Sistem Rekomendasi Film Menggunakan Metode Content-Based Filtering dan Algoritma K-Nearest Neighbors (KNN)

Abdillah Rochmad Wahono[1*], Bintang Aji Saputra[2], Farid Fadlu Rahman[3]

[1] *Teknik Informatika/Fakultas Ilmu Komputer*
Surakarta, Indonesia
[1*]*230103278@mhs.udb.ac.id*

[2] *Teknik Informatika/Fakultas Ilmu Komputer*
Surakarta, Indonesia
[2]230103279@mhs.udb.ac.id

[3] *Teknik Informatika/Fakultas Ilmu Komputer*
Surakarta, Indonesia
[3]230103280@mhs.udb.ac.id

Abstrak— **alam era modern yang dipenuhi dengan berbagai media hiburan seperti media sosial, televisi, dan permainan, film layar lebar tetap mempertahankan popularitasnya sebagai salah satu bentuk hiburan utama. Film tidak hanya menawarkan pengalaman visual yang mendalam tetapi juga mampu menghadirkan realitas kehidupan dengan cara yang khas. Dengan teknologi sinematik yang terus berkembang, film menjadi jendela ajaib yang mengangkat berbagai cerita dari berbagai sudut pandang budaya dan sosial di seluruh dunia.Namun, di tengah kemudahan akses informasi melalui internet, pengguna dihadapkan pada dilema memilih film yang tepat. Motivasi menonton film pun bervariasi, mulai dari keinginan untuk mengeksplorasi karya sastra yang diadaptasi hingga sekadar mencari hiburan spontan. Hal ini menciptakan tantangan bagi perusahaan film dan platform streaming untuk menyediakan konten yang menarik dan relevan.Untuk mengatasi masalah ini, penelitian ini bertujuan untuk mengembangkan sistem rekomendasi film menggunakan pendekatan berbasis konten dan algoritma K-Nearest Neighbors (KNN). Sistem ini mengandalkan analisis data film berdasarkan fitur-fitur seperti genre, aktor, sutradara, dan tema untuk memberikan rekomendasi yang personal dan relevan sesuai dengan preferensi pengguna. Diharapkan sistem ini dapat meningkatkan pengalaman menonton pengguna serta mendukung peningkatan keuntungan bagi perusahaan film dan platform streaming online.**
Kata kunci— Film, Sistem Rekomendasi, Content-Based, K-Nearest Neighbors

Abstract— **In a modern era filled with various entertainment media such as social media, television, and games, feature films have maintained their popularity as one of the main forms of entertainment. Movies not only offer an immersive visual experience but are also able to present the reality of life in a distinctive way. With ever-evolving cinematic technology, movies have become a magical window into stories from different cultural and social viewpoints around the world. However, in the midst of easy access to information through the internet, users are faced with the dilemma of choosing the right movie. Motivations for watching movies also vary, from the desire to explore adapted literary works to simply seeking spontaneous entertainment. This creates a challenge for movie companies and streaming platforms to provide engaging and relevant content. To address this issue, this research aims to develop a movie recommendation system using a content-based approach and the K-Nearest Neighbors (KNN) algorithm. The system relies on analyzing movie data based on features such as genre, actor, director, and theme to provide personalized and relevant recommendations according to user preferences. It is expected that this system can improve the user's viewing experience and support increased profits for movie companies and online streaming platforms.**
Keywords— Films, Recommendation System, Content-Based, K-Nearest Neighbors.

I. PENDAHULUAN

Di tengah gempuran media hiburan baru seperti media sosial, TV, dan game, film layar lebar masih menjadi primadona bagi banyak orang. Industri film pun kokoh bertahan, menunjukkan daya tariknya yang tidak habis oleh waktu. Keistimewaan film kemungkinan terletak pada kemampuannya untuk membawa realitas kehidupan ke dalam layar lebar.

Film bagaikan jendela ajaib yang membawa kita ke dunia lain. Perpaduan gambar bergerak dan suara, direkam dengan teknologi canggih, menghidupkan cerita di layar lebar. Beragam genre film lahir dengan teknik sinematik yang unik, mencerminkan budaya dan realitas sosial bangsa pembuatnya. Industri film pun tak hanya terpusat di satu tempat, tapi mendunia, seperti Amerika, Eropa, Asia, dan menghadirkan kisah-kisah dari berbagai penjuru.

