

**SISTEM REKOMENDASI FILM BERBASIS *COLLABORATIVE*
FILTERING MENGGUNAKAN ALGORITMA *MATRIX*
FACTORIZATION (STUDI KASUS PLATFROM VIU)**

SKRIPSI

DISUSUN OLEH

FARAH ZHAFIRA MUNTHE

NPM. 2109020072



UMSU

Unggul | Cerdas | Terpercaya

**PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA**

MEDAN

2025

**SISTEM REKOMENDASI FILM BERBASIS *COLLABORATIVE*
FILTERING MENGGUNAKAN ALGORITMA *MATRIX*
FACTORIZATION (STUDI KASUS PLATFROM VIU)**

SKRIPSI

**Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana
Komputer (S.Kom) dalam Program Studi Teknologi Informasi pada
Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas
Muhammadiyah Sumatera Utara**

FARAH ZHAFIRA MUNTHER

NPM. 2109020072

**PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA**

MEDAN

2025

LEMBAR PENGESAHAN

Judul Skripsi : SISTEM REKOMENDASI FILM BERBASIS
COLLABORATIVE FILTERING MENGGUNAKAN
ALGORITMA *MATRIX FACTORIZATION* (STUDI KASUS
PLATFROM VIU)

Nama Mahasiswa : FARAH ZHAFIRA MUNTHE

NPM : 2109020072

Program Studi : TEKNOLOGI INFORMASI

Menyetujui
Komisi Pembimbing



(Farid Akbar Siregar, S.Kom., M.Kom.)
NIDN. 0104019401

Ketua Program Studi



(Fatma Sari Nutagalung, S.Kom., M.Kom.)
NIDN. 0017019301

Dekan



(Dr. Al-Khowarizmi, S.Kom., M.Kom.)
NIDN. 0127099201

PERNYATAAN ORISINALITAS

SISTEM REKOMENDASI FILM BERBASIS *COLLABORATIVE* *FILTERING* MENGGUNAKAN ALGORITMA *MATRIX* *FACTORIZATION* (STUDI KASUS PLATFROM VIU)

SKRIPSI

Saya menyatakan bahwa karya tulis ini adalah hasil karya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing disebutkan sumbernya.

Medan, 2025

Yang membuat pernyataan



Farah Zhafira Munthe

NPM. 2109020072

PERNYATAAN PERSETUJUAN PUBLIKASI
KARYA ILMIAH UNTUK KEPENTINGAN
AKADEMIS

Sebagai sivitas akademika Universitas Muhammadiyah Sumatera Utara, saya bertanda tangan dibawah ini:

Nama : Farah Zhafira Munthe
NPM : 2109020072
Program Studi : Teknologi Informasi
Karya Ilmiah : Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Muhammadiyah Sumatera Utara Hak Bedas Royalti Non-Eksekutif (*Non-Exclusive Royalty free Right*) atas penelitian skripsi saya yang berjudul:

SISTEM REKOMENDASI FILM BERBASIS *COLLABORATIVE*
FILTERING* MENGGUNAKAN ALGORITMA *MATRIX
***FACTORIZATION* (STUDI KASUS PLATFROM VIU)**

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksekutif ini, Universitas Muhammadiyah Sumatera Utara berhak menyimpan, mengalih media, memformat, mengelola dalam bentuk database, merawat dan mempublikasikan Skripsi saya ini tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis dan sebagai pemegang dan atau sebagai pemilik hak cipta.

Demikian pernyataan ini dibuat dengan sebenarnya.

Medan, 2025
Yang membuat pernyataan


Farah Zhafira Munthe
NPM. 2109020072

RIWAYAT HIDUP

DATA PRIBADI

Nama Lengkap : Farah Zhafira Munthe
Tempat dan Tanggal Lahir : Medan, 31 Januari 2004
Alamat Rumah : Jl. Kutilang No. 4 Sei Siskambing B
Telepon/Faks/HP : 081269003959
E-mail : farahzhafira3101@gmail.com
Instansi Tempat Kerja : -
Alamat Kantor : -

DATA PENDIDIKAN

SD	: SD Harapan 1 Medan	TAMAT: 2015
SMP	: SMP Harapan 1 Medan	TAMAT: 2018
SMA	: SMA Harapan 1 Medan	TAMAT: 2021

KATA PENGANTAR



Puji syukur Penulis ucapkan kepada Allah SWT atas rahmat-Nya, sehingga Penulis dapat menyelesaikan skripsi ini berjudul “**Sistem Rekomendasi Film Berbasis Collaborative Filtering Menggunakan Algoritma Matrix Factorization (Studi Kasus Platform Viu)**”. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Komputer pada Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Muhammadiyah Sumatera Utara. Penulis tentunya berterima kasih kepada berbagai pihak dalam dukungan serta doa dalam penyelesaian skripsi. Penulis juga mengucapkan terima kasih kepada:

1. Bapak Prof. Dr. Agussani, M.AP., selaku Rektor Universitas Muhammadiyah Sumatera Utara.
2. Bapak Prof. Dr. Muhammad Arifin, S.H., M.Hum, selaku Wakil Rektor I Universitas Muhammadiyah Sumatera Utara.
3. Bapak Prof. Dr. Akrim, M.Pd, selaku Wakil Rektor II Universitas Muhammadiyah Sumatera Utara.
4. Bapak Assoc. Prof. Dr. Rudianto, S.Sos., M.Si selaku Wakil Rektor III Universitas Muhammadiyah Sumatera Utara.
5. Bapak Dr. Al-Khowarizmi, S.Kom., M.Kom. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi.
6. Ibu Fatma Sari Hutagalung, S.Kom., M.Kom, selaku Ketua Program Studi Teknologi Informasi.
7. Bapak Muhammad Basri, S.Si., M.Kom, selaku Sekretaris Program Studi Teknologi Informasi.
8. Bapak Farid Akbar Siregar, S.Kom., M.Kom, selaku Dosen Pembimbing Penulis yang membantu dalam membimbing penulisan skripsi Penulis dan yang telah mempermudah Penulis dalam pembimbingan sehingga skripsi Penulis dapat selesai dengan tepat waktu.

9. Seluruh staff akademik Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Muhammadiyah Sumatera Utara, atas fasilitas dan dukungan administratif yang diberikan selama masa perkuliahan hingga penyusunan skripsi.
10. Kedua orang tua yakni Dedek Rizal Munthe, S.Sos., M.AP, selaku ayah Penulis dan Titri Suhandayani, S.Sos, selaku ibu Penulis yang selalu mendoakan dan memberikan dukungan dalam menyelesaikan skripsi ini.
11. Abang kandung Penulis, yakni Fachrurozy Mikraj Munthe, S.Ak. yang mendukung dan mendoakan Penulis dalam menyelesaikan skripsi ini.
12. Adik kandung Penulis, yakni Farel Ramadhan Munthe yang mendukung dan mendoakan Penulis dalam menyelesaikan skripsi ini.
13. Kakak saudara Penulis, yakni Hazrina Syahirah Putri, S.I.Kom., M.I.Kom. dan Humairah Hafshah Putri, S.Psi. yang mendukung dan mendoakan Penulis dalam menyelesaikan skripsi ini.
14. Sahabat Penulis dari SMP, yakni Azka Nabila Nasution dan Amelia Andari Wicaksana Nasution yang selalu menemani Penulis dalam susah senang, memiliki cara untuk menghibur Penulis dan memberikan dukungan dalam menyelesaikan skripsi ini.
15. Teman seperjuangan Anak Digital Forensik, yakni Dea Ikwa Cahya Syafitri, Julia Namira Nasution, Aqilah Tahara, Laila Salsabila, Umi Khoirunnisa, dan Suci Indah Ismana yang menemani Penulis selama mengerjakan dan mendukung sampai selesai menyelesaikan skripsi ini.
16. Teman – teman sekelas yakni kelas B1 Pagi Teknologi Informasi dan kelompok KKN Desa Tanah Merah yang sudah menemani masa – masa perkuliahan Penulis sehingga hari yang dilalui menjadi lebih menyenangkan.

Medan, 2025
Penulis



Farah Zhafira Munthe

SISTEM REKOMENDASI FILM BERBASIS *COLLABORATIVE FILTERING* MENGUNAKAN ALGORITMA *MATRIX FACTORIZATION* (STUDI KASUS PLATFROM VIU)

ABSTRAK

Meningkatnya jumlah konten film dan drama di platform streaming digital seperti VIU menyebabkan pengguna kesulitan dalam menemukan tontonan yang sesuai dengan preferensi mereka. Untuk mengatasi masalah tersebut, penelitian ini mengembangkan sistem rekomendasi film berbasis *Collaborative Filtering* menggunakan pendekatan *Matrix Factorization* dengan algoritma *Singular Value Decomposition* (SVD). Sistem ini memanfaatkan data rating pengguna terhadap film untuk memprediksi kemungkinan ketertarikan pengguna terhadap film lain yang belum ditonton. Dalam penelitian ini, Data yang digunakan meliputi 200 data pengguna, 30 data film, dan 10 data prediksi rating hasil dari model. Sistem memungkinkan pengguna memasukkan ID untuk melihat daftar film yang direkomendasikan secara khusus. Selain itu, evaluasi dilakukan menggunakan metrik klasifikasi (*accuracy* bernilai 0.8, *precision* bernilai 1.0, *recall* bernilai 0.6667, dan *F1-score* bernilai 0.8) serta *Mean Absolute Error* (MAE) untuk mengetahui tingkat akurasi sistem bernilai 0.4551. Hasilnya menunjukkan bahwa sistem dapat memberikan rekomendasi yang cukup sesuai dengan preferensi pengguna.

Kata Kunci: Sistem Rekomendasi, *Collaborative Filtering*, *Matrix Factorization*, *Singular Value Decomposition*, *Flask*, *Mean Absolute Error*, *Confusion Matrix*

MOVIE RECOMMENDATION SYSTEM BASED ON COLLABORATIVE FILTERING USING MATRIX FACTORIZATION ALGORITHM (CASE STUDY VIU PLATFORM)

ABSTRACT

The increasing number of movies and dramas on digital streaming platforms such as VIU makes it difficult for users to find content that suits their preferences. To address this issue, this study develops a movie recommendation system based on Collaborative Filtering using the Matrix Factorization approach with the Singular Value Decomposition (SVD) algorithm. The system utilizes user rating data to predict the likelihood of a user's interest in movies they have not yet watched. The data used in this study includes 200 user data, 30 movie data, and 10 predicted rating results from the model. The system allows users to enter their ID to view a personalized list of recommended movies. In addition, evaluation is carried out using classification metrics (accuracy = 0.8, precision = 1.0, recall = 0.6667, and F1-score = 0.8) as well as Mean Absolute Error (MAE), which resulted in a value of 0.4551. The results show that the system is able to provide recommendations that are reasonably aligned with user preferences.

Keywords: Recommendation System, Collaborative Filtering, Matrix Factorization, Singular Value Decomposition, Flask, Mean Absolute Error, Confusion Matrix

DAFTAR ISI

LEMBAR PENGESAHAN	iii
PENYATAAN ORISINALITAS	iv
PENYATAAN PERSETUJUAN PUBLIKASI.....	v
RIWAYAT HIDUP	vi
KATA PENGANTAR.....	vii
ABSTRAK	ix
ABSTRACT	x
DAFTAR ISI.....	xi
DAFTAR TABEL	xiv
DAFTAR GAMBAR.....	xv
BAB I. PENDAHULUAN.....	1
1.1 Latar Belakang Masalah	1
1.2 Rumusan Masalah	3
1.3 Batasan Masalah.....	4
1.4 Tujuan Penelitian.....	4
1.5 Manfaat Penelitian.....	5
BAB II. LANDASAN TEORI	6
2.1 Sistem Rekomendasi	6
2.2 <i>Collaborative Filtering</i>	7
2.2.1 <i>Metode Collaborative Filtering</i>	7
2.3 <i>Matrix Factorization</i>	8
2.3.1 <i>Singular Value Decomposition (SVD)</i>	10
2.4 Teknik <i>Clipping</i> dalam <i>Prediksi Rating</i>	12
2.5 Fitur Laten	13
2.6 <i>Mean Absolute Error (MAE)</i>	14
2.7 <i>Confusion Matrix</i>	14
2.8 Dataset.....	16
2.9 <i>Website</i>	16
3.0 <i>Flowchart</i>	16
3.1 <i>Unified Modelling Language (UML)</i>	17
3.1.1 <i>Use Case Diagram</i>	18

3.1.2 Activity Diagram	19
3.2 Platform.....	20
3.3 Visual Studio Code	20
3.4 Penelitian Terdahulu.....	20
BAB III. METODOLOGI PENELITIAN	22
3.1 Desain Penelitian	22
3.1.1 Identifikasi Masalah	22
3.1.2 Studi Literatur	23
3.1.3 Pengumpulan Data	23
3.1.4 Preprocessing Data.....	23
3.1.5 Implementasi	23
3.1.6 Evaluasi	24
3.2 Perancangan Sistem.....	24
3.3 Flowchart Algoritma.....	26
3.3.1 Flowchart Collaborative Filtering.....	26
3.3.2 Flowchart Matrix Factorization.....	27
3.4 Use Case Diagram	28
3.5 Activity Diagram	29
3.6 Interface UI/UX.....	30
3.6.1 Halaman Login	30
3.6.2 Homepage.....	31
BAB IV. HASIL DAN PEMBAHASAN	33
4.1 Hasil Implementasi.....	33
4.2 Analisis Kepadatan dan Kekosongan Data Rating.....	33
4.3 Perhitungan Matriks Singular Value Decomposition (SVD) dan Matrix Factorization (MF).....	34
4.3.1 Dekomposisi Matriks Rating Menggunakan Singular Value Decomposition	34
4.3.2 Transformasi SVD ke Matrix Factorization (MF)	39
4.4 Penyesuaian Nilai Prediksi (Clipping)	41
4.5 Evaluasi Akurasi Prediksi Menggunakan Mean Absolute Error (MAE)	42

4.6 Perbandingan dengan Metode Tradisional Berbasis Memori	43
4.7 Melakukan Evaluasi dengan <i>Confusion Matrix</i>	44
4.8 Hasil Pengujian Sistem pada Algoritma <i>Matrix Factorization</i> dengan Teknik <i>Singular Value Decomposition</i> (SVD).....	45
4.8.1 Halaman Awal <i>User</i>	45
4.8.2 Halaman Rekomendasi <i>User</i>	46
4.8.3 Halaman Film yang Sudah Ditonton	47
4.8.4 Halaman <i>Login Admin</i>	47
4.8.5 Halaman <i>Dashboard Admin</i>	48
4.8.6 Halaman Kelola Film	48
BAB V. KESIMPULAN DAN SARAN	49
5.1 Kesimpulan.....	49
5.2 Saran.....	50
DAFTAR PUSTAKA.....	51
LAMPIRAN.....	54

DAFTAR TABEL

	HALAMAN
Tabel 2.4 <i>Confusion Matrix</i>	15
Tabel 3.1 Simbol <i>Flochart</i>	17
Tabel 3.2 Simbol <i>Use Case Diagram</i>	18
Tabel 3.3 Simbol <i>Activity Diagram</i>	19
Tabel 3.4 Penelitian Terdahulu	20
Tabel 4.3 Data <i>Sample User - Rating</i>	35
Tabel 4.4 Nilai Prediksi <i>Rating</i>	41
Tabel 4.5 Data Nilai <i>Clipping</i>	42
Tabel 4.6 Perbandingan Metode <i>Matrix Factorization</i> dengan <i>User-Based</i> ...	43
Tabel 4.7 Evaluasi dengan <i>Confusion Matrix</i>	44

DAFTAR GAMBAR

	HALAMAN
Gambar 2.1 Teknik Sistem Rekomendasi.....	6
Gambar 2.2 Metode <i>Collaborative Filtering</i>	7
Gambar 2.3 <i>User-based</i> dan <i>Item-based filtering</i>	8
Gambar 2.4 <i>Singular Value Decomposition Matrix</i>	11
Gambar 3.2 Alur Perancangan Sistem	24
Gambar 3.3 Alur <i>Collaborative Filtering</i>	26
Gambar 3.4 Alur <i>Matrix Factorization</i>	27
Gambar 3.5 <i>User Case Diagram</i>	28
Gambar 3.6 <i>Activity Diagram</i>	29
Gambar 3.7 Halaman <i>Login</i>	30
Gambar 3.8 <i>Homepage</i>	31
Gambar 4.7 Grafik <i>Confusion Matrix</i>	44
Gambar 4.8 Halaman Awal <i>User</i>	45
Gambar 4.9 Halaman Rekomendasi <i>User</i>	45
Gambar 5.0 Halaman Film yang Sudah Ditonton.....	47
Gambar 5.1 Halaman <i>Login Admin</i>	47
Gambar 5.2 Halaman <i>Dashboard</i>	48
Gambar 5.3 Halaman Kelola Film	48

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Dunia perfilman sudah sangat populer di masyarakat, bahkan film dianggap sebagai hiburan yang wajib bagi semua orang, terutama di Indonesia. Pada awal abad ke-20, Belanda memperkenalkan film, yang merupakan hiburan *visual*. Karena mereka masih berada di bawah naungan penjajahan, hiburan masyarakat saat itu hanyalah wayang, seni tradisional (Fajar Yulia Fahmi & Nugroho Bayu Aji, 2022).

Seiring berkembangnya internet dan *website* yang mendukung pertukaran informasi menjadi lebih cepat, hal itu juga berdampak pada industri perfilman. Banyak bermunculan platform yang memungkinkan orang menonton film di rumah tanpa harus pergi ke bioskop. Dengan *platform streaming* film ini, Anda dapat menonton film kapan saja dan di mana saja melalui laptop, komputer, tablet, atau *smartphone* yang terhubung ke internet, tanpa perlu mendaftar atau membayar. Meskipun demikian, hal ini sering menyebabkan perselisihan karena banyak kasus pembajakan film, di mana film diputar di *platform* film tersebut tidak memiliki izin edar dan melanggar hak cipta (Ellysinta et al., 2020).

Munculnya fenomena ini, pemerintah mengeluarkan surat edaran Kominfo No.3 pada tahun 2016 yang mengatur penyedia layanan aplikasi dan/atau konten online yang harus mematuhi peraturan perundang-undangan dan juga diawasi oleh KPI (Komisi Penyiaran Indonesia). *Platform* film streaming adalah layanan yang menawarkan akses eksklusif untuk koleksi perfilman melalui kolaborasi dengan pembuat film. *Platform* ini sangat populer di Indonesia untuk mengakses tayangan video atau film baik lokal maupun

internasional, seperti Netflix, Viu, Video, MAXStream, WeTV, Disney+ Hotstar, HOOQ, Iflix, dan Vision+ (Wibowo Okta, 2018).

Setiap orang pasti memiliki selera yang berbeda. Seseorang mungkin menyukai film berdasarkan *genre*, rumah produksi, atau aktornya. Ini adalah masalah seseorang saat memilih film yang memenuhi ekspektasi. Seseorang pasti tidak akan memiliki cukup waktu untuk membaca sinopsis atau trailer setiap film, mengingat banyaknya jenis film. bahkan jika ada film baru yang judulnya belum diketahui. Oleh karena itu, keinginan seseorang adalah mendapatkan rekomendasi film yang ideal dari berbagai aspek dengan waktu yang singkat.

Untuk mengatasi masalah ini, sistem rekomendasi film menjadi alat penting yang dapat membantu pengguna menemukan konten yang relevan berdasarkan minat dan perilaku mereka. Sistem rekomendasi dapat memberikan rekomendasi untuk item melalui penyaringan dan pemilihan informasi dengan menggunakan preferensi pengguna, seperti *profil*, peringkat, dan perilaku. Ini membantu individu atau komunitas menemukan konten atau item yang menarik yang kemungkinan besar akan dipilih, digunakan, atau dibeli (Prayogo et al., 2021).

Salah satu pendekatan yang banyak digunakan dalam pengembangan sistem rekomendasi adalah *Collaborative Filtering* (CF). Metode ini bekerja dengan menganalisis pola interaksi antara pengguna dan item (dalam hal ini, film) untuk memberikan rekomendasi yang dipersonalisasi. CF terbagi menjadi dua kategori utama: *user-based* dan *item-based*. Namun, kedua pendekatan ini memiliki keterbatasan, terutama dalam menangani data yang bersifat *sparse* (nilai kosong) (Ryana Agustian & Prasetyo Nugroho, 2020).

Untuk mengatasi keterbatasan tersebut, algoritma *Matrix Factorization* (MF) diperkenalkan sebagai solusi yang lebih efisien dan akurat. MF bekerja dengan mendekomposisi matriks interaksi pengguna-item menjadi dua matriks dengan dimensi lebih rendah yang mewakili faktor laten dari pengguna dan item. Pendekatan ini memungkinkan sistem untuk menangkap hubungan tersembunyi antara pengguna dan item, sehingga dapat memberikan rekomendasi yang lebih relevan (Koren et al., 2009).

Platform film yang akan peneliti ambil untuk penelitian ini adalah platform film Viu. Platform streaming video over-the-top (OTT) yang menawarkan berbagai konten hiburan Asia, termasuk drama Korea, drama Jepang, drama Cina, dan film Indonesia. Pengguna dapat menikmati konten secara gratis dengan iklan atau berlangganan Viu Premium untuk menikmati konten bebas iklan dan fitur tambahan lainnya. Dengan bertujuan penelitian ini dirancang dengan judul “SISTEM REKOMENDASI FILM BERBASIS *COLLABORATIVE FILTERING* MENGGUNAKAN ALGORITMA *MATRIX FACTORIZATION* (STUDI KASUS PLATFORM VIU)”. Diharapkan sistem yang dirancang dapat menemukan rekomendasi film yang sesuai dengan preferensi mereka berdasarkan pola interaksi dan memberikan pengalaman menonton yang lebih personal dan relevan bagi pengguna Viu.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka rumusan masalah dalam penelitian ini adalah bagaimana meningkatkan akurasi rekomendasi agar lebih sesuai dengan preferensi pengguna di *platform* Viu. Selain itu, perlu diteliti bagaimana metode ini dapat mengatasi masalah *sparsity* yang sering terjadi dalam sistem rekomendasi. Efektivitas *Matrix Factorization* dengan teknik *Singular Value Decomposition* (SVD)

juga perlu dibandingkan dengan metode tradisional berbasis memori untuk menentukan sejauh mana pendekatan ini mampu meningkatkan akurasi rekomendasi.

1.3 Batasan Masalah

Agar penelitian ini lebih terarah dan fokus, terdapat beberapa batasan masalah yang diterapkan sebagai berikut:

1. Studi kasus ini hanya menggunakan *platform* Viu sebagai subjek penelitian.
2. Sistem yang direkomendasikan hanya menggunakan metode *Collaborative Filtering* berbasis *Model-Based* dengan algoritma *Matrix Factorization* dan teknik *Singular Value Decomposition* (SVD).
3. Pada tujuan penelitian ini, data yang digunakan berasal dari interaksi pengguna dengan film di *platform* Viu, seperti rating atau riwayat tontonan.
4. Metrik evaluasi klasifikasi seperti *Accuracy*, *Precision*, *Recall*, dan *F1-Score* dan evaluasi regresi seperti MAE.
5. Film yang diambil tidak lebih sebanyak 30 film.
6. Sistem rekomendasi film ini berbasis web

1.4 Tujuan Penelitian

Penelitian ini bertujuan untuk :

1. Mengembangkan sistem rekomendasi film berbasis *Collaborative Filtering* menggunakan algoritma *Matrix Factorization* dengan teknik *Singular Value Decomposition* (SVD) pada platform Viu.
2. Meningkatkan akurasi rekomendasi dengan menganalisis pola preferensi pengguna berdasarkan data rating yang tersedia.
3. Mengatasi permasalahan *sparsity* dalam sistem rekomendasi film.

