JOMLAI: Journal of Machine Learning and Artificial Intelligence
Vol. 4, No. 2, Juni 2025, pp. 88~98
ISSN: 2828-9102 (Print) | 2828-9099 (Online)
DOI: 10.55123/jomlai.v4i2.5960

Penerapan Algoritma Gaussian Mixture Models (GMM) untuk Identifikasi Pola Kesalahan pada Kompilasi

Application of the Gaussian Mixture Models (GMM) Algorithm to Identify Error Patterns in Compilation

Ayu Utari Nasution[1], Eko Prima Ambarita[2], Nurhidayanti[3], Heba Elsisy Fadlia[4], Victor Asido Elyakim P[5]
[1,2,3,4,5]STIKOM Tunas Bangsa, Pematangsiantar, Indonesia

Article Info

Genesis Artikel:

Diterima, 02 April 2025
Direvisi, 09 Mei 2025
Disetujui, 20 Juni 2025

Kata Kunci:

Gaussian Mixture Models
Identifikasi Pola Kesalahan
Kompilasi

ABSTRAK

Dalam pengembangan perangkat lunak, proses kompilasi adalah langkah penting yang mengubah kode sumber menjadi program yang dapat dijalankan. Kesalahan kompilasi ini dapat bervariasi, mulai dari kesalahan sintaksis yang sederhana hingga kesalahan semantik yang lebih kompleks, yang sering kali menuntut waktu dan usaha yang tidak sedikit untuk diidentifikasi dan diperbaiki. Sebagai hasilnya, identifikasi pola kesalahan yang berulang dalam proses kompilasi menjadi penting untuk meningkatkan efisiensi pengembangan perangkat lunak. Penelitian ini bertujuan untuk mengeksplorasi penerapan GMM dalam mengidentifikasi pola kesalahan pada proses kompilasi. hasil penelitian ini menunjukkan bahwa Nilai 0,58 pada *Silhouette Score* menunjukkan bahwa clustering yang dilakukan oleh GMM cukup baik dalam mengidentifikasi pola kesalahan pada kompilasi. Cluster terbagi menjadi 3 yaitu, Cluster 0 mungkin mengindikasikan jenis kesalahan yang lebih cepat terjadi (mungkin terkait dengan kesalahan sintaksis), dengan jumlah baris kode yang lebih sedikit dan frekuensi kesalahan yang lebih rendah. Cluster 1 dapat mewakili kesalahan yang lebih kompleks dan lebih jarang terjadi (misalnya, kesalahan linker atau runtime), dengan jumlah baris kode yang lebih banyak. Cluster 2 mungkin berisi kesalahan dengan pola yang berbeda, seperti durasi kompilasi yang lebih tinggi atau frekuensi kesalahan yang lebih sering terjadi.

Keywords:

Gaussian Mixture Models
Identify Error Patterns
Compilation

ABSTRACT

In software development, the compilation process is a critical step that transforms source code into executable programs. These compilation errors can vary from simple syntax errors to more complex semantic errors, which often require a lot of time and effort to identify and fix. As a result, identifying recurring error patterns in the compilation process is important to improve software development efficiency. This study aims to explore the application of GMM in identifying error patterns in the compilation process. The results of this study indicate that the value of 0.58 on the Silhouette Score indicates that the clustering performed by GMM is quite good at identifying error patterns in compilation. The clusters are divided into 3, namely, Cluster 0 may indicate types of errors that occur more quickly (possibly related to syntax errors), with fewer lines of code and lower error frequency. Cluster 1 may represent more complex and less frequent errors (e.g., linker or runtime errors), with more lines of code. Cluster 2 may contain errors with different patterns, such as higher compilation duration or more frequent error frequency.

Penulis Korespondensi:

Ayu Utari Nasution,
Program Studi Teknik Informatika,
STIKOM Tunas Bangsa,
Email: ayuutarinasution01@gmail.com

1. PENDAHULUAN

Dalam pengembangan perangkat lunak, proses kompilasi adalah langkah penting yang mengubah kode sumber menjadi program yang dapat dijalankan [1]-[2]. Namun, selama proses kompilasi, sering kali muncul berbagai jenis kesalahan yang dapat menghambat jalannya pengembangan[3]. Kesalahan kompilasi ini dapat bervariasi, mulai dari kesalahan sintaksis yang sederhana hingga kesalahan semantik yang lebih kompleks, yang sering kali menuntut waktu dan usaha yang tidak sedikit untuk diidentifikasi dan diperbaiki [4]. Sebagai hasilnya, identifikasi pola kesalahan yang berulang dalam proses kompilasi menjadi penting untuk meningkatkan efisiensi pengembangan perangkat lunak.

Dalam menghadapi tantangan ini, pemanfaatan algoritma *Machine Learning* untuk menganalisis pola kesalahan dapat memberikan wawasan yang lebih dalam dan otomatisasi dalam mengidentifikasi kesalahan yang sering terjadi [5]-[6]. Salah satu pendekatan yang menjanjikan adalah penggunaan *Gaussian Mixture Models (GMM)*, sebuah algoritma berbasis probabilistik yang dapat digunakan untuk mengidentifikasi pola tersembunyi dalam data yang terdistribusi dalam beberapa kelompok [7]. GMM memiliki kemampuan untuk mengelompokkan data ke dalam beberapa distribusi normal yang saling tumpang tindih, yang sangat relevan dengan kasus kesalahan kompilasi yang sering kali bersifat kompleks dan tidak teratur [8][9].