Membludaknya film dari berbagai penjuru dunia di internet bagaikan lautan informasi yang luas. Di sisi lain, hal ini memicu dilema bagi para pecinta film dalam memilih tontonan yang tepat. Motivasi mereka pun beragam, ada yang memiliki selera khusus dan sengaja datang ke bioskop untuk menonton film favorit, sementara yang lain hanya ingin mencari hiburan dan memilih film secara spontan. Menariknya, sebagian besar penikmat film layar lebar ternyata lebih menyukai film yang diadaptasi dari karya sastra seperti novel.

Lautan informasi film di internet bagaikan pisau bermata dua. Di satu sisi, ini memudahkan pengguna untuk menemukan film yang sesuai dengan selera

mereka. Namun, di sisi lain, banyaknya pilihan ini justru membuat banyak orang bingung dan frustrasi dalam memilih film yang ingin ditonton. Hal ini juga menjadi tantangan bagi perusahaan film dan platform streaming online, karena mereka kesulitan untuk membuat konten yang menarik perhatian di tengah lautan film yang ada. Memahami keinginan dan selera penonton menjadi kunci untuk mengatasi kebingungan ini. Perusahaan perlu menemukan cara untuk merekomendasikan film yang tepat kepada pengguna, agar mereka tidak membatalkan niat menonton dan meninggalkan platform. Sistem rekomendasi film hadir sebagai solusi, sistem rekomendasi content-based membantu pengguna menemukan film yang mereka sukai dengan cara merekomendasikan film yang mirip dengan film yang mereka inputkan dalam kolom pencarian. Sistem rekomendasi film memanfaatkan teknik data mining, salah satunya algoritma K-Nearest Neighbors (KNN), untuk memberikan rekomendasi yang tepat kepada pengguna. Caranya dengan menganalisis fitur dan atribut film, seperti genre, aktor, sutradara, dan tema. Sistem ini memanfaatkan data tentang film dan preferensi pengguna untuk memberikan saran film yang sesuai dengan selera mereka. Dengan cara ini, pengguna dapat menemukan film yang ingin ditonton dengan lebih mudah dan perusahaan pun dapat meningkatkan keuntungan mereka.

Penelitian ini bertujuan untuk membangun sistem rekomendasi film menggunakan metode content-based dan algoritma K-Nearest Neighbors (KNN). Sistem ini diharapkan dapat membantu pengguna dalam menemukan film yang sesuai dengan selera mereka dan meningkatkan pendapatan perusahaan film dan platform streaming online.

II. METODOLOGI PENELITIAN

A. Sistem Rekomendasi

Sistem rekomendasi adalah sebuah alat atau teknik perangkat lunak yang membantu pengguna menemukan item yang menarik dan relevan dengan minat mereka. Sistem ini bekerja dengan menganalisis data pengguna, seperti riwayat pembelian, riwayat pencarian, dan penilaian yang diberikan, untuk memprediksi item mana yang mungkin disukai pengguna (Ricci, et al., 2011). Rekomendasi yang dihasilkan dapat membantu pengguna dalam berbagai keputusan, seperti memilih film yang ingin ditonton, musik yang ingin didengarkan, berita yang ingin dibaca, atau produk yang ingin dibeli.

Sistem rekomendasi memiliki dua jenis input utama: implicit input dan explicit input (Hu, et al., 2008). Implicit input diperoleh dengan mengamati perilaku pengguna, seperti riwayat pembelian, riwayat penelusuran, pola pencarian, dan interaksi lainnya dengan sistem. Explicit input, di sisi lain, berasal dari penilaian yang secara langsung diberikan oleh pengguna, seperti rating, favorit, atau tanda suka/tidak suka pada item tertentu.

