KOMPUTA : Jurnal Ilmiah Komputer dan Informatika
Vol. xx, No. x, mmmm yyyy, ISSN : 2089-9033, e- ISSN : 2715-7849

Ramgkuman Materi Perkuliahan Machine learning Kelompok 21

Muhammad Shaadam haidar Yuwono[1], Muhammad labib[2]

1.Deskripsi singkat

Pertemuan ini membahas teknik pengelompokan data (*clustering*) tingkat lanjut dengan membandingkan metode konvensional berbasis jarak seperti K-Means dengan metode berbasis distribusi statistik yaitu **Gaussian Mixture Model (GMM)**. Fokus utamanya adalah memahami bagaimana mesin mengidentifikasi kelompok tersembunyi berdasarkan pola probabilitas dan distribusi data.

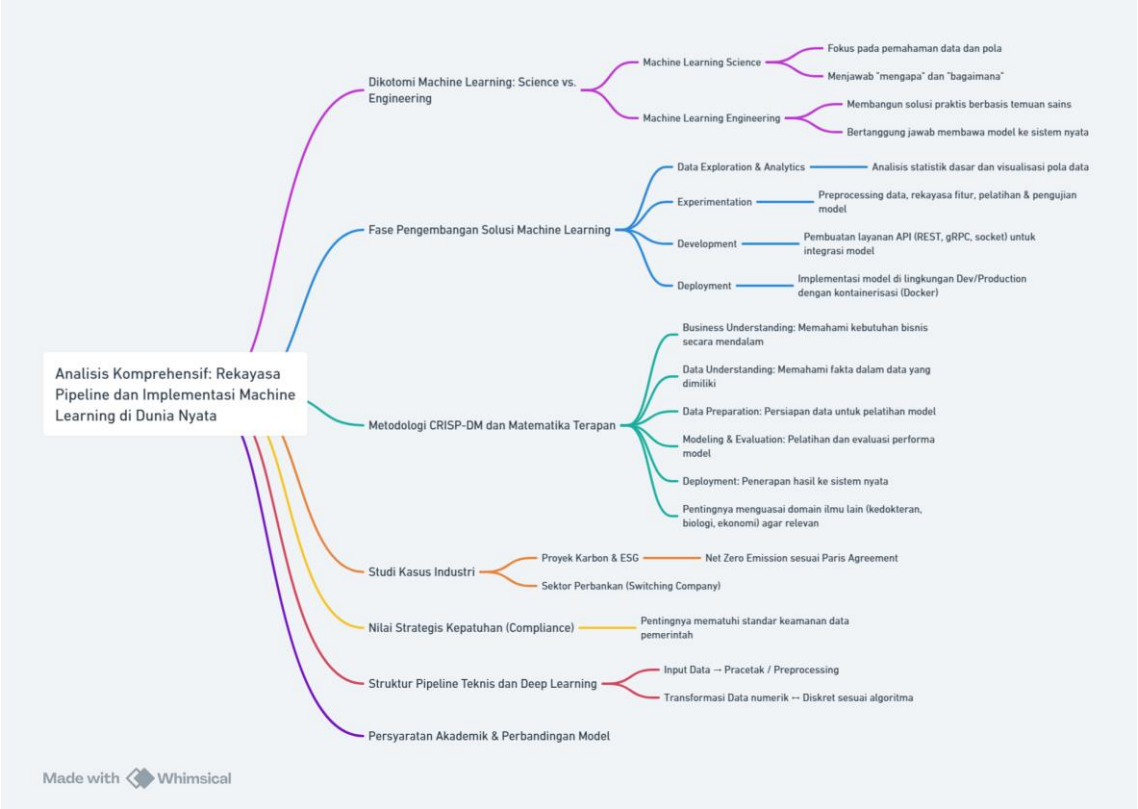
Poin-poin Utama:

