Penggunaan Dekomposisi Matriks SVD dalam Sistem Rekomendasi untuk Platform Streaming Film

Muhammad Farrel Wibowo 13523153[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13523153@mahasiswa.itb.ac.id, [2]faawibowo@gmail.com

***Abstract*—sistem rekomendasi merupakan sistem yang esensial dalam sebuah platform streaming film untuk menyaring jumlah film yang banyak dan terpersonalisasi oleh preferensi pengguna. Makalah ini membahas implementasi SVD pada kumpulan data film untuk memprediksi peringkat pengguna dan memberikan rekomendasi. Langkah pra-pemrosesan, seperti normalisasi data dan pengisian nilai hilang, diikuti dengan dekomposisi matriks menggunakan SVD. Evaluasi model menggunakan metrik RMSE dan MAE menunjukkan keefektifan SVD dalam meningkatkan akurasi rekomendasi.**

***Keywords*—Dekomposisi Matriks, Singular Value Decomposition (SVD), Platform Streaming Film, Sistem Rekomendasi.**

I. PENDAHULUAN

Platform streaming film merupakan sebuah layanan digital yang memungkinkan pengguna untuk mengakses berbagai jenis konten seperti film, serial TV, kartun, atau acara TV melalui internet secara langsung. Platform ini menggunakan teknologi streaming, yaitu data dikirimkan secara real-time sehingga pengguna dapat secara langsung menikmati konten dengan koneksi internet tanpa harus mengunduhnya terlebih dahulu. Platform streaming film seperti Netflix, Disney+, dan Amazon Prime Video telah menjadi bagian integral dari hiburan modern. Netflix yang merupakan layanan streaming terbesar di dunia memiliki lebih dari 269,6 juta pelanggan berbayar global per 31 Maret 2024. Sama halnya dengan platform streaming lainnya seperti Disney+, Amazon Prime Video, Max dan Paramount+ yang memiliki rata rata pelanggan lebih dari 100 juta pelanggan. Di Indonesia Pada 2021, jumlah pelanggan Subscription Video on Demand (SVoD) di Indonesia mencapai 11,5 juta.

Berdasarkan data yang telah dipaparkan dibuktikan bahwa terdapat banyak orang yang menggunakan platform streaming film. Dengan jutaan pengguna yang dimiliki platform streaming film memiliki tantangan yang kompleks dalam menentukan rekomendasi film bagi setiap pengguna yang tiap penggunanya memiliki preferensi yang berbeda beda. Oleh karena itu diperlukan sebuah sistem rekomendasi untuk menyaring informasi dan membantu pengguna menemukan konten yang sesuai dengan minat mereka.

Namun, tantangan seperti volume data yang besar, keberagaman selera pengguna, dan masalah cold-start sering kali membatasi kemampuan sistem rekomendasi tradisional. Salah satu pendekatan yang efektif untuk mengatasi tantangan ini adalah dengan menggunakan Singular Value Decomposition (SVD), metode berbasis faktor laten yang memanfaatkan pola tersembunyi dalam data.

Makalah ini berfokus pada implementasi SVD pada dataset film yang tersedia secara publik, seperti kaggle. Penelitian ini tidak mencakup permasalahan spesifik cold-start pada pengguna baru.

II. LANDASAN TEORI

A. Dekomposisi Matrix

Dekomposisi Matriks adalah proses memfaktorkan sebuah matriks menjadi hasil kali dari beberapa matriks lain yang lebih sederhana. Proses ini bertujuan untuk mempermudah operasi matematika, seperti menyelesaikan sistem persamaan linear, menghitung invers, atau menganalisis data. Secara matematis, jika terdapat matriks *A*, maka dekomposisi matriks berarti menuliskan *A* sebagai hasil kali:

$$A = P_1 \times P_2 \times \ldots \times P_k$$

Di mana $P_1, P_2, \ldots, P_k$ adalah matriks-matriks yang membentuk faktor-faktor penyusun *A*.