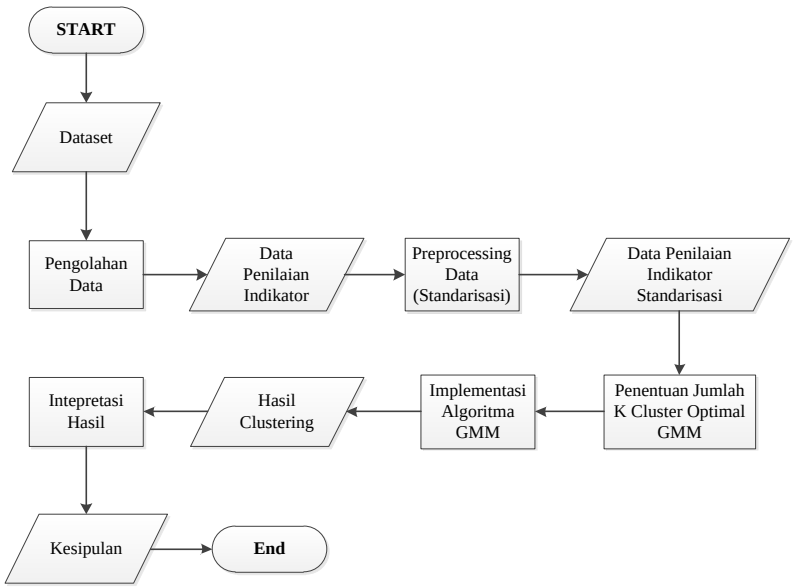
Penelitian terkait penggunaan Algoritma *Gaussian Mixture Models (GMM)* yang berjudul Klasifikasi Tingkat Sengketa Pemilu 2024 di Indonesia Menggunakan Metode *Gaussian Mixture Model*, diperoleh bahwa asil klasifikasi menunjukkan: pada tingkat provinsi, 34% dalam kategori Rendah, 45% Sedang, dan 21% Tinggi. Di tingkat kabupaten/kota, 99,2% berada di kategori Rendah, dengan Demak sebagai satu-satunya dalam kategori Tinggi (0,8%). Di tingkat kecamatan, 99,55% dikategorikan Rendah, dan Karanganyar satu-satunya di kategori Tinggi (0,45%). GMM menunjukkan bahwa meskipun mayoritas wilayah mengalami sengketa rendah, daerah seperti Jawa Tengah dan Demak menunjukkan sengketa tinggi, menyoroti perlunya peningkatan kapasitas peradilan untuk menjaga stabilitas politik [10]. Penelitian lain yang berjudul Prediksi Pergerakan Ikan Di Pesisir Pulau Madura Menggunakan Metode *Gaussian Mixture Model* Dan K-Means Clustering, diperoleh hasil perbandingan metode K-Means Clustering dan Gaussian Mixture Model menunjukkan Gaussian memiliki hasil yang lebih baik dimana banyak kemiripan antara hasil estimasi dan data hasil produksi ikan [11]. Berdasarkan hasil penelitian tersebut dapat diketahui bahwa penggunaan Algoritma GMM dalam klasifikasi sangatlah baik.

Pembaharuan yang diberikan pada penelitian ini adalah penerapan Algoritma GMM untuk mengidentifikasi pola kesalahan pada proses kompilasi. Adapun kriteria yang digunakan adalah Durasi Kompilasi (detik) (X1), Jumlah Baris Kode (X2), Tipe Kesalahan (X3), Kode Kesalahan (X4) dan Frekuensi Kesalahan (X5). Penelitian ini bertujuan untuk mengeksplorasi penerapan GMM dalam mengidentifikasi pola kesalahan pada proses kompilasi. Dengan memanfaatkan log kesalahan kompilasi dari berbagai bahasa pemrograman dan kompilator, penelitian ini akan mencoba untuk mengungkapkan kelompok-kelompok kesalahan yang memiliki pola tertentu, sehingga memungkinkan pengembang untuk fokus pada perbaikan yang lebih efektif dan terarah. Pendekatan ini diharapkan dapat memberikan kontribusi yang signifikan dalam meningkatkan efisiensi proses debugging dan pengembangan perangkat lunak secara keseluruhan.

2. METODE PENELITIAN

2.1 Rancangan Penelitian

Rancangan penelitian digunakan untuk menguraikan dan menyelesaikan masalah dalam penelitian [12]. Rancangan penelitian dapat dilihat pada diagram berikut ini :



Gambar 1. Rancangan Penelitian

a. **Start**
Penelitian dimulai dengan menentukan tujuan dan merancang alur kerja.
b. **Dataset**
Pengumpulan Data: Dataset yang relevan dikumpulkan dari sumber-sumber terpercaya.
c. **Pengolahan Data**
Data Cleaning: Menghapus nilai kosong *(missing values)*, data duplikat, atau *outlier* yang bisa mengganggu analisis.
d. **Data Penilaian Indikator**
Indikator dari dataset dianalisis untuk memberikan penilaian awal.
e. ***Preprocessing Data* (Standarisasi)**
Normalisasi/Standarisasi: Data diubah menjadi skala yang seragam, misalnya menggunakan *z-score* atau *min-max scaling*. Jika satu indikator memiliki rentang nilai 1-100 dan yang lain 0-1, standarisasi dilakukan agar keduanya memiliki kontribusi yang setara dalam analisis.
f. **Data Penilaian Indikator Standarisasi**
Data yang telah distandarisasi diperiksa kembali untuk memastikan keseragaman dan kelengkapan. Hasil ini digunakan untuk langkah-langkah analisis lebih lanjut.
g. **Penentuan Jumlah K Cluster Optimal GMM**
Identifikasi K Optimal Menggunakan metode seperti *Akaike Information Criterion (AIC)* atau *Bayesian Information Criterion (BIC)* untuk menentukan jumlah cluster yang paling sesuai. Tujuan: Menentukan berapa banyak kelompok (cluster) yang paling mewakili pola dalam dataset..
h. **Implementasi Algoritma GMM**
Algoritma diterapkan untuk melakukan clustering berdasarkan distribusi probabilistik. GMM membagi data menjadi beberapa *cluster* berdasarkan parameter seperti *mean, variance*, dan *covariance.*
i. **Hasil Clustering**
Output Clustering: Hasil berupa pembagian data ke dalam cluster-cluster berdasarkan kemiripan atau pola tertentu.
Visualisasi: Hasil clustering biasanya divisualisasikan dalam grafik (misalnya *scatter plot*) untuk mempermudah interpretasi.
j. **Interpretasi Hasil**
Analisis Makna: Setiap cluster dianalisis untuk memahami karakteristik uniknya.
Relevansi Penemuan: Hasil dihubungkan dengan tujuan awal penelitian untuk memastikan temuan sesuai dengan yang diharapkan.
k. **Kesimpulan**
Temuan Utama: Kesimpulan dibuat berdasarkan analisis dan hasil *clustering*.
Rekomendasi: Memberikan saran untuk penelitian lebih lanjut atau aplikasi praktis dari hasil penelitian.
l. **End**
Penelitian selesai dengan dokumen akhir disiapkan.

2.2 Data Penelitian

Data penelitian ini menggunakan data *Synthetic Compilation Error* yang bersumber dari Kagle. Berikut data yang digunakan :

Tabel 1. Data yang Digunakan

Durasi Kompilasi (detik)	Jumlah Baris Kode	Tipe Kesalahan	Kode Kesalahan	Frekuensi Kesalahan
0.5	242	Linker Error	LE-003	3
0.79	170	Linker Error	LE-003	3
0.56	224	Linker Error	LE-002	3
0.89	249	Runtime Error	SE-003	4
0.77	299	Runtime Error	SE-001	1
0.98	187	Linker Error	SE-001	2
0.35	170	Syntax Error	SE-003	1
0.44	171	Runtime Error	SE-002	1
0.55	198	Runtime Error	RE-002	3
0.69	164	Linker Error	LE-002	2
0.27	200	Runtime Error	LE-003	4
Durasi Kompilasi (detik)	**Jumlah Baris Kode**	**Tipe Kesalahan**	**Kode Kesalahan**	**Frekuensi Kesalahan**
0.61	213	Linker Error	LE-001	1
0.24	200	Linker Error	LE-003	3
0.34	167	Linker Error	SE-002	4
0.95	163	Runtime Error	SE-002	2
0.51	239	Linker Error	SE-001	1
0.38	241	Runtime Error	LE-002	3
0.75	157	Syntax Error	LE-002	3
0.34	230	Syntax Error	LE-002	4

0.35	281	Syntax Error	LE-001	2
0.54	203	Syntax Error	SE-003	2
0.65	295	Runtime Error	LE-003	2
0.56	163	Runtime Error	LE-001	3
0.94	189	Syntax Error	RE-002	1
0.51	231	Syntax Error	LE-002	3
0.86	273	Runtime Error	SE-001	1
0.31	164	Runtime Error	LE-002	1
0.26	220	Runtime Error	LE-003	1
0.82	285	Syntax Error	RE-001	4
0.21	230	Syntax Error	RE-001	4
0.78	182	Runtime Error	LE-003	3
0.26	190	Runtime Error	RE-001	4
0.89	221	Syntax Error	SE-003	4
0.46	197	Runtime Error	LE-002	3
0.45	186	Syntax Error	SE-002	3
0.67	184	Syntax Error	SE-003	1
0.51	280	Linker Error	SE-003	1
0.78	291	Syntax Error	LE-002	3
0.23	164	Linker Error	SE-003	2
0.22	212	Syntax Error	SE-003	4
0.71	201	Syntax Error	RE-003	1
0.61	292	Linker Error	SE-001	3
0.36	162	Runtime Error	SE-002	4
0.68	235	Syntax Error	RE-003	4
0.94	194	Linker Error	LE-002	2
0.71	283	Runtime Error	LE-001	4
0.84	193	Linker Error	LE-003	4
0.91	224	Syntax Error	RE-003	4
0.85	241	Runtime Error	LE-002	1
0.45	278	Runtime Error	LE-002	1
0.72	270	Linker Error	SE-001	4
0.48	152	Linker Error	SE-003	3
0.33	286	Syntax Error	LE-002	2
0.38	200	Linker Error	LE-001	2
0.47	208	Runtime Error	LE-001	2
0.38	262	Linker Error	SE-002	2
0.98	201	Linker Error	RE-003	4
0.4	279	Linker Error	SE-001	3
0.44	262	Linker Error	RE-002	4
0.23	262	Syntax Error	SE-002	2
0.6	203	Syntax Error	LE-003	3
0.42	278	Runtime Error	LE-002	3
0.74	202	Runtime Error	LE-003	4
0.56	217	Linker Error	RE-002	3
0.81	272	Syntax Error	SE-002	1
0.78	173	Syntax Error	RE-001	1
0.71	288	Syntax Error	LE-003	3
0.27	246	Linker Error	SE-003	1
0.46	219	Runtime Error	RE-002	4
0.23	297	Syntax Error	SE-003	4

Durasi Kompilasi (detik)	**Jumlah Baris Kode**	**Tipe Kesalahan**	**Kode Kesalahan**	**Frekuensi Kesalahan**
0.74	296	Syntax Error	RE-001	2
0.61	296	Syntax Error	SE-001	4
0.72	201	Linker Error	RE-002	1
0.75	277	Runtime Error	LE-002	3
0.95	278	Linker Error	RE-003	3
0.47	191	Syntax Error	SE-002	3
0.38	293	Syntax Error	RE-003	2
0.99	209	Linker Error	SE-001	1
0.39	197	Linker Error	LE-002	2
0.92	186	Syntax Error	LE-002	4
0.71	158	Runtime Error	RE-003	1
0.48	197	Runtime Error	SE-002	4
0.92	203	Syntax Error	RE-001	4
0.91	265	Linker Error	SE-002	3
0.95	253	Linker Error	SE-003	4