Berdasarkan cara kerjanya, sistem rekomendasi dapat diklasifikasikan menjadi beberapa jenis, yaitu:

a. Content-based: Sistem ini merekomendasikan item yang mirip dengan item yang pernah disukai atau dilihat pengguna sebelumnya.
b. Collaborative-based: Sistem ini merekomendasikan item yang disukai oleh pengguna lain dengan profil yang mirip dengan pengguna.
c. Hybrid-based: Sistem ini menggabungkan pendekatan content-based dan collaborative-based untuk menghasilkan rekomendasi yang lebih akurat dan personal.

B. Content-Based Filtering

Sistem rekomendasi Content-Based Filtering tidak menggunakan data dari pengguna lain untuk menentukan rekomendasi. Sistem ini hanya mengandalkan informasi dari pengguna itu sendiri, seperti apa yang pernah dilihat atau disukai. Berdasarkan informasi ini, algoritma memilih item dengan konten yang mirip untuk direkomendasikan. Keuntungan dari Content-Based Filtering adalah dapat memberikan rekomendasi yang relevan, bahkan bagi pengguna yang belum pernah memberikan penilaian (rating). Namun, kelemahannya adalah sistem ini mungkin tidak dapat menjangkau pengguna yang memiliki selera yang berbeda dari profilnya. Sebagai contoh, bayangkan pengguna A menyukai film dystopian dan komedi gelap. Pengguna B juga menyukai film dystopian, tetapi tidak pernah menonton komedi gelap. Content-Based Filtering hanya akan merekomendasikan film dystopian atau genre yang

mirip kepada pengguna B, meskipun ia mungkin juga menikmati film komedi gelap.

Sistem Content-Based Filtering dapat diimplementasikan dengan berbagai cara. Dalam film, sistem ini dapat dibangun berdasarkan genre, sutradara, aktor utama, atau kombinasi dari beberapa kategori. Sistem ini bekerja dengan mengekstrak informasi dari item dan membandingkannya dengan informasi item yang pernah dilihat atau disukai oleh pengguna. Metode Content-Based Filtering banyak digunakan untuk merekomendasikan berbagai jenis konten, seperti artikel, berita, dan situs web. Beberapa teknik yang biasa digunakan dalam Content-Based Filtering adalah TF-IDF, Bayesian Classifiers, Cluster analysis, decision trees, dan artificial neural networks.

C. K-Nearest Neighbors (KNN)

Algoritma KNN (K-Nearest Neighbors) merupakan salah satu algoritma klasifikasi yang populer dan mudah diterapkan. Algoritma ini termasuk dalam kategori lazy learning, yang berarti algoritma tidak membangun model secara eksplisit, melainkan hanya menyimpan data latih untuk digunakan pada saat prediksi (Alkhatib et al., 2013). Dalam penerapan algoritma KNN, data dibagi menjadi dua bagian: data latih dan data uji. Data latih digunakan oleh algoritma untuk mempelajari pola dan membuat dasar prediksi, sedangkan data uji digunakan untuk mengevaluasi performa algoritma (Imandoust dan Bolandraftar, 2013). Sebelum proses prediksi, data latih diubah menjadi vektor numerik. Kemudian, algoritma menghitung jarak antara data uji dan setiap data latih menggunakan berbagai metode, seperti euclidean distance atau cosine similarity. Penelitian ini menggunakan Cosine Similarity untuk mengukur kesamaan antara halaman web. Caranya dengan menghitung jumlah kata kunci yang sama yang muncul pada halaman-halaman yang tercantum dalam daftar indeks. Semakin banyak kata kunci yang sama, semakin tinggi nilai Cosine Similarity yang dihasilkan. Berdasarkan jarak yang dihitung, algoritma KNN mengidentifikasi k data latih terdekat dengan data uji. K merupakan nilai parameter yang menentukan jumlah tetangga terdekat yang akan dipertimbangkan. Nilai prediksi untuk data uji kemudian ditentukan berdasarkan mayoritas kelas dari k tetangga terdekat tersebut. Langkah-langkah algoritma KNN:

1. Menentukan parameter k (jumlah tetangga terdekat).
2. Hitung jarak data latih dengan semua data uji
3. Urutkan jarak tersebut berdasarkan nilai yang terkecil sejumlah k.
4. Tentukan kelompok data uji berdasarkan label mayoritas pada k.

