

## Clustering

“Most of us cluster somewhere in the middle of most statistical distributions. But there are lots of bell curves, and pretty much everyone is on a tail of at least one of them. We may collect strange memorabilia or read esoteric books, hold unusual religious beliefs or wear odd-sized shoes, suffer rare diseases or enjoy obscure movies.”

---

Virginia Postrel

Pada bab 4, kamu sudah mempelajari salah satu teknik *clustering* yang cukup umum yaitu K-means. Bab ini akan mengupas *clustering* secara lebih dalam. Kami sarankan kamu untuk membaca paper [33], walaupun relatif lama, tetapi paper tersebut memberikan penjelasan yang mudah dimengerti tentang *clustering*. Selain itu, kamu juga dapat membaca *paper* oleh Saad et al. [34].

**Clustering** adalah pengelompokan data dengan sifat yang mirip. Data untuk *clustering* tidak memiliki label (kelas). Secara umum, algoritma *clustering* dapat dikategorikan menjadi dua macam berdasarkan hasil yang diinginkan [35]: (1) *partitional*, yaitu menentukan partisi sejumlah  $K$  dan (2) *hierarchical*, yaitu mengelompokan data berdasarkan struktur taksonomi. Contoh algoritma *partitional* adalah **K-means** pada subbab 8.1, sementara contoh algoritma *hierarchical* adalah **agglomerative clustering** pada subbab 8.2.

## 8.1 K-means, Pemilihan Centroid, Kemiripan Data

Algoritma K-means mengelompokkan data menjadi sejumlah  $K$  kelompok sesuai yang kita definisikan. Algoritma ini disebut juga sebagai *flat clustering*, artinya kelompok satu memiliki kedudukan sejajar dengan kelompok lainnya. Kita tinjau kembali tahapan-tahapan algoritma K-means sebagai berikut:

1. Tentukan sejumlah  $K$  kelompok yang kita inginkan.
2. Inisiasi **centroid** untuk setiap kelompok. Centroid ibarat seperti “ketua kelompok”, yang merepresentasikan kelompok.
3. Hitung kedekatan suatu data terhadap *centroid*, kemudian masukkan data tersebut ke kelompok yang **centroid**-nya memiliki sifat terdekat dengan dirinya.
4. Pilih kembali centroid untuk masing-masing kelompok, yaitu dari anggota kelompok tersebut.
5. Ulangi langkah-langkah sebelumnya sampai tidak ada perubahan anggota untuk semua kelompok.

Perhatikan, ada dua hal penting pada algoritma K-means yaitu: (1) memilih centroid dan (2) Perhitungan kemiripan data. Pada bab 4, dijelaskan salah satu metode pemilihan centroid paling sederhana yaitu secara acak. Pada kenyataannya, inisiasi centroid untuk setiap kelompok/*cluster* dapat dilakukan secara acak; tetapi pada tahap berikutnya, secara umum centroid dipilih menggunakan nilai rata-rata/*mean*. Dengan demikian, centroid bisa saja merupakan suatu vektor yang tidak ada *entry*-nya di dataset.

Diberikan sekumpulan data  $\mathbf{D} = \{\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_N\}$  pada suatu *cluster* dalam bentuk *feature vector*, maka centroid  $\mathbf{c}$  untuk *cluster* itu dihitung dengan

$$\mathbf{c} = \frac{1}{N} \sum_{i=1}^N \mathbf{d}_i \quad (8.1)$$

yaitu nilai rata-rata setiap elemen *feature vector* untuk seluruh anggota *cluster* tersebut, dimana  $N$  adalah banyaknya anggota *cluster* dan  $\mathbf{d}_i$  adalah anggota ke- $i$  dalam representasi *feature vector*. Dengan ini, centroid secara umum bisa jadi tidak merupakan elemen anggota *cluster* (centroid bukan sebuah instans data).

Pada bab 4 dijelaskan salah satu metode perhitungan kemiripan data sederhana yaitu dengan menghitung banyaknya nilai atribut yang sama di antara dua *feature vector*. Selain metode tersebut, terdapat banyak perhitungan kemiripan data lainnya tergantung pada tipe, contohnya:

1. **Numerik.** Euclidean Distance, Manhattan Distance, Cosine Distance, dsb.
2. **Boolean.** Jaccard Dissimilarity, Rogers Tanimoto Dissimilarity, dsb.
3. **String.** Levenshtein Distance, Hamming Distance, dsb.

Perhitungan yang paling populer digunakan adalah *cosine similarity*<sup>1</sup> (kebetulan pada kebanyakan kasus kita bekerja dengan data numerik), didefinisikan pada persamaan 8.2, yaitu *dot product* antara dua vektor dibagi dengan perkalian *norm* kedua vektor.

$$\text{cosSim}(\mathbf{d}_i, \mathbf{d}_j) = \frac{\mathbf{d}_i \mathbf{d}_j}{\|\mathbf{d}_i\| \|\mathbf{d}_j\|} \quad (8.2)$$

Clusters yang terbentuk, nantinya dapat digunakan sebagai pengelompokkan untuk label klasifikasi. Seumpama *cluster*<sub>1</sub> dapat dianggap sebagai data untuk kelas ke-1, dst.