Terdapat beberapa metode umum yang digunakan untuk mendekomposisi matriks. Di antaranya:

1. Dekomposisi LU

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & l_{32} & 1 & 0 \\ l_{41} & l_{42} & l_{43} & 1 \end{bmatrix} \times \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix}$$

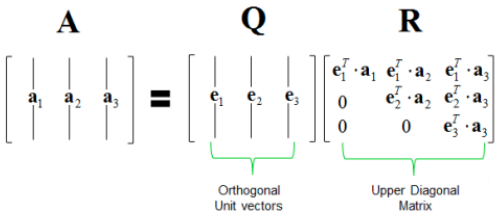
A L U

Gambar 1. Dekomposisi LU
https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-23-Dekomposisi-LU-2023.pdf

Dekomposisi LU merupakan metode dekomposisi yang memfaktorkan matriks menjadi dua matriks, yaitu matriks segitiga bawah (*Lower triangular matrix*) dan matriks segitiga atas (*Upper triangular matrix*). Dekomposisi matriks LU diperkenalkan oleh Alan Turing pada tahun 1948, yang menciptakan mesin turing.

Dekomposisi LU adalah teknik dasar yang digunakan dalam aljabar linear untuk memecahkan sistem persamaan linear dan penghitungan determinan.

2. Dekomposisi QR

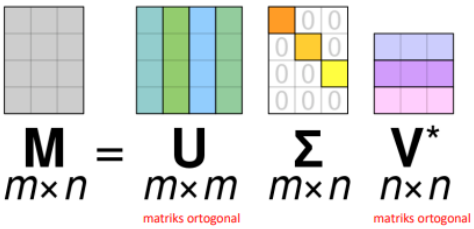


Gambar 2. Dekomposisi QR
https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-21-Singular-value-decomposition-Bagian1-2023.pdf

Dekomposisi QR adalah metode yang memfaktorkan sebuah matriks berukuran $m \times n$ menjadi hasil kali dari dua matriks, yaitu matriks ortogonal Q dan matriks segitiga atas R. Proses ini dapat dinyatakan secara matematis sebagai $A = Q \cdot R$, di mana Q adalah matriks ortonormal dan R adalah matriks segitiga atas.

Matriks ortonormal Q memiliki sifat khusus, yaitu $QQ^{\mathrm{T}} = I$, di mana I adalah matriks identitas. Matriks ortonormal ini juga memiliki kolom-kolom yang merupakan vektor-vektor ortogonal dengan norma satu. Sebagai tambahan, jika setiap vektor dalam matriks tersebut adalah vektor satuan, maka matriks tersebut disebut ortonormal.

3. Dekomposisi Nilai Singular



Gambar 3. SVD
https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-21-Singular-value-decomposition-Bagian1-2023.pdf

Singular Value Decomposition (SVD) adalah metode dalam aljabar linear yang memecah matriks menjadi tiga matriks: U, Σ, dan V^T. Matriks U adalah matriks ortogonal yang kolom-kolomnya disebut vektor singular kiri. Matriks Σ adalah matriks diagonal yang berisi nilai singular, yang menunjukkan besarnya kontribusi masing-masing vektor singular dalam representasi matriks asli. Matriks V^T adalah transpos dari matriks ortogonal V yang kolom-kolomnya disebut vektor singular kanan. Secara matematis, ini ditulis sebagai $A=U\Sigma V^T$, di mana A adalah matriks asli. SVD digunakan dalam berbagai aplikasi seperti reduksi dimensi, kompresi data, dan pemrosesan sinyal. Dalam konteks reduksi dimensi, SVD dapat membantu mengidentifikasi pola dalam data dengan mengurangi jumlah variabel tanpa kehilangan informasi penting.

B. Root Mean Square Error dan Mean Absolute Error

RMSE (Root-Mean-Squared Error) dan MAE (Mean Absolute Error) adalah dua metrik standar yang sering digunakan dalam evaluasi model. Untuk sampel dengan n pengamatan $\mathbf{y}(y_i)$ dan n prediksi model (y_i), MAE dan RMSE didefinisikan sebagai berikut:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \quad (1)$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|. \quad (2)$$

Gambar 4. Rumus RMSE dan MAE
https://gmd.copernicus.org/articles/15/5481/2022/gmd-15-5481-2022.html

Sesuai dengan namanya, RMSE adalah akar kuadrat dari MSE (Mean Squared Error). Proses pengambilan akar kuadrat ini tidak memengaruhi peringkat relatif antar model, tetapi menghasilkan metrik dengan satuan yang sama dengan **y**, yang secara praktis merepresentasikan kesalahan rata-rata atau "standar" untuk distribusi kesalahan yang bersifat normal. MSE dan MAE merupakan bentuk rata-rata dari norma L2 dan L1, yang juga dikenal sebagai jarak Euclidean dan Manhattan.