$$SIM\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \frac{\overrightarrow{t_a} \cdot \overrightarrow{t_b}}{|\overrightarrow{t_a}| \; x \; |\overrightarrow{t_b}|}$$

Cosine Similarity dihitung berdasarkan dua vektor multi-dimensi, ta $\vec{}$ dan tb $\vec{}$, yang mewakili kumpulan istilah T = {$t1$,$t2$,$t3$, … ,tn }. Setiap dimensi pada vektor tersebut merepresentasikan kemunculan istilah dan bobotnya dalam sebuah dokumen, dengan nilai non-negatif. Hasil perhitungan Cosine Similarity menghasilkan nilai non-negatif antara 0 dan 1 (Huang, 2008)

D. Precision and Recall

Dua metrik yang paling umum digunakan untuk mengevaluasi kinerja sistem rekomendasi adalah precision dan recall (Christopher et al., 2008). Metrik ini memberikan gambaran tentang seberapa akurat dan lengkap rekomendasi yang dihasilkan oleh sistem. Gagasan ini dapat dijelaskan dengan confusion matrix pada Gambar 1.

	Prediksi	
	Relevan	**Tidak Relevan**
Relevan	True Positive	False Positive
Tidak Relevan	False Negative	True Negative

Gambar 1. Confusion Matrix

Dalam evaluasi sistem klasifikasi, terdapat beberapa istilah penting yang perlu dipahami:

1. True Positive (TP): Jumlah data yang positif dan diprediksi sebagai positif oleh sistem.
2. False Negative (FN): Jumlah data yang positif, namun diprediksi sebagai negatif oleh sistem.
3. True Negative (TN): Jumlah data yang negatif dan diprediksi sebagai negatif oleh sistem.

4. False Positive (FP): Jumlah data yang negatif, namun diprediksi sebagai positif oleh sistem.

Nilai-nilai ini dapat digunakan untuk menghitung beberapa metrik performa, seperti precision, recall, dan F-measure. Precision adalah proporsi dokumen yang direkomendasikan dan benar-benar relevan dengan kebutuhan pengguna.

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

Recall adalah proporsi dokumen yang relevan dengan kebutuhan pengguna dan berhasil direkomendasikan oleh sistem.

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

F-measure adalah kombinasi dari precision dan recall yang memberikan gambaran komprehensif tentang kinerja sistem. Nilai-nilai ini dapat dihitung berdasarkan data yang disajikan dalam Gambar 1.

$$F - measure = \frac{2PR}{P+R}$$

E. Metode Pengumpulan Data

Data untuk penelitian ini berasal dari situs https://grouplens.org/datasets/movielens/latest/. Dataset ini terbagi menjadi dua yaitu dataset film dan dataset rating. Dataset film berisi informasi tentang film, seperti ID film, judul, dan genre. Dataset rating berisi informasi tentang rating film yang diberikan oleh pengguna, termasuk ID pengguna, ID film, rating yang diberikan, dan waktu pemberian rating.

III. HASIL DAN PEMBAHASAN

Kebutuhan Hardware dan Software yang digunakan.

1. Hardware yang diperlukan untuk menjalankan program yaitu :
   a. Proccessor Corei5-3320M
   b. RAM 12Gb DDR4
   c. Harddisk 1 TB
   d. Keyboard
   e. Monitor
   f. Mouse
2. Software yang diperlukan untuk menjalankan program yaitu:
   a. Sistem Operasi Windows minimal 10
   b. Web Browser

A. Perhitungan

Proses perhitungan pada penelitian ini mengguakan Google Collab dengan bahasa Python. Diawali dengan menginisilisasi library dan data yang akan diproses.

```
import pandas as pd
import numpy as np
from sklearn.neighbors import NearestNeighbors
from scipy.sparse import csr_matrix
from sklearn.metrics import recall_score
movies_df=pd.read_csv("/content/drive/MyDrive/Dataset/movies.csv")
ratings_df=pd.read_csv("/content/drive/MyDrive/Dataset/ratings.csv")
```