## 8.2 Hierarchical Clustering

*Hierarchical clustering* adalah teknik untuk membentuk pembagian bersarang (*nested partition*). Berbeda dengan K-means yang hasil *clustering*-nya berbentuk *flat* atau rata, *hierarchical clustering* memiliki satu *cluster* paling atas yang mencakup konsep seluruh *cluster* dibawahnya. Ada dua cara untuk membentuk *hierarchical clustering* [33]:

1. **Agglomerative.** Dimulai dari beberapa *flat clusters*; pada setiap langkah iterasi, kita menggabungkan dua *clusters* termirip. Artinya, kita harus mendefinisikan arti “kedekatan” dua *clusters*.
2. **Divisive.** Dimulai dari satu *cluster* (seluruh data), kemudian kita memecah belah *cluster*. Artinya, kita harus mendefinisikan *cluster* mana yang harus dipecah dan bagaimana cara memecahnya.

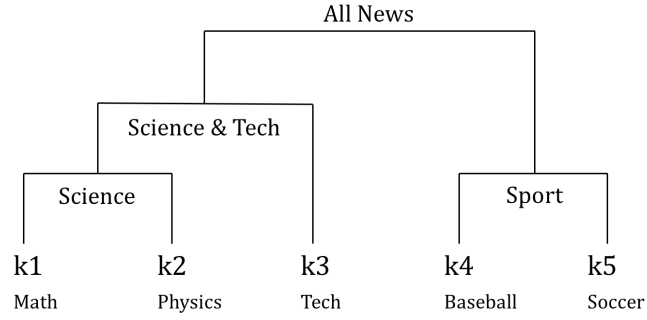
Sebagai contoh, algoritma *hierarchical clustering* menghasilkan struktur hirarkis seperti pada gambar 8.1 yang disebut **dendogram**. Dendogram melambangkan taksonomi, sebagai contoh taksonomi dokumen berita *sport*, dipecah menjadi *baseball* dan *soccer*.

Sejauh ini, teknik **agglomerative clustering** lebih populer, karena pendekatan ini bersifat *bottom-up*. Secara umum, pendekatan *bottom-up* memang relatif lebih populer dibanding pendekatan *top-down*. Langkah-langkah *agglomerative clustering* sebagai berikut:

1. Sediakan sejumlah *K clusters*. Kamu dapat menganggap satu instans data sebagai suatu *cluster*.
2. Gabung dua *clusters* paling mirip.
3. Ulangi langkah ke-2 sampai hanya satu *cluster* tersisa.

Perhatikan! untuk menggabungkan dua *clusters* termirip, kita membutuhkan definisi “mirip”. Definisi tersebut dikuantifikasi dengan formula matematis (seperti definisi kemiripan data pada subbab 8.1). Perhitungan kemiripan *clusters* dapat dihitung dengan tiga metode [35] (untuk data numerik):

<sup>1</sup> [https://en.wikipedia.org/wiki/Cosine\\_similarity](https://en.wikipedia.org/wiki/Cosine_similarity)



**Gambar 8.1.** Ilustrasi *Hierarchical Clustering*.

1. **Single Link.** Nilai kemiripan dua *clusters*  $\mathbf{U}$  dan  $\mathbf{V}$  dihitung berdasarkan nilai kemiripan **maksimum** diantara anggota kedua *clusters* tersebut<sup>2</sup>.

$$Sim_{single-link}(\mathbf{U}, \mathbf{V}) = \max_{\mathbf{u}_i \in \mathbf{U}, \mathbf{v}_j \in \mathbf{V}} cosSim(\mathbf{u}_i, \mathbf{v}_j) \quad (8.3)$$

2. **Complete Link.** Nilai kemiripan dua *clusters* dihitung berdasarkan nilai kemiripan **minimum** diantara anggota kedua *clusters* tersebut.

$$Sim_{complete-link}(\mathbf{U}, \mathbf{V}) = \min_{\mathbf{u}_i \in \mathbf{U}, \mathbf{v}_j \in \mathbf{V}} cosSim(\mathbf{u}_i, \mathbf{v}_j) \quad (8.4)$$

3. **UPGMA (Average Link).** Nilai kemiripan dua *clusters* dihitung berdasarkan nilai kemiripan **rata-rata** diantara anggota kedua *clusters* tersebut.

$$Sim_{UPGMA}(\mathbf{U}, \mathbf{V}) = \frac{1}{|\mathbf{U}| |\mathbf{V}|} \sum_{\mathbf{u}_i \in \mathbf{U}, \mathbf{v}_j \in \mathbf{V}} cosSim(\mathbf{u}_i, \mathbf{v}_j) = \frac{\mathbf{c}^U \mathbf{c}^V}{|\mathbf{U}| |\mathbf{V}|} \quad (8.5)$$

dimana  $|\mathbf{U}|$  adalah banyaknya data pada *cluster*  $\mathbf{U}$  dan  $\mathbf{c}^U$  adalah centroid untuk *cluster*  $\mathbf{U}$ .