4. Mengevaluasi efektivitas algoritma *Matrix Factorization* dengan teknik *Singular Value Decomposition* (SVD) dibandingkan metode tradisional berbasis memori dalam memberikan rekomendasi yang lebih relevan dan personal bagi pengguna Viu.

1.5 Manfaat Penelitian

Penelitian ini diharapkan bermanfaat sebagai berikut :

Bagi Penulis

Menambah wawasan dan pengetahuan khususnya mengenai sistem rekomendasi film berbasis Collaborative Filtering dengan algoritma *Matrix Factorization* dengan teknik *Singular Value Decomposition* (SVD).

Bagi Pengguna *platform* film

1. Membantu pengguna platform Viu dalam menemukan film yang sesuai dengan preferensi mereka secara lebih akurat dan efisien.
2. Sistem ini dapat memperkenalkan pengguna dengan film-film yang mungkin belum pernah mereka ketahui sebelumnya, memperluas cakrawala mereka dan membuka peluang untuk menemukan film-film baru yang menarik.

Bagi *Platform*

1. Sistem rekomendasi yang efektif dapat meningkatkan retensi pengguna dengan memberikan pengalaman yang lebih baik dan membuat pengguna lebih tertarik untuk menggunakan platform.
2. Sistem rekomendasi dapat mendorong pengguna untuk lebih aktif di platform dengan memberikan rekomendasi yang menarik dan relevan, sehingga meningkatkan engagement dan interaksi pengguna.

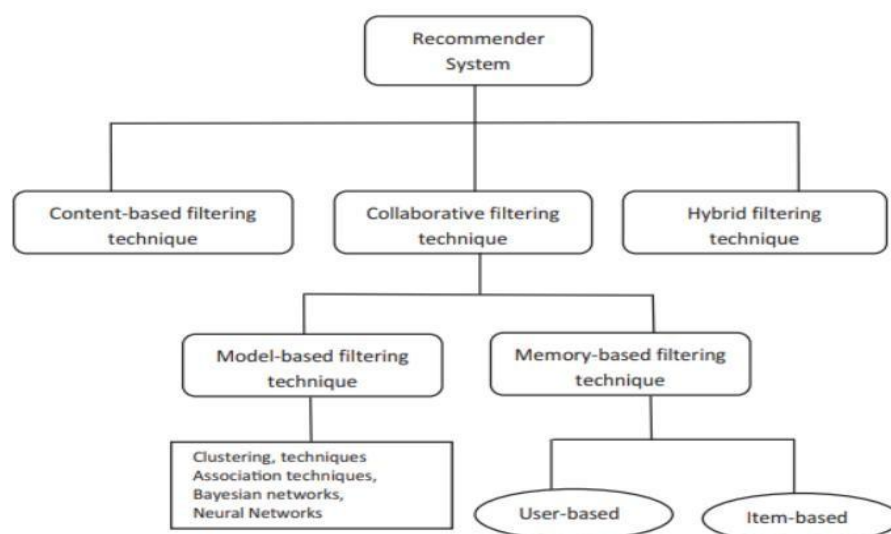
BAB II

LANDASAN TEORI

2.1 Sistem Rekomendasi

Sistem rekomendasi adalah sistem perangkat lunak yang dapat menyarankan atau merekomendasikan item kepada pengguna. Item adalah istilah umum yang digunakan untuk menunjukkan apa yang disarankan sistem kepada pengguna, seperti memilih barang yang harus dibeli, mendengarkan musik, atau membaca berita online (Ricci et al., 2011).

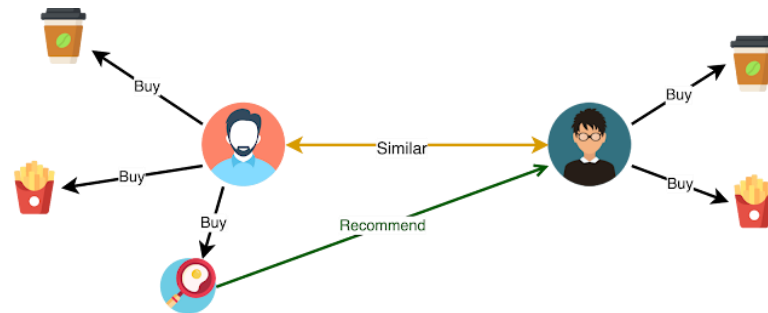
Tujuan utama sistem rekomendasi adalah untuk membuat sistem yang membantu pengguna dalam membuat keputusan dan memberikan akses ke rekomendasi yang sesuai dengan preferensi dan minat pengguna untuk komunitas besar. Sulit bagi sistem rekomendasi untuk membuat rekomendasi yang terjangkau, terpersonalisasi, dan berkualitas tinggi. Sistem rekomendasi bekerja dengan memahami selera pengguna dan secara otomatis menemukan konten baru yang diharapkan oleh pengguna (Mahendra Wiputra & Jauhari Shandi, 2021).



Gambar 2.1 Teknik Sistem Rekomendasi

2.2 Collaborative Filtering

Collaborative Filtering (CF) adalah merupakan proses menggunakan pendapat orang lain untuk memilih atau menilai sesuatu. Ide utamanya adalah untuk memprediksi item mana yang akan disukai atau menarik bagi seorang pengguna dengan menggunakan data tentang perilaku pengguna sebelumnya dan pendapat mereka.



Gambar 2.2 Metode *Collaborative Filtering*

Sementara CF murni menggunakan matriks yang berisi rating item user sebagai satu-satunya input, outputnya terdiri dari dua jenis: (1) prediksi (numerik) yang menggambarkan tingkat kesukaan pengguna terhadap sebuah item, dan (2) menyediakan daftar item yang disarankan. Dalam CF, istilah "pengguna" mengacu pada individu yang menilai komponen sistem dan kemudian menerima saran dari sistem (Jannach et al., 2011).

2.2.1 Metode *Collaborative Filtering*

Model peratingan dalam sistem rekomendasi akan berbeda dengan model berbasis *Collaborative Filtering*, yang memiliki dua model :

1. *User-Based Collaborative Filtering*

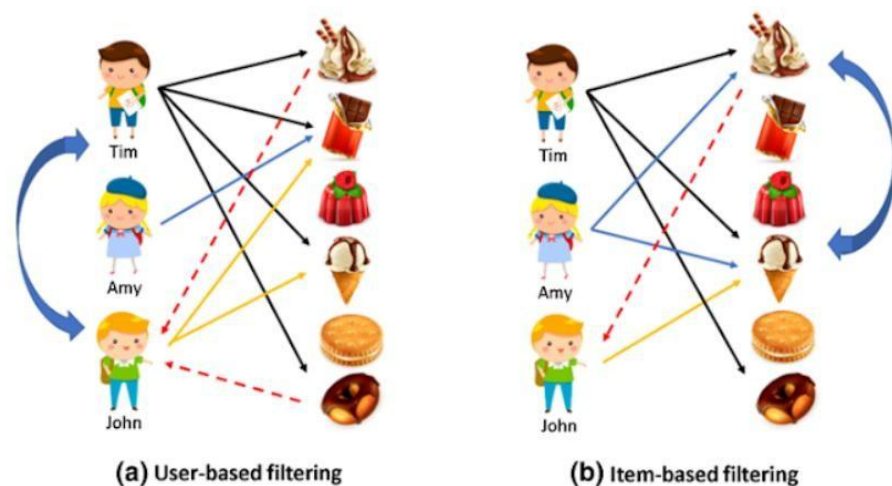
Metode ini bergantung pada kenyataan bahwa seorang pengguna mengikuti kelompok yang lebih besar, yang memiliki perilaku individu yang seragam. Rekomendasi didasarkan pada barang-barang yang disukai atau

dibeli oleh berbagai anggota kelompok. Berdasarkan item-item yang telah dipilih oleh tetangga terdekat seorang user, item-item yang kemungkinan akan dipilih oleh user tersebut di masa mendatang diprediksi. Metode ini disebut nearest neighbors method (Surendra et al., 2011).

2. Item-Based Collaborative Filtering

Metode ini mengandalkan analisis informasi historis tentang hubungan antar item. Dengan demikian, ketika seseorang memutuskan untuk membeli satu item, itu mengarah pada pembelian sejumlah item lainnya, juga dikenal sebagai "himpunan item".

Rekomendasi didasarkan pada fakta bahwa pengguna cenderung memilih barang yang mirip dengan barang yang telah mereka pilih sebelumnya (Deshpande & Karypis, 2004).



Gambar 2.3 User-based dan Item-based filtering

2.3 Matrix Factorization

Matrix Factorization merupakan algoritma umum yang digunakan dalam *Collaborative Filtering* untuk memodelkan hubungan diantara pengguna dan juga item dengan memecah matriks peringkat pengguna-item menjadi dua atau lebih matriks yang

lebih kecil, bertujuan untuk menemukan representasi-fitur tersembunyi (fitur laten) dari pengguna dan item yang dapat digunakan untuk memprediksi peringkat yang hilang (*sparsity*) atau memperbaiki rekomendasi. Dalam *Matrix Factorization* ini, untuk menggambarkan setiap pengguna dan item dalam ruang dimensi yang lebih rendah, di mana setiap dimensi mewakili fitur-fitur yang penting dari pengguna atau item. (Arviyanto & Soebroto, 2017). Sebelumnya mencari matriks P (*User*) dan Q (*Item*) dapat dilihat pada **persamaan 1** dan **persamaan 2**.

$$P = U_i \cdot \sqrt{\Sigma}$$

Dimana:

P adalah matriks fitur laten untuk pengguna (*user*)

Σ adalah matriks diagonal berisi nilai *singular*

$\sqrt{\Sigma}$ adalah akar dari nilai *singular* (digunakan untuk menyeimbangkan kontribusi)

$$Q = V_i \cdot \sqrt{\Sigma}$$

Dimana:

Q adalah matriks fitur laten untuk item

V_i adalah komponen hasil dekomposisi dari matriks item (biasanya dari transpose matriks rating)

Di mana P dan Q mewakili representasi fitur laten dari pengguna dan item. Prediksi rating dilakukan dengan mengalikan matriks tersebut dilihat pada **persamaan 3**.

$$\hat{r}_{ij} = P \cdot Q^T$$

Dimana :

$\hat{r}_{ij}$ adalah prediksi rating *user* terhadap item.

P adalah matriks pengguna (*user latent features*)

Q^T adalah *transpose* dari matriks *item* (*item latent features*).

Hasil perkalian ini akan menghasilkan rating yang diprediksi oleh model untuk setiap pasangan pengguna – *item*.

2.3.1 Singular Value Decomposition (SVD)

Sebelum memberi prediksi rating dengan algoritma *Matrix Factorization*, dalam bidang *Machine Learning*, *Singular Value Decomposition* (SVD) adalah teknik dekomposisi matriks yang sering digunakan. SVD sangat bermanfaat untuk mengurai matriks menjadi tiga matriks yang lebih sederhana. SVD membantu memprediksi preferensi pengguna dan memberikan rekomendasi yang dipersonalisasi (Sitanggang et al., 2023)

Hasil rekomendasi item yang dihasilkan oleh model ini akan diurutkan dari nilai prediksi rating tertinggi hingga yang terendah. Teknik ini juga dapat mengatasi kelemahan metode *Collaborative Filtering* (CBF), yaitu tidak dapat menghasilkan rekomendasi produk yang disesuaikan dengan individu (Al Amin et al., 2021). Menurut Yuan dkk. (2019)

Matrix Factorization pada sistem rekomendasi menggunakan SVD dengan sering. Dengan SVD, hubungan antara user dan item dapat digabungkan secara linear oleh faktor-faktor tertentu. Dengan kata lain, SVD mendekomposisi matriks rating menjadi dua matriks rating rendah, dan hasil kali keduanya adalah matriks rating yang lebih dekat dengan nilai sebenarnya.

SVD dapat didefinisikan melalui **persamaan 4**

$$A = U S V^T$$

Keterangan:

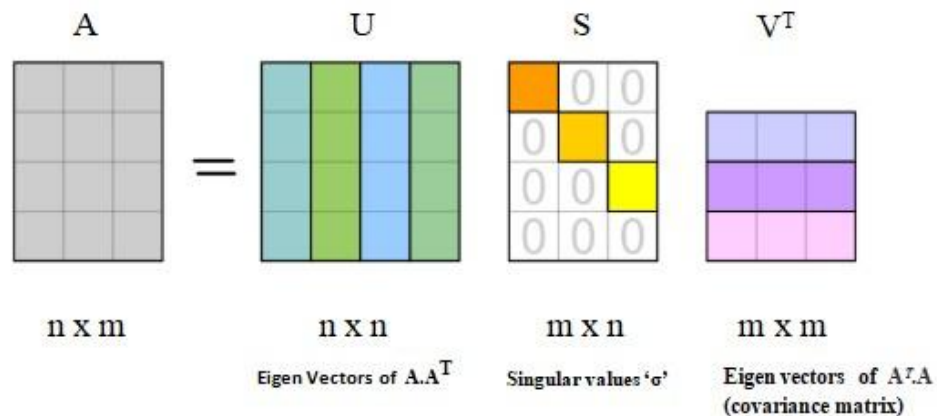
A adalah matriks *rating* berukuran $m \times n$.

U adalah matriks *singular* kiri ortogonal berukuran $n \times n$ yang mewakili matrik fitur pengguna.

$S (\Sigma)$ adalah matriks diagonal singular value berukuran $m \times n$.

V^T adalah matriks *singular* kanan ortogonal berukuran $m \times m$ yang menunjukkan kemiripan antara item dan faktor laten.

Setiap fitur diwakili oleh U sebagai jumlah pengguna yang "menyukai"-nya, dan V^T menunjukkan seberapa relevan setiap fitur dengan setiap film. Dengan mengeluarkan faktor laten dan meletakkan setiap pengguna ke dalam ruang laten berdimensi r , SVD mengurangi dimensi matriks *rating* A (Bhalse & Thakur, 2021).



Gambar 2.4 Singular Value Decomposition Matrix

Dimana matriks A merupakan matriks yang merepresentasikan *rating* pengguna terhadap setiap *item*, matriks U merupakan *eigenvector* dari matriks AA^T , matriks V merupakan *eigenvector* dari matriks A^TA , dan S merupakan dapat diperoleh dari akar kuadrat dari eigenvalues dari matriks AA^T atau A^TA (Al Amin et al., 2021).

Untuk mencari matriks U dilakukan normalisasi terhadap *eigenvektor* yang telah didapat. Normalisasi pada eigenvector terdapat pada **persamaan 5**

$$\mathbf{v}_{norm} = \frac{\mathbf{v}}{\|\mathbf{v}\|}$$

Keterangan:

$\mathbf{v}_{norm}$ adalah eigenvector yang dinormalisasi

$\mathbf{v}$ adalah eigenvector asli

$\|\mathbf{v}\|$ adalah modulus eigenvector $\mathbf{v}$, yang didapat dengan **persamaan 6**

$$\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2 + \dots + v_n^2}$$

Dimana $v_1^2 + v_2^2 + \dots + v_n^2$ adalah elemen dari *eigenvector* $\mathbf{v}$.

Setelah memperoleh matriks S dan matriks U, langkah selanjutnya adalah mencari matriks $\mathbf{V}^T$. Untuk menemukan matriks $\mathbf{V}^T$, berikut dapat digunakan **persamaan 7**

$$\mathbf{V}^T = \mathbf{S}^{-1} \cdot \mathbf{U}^T \cdot \mathbf{A}$$

Keterangan:

$\mathbf{S}^{-1}$ = invers dari matriks S

$\mathbf{U}^T$ = transpose matriks U

A = matriks rating awal

2.4 Teknik *Clipping* dalam Prediksi Rating

Dalam proses prediksi pada sistem rekomendasi dengan *Matrix Factorization* maupun *Singular Value Decomposition* (SVD), terkadang muncul nilai prediksi yang berada di luar rentang skala rating yang telah ditentukan. Hal ini umumnya disebabkan oleh hasil perkalian matriks yang tidak dibatasi secara eksplisit. Sebagai contoh, skala

rating pada sistem bisa saja ditetapkan antara 1 hingga 5, namun hasil prediksi menunjukkan nilai seperti 5.6 atau bahkan -0.3.

Untuk mengatasi permasalahan tersebut, diterapkan teknik yang dikenal sebagai *clipping* atau pemangkasan nilai prediksi. Teknik ini berfungsi membatasi hasil prediksi agar tetap berada dalam batas minimum dan maksimum rating. Dengan kata lain, jika hasil prediksi lebih rendah dari nilai terendah skala (misalnya 1), maka nilai tersebut akan diubah menjadi 1. Sebaliknya, jika hasil prediksi lebih tinggi dari nilai maksimum (misalnya 5), maka akan dikoreksi menjadi 5.

Penggunaan *clipping* penting untuk menjaga agar hasil rekomendasi tetap relevan dan masuk akal secara logika pengguna. Tanpa teknik ini, sistem bisa merekomendasikan item berdasarkan prediksi yang secara teknis valid, tetapi secara praktis tidak realistis.

Menurut Koren, Bell, dan Volinsky (2009), *clipping* digunakan dalam proses evaluasi prediksi rating untuk menghindari efek negatif dari nilai ekstrim yang terlalu rendah atau terlalu tinggi. Pemangkasan ini juga menjaga konsistensi sistem dalam membandingkan nilai prediksi dengan data aktual saat dilakukan pengukuran seperti RMSE atau MAE.

2.5 Fitur Laten

Representasi dari pengguna dan item yang memungkinkan untuk memprediksi rating atau preferensi. Fitur ini memiliki keuntungan memiliki kemampuan untuk menemukan pola kompleks dan efektif dalam menangani data yang *spars* (jarang), tetapi juga memiliki kekurangan pada pelatihan yang lama dan kompleksitas komputasi yang tinggi (Koren et al., 2009).

2.6 *Mean Absolute Error (MAE)*

Mean Absolute Error (MAE) adalah salah satu metode evaluasi kinerja model prediksi yang umum digunakan dalam sistem rekomendasi, regresi, maupun peramalan. MAE digunakan untuk menghitung rata-rata dari selisih absolut antara nilai prediksi dengan nilai aktual. Nilai MAE yang lebih kecil menunjukkan bahwa model memberikan hasil prediksi yang lebih mendekati nilai sebenarnya. Rumus MAE terdapat pada **persamaan 8**

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

Dimana :

y_i adalah nilai aktual (asli) ke-i,

$\hat{y}_i$ adalah nilai prediksi ke-i,

n adalah jumlah data yang diuji,

$|y_i - \hat{y}_i|$ adalah selisih absolut antara nilai aktual dan prediksi.

Mean Absolute Error (MAE) digunakan dalam penelitian ini sebagai alat evaluasi akurasi dari hasil prediksi model terhadap data asli. MAE dipilih karena perhitungannya yang sederhana dan tidak sensitif terhadap nilai ekstrem (outlier). Dalam penelitian Saputra dan Purwarianti (2016), MAE digunakan untuk mengukur akurasi sistem rekomendasi dengan algoritma *Matrix Factorization*, dan terbukti dapat merepresentasikan performa model secara umum.

2.7 *Confusion Matrix*

Confusion matrix adalah tabel yang digunakan untuk mengevaluasi kinerja model klasifikasi dengan membandingkan hasil prediksi dengan label aktual. Dalam konteks

sistem rekomendasi film, confusion matrix membantu menilai seberapa baik model dalam memprediksi preferensi pengguna terhadap film tertentu (Habibi et al., 2023).

Tabel 2.5 *Confusion Matrix*

	Prediksi: Suka	Prediksi: Tidak Suka
Aktual: Suka	<i>True Positive (TP)</i>	<i>False Negative (FN)</i>
Aktual: Tidak Suka	<i>False Positive (FP)</i>	<i>True Negative (TN)</i>

Di mana:

True Positive (TP) : Prediksi "suka" yang benar.

False Positive (FP) : Prediksi "suka" yang salah (sebenarnya "tidak suka").

False Negative (FN) : Prediksi "tidak suka" yang salah (sebenarnya "suka").

True Negative (TN) : Prediksi "tidak suka" yang benar.

Dari *confusion matrix*, beberapa metrik evaluasi yang dapat dihitung meliputi:

1. *Accuracy* : Mengukur proporsi prediksi yang benar terhadap total prediksi.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

2. *Precision* : Mengukur proporsi prediksi positif yang benar – benar positif.

$$Precision = \frac{TP}{TP+FP}$$

3. *Recall* : Mengukur proporsi data positif yang berhasil diprediksi dengan benar.

$$Recall = \frac{TP}{TP+FN}$$

4. *F1-score* : Harmonik *mean* dari *precision* dan *recall*, memberikan keseimbangan antara keduanya.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

2.8 Dataset

Dataset adalah sistem rekomendasi berisi data yang menunjukkan interaksi pengguna dengan item yang direkomendasikan, seperti film, barang, musik, atau artikel. Dataset mencakup beberapa macam elemen, yaitu : *User Data*, *Item Data*, dan *Interaction Data*.

Dataset ini sering digunakan dalam penelitian untuk menguji algoritma rekomendasi, termasuk *Collaborative Filtering* dan *Matrix Factorization*. Dengan menganalisis pola peringkat dalam dataset, peneliti dapat menemukan fitur laten yang mempengaruhi preferensi pengguna terhadap film tertentu. Ini memungkinkan pengembangan model rekomendasi yang lebih akurat dan efektif (Quadrana et al., 2018).

2.9 Website

Kumpulan halaman web yang berisi informasi data digital, seperti teks, gambar, animasi, suara, dan video, yang disediakan melalui jalur internet, sehingga setiap orang di seluruh dunia dapat mengaksesnya dan melihatnya. Halaman web dibuat menggunakan bahasa standar yang dikenal sebagai HTML. Web browser menerjemahkan skrip HTML ini sehingga informasi dapat ditampilkan dalam bentuk yang dapat dibaca oleh semua orang (Abdullah, 2018:1).

3.0 Flowchart

Flowchart adalah gambaran yang terdiri dari suatu grafik yang berisi langkah langkah dan urutan prosedur program. *Flowchart* biasanya mempermudah penyelesaian masalah setelah evaluasi lebih lanjut. Ini digunakan dalam proses analisis, perancangan, dan pengkodean untuk membagi masalah menjadi bagian yang lebih kecil untuk digunakan. Setiap langkah digambarkan dalam bentuk diagram dan dihubungkan dengan garis atau arah panah. (Zalukhu et al., 2023).

Tabel 3.1 Simbol *Flowchart*

	Flow Direction Symbol Yaitu simbol yang digunakan untuk menghubungkan antara simbol yang satu dengan simbol yang lain. Simbol ini disebut juga <i>connecting line</i> .		Symbol Manual Input Simbol untuk pemasukan data secara manual <i>on-line keyboard</i> .
	Terminator Symbol Yaitu simbol untuk permulaan (<i>start</i>) atau akhir (<i>stop</i>) dari suatu kegiatan.		Symbol Preparation Simbol untuk mempersiapkan penyimpanan yang akan digunakan sebagai tempat pengolahan di dalam <i>storage</i> .
	On-Page Reference Yaitu simbol untuk keluar-masuk atau penyambungan proses dalam lembar/halaman yang sama.		Symbol Predefine Proses Simbol untuk pelaksanaan suatu bagian (<i>sub-program</i>)/ <i>prosedure</i> .
	Off-Page Reference Yaitu simbol untuk keluar-masuk atau penyambungan proses dalam lembar/halaman yang berbeda.		Symbol Display Simbol yang menyatakan peralatan <i>output</i> yang digunakan yaitu layar, <i>plotter</i> , dan sebagainya.
	Processing Symbol Simbol yang menunjukkan pengolahan yang dilakukan komputer.		Symbol disk and On-line Storage Simbol yang menyatakan <i>input</i> yang berasal dari <i>disk</i> atau disimpan ke <i>disk</i> .
	Symbol Manual Operation Simbol yang menunjukkan pengolahan yang tidak dilakukan komputer.		Symbol Magnetik Tape Unit Simbol yang menyatakan <i>input</i> berasal dari pita <i>magnetik</i> atau <i>output</i> disimpan ke pita <i>magnetik</i> .
	Symbol Decision Simbol pemilihan proses berdasarkan kondisi yang ada.		Symbol Punch Card Simbol yang menyatakan bahwa <i>input</i> berasal dari kartu atau <i>output</i> ditulis ke kartu.
	Symbol Input-Output Simbol yang menyatakan proses <i>input</i> dan <i>output</i> tanpa tergantung dengan jenis peralatannya.		Symbol Dokumen Simbol yang menyatakan <i>input</i> berasal dari dokumen dalam bentuk kertas atau <i>output</i> dicetak ke kertas.