RMSE telah digunakan sebagai metrik statistik standar untuk mengukur kinerja model dalam studi penelitian meteorologi, kualitas udara, dan iklim. MAE adalah ukuran lain yang berguna dan banyak digunakan dalam evaluasi model. Meskipun keduanya telah digunakan untuk menilai kinerja model selama bertahun-tahun, belum ada konsensus mengenai metrik yang paling tepat untuk kesalahan model (chai & draxler, 2014).

III. METODOLOGI

A. Metode Pengolahan

A.1. Pengolahan data awal

Dataset yang didapatkan dari situs Kaggle.com memiliki atribut user_id, rating, dan movie_id dalam bentuk csv, oleh karena itu dataset tersebut perlu di konversi terlebih dahulu menjadi bentuk matriks, apabila dimisalkan dataset dikonversi menjadi matrix R maka baris dari matrix atau R_i akan mewakili user atau pengguna, dan jumlah baris akan sesuai dengan jumlah user yang ada pada dataset. Sedangkan R_j akan mewakili film, dan jumlah kolom akan sesuai dengan jumlah film yang ada pada dataset. Kemudian R_{ij} merupakan nilai rating yang diberikan user i terhadap film j apabila pengguna atau user tidak memberikan rating terhadap film j maka R_{ij} akan direpresentasikan sebagai nilai 0.

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1m} \\ r_{21} & r_{22} & \cdots & r_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{nm} \end{bmatrix}$$

Dengan:

- r_{ij} merupakan nilai rating yang diberikan user *i* terhadap film *j*
- r_{ij} = 0 jika user *i* belum memberikan rating terhadap film *j*

A.2. Pengisian Nilai Nol dengan Rata-Rata Kolom

Setelah dataset dikonversi menjadi matriks R maka akan dilakukan pengisian nilai nol dengan rata rata setiap kolom.

$$\text{mean}_j = \frac{\sum_{i=1}^{n} r_{ij} \cdot \delta(r_{ij})}{\sum_{i=1}^{n} \delta(r_{ij})}$$

Dengan:

$$\delta(r_{ij}) = \begin{cases} 1, & \text{jika } r_{ij} \neq 0 \\ 0, & \text{jika } r_{ij} = 0 \end{cases}$$

Mengisi nilai nol dalam R

$$R_{\text{filled}}[i,j] = \begin{cases} r_{ij}, & \text{jika } r_{ij} \neq 0 \\ \text{mean}_j, & \text{jika } r_{ij} = 0 \end{cases}$$

Langkah ini bertujuan untuk mengatasi masalah data yang hilang (missing values) dengan menggantinya menggunakan rata-rata rating pada kolom terkait. Kolom merepresentasikan item, sehingga pengisian ini didasarkan pada rata-rata rating item oleh pengguna lain. Langkah ini memastikan bahwa matriks tidak memiliki elemen kosong, yang penting untuk operasi selanjutnya seperti normalisasi dan dekomposisi SVD. Pengisian ini juga mencerminkan asumsi bahwa jika pengguna belum memberikan rating, maka nilai prediksi awalnya mendekati rata-rata item tersebut.

A.3. Normalisasi Matriks

Setelah pengisian nilai nol, Langkah selanjutnya adalah normalisasi. Matriks dinormalisasi dengan cara mengurangi rata-rata rating pengguna (rata-rata baris) dari setiap elemen pada baris tersebut.

$$\text{mean}_i = \frac{1}{m} \sum_{j=1}^{m} R_{\text{filled}}[i,j]$$

$$R_{\text{normalized}}[i,j] = R_{\text{filled}}[i,j] - \text{mean}_i$$

Proses ini menghasilkan matriks yang lebih berfokus pada pola deviasi daripada nilai absolut. Normalisasi menghilangkan bias pengguna terhadap rating mereka. Misalnya, jika seorang pengguna cenderung memberikan rating tinggi (5 untuk semua item), normalisasi mengurangi efek ini, sehingga model lebih fokus pada pola preferensi relatif.