### 8.3 Evaluasi

Diberikan sekumpulan data  $\mathbf{D} = \{\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_N\}$  untuk suatu *cluster*. Saat tidak tersedianya informasi label/kelas untuk setiap data, kualitas hasil *clustering* dapat dihitung dengan tiga kriteria yaitu:

<sup>2</sup> *Cluster* dapat dianggap sebagai matriks karena merupakan kumpulan *feature vector*.

1. **Intra-cluster similarity**, yaitu menghitung rata-rata kedekatan antara suatu anggota dan anggota *cluster* lainnya.

$$I = \frac{1}{N^2} \sum_{\mathbf{d}_i, \mathbf{d}_j, i \neq j} \cosSim(\mathbf{d}_i, \mathbf{d}_j) \quad (8.6)$$

Perhitungan kedekatan antar tiap pasang anggota *cluster* sama halnya dengan menghitung *norm* dari centroid *cluster* tersebut, ketika centroid dihitung menggunakan *mean* (buktikan!).

$$I = \frac{1}{N^2} \sum_{\mathbf{d}_i, \mathbf{d}_j, i \neq j} \cosSim(\mathbf{d}_i, \mathbf{d}_j) = \|\mathbf{c}\|^2 \quad (8.7)$$

Perhitungan ini dapat dinormalisasi sesuai dengan banyaknya anggota *cluster*

$$I' = \frac{\|\mathbf{c}\|^2}{N} \quad (8.8)$$

Semakin tinggi kemiripan anggota pada suatu *cluster*, semakin baik kualitas *cluster* tersebut.

2. **Inter-cluster similarity**, yaitu menghitung bagaimana perbedaan antara suatu *cluster* dan *cluster* lainnya. Hal tersebut dihitung dengan *co-sine similarity* antara centroid suatu *cluster* dan centroid dari seluruh data [34].

$$E = \sum_{k=1}^K N_k \frac{\mathbf{c}_k \mathbf{c}^D}{\|\mathbf{c}_k\|} \quad (8.9)$$

dimana  $\mathbf{c}_k$  adalah centroid *cluster* ke- $k$ ,  $\mathbf{c}^D$  adalah centroid (*mean*) dari seluruh data,  $N_k$  adalah banyaknya anggota *cluster* ke- $k$ , dan  $K$  adalah banyaknya *clusters*. Semakin kecil nilai *inter-cluster similarity*, maka semakin baik kualitas *clustering*.

3. **Hybrid**. Perhitungan *intra-cluster* dan *inter-cluster* mengoptimalkan satu hal sementara tidak memperdulikan hal lainnya. *Intra-cluster* menghitung keerratan anggota *cluster*, sementara *Inter-cluster* menghitung separasi antar *clusters*. Kita dapat menggabungkan keduanya sebagai *hybrid* (gabungan), dihitung dengan:

$$H = \frac{\sum_{k=1}^K I'_k}{E} = \frac{\sum_{k=1}^K \frac{\|\mathbf{c}_k\|^2}{N_k}}{\sum_{k=1}^K N_k \frac{\mathbf{c}_k \mathbf{c}^D}{\|\mathbf{c}_k\|}} = \sum_{k=1}^K \frac{\|\mathbf{c}_k\|}{N_k^2 \mathbf{c}_k \mathbf{c}^D} \quad (8.10)$$

Semakin besar nilai perhitungan *hybrid*, semakin bagus kualitas *clusters*.

Apabila terdapat informasi label/ kelas untuk setiap data, kita juga dapat menghitung kualitas algoritma *clustering* (perhatikan! tujuan pengukuran adalah kualitas algoritma) dengan **Entropy** dan **Purity**.

## Soal Latihan

### 8.1. Intra-cluster Evaluation

Buktikan kebenaran persamaan 8.7.

### 8.2. Entropy

Bagaimana cara menghitung kualitas algoritma *clustering*, jika diberikan informasi label/ kelas setiap data menggunakan: (hint, baca [33])

- (a) Entropy
- (b) Purity

### 8.3. Kompleksitas

Hitunglah kompleksitas algoritma untuk:

- (a) K-means
- (b) Agglomerative Clustering
- (c) Divisive Clustering

### 8.4. Kemiripan Data

Sebutkanlah contoh perhitungan kemiripan untuk data *string*. Bagaimana adaptasi perhitungan tersebut pada formula-formula yang sudah diberikan pada algoritma K-means dan agglomerative clustering.

### 8.5. Agglomerative vs Divisive Clustering

Menurut kamu, mengapa pendekatan *bottom-up* (agglomerative) lebih populer dibanding *top-down* (divisive)? Apa perbedaan kedua pendekatan tersebut (keuntungan dan kerugian masing-masing)?

### 8.6. Agglomerative Link

Jelaskan apa kelebihan dan kekurangan masing-masing metode perhitungan kemiripan cluster pada agglomerative clustering!