Gambar 2. Inisialisasi dan Data

Pada Gambar 2 di atas, terlihat bahwa library yang aka digunakan adalah Pandas, Numpy, library Nearest neighbors dan csr_matrix. Sedangkan dataset yang akan diinisiasi atau dirubah menjadi data frame adalah data movies dan ratings.

```
combined_df=pd.merge(ratings_df,movies_df,on='movieId')
combined_df.head(5)
```

	userId	movieId	rating	timestamp	title	genres
0	1	1	4.0	964982703	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	5	1	4.0	847434962	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	7	1	4.5	1106635946	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
3	15	1	2.5	1510577970	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
4	17	1	4.5	1305696483	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

```
[6] combined_df.drop('timestamp',inplace=True,axis=1)
```

Gambar 3. Penggabungan Dataset Movies dan Ratings

Setelah dilakukan inisialisasi maka selanjutnya adalah penggabungan dataset dengan parameter movieId. Setelah dilakukan penggabungan kemudian dilakukan normalisasi dataset dengan menghapus kolom timestamp.

```
[13] rating_popular_movie=rating_with_totalRatingCount.loc[rating_with_totalRatingCount.totalRatingCount>50,]
     rating_popular_movie.shape

(40712, 6)

movie_features_df=combined_df.pivot_table(index='title',columns='userId',values='rating').fillna(0)
movie_features_df.head(5)
```

Gambar 4. Pengambilan Data dengan Jumlah Rating diatas 50

Kemudian dilakukan filterisasi data, dimana dataset yang akan digunakan hanya data yang memiliki jumlah rating di atas 50. Setelah itu hasil penggabungan dataset yang sudah dilakukan akan diubah menjadi pivot table dengan indexnya adalah

title, kolomnya berisi userId, dan pivot tablenya berisi rating.

userId	1	2	3	4	5	6	7	8	9	10	...	601	602	603	604	605	606	607	608	609	610
title																					
'71 (2014)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0
'Hellboy': The Seeds of Creation (2004)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
'Round Midnight (1986)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
'Salem's Lot (2004)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
'Til There Was You (1997)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 610 columns

Gambar 5. Hasil Konversi Pivot Tabel

Selanjutnya setelah konversi dilakukan adalah proses perhitungan menggunakan KNN dengan parameter matrix cosine dimana pengguna dapat memasukkan judul film yang akan menjadi patokan untuk mencari film-film lain yang memiliki kemiripan dengan film tersebut.

```
[36] knn=NearestNeighbors(metric='cosine')
     knn.fit(movie_features_df)
     title_film = "DOA: Dead or Alive (2006)"
     query_index = movie_features_df.index.get_loc(title_film)
     #x = movie_features_df.index[query_index]
     print(query_index)
     distances, indices =knn.kneighbors(movie_features_df.iloc[query_index,:].values.reshape(1, -1), n_neighbors =20)
```

Gambar 6. Proses Perhitungan KKN

Dapat dilihat pada proses perhitungan KNN di atas, setelah pengguna memasukkan judul film yang ingin dicari kemiripannya. Selanjutnya judul film tadi akan dikonversi menjadi index dengan fungsi get_loc. Kemudian perhitungan KNN dilakukan dengan index yang telah dikonversi tadi dan menggunakan parameter k=20.

```
[42] for i in range(0,len(distances.flatten())):
         if i == 0:
             print('Recommendations for {0}:\n'.format(movie_features_df.index[query_index]))
         else:
             print('{0}: {1}, with distance of {2}:'.format(i, movie_features_df.index[indices.flatten()[i]], distances.flatten()[i]))
```