A.4. Dekomposisi Matriks Nilai Singular

Langkah ini menggunakan metode dekomposisi matriks nilai singular untuk mendekomposisi matriks yang telah di normalisasi menjadi tiga komponen yaitu matriks *U* yaitu matriks orthogonal yang mewakili pengguna dalam dimensi laten, Σ yaitu matriks diagonal yang berisi nilai singular yang menunjukkan bobot pentingnya dimensi laten, dan V^T yaitu matriks orthogonal yang mewakili film dalam dimensi laten. Digunakan parameter *k* untuk menentukan jumlah dimensi laten yang dipertahankan.

$$R_{\text{normalized}} = U \cdot \Sigma \cdot V^T$$

dengan:

- $U \in \mathbb{R}^{n \times k}$: Matriks pengguna
- $\Sigma \in \mathbb{R}^{k \times k}$: Matriks diagonal berisi singular values terbesar
- $V^T \in \mathbb{R}^{k \times m}$: Matriks film

Dimensi laten ini adalah representasi abstrak dari hubungan mendasar antara pengguna dan item. SVD memungkinkan model untuk memadatkan informasi kompleks dalam data ke dalam dimensi yang lebih kecil, mengidentifikasi pola-pola utama yang tersembunyi dalam data. Ini juga membantu menangani noise atau outlier.

A.5. Rekontruksi Matriks

Setelah dilakukannya Langkah dekomposisi dengan menggunakan SVD, Matriks direkonstruksi dengan:

$$R_{predicted} = U \cdot \Sigma \cdot V^T + \textit{rata-rata baris}$$

Langkah ini mengembalikan prediksi rating untuk semua elemen dalam matriks, termasuk elemen-elemen yang awalnya kosong. Proses rekonstruksi menghasilkan matriks rating yang diprediksi, di mana setiap elemen menunjukkan estimasi rating pengguna terhadap item berdasarkan pola laten yang ditemukan. Penambahan rata-rata baris memastikan bahwa prediksi kembali ke skala asli data.

A.6. Clipping Nilai Prediksi

Hasil prediksi terkadang bisa melampaui skala yang diinginkan (misalnya, kurang dari 0 atau lebih dari 5). Oleh karena itu, clipping digunakan untuk membatasi nilai-nilai prediksi agar tetap berada dalam rentang valid [0, 5].

$$R_{predicted}[i,j] = max(0,min(5,R_{predicted}[i,j]))$$

Clipping memastikan bahwa hasil prediksi tetap realistis dan konsisten dengan skala rating awal.

C. Eksperimen menggunakan Pemrograman

B.1. Deskripsi Program

Pada penelitian ini digunakan Bahasa pemrograman python, dengan bantuan library seperti NumPy, SciPy, Matplotlib, dan Seaborn. Bahasa pemrograman Python dipilih karena memiliki library yang banyak dan mudah untuk digunakan. Dengan detail penggunaan library sebagai berikut:

1. NumPy (numpy):
   - Library untuk komputasi numerik, digunakan untuk operasi matriks, seperti pengisian elemen nol, perhitungan rata-rata, dan normalisasi.
   - Fungsi utama yang digunakan:
     - np.array(): Membuat array/matriks.
     - np.mean(): Menghitung rata-rata.
     - np.where(): Mengisi elemen tertentu dalam array.
     - np.dot(): Operasi perkalian matriks.
     - np.clip(): Membatasi nilai dalam rentang tertentu.
2. SciPy (scipy.sparse.linalg):
   - Library untuk operasi matematika lanjutan, seperti dekomposisi matriks.
   - Fungsi utama yang digunakan:
     - svds(): Melakukan dekomposisi SVD pada matriks yang memiliki elemen nol.
3. Matplotlib (matplotlib.pyplot):
   - Library untuk membuat visualisasi grafik.
   - Fungsi utama yang digunakan:
     - plt.figure(): Mengatur ukuran plot.
     - plt.title(), plt.xlabel(), plt.ylabel(): Memberikan label dan judul pada plot.
4. Seaborn (seaborn):
   - Library untuk visualisasi data yang lebih estetik, digunakan untuk membuat heatmap.
   - Fungsi utama yang digunakan:
     - sns.heatmap(): Membuat heatmap dari matriks prediksi.

B.2. Tahapan Program

B2.1. Import Library

```
import numpy as np
from scipy.sparse.linalg import svds
import matplotlib.pyplot as plt
import seaborn as sns
```

Gambar 2.1 program import library

Diawal program dilakukan import terhadap library yang akan digunakan dan telah dijelaskan sebelumnya seperti numpy, scipy.sparse.linalg.svds, matplotlib.pyplot, seaborn

B2.2. Deklarasi Matriks Awal

```
R = np.array([
    [0,   1,    0,  1 , 0 , 2],
    [3.5, 0,    5,  5 , 5 , 0],
    [0,   2,    1,  5 , 5 , 5],
    [1,   0,    0,  1 , 2 , 5]
], dtype=float)
```

Gambar 2.2 program deklarasi matriks

Pada eksperimen penelitian kali ini digunakan matriks yang sudah dideklarasikan di awal sebagai asumsi bahwa dataset sudah di konversi menjadi sebuah matriks dari dataset. Sama seperti yang sudah di jelaskan pada metode pengolahan baris pada matriks mewakili user atau pengguna yang berarti jumlah baris sama dengan jumlah user. Sedangkan kolom pada matriks mewakili film dan jumlah kolom akan sesuai dengan jumlah film yang ada pada dataset. Elemen bernilai 0 menunjukkan film yang belum dirating oleh pengguna.

B.2.3. Mengisi Elemen Nol dengan Rata Rata Kolom

```
col_mean = np.mean(R, axis=0, where=R != 0)
R_filled = np.where(R == 0, col_mean, R)
```

Gambar 2.3 program mengisi elemen nol

col_mean merupakan hasil rata rata per kolom. Elemen nol diabaikan saat menghitung rata-rata kolom, sehingga nilai rata-rata hanya mempertimbangkan item yang sudah dirating. R_filled merupakan matriks yang elemen 0 nya diisi dengan rata rata kolomnya.

B.2.4. Normalisasi Matriks

```
R_mean = np.mean(R_filled, axis=1)  # Rata-rata baris
R_normalized = R_filled - R_mean[:, np.newaxis]
```

Gambar 2.4 program normalisasi matriks

R_mean: Menghitung rata-rata rating dari setiap pengguna. Ini dilakukan untuk menghilangkan bias pengguna dalam memberikan rating.

R_normalized: Mengurangi rata-rata baris (pengguna) dari setiap elemen di matriks. Normalisasi ini menghasilkan matriks yang lebih seimbang dan siap untuk dekomposisi.

B.2.5. Rank Efektif dan Nilai k Maksimal

```
rank_effective = np.linalg.matrix_rank(R_normalized)
k_max = min(rank_effective, R_normalized.shape[0] - 1, R_normalized.shape[1] - 1)
```

Gambar 2.5 program k maks

rank_effective Menggunakan fungsi np.linalg.matrix_rank, rank efektif dari matriks R_normalized dihitung. Rank efektif menunjukkan jumlah dimensi independen dalam matriks. k_max merupakan Nilai k, yang merupakan jumlah komponen singular, tidak boleh melebihi rank efektif atau dimensi minimum dari matriks. Hal ini memastikan hasil

dekomposisi tetap valid dan efisien.

B.2.6. Dekomposisi SVD

```
U, Sigma, Vt = svds(R_normalized, k_max)
Sigma = np.diag(Sigma)
```

Gambar 2.6 program dekomposisi SVD

svds: Melakukan dekomposisi nilai singular (SVD) pada matriks R_normalized. Matriks dipecah menjadi tiga komponen:

- U: Matriks singular kiri (m×k), merepresentasikan hubungan antara pengguna dan dimensi laten.
- Σ: Matriks diagonal (k×k) yang berisi singular values. Singular values menunjukkan pentingnya setiap dimensi laten.
- V^T: Matriks singular kanan (k×n), merepresentasikan hubungan antara item dan dimensi laten.

np.diag: Mengubah singular values menjadi matriks diagonal.

B.2.7. Rekonstruksi Matriks

```
R_predicted = np.dot(U, np.dot(Sigma, Vt)) + R_mean[:, np.newaxis]
R_predicted = np.clip(R_predicted, 0, 5)
```

Gambar 2.7 program rekonstruksi matriks

Matriks asli R_normalized direkonstruksi kembali menggunakan komponen U, Σ, dan V^T. $R_{predicted}$ = $U \cdot \Sigma \cdot V^T + Rmean$.

np.clip berfungsi untuk Membatasi nilai prediksi agar tetap dalam rentang valid (0 hingga 5). Ini memastikan hasil prediksi realistis.

B.2.7. Menampilkan Hasil

```
# Hasil prediksi
print("Matriks Prediksi:\n", R_predicted)
```

Gambar 2.8 program menampilkan hasil

Matriks prediksi ditampilkan, menunjukkan estimasi rating untuk elemen-elemen yang sebelumnya kosong di matriks asli.

B.2.7. Membuat Heatmap

```
# Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(
            R_predicted,
            annot=True, cmap='YlGnBu', cbar=True,
            xticklabels=["Film 1", "Film 2", "Film 3","Film 4", "Film 5", "Film 6"],
            yticklabels=["User 1", "User 2", "User 3", "User 4"]
        )
plt.title("Predicted Ratings")
plt.xlabel("Films")
plt.ylabel("Users")
plt.show()
```

Gambar 2.9 program heatmap

Heatmap dibuat agar hasil prediksi dapat tervisualisasi dan lebih mudah untuk dianalisis untuk penelitian ini.

plt.figure: Menentukan ukuran heatmap agar mudah dibaca.

sns.heatmap: Membuat heatmap dari matriks prediksi R_predicted.

- Nilai pada setiap sel divisualisasikan dengan warna dan anotasi angka.
- Skema warna hijau-kuning-biru digunakan untuk menonjolkan perbedaan nilai.
- Kolom diberi label sesuai dengan item (misalnya, "Film 1", "Film 2").
- Baris diberi label sesuai dengan pengguna (misalnya, "User 1", "User 2").

IV. PERCOBAAN DAN PEMBAHASAN

Pada percobaan yang dilakukan, digunakan dataset sederhana dengan jumlah 4 user dan 6 film dengan detail dataset sebagai berikut:

User_Id	Rating	Movie_ID
2	3.5	1
4	1	1
1	1	2
3	2	2
2	5	3
3	1	3
1	1	4
2	5	4
3	5	4
4	1	4
2	5	5
3	5	5
4	2	5
1	2	6
3	5	6
4	5	6

Tabel 4.1 Dataset Eksperimen

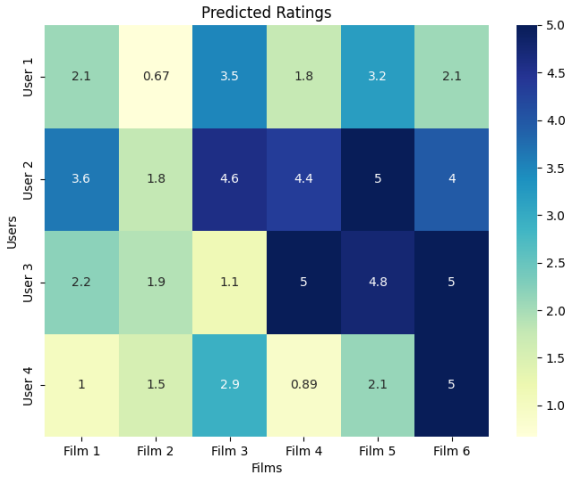
Setelah didapatkan dataset dalam bentuk table seperti diatas selanjutnya adalah mengonversi table tersebut ke dalam bentuk matriks dan dideklarasikan di awal program.

User_ID\Film_ID	1	2	3	4	5	6
1	0	1	0	1	0	2
2	3.5	0	5	5	5	0
3	0	2	1	5	5	5
4	1	0	0	1	1	5

Tabel 4.2 Matriks Dataset eksperimen

A. *Hasil Eksperimen Dengan Dimensi Laten maksimal*

Dalam dataset yang dimiliki memiliki dimensi laten maksimum 3. Berikut hasil gambar eksperimen yang dihasilkan oleh program.

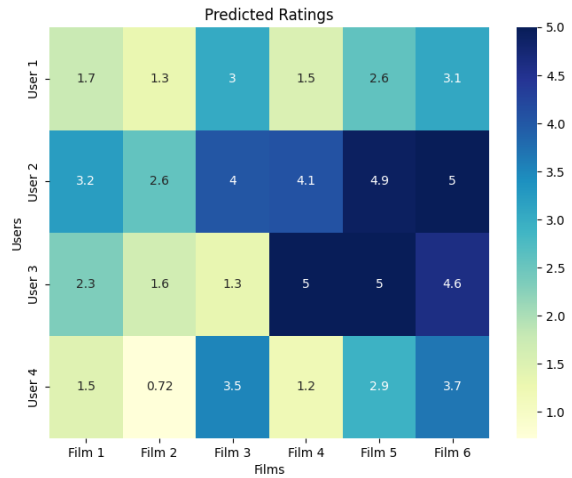


Gambar 4.1 Hasil Percobaan 1

Pada gambar ditemukan perbedaan rating yang sudah ada di awal dengan yang dihasilkan oleh program, hal ini disebabkan oleh rating yang dihasilkan oleh program merupakan Langkah kombinasi dari normalisasi, dekomposisi, rekonstruksi dan penyesuaian nilai prediksi. Algoritma program tidak dirancang untuk sepenuhnya mempertahankan nilai rating awal, namun algoritma lebih dirancang untuk memprediksi rating yang hilang berdasarkan pola data.

B. Hasil Eksperimen Dengan Dimensi Laten lebih rendah

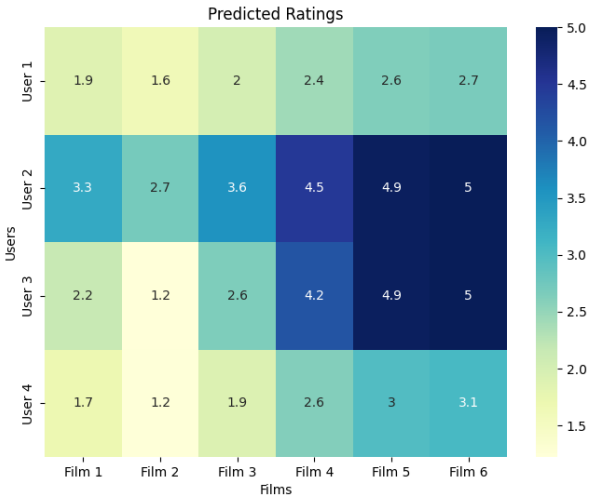
Pada eksperimen kedua digunakan dataset yang sama namun dengan jumlah laten yang lebih rendah yaitu 2



Gambar 4.2 Hasil Percobaan 2

Berdasarkan gambar 4.2 didapatkan perbedaan hasil dengan gambar 4.1. Sehingga didapati bahwa perbedaan laten dapat menyebabkan perbedaan hasil.

C. Hasil Eksperimen dengan dimensi Laten 1



Gambar 4.3 Hasil Percobaan 3

D. Evaluasi dengan Root Mean Squared Error

```python
import numpy as np

# Matriks asli (ground truth)
R = np.array([
    [0,   1,    0,  1 , 0 , 2],
    [3.5, 0,    5,  5 , 5 , 0],
    [0,   2,    1,  5 , 5 , 5],
    [1,   0,    0,  1 , 2 , 5]
], dtype=float)

# Matriks prediksi
laten_1 = np.array([[1.87837308, 1.62143268, 2.00872892, 2.42463119, 2.63708597, 2.67974817],
 [3.27532637, 2.71102224, 3.56161976, 4.47504312, 4.94164592, 5.        ],
 [2.16591525, 1.22439965, 2.64358266, 4.16758769, 4.94609309, 5.        ],
 [1.68434855, 1.24387514, 1.9078178,  2.6207998,  2.98501141, 3.0581473 ]])

laten_2 = np.array([
    [1.74472929, 1.29244198, 3.03162108, 1.52618477, 2.59262887, 3.062394],
    [3.2144528,  2.56116985, 4.0275382,  4.06580864, 4.92139611, 5.0],
    [2.33781023, 1.64755323, 1.32792087, 5.0,        5.0,        4.6102559],
    [1.47286675, 0.72327077, 3.5264719,  1.19907224, 2.91466122, 3.66365712]
])

laten_3 = np.array([
    [2.08524795, 0.66891529, 3.48169973, 1.7577819,  3.19993457, 2.05642056],
    [3.64015713, 1.78165892, 4.59021054, 4.35534314, 5.0,        3.95200215],
    [2.20009853, 1.89971832, 1.14590121, 5.0,        4.75766955, 5.0],
    [1.02320961, 1.5466419,  2.93214006, 0.89324671, 2.11271003, 4.9920517]
])


# Fungsi RMSE
def calculate_rmse(predicted, original):
    mask = original > 0  # Gunakan hanya nilai yang diketahui
    return np.sqrt(np.mean((predicted[mask] - original[mask]) ** 2))

# Hitung RMSE untuk kedua laten
rmse_laten_1 = calculate_rmse(laten_1, R)
rmse_laten_2 = calculate_rmse(laten_2, R)
rmse_laten_3 = calculate_rmse(laten_3, R)

print("RMSE Laten 1:", rmse_laten_1)
print("RMSE Laten 2:", rmse_laten_2)
print("RMSE Laten 3:", rmse_laten_3)
```

Gambar 4.4 program Evaluasi

Hasil matriks prediksi dari setiap laten dimasukkan pada program dan dihitung dengan menggunakan RMSE dan didapatkan hasil output sebagai berikut:

- RMSE Laten 1: 1.0341455242468194
- RMSE Laten 2: 0.6454884974785422
- RMSE Laten 3: 0.296327201459178

V. KESIMPULAN

Berdasarkan percobaan yang telah dilakukan bahwa metode SVD dapat memprediksi rating film yang belum di rating oleh pengguna. Didapatkan juga bahwa perbedaan laten yang digunakan dapat menyebabkan hasil prediksi yang berbeda. Setelah dilakukannya percobaan

dengan beberapa laten yang berbeda dan didapatkan untuk algoritma yang dibuat semakin besar laten yang digunakan semakin akurat juga prediksinya berdasarkan RMSE yang semakin kecil dengan semakin membesarnya dimensi laten. Namun percobaan ini masih belum dicoba dari segi efektifitasnya untuk dataset yang lebih besar baik dari kompleksitas waktu, dan aspek lain yang harus dipertimbangkan seperti preferensi genre film dari setiap pengguna, Namun secara garis besar dibuktikan bahwa SVD dapat digunakan untuk memprediksi nilai kosong pada sebuah matriks dalam hal ini adalah rating film.

REFERENSI

[1] Hodson, T. O.: Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not, Geosci. Model Dev., 15, 5481–5487, https://doi.org/10.5194/gmd-15-5481-2022, 2022, diakses 30 Desember 2024.

[2] Chai, T. and Draxler, R. R.: Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature, Geosci. Model Dev., 7, 1247–1250, https://doi.org/10.5194/gmd-7-1247-2014, 2014. a, b, c, d, e, f, g, diakses 30 Desember 2024.

[3] R. Munir, "Singular Value Decomposition (SVD) Bagian 1," Bahan kuliah IF2123 Aljabar Linier dan Geometri, Program Studi Teknik Informatika, STEI-ITB, 2023. diakses 23 Desember 2024.

[4] R. Munir, "Singular Value Decomposition (SVD) Bagian 2," Bahan kuliah IF2123 Aljabar Linier dan Geometri, Program Studi Teknik Informatika, STEI-ITB, 2023. diakses 26 Desember 2024.

[5] Qilong Ba, Xiaoyong Li and Zhongying Bai, "Clustering collaborative filtering recommendation system based on SVD algorithm," 2013 IEEE 4th International Conference on Software Engineering and Service Science, Beijing, 2013, pp. 963-967, doi: 10.1109/ICSESS.2013.6615466. diakses tanggal 25 Desember 2024.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 27 Desember 2024

Muhammad Farrel Wibowo 13523153