3.1 *Unified Modelling Language (UML)*

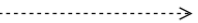
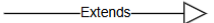
Setiap elemen sistem perangkat lunak dapat digambarkan dan divisualisasikan melalui pemodelan grafis, yang disebut *Unified Modelling Language (UML)*. Tujuan dari model ini adalah untuk mendapatkan pemahaman tentang komponen yang ada di dalam lingkup area sistem, serta hubungan antara sistem dan subsistem. Secara umum, *Unified Modeling Language* adalah bahasa yang digunakan untuk menampilkan spesifikasi

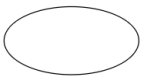
sistem pengembangan software berbasis objek oriented dalam grafik atau gambar (Heriyanto, 2018). Untuk perancangan sistem UML yang peneliti gunakan antara lain *use case diagram* dan *activity diagram*.

3.1.1 Use Case Diagram

Use case diagram adalah jenis diagram *Unified Modeling Language* (UML) yang digunakan dalam rekayasa perangkat lunak untuk menunjukkan interaksi antara berbagai aktor, seperti pengguna atau sistem eksternal, dan suatu sistem secara visual. Diagram ini menunjukkan bagaimana pengguna berinteraksi dengan sistem untuk mencapai tujuan tertentu. Dari perspektif pengguna, aktor digambarkan sebagai figur manusia, dan use case digambarkan sebagai oval. Garis hubung menunjukkan hubungan dan interaksi antara aktor dan use case. Ini menunjukkan bagaimana sistem berfungsi dan bertindak (Septiansyah et al., 2024).

Tabel 3.2 Simbol *Use Case Diagram*

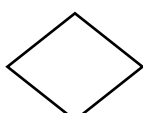
No.	Gambar	Nama	Keterangan
1		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case.
2		<i>Dependency</i>	Hubungan dimana perubahan pada suatu elemen independen akan mempengaruhi elemen yang tergantung padanya.
3		<i>Generalization</i>	Hubungan dimana objek anak (descendent) mewarisi perilaku dan struktur dari objek induk (ancestor).
4		<i>Include</i>	Menspesifikasikan bahwa use case sumber secara eksplisit menyertakan use case lain.
5		<i>Extend</i>	Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan.
6		<i>Association</i>	Hubungan antara objek satu dengan objek lainnya.

7		<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang menghasilkan suatu hasil tertentu bagi aktor.
9		<i>Collaboration</i>	Interaksi elemen yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah elemen penyusunnya.
10		<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan sumber daya komputasi.

3.1.2 Activity Diagram

Activity diagram adalah bentuk diagram *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan proses atau aliran kerja yang terjadi dalam suatu proses atau sistem. Diagram ini menunjukkan kumpulan tindakan, kegiatan, dan keputusan yang terjadi sepanjang waktu (Septiansyah et al., 2024).

Tabel 3.3 Simbol *Activity Diagram*

No	Gambar	Nama	Keterangan
1		<i>Activity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
2		<i>Action</i>	State dari sistem yang mencerminkan eksekusi dari suatu aksi.
3		<i>Initial Node</i>	Bagaimana objek dibentuk atau diawali.
4		<i>Activity Final Node</i>	Bagaimana objek dibentuk dan diakhiri.
5		<i>Decision</i>	Digunakan untuk menggambarkan suatu keputusan / tindakan yang harus diambil pada kondisi tertentu.
6		<i>Line Connector</i>	Digunakan untuk menghubungkan satu simbol dengan simbol lainnya.

3.2 Platform

Platform adalah sebuah kotak digital yang banyak digunakan orang untuk berbagai alasan. Secara umum, *platform* adalah sebuah wadah yang digunakan untuk memulai sistem untuk bekerja sesuai dengan rencana program yang telah dibuat. Misalnya, platform digital digunakan untuk pendidikan online(Wibawa, 2021).

3.3 Visual Studio Code

Visual Studio Code (VS Code) adalah sebuah teks editor yang ringan dan berfungsi dengan baik yang dikembangkan oleh Microsoft untuk berbagai sistem operasi, termasuk *Linux*, *Mac*, dan *Windows* versi. Ini mendukung bahasa pemrograman *JavaScript*, *Typescript*, dan *Node.js* secara langsung, serta bahasa pemrograman lainnya dengan bantuan plugin yang dapat dipasang melalui marketplace *Visual Studio Code*, seperti *C++*, *C#*, *Python*, *Go*, *Java*, *PHP*, dan lainnya (Surya Ningsih et al., 2022).

3.4 Penelitian Terdahulu

Tabel 3.4 Penenlitian Terdahulu

No.	Nama Peneliti	Judul	Metode	Kesimpulan
1	Muhammad Rizky Sitorus (2023)	Implementasi Model-Based Collaborative Filtering pada Sistem Rekomendasi Film Menggunakan Algoritma SVD (<i>Singular Value Decomposition</i>)	<i>Singular Value Decomposition</i> (SVD)	Penerapan SVD dalam <i>Collaborative Filtering</i> efektif meningkatkan akurasi rekomendasi, namun memerlukan sumber daya komputasi yang memadai.

2	Hao Wang (2022)	<i>MovieMat: Context-aware Movie Recommendation with Matrix Factorization by Matrix Fitting</i>	<i>Matrix Factorization</i> dengan <i>Matrix Fitting</i>	Pendekatan ini menunjukkan bahwa mempertimbangkan konteks pengguna dapat meningkatkan akurasi rekomendasi film.
3.	Amin, M., & Jafar, A. (2021)	Mereduksi Error Prediksi Pada Sistem Rekomendasi Menggunakan Pendekatan <i>Collaborative Filtering</i> Berbasis Model <i>Matrix Factorization</i>	<i>Matrix Factorization</i>	<i>Matrix Factorization</i> terbukti efektif dalam meningkatkan akurasi prediksi dalam sistem rekomendasi, tetapi membutuhkan dataset besar agar hasilnya optimal.
4	Agustian, H. (2020)	Sistem Rekomendasi Film Menggunakan Metode <i>Collaborative Filtering</i> dan <i>K-Nearest Neighbors</i>	<i>Collaborative Filtering</i> dan K-NN	<i>Collaborative Filtering</i> berbasis pengguna cukup efektif, tetapi menghadapi tantangan <i>cold-start</i> ketika pengguna baru memiliki sedikit data.
5	Ahmad, R., & Setiawan, A. (2020)	Penerapan <i>Matrix Factorization</i> pada Sistem Rekomendasi Film	<i>Matrix Factorization</i> (SVD)	Kombinasi <i>Collaborative Filtering</i> dengan K-NN dapat meningkatkan hasil rekomendasi, tetapi masih bergantung pada jumlah data pengguna yang cukup.

BAB III

METODOLOGI PENELITIAN

3.1 Desain Penelitian

Penelitian ini menggunakan pendekatan kuantitatif dan eksperimental dalam mengembangkan sistem rekomendasi film berbasis *Collaborative Filtering* dengan metode *Matrix Factorization*. Pendekatan kuantitatif digunakan untuk menganalisis data pengguna dan pola pemberian rating terhadap film, yang kemudian diproses menggunakan teknik *Collaborative Filtering* guna menghasilkan rekomendasi yang lebih akurat.

Evaluasi dilakukan untuk mengetahui seberapa akurat hasil prediksi sistem rekomendasi. Penilaian dilakukan dengan dua pendekatan, yaitu klasifikasi dan regresi. Pada pendekatan klasifikasi digunakan metrik seperti *accuracy*, *precision*, *recall*, dan *F1 score*. Sedangkan untuk pendekatan regresi digunakan *Mean Absolute Error* (MAE) untuk menghitung rata-rata selisih antara rating sebenarnya dan hasil prediksi. Nilai MAE yang kecil menandakan prediksi sistem semakin mendekati nilai yang diharapkan.

3.1.1 Identifikasi Masalah

Pada tahap identifikasi masalah, penelitian ini akan menganalisis:

1. Sistem rekomendasi di Viu belum optimal dalam menyesuaikan rekomendasi film dengan preferensi pengguna.
2. Sebagian besar pengguna hanya memberikan sedikit rating, menyebabkan matriks rating menjadi jarang terisi dan menghambat kinerja sistem rekomendasi.
3. Pengguna baru yang belum memiliki riwayat interaksi sulit mendapatkan rekomendasi yang relevan.

3.1.2 Studi Literatur

Melakukan penelitian tentang penelitian dan teori yang relevan termasuk:

1. Teori dan penelitian sebelumnya tentang *Collaborative Filtering*, *Matrix Factorization*, serta metode evaluasi seperti *accuracy*, *precision*, *recall*, dan *F -1 score* dan regresi digunakan *Mean Absolute Error* (MAE).
2. Menganalisis bagaimana *Matrix Factorization* mengatasi masalah *sparsity*.
3. Mengkaji teknik *Singular Value Decomposition* (SVD) dalam meningkatkan akurasi rekomendasi.
4. Memahami dataset yang akan digunakan.

3.1.3 Pengumpulan Data

Dataset yang digunakan dalam penelitian ini diperoleh dari *Platform Viu*, yang berisi data pengguna, film, serta *rating* yang diberikan pengguna terhadap film tertentu.

3.1.4 Preprocessing Data

Tahap ini sangat penting untuk memastikan bahwa data yang digunakan bersih, relevan dan siap untuk diolah, meliputi:

1. Membersihkan data dari duplikasi atau nilai yang tidak valid.
2. Melakukan normalisasi *rating* agar lebih sesuai untuk analisis.
3. Membagi dataset menjadi *training* dan *testing set* untuk evaluasi model.

3.1.5 Implementasi

Pada tahap ini, dilakukan penerapan metode *Collaborative Filtering* menggunakan algoritma *Matrix Factorization* untuk membangun sistem rekomendasi film. Implementasi ini mencakup beberapa langkah utama, yaitu

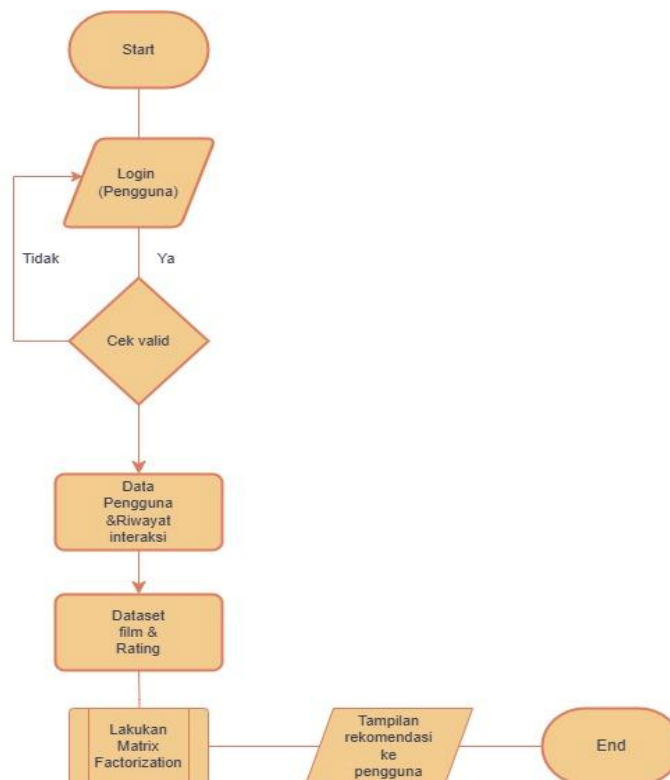
pemodelan, pelatihan, serta pembuatan prediksi rating berdasarkan data yang telah diproses sebelumnya.

1. Menggunakan teknik *Singular Value Decomposition* (SVD) untuk mendekomposisi matriks user-item.
2. Melatih model menggunakan dataset yang telah diproses.

3.1.6 Evaluasi

Tahap evaluasi bertujuan untuk mengukur seberapa baik model rekomendasi yang telah diimplementasikan dalam memprediksi rating film. Evaluasi dilakukan dengan menggunakan metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *F-1 score*.

3.2 Perancangan Sistem



Gambar 3.2 Alur Perancangan Sistem

Proses dimulai dengan start, yang menandakan awal sistem. Pengguna kemudian diminta untuk *login* ke sistem. Jika *login* gagal, sistem akan meminta pengguna untuk mencoba kembali. Jika *login* berhasil, sistem akan melakukan cek valid, yaitu memverifikasi akun pengguna. Jika akun tidak valid, pengguna harus login ulang. Jika valid, sistem melanjutkan ke proses berikutnya. Sistem kemudian mengambil data pengguna & riwayat interaksi, yaitu informasi pengguna serta film yang telah mereka tonton dan beri rating sebelumnya. Setelah itu, sistem juga mengambil dataset film & *rating*, yang berisi daftar film beserta *rating* yang telah diberikan oleh pengguna lain. Setelah mendapatkan data, sistem akan menjalankan *Matrix Factorization*, yaitu teknik yang digunakan untuk menemukan pola tersembunyi dalam *rating* film guna memprediksi preferensi pengguna terhadap film yang belum mereka tonton. Setelah proses ini selesai, sistem akan menampilkan rekomendasi ke pengguna, yaitu daftar film yang dipersonalisasi berdasarkan hasil perhitungan. Terakhir, sistem mencapai *end*, yang menandakan bahwa proses rekomendasi telah selesai.

3.3 Flowchart Algoritma

3.3.1 Flowchart Collaborative Filtering

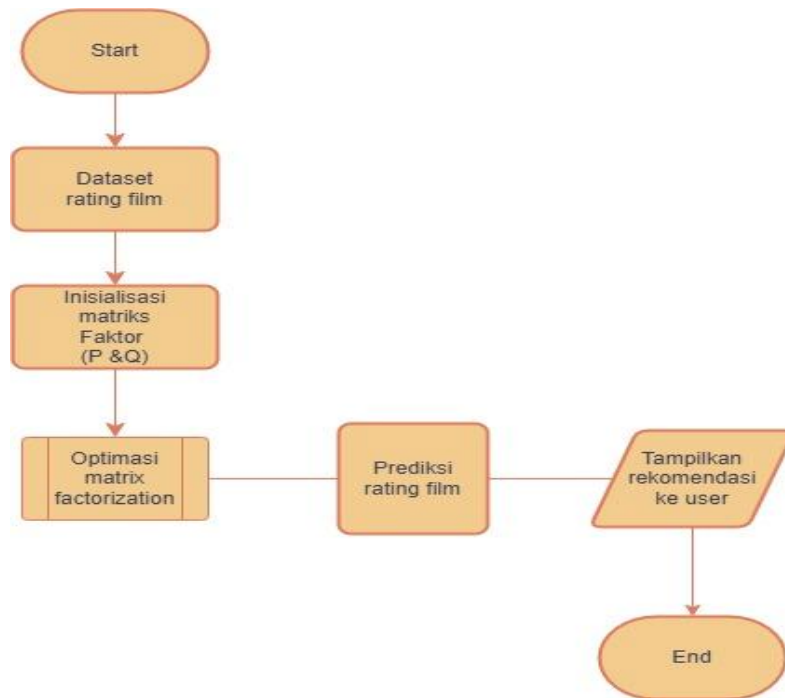


Gambar 3.3 Alur *Collaborative Filtering*

Proses dimulai dari *start*, yang menunjukkan mulainya sistem rekomendasi. Selanjutnya, sistem mengumpulkan data rating pengguna dan film, yang berisi data tentang peringkat yang diberikan pengguna terhadap film yang telah mereka tonton. Data ini digunakan oleh sistem untuk memahami preferensi pengguna. Setelah mendapatkan data, sistem akan menghitung prediksi rating untuk film yang belum ditonton oleh pengguna. Perhitungan ini dilakukan dengan metode *Collaborative Filtering*, baik berbasis pengguna (*User-Based Collaborative Filtering*) maupun berbasis *item* (*Item Based Collaborative Filtering*). Dalam metode berbasis pengguna, sistem mencari pengguna lain dengan preferensi serupa dan menggunakan rating mereka untuk memprediksi rating pengguna target. Dalam metode berbasis *item*, sistem mencari film yang mirip dengan yang sudah ditonton

dan memperkirakan rating berdasarkan kemiripan tersebut. Pada langkah berikutnya, pengguna dapat melihat daftar film yang telah dipilih sebagai rekomendasi untuk menonton yang sesuai dengan preferensi mereka. Terakhir, sistem mencapai end, menandakan selesainya proses rekomendasi.

3.3.2 Flowchart Matrix Factorization



Gambar 3.4 Alur *Matrix Factorization*

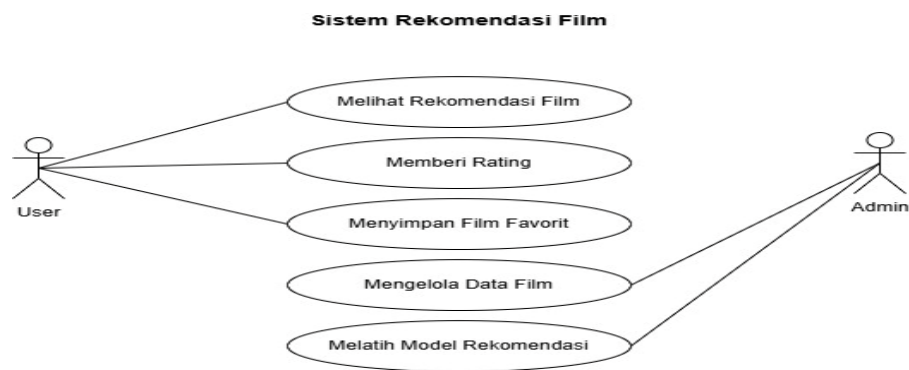
Proses dimulai dengan *Start*, yang merupakan tanda awal sistem rekomendasi. Selanjutnya, sistem mengambil Dataset Penilaian Film, yang merupakan kumpulan data yang mengandung penilaian film dari berbagai pengguna. Setelah dataset ini tersedia, sistem memulai *Matrix Factorization* (P & Q), yang menunjukkan hubungan laten antara pengguna dan film.

Sistem menjalankan optimasi *Matrix Factorization* setelah inisialisasi. Pada titik ini, nilai dalam matriks faktor diperhalus dengan menggunakan teknik *Singular*

Value Decomposition (SVD) untuk menunjukkan preferensi pengguna dengan lebih akurat.

Hasil dari proses optimasi memungkinkan sistem untuk melakukan prediksi rating film, yang berarti perkiraan nilai yang mungkin diberikan pengguna terhadap film yang belum mereka tonton. Berdasarkan perkiraan ini, sistem kemudian menampilkan rekomendasi kepada pengguna, yang terdiri dari daftar film yang sesuai dengan preferensi pengguna. Terakhir, proses berakhir dengan *end*, yang menandakan bahwa rekomendasi telah diberikan kepada pengguna.

3.4 Use Case Diagram



Gambar 3.5 Use Case Diagram

Use Case Diagram di atas menggambarkan alur sistem rekomendasi film yang melibatkan dua aktor utama, yaitu *User* dan *Admin*. Berikut adalah penjelasan alurnya:

User

1. *User* dapat melihat daftar film yang direkomendasikan berdasarkan riwayat tontonan dan *rating* sebelumnya.
2. *User* dapat memberikan rating pada film yang telah ditonton untuk meningkatkan kualitas rekomendasi di masa depan.

Admin

1. Admin bertanggung jawab untuk menambah, menghapus, atau memperbarui informasi film dalam sistem.
2. Admin menjalankan proses pembelajaran mesin menggunakan *Collaborative Filtering* dan *Matrix Factorization* untuk meningkatkan akurasi rekomendasi film.

Alur Interaksi:

User login ke dalam sistem, kemudian sistem menampilkan rekomendasi film berdasarkan preferensi *user*. Setelah itu, user dapat memberi rating atau menyimpan film ke favorit sebagai umpan balik. Data yang diberikan oleh user akan digunakan oleh sistem untuk memperbarui model rekomendasi agar lebih akurat. Sementara itu, admin bertanggung jawab dalam mengelola data film dengan menambahkan atau menghapus film yang tersedia di sistem. Selain itu, admin juga melatih model rekomendasi secara berkala agar hasil rekomendasi yang diberikan kepada user semakin relevan dan sesuai dengan preferensi mereka.

3.5 Activity Diagram



Gambar 3.6 Activity Diagram

User memulai dengan *login* ke dalam sistem untuk mengakses rekomendasi film yang dipersonalisasi. Setelah berhasil masuk, sistem menganalisis preferensi user berdasarkan riwayat tontonan dan rating yang telah diberikan sebelumnya. Selanjutnya, sistem mengambil data rating dan informasi film dari database untuk digunakan dalam proses perhitungan rekomendasi. Dengan menerapkan *Collaborative Filtering* dengan algoritma *Matrix Factorization*, sistem memprediksi *rating* film yang belum ditonton user berdasarkan pola *rating* dari pengguna lain yang memiliki preferensi serupa. Hasil dari proses ini adalah daftar rekomendasi film yang paling relevan bagi *user*. Setelah mendapatkan rekomendasi, *user* dapat memberikan umpan balik dengan memberikan rating atau menyimpan film ke daftar favorit. Sistem kemudian menggunakan umpan balik ini untuk memperbarui model rekomendasi, sehingga kualitas prediksi semakin meningkat seiring waktu.

3.6 *Interface UI/UX*

3.6.1 *Halaman Login*

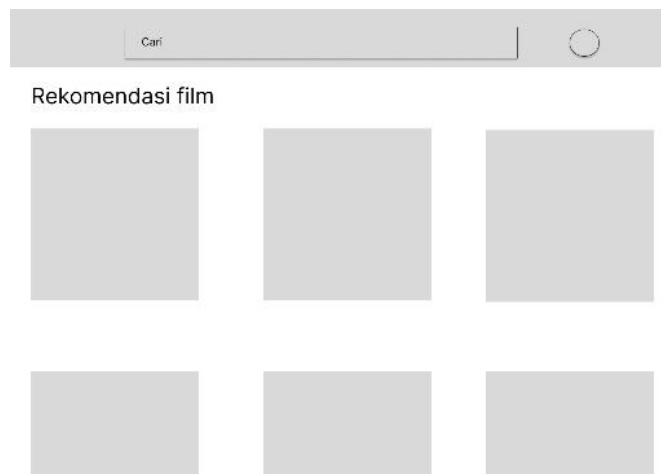
The image shows a login interface on a light gray background. At the top center is the word "Login" in a large, bold, black font. Below it are two white rectangular input fields. The first field is labeled "Username" in a small, gray font. The second field is labeled "Password" in a small, gray font. Below these two fields is a white rectangular button with the word "Login" in a small, gray font.

Gambar 3.7 Halaman *Login*

Menampilkan halaman *login* sederhana yang terdiri dari beberapa elemen utama. Di bagian atas terdapat teks "*Login*" sebagai judul, menunjukkan bahwa ini adalah halaman untuk masuk ke dalam sistem. Di bawahnya, terdapat dua kotak

input, satu untuk "*Username*" dan satu lagi untuk "*Password*". Kotak *input* pertama digunakan untuk memasukkan nama pengguna, sedangkan kotak *input* kedua digunakan untuk memasukkan kata sandi yang biasanya ditampilkan dalam bentuk bintang atau titik untuk alasan keamanan. Di bagian bawah, terdapat tombol "*Login*" yang berfungsi untuk mengirimkan data yang telah dimasukkan ke sistem untuk proses verifikasi. Jika *username* dan *password* sesuai, pengguna akan diarahkan ke halaman berikutnya, sedangkan jika tidak sesuai, sistem biasanya akan menampilkan pesan kesalahan.

3.6.2 *Homepage*



Gambar 3.8 *Homepage*

Menampilkan halaman rekomendasi film dengan tampilan sederhana dan terstruktur. Di bagian atas terdapat kolom pencarian yang memungkinkan pengguna untuk mencari film atau drama tertentu dengan memasukkan kata kunci. Di sebelahnya, terdapat ikon pencarian yang biasanya berbentuk kaca pembesar, yang ketika diklik akan memproses pencarian berdasarkan input pengguna.

Di bawah kolom pencarian, terdapat judul "*Rekomendasi Film*" yang menunjukkan bahwa bagian ini berisi daftar film yang disarankan untuk pengguna.

Film-film ini kemungkinan besar dipilih berdasarkan riwayat tontonan, *rating*, atau sistem rekomendasi berbasis algoritma seperti collaborative filtering.

Tata letak film disusun dalam format *grid*, dengan beberapa kotak yang mewakili poster film atau drama yang direkomendasikan. Setiap kotak kemungkinan berisi gambar film, judul, dan mungkin informasi tambahan seperti genre atau rating. Format *grid* ini dirancang untuk memanfaatkan ruang secara efisien, sehingga pengguna dapat dengan mudah melihat beberapa pilihan film dalam satu layar.

Secara keseluruhan, halaman ini bertujuan untuk memberikan pengalaman pengguna yang lebih baik dengan menampilkan rekomendasi film yang sesuai dengan preferensi mereka.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil Implementasi

Penelitian ini menghasilkan sebuah sistem rekomendasi film berbasis *Collaborative Filtering* dengan algoritma *Matrix Factorization* menggunakan teknik *Singular Value Decomposition* (SVD). Dataset yang digunakan merupakan simulasi interaksi pengguna terhadap 30 film populer di *platform* Viu, dengan 200 pengguna sebagai partisipan. Sistem dibangun menggunakan bahasa pemrograman Python dengan framework Flask untuk pengembangan backend, serta antarmuka web sederhana sebagai frontend.

4.2 Analisis Kepadatan dan Kekosongan Data *Rating*

Kualitas prediksi sistem rekomendasi film berbasis kerja sama filter sangat bergantung pada seberapa lengkap data interaksi antara pengguna dan item tertentu, dalam hal ini film. Hasil pengolahan data menunjukkan bahwa dataset memiliki jumlah total rating sebanyak 1273, yang berasal dari 200 pengguna yang telah memberikan penilaian terhadap berbagai film, dengan total 30 film yang tersedia di *platform* penelitian. Oleh karena itu, total kemungkinan interaksi ($user \times item$) adalah 6000. Tingkat kekosongan data dalam matriks rating dapat dihitung dengan membandingkan jumlah rating aktual sebanyak 1273 dapat dihitung tingkat *sparsity* atau kekosongan data dalam matriks *rating*. *Sparsity* menunjukkan seberapa banyak bagian dari matriks yang tidak terisi, dan dalam kasus ini diperoleh nilai *sparsity* sebesar 0,7878 atau 78,78%. Untuk menghitung tingkat kekosongan (*sparsity*) dalam matriks *rating*, digunakan rumus pada **persamaan 10**.

$$Sparsity = 1 - \frac{\text{Jumlah Data Rating}}{\text{Jumlah Pengguna} \times \text{Jumlah Film}}$$

$$= 1 - \frac{1273}{200 \times 30} = 1 - \frac{1273}{6000} = 1 - 0.21216$$

$$= 0.7878$$

Dengan kata lain, sekitar tiga perempat dari film dan data interaksi pengguna tidak tersedia. Ini menunjukkan bahwa matriks penilaian sangat sparse karena sebagian besar pengguna belum memberikan penilaian terhadap banyak film yang tersedia. Karena data yang tidak lengkap dapat menurunkan akurasi prediksi, keadaan ini adalah masalah umum saat membuat sistem rekomendasi. Oleh karena itu, untuk menyelesaikan masalah ini, algoritma *Matrix Factorization* dengan teknik *Singular Value Decomposition* (SVD) digunakan. teknik ini memiliki kemampuan untuk memanfaatkan pola tersembunyi (fitur laten) dalam data yang jarang sekalipun dan memberikan prediksi rating yang lebih relevan.

4.3 Perhitungan Matriks *Singular Value Decomposition* (SVD) dan *Matrix Factorization* (MF)

Setelah matriks *rating* dari pengguna terhadap sejumlah film berhasil dibentuk, tahap selanjutnya dalam penelitian ini adalah melakukan proses dekomposisi menggunakan teknik *Singular Value Decomposition* (SVD). Proses ini bertujuan untuk memperoleh representasi dari masing-masing pengguna dan film dalam bentuk vektor laten, yang nantinya akan digunakan untuk memprediksi nilai *rating* yang belum diberikan.

4.3.1 Dekomposisi Matriks *Rating* Menggunakan *Singular Value Decomposition* (SVD)

Pada tahap ini data yang sudah dinormalisasi diubah menjadi matriks dengan kolom merepresentasikan setiap item dan baris merepresentasikan setiap user.

Selanjutnya matriks tersebut dapat dipecah menjadi tiga matriks, yaitu matriks U , matriks S , dan matriks V^T . Berikut adalah sampel data dengan user yang belum memberi rating pada film tertentu yang akan digunakan dalam pembuatan algoritma Matrix Factorization dengan teknik SVD:

Tabel 4.3 Data *Sample User - Rating*

	M5	M6	M9	M10	M11	M12	M17	M19	M28	M29
U2	2		5	1		3	2		2	5
U5		4		2			3		3	
U8	2		3		1		4	2		3
U23		4		4		2		3		
U26			1		5		4	5		5
U40	4			5		3			4	5
U56		5	5		4		4		2	
U64	4	5		5		1		5		5
U78		2		5	3		3		4	
U93	2			5			2		2	1

Keterangan :

M5, M6, M9, M10, M11, M12, M17, M19, M28, M29 = data movie atau film U2,

U5, U8, U23, U26, U40, U56, U64, U78, U93 = data user

Nilai pada tabel diatas merupakan rating belum yang diberikan user pada film tertentu.

Tahap pertama adalah mengubah data pada tabel 4.3 ke dalam bentuk matriks:

$$A = \begin{bmatrix} 2 & 0 & 5 & 1 & 0 & 3 & 2 & 0 & 2 & 5 \\ 0 & 4 & 0 & 2 & 0 & 0 & 3 & 0 & 3 & 0 \\ 2 & 0 & 3 & 0 & 1 & 0 & 4 & 2 & 0 & 3 \\ 0 & 4 & 0 & 4 & 0 & 2 & 0 & 3 & 0 & 0 \\ 0 & 0 & 1 & 0 & 5 & 0 & 4 & 5 & 0 & 5 \\ 4 & 0 & 0 & 5 & 0 & 3 & 0 & 0 & 4 & 5 \\ 0 & 5 & 5 & 0 & 4 & 0 & 4 & 0 & 2 & 0 \\ 4 & 5 & 0 & 5 & 0 & 1 & 0 & 5 & 0 & 5 \\ 0 & 2 & 0 & 5 & 3 & 0 & 3 & 0 & 4 & 0 \\ 2 & 0 & 0 & 5 & 0 & 0 & 2 & 0 & 2 & 1 \end{bmatrix}$$

Selanjutnya matriks A didekomposisikan menjadi 3 matriks, yaitu matriks U , matriks S , dan matriks V^T . Untuk memperoleh matriks U harus mencari nilai *eigenvector* dari matriks AA^T .

$$AA^T = \begin{bmatrix} 72 & 14 & 42 & 10 & 38 & 55 & 37 & 41 & 19 & 22 \\ 14 & 38 & 12 & 24 & 12 & 22 & 38 & 30 & 39 & 22 \\ 42 & 12 & 43 & 6 & 49 & 23 & 35 & 33 & 15 & 15 \\ 10 & 0 & 6 & 29 & 15 & 26 & 20 & 57 & 28 & 20 \\ 38 & 12 & 49 & 15 & 92 & 25 & 41 & 50 & 27 & 13 \\ 55 & 22 & 23 & 26 & 25 & 91 & 8 & 69 & 41 & 46 \\ 37 & 38 & 35 & 20 & 41 & 8 & 86 & 25 & 42 & 12 \\ 41 & 30 & 33 & 57 & 50 & 69 & 25 & 117 & 35 & 38 \\ 19 & 39 & 15 & 28 & 27 & 41 & 42 & 35 & 63 & 39 \\ 22 & 22 & 15 & 20 & 13 & 46 & 12 & 38 & 39 & 38 \end{bmatrix}$$

Nilai *eigenvalue* dan *eigenvector* dari matriks AA^T

Keterangan:

v : *eigenvector* (User)

λ : *eigenvalue*

$$v_1 = \begin{bmatrix} 0.326 \\ 0.217 \\ 0.250 \\ 0.233 \\ 0.337 \\ 0.389 \\ 0.294 \\ 0.477 \\ 0.305 \\ 0.242 \end{bmatrix}, \lambda_1 = 352.666$$

$$v_2 = \begin{bmatrix} 0.133 \\ 0.026 \\ 0.300 \\ -0.204 \\ 0.434 \\ -0.430 \\ 0.559 \\ -0.336 \\ -0.014 \\ -0.225 \end{bmatrix}, \lambda_2 = 112.770$$

$$v_3 = \begin{bmatrix} 0.133 \\ 0.026 \\ 0.300 \\ -0.204 \\ 0.434 \\ -0.430 \\ 0.559 \\ -0.336 \\ -0.014 \\ -0.225 \end{bmatrix}, \lambda_3 = 88.408$$

$$v_4 = \begin{bmatrix} 0.133 \\ 0.026 \\ 0.300 \\ -0.204 \\ 0.434 \\ -0.430 \\ 0.559 \\ -0.336 \\ -0.014 \\ -0.225 \end{bmatrix}, \lambda_4 = 66.148$$

$$v_5 = \begin{bmatrix} 0.133 \\ 0.026 \\ 0.300 \\ -0.204 \\ 0.434 \\ -0.430 \\ 0.559 \\ -0.336 \\ -0.014 \\ -0.225 \end{bmatrix}, \lambda_5 = 38.650$$

Setelah mendapatkan nilai *eigenvalue*, maka didapat matriks *S* dengan nilai akar *eigenvalue* menjadi nilai matriks diagonal *S* berurutan dari yang terbesar ke yang terkecil.

$$S = \begin{bmatrix} \sqrt{352.666} & 0 & 0 & 0 & 0 \\ 0 & \sqrt{112.770} & 0 & 0 & 0 \\ 0 & 0 & \sqrt{88.408} & 0 & 0 \\ 0 & 0 & 0 & \sqrt{66.148} & 0 \\ 0 & 0 & 0 & 0 & \sqrt{38.650} \end{bmatrix}$$

$$S = \begin{bmatrix} 18.779 & 0 & 0 & 0 & 0 \\ 0 & 10.619 & 0 & 0 & 0 \\ 0 & 0 & 9.403 & 0 & 0 \\ 0 & 0 & 0 & 8.133 & 0 \\ 0 & 0 & 0 & 0 & 6.217 \end{bmatrix}$$

$$\sqrt{\Sigma} = \begin{bmatrix} \sqrt{18.779} \\ \sqrt{10.619} \\ \sqrt{9.403} \\ \sqrt{8.133} \\ \sqrt{6.217} \end{bmatrix} = \begin{bmatrix} 4.333 \\ 3.259 \\ 3.066 \\ 2.852 \\ 2.493 \end{bmatrix}$$

Dengan memakai **persamaan 3** dan **persamaan 4**, maka bisa mendapatkan matriks U (*eigenvector* yang dinormalisasikan) :

$$U = \begin{bmatrix} U_2 & 0.326 & 0.133 & -0.353 & -0.458 & 0.400 \\ U_5 & 0.217 & 0.026 & 0.422 & -0.039 & 0.065 \\ U_8 & 0.250 & 0.300 & -0.258 & -0.049 & 0.011 \\ U_{23} & 0.233 & -0.204 & 0.242 & 0.385 & 0.219 \\ U_{26} & 0.337 & 0.434 & -0.347 & 0.338 & -0.549 \\ U_{40} & 0.389 & -0.430 & -0.195 & -0.407 & -0.095 \\ U_{56} & 0.294 & 0.559 & 0.391 & -0.071 & 0.361 \\ U_{64} & 0.477 & -0.336 & -0.140 & 0.527 & 0.258 \\ U_{78} & 0.305 & -0.014 & 0.472 & -0.176 & -0.450 \\ U_{93} & 0.242 & -0.225 & 0.134 & -0.212 & -0.280 \end{bmatrix}$$

Selanjutnya menghitung matriks V^T

$$S^{-1} = \begin{bmatrix} \frac{1}{18.779} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{10.619} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{9.403} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{8.133} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{6.217} \end{bmatrix}$$

$$S^{-1} = \begin{bmatrix} 0.053 & 0 & 0 & 0 & 0 \\ 0 & 0.094 & 0 & 0 & 0 \\ 0 & 0 & 0.106 & 0 & 0 \\ 0 & 0 & 0 & 0.123 & 0 \\ 0 & 0 & 0 & 0 & 0.161 \end{bmatrix}$$

$$U^T = \begin{bmatrix} U_2 & U_5 & U_8 & U_{23} & U_{26} & U_{40} & U_{56} & U_{64} & U_{78} & U_{93} \\ 0.326 & 0.217 & 0.250 & 0.233 & 0.337 & 0.389 & 0.294 & 0.477 & 0.305 & 0.242 \\ 0.133 & 0.026 & 0.300 & -0.204 & 0.434 & -0.430 & 0.559 & -0.336 & -0.014 & -0.225 \\ -0.353 & 0.422 & -0.258 & 0.242 & -0.347 & -0.195 & 0.391 & -0.140 & 0.472 & 0.134 \\ -0.458 & -0.039 & -0.049 & 0.385 & 0.338 & -0.407 & -0.071 & 0.527 & -0.176 & -0.212 \\ 0.400 & 0.065 & 0.011 & 0.219 & -0.549 & -0.095 & 0.361 & 0.258 & -0.450 & -0.280 \end{bmatrix}$$

$$A = \begin{bmatrix} 2 & 0 & 5 & 1 & 0 & 3 & 2 & 0 & 2 & 5 \\ 0 & 4 & 0 & 2 & 0 & 0 & 3 & 0 & 3 & 0 \\ 2 & 0 & 3 & 0 & 1 & 0 & 4 & 2 & 0 & 3 \\ 0 & 4 & 0 & 4 & 0 & 2 & 0 & 3 & 0 & 0 \\ 0 & 0 & 1 & 0 & 5 & 0 & 4 & 5 & 0 & 5 \\ 4 & 0 & 0 & 5 & 0 & 3 & 0 & 0 & 4 & 5 \\ 0 & 5 & 5 & 0 & 4 & 0 & 4 & 0 & 2 & 0 \\ 4 & 5 & 0 & 5 & 0 & 1 & 0 & 5 & 0 & 5 \\ 0 & 2 & 0 & 5 & 3 & 0 & 3 & 0 & 4 & 0 \\ 2 & 0 & 0 & 5 & 0 & 0 & 2 & 0 & 2 & 1 \end{bmatrix}$$

Selanjutnya hasil perkalian dari matriks S^{-1} , U^T , A sehingga didapatkan matriks

V^T :

$$V^T = \begin{bmatrix} M_5 & M_6 & M_9 & M_{10} & M_{11} & M_{12} & M_{17} & M_{19} & M_{28} & M_{29} \\ 0.271 & 0.333 & 0.223 & 0.466 & 0.214 & 0.164 & 0.331 & 0.280 & 0.274 & 0.460 \\ -0.250 & 0.035 & 0.451 & -0.533 & 0.439 & -0.154 & 0.473 & 0.045 & -0.072 & -0.031 \\ -0.244 & 0.516 & -0.099 & 0.299 & 0.105 & -0.139 & 0.148 & -0.237 & 0.289 & -0.619 \\ -0.118 & 0.407 & -0.302 & -0.041 & 0.102 & -0.160 & -0.137 & 0.662 & -0.483 & -0.044 \\ 0.147 & 0.535 & 0.528 & -0.230 & -0.425 & 0.259 & -0.262 & -0.125 & -0.165 & -0.029 \end{bmatrix}$$

$$V = \begin{bmatrix} M_5 & 0.271 & -0.250 & -0.244 & -0.118 & 0.147 \\ M_6 & 0.333 & 0.035 & 0.516 & 0.407 & 0.535 \\ M_9 & 0.223 & 0.451 & -0.099 & -0.302 & 0.528 \\ M_{10} & 0.466 & -0.533 & 0.299 & -0.041 & -0.230 \\ M_{11} & 0.214 & 0.439 & 0.105 & 0.102 & -0.425 \\ M_{12} & 0.164 & -0.154 & -0.139 & -0.160 & 0.259 \\ M_{17} & 0.331 & 0.473 & 0.148 & -0.137 & -0.262 \\ M_{19} & 0.280 & 0.045 & -0.237 & 0.662 & -0.125 \\ M_{28} & 0.274 & -0.072 & 0.289 & -0.483 & -0.165 \\ M_{29} & 0.460 & -0.031 & -0.619 & -0.044 & -0.029 \end{bmatrix}$$

Setelah mendapatkan semua matriks yang dibutuhkan, selanjutnya adalah melakukan pengujian prediksi *rating* dengan algoritma *Matrix Factorization*.

4.3.2 Transformasi SVD ke *Matrix Factorization* (MF)

Untuk prediksi *rating* secara manual menggunakan pendekatan MF, dilakukan transformasi menjadi P (*User*) dan Q (*Item*). Berikut adalah contoh perhitungan dengan menggunakan algoritma *Matrix Factorization*:

Percobaan pada U2 dengan M6

$$\begin{aligned} P &= U_2 \cdot \sqrt{\Sigma} = [0.326 \cdot 4.333, 0.133 \cdot 3.259, -0.353 \cdot 3.066, -0.458 \cdot 2.852, 0.400 \cdot 2.493] \\ &= [1.412, 0.433, -1.082, -1.306, 0.997] \end{aligned}$$

$$\begin{aligned} Q &= V_6 \cdot \sqrt{\Sigma} = [0.333 \cdot 4.333, 0.035 \cdot 3.259, 0.516 \cdot 3.066, 0.407 \cdot 2.852, 0.535 \cdot 2.493] \\ &= [1.443, 0.114, 1.582, 1.161, 1.334] \end{aligned}$$

$$\begin{aligned} \hat{r}_{U_2 V_6} &= P_{U_2} \cdot Q_{M_6}^T = (1.412)(1.443) + (0.433)(0.114) + (-1.082)(1.582) + (-1.306)(1.161) + \\ &\quad (0.997)(1.334) = 2.037 + 0.049 - 1.711 - 1.516 + 1.334 = 0.189 \end{aligned}$$

Jadi, untuk prediksi rating pada User-2 untuk Movie-6 adalah **0.189**

Percobaan pada U5 dengan M5

$$\begin{aligned} P &= U_5 \cdot \sqrt{\Sigma} = [0.217 \cdot 4.333, 0.026 \cdot 3.259, 0.422 \cdot 3.066, -0.039 \cdot 2.852, 0.065 \cdot 2.493] \\ &= [0.940, 0.085, 1.294, -0.111, 0.162] \end{aligned}$$

$$\begin{aligned} Q &= V_5 \cdot \sqrt{\Sigma} = [0.271 \cdot 4.333, -0.250 \cdot 3.259, -0.244 \cdot 3.066, -0.118 \cdot 2.852, 0.147 \cdot 2.493] \\ &= [1.443, 0.114, 1.582, 1.161, 1.334] \end{aligned}$$

$$\begin{aligned} \hat{r}_{U_5 V_5} &= P_{U_5} \cdot Q_{M_5}^T = (0.940)(1.174) + (0.085)(-0.815) + (1.294)(-0.748) + (-0.111)(-0.337) + \\ &\quad (0.162)(0.366) = 2.039 + 0.049 - 1.711 - 1.516 + 1.334 = 0.164 \end{aligned}$$

Jadi, untuk prediksi rating pada User-5 untuk Movie-5 adalah **0.164**

Percobaan pada U8 dengan M10

$$\begin{aligned} P &= U_8 \cdot \sqrt{\Sigma} = [0.250 \cdot 4.333, 0.300 \cdot 3.259, -0.258 \cdot 3.066, -0.049 \cdot 2.852, 0.011 \cdot 2.493] \\ &= [1.083, 0.978, -0.791, -0.140, 0.027] \end{aligned}$$

$$\begin{aligned} Q &= V_{10} \cdot \sqrt{\Sigma} = [0.466 \cdot 4.333, -0.533 \cdot 3.259, 0.299 \cdot 3.066, -0.041 \cdot 2.852, -0.230 \cdot 2.493] \\ &= [2.020, -1.737, 0.916, -0.117, -0.573] \end{aligned}$$

$$\begin{aligned} \hat{r}_{U_8 V_{10}} &= P_{U_8} \cdot Q_{M_{10}}^T = (0.940)(1.174) + (0.085)(-0.815) + (1.294)(-0.748) + (-0.111)(-0.337) + \\ &\quad (0.027)(-0.573) = 2.188 - 1.699 - 0.725 + 0.016 - 0.015 = -0.235 \end{aligned}$$

Jadi, untuk prediksi rating pada User-8 untuk Movie-10 adalah **-0.235**

Setelah selesai melakukan perhitungan dengan algoritma *Matrix Factorization*, berikut adalah hasil nilai prediksi *rating* pada tabel 4.4

Tabel 4.4 Nilai Prediksi *Rating*

	M5	M6	M9	M10	M11	M12	M17	M19	M28	M29
U2	2	0.189	5	1	0.145	3	2	-0.213	2	5
U5	0.164	4	0.949	2	1.206	0.230	3	-0.047	3	-0.588
U8	2	0.297	3	-0.236	1	0.698	4	2	0.537	3
U23	1.003	4	-0.454	4	-0.035	2	-0.026	3	0.275	0.494
U26	0.532	-0.121	1	0.188	5	-0.542	4	5	-0.306	5
U40	4	-0.337	0.438	5	-0.721	3	0.594	0.156	4	5
U56	-0.487	5	5	0.014	4	0.154	4	0.279	2	0.040
U64	4	5	0.070	5	-0.033	1	0.074	5	-0.005	5
U78	0.264	2	1.209	5	3	-0.150	3	-0.053	4	0.036
U93	2	0.446	-0.587	5	0.619	0.763	2	-0.058	2	1

4.4 Penyesuaian Nilai Prediksi (*Clipping*)

Setelah proses prediksi *rating* dilakukan, terdapat beberapa nilai yang tidak sesuai dengan skala sistem, seperti nilai negatif atau lebih dari 5. Nilai-nilai ini tentu tidak bisa langsung digunakan karena di luar batas penilaian yang diterapkan. Untuk mengatasi hal tersebut, digunakan teknik *clipping*, yaitu metode pemangkasan nilai agar tetap berada dalam rentang yang valid. Dalam hal ini, setiap nilai prediksi > 1 akan disesuaikan menjadi 1, sedangkan nilai < 5 akan dikembalikan ke angka 5. Sementara itu, nilai yang berada di antara 1 hingga 5 tetap dipertahankan tanpa perubahan. Proses ini bertujuan agar hasil prediksi tetap relevan, mudah dipahami oleh pengguna, dan sesuai dengan skala sistem rekomendasi yang digunakan.

Tabel 4.5 Data Nilai *Clipping*

	M5	M6	M9	M10	M11	M12	M17	M19	M28	M29
U2	2	1	5	1	1	3	2	1	2	5
U5	1	4	1	2	1.206	1	3	1	3	1
U8	2	1	3	1	1	1	4	2	1	3
U23	1.003	4	1	4	1	2	1	3	1	1
U26	1	1	1	1	5	1	4	5	1	5
U40	4	1	1	5	1	3	1	1	4	5
U56	1	5	5	1	4	1	4	1	2	1
U64	4	5	1	5	1	1	1	5	1	5
U78	1	2	1	5	3	1	3	1	4	1
U93	2	1	1	5	1	1	2	1	2	1

4.5 Evaluasi Akurasi Prediksi Menggunakan *Mean Absolute Error* (MAE)

Evaluasi dilakukan untuk mengetahui seberapa akurat prediksi rating yang dihasilkan oleh algoritma *Matrix Factorization* (MF). Dalam pelaksanaan pengujian, sebagian data dengan nilai rating asli diambil secara acak sebanyak 10 pasang *user-item* untuk digunakan sebagai data uji / *testing*. Nilai *rating* pada pasangan tersebut dihapus dari data latih, dan kemudian diprediksi kembali untuk menghitung tingkat akurasi.

Berikut contoh perhitungan manual untuk evaluasi MAE:

U93 dengan M5

$$P = U_{93} \cdot \sqrt{\Sigma} = [0.242 \cdot 4.333, -0.225 \cdot 3.259, 0.134 \cdot 3.066, -0.212 \cdot 2.852, -0.280 \cdot 2.493]$$

$$= [1.048, -0.733, 0.411, -0.605, -0.698]$$

$$Q = V_5 \cdot \sqrt{\Sigma} = [0.271 \cdot 4.333, -0.250 \cdot 3.259, -0.244 \cdot 3.066, -0.118 \cdot 2.852, 0.147 \cdot 2.493]$$

$$= [1.175, -0.815, -0.750, -0.337, 0.366]$$

$$\hat{r}_{U_{93}V_5} = P_{U_{93}} \cdot Q_{M_5}^T = (1.048 \cdot 1.175) + (-0.733 \cdot -0.815) + (0.411 \cdot -0.750) + (-0.605 \cdot -0.337) +$$

$$(-0.698 \cdot 0.366) = 1.231 + 0.597 - 0.308 + 0.204 - 0.256 = 1.473$$

Berdasarkan data, nilai rating asli pada posisi tersebut adalah 2, sedangkan hasil prediksi dari *Matrix Factorization* dengan $k = 5$ adalah **1.473**. Maka, kesalahan absolut atau MAE-nya dihitung dengan **persamaan 8** :

$$MAE = \frac{|2 - 1.473|}{1} = 0.527$$

Berdasarkan perhitungan di atas, dapat dilihat bahwa nilai MAE dari prediksi *rating user* - 93 terhadap *movie* - 5 adalah sebesar 0,527. Jika dibandingkan nilai MAE pada sistem yang bernilai 0.4551 tidak begitu jauh.

4.6 Perbandingan dengan Metode Tradisional Berbasis Memori

Penelitian ini tidak hanya menerapkan metode dengan pendekatan *Singular Value Decomposition* (SVD), tetapi juga membandingkan efektivitasnya dengan metode rekomendasi tradisional, yaitu *User-Based Collaborative Filtering* (*User-Based CF*). Tujuan dari perbandingan ini adalah untuk mengetahui sejauh mana pendekatan berbasis model seperti SVD dapat meningkatkan akurasi prediksi rating film.

Dari hasil evaluasi, diketahui bahwa metode *Matrix Factorization* (SVD) menghasilkan nilai *Mean Absolute Error* (MAE) sebesar 0.4551, sedangkan metode *User-Based CF* menghasilkan MAE sebesar 1.3615. Nilai MAE yang lebih rendah pada metode SVD menandakan bahwa hasil prediksinya lebih mendekati nilai rating asli yang diberikan oleh pengguna.

Tabel 4.6 Perbandingan Metode *Matrix Factorization* dengan *User-Based*

Metode	Pendekatan	MAE	Kelebihan
Matrix Factorization (SVD)	Berbasis model (SVD)	0.4551	Akurasi tinggi, mampu menangani data yang sparse
User-Based Collaborative Filter	Berbasis memori	1.3615	Sederhana, mudah dipahami namun kurang stabil

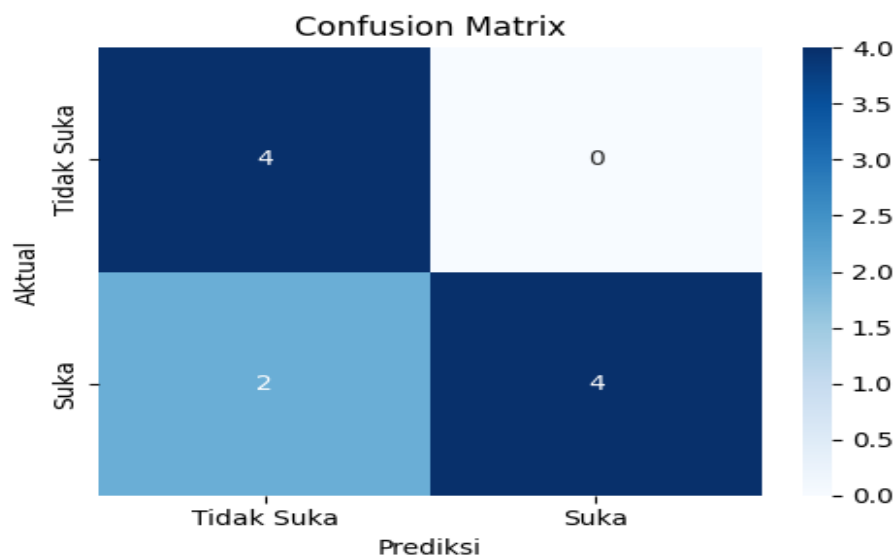
Metode tradisional berbasis memori menghitung kemiripan antar pengguna dari riwayat *rating*. Namun, pendekatan ini kurang efektif saat data banyak yang kosong (*sparse*), karena sulit menemukan kecocokan antar pengguna. Sebaliknya, *Matrix Factorization* (SVD) dapat mengatasi masalah tersebut dengan membentuk pola laten antara pengguna dan item melalui dekomposisi matriks. Hal ini membuat prediksi tetap akurat meskipun data terbatas. Karena itu, metode SVD lebih unggul dibanding metode tradisional, baik dari sisi akurasi maupun kemampuannya dalam menghadapi data yang tidak lengkap.

4.7 Melakukan Evaluasi dengan *Confusion Matrix*

Evaluasi model dilakukan menggunakan Confusion Matrix. Hasil confusion matrix dari sistem adalah sebagai berikut:

Tabel 4.7 Evaluasi dengan *Confusion Matrix*

	Prediksi: Suka	Prediksi: Tidak Suka
Aktual: Suka	TP = 4	FN = 2
Aktual: Tidak Suka	FP = 0	TN = 4



Gambar 4.7 Grafik *Confusion Matrix*

$$1. Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{4 + 4}{4 + 4 + 2 + 0} = \frac{8}{10} = 0,8$$

Artinya: 80% dari seluruh prediksi model adalah benar (baik yang suka maupun tidak suka).

$$2. Precision = \frac{TP}{TP + FP} = \frac{4}{4 + 0} = \frac{4}{4} = 1,0$$

Artinya: semua prediksi “suka” memang benar-benar suka, tidak ada yang salah (FP = 0).

$$3. Recall = \frac{TP}{TP + FN} = \frac{4}{4 + 2} = \frac{4}{6} = 0,667$$

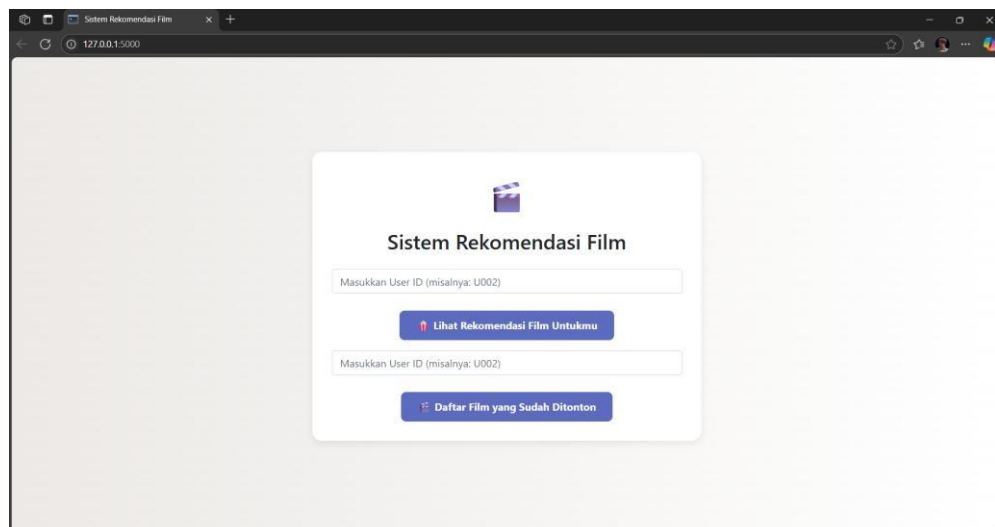
Artinya: dari semua yang seharusnya suka, hanya 66.67% berhasil diprediksi oleh model. Sisanya (FN = 2) lolos.

$$4. F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{1.0 \times 0.667}{1.0 + 0.667} = 0,8$$

F1-Score adalah rata-rata harmonis antara Precision dan Recall, seimbang antara kualitas prediksi dan kelengkapannya.

4.8 Hasil Pengujian Sistem pada Algoritma *Matrix Factorizaion* dengan Teknik *Singular Value Decomposition* (SVD)

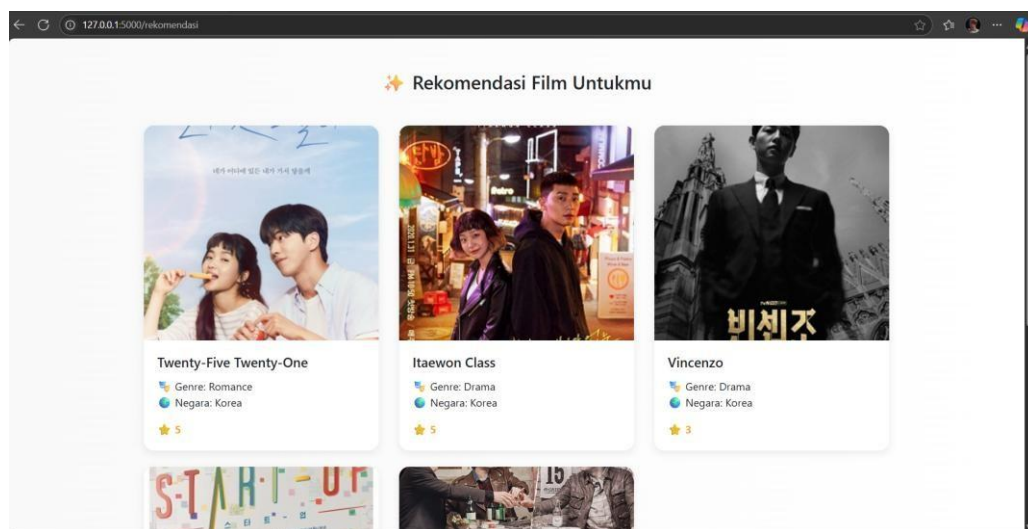
4.8.1 Halaman Awal User



Gambar 4.8 Halaman Awal User

Gambar tersebut menunjukkan tampilan utama dari program rekomendasi film. Pengguna hanya perlu memasukkan *User ID* satu kali untuk mengakses dua fitur utama, yaitu melihat rekomendasi film berdasarkan hasil prediksi dan melihat daftar film yang pernah ditonton. Desain halaman dibuat sederhana agar mudah digunakan, dan sistem akan menampilkan data sesuai dengan *User ID* yang dimasukkan. Contoh user yang digunakan adalah U002.

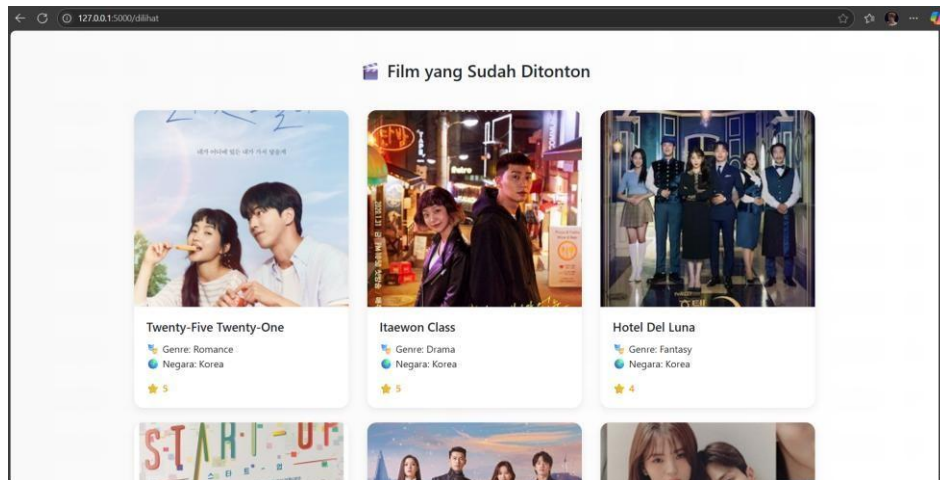
4.8.2 Halaman Rekomendasi *User*



Gambar 4.9 Halaman Rekomendasi *User*

Halaman ini menampilkan lima film rekomendasi untuk pengguna U002 berdasarkan data prediksi rating tertinggi dari hasil perhitungan *Matrix Factorization* (MF). Setiap film dilengkapi dengan judul, *genre*, negara asal, *rating* prediksi, dan poster film.

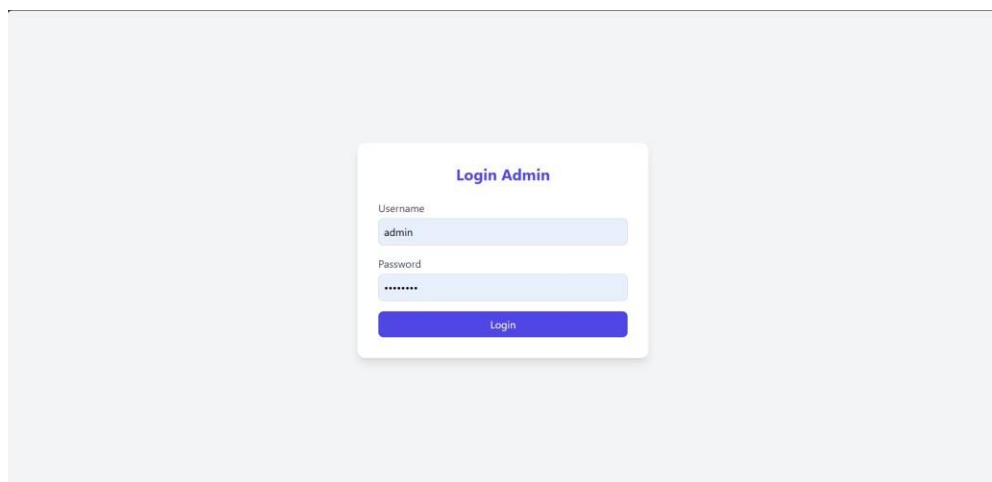
4.8.3 Halaman Film yang Sudah Ditonton



Gambar 5.0 Halaman Film yang Sudah Ditonton

Halaman ini menampilkan daftar film yang telah ditonton oleh pengguna U002, berdasarkan data *rating* yang diberikan. Setiap film ditampilkan dengan judul, *genre*, asal negara, *rating*, dan poster.

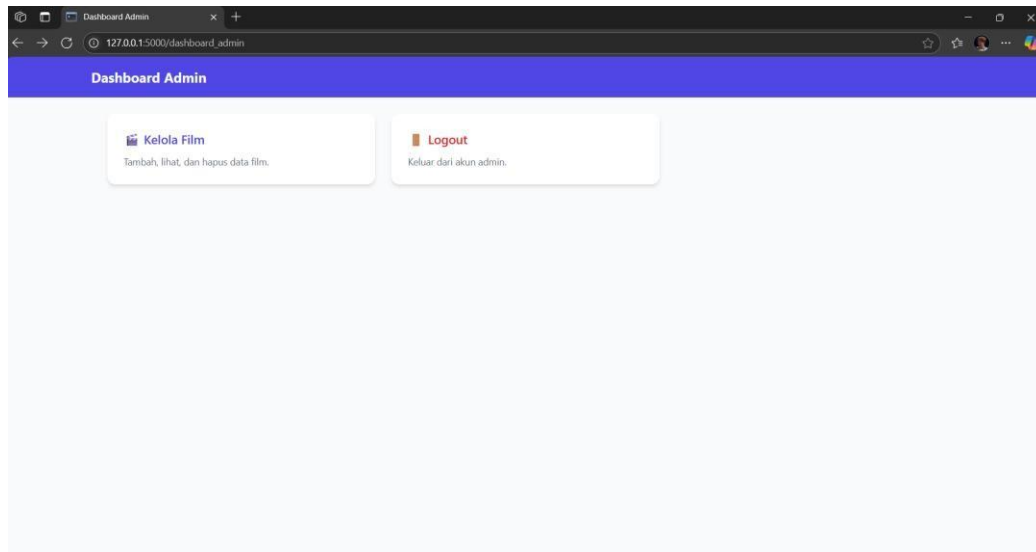
4.8.4 Halaman *Login Admin*



Gambar 5.1 Halaman *Login Admin*

Halaman ini merupakan pintu masuk bagi admin untuk mengakses sistem. Admin cukup memasukkan *username* dan *password* yang valid untuk *login* dan mengelola data yang ada.

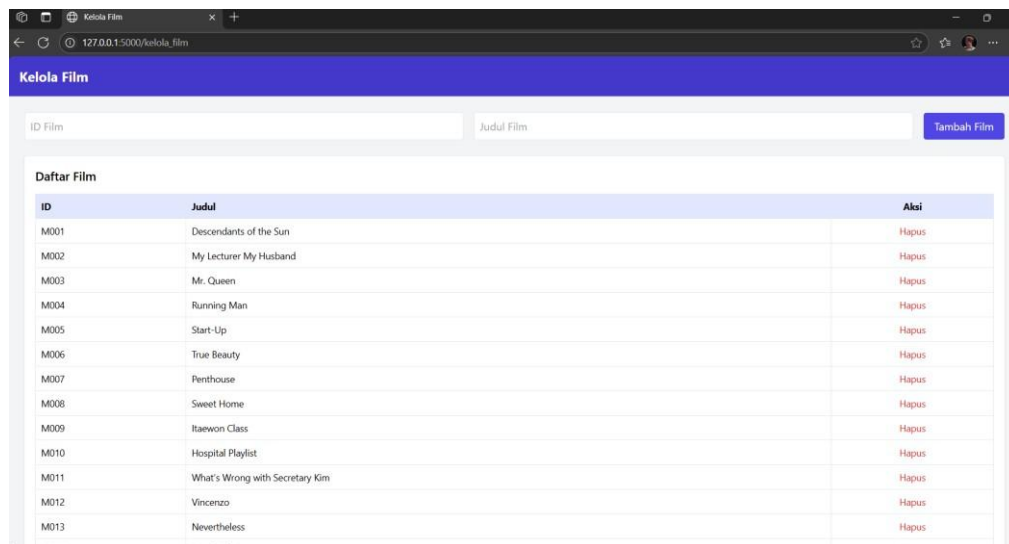
4.8.5 Halaman *Dashboard Admin*



Gambar 5.2 Halaman *Dasboard*

Halaman ini adalah beranda admin yang berisi dua menu utama: untuk mengelola data film dan tombol logout untuk keluar dari akun.

4.8.6 Halaman *Kelola Film*



Gambar 5.3 Halaman *Kelola Film*

Halaman ini digunakan untuk mengelola daftar film, memungkinkan admin menambah dan menghapus data berdasarkan ID dan judul film.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil dari penelitian dan implementasi sistem rekomendasi film menggunakan metode *Matrix Factorization* (MF) serta evaluasi menggunakan *Mean Absolute Error* (MAE) dan *confusion matrix*, dapat disimpulkan beberapa hal berikut:

1. Sistem rekomendasi berhasil memberikan prediksi *rating* film kepada pengguna berdasarkan pola penilaian sebelumnya yang diperoleh dari matriks *user-item*.
2. Tingginya tingkat *sparsity* menunjukkan bahwa sebagian besar pengguna belum memberi *rating* pada banyak film. Hal ini menyebabkan akurasi sistem rekomendasi bisa menurun. Untuk mengatasi kondisi ini, *Matrix Factorization* dengan teknik SVD digunakan karena mampu mengenali pola tersembunyi dan tetap memberikan prediksi yang baik meskipun data tidak lengkap.
3. Model yang dibangun dengan teknik *Matrix Factorization* dapat memprediksi nilai *rating* dengan tingkat akurasi yang cukup baik, yang ditunjukkan oleh nilai MAE bernilai 0.4551 tidak begitu jauh dengan nilai MAE yang pengerjaannya manual yang bernilai 0,527 yang relatif rendah dan metrik evaluasi klasifikasi (seperti *accuracy* bernilai 0.8, *precision* bernilai 1.0, *recall* bernilai 0.6667, dan *F1-score* bernilai 0.8) yang berada pada kisaran yang memuaskan.
4. *Matrix Factorization* (SVD) lebih akurat dibanding metode berbasis memori, dengan MAE lebih rendah (0.4551 dengan 1.3615). SVD tetap efektif meski data *rating* tidak lengkap, karena mampu mengenali pola laten antar pengguna dan *item*.

5.2 Saran

Adapun saran yang dapat diberikan untuk pengembangan penelitian di masa mendatang adalah:

1. Penelitian selanjutnya dapat mempertimbangkan penerapan metode rekomendasi lain, seperti pendekatan berbasis konten (*content-based*), hibrida (*hybrid*), atau pendekatan berbasis *deep learning*, guna meningkatkan akurasi dan relevansi hasil rekomendasi.
2. Perlu dilakukan pengujian pada dataset yang lebih besar dan lebih kompleks agar sistem dapat diuji ketahanannya pada skenario dunia nyata.
3. Evaluasi kinerja dapat diperluas dengan menambahkan metrik lain seperti *Root Mean Square Error* (RMSE) agar hasil evaluasi lebih komprehensif.

DAFTAR PUSTAKA

- Akbar, F., Mawengkang, H., & Efendi, S. (2018). Comparative analysis of RC4+ algorithm, RC4 NGG algorithm and RC4 GGHN algorithm on image file security. *IOP Conference Series: Materials Science and Engineering*, 420(1). <https://doi.org/10.1088/1757-899X/420/1/012131>
- Al Amin, A., Sunyoto, A., Al Fatta, H., & Yogyakarta, A. (2021). *Mereduksi Error Prediksi Pada Sistem Rekomendasi Menggunakan Pendekatan Colalborative Filtering Berbasis Model Matrix Factorization 8 Mereduksi Error Prediksi Pada Sistem Rekomendasi Menggunakan Pendekatan Collaborative Filtering Berbasis Model Matrix Factorization*.
- Arviyanto, F. B., & Soebroto, A. A. (2017). *Sistem Rekomendasi Model Sandal Pada Home Industri Menggunakan Collaborative Filtering dan Algoritma Matrix Factorization* (Vol. 1, Issue 1). <http://j-ptiik.ub.ac.id>
- Deshpande, M., & Karypis, G. (2004). *Item-Based Top-N Recommendation Algorithms*.
- Ellysinta, V., Vernando, W., & Kurniawan, K. (2020). Pengaruh Illegal Movie Streaming Terhadap Popularitas Film Bagi Mahasiswa. *Jurnal Teknologi Informasi*, 6. <http://ejournal.urindo.ac.id/index.php/TI>
- Fajar Yulia Fahmi, M., & Nugroho Bayu Aji, R. (2022). DINAMIKA PERFILMAN INDONESIA TAHUN (1940-1966). In *Journal Pendidikan Sejarah* (Vol. 12, Issue 3).
- Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2011). *Recommender Systems : An Introduction*.

- Koren, Y., Bell, R., & Chris, V. (2009). *MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS*.
- Mahendra Wiputra, M., & Jauhari Shandi, Y. (2021). PERANCANGAN SISTEM REKOMENDASI MENGGUNAKAN METODE COLLABORATIVE FILTERING DENGAN STUDI KASUS PERANCANGAN WEBSITE REKOMENDASI FILM. In *Media Informatika* (Vol. 20, Issue 1).
- Prayogo, J. E., Suharso, A., & Rizal, A. (2021). Analisis Perbandingan Model Matrix Factorization dan K-Nearest Neighbor dalam Mesin Rekomendasi Collaborative Berbasis Prediksi Rating. *Jurnal Informatika Universitas Pamulang*, 5(4), 506. <https://doi.org/10.32493/informatika.v5i4.7379>
- Quadrana, M., Cremonesi, P., & Jannach, D. (2018). *Sequence-Aware Recommender Systems*. <http://arxiv.org/abs/1802.08452>
- Ricci, F., Lior, ., Bracha, R., & Editors, S. (2011). *Recommender Systems Handbook Second Edition*. <https://doi.org/10.1007/978-1-4899-7637-6>
- Ryana Agustian, E., & Prasetyo Nugroho, E. (2020). *Sistem Rekomendasi Film Menggunakan Metode Collaborative Filtering dan K-Nearest Neighbors Film Recommendation System Using Collaborative Filtering Method and K-Nearest Neighbors* (Vol. 3, Issue 1). <https://ejournal.upi.edu/index.php/JATIKOM>
- Septiansyah, A., Hasanah, S., Nita Permatasari, V., & Yuliawati, A. (2024). *SISTEM INFORMASI OTOMATISASI PELAPORAN DATA PENJUALAN TOKO BUKU NAZWA YANG MASUK DAN YANG KELUAR*. https://doi.org/10.37817/ikraith_informatika.v8i1
- Sitanggang, A., Dongan Harahap, A., Karimullah, A., Dewantara, Y. A., Rozikin, C., Komputer, I., Karawang, S., Ronggo Waluyo, J. H., & Timur, T. (2023).

Sistem 72 Rekomendasi Anime Menggunakan Metode Singular Value Decomposition (SVD) dan Cosine Similarity. <http://jurnal.utu.ac.id/JTI>

Surendra, M., Babu, P., Raja, B., & Kumar, S. (2011). *An Implementation of the User based Collaborative Filtering Algorithm.*

Surya Ningsih, K., Jamilah Aruan, N., & Taufik Al Afkari Siahaan, A. (2022).

APLIKASI BUKU TAMU MENGGUNAKAN FITUR KAMERA DAN AJAX BERBASIS WEBSITE PADA KANTOR DISPORA KOTA MEDAN. *Jurnal Sains, Informatika, Dan Teknologi.*

Wibawa, A. E. Y. (2021). *IMPLEMENTASI PLATFORM DIGITAL SEBAGAI MEDIA PEMBELAJARAN DARING DI MI MUHAMMADIYAH PK KARTASURA PADA MASA PANDMEI COVID-19.* 1, 78.

<https://doi.org/10.47353/bj.v1i2.15>

Wibowo Okta, T. (2018). Fenomena Website Streaming Film di Era Media Baru: Godaan, Perselisihan, dan Kritik. *Jurnal Kajian Komunikasi*, 6, 191–203.

<https://doi.org/10.24198/jkk.v6i2.15623>

Zalukhu, A., Purba, S., Darma, D., Zalukhu1, A., Purba2, S., Darma3, D., Teknik Informatika, M., & Industri, F. T. (2023). PERANGKAT LUNAK APLIKASI PEMBELAJARAN FLOWCHART. *Jurnal Teknologi Informasi Dan Industri*, 4(1).

LAMPIRAN

Surat Penetapan Dosen Pembimbing



UMSU
Unggul | Cerdas | Terpercaya

Bila memaknai surat ini agar di sebarkan
nomor dan tanggalnya

MAJELIS PENDIDIKAN TINGGI PENELITIAN & PENGEMBANGAN PIMPINAN PUSAT MUHAMMADIYAH
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI

UMSU Terakreditasi A Berdasarkan Keputusan Badan Akreditasi Nasional Perguruan Tinggi No. 89/SK/BAN-PT/Akred/PT/III/2019
Pusat Administrasi: Jalan Mukhtar Basri No. 3 Medan 20238 Telp. (061) 6622400 - 66224567 Fax. (061) 6625474 - 6631003
<https://fkti.umsu.ac.id> fkti@umsu.ac.id [umsu.medan](https://www.facebook.com/umsu.medan) [umsu.medan](https://www.instagram.com/umsu.medan) [umsu.medan](https://www.youtube.com/channel/UC...) [umsu.medan](https://www.tiktok.com/@umsu.medan)

PENETAPAN DOSEN PEMBIMBING
PROPOSAL/SKRIPSI MAHASISWA
NOMOR : 247/II.3-AU/UMSU-09/F/2025

Assalamu'alaikum Warahmatullahi Wabarakatuh

Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Muhammadiyah Sumatera Utara, berdasarkan Persetujuan permohonan judul penelitian Proposal / Skripsi dari Ketua / Sekretaris.

Program Studi : Teknologi Informasi
Pada tanggal : 04 Februari 2025

Dengan ini menetapkan Dosen Pembimbing Proposal / Skripsi Mahasiswa.

Nama : Farah Zhafira Munthe
NPM : 2109020072
Semester : VII (Tujuh)
Program studi : Teknologi Informasi
Judul Proposal / Skripsi : Sistem Rekomendasi Film Bebas Collaborative Filtering Menggunakan Algoritma Matrix Factorization (Studi Kasus: Platform Filmzie)

Dosen Pembimbing : Farid Akbar Siregar, S.Kom., M.Kom.

Dengan demikian di izinkan menulis Proposal / Skripsi dengan ketentuan

1. Penulisan berpedoman pada buku panduan penulisan Proposal / Skripsi Fakultas Ilmu Komputer dan Teknologi Informasi UMSU
2. Pelaksanaan Sidang Skripsi harus berjarak 3 bulan setelah dikeluarkannya Surat Penetapan Dosen Pembimbing Skripsi.
3. **Proyek Proposal / Skripsi dinyatakan "BATAL" bila tidak selesai sebelum Masa Kadalua tanggal : 04 Februari 2026**
4. Revisi judul.....

Wassalamu'alaikum Warahmatullahi Wabarakatuh.

Ditetapkan di : Medan
Pada Tanggal : 05 Sya'ban 1446 H
04 Februari 2025M




Dekan
Dr. M. Khawarizmi, M.Kom.
NIDN : 0127099201

Cc. File



Scanned with CamScanner



Surat Acara Pembimbingan Skripsi



MAJELIS PENDIDIKAN TINGGI PENELITIAN & PENGEMBANGAN PIMPINAN PUSAT MUHAMMADIYAH
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
 UMSU Terakreditasi A Berdasarkan Keputusan Badan Akreditasi Nasional Perguruan Tinggi No. 89/SK/DAN-PT/Akred/PT/III/2019
 Pusat Administrasi: Jalan Mukhtar Basri No. 3 Medan 20238 Telp. (061) 6622400 - 66224567 Fax. (061) 6625474 - 6631003
 https://id.umsu.ac.id Email: info@umsu.ac.id umsumedan umsumedan umsumedan umsumedan

Berita Acara Pembimbingan Skripsi

Nama Mahasiswa : Farah Zhafira Munthe Program Studi : Teknologi Informasi
 NPM : 2109020072 Konsentrasi :
 Nama Dosen Pembimbing : Farid Akbar Siregar, S.kom, M.kom Judul Penelitian : Sistem Rekomendasi Film Berbasis
 Collaborative Filtering menggunakan
 Algoritma Matrix
 Factorization (Studi Kasus
 Platform VU)

Item	Hasil Evaluasi	Tanggal	Paraf Dosen
Bab 1-3	Revisi Sempro Selesai	26/4/2025	Fl
Bab 4-5	Lanjut Bab 4 dan 5	26/4/2025	Fl
Bab 4	Bimbingan Bab 4, lanjut bab 5	10/6/2025	Fl
Bab 5	Bimbingan Bab 5	11/6/2025	Fl
Bab 4-5	Revisi Bab 4 dan Bab 5	11/6/2025	Fl
Bab 4	Uji coba web	12/6/2025	Fl
Bab 4	Revisi web (penambahan gambar dan UI)	12/6/2025	Fl
	Acc Sidang	12/6/2025	Fl

Diketahui oleh :
 Ketua Program Studi
 Teknologi Informasi
 (Farid Akbar Siregar, S.kom, M.kom)

Medan, 12 Juni 2025
 Disetujui oleh :
 Dosen Pembimbing
 (Farid Akbar Siregar, S.kom, M.kom)



Data Movie

<i>movie_id</i>	<i>title</i>	<i>genre</i>	<i>country</i>	<i>year</i>
M001	<i>Descendants of the Sun</i>	<i>Drama</i>	Korea	2016
M002	<i>My Lecturer My Husband</i>	<i>Romance</i>	Indonesia	2020
M003	<i>Mr. Queen</i>	<i>Comedy</i>	Korea	2021
M004	<i>Running Man</i>	<i>Reality Show</i>	Korea	2010
M005	<i>Start-Up</i>	<i>Drama</i>	Korea	2020
M006	<i>True Beauty</i>	<i>Romance</i>	Korea	2020
M007	<i>Penthouse</i>	<i>Drama</i>	Korea	2020
M008	<i>Sweet Home</i>	<i>Horror</i>	Korea	2020
M009	<i>Itaewon Class</i>	<i>Drama</i>	Korea	2020
M010	<i>Hospital Playlist</i>	<i>Drama</i>	Korea	2020
M011	<i>What's Wrong with Secretary Kim</i>	<i>Romance</i>	Korea	2018
M012	<i>Vincenzo</i>	<i>Drama</i>	Korea	2021
M013	<i>Nevertheless</i>	<i>Romance</i>	Korea	2021
M014	<i>Hotel Del Luna</i>	<i>Fantasy</i>	Korea	2019
M015	<i>The World of the Married</i>	<i>Drama</i>	Korea	2020
M016	<i>Tale of the Nine Tailed</i>	<i>Fantasy</i>	Korea	2020
M017	<i>Crash Landing on You</i>	<i>Romance</i>	Korea	2019
M018	<i>Extraordinary You</i>	<i>Fantasy</i>	Korea	2019
M019	<i>Doom at Your Service</i>	<i>Fantasy</i>	Korea	2021
M020	<i>Flower of Evil</i>	<i>Thriller</i>	Korea	2020
M021	<i>My ID is Gangnam Beauty</i>	<i>Romance</i>	Korea	2018
M022	<i>Because This Is My First Life</i>	<i>Romance</i>	Korea	2017
M023	<i>Kairos</i>	<i>Thriller</i>	Korea	2020
M024	<i>W: Two Worlds</i>	<i>Fantasy</i>	Korea	2016
M025	<i>While You Were Sleeping</i>	<i>Fantasy</i>	Korea	2017
M026	<i>Love Alarm</i>	<i>Romance</i>	Korea	2019
M027	<i>My Name</i>	<i>Action</i>	Korea	2021
M028	<i>Signal</i>	<i>Thriller</i>	Korea	2016
M029	<i>Twenty-Five Twenty-One</i>	<i>Romance</i>	Korea	2022
M030	<i>Parasite</i>	<i>Thriller</i>	Korea	2019

Data User dan Rating

<i>user_id</i>	<i>movie_id</i>	<i>rating</i>	<i>liked</i>
U001	M023	4	1
U001	M004	1	0
U001	M016	2	0
U001	M002	4	1
U001	M029	3	0
U001	M008	3	0
U001	M022	1	0
U001	M025	1	0
U001	M009	4	1
U001	M020	3	0
U002	M005	2	0
U002	M010	1	0
U002	M009	5	1
U002	M013	2	0
U002	M015	2	0
U002	M002	1	0
U002	M014	4	1
U002	M017	2	0
U002	M028	2	0
U002	M029	5	1
U003	M019	2	0
U003	M030	4	1
U003	M027	3	0
U003	M026	3	0
U003	M006	2	0
U004	M027	1	0
U004	M017	4	1
U004	M015	4	1
U004	M004	2	0
U004	M025	5	1
U004	M023	2	0
U004	M018	3	0
U004	M030	4	1
U004	M021	5	1
U005	M017	3	0
U005	M001	5	1
U005	M007	3	0
U005	M025	3	0
U005	M002	3	0
U005	M006	4	1
U005	M010	2	0
U005	M028	3	0

U006	M025	2	0
U006	M010	5	1
U006	M002	2	0
U006	M011	4	1
U006	M009	1	0
U006	M012	4	1
U006	M015	4	1
U007	M023	3	0
U007	M018	1	0
U007	M012	1	0
U007	M011	1	0
U007	M024	2	0
U007	M002	3	0
U008	M019	2	0
U008	M005	2	0
U008	M017	4	1
U008	M029	3	0
U008	M022	3	0
U008	M009	3	0
U008	M011	1	0
U008	M029	3	0
U009	M023	1	0
U009	M001	3	0
U009	M027	1	0
U009	M025	1	0
U009	M005	4	1
U009	M013	4	1
U009	M020	2	0
U009	M016	4	1
U009	M004	1	0
U010	M019	3	0
U010	M018	4	1
U010	M008	4	1
U010	M029	4	1
U010	M014	1	0
U010	M007	3	0
U010	M002	1	0
U011	M025	3	0
U011	M015	2	0
U011	M004	5	1
U011	M008	3	0
U011	M026	2	0
U012	M010	4	1
U012	M021	4	1
U012	M005	2	0

U012	M022	5	1
U012	M026	4	1
U012	M030	1	0
U013	M009	3	0
U013	M014	1	0
U013	M025	2	0
U013	M029	1	0
U013	M027	2	0
U013	M011	1	0
U013	M016	4	1
U013	M010	3	0
U014	M025	4	1
U014	M008	2	0
U014	M020	4	1
U014	M001	2	0
U014	M015	3	0
U014	M027	3	0
U014	M009	2	0
U014	M006	2	0
U014	M010	3	0
U015	M025	3	0
U015	M004	2	0
U015	M013	5	1
U015	M009	3	0
U015	M016	5	1
U015	M017	4	1
U015	M019	1	0
U016	M006	3	0
U016	M019	3	0
U016	M020	3	0
U016	M022	1	0
U016	M002	3	0
U016	M007	3	0
U016	M018	3	0
U017	M022	3	0
U017	M020	3	0
U017	M007	3	0
U017	M005	5	1
U017	M021	2	0
U017	M006	1	0
U017	M014	4	1
U017	M026	5	1
U017	M025	3	0
U017	M019	5	1
U018	M010	5	1

U018	M019	3	0
U018	M008	4	1
U018	M030	5	1
U018	M014	5	1
U019	M019	2	0
U019	M006	4	1
U019	M025	3	0
U019	M007	4	1
U019	M011	5	1
U019	M010	1	0
U019	M021	1	0
U019	M030	4	1
U019	M008	2	0
U019	M026	3	0
U020	M009	4	1
U020	M026	3	0
U020	M028	1	0
U020	M017	4	1
U020	M006	3	0
U020	M025	1	0
U020	M021	1	0
U021	M030	1	0
U021	M024	1	0
U021	M021	1	0
U021	M004	4	1
U021	M020	1	0
U021	M022	5	1
U021	M025	3	0
U022	M025	5	1
U022	M014	5	1
U022	M007	4	1
U022	M028	1	0
U022	M010	5	1
U022	M016	2	0
U022	M015	5	1
U022	M013	4	1
U023	M021	1	0
U023	M008	3	0
U023	M006	4	1
U023	M012	2	0
U023	M019	3	0
U023	M006	4	0
U023	M010	4	1
U024	M007	1	0
U024	M001	1	0

U024	M005	3	0
U024	M006	4	1
U024	M003	4	1
U024	M018	4	1
U025	M029	1	0
U025	M004	3	0
U025	M026	3	0
U025	M022	4	1
U025	M018	4	1
U025	M012	5	1
U025	M007	5	1
U025	M019	4	1
U025	M014	2	0
U025	M016	3	0
U026	M009	1	0
U026	M029	5	1
U026	M017	4	1
U026	M019	5	1
U026	M011	5	1
U026	M029	5	1
U027	M011	5	1
U027	M024	3	0
U027	M027	2	0
U027	M016	2	0
U027	M001	4	1
U027	M018	5	1
U027	M023	2	0
U027	M014	4	1
U027	M005	4	1
U027	M009	1	0
U028	M009	3	0
U028	M002	1	0
U028	M004	4	1
U028	M018	5	1
U028	M008	1	0
U028	M024	4	1
U028	M027	4	1
U028	M007	2	0
U028	M016	2	0
U029	M014	4	1
U029	M013	2	0
U029	M006	4	1
U029	M016	5	1
U029	M009	3	0
U029	M005	2	0

U029	M008	5	1
U029	M001	4	1
U029	M023	1	0
U030	M013	2	0
U030	M004	4	1
U030	M010	4	1
U030	M011	4	1
U030	M009	4	1
U031	M022	5	1
U031	M010	5	1
U031	M011	1	0
U031	M005	2	0
U031	M029	1	0
U031	M028	2	0
U032	M029	1	0
U032	M028	4	1
U032	M030	5	1
U032	M020	3	0
U032	M004	5	1
U032	M002	2	0
U032	M003	1	0
U033	M002	4	1
U033	M025	4	1
U033	M020	2	0
U033	M021	5	1
U033	M004	5	1
U033	M001	1	0
U033	M007	1	0
U033	M003	3	0
U034	M021	2	0
U034	M010	2	0
U034	M006	3	0
U034	M003	2	0
U034	M001	4	1
U034	M026	3	0
U034	M017	3	0
U035	M027	5	1
U035	M014	5	1
U035	M002	4	1
U035	M030	1	0
U035	M017	5	1
U035	M028	3	0
U035	M003	2	0
U035	M019	4	1
U035	M018	3	0

U035	M004	1	0
U036	M004	2	0
U036	M029	1	0
U036	M027	4	1
U036	M030	3	0
U036	M007	5	1
U036	M025	2	0
U036	M015	3	0
U037	M008	3	0
U037	M029	2	0
U037	M026	5	1
U037	M030	3	0
U037	M012	2	0
U037	M001	2	0
U037	M028	1	0
U037	M019	1	0
U038	M018	4	1
U038	M005	2	0
U038	M004	1	0
U038	M015	1	0
U038	M028	5	1
U038	M006	2	0
U038	M009	5	1
U039	M008	3	0
U039	M017	1	0
U039	M006	4	1
U039	M029	3	0
U039	M025	5	1
U039	M007	1	0
U039	M005	3	0
U040	M026	4	1
U040	M012	3	0
U040	M010	5	1
U040	M016	4	1
U040	M029	5	1
U040	M005	4	1
U040	M012	3	0
U040	M028	4	1
U041	M012	5	1
U041	M009	1	0
U041	M020	1	0
U041	M027	3	0
U041	M015	1	0
U041	M002	2	0
U041	M004	5	1

U041	M011	5	1
U041	M021	3	0
U042	M008	1	0
U042	M027	3	0
U042	M014	5	1
U042	M005	4	1
U042	M009	4	1
U042	M011	4	1
U042	M025	3	0
U042	M018	5	1
U042	M015	1	0
U042	M020	3	0
U043	M007	4	1
U043	M005	4	1
U043	M002	4	1
U043	M003	3	0
U043	M013	1	0
U043	M021	1	0
U043	M015	5	1
U043	M006	3	0
U043	M026	5	1
U043	M022	5	1
U044	M005	3	0
U044	M019	3	0
U044	M025	5	1
U044	M011	3	0
U044	M018	2	0
U044	M012	2	0
U045	M028	2	0
U045	M001	4	1
U045	M003	1	0
U045	M006	2	0
U045	M008	5	1
U045	M022	2	0
U045	M024	5	1
U046	M018	5	1
U046	M011	3	0
U046	M023	2	0
U046	M013	4	1
U046	M019	5	1
U046	M007	2	0
U046	M026	2	0
U046	M020	2	0
U047	M023	2	0
U047	M025	5	1

U047	M018	3	0
U047	M015	1	0
U047	M026	2	0
U047	M003	3	0
U047	M011	2	0
U047	M007	1	0
U047	M030	3	0
U048	M010	5	1
U048	M020	5	1
U048	M029	4	1
U048	M018	2	0
U048	M022	1	0
U048	M028	1	0
U048	M016	3	0
U048	M025	3	0
U048	M017	3	0
U048	M013	5	1
U049	M029	5	1
U049	M021	3	0
U049	M028	3	0
U049	M017	3	0
U049	M010	2	0
U049	M018	1	0
U049	M011	3	0
U049	M003	4	1
U050	M019	3	0
U050	M023	4	1
U050	M022	3	0
U050	M003	1	0
U050	M014	5	1
U050	M027	3	0
U050	M002	1	0
U050	M029	1	0
U050	M025	2	0
U050	M009	5	1
U051	M009	5	1
U051	M012	4	1
U051	M006	2	0
U051	M003	2	0
U051	M001	1	0
U052	M005	5	1
U052	M023	5	1
U052	M002	4	1
U052	M008	4	1
U052	M016	3	0

U052	M024	1	0
U052	M012	5	1
U052	M026	5	1
U053	M029	4	1
U053	M003	4	1
U053	M011	1	0
U053	M014	2	0
U053	M012	2	0
U053	M022	5	1
U053	M024	2	0
U053	M013	3	0
U053	M015	2	0
U054	M015	1	0
U054	M003	3	0
U054	M012	2	0
U054	M005	1	0
U054	M011	3	0
U054	M022	3	0
U054	M007	1	0
U054	M025	1	0
U055	M017	3	0
U055	M003	1	0
U055	M015	2	0
U055	M023	2	0
U055	M028	1	0
U055	M013	2	0
U056	M006	5	1
U056	M011	4	1
U056	M009	5	1
U056	M027	1	0
U056	M003	2	0
U056	M009	5	1
U056	M017	4	1
U056	M028	2	0
U057	M001	3	0
U057	M016	2	0
U057	M008	2	0
U057	M030	3	0
U057	M013	5	1
U057	M022	4	1
U058	M027	1	0
U058	M001	3	0
U058	M017	1	0
U058	M026	1	0
U058	M022	3	0

U058	M006	4	1
U058	M024	2	0
U058	M021	5	1
U058	M013	1	0
U059	M015	5	1
U059	M004	5	1
U059	M014	1	0
U059	M019	2	0
U059	M012	2	0
U059	M027	4	1
U059	M011	1	0
U059	M025	2	0
U059	M003	5	1
U060	M006	4	1
U060	M022	1	0
U060	M018	2	0
U060	M019	4	1
U060	M004	4	1
U060	M011	3	0
U060	M016	3	0
U061	M014	3	0
U061	M011	1	0
U061	M018	3	0
U061	M013	4	1
U061	M015	4	1
U061	M027	5	1
U061	M001	1	0
U061	M025	4	1
U061	M012	2	0
U062	M012	1	0
U062	M025	2	0
U062	M017	5	1
U062	M008	1	0
U062	M019	3	0
U062	M011	5	1
U062	M014	3	0
U063	M025	3	0
U063	M006	4	1
U063	M030	1	0
U063	M001	5	1
U063	M019	2	0
U063	M018	3	0
U063	M020	4	1
U063	M017	1	0
U063	M015	2	0