Gambar 7. Pengulangan Untuk Menampilkan Rekomendasi Film

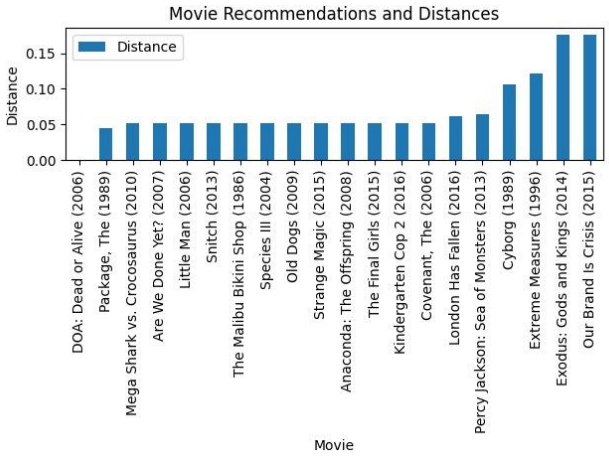
Pada Gambar 7 memperlihatkan bahwa terdapat proses for loop yang digunakan untuk menampilkan 10 hasil rekomendasi film berdasarkan perhitungan KNN.

```
[48] import matplotlib.pyplot as plt
     # Create a DataFrame from recommendations and distances
     judul_df = pd.DataFrame({
         'Movie': movie_features_df.index[indices.flatten()],
         'Distance': distances.flatten()
     })

     # Plot the distances
     judul_df.plot(x='Movie', y='Distance', kind='bar')
     plt.title('Movie Recommendations and Distances')
     plt.xlabel('Movie')
     plt.ylabel('Distance')
     plt.xticks(rotation=90)  # Rotate x-axis labels for readability
     plt.tight_layout()  # Adjust layout to prevent labels from overlapping
     plt.show()
```

Gambar 8. Kode Program

Agar lebih mudah dibaca maka dapat dibuat grafik yang dapat menampilkan hasil dari perhitungan KNN. Data yang akan dimasukkan ke dalam grafik adalah data hasil film yang direkomendasikan beserta jaraknya.



Gambar 9. Gafik Hasil Rekomendasi

```
def calculate_precision(user_id, recommendations, threshold=4):
    """Calculates precision for a given user."""
    user_ratings = combined_df[combined_df['userId'] == user_id]
    relevant_movies = user_ratings[user_ratings['rating'] >= threshold]['title']
    recommended_movies = movie_features_df.iloc[recommendations].index # get the titles of recommended movies using iloc
    relevant_recommendations = recommended_movies.intersection(relevant_movies)
    return len(relevant_recommendations) / len(recommended_movies) if len(recommended_movies) > 0 else 0

def calculate_recall(user_id, recommendations, threshold=4):
    """Calculates recall for a given user."""
    user_ratings = combined_df[combined_df['userId'] == user_id]
    relevant_movies = user_ratings[user_ratings['rating'] >= threshold]['title']
    recommended_movies = movie_features_df.iloc[recommendations].index
    relevant_recommendations = recommended_movies.intersection(relevant_movies)
    return len(relevant_recommendations) / len(relevant_movies) if len(relevant_movies) > 0 else 0
# Input The User
user_id = 1
num_recommendations = 10
# Get the index of the first movie rated by the user **from movie_features_df**
query_index = movie_features_df.index.get_loc(combined_df[combined_df['userId'] == user_id]['title'].iloc[0])
# Find recommendations for this movie
distances, indices = knn.kneighbors(movie_features_df.iloc[query_index].values.reshape(1, -1), n_neighbors=num_recommendations)
precision = calculate_precision(user_id, indices.flatten())
recall = calculate_recall(user_id, indices.flatten())
print(f"Precision for user {user_id}: {precision}")
print(f"Recall for user {user_id}: {recall}")
```

Gambar 10. Perhitungan Precision dan Recall

Gambar 10 menunnjukkan kode program untuk menghitung precision dan recall dari sistem rekomendasi. Kode program ini dijalankan dengan cara memasukkan user_id pengguna kemudian mencari film pertama yang diratig oleh user tersebut.

```
[58] # Calculate average precision across all users
     precision_scores = []
     recall_scores = []
     for user_id in combined_df['userId'].unique():
         # Get the index of the first movie rated by the user **from movie_features_df**
         query_index = movie_features_df.index.get_loc(combined_df[combined_df['userId'] == user_id]['title'].iloc[0])
         # Find recommendations for this movie
         distances, indices = knn.kneighbors(movie_features_df.iloc[query_index].values.reshape(1, -1), n_neighbors=num_recommendations)
         precision = calculate_precision(user_id, indices.flatten())
         precision_scores.append(precision)
         recall = calculate_recall(user_id, indices.flatten())
         recall_scores.append(recall)

     average_precision = np.mean(precision_scores)
     average_recall = np.mean(recall_scores)
     f_measure = 2 * (average_precision * average_recall) / (average_precision + average_recall)
     print(f"F-measure: {f_measure}")
     print(f"Average Recall: {average_recall}")
     print(f"Average Precision: {average_precision}")

F-measure: 0.12459025057059903
Average Recall: 0.07768665203319576
Average Precision: 0.3144262295081967
```