0.66	261	Linker Error	SE-001	2
0.95	242	Linker Error	SE-003	2
0.33	231	Syntax Error	SE-001	2
0.75	217	Syntax Error	SE-003	1
0.6	197	Runtime Error	SE-002	4
0.49	285	Runtime Error	RE-003	3
0.78	277	Runtime Error	SE-001	1
0.63	264	Syntax Error	LE-002	2
0.41	187	Linker Error	RE-001	2
0.51	157	Runtime Error	RE-002	3
0.55	170	Syntax Error	SE-003	2
0.72	260	Runtime Error	RE-003	4
0.89	210	Syntax Error	SE-001	4
0.42	184	Runtime Error	SE-003	4
0.72	166	Syntax Error	SE-002	4
0.95	195	Linker Error	RE-003	1
0.91	273	Runtime Error	RE-001	1
0.96	242	Runtime Error	SE-003	1
0.64	244	Runtime Error	LE-001	3
0.98	265	Linker Error	RE-003	3
0.44	216	Runtime Error	RE-002	4
0.5	203	Syntax Error	SE-001	2
0.94	173	Linker Error	SE-003	2
0.69	235	Linker Error	SE-001	3
0.31	279	Linker Error	LE-001	2
0.79	286	Linker Error	RE-003	3
0.77	188	Runtime Error	RE-001	1
0.44	175	Runtime Error	RE-002	3
0.4	162	Syntax Error	LE-003	4
0.61	206	Runtime Error	LE-001	4
0.84	214	Syntax Error	RE-003	4
0.76	293	Linker Error	LE-002	2
0.91	292	Linker Error	RE-003	4
0.66	181	Linker Error	RE-002	3
0.74	200	Linker Error	RE-001	1
0.32	274	Syntax Error	SE-003	2
0.67	235	Syntax Error	RE-001	1
0.23	219	Syntax Error	LE-002	3
0.49	250	Syntax Error	SE-002	1
0.86	209	Runtime Error	LE-002	4
0.97	252	Linker Error	RE-002	4
0.87	265	Runtime Error	LE-001	2
0.63	204	Syntax Error	SE-001	4
0.78	162	Syntax Error	RE-002	2
Durasi Kompilasi (detik)	**Jumlah Baris Kode**	**Tipe Kesalahan**	**Kode Kesalahan**	**Frekuensi Kesalahan**
0.61	296	Syntax Error	RE-003	1
0.52	168	Linker Error	LE-002	4
0.8	239	Linker Error	RE-001	1
0.35	211	Linker Error	SE-001	3
0.54	289	Runtime Error	LE-001	1
0.25	271	Syntax Error	LE-003	1
0.55	245	Syntax Error	LE-001	2
0.28	238	Linker Error	RE-001	4
0.95	278	Linker Error	SE-001	4
0.61	218	Syntax Error	LE-001	2
0.78	225	Syntax Error	SE-003	2
0.48	235	Runtime Error	SE-002	1
0.47	218	Runtime Error	RE-001	3
0.23	293	Linker Error	RE-003	2
0.81	234	Linker Error	SE-001	3
0.48	250	Runtime Error	SE-002	3
0.24	218	Runtime Error	LE-001	4
0.24	287	Syntax Error	SE-001	2
0.95	245	Runtime Error	SE-003	1
0.63	153	Runtime Error	LE-002	4
0.26	151	Linker Error	SE-003	1
0.89	233	Syntax Error	RE-002	4

0.74	273	Syntax Error	SE-002	1
0.57	250	Linker Error	RE-003	4
0.51	223	Syntax Error	SE-001	3
0.36	290	Syntax Error	SE-002	4
0.28	279	Runtime Error	SE-001	1
0.28	184	Linker Error	RE-003	3
0.26	157	Syntax Error	SE-001	1
0.88	239	Syntax Error	RE-002	2
0.85	254	Runtime Error	SE-003	3
0.29	263	Runtime Error	RE-001	1
0.95	269	Linker Error	RE-003	4
0.53	287	Runtime Error	LE-002	1
0.8	248	Runtime Error	RE-001	4
0.3	245	Linker Error	RE-001	4
0.89	186	Syntax Error	LE-002	4
0.54	162	Linker Error	SE-001	3
0.8	254	Runtime Error	RE-001	2
0.92	211	Linker Error	SE-003	4
0.86	235	Linker Error	RE-002	1
0.92	249	Runtime Error	LE-003	4
0.97	296	Linker Error	RE-002	4
0.48	177	Runtime Error	SE-001	2
0.74	201	Syntax Error	LE-001	3
0.59	293	Runtime Error	SE-001	1
0.27	289	Syntax Error	RE-001	1
0.64	202	Linker Error	RE-003	3
0.83	188	Runtime Error	LE-003	1
0.94	291	Runtime Error	SE-001	3
0.24	224	Linker Error	SE-003	2
0.97	158	Linker Error	SE-003	2
0.87	294	Syntax Error	RE-001	1
0.87	290	Syntax Error	LE-002	4
0.26	176	Runtime Error	RE-002	2
0.32	178	Runtime Error	LE-001	1
0.39	285	Syntax Error	SE-001	4
0.89	235	Linker Error	RE-001	4
0.39	220	Linker Error	RE-001	1
Durasi Kompilasi (detik)	**Jumlah Baris Kode**	**Tipe Kesalahan**	**Kode Kesalahan**	**Frekuensi Kesalahan**
0.67	185	Linker Error	RE-003	2
0.77	168	Syntax Error	SE-003	1
0.5	240	Syntax Error	SE-003	2
0.93	188	Runtime Error	LE-001	2
0.59	290	Linker Error	SE-003	2
0.57	207	Syntax Error	LE-001	4
0.3	210	Linker Error	LE-003	3
0.72	150	Syntax Error	SE-001	3
0.67	262	Runtime Error	SE-002	1
0.89	266	Syntax Error	SE-002	2
0.84	210	Linker Error	LE-002	3
0.91	267	Linker Error	LE-002	1

Kriteria yang digunakan Durasi Kompilasi (detik) (X1), Jumlah Baris Kode (X2), Tipe Kesalahan (X3), Kode Kesalahan (X4), Frekuensi Kesalahan(X5).

2.3 Alur Algoritma GMM

Berikut merupakan tahapan pengerjaan Algoritma GMM [13] :

a. Inisialisasi Parameter
   1) Tetapkan jumlah cluster yang diinginkan (misalnya *K*).
   2) Inisialisasi parameter awal untuk setiap distribusi *Gaussian*:
      a) ***Mean* (μk)**: Posisi pusat cluster.
      b) ***Covariance* (Σk)**: Variasi dan orientasi cluster.
      c) ***Mixing Coefficient* (πk)**: Probabilitas bahwa data berasal dari *cluster* k,

      Dengan :

$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{.....(1)}$$

b. Langkah *Expectation (E-step)*
   1) Hitung ***posterior probability*** atau probabilitas data *xi* menjadi anggota cluster k berdasarkan distribusi *Gaussian*:
   $$\mathbf{r_{ik}} = \frac{\pi_k . N\ (xi\ |\mu k, \sum k)}{\sum_{j=1}^{K} \pi j\ .N\ (xi\ |\mu j, \sum j)} \quad \text{.....(2)}$$
   2) rik: Probabilitas data xi berasal dari *cluster* k.
   3) Fungsi distribusi *Gaussian*
   $$N\ (xi \mid \mu_k, \sum k) \quad \text{.....(3)}$$

c. Langkah *Maximization (M-step)*
   1) Perbarui parameter distribusi *Gaussian* berdasarkan hasil dari *E-step:*
      a) *Mean* (μ_k) **:**
      $$\mu_k = \frac{\sum_{i=1}^{N} r_{ik} . x_i}{\sum_{i=1}^{N} r_{ik}} \quad \text{.....(4)}$$
      b) *Covariance* (Σk) :
      $$\mathbf{\Sigma k} = \frac{\sum_{i=1}^{N} r_{ik} . (x_i - \mu k)^T}{\sum_{i=1}^{N} r_{ik}} \quad \text{.....(5)}$$
      c) *Mixing Coefficient* (πk) :
      $$\boldsymbol{\pi k} = \frac{\sum_{i=1}^{N} rik}{N} \quad \text{.....(6)}$$

d. Evaluasi *Log-Likelihood*
   1) Hitung *log-likelihood* untuk mengevaluasi model:
   $$\mathcal{L} = \sum_{i=1}^{N} log\ (\sum_{k=1}^{K} \pi k\ .\ N\ (xi\ |\mu k, \sum k)) \quad \text{.....(7)}$$
   2) Jika perubahan *log-likelihood* antara iterasi sebelumnya dan saat ini berada di bawah ambang batas tertentu (konvergen), maka proses berhenti.

e. Iterasi hingga Konvergensi
   1) Ulangi langkah E-step dan M-step hingga:
      a) *Log-likelihood* konvergen.
      b) Jumlah iterasi maksimum tercapai.

f. *Output*
   1) Model akhir berisi
      a) Parameter mean (μk)
      b) Parameter covariance (Σk)
      c) Mixing coefficient (πk)
   2) Hasil *clustering* berupa probabilitas setiap data menjadi anggota cluster k.

3. HASIL DAN PEMBAHASAN

Pengolahan data dilakukan sepenuhnya menggunakan python, berikut hasil selengkapnya :

3.1. *Praprocessing Data*

Handling Missing Data: Periksa apakah ada data yang hilang (NaN) dan tangani dengan cara yang sesuai (misalnya, dengan imputasi atau menghapus baris). *Encoding* Kategorikal Data: Karena beberapa kolom seperti Tipe Kesalahan dan Kode Kesalahan merupakan data kategorikal, kamu perlu mengonversinya ke bentuk numerik. Kamu bisa menggunakan teknik seperti *one-hot encoding* atau label *encoding*. Normalisasi atau Standardisasi: Untuk model seperti GMM, sangat penting untuk menormalkan atau menstandarisasi data numerik (misalnya, Durasi Kompilasi (detik), Jumlah Baris Kode, dan Frekuensi Kesalahan) agar masing-masing fitur memiliki skala yang serupa [14]-[15].
Berikut *Script Praprocessing* Data yang digunakan :

```
from sklearn.preprocessing import StandardScaler, LabelEncoder
# Encoding data kategorikal
label_encoder = LabelEncoder()
df_synthetic_data['Tipe Kesalahan'] = label_encoder.fit_transform(df_synthetic_data['Tipe Kesalahan'])
df_synthetic_data['Kode Kesalahan'] = label_encoder.fit_transform(df_synthetic_data['Kode Kesalahan'])
# Normalisasi data numerik
scaler = StandardScaler()
df_synthetic_data[['Durasi Kompilasi (detik)', 'Jumlah Baris Kode', 'Frekuensi Kesalahan']] = scaler.fit_transform(
    df_synthetic_data[['Durasi Kompilasi (detik)', 'Jumlah Baris Kode', 'Frekuensi Kesalahan']]
)
# Menampilkan data setelah prapemrosesan
df_synthetic_data.head()
```

Berikut hasil yang ditampilkan :

[2]:

	Durasi Kompilasi (detik)	Jumlah Baris Kode	Tipe Kesalahan	Kode Kesalahan	Frekuensi Kesalahan
0	-0.502419	0.324483	0	2	0.390567
1	0.740238	-1.343095	0	2	0.390567
2	-0.245318	-0.092412	0	1	0.390567
3	1.168741	0.486609	1	8	1.258493
4	0.654538	1.644649	1	6	-1.345285

Gambar 2. Hasil *Praprocessing* Data

3.2 Penerapan Gaussian Mixture Model (GMM)

Berikut adalah kode untuk menerapkan GMM:

```
from sklearn.mixture import GaussianMixture
from sklearn.metrics import silhouette_score
# Tentukan jumlah cluster (misalnya 3, berdasarkan analisis sebelumnya)
n_components = 3
gmm = GaussianMixture(n_components=n_components, random_state=42)
# Latih model GMM
gmm.fit(df_synthetic_data[['Durasi Kompilasi (detik)', 'Jumlah Baris Kode', 'Frekuensi Kesalahan']])
# Prediksi label cluster
df_synthetic_data['Cluster'] = gmm.predict(df_synthetic_data[['Durasi Kompilasi (detik)', 'Jumlah Baris Kode',
'Frekuensi Kesalahan']])
# Menampilkan hasil clustering
print(df_synthetic_data.head())
```

Berikut hasil yang diperoleh :

```
Durasi Kompilasi (detik)  Jumlah Baris Kode  Tipe Kesalahan  \
0                 -0.502419           0.324483               0
1                  0.740238          -1.343095               0
2                 -0.245318          -0.092412               0
3                  1.168741           0.486609               1
4                  0.654538           1.644649               1

   Kode Kesalahan  Frekuensi Kesalahan  Cluster
0               2             0.390567        0
1               2             0.390567        2
2               1             0.390567        2
3               8             1.258493        0
4               6            -1.345285        1
```

Dari hasil tersebut, beberapa hal yang dapat dipahami adalah sebagai berikut:

a. Durasi Kompilasi (detik) dan Jumlah Baris Kode:

1) Kolom **Durasi Kompilasi (detik)** dan **Jumlah Baris Kode** telah dinormalisasi (standardized), yang berarti nilai-nilai tersebut tidak dalam satuan asli mereka, melainkan dalam satuan standar deviasi dari rata-rata.
   a) Nilai **negatif** pada kolom Durasi Kompilasi menunjukkan bahwa durasi kompilasi tersebut lebih cepat dibandingkan dengan rata-rata durasi untuk dataset tersebut.
   b) Nilai **positif** pada kolom Durasi Kompilasi menunjukkan bahwa durasi kompilasi lebih lama dibandingkan rata-rata.
   c) Hal yang sama berlaku untuk **Jumlah Baris Kode**: nilai negatif berarti jumlah baris kode lebih sedikit dibandingkan rata-rata, sedangkan nilai positif berarti jumlah baris kode lebih banyak.

b. Tipe Kesalahan dan Kode Kesalahan:

1) Kolom **Tipe Kesalahan** telah dienkode ke dalam bentuk numerik (misalnya, 0 dan 1), di mana angka-angka tersebut mungkin mewakili jenis kesalahan yang berbeda, seperti:
   a) **0** mungkin mewakili **Syntax Error**.
   b) **1** mungkin mewakili **Linker Error** atau **Runtime Error**.

2) **Kode Kesalahan** adalah variabel lain yang juga telah dienkode. Angka yang lebih tinggi menunjukkan jenis kesalahan yang lebih kompleks atau lebih spesifik.
Jadi, pada baris pertama (`Tipe Kesalahan = 0`, `Kode Kesalahan = 2`), ini mungkin menunjukkan bahwa kesalahan adalah `Syntax Error` dengan kode kesalahan tertentu (misalnya, kesalahan sintaksis terkait penggunaan operator atau tanda kurung).

c. Frekuensi Kesalahan:

1) **Frekuensi Kesalahan** menunjukkan seberapa sering kesalahan tertentu terjadi dalam kumpulan data.
   a) Angka **positif** pada kolom ini menunjukkan jumlah kejadian kesalahan.
   b) Misalnya, pada baris pertama, frekuensi kesalahan adalah **0.39**, yang menunjukkan bahwa kesalahan tersebut relatif jarang terjadi.

d. Cluster (Hasil Klasifikasi dengan GMM):

1) Kolom **Cluster** menunjukkan hasil pengelompokan menggunakan Gaussian Mixture Models (GMM). Data telah dikelompokkan menjadi beberapa cluster berdasarkan distribusi data yang mendasarinya.
2) **Cluster 0, 1, 2** menunjukkan tiga kelompok yang berbeda dari pola kesalahan yang teridentifikasi berdasarkan fitur-fitur seperti durasi kompilasi, jumlah baris kode, dan frekuensi kesalahan.
   a) Misalnya, pada baris pertama, `Cluster = 0` menunjukkan bahwa data tersebut termasuk dalam kelompok kesalahan yang memiliki pola tertentu dalam durasi kompilasi, jumlah baris kode, dan tipe kesalahan.
   b) Baris kedua, yang termasuk `Cluster = 2`, menunjukkan bahwa kesalahan pada baris ini memiliki pola yang berbeda (misalnya, durasi kompilasi yang lebih lama atau lebih pendek, jumlah baris kode lebih banyak atau lebih sedikit, dan frekuensi kesalahan berbeda).

3.3 Evaluasi Model

Evaluasi hasil *clustering* sangat penting untuk memastikan bahwa model GMM berhasil mengidentifikasi pola yang relevan [16]. Beberapa cara evaluasi yang dapat digunakan adalah *Silhouette Score*, ini memberikan ukuran seberapa baik data terklaster. Nilai mendekati 1 menunjukkan bahwa data terklaster dengan baik, sementara nilai mendekati -1 menunjukkan bahwa data mungkin salah diklaster [17]. Berikut Script yang digunakan :

```
silhouette_avg = silhouette_score(df_synthetic_data[['Durasi Kompilasi (detik)', 'Jumlah Baris Kode', 'Frekuensi Kesalahan']], df_synthetic_data['Cluster'])
print(f"Silhouette Score: {silhouette_avg}")
```

Hasil *Silhouette Score*: 0.58055813634119703

Penafsiran Silhouette Score [18] **:**

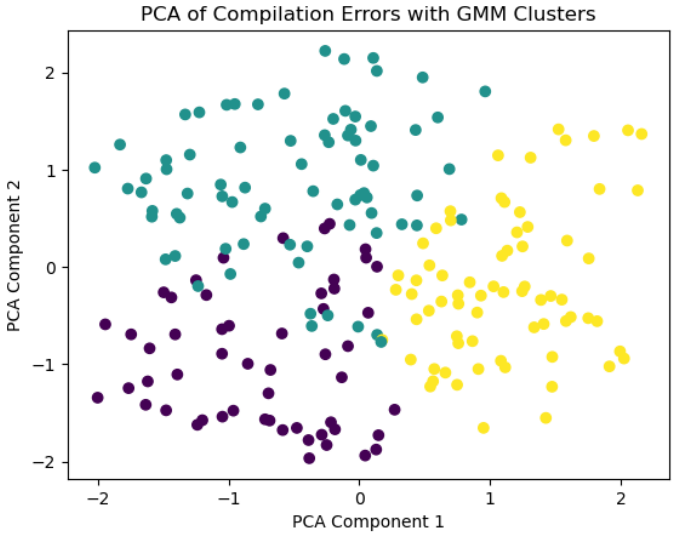
a. **Silhouette Score** berkisar antara **-1 hingga 1**.
   1) Nilai **1** berarti bahwa objek berada pada cluster yang sangat tepat, sangat terpisah dari cluster lainnya.
   2) Nilai **0** berarti bahwa objek berada di antara dua cluster, sehingga sulit untuk menentukan ke cluster mana objek tersebut seharusnya termasuk.
   3) Nilai **-1** menunjukkan bahwa objek mungkin telah dimasukkan ke dalam cluster yang salah

Dari hasil Silhouette Score: 0.58, kita dapat memahami bahwa nilai ini terbilang baik, karena biasanya *Silhouette Score* di atas 0.5 dianggap baik, dan nilai di atas 0.7 sangat baik.

Selanjutnya dilakukan visualisasikan hasil *clustering* dengan menggunakan teknik pengurangan dimensi seperti *PCA (Principal Component Analysis)*, tujuannya untuk melihat bagaimana data dikelompokkan dalam ruang dua dimensi [17]. Berikut Script yang digunakan :

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
# Menggunakan PCA untuk mereduksi dimensi ke 2
pca = PCA(n_components=2)
pca_result = pca.fit_transform(df_synthetic_data[['Durasi Kompilasi (detik)', 'Jumlah Baris Kode', 'Frekuensi Kesalahan']])
# Visualisasi
plt.scatter(pca_result[:, 0], pca_result[:, 1], c=df_synthetic_data['Cluster'], cmap='viridis')
plt.title("PCA of Compilation Errors with GMM Clusters")
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.show()
```

Berikut hasil yang diperoleh :



Gambar 3. Visualisasi Hasil *Clustering*

4. KESIMPULAN

Kesimpulan dari hasil penelitian ini menunjukkan bahwa Nilai 0,58 pada *Silhouette Score* menunjukkan bahwa clustering yang dilakukan oleh GMM cukup baik dalam mengidentifikasi pola kesalahan pada kompilasi. Cluster terbagi menjadi 3 yaitu, Cluster 0 mungkin mengindikasikan jenis kesalahan yang lebih cepat terjadi (mungkin terkait dengan kesalahan sintaksis), dengan jumlah baris kode yang lebih sedikit dan frekuensi kesalahan yang lebih rendah. Cluster 1 dapat mewakili kesalahan yang lebih kompleks dan lebih jarang terjadi (misalnya, kesalahan linker atau runtime), dengan jumlah baris kode yang lebih banyak. Cluster 2 mungkin berisi kesalahan dengan pola yang berbeda, seperti durasi kompilasi yang lebih tinggi atau frekuensi kesalahan yang lebih sering terjadi.

REFERENSI

[1] R. ramadhan Rahmat, "Analisis Kompilasi Hukum Islam Pasal 84 Tentang Nusyuz Istri Perspektif Mazhab Hanafi Dan Mazhab Syafi'I," *Comp. J. Ilm. Perbandingan Maz. dan Huk.*, vol. 2, no. 1, pp. 54–73, 2022, doi: 10.24239/comparativa.v2i1.21.

[2] D. Dalimunthe, K. Kunci, H. Warisan, and K. A. dan KUUHPerdata Pendahuluan, "COMPARASI PENGALIHAN HARTA HIBAH MENJADI HARTA WARISAN PERSPEKTIF KOMPILASI HUKUM ISLAM DAN KITAB UNDANG-UNDANG HUKUM PERDATA Comparation of the transfer of grant assets is inherited from the perspective of KHI and the Civil Code, in the KHI the process o," *J. Huk. Ekon.*, vol. 6, no. 1, pp. 13–26, 2020.

[3] A. D. Setyoko and A. Zahra, "Perbandingan Efisiensi Proses CI/CD Multi-Lingkungan melalui Implementasi Paralel dan Berurutan," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 4, no. 3, pp. 911–925, 2024, doi: 10.57152/malcom.v4i3.1334.

[4] Y. Risyani, S. Japit, and T. Selamat, "Masalah Value Trace untuk pembacaan koding dalam Bahasa Pemrograman C," *J. Minfo Polgan*, vol. 13, no. 1, pp. 600–605, 2024, doi: 10.33395/jmp.v13i1.13753.

[5] W. Prastiwinarti *et al.*, "Perancangan Pemanfaatan Machine Learning untuk Deteksi Cacat Kemasan Produk," *Sniv Semin. Nas. Inov. Vokasi*, vol. 2, no. 1, pp. 97–102, 2023.

[6] F. W. Atmojo *et al.*, "ANALISIS PEMANFAATAN MACHINE LEARNING GUNA PREDIKSI INDEKS," vol. 9, no. 2, 2024.

[7] P. Chyan, "Segmentasi Kulit Manusia Dengan Ekstraksi Fitur Warna Dan Algoritma GMM-EM," *J. Pendidik. Teknol. Inf.*, no. April, pp. 151–156, 2022.

[8] N. N. Alyarahma, G. Kholijah, and C. Sormin, "Pengelompokan Provinsi di Indonesia Menggunakan Gaussian Mixture Model Berdasarkan Indikator Kemiskinan," vol. 6, no. 2, pp. 158–167, 2024, doi: 10.31605/jomta.v6i2.4032.

[9] R. Adi, "Deteksi Api pada Video dengan Gaussian Mixture Model untuk Deteksi Gerakan dan Segmentasi Warna Api dalam Ruang Warna YCbCr (Telah disetujui Tim Penguji …," *Publ. Tugas Akhir S-1 PSTI FT-UNRAM*, 2020.

[10] Adek Maulidya, Khairul, Zulham Sitorus, Andysah Putera Utama Siahaan, and Muhammad Iqbal, "Analysis Of Increasing Student Service Satisfaction Using K-Means Clustering Algorithm and Gaussian Mixture Models (GMM)," *Int. J. Comput. Sci. Math. Eng.*, vol. 3, no. 1, pp. 29–35, 2024, doi: 10.61306/ijecom.v3i1.62.

[11] C. N. Prabiantissa and G. E. Yuliastuti, "Prediksi Pergerakan Ikan Di Pesisir Pulau Madura Menggunakan Metode Gaussian Mixture Model Dan K-Means Clustering," *J. Teknol. Inf. dan Terap.*, vol. 8, no. 2, pp. 121–128, 2021, doi: 10.25047/jtit.v8i2.244.

[12] J. Riyono, S. D. Puspa, and C. E. Pujiastuti, "Simulasi Clustering Provinsi di Indonesia dalam Penyebaran Covid-19 Berdasarkan Indikator Kesehatan Masyarakat Menggunakan Algoritma Gaussian Mixture Model," *MAJAMATH J. Mat. dan Pendidik. Mat.*, vol. 5, no. 1, pp. 43–60, 2022.

[13] I. A. Nafiudin, R. T. Hidayat, A. M. Putri, and A. R. Maulana, "Deteksi Jumlah Kendaraan Dengan Algoritma Gaussian Mixture Model

Di Area Jalan Raya," *Method. J. Tek. Inform. dan Sist. Inf.*, vol. 7, no. 1, pp. 37–44, 2021, doi: 10.46880/mtk.v7i1.258.
[14] A. Fitri and A. Fachrur, "Klasifikasi Tingkat Sengketa Pemilu 2024 di Indonesia Menggunakan Metode Gaussian Mixture Model," no. 2022, pp. 404–416, 2024.
[15] D. Y. Faidah, A. M. Hudzaifa, N. Theresia, and C. E. Widiantoro, "Optimalisasi Strategi Pengelompokkan Potensi Padi Sebagai Solusi Efektif Kelangkaan Beras Di Jawa Barat," *J. Lebesgue J. Ilm. Pendidik. Mat. Mat. dan Stat.*, vol. 5, no. 1, pp. 529–537, 2024, doi: 10.46306/lb.v5i1.592.
[16] N. Hendrastuty, "Penerapan Data Mining Menggunakan Algoritma K-Means Clustering Dalam Evaluasi Hasil Pembelajaran Siswa," *J. Ilm. Inform. Dan Ilmu Komput.*, vol. 3, no. 1, pp. 46–56, 2024, [Online]. Available: https://doi.org/10.58602/jima-ilkom.v3i1.26
[17] G. Vardakas, I. Papakostas, and A. Likas, "Deep Clustering Using the Soft Silhouette Score: Towards Compact and Well-Separated Clusters," 2024, [Online]. Available: http://arxiv.org/abs/2402.00608
[18] R. Saputra and I. Purnamai, "OPTIMIZING K-MEANS ALGORITHM WITH ELBOW AND SILHOUETTE METHODS FOR NATIONAL EXAM SCORE DATA CLUSTERING," *J. Ilmu Komput. Ruru*, vol. 1, pp. 1–6, 2024.
[19] MDPI (2022). *An error overbounding method based on a GMM with uncertainty estimation for dual-frequency augmentation systems*. *Remote Sens*, 14(5), 1111.
[20] Reddit (2025). "I built a GMM from scratch… clustering SDSS data." *r/learnmachinelearning*, Jan 16 2025. > "I implemented a GMM… E-step, M-step, and convergence criteria.".