U063	M007	2	0
U064	M004	2	0
U064	M006	5	1
U064	M012	1	0
U064	M005	4	1
U064	M023	1	0
U064	M010	5	1
U064	M019	5	1
U064	M029	5	1
U065	M020	3	0
U065	M022	1	0
U065	M025	3	0
U065	M024	5	1
U065	M013	2	0
U065	M009	3	0
U065	M027	5	1
U066	M014	5	1
U066	M008	3	0
U066	M017	1	0
U066	M023	2	0
U066	M003	5	1
U066	M029	3	0
U066	M019	3	0
U066	M001	5	1
U067	M030	4	1
U067	M021	2	0
U067	M024	5	1
U067	M002	5	1
U067	M004	3	0
U068	M012	5	1
U068	M020	1	0
U068	M027	4	1
U068	M005	4	1
U068	M010	1	0
U068	M013	5	1
U069	M023	2	0
U069	M002	2	0
U069	M009	2	0
U069	M025	4	1
U069	M010	5	1
U069	M017	4	1
U069	M014	4	1
U069	M004	2	0
U069	M016	1	0
U069	M013	5	1

U070	M012	3	0
U070	M021	4	1
U070	M004	2	0
U070	M014	2	0
U070	M013	3	0
U070	M026	2	0
U070	M002	4	1
U070	M022	1	0
U071	M022	1	0
U071	M003	2	0
U071	M016	2	0
U071	M011	3	0
U071	M019	4	1
U071	M027	2	0
U071	M017	1	0
U071	M021	5	1
U071	M010	5	1
U072	M008	5	1
U072	M009	3	0
U072	M001	2	0
U072	M026	4	1
U072	M019	5	1
U072	M020	2	0
U073	M013	3	0
U073	M003	4	1
U073	M009	5	1
U073	M018	2	0
U073	M004	2	0
U074	M013	5	1
U074	M022	5	1
U074	M016	4	1
U074	M011	3	0
U074	M026	3	0
U074	M025	2	0
U074	M001	1	0
U074	M028	4	1
U074	M004	1	0
U074	M007	3	0
U075	M027	2	0
U075	M028	4	1
U075	M018	4	1
U075	M002	2	0
U075	M005	2	0
U075	M001	3	0
U075	M029	2	0

U076	M007	2	0
U076	M019	2	0
U076	M010	5	1
U076	M003	1	0
U076	M024	1	0
U076	M015	4	1
U076	M005	4	1
U076	M006	2	0
U076	M030	1	0
U076	M025	3	0
U077	M002	2	0
U077	M025	3	0
U077	M003	4	1
U077	M011	2	0
U077	M022	5	1
U077	M010	2	0
U077	M015	4	1
U077	M021	4	1
U077	M007	1	0
U078	M028	4	1
U078	M001	4	1
U078	M018	2	0
U078	M021	1	0
U078	M010	5	1
U078	M006	2	0
U078	M011	3	0
U078	M017	3	0
U078	M028	4	1
U079	M022	1	0
U079	M029	3	0
U079	M019	2	0
U079	M024	1	0
U079	M001	4	1
U080	M015	5	1
U080	M001	5	1
U080	M004	2	0
U080	M030	4	1
U080	M024	2	0
U080	M026	3	0
U080	M023	2	0
U080	M018	1	0
U081	M001	5	1
U081	M002	3	0
U081	M025	2	0
U081	M028	4	1

U081	M005	3	0
U081	M007	2	0
U081	M012	3	0
U082	M005	4	1
U082	M006	2	0
U082	M007	3	0
U082	M024	1	0
U082	M008	2	0
U083	M022	5	1
U083	M014	1	0
U083	M011	2	0
U083	M003	2	0
U083	M001	1	0
U084	M025	2	0
U084	M023	2	0
U084	M019	2	0
U084	M002	5	1
U084	M003	5	1
U084	M030	5	1
U085	M019	1	0
U085	M012	1	0
U085	M009	5	1
U085	M016	3	0
U085	M004	1	0
U085	M013	2	0
U085	M030	5	1
U085	M003	4	1
U085	M010	1	0
U085	M029	1	0
U086	M012	4	1
U086	M017	4	1
U086	M002	3	0
U086	M005	4	1
U086	M019	4	1
U086	M024	5	1
U086	M029	3	0
U086	M018	3	0
U086	M023	5	1
U087	M022	3	0
U087	M028	1	0
U087	M018	3	0
U087	M024	2	0
U087	M017	1	0
U088	M027	1	0
U088	M006	4	1

U088	M005	5	1
U088	M009	5	1
U088	M013	4	1
U088	M016	3	0
U089	M019	4	1
U089	M014	5	1
U089	M016	1	0
U089	M023	1	0
U089	M004	2	0
U089	M026	4	1
U089	M008	5	1
U089	M027	4	1
U089	M002	5	1
U090	M019	4	1
U090	M002	4	1
U090	M010	1	0
U090	M008	1	0
U090	M001	4	1
U090	M022	5	1
U091	M004	3	0
U091	M014	5	1
U091	M023	3	0
U091	M015	1	0
U091	M021	4	1
U091	M026	4	1
U091	M008	5	1
U091	M012	1	0
U091	M027	4	1
U091	M024	1	0
U092	M007	3	0
U092	M021	2	0
U092	M024	2	0
U092	M009	3	0
U092	M020	2	0
U092	M023	1	0
U092	M008	4	1
U092	M028	2	0
U092	M012	4	1
U092	M010	1	0
U093	M007	3	0
U093	M008	4	1
U093	M005	2	0
U093	M010	5	1
U093	M028	2	0
U093	M017	2	0

U093	M029	1	0
U094	M025	5	1
U094	M003	1	0
U094	M021	3	0
U094	M029	1	0
U094	M020	3	0
U094	M018	3	0
U094	M026	1	0
U094	M027	5	1
U095	M011	3	0
U095	M012	3	0
U095	M001	2	0
U095	M009	3	0
U095	M027	5	1
U095	M010	2	0
U095	M017	5	1
U095	M029	1	0
U095	M003	1	0
U095	M024	2	0
U096	M020	3	0
U096	M024	2	0
U096	M028	3	0
U096	M005	1	0
U096	M021	4	1
U096	M017	1	0
U097	M026	5	1
U097	M021	5	1
U097	M027	3	0
U097	M010	3	0
U097	M024	5	1
U098	M030	5	1
U098	M026	2	0
U098	M023	5	1
U098	M028	5	1
U098	M024	2	0
U098	M007	5	1
U098	M003	1	0
U099	M020	2	0
U099	M028	4	1
U099	M023	4	1
U099	M011	3	0
U099	M009	3	0
U099	M019	4	1
U099	M002	3	0
U099	M021	5	1

U099	M017	1	0
U100	M010	5	1
U100	M025	2	0
U100	M009	4	1
U100	M018	5	1
U100	M012	3	0
U100	M015	1	0
U100	M013	3	0
U100	M020	2	0
U100	M002	3	0
U100	M003	1	0
U101	M030	2	0
U101	M002	3	0
U101	M004	1	0
U101	M007	4	1
U101	M003	5	1
U102	M015	1	0
U102	M003	2	0
U102	M014	3	0
U102	M022	3	0
U102	M009	3	0
U103	M028	5	1
U103	M004	2	0
U103	M018	2	0
U103	M010	5	1
U103	M017	3	0
U104	M026	3	0
U104	M024	5	1
U104	M005	4	1
U104	M018	5	1
U104	M011	2	0
U105	M006	4	1
U105	M009	2	0
U105	M013	2	0
U105	M007	4	1
U105	M014	4	1
U106	M023	4	1
U106	M008	4	1
U106	M029	5	1
U106	M015	4	1
U106	M009	3	0
U107	M008	3	0
U107	M029	4	1
U107	M019	4	1
U107	M017	2	0

U107	M007	2	0
U108	M015	5	1
U108	M026	3	0
U108	M024	5	1
U108	M012	3	0
U108	M011	2	0
U109	M010	3	0
U109	M008	5	1
U109	M017	5	1
U109	M002	3	0
U109	M023	1	0
U110	M009	4	1
U110	M021	4	1
U110	M008	1	0
U110	M003	3	0
U110	M006	1	0
U111	M020	4	1
U111	M007	2	0
U111	M012	4	1
U111	M021	5	1
U111	M015	2	0
U112	M016	4	1
U112	M023	5	1
U112	M025	2	0
U112	M027	2	0
U112	M011	2	0
U113	M007	2	0
U113	M029	3	0
U113	M015	2	0
U113	M023	1	0
U113	M021	1	0
U114	M009	2	0
U114	M028	5	1
U114	M022	5	1
U114	M016	5	1
U114	M027	5	1
U115	M002	5	1
U115	M003	4	1
U115	M014	3	0
U115	M007	5	1
U115	M006	2	0
U116	M016	5	1
U116	M022	3	0
U116	M001	5	1
U116	M018	1	0

U116	M029	5	1
U117	M001	4	1
U117	M010	3	0
U117	M029	1	0
U117	M019	1	0
U117	M016	2	0
U118	M022	1	0
U118	M021	2	0
U118	M028	3	0
U118	M005	3	0
U118	M008	2	0
U119	M026	5	1
U119	M029	1	0
U119	M009	5	1
U119	M025	4	1
U119	M016	2	0
U120	M028	5	1
U120	M008	4	1
U120	M026	5	1
U120	M020	3	0
U120	M007	2	0
U121	M029	4	1
U121	M009	5	1
U121	M003	2	0
U121	M012	1	0
U121	M002	4	1
U122	M024	2	0
U122	M009	2	0
U122	M006	1	0
U122	M010	5	1
U122	M026	2	0
U123	M022	3	0
U123	M028	2	0
U123	M030	1	0
U123	M012	1	0
U123	M001	2	0
U124	M011	3	0
U124	M026	3	0
U124	M003	4	1
U124	M018	5	1
U124	M017	2	0
U125	M014	2	0
U125	M012	1	0
U125	M003	1	0
U125	M022	4	1

U125	M026	3	0
U126	M025	2	0
U126	M017	4	1
U126	M012	1	0
U126	M006	4	1
U126	M014	5	1
U127	M021	3	0
U127	M030	2	0
U127	M012	4	1
U127	M022	5	1
U127	M005	3	0
U128	M029	5	1
U128	M024	2	0
U128	M011	4	1
U128	M025	3	0
U128	M002	2	0
U129	M004	2	0
U129	M024	1	0
U129	M022	4	1
U129	M006	4	1
U129	M015	1	0
U130	M017	4	1
U130	M014	2	0
U130	M006	1	0
U130	M015	3	0
U130	M021	3	0
U131	M014	4	1
U131	M011	1	0
U131	M010	4	1
U131	M017	2	0
U131	M020	3	0
U132	M005	4	1
U132	M013	3	0
U132	M018	5	1
U132	M024	5	1
U132	M014	4	1
U133	M003	5	1
U133	M012	1	0
U133	M001	3	0
U133	M006	5	1
U133	M014	2	0
U134	M017	3	0
U134	M012	5	1
U134	M002	4	1
U134	M013	1	0

U134	M007	3	0
U135	M029	5	1
U135	M023	5	1
U135	M009	5	1
U135	M001	5	1
U135	M017	2	0
U136	M005	5	1
U136	M008	4	1
U136	M023	3	0
U136	M027	1	0
U136	M026	4	1
U137	M024	4	1
U137	M030	1	0
U137	M002	4	1
U137	M005	2	0
U137	M004	5	1
U138	M016	2	0
U138	M011	1	0
U138	M019	5	1
U138	M003	4	1
U138	M030	4	1
U139	M022	2	0
U139	M005	4	1
U139	M025	1	0
U139	M018	5	1
U139	M029	4	1
U140	M020	5	1
U140	M001	2	0
U140	M012	4	1
U140	M003	3	0
U140	M024	4	1
U141	M020	5	1
U141	M010	5	1
U141	M014	3	0
U141	M008	1	0
U141	M003	5	1
U142	M017	2	0
U142	M025	1	0
U142	M005	4	1
U142	M011	2	0
U142	M024	2	0
U143	M017	1	0
U143	M024	4	1
U143	M028	4	1
U143	M004	5	1

U143	M007	5	1
U144	M003	4	1
U144	M026	1	0
U144	M002	3	0
U144	M013	2	0
U144	M016	3	0
U145	M003	5	1
U145	M026	4	1
U145	M017	2	0
U145	M014	2	0
U145	M012	2	0
U146	M002	4	1
U146	M013	5	1
U146	M021	3	0
U146	M011	1	0
U146	M003	3	0
U147	M002	4	1
U147	M006	1	0
U147	M024	2	0
U147	M027	1	0
U147	M004	1	0
U148	M029	2	0
U148	M021	4	1
U148	M028	5	1
U148	M013	5	1
U148	M019	5	1
U149	M001	4	1
U149	M004	4	1
U149	M025	1	0
U149	M020	5	1
U149	M029	5	1
U150	M009	4	1
U150	M002	1	0
U150	M012	3	0
U150	M023	2	0
U150	M003	1	0
U151	M015	5	1
U151	M007	2	0
U151	M010	2	0
U151	M018	3	0
U151	M025	1	0
U152	M021	1	0
U152	M013	4	1
U152	M015	2	0
U152	M007	1	0

U152	M016	2	0
U153	M012	3	0
U153	M005	4	1
U153	M017	5	1
U153	M007	1	0
U153	M014	3	0
U154	M028	3	0
U154	M025	2	0
U154	M002	3	0
U154	M009	5	1
U154	M026	4	1
U155	M006	3	0
U155	M019	4	1
U155	M011	1	0
U155	M027	5	1
U155	M017	4	1
U156	M011	4	1
U156	M025	1	0
U156	M024	5	1
U156	M005	4	1
U156	M028	1	0
U157	M013	4	1
U157	M024	2	0
U157	M008	1	0
U157	M020	2	0
U157	M011	5	1
U158	M010	5	1
U158	M006	5	1
U158	M023	1	0
U158	M005	2	0
U158	M002	4	1
U159	M019	1	0
U159	M008	5	1
U159	M003	5	1
U159	M016	1	0
U159	M002	2	0
U160	M002	5	1
U160	M011	2	0
U160	M016	4	1
U160	M009	5	1
U160	M006	1	0
U161	M029	5	1
U161	M022	2	0
U161	M001	1	0
U161	M028	1	0

U161	M006	4	1
U162	M002	1	0
U162	M016	1	0
U162	M015	1	0
U162	M001	4	1
U162	M021	2	0
U163	M012	2	0
U163	M023	3	0
U163	M022	3	0
U163	M011	4	1
U163	M003	4	1
U164	M004	4	1
U164	M002	2	0
U164	M024	3	0
U164	M020	2	0
U164	M030	5	1
U165	M003	5	1
U165	M009	1	0
U165	M020	1	0
U165	M006	5	1
U165	M002	1	0
U166	M023	1	0
U166	M006	1	0
U166	M008	3	0
U166	M012	2	0
U166	M022	1	0
U167	M012	3	0
U167	M002	5	1
U167	M016	3	0
U167	M019	2	0
U167	M028	1	0
U168	M008	2	0
U168	M015	5	1
U168	M023	5	1
U168	M018	5	1
U168	M025	5	1
U169	M027	3	0
U169	M006	1	0
U169	M025	1	0
U169	M008	5	1
U169	M004	2	0
U170	M006	5	1
U170	M030	5	1
U170	M016	3	0
U170	M009	5	1

U170	M019	3	0
U171	M030	4	1
U171	M018	3	0
U171	M028	1	0
U171	M009	5	1
U171	M027	3	0
U172	M006	4	1
U172	M009	3	0
U172	M014	5	1
U172	M025	2	0
U172	M011	2	0
U173	M022	1	0
U173	M013	1	0
U173	M011	3	0
U173	M008	2	0
U173	M002	2	0
U174	M006	3	0
U174	M018	1	0
U174	M005	5	1
U174	M003	1	0
U174	M007	4	1
U175	M013	5	1
U175	M026	1	0
U175	M015	1	0
U175	M023	1	0
U175	M003	4	1
U176	M007	5	1
U176	M013	4	1
U176	M009	3	0
U176	M001	2	0
U176	M017	4	1
U177	M025	3	0
U177	M024	2	0
U177	M021	5	1
U177	M026	1	0
U177	M002	1	0
U178	M025	4	1
U178	M003	2	0
U178	M023	4	1
U178	M029	1	0
U178	M011	4	1
U179	M013	5	1
U179	M018	3	0
U179	M008	4	1
U179	M001	3	0

U179	M007	4	1
U180	M007	1	0
U180	M019	3	0
U180	M017	5	1
U180	M027	5	1
U180	M003	1	0
U181	M020	5	1
U181	M010	5	1
U181	M025	4	1
U181	M014	2	0
U181	M003	4	1
U182	M023	5	1
U182	M017	1	0
U182	M004	2	0
U182	M005	3	0
U182	M014	2	0
U183	M019	2	0
U183	M007	4	1
U183	M012	2	0
U183	M004	5	1
U183	M001	1	0
U184	M024	4	1
U184	M029	4	1
U184	M013	1	0
U184	M009	2	0
U184	M026	4	1
U185	M013	4	1
U185	M010	3	0
U185	M003	3	0
U185	M011	5	1
U185	M006	3	0
U186	M008	4	1
U186	M016	4	1
U186	M025	5	1
U186	M001	2	0
U186	M009	4	1
U187	M018	5	1
U187	M029	5	1
U187	M023	1	0
U187	M006	1	0
U187	M010	4	1
U188	M009	1	0
U188	M028	5	1
U188	M001	5	1
U188	M030	2	0

U188	M003	1	0
U189	M021	4	1
U189	M020	3	0
U189	M027	1	0
U189	M023	5	1
U189	M002	4	1
U190	M002	3	0
U190	M022	4	1
U190	M014	4	1
U190	M027	2	0
U190	M023	5	1
U191	M020	3	0
U191	M004	1	0
U191	M023	4	1
U191	M029	4	1
U191	M018	1	0
U192	M005	5	1
U192	M030	2	0
U192	M006	3	0
U192	M001	1	0
U192	M017	3	0
U193	M004	1	0
U193	M018	4	1
U193	M030	1	0
U193	M001	1	0
U193	M019	1	0
U194	M002	4	1
U194	M022	1	0
U194	M012	4	1
U194	M010	3	0
U194	M011	4	1
U195	M014	4	1
U195	M005	1	0
U195	M025	3	0
U195	M023	1	0
U195	M030	1	0
U196	M003	4	1
U196	M008	1	0
U196	M017	2	0
U196	M015	5	1
U196	M030	1	0
U197	M028	3	0
U197	M005	5	1
U197	M029	2	0
U197	M001	2	0

U197	M003	1	0
U198	M012	3	0
U198	M009	2	0
U198	M005	5	1
U198	M030	4	1
U198	M024	1	0
U199	M001	5	1
U199	M002	4	1
U199	M018	4	1
U199	M011	2	0
U199	M006	2	0
U200	M008	1	0
U200	M015	3	0
U200	M005	4	1
U200	M009	1	0
U200	M030	3	0

Source Code

Menentukan Sparsity

```
import pandas as pd

# Load file ratings.csv
ratings = pd.read_csv("ratings.csv")
# Tampilkan 5 baris awal
print("Contoh data:")
print(ratings.head())
# Jumlah total rating
total_ratings = len(ratings)
# Jumlah user unik dan movie unik
num_users = ratings['user_id'].nunique()
num_movies = ratings['movie_id'].nunique()
# Hitung sparsity
sparsity = 1 - (total_ratings / (num_users * num_movies))

print("\n=== ANALISIS SPARSITY ===")
print(f"Total Rating      : {total_ratings}")
print(f"Jumlah User Unik   : {num_users}")
print(f"Jumlah Film Unik    : {num_movies}")
print(f"Sparsity (kekosongan data) : {sparsity:.4f}")
# Hitung jumlah rating per user dan per movie
user_rating_counts = ratings['user_id'].value_counts()
movie_rating_counts = ratings['movie_id'].value_counts()
```

Melakukan SVD

```
import numpy as np
import pandas as pd

# Matriks A dari gambar (10x10)
A = np.array([
    [2, 0, 5, 1, 0, 3, 2, 0, 2, 5],
    [0, 4, 0, 2, 0, 0, 3, 0, 3, 0],
    [2, 0, 3, 0, 1, 0, 4, 2, 0, 3],
    [0, 4, 0, 4, 0, 2, 0, 3, 0, 0],
    [0, 0, 1, 0, 5, 0, 4, 5, 0, 5],
    [4, 0, 0, 5, 0, 3, 0, 0, 4, 5],
    [0, 5, 5, 0, 4, 0, 4, 0, 2, 0],
    [4, 5, 0, 5, 0, 1, 0, 5, 0, 5],
    [0, 2, 0, 5, 3, 0, 3, 0, 4, 0],
    [2, 0, 0, 5, 0, 0, 2, 0, 2, 1]
])

# Matriks AA^T (10x10)
A_AT = np.dot(A, A.T)

# Langkah-langkah SVD manual dengan k = 5
eigenvalues, eigenvectors = np.linalg.eig(A_AT)

# Urutkan berdasarkan eigenvalue terbesar
sorted_indices = np.argsort(eigenvalues)[::-1]
eigenvalues = eigenvalues[sorted_indices]
eigenvectors = eigenvectors[:, sorted_indices]

# Ambil 5 komponen laten terbesar
k = 5
top_eigenvalues = eigenvalues[:k]
top_eigenvectors = eigenvectors[:, :k]

# Matriks S (Sigma)
sigma = np.sqrt(np.maximum(top_eigenvalues, 0))
Sigma = np.diag(sigma)
```

```

# Matriks U
U = top_eigenvectors / np.linalg.norm(top_eigenvectors,axis=0)
# Matriks V^T
S_inv = np.linalg.pinv(Sigma)
U_T = U.T
V_T = np.dot(np.dot(S_inv, U_T), A)
# Hitung vektor akar dari Sigma
sqrt_sigma = np.sqrt(np.diag(Sigma))
# Matriks V dikembalikan dari V_T
V = V_T.T
# Buat DataFrame
eigenval_df = pd.DataFrame(eigenvalues,columns=["Eigenvalue"])
U_df = pd.DataFrame(U, columns=[f"u{i+1}" for i in range(U.shape[1])])
Sigma_df = pd.DataFrame(Sigma, columns=[f"σ{i+1}" for i in range(Sigma.shape[1])])
S_inv_df = pd.DataFrame(S_inv, columns=[f"σ-1{i+1}" for i in range(S_inv.shape[1])])
V_T_df = pd.DataFrame(V_T, columns=[f"v{i+1}" for i in range(V_T.shape[1])])
# Tampilkan hasil
print("Matriks AA^T:")
print(np.round(A_AT, 2))

print("Eigenvalue dari AA^T:")
print(np.round(eigenval_df, 3))

print("Matriks Sigma (S):")
print(np.round(Sigma_df, 3))

print("Matriks U (5 laten):")
print(np.round(U_df, 3))

print("Matriks V^T (5 laten):")

```

```

print(np.round(V_T_df, 3))

print("Matriks V (10x5):")
print(np.round(V, 3))

print("Matriks Akar Sigma (5 laten):")
print(np.round(sqrt_sigma, 3))