1. **Transisi Paradigma Klasterisasi:**
    a. Berbeda dengan K-Means yang hanya menghitung jarak fisik ke titik tengah (*centroid*), GMM bekerja dengan asumsi bahwa setiap kelompok data mengikuti pola **Distribusi Normal (Gaussian)**. Hal ini membuat GMM lebih fleksibel dalam menangani kelompok data yang bentuknya tidak bulat sempurna atau lonjong.
2. **Identifikasi Kelompok melalui Grafik Frekuensi:**
    a. Dosen menjelaskan bahwa kelompok yang berbeda dapat dideteksi melalui "puncak" atau kenaikan data yang tiba-tiba pada grafik frekuensi. Setiap puncak merepresentasikan satu distribusi normal yang unik dalam sebuah populasi data yang besar.
3. **Dua Parameter Kunci (μ dan σ):**
    a. **Mu (μ):** Menunjukkan rata-rata atau lokasi pusat dari sebuah kelompok.
    b. **Sigma (σ):** Menunjukkan standar deviasi atau lebar persebaran data. Memahami kedua nilai ini sangat penting untuk mendefinisikan identitas matematis dari sebuah klaster.
4. **Konsep Keanggotaan Lembut (*Soft Membership*):**
    a. Dalam GMM, satu data tidak harus kaku dimiliki oleh satu kelompok saja. Sebaliknya, setiap data memiliki nilai **probabilitas** (tanggung jawab) untuk menjadi bagian dari beberapa kelompok sekaligus, yang dihitung menggunakan fungsi padat probabilitas.
5. **Optimasi dengan Algoritma EM:**
    a. GMM menggunakan siklus **Expectation-Maximization (EM)** untuk menemukan posisi dan lebar kelompok yang paling pas. Proses ini mirip dengan logika optimasi SGD pada *Neural Networks*, namun digunakan untuk model generatif dan probabilistik.

1.

2.Mindmap

Naskah dikirim: dd-mm-yyyy; direvisi dd-mm-yyyy; diterima: dd-mm-yyyy



3.penjelasan detil

Analisis Komprehensif: Pemodelan Probabilistik dan Algoritma Campuran Gaussian (GMM)

Pertemuan ketiga ini menandai pendalaman materi yang sangat krusial dalam memahami bagaimana mesin "melihat" kelompok dalam data yang tidak berlabel. Fokus utama bahasan adalah pergeseran paradigma dari metode klasterisasi tradisional yang kaku (seperti K-Means) menuju metode yang lebih luwes dan berbasis peluang, yaitu **Gaussian Mixture Model (GMM)**. Pemahaman ini penting karena menjadi fondasi bagi model-model generatif modern, termasuk teknologi di balik GPT dan model probabilitas lainnya.

1. Filosofi Pengelompokan: Dari Jarak Fisik ke Distribusi Peluang

Dosen mengawali sesi dengan memberikan komparasi mendasar antara dua cara mesin mengelompokkan data.

- **K-Means (Berbasis Jarak):** Algoritma ini bekerja dengan menempatkan titik pusat (*centroid*) secara acak, lalu menghitung jarak fisik (Euclidean) setiap data ke pusat tersebut. Titik pusat kemudian bergeser secara iteratif hingga berada di tengah-tengah massa data. Keterbatasan utama K-Means adalah asumsinya bahwa setiap kelompok berbentuk bulat sempurna, yang jarang terjadi pada data dunia nyata.

- **GMM (Berbasis Distribusi):** Berbeda dengan K-Means, GMM menggunakan konsep **distribusi normal (Gaussian)**. Model ini tidak hanya melihat "di mana" pusat kelompoknya, tetapi juga "seberapa lebar" dan "bagaimana bentuk" persebaran datanya. GMM mengasumsikan bahwa data yang kita miliki merupakan campuran dari beberapa distribusi normal yang saling tumpang tindih.

