Kevin Alif Bagaskara

1103210075

UTS Machine Learning

1. Jika algoritma K-Means menghasilkan nilai silhouette score rendah (0.3) meskipun elbow method menunjukkan K=5 sebagai optimal pada dataset ini, faktor apa yang menyebabkan inkonsistensi ini? Bagaimana strategi validasi alternatif (misal: analisis gap statistic atau validasi stabilitas cluster via bootstrapping) dapat mengatasi masalah ini, dan mengapa distribusi data non-spherical menjadi akar masalahnya?

Ketika elbow method menunjukkan K=5 sebagai titik optimal tetapi silhouette score justru rendah (misalnya 0.3), ini tanda bahwa cluster yang terbentuk secara geometri mungkin meminimalkan inertia (jarak dalam cluster), tetapi tidak mencerminkan pemisahan yang baik antar cluster. Ini sering terjadi ketika distribusi data tidak berbentuk non-spherical atau mengandung outlier yang mengacaukan centroid. K-Means memang mengasumsikan distribusi cluster bersifat isotropik dan setara ukurannya. Cara mengatasinya menggunakan pendekatan alternatif seperti gap statistic yang membandingkan inertia aktual dengan distribusi acak atau bootstrapping cluster stability yang mengukur konsistensi hasil clustering di subset acak data. Ini lebih tahan terhadap bentuk cluster yang kompleks atau tidak simetris.

2. Dalam dataset dengan campuran fitur numerik (Quantity, UnitPrice) dan kategorikal high-cardinality (Description), metode preprocessing apa yang efektif untuk menyelaraskan skala dan merepresentasikan fitur teks sebelum clustering? Jelaskan risiko menggunakan One-Hot Encoding untuk Description, dan mengapa teknik seperti TF-IDF atau embedding berdimensi rendah (UMAP) lebih robust untuk mempertahankan struktur cluster.

Pada dataset dengan campuran fitur numerik dan fitur kategorikal teks Description, tantangan utama adalah menyelaraskan representasi antar tipe data untuk clustering yang seimbang. Standardisasi dan normalisasi (z-score dan Min-Max) cocok untuk fitur numerik agar semua skala setara. Untuk Description, menggunakan One-Hot Encoding tidak disarankan karena menghasilkan dimensi sangat tinggi. TF-IDF lebih cocok karena memberi bobot berdasarkan informasi penting, bukan frekuensi semata, dan tetap menghasilkan vektor berdimensi sedang. Untuk representasi yang lebih padat dan struktur-preserving, embedding berdimensi rendah seperti UMAP atau Autoencoder bisa digunakan setelah TF-IDF.

3. Hasil clustering dengan DBSCAN sangat sensitif terhadap parameter epsilon—bagaimana menentukan nilai optimal epsilon secara adaptif untuk memisahkan cluster padat dari noise pada data transaksi yang tidak seimbang (misal: 90% pelanggan dari UK)? Jelaskan peran k-distance graph dan kuartil ke-3 dalam automasi parameter, serta mengapa MinPts harus disesuaikan berdasarkan kerapatan regional!

DBSCAN bergantung pada dua parameter; epsilon (jarak maksimum untuk menganggap titik sebagai tetangga) dan MinPts (jumlah minimum tetangga agar titik dianggap sebagai inti). Pada dataset tidak seimbang seperti dominasi 90% pelanggan dari UK, menentukan epsilon sembarangan bisa menyebabkan seluruh minoritas dianggap noise. Untuk pendekatan adaptif, k-distance graph adalah alat yang kuat. Jarak ke tetangga ke-k (biasanya k = MinPts) diplot, dan elbow atau titik infleksi pada grafik menunjukkan epsilon ideal. Pendekatan statistik seperti menggunakan kuartil ke-3 dari

distribusi k-distance dapat otomatis menetapkan epsilon yang memisahkan noise dari cluster padat. Sementara itu, MinPts sebaiknya disesuaikan dengan regional density. Lebih tinggi untuk cluster besar dan lebih rendah untuk yang jarang. Ini penting agar DBSCAN dapat mengenali struktur lokal, bukan hanya pola global, apalagi di data transaksi yang bersifat heterogen secara geografis atau frekuensi.

4. Jika analisis post-clustering mengungkapkan overlap signifikan antara cluster "high-value customers" dan "bulk buyers" berdasarkan total pengeluaran, bagaimana teknik semi-supervised (contoh: constrained clustering) atau integrasi metric learning (Mahalanobis distance) dapat memperbaiki pemisahan cluster? Jelaskan tantangan dalam mempertahankan interpretabilitas bisnis saat menggunakan pendekatan non-Euclidean!

Ketika dua segmen penting seperti "high-value customers" dan "bulk buyers" tampak tumpang tindih pada dimensi pengeluaran total, pendekatan clustering murni sering gagal mengisolasi mereka karena fitur dominan mendistorsi hasil. Semi-supervised clustering dengan constraint-based methods seperti Must-Link dan Cannot-Link dapat memaksa pembentukan cluster berdasarkan wawasan domain, misalnya memisahkan pelanggan berdasarkan kebiasaan pembelian, bukan hanya nilai transaksi. Di sisi lain, metric learning (misalnya Mahalanobis distance) memungkinkan penyesuaian jarak antar data berdasarkan korelasi fitur, sehingga model bisa belajar bahwa dua pelanggan dengan nilai transaksi sama bisa berbeda secara waktu pembelian atau frekuensi. Tantangannya adalah, pendekatan ini menurunkan interpretabilitas karena jarak non-Euclidean sulit dijelaskan dalam istilah bisnis. Maka penting untuk menyelaraskan hasil clustering dengan dimensi yang masih bisa ditafsirkan secara logis, misalnya melalui prototipe per-cluster atau fitur representative.

5. Bagaimana merancang temporal features dari InvoiceDate (misal: hari dalam seminggu, jam pembelian) untuk mengidentifikasi pola pembelian periodik (seperti transaksi pagi vs. malam)? Jelaskan risiko data leakage jika menggunakan agregasi temporal (misal: rata-rata pembelian bulanan) tanpa time-based cross-validation, dan mengapa lag features (pembelian 7 hari sebelumnya) dapat memperkenalkan noise pada cluster!

InvoiceDate bisa diubah menjadi fitur temporal yang kaya, seperti day_of_week, hour_of_day, atau bahkan segmentasi morning, afternoon, evening untuk menangkap pola pembelian periodik. Fitur ini sangat berguna untuk menemukan klaster pelanggan dengan kebiasaan waktu belanja tertentu. Namun, agregasi temporal seperti “rata-rata pembelian bulanan” atau “frekuensi pembelian per minggu” bisa menyebabkan data leakage jika dihitung dari seluruh data, termasuk periode test. Ini artinya model belajar dari informasi masa depan yang seharusnya tidak diketahui. Solusinya adalah menggunakan time-based cross-validation (misalnya time series split) saat membuat agregasi, agar hanya data sebelumnya yang digunakan untuk membentuk fitur. Selain itu, penggunaan lag features seperti “jumlah pembelian 7 hari sebelumnya” bisa menjadi noise jika pelanggan bersifat sporadis atau tidak memiliki histori lengkap. Oleh karena itu, lag features harus diuji stabilitasnya, misalnya dengan mengukur missingness dan distribusinya sebelum disertakan dalam clustering.