```

Mencari Prediksi Rating dengan *Matrix Factorization* (MF)

```

import numpy as np
import pandas as pd

# Matriks data asli
A = np.array([
    [2, 0, 5, 1, 0, 3, 2, 0, 2, 5], # U2
    [0, 4, 0, 2, 0, 0, 3, 0, 3, 0], # U5
    [2, 0, 3, 0, 1, 0, 4, 2, 0, 3], # U8
    [0, 4, 0, 4, 0, 2, 0, 3, 0, 0], # U23
    [0, 0, 1, 0, 5, 0, 4, 5, 0, 5], # U26
    [4, 0, 0, 5, 0, 3, 0, 0, 4, 5], # U40
    [0, 5, 5, 0, 4, 0, 4, 0, 2, 0], # U56
    [4, 5, 0, 5, 0, 1, 0, 5, 0, 5], # U64
    [0, 2, 0, 5, 3, 0, 3, 0, 4, 0], # U78
    [2, 0, 0, 5, 0, 0, 2, 0, 2, 1] # U93
])

# Matriks hasil SVD manual (5 laten)
U_matrix = np.array([
    [0.326, 0.133, -0.353, -0.458, 0.400],
    [0.217, 0.026, 0.422, -0.039, 0.065],
    [0.250, 0.300, -0.258, -0.049, 0.011],
    [0.233, -0.204, 0.242, 0.385, 0.219],

```



```

[0.337, 0.434, -0.347, 0.338, -0.549],
[0.389, -0.430, -0.195, -0.407, -0.095],
[0.294, 0.559, 0.391, -0.071, 0.361],
[0.477, -0.336, -0.140, 0.527, 0.258],
[0.305, -0.014, 0.472, -0.176, -0.450],
[0.242, -0.225, 0.134, -0.212, -0.280],
])

```

```

V_matrix = np.array([
[0.271, -0.250, -0.244, -0.118, 0.147],
[0.333, 0.035, 0.516, 0.407, 0.535],
[0.223, 0.451, -0.099, -0.302, 0.528],
[0.466, -0.533, 0.299, -0.041, -0.230],
[0.214, 0.439, 0.105, 0.102, -0.425],
[0.164, -0.154, -0.139, -0.160, 0.259],
[0.331, 0.473, 0.148, -0.137, -0.262],
[0.280, 0.045, -0.237, 0.662, -0.125],
[0.274, -0.072, 0.289, -0.483, -0.165],
[0.460, -0.031, -0.619, -0.044, -0.029],
])

```

```

# =====  PREDIKSI DENGAN MF =====

```

```

sqrt_sigma = np.array([4.333, 3.259, 3.066, 2.852, 2.493])

```

```

# Label user dan item

```

```

user_labels = [f"U{i}" for i in [2,5,8,23,26,40,56,64,78,93]]
item_labels = [f"M{i}" for i in [5,6,9,10,11,12,17,19,28,29]]

```

```

# Temukan nilai kosong dalam data

```

```

missing_indices = np.argwhere(A == 0)

```

```

# Hitung prediksi rating
missing_predictions = []
for u_idx, m_idx in missing_indices:
    P = U_matrix[u_idx] * sqrt_sigma
    Q = V_matrix[m_idx] * sqrt_sigma
    r_hat = np.dot(P, Q)

    missing_predictions.append({
        'User': user_labels[u_idx],
        'Item': item_labels[m_idx],
        'Prediksi_Rating': round(r_hat, 3)
    })

# Buat DataFrame hasil prediksi
df_missing_mf = pd.DataFrame(missing_predictions)

# Tampilkan hasil prediksi
print("Hasil Matrix Factorization (k = 5):")
print(df_missing_mf)

```

Pengujian dengan Evaluasi MAE dan *Confusion Matrix*

```

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import mean_absolute_error, confusion_matrix, accuracy_score,
precision_score, recall_score, f1_score

# ===== Data Rating Asli =====
data_asli = np.array([
    [2, 0, 5, 1, 0, 3, 2, 0, 2, 5],
    [0, 4, 0, 2, 0, 0, 3, 0, 3, 0],
    [2, 0, 3, 0, 1, 0, 4, 2, 0, 3],
    [0, 4, 0, 4, 0, 2, 0, 3, 0, 0],

```

```

[0, 0, 1, 0, 5, 0, 4, 5, 0, 5],
[4, 0, 0, 5, 0, 3, 0, 0, 4, 5],
[0, 5, 5, 0, 4, 0, 4, 0, 2, 0],
[4, 5, 0, 5, 0, 1, 0, 5, 0, 5],
[0, 2, 0, 5, 3, 0, 3, 0, 4, 0],
[2, 0, 0, 5, 0, 0, 2, 0, 2, 1]
])

# ===== Langkah 1: SVD =====
U, Sigma, VT = np.linalg.svd(data_asli, full_matrices=False)
k = 5
U_k = U[:, :k]
Sigma_k = Sigma[:k]
VT_k = VT[:, :k]

# ===== Langkah 2: Akar Sigma =====
sqrt_sigma_k = np.diag(np.sqrt(Sigma_k))

# ===== Langkah 3: Hitung P dan Q (full komponen) =====
P_k = np.dot(U_k, sqrt_sigma_k) #  $P = U * \sqrt{\Sigma}$ 
Q_k = np.dot(sqrt_sigma_k, VT_k) #  $Q = \sqrt{\Sigma} * V$ 

# ===== Langkah 4: Prediksi Rating (k = full) =====
prediksi_rating_k5 = np.dot(P_k, Q_k)

# ===== Langkah 5: Ambil 20% Data Testing =====
non_zero_indices = np.argwhere(data_asli != 0)
jumlah_testing = int(0.2 * len(non_zero_indices))

np.random.seed(42)
test_indices = non_zero_indices[np.random.choice(len(non_zero_indices),
size=jumlah_testing, replace=False)]

# ===== Langkah 6: Hitung MAE (full komponen) =====
y_true = []

```

```

y_pred = []

for i, j in test_indices:
    y_true.append(data_asli[i, j])
    y_pred_k5 = [prediksi_rating_k5[i, j] for i, j in test_indices]

mae_k5 = mean_absolute_error(y_true, y_pred_k5)

print("==== Evaluasi MAE (MF = P x Q, Full) ====")
print("Jumlah data testing:", len(test_indices))
print("MAE:", np.round(mae_k5, 4))
print("Index Data Testing:\n", test_indices)
print("Nilai Asli:", np.round(y_true))
print("Nilai Prediksi (k=5):", np.round(y_pred_k5, 4))

# ===== Langkah 7: Evaluasi Confusion Matrix =====
def binarize_ratings(ratings, threshold=3.0):
    return [1 if r >= threshold else 0 for r in ratings]

def plot_confusion(cm):
    plt.figure(figsize=(5, 4))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["Tidak Suka", "Suka"],
yticklabels=["Tidak Suka", "Suka"])
    plt.xlabel('Prediksi')
    plt.ylabel('Aktual')
    plt.title('Confusion Matrix')
    plt.tight_layout()
    plt.show()

true_binary = binarize_ratings(y_true)
pred_binary = binarize_ratings(y_pred_k5)

cm = confusion_matrix(true_binary, pred_binary)
acc = accuracy_score(true_binary, pred_binary)
prec = precision_score(true_binary, pred_binary, zero_division=0)

```

```

rec = recall_score(true_binary, pred_binary, zero_division=0)
f1 = f1_score(true_binary, pred_binary, zero_division=0)

print("\n=== Confusion Matrix ===")
print(cm)
print("Accuracy :", round(acc, 4))
print("Precision:", round(prec, 4))
print("Recall  :", round(rec, 4))
print("F1-Score :", round(f1, 4))

# ===== Visualisasi Confusion Matrix =====
plot_confusion(cm)

```

Perbandingan MF dengan Berbasis Memori (Tradisional)

```

import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.metrics import mean_absolute_error
import numpy as np

# Load data
ratings = pd.read_csv("ratings.csv") # kolom: user_id, movie_id, rating

# Buat pivot table user-item
user_item_matrix = ratings.pivot_table(index="user_id", columns="movie_id",
values="rating").fillna(0)

# Hitung similarity antar user
user_similarity = cosine_similarity(user_item_matrix)
np.fill_diagonal(user_similarity, 0) # set diagonal ke 0
user_similarity_df = pd.DataFrame(user_similarity, index=user_item_matrix.index,
columns=user_item_matrix.index)

# Fungsi prediksi rating user untuk film tertentu berdasarkan user-user similarity
def predict_rating_user_based(user_id, movie_id, k=5):
    if movie_id not in user_item_matrix.columns:

```

```

    return np.nan

sim_scores = user_similarity_df[user_id].copy()
sim_users = sim_scores.sort_values(ascending=False).head(k).index

rating_values = user_item_matrix.loc[sim_users, movie_id]
sim_values = sim_scores.loc[sim_users]

mask = rating_values > 0
if mask.sum() == 0:
    return np.nan

pred_rating = np.dot(rating_values[mask], sim_values[mask]) / sim_values[mask].sum()
return pred_rating

# Evaluasi MAE dengan data testing (misalnya 20% dari ratings)
from sklearn.model_selection import train_test_split
train_data, test_data = train_test_split(ratings, test_size=0.2, random_state=42)

# Buat ulang matriks berdasarkan train data
user_item_matrix = train_data.pivot_table(index="user_id", columns="movie_id",
values="rating", aggfunc='mean').fillna(0)
user_similarity = cosine_similarity(user_item_matrix)
np.fill_diagonal(user_similarity, 0)
user_similarity_df = pd.DataFrame(user_similarity, index=user_item_matrix.index,
columns=user_item_matrix.index)

# Prediksi dan evaluasi
actuals = []
predictions = []

for _, row in test_data.iterrows():
    uid = row["user_id"]
    mid = row["movie_id"]
    actuals.append(row["rating"])

```

```

pred = predict_rating_user_based(uid, mid)
predictions.append(pred if not np.isnan(pred) else 3.0) # fallback prediksi ke 3.0 jika
tidak bisa dihitung

mae = mean_absolute_error(actuals, predictions)
print(f"MAE User-Based CF: {round(mae, 4)}")

```

Website Rekomendasi Film untuk *User* dan Admin

```

from flask import Flask, render_template, request, redirect, session, url_for
import pandas as pd
import numpy as np

app = Flask(__name__)
app.secret_key = 'rekomendasi123'

data_admin = {'admin': 'admin123'}
data_film = pd.read_csv('movies.csv')
data_prediksi = pd.read_csv('data_prediksi.csv')
data_rating = pd.read_csv('ratings.csv')

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/recommend', methods=['POST'])
def recommend():
    user_id = request.form['user_id'].strip().upper()
    if user_id not in data_rating['user_id'].unique() and user_id not in
data_prediksi['user_id'].unique():
        return render_template('index.html', message="User ID tidak ditemukan dalam data.")

    session['user_id'] = user_id # simpan user_id ke session
    return redirect(url_for('rekomendasi_film'))

@app.route('/rekomendasi')

```

```

def rekomendasi_film():
    user_id = session.get('user_id')
    if not user_id:
        return redirect(url_for('index'))

    if user_id not in data_prediksi['user_id'].unique():
        return render_template('index.html', message="User ini tidak memiliki data prediksi
rating.")

    df_user = data_prediksi[data_prediksi['user_id'] == user_id]
    df_user_sorted = df_user.sort_values(by='rating', ascending=False).head(5)

    rekomendasi = []
    for _, row in df_user_sorted.iterrows():
        movie_info = data_film[data_film['movie_id'] == row['movie_id']]
        if not movie_info.empty:
            movie = movie_info.iloc[0]
            rekomendasi.append({
                'title': movie['title'],
                'genre': movie.get('genre', '-'),
                'country': movie.get('country', '-'),
                'rating': round(row['rating'], 2),
                'poster_url': movie.get('poster_url') if pd.notna(movie.get('poster_url')) and
movie.get('poster_url') != "" else
f"https://via.placeholder.com/300x450?text={movie['title']}"
            })

    return render_template('rekomendasi.html', recommendations=rekomendasi,
header_title='💎 Rekomendasi Film Untukmu')

@app.route('/watched', methods=['POST'])
def watched_redirect():
    user_id = request.form.get('user_id', '').strip().upper()
    if user_id:
        session['user_id'] = user_id

```



```

        return redirect(url_for('sudah_ditonton'))
    return redirect(url_for('index'))

@app.route('/dilihat')
def sudah_ditonton():
    user_id = session.get('user_id')
    if not user_id:
        return redirect(url_for('index'))

    df_user = data_rating[data_rating['user_id'] == user_id]
    df_user_sorted = df_user.sort_values(by='rating', ascending=False)

    ditonton = []
    for _, row in df_user_sorted.iterrows():
        movie_info = data_film[data_film['movie_id'] == row['movie_id']]
        if not movie_info.empty:
            movie = movie_info.iloc[0]
            ditonton.append({
                'title': movie['title'],
                'genre': movie.get('genre', '-'),
                'country': movie.get('country', '-'),
                'rating': round(row['rating'], 2),
                'poster_url': movie.get('poster_url') if pd.notna(movie.get('poster_url')) and
movie.get('poster_url') != "" else
f"https://via.placeholder.com/300x450?text={movie['title']}"
            })

    return render_template('rekomendasi.html', recommendations=ditonton,
header_title=' 🎬 Film yang Sudah Ditonton')

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']

```

```

    if username in data_admin and data_admin[username] == password:
        session['admin'] = username
        return redirect(url_for('dashboard_admin'))
    else:
        return render_template('admin_login.html', error="Login gagal. Periksa username
dan password.")
    return render_template('admin_login.html')

@app.route('/dashboard_admin')
def dashboard_admin():
    if 'admin' not in session:
        return redirect(url_for('login'))
    return render_template('admin_dashboard.html', films=data_film)

@app.route('/kelola_film', methods=['GET', 'POST'])
def kelola_film():
    if 'admin' not in session:
        return redirect(url_for('login'))
    df = pd.read_csv('movies.csv')
    if request.method == 'POST':
        movie_id = request.form['movie_id'].strip()
        title = request.form['title'].strip()
        if movie_id and title:
            if movie_id not in df['movie_id'].values:
                new_row = {'movie_id': movie_id, 'title': title, 'genre': "", 'country': "", 'year': "",
'poster_url': ""}
                df = pd.concat([df, pd.DataFrame([new_row])], ignore_index=True)
                df.to_csv('movies.csv', index=False)
            return redirect(url_for('kelola_film'))
    return render_template('kelola_film.html', films=df)

@app.route('/hapus_film/<movie_id>')
def hapus_film(movie_id):
    if 'admin' not in session:
        return redirect(url_for('login'))

```

```

df = pd.read_csv('movies.csv')
df = df[df['movie_id'] != movie_id]
df.to_csv('movies.csv', index=False)
return redirect(url_for('kelola_film'))

@app.route('/logout')
def logout():
    session.clear()
    return redirect(url_for('login'))

if __name__ == '__main__':
    app.run(debug=True)

```

Model Rekomendasi Film dengan Prediksi Rating (*Matrix Factorization*)

```

import pandas as pd

# Load datasets
df_ratings = pd.read_csv('ratings.csv')
df_movies = pd.read_csv('movies.csv')
df_prediksi = pd.read_csv('data_prediksi.csv')

def get_userRated_movies(user_id):
    """Ambil film yang sudah dirating user"""
    user_id = user_id.strip().upper()
    rated = df_ratings[df_ratings['user_id'] == user_id]
    merged = pd.merge(rated, df_movies, on='movie_id', how='inner')

    hasil = []
    for _, row in merged.iterrows():
        hasil.append({
            'movie_id': row['movie_id'],
            'title': row['title'],
            'rating': row['rating'],
            'genre': row.get('genre', 'N/A'),
            'country': row.get('country', 'N/A')
        })

```

```

    })
    return hasil

def generate_recommendations(user_id, top_n=5):
    """Ambil rekomendasi film berdasarkan data prediksi"""
    user_id = user_id.strip().upper()
    user_pred = df_prediksi[df_prediksi['user_id'] == user_id]
    if user_pred.empty:
        return []

    rated_movies = df_ratings[df_ratings['user_id'] == user_id]['movie_id'].tolist()
    unseen_recs = user_pred[~user_pred['movie_id'].isin(rated_movies)]

    merged = pd.merge(unseen_recs, df_movies, on='movie_id')
    top_recs = merged.sort_values(by='rating', ascending=False).head(top_n)

    hasil = []
    for _, row in top_recs.iterrows():
        hasil.append({
            'movie_id': row['movie_id'],
            'title': row['title'],
            'rating': round(row['rating'], 2),
            'genre': row.get('genre', 'N/A'),
            'country': row.get('country', 'N/A'),
            'poster_url': row.get('poster_url',
                'https://via.placeholder.com/300x450?text=No+Image')
        })
    return hasil

```

Tampilan untuk Halaman Awal User

```

<!DOCTYPE html>
<html lang="id">
<head>
<meta charset="UTF-8">
<title>Sistem Rekomendasi Film</title>

```

```

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
<style>
  body {
    background: linear-gradient(to right, #ece9e6, #ffffff);
    font-family: 'Segoe UI', sans-serif;
  }
  .header-icon {
    font-size: 48px;
    color: #5c6bc0;
  }
  .btn-rekomendasi {
    background-color: #5c6bc0;
    color: white;
    font-weight: bold;
    border-radius: 8px;
    padding: 10px 25px;
    margin-top: 10px;
  }
  .btn-rekomendasi:hover {
    background-color: #3f51b5;
  }
  .custom-card {
    background-color: white;
    padding: 30px;
    border-radius: 15px;
    box-shadow: 0 4px 12px rgba(0,0,0,0.08);
  }
</style>
</head>
<body class="d-flex align-items-center justify-content-center vh-100">
  <div class="container">
    <div class="custom-card text-center mx-auto" style="max-width: 600px;">
      <div class="header-icon mb-3"> 🎬 </div>
      <h2 class="mb-4">Sistem Rekomendasi Film</h2>

```

```

<!-- Tombol untuk user yang mendapatkan prediksi berdasarkan MF -->
<form method="POST" action="/recommend">
  <div class="form-group">
    <input type="text" name="user_id" placeholder="Masukkan User ID (misalnya:
U002)" class="form-control" required>
  </div>
  <button type="submit" class="btn btn-rekomendasi">  Lihat Rekomendasi Film
Untukmu</button>
</form>

```

```

<!-- Tombol untuk semua user (film yang sudah ditonton) -->
<form method="POST" action="/watched">
  <div class="form-group mt-3">
    <input type="text" name="user_id" placeholder="Masukkan User ID (misalnya: U002)"
class="form-control" required>
  </div>
  <button type="submit" class="btn btn-rekomendasi">  Daftar Film yang Sudah
Ditonton</button>
</form>

```

```

{% if message %}
  <div class="alert alert-warning text-center mt-4">{{ message }}</div>
{% endif %}

{% if watched %}
<div class="mt-5 text-left">
  <h5>  Film yang Sudah Ditonton:</h5>
  <ul class="list-group">
    {% for film in watched %}
      <li class="list-group-item d-flex justify-content-between align-items-center">
        {{ film.title }} <span class="badge badge-info">Rating: {{ film.rating }}</span>
      </li>
    {% endfor %}
  </ul>

```

```

    </div>
    {% endif %}
</div>
</div>
</body>
</html>

```

Tampilan untuk Halaman Rekomendasi *User*

```

<!DOCTYPE html>
<html lang="id">
<head>
  <meta charset="UTF-8">
  <title>Rekomendasi Film Untukmu</title>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
  <style>
    body {
      background: linear-gradient(to right, #f9f9f9, #ffffff);
      font-family: 'Segoe UI', sans-serif;
    }
    .card-movie {
      border: none;
      border-radius: 15px;
      box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
      transition: transform 0.2s ease-in-out;
    }
    .card-movie:hover {
      transform: scale(1.02);
    }
    .card-img-top {
      height: 320px;
      object-fit: cover;
      border-top-left-radius: 15px;
      border-top-right-radius: 15px;
    }
    .rating {

```

```

}

.btn-back {
  background-color: #e0e0e0;
  color: #333;
  border-radius: 8px;
  padding: 8px 18px;
}

.btn-back:hover {
  background-color: #d5d5d5;
}

</style>
</head>
<body>
  <div class="container py-5">
    <h3 class="text-center mb-5">{{ header_title }}</h3>

    <div class="row">
      {% for film in recommendations %}
        <div class="col-md-4 mb-4">
          <div class="card card-movie h-100">
            
            <div class="card-body">
              <h5 class="card-title">{{ film.title }}</h5>
              <p class="card-text">🎬 Genre: {{ film.genre }}<br>🌐 Negara: {{ film.country
            }}</p>
              <p class="card-text rating">★ {{ film.rating }}</p>
            </div>
          </div>
        </div>
      {% endfor %}
    </div>

    <div class="text-center mt-4">

```



```

    <a href="/" class="btn btn-back">← Kembali ke Halaman Utama</a>
  </div>
</div>
</body>
</html>

```

Tampilan untuk Halaman *Dashboard Admin*

```

<!DOCTYPE html>
<html lang="id">
<head>
  <meta charset="UTF-8">
  <title>Dashboard Admin</title>
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body class="bg-gray-50">
  <header class="bg-indigo-600 text-white p-4">
    <div class="container mx-auto">
      <h1 class="text-xl font-bold">Dashboard Admin</h1>
    </div>
  </header>

  <main class="container mx-auto p-6">
    <div class="grid gap-6 md:grid-cols-3">
      <!-- Kelola Film -->
      <a href="/kelola_film" class="bg-white shadow-md p-6 rounded-xl
hover:shadow-lg hover:bg-indigo-50 transition">
        <h2 class="text-lg font-semibold text-indigo-600">🎬 Kelola Film</h2>
        <p class="text-sm text-gray-500 mt-2">Tambah, lihat, dan hapus data film.</p>
      </a>
      <!-- Logout -->
      <a href="/logout" class="bg-white shadow-md p-6 rounded-xl hover:shadow-lg
hover:bg-red-50 transition">
        <h2 class="text-lg font-semibold text-red-600">🚪 Logout</h2>
        <p class="text-sm text-gray-500 mt-2">Keluar dari akun admin.</p>
      </a>
    </div>
  </main>
</body>
</html>

```

Tampilan untuk Halaman Kelola Film

```
<!DOCTYPE html>
<html lang="id">
<head>
  <meta charset="UTF-8">
  <title>Kelola Film</title>
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body class="bg-gray-100 min-h-screen">
  <header class="bg-indigo-700 text-white p-4">
    <h1 class="text-xl font-bold">Kelola Film</h1>
  </header>

  <main class="p-6">
    <form method="POST" class="flex flex-wrap gap-4 mb-6 items-center">
      <input type="text" name="movie_id" placeholder="ID Film" required class="p-2
rounded border flex-1" />
      <input type="text" name="title" placeholder="Judul Film" required class="p-2
rounded border flex-1" />
      <button type="submit" class="bg-indigo-600 text-white px-4 py-2 rounded hover:bg-
indigo-700">Tambah Film</button>
    </form>

    <div class="bg-white rounded shadow p-4">
      <h2 class="text-lg font-semibold mb-3">Daftar Film</h2>
      <table class="w-full text-sm text-left border-collapse">
        <thead class="bg-indigo-100">
          <tr>
            <th class="p-2 border">ID</th>
            <th class="p-2 border">Judul</th>
            <th class="p-2 border text-center">Aksi</th>
          </tr>
        </thead>
        <tbody>
          {% for film in films.itertuples() %}
          <tr class="hover:bg-indigo-50">
            <td class="p-2 border">{{ film.movie_id }}</td>
            <td class="p-2 border">{{ film.title }}</td>
            <td class="p-2 border text-center">
              <a href="/hapus_film/{{ film.movie_id }}" class="text-red-600
hover:underline">Hapus</a>
            </td>
          </tr>
          {% endfor %}
        </tbody>
      </table>
    </div>

    <div class="mt-6">
      <a href="/dashboard_admin" class="text-indigo-600 hover:underline">← Kembali ke
Dashboard</a>
    </div>
```

```
</main>  
</body>  
</html>
```