2. Mengidentifikasi Kelompok Melalui Fenomena Distribusi Normal

Salah satu poin paling penting dalam kuliah ini adalah cara mendeteksi keberadaan kelompok tersembunyi (*hidden clusters*) melalui grafik frekuensi. Dosen menjelaskan bahwa dalam satu kumpulan data yang tampak menyatu, sebenarnya bisa terdapat beberapa "gunung" distribusi yang berbeda.

Sebagai contoh, dalam distribusi nilai ujian mahasiswa (skala 0-100), jika kita melihat grafik frekuensi dan menemukan tumpukan data yang tinggi di angka 60-70, itu adalah satu kelompok distribusi normal. Namun, jika tiba-tiba muncul puncak kecil lagi di angka 90, mesin harus mampu mengenali bahwa itu adalah kelompok yang berbeda dengan karakteristik yang berbeda pula. Munculnya "kenaikan data" atau puncak frekuensi yang tiba-tiba dalam grafik adalah sinyal kuat adanya kelompok yang berbeda. Hal inilah yang menjadi dasar mengapa model probabilistik jauh lebih akurat dalam mendeteksi anomali atau segmentasi data yang halus.

3. Anatomi Distribusi Gaussian: Peran Mu (μ) dan Sigma (σ)

Untuk mendefinisikan sebuah kelompok secara matematis dalam GMM, dosen menekankan pentingnya memahami dua parameter statistik utama:

- **Mu (μ / Mean):** Ini mewakili nilai rata-rata atau puncak tertinggi dari sebuah kurva. Dalam konteks klasterisasi, μ menunjukkan lokasi pusat dari kelompok tersebut.
- **Sigma (σ / Standard Deviation):** Parameter ini sangat krusial karena menentukan "cakupan" atau lebar dari kelompok tersebut. Jika σ kecil, maka data sangat rapat di sekitar rata-rata (gunung yang lancip). Jika σ besar, data tersebar jauh dari rata-rata (gunung yang landai).

Kelebihan GMM adalah kemampuannya menyesuaikan nilai σ untuk setiap kelompok secara berbeda. Artinya, satu kelompok bisa saja sangat rapat, sementara kelompok lainnya sangat menyebar. Fleksibilitas ini tidak dimiliki oleh K-Means yang cenderung memaksakan semua kelompok memiliki ukuran (variansi) yang sama.

4. Konsep Keanggotaan Lembut (*Soft Membership*) dan Probabilitas

Poin detil berikutnya yang dibahas adalah mengenai bagaimana sebuah data "mengklaim" keanggotaannya dalam suatu kelompok. Dalam model statistik, ini dikenal dengan istilah Responsibility atau Gamma (γ).

Dosen menjelaskan bahwa dalam GMM, sebuah titik data tidak secara kaku dimiliki oleh satu klaster (seperti pada K-Means). Sebaliknya, setiap titik memiliki nilai probabilitas untuk menjadi bagian dari setiap klaster yang ada. Misalnya, sebuah titik data mungkin memiliki peluang 80% masuk ke Klaster A dan 20% masuk ke Klaster B.

Perhitungan ini melibatkan rumus **Fungsi Padat Probabilitas (PDF)**. Komponen dalam rumus ini melibatkan bilangan eksponensial, nilai π (22/7), serta parameter μ dan σ. Intinya, model menghitung seberapa "cocok" sebuah titik data dengan profil distribusi (gunung) dari masing-masing kelompok. Semakin dekat titik tersebut ke puncak kurva sebuah klaster, semakin besar tanggung jawab klaster tersebut terhadap titik tersebut.

5. Algoritma Expectation-Maximization (EM): Mesin di Balik GMM

Karena pada awalnya kita tidak tahu di mana letak puncak gunung (μ) dan seberapa lebar persebarannya (σ), model menggunakan algoritma iteratif yang disebut **Expectation-Maximization (EM)**. Proses ini dijelaskan dosen sebagai sebuah siklus:

1. **Tahap Expectation (E-Step):** Pada tahap ini, model "menebak" atau menghitung probabilitas keanggotaan setiap data berdasarkan parameter (μ dan σ) yang ada saat itu. Ini adalah tahap estimasi tanggung jawab.
2. **Tahap Maximization (M-Step):** Setelah mendapatkan probabilitas keanggotaan, model memperbarui nilai μ dan σ. Model akan menggeser puncak gunung dan mengubah lebarnya sedemikian rupa agar "cocok" dengan data-data yang memiliki probabilitas tinggi di kelompok tersebut.

Proses ini terus berulang hingga mencapai titik **konvergen**, yaitu kondisi di mana nilai μ dan σ sudah tidak berubah secara signifikan lagi.

6. Relevansi dengan Stochastic Gradient Descent (SGD) dan Model Generatif

Dosen secara cerdas mengaitkan materi klasterisasi ini dengan algoritma optimasi yang dibahas pada pertemuan sebelumnya, yaitu **Stochastic Gradient Descent (SGD)**.

- Dalam *Supervised Learning* (Neural Networks), SGD digunakan untuk meminimalkan error atau kesalahan prediksi.
- Dalam konteks GMM dan model probabilistik, logika optimasi serupa digunakan untuk mencapai hasil yang paling optimal (maksimum).

4.pseoducode

// ALGORITMA GAUSSIAN MIXTURE MODEL (GMM) // Berdasarkan konsep Distribusi Normal dan Keanggotaan Lembut (Soft Membership)

BEGIN GMM_ALGORITHM

```
// 1. INISIALISASI PARAMETER
INPUT dataset
SET jumlah_klaster = K (Misal: Z1, Z2, Z3 berdasarkan jumlah
puncak frekuensi)

FOR EACH klaster k IN K:
    INITIALIZE Mu_k (Titik pusat/rata-rata rata secara acak)
    INITIALIZE Sigma_k (Standar deviasi/lebar distribusi)
    INITIALIZE Phi_k (Bobot awal setiap klaster)
END FOR

// 2. PROSES ITERASI (Mencapai Konvergensi)
WHILE NOT konvergen (Parameter Mu dan Sigma tidak berubah
signifikan):

    // --- TAHAP EXPECTATION (E-Step) ---
    // Menghitung "Responsibility" atau probabilitas keanggotaan
    FOR EACH data_point x IN dataset:
        FOR EACH klaster k IN K:
            // Menghitung Gamma (Probabilitas x bagian dari
klaster k)
            CALCULATE Gamma(x,k) = (Phi_k * Gaussian_PDF(x,
Mu_k, Sigma_k)) / Total_Probability_All_Clusters

            // Catatan: Gaussian_PDF menggunakan rumus
distribusi normal
            // yang melibatkan eksponensial, Pi (22/7), Mu, dan
Sigma.
        END FOR
    END FOR

    // --- TAHAP MAXIMIZATION (M-Step) ---
    // Memperbarui parameter berdasarkan hasil probabilitas di
E-Step
    FOR EACH klaster k IN K:
        // 1. Perbarui Jumlah Efektif Data di Klaster k (N_k)
        N_k = SUM(Gamma(x,k) untuk semua x)

        // 2. Perbarui Mu (Rata-rata baru sebagai puncak gunung)
        Mu_k = (1 / N_k) * SUM(Gamma(x,k) * x)

        // 3. Perbarui Sigma (Standar deviasi baru sebagai lebar
gunung)
        Sigma_k = (1 / N_k) * SUM(Gamma(x,k) * (x - Mu_k)^2)
```

```
        // 4. Perbarui Phi (Bobot campuran klaster)
        Phi_k = N_k / Total_Data
    END FOR

    // 3. CEK KONVERGENSI (Optimasi mirip SGD pada Neural
Network)
    IF perubahan_Mu < threshold AND perubahan_Sigma < threshold
THEN
        status = "KONVERGEN"
    END IF

END WHILE

// 4. OUTPUT HASIL
RETURN Final_Mu, Final_Sigma, Cluster_Memberships


END GMM_ALGORITHM
```