Gambar 11. Perhitungan F-Measure, Average, Precision dan Recall

Kode program di atas merupakan kode program yang digunakan untuk menghitung rata-rata dari

precision dan recall. Dimana rata-rata ini nantinya akan digunakan untuk menghitung F-measure.

B. Uji Coba

```
knn=NearestNeighbors(metric='cosine')
knn.fit(movie_features_df)
title_film = "London Has Fallen (2016)"
query_index = movie_features_df.index.get_loc(title_film)
#x = movie_features_df.index[query_index]
print(query_index)
distances, indices =knn.kneighbors(movie_features_df.iloc[query_index,:].values.reshape(1, -1), n_neighbors =20)
```

```
2108
```

```
[42] for i in range(0,len(distances.flatten())):
        if i == 0:
            print('Recommendations for {0}:\n'.format(movie_features_df.index[query_index]))
        else:
            print('{0}: {1}, with distance of {2}:'.format(i, movie_features_df.index[indices.flatten()[i]], distances.flatten()[i]))
```

Gambar 12. Uji Coba Pencarian Film

Gambar 12 merupakan uji coba pencarian film yang memiliki kemiripan dengan film “London has Fallen (2016)”.

```
Recommendations for London Has Fallen (2016):

1: Percy Jackson: Sea of Monsters (2013), with distance of 0.00027030680315420774:
2: Kindergarten Cop 2 (2016), with distance of 0.010050506338833531:
3: The Malibu Bikini Shop (1986), with distance of 0.010050506338833531:
4: Covenant, The (2006), with distance of 0.010050506338833531:
5: Are We Done Yet? (2007), with distance of 0.010050506338833531:
6: Anaconda: The Offspring (2008), with distance of 0.010050506338833531:
7: Old Dogs (2009), with distance of 0.010050506338833531:
8: Snitch (2013), with distance of 0.010050506338833531:
9: The Final Girls (2015), with distance of 0.010050506338833531:
10: Mega Shark vs. Crocosaurus (2010), with distance of 0.010050506338833531:
11: Strange Magic (2015), with distance of 0.010050506338833531:
12: Species III (2004), with distance of 0.010050506338833531:
13: Little Man (2006), with distance of 0.010050506338833531:
14: DOA: Dead or Alive (2006), with distance of 0.06085144945008847:
15: Our Brand Is Crisis (2015), with distance of 0.1404831410791113:
16: Exodus: Gods and Kings (2014), with distance of 0.1404831410791113:
17: Annabelle (2014), with distance of 0.1944449622037493:
18: Package, The (1989), with distance of 0.1944449622037493:
19: Extreme Measures (1996), with distance of 0.23003928270798157:
```

Gambar 13. Hasil Rekomendasi

Setelah dilakukan pencarian film maka akan mengahasilkan 20 rekomendasi film yang serupa dengan film “London has Fallen (2016)”

IV. KESIMPULAN

Berdasarkan hasil dan pembahasan pada poin sebelumnya, maka penulis menarik kesimpulan bahwa program sistem rekomendasi film menggunakan metode content-based filtering dan algoritma k-nearest neighbors (KNN) dapat berjalan dengan baik dan memiliki tingkat precision dan recall yang bagus. Kemudian dengan adanya sistem rekomendasi film ini diharapkan dapat menjadi rujukan untuk pengembangan sistem rekomendasi di masa depan.

UCAPAN TERIMA KASIH

Kepada Universitas Duta Bangsa yang telah memberikan kesempatