican las flechas), la figura se ve así (figura 8):

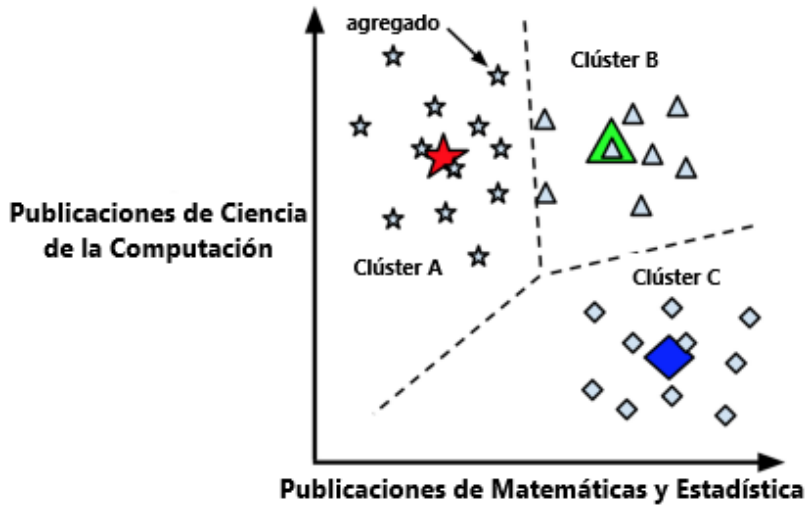

Figura 9.7: La fase de actualización desplaza los centros de los grupos, lo que provoca la reasignación de un punto.

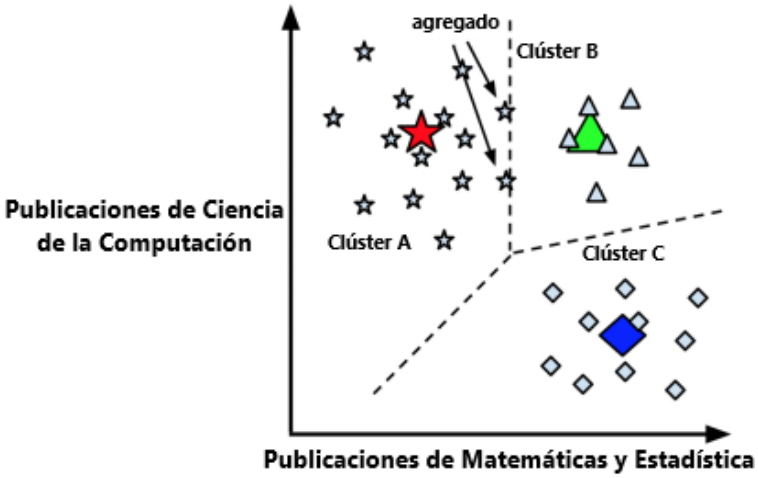

Figura 9.8: Después de otra actualización, se reasignan dos puntos más al centro del clúster más cercano.

Debido a que se reasignaron dos puntos más, debe ocurrir otra actualización, que mueve los centroides y actualiza los límites del clúster. Sin embargo, debido a que estos cambios no dan como resultado reasignaciones, el algoritmo k-means se detiene. Las asignaciones de clústeres ahora son definitivas:

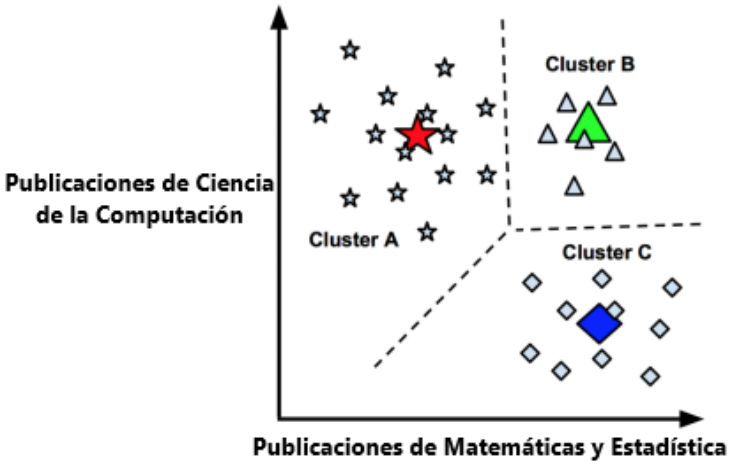

Figura 9.9: La agrupación se detiene después de que la fase de actualización no dé como resultado nuevas asignaciones de clústeres.

Los clústeres finales se pueden informar de una de dos maneras. Primero, puedes simplemente informar las asignaciones de clústeres de A, B o C para cada ejemplo. Alternativamente, puedes informar las coordenadas de los centroides del clúster después de la actualización final.

Dado cualquiera de los métodos de informe, puedes calcular el otro; Puedes calcular los centroides utilizando las coordenadas de los ejemplos de cada grupo, o puedes utilizar las coordenadas del centroide para asignar cada ejemplo a su centro de grupo más cercano.

Elección del número adecuado de grupos

En la introducción a k-means, aprendimos que el algoritmo es sensible a los centros de los grupos elegidos aleatoriamente. De hecho, si hubiéramos seleccionado una combinación diferente de tres puntos de partida en el ejemplo anterior, podríamos haber encontrado grupos que dividieran los datos de manera diferente a la que esperábamos. De manera similar, k-means es sensible al número de grupos; la elección requiere un equilibrio delicado. Establecer *k* muy grande mejorará la homogeneidad de los grupos y, al mismo tiempo, corre el riesgo de sobreajustar los datos.

Lo ideal es que tengas un conocimiento *a priori* (una creencia previa) sobre las agrupaciones verdaderas y puedas aplicar esta información para elegir el número de grupos. Por ejemplo, si agrupas películas, puedes comenzar estableciendo *k* igual al número de géneros considerados para los Premios de la Academia. En el problema de asientos de la conferencia de ciencia de datos que analizamos anteriormente, *k* podría reflejar la cantidad de campos académicos de estudio a los que pertenecen los invitados.

A veces, la cantidad de clústeres está determinada por los requisitos comerciales o la motivación para el análisis. Por ejemplo, la cantidad de mesas en la sala de reuniones podría determinar cuántos grupos de personas se deben crear a partir de la lista de asistentes de ciencia de datos. Si ampliamos esta idea a otro caso comercial, si el departamento de marketing solo tiene los recursos para crear tres campañas publicitarias distintas, podría tener sentido establecer $k = 3$ para asignar todos los clientes potenciales a una de las tres propuestas.

Sin ningún conocimiento previo, una regla general sugiere establecer *k* igual a la raíz cuadrada de (*n* / 2), donde *n* es la cantidad de ejemplos en el conjunto de datos. Sin embargo, es probable que esta regla general dé como resultado una cantidad de clústeres difícil de manejar para conjuntos de datos grandes. Afortunadamente, existen otros métodos cuantitativos que pueden ayudar a encontrar un conjunto de clústeres de k-medias adecuado.

Una técnica conocida como el **método del codo** intenta medir cómo cambia la homogeneidad o heterogeneidad dentro de los clústeres para varios valores de *k*. Como se ilustra en los siguientes diagramas, se espera que la homogeneidad dentro de los clústeres aumente a medida que se agregan clústeres adicionales; de manera similar, la heterogeneidad dentro de los clústeres debería disminuir con más clústeres.

Debido a que se podrían seguir viendo mejoras hasta que cada ejemplo esté en su propio clúster, el objetivo no es maximizar la homogeneidad o minimizar la heterogeneidad infinitamente, sino más bien encontrar k de modo que haya rendimientos decrecientes más allá de ese valor. Este valor de k se conoce como el punto del codo porque se dobla como la articulación del codo del brazo humano.

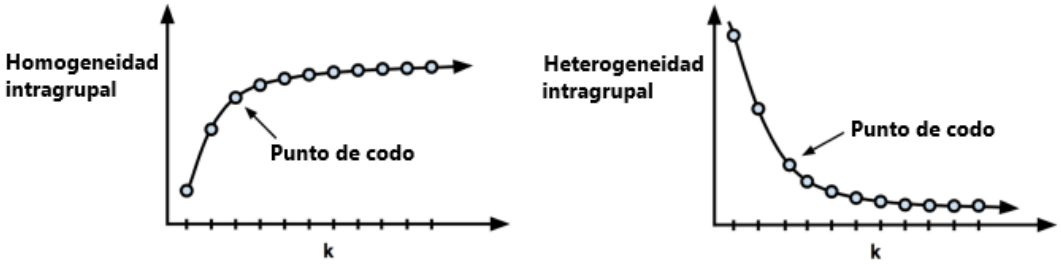

Figura 10: El codo es el punto en el que el aumento de k produce mejoras relativamente pequeñas.

Existen numerosas estadísticas para medir la homogeneidad y la heterogeneidad dentro de los clústeres que se pueden utilizar con el método del codo (el cuadro de información que sigue proporciona una cita para obtener más detalles). Sin embargo, en la práctica, no siempre es posible probar iterativamente una gran cantidad de valores *k*. Esto se debe en parte a que la agrupación de grandes conjuntos de datos puede llevar bastante tiempo; agrupar los datos repetidamente es aún peor. Además, las aplicaciones que requieren el conjunto exacto y

óptimo de clústeres son poco frecuentes. En la mayoría de las aplicaciones de agrupación, basta con elegir un valor *k* en función de la conveniencia en lugar del que crea los clústeres más homogéneos.

Para una revisión exhaustiva de la amplia variedad de medidas de rendimiento de los clústeres, consulta On Clustering Validation Techniques, Halkidi, M, Batistakis, Y, Vazirgiannis, M, Journal of Intelligent Information Systems, 2001, vol. 17, págs. 107-145.

El proceso de establecer *k* en sí mismo puede a veces llevar a ideas interesantes. Al observar cómo cambian las características de los grupos a medida que cambia *k*, se puede inferir dónde los datos tienen límites definidos naturalmente. Los grupos que están agrupados más estrechamente cambiarán muy poco, mientras que los grupos menos homogéneos se formarán y se disolverán con el tiempo.

En general, puede ser prudente dedicar poco tiempo a preocuparse por obtener *k* exactamente. El siguiente ejemplo demostrará cómo incluso un poco de conocimiento de la materia tomado de una película de Hollywood se puede utilizar para establecer *k* de manera que se encuentren grupos procesables e interesantes. Como la agrupación no está supervisada, la tarea realmente depende de lo que hagas con ella; el valor está en los conocimientos que extraes de los hallazgos del algoritmo.

Resumen

Nuestros hallazgos respaldan el adagio popular de que “los pájaros del mismo plumaje vuelan juntos”. Al utilizar métodos de aprendizaje automático para agrupar a los adolescentes con otros que tienen intereses similares, pudimos desarrollar una tipología de identidades adolescentes, que predijo características personales como el género y la cantidad de amigos. Estos mismos métodos se pueden aplicar a otros contextos con resultados similares (los constatarás en el ejemplo).

En este documento se han tratado solo los aspectos básicos de la agrupación en clústeres. Existen muchas variantes del algoritmo k-means, así como muchos otros algoritmos de agrupación en clústeres, que aportan sesgos y heurísticas únicos a la tarea. Con base en los fundamentos de este documento, podrás comprender estos métodos de agrupación en clústeres y aplicarlos a nuevos problemas.

En el próximo tema, comenzaremos a analizar métodos para medir el éxito de un algoritmo de aprendizaje que se pueden aplicar en muchas tareas de aprendizaje automático. Si bien nuestro proceso siempre ha dedicado cierto esfuerzo a evaluar el éxito del aprendizaje, para obtener el mayor grado de rendimiento, es fundamental poder definirlo y medirlo en los términos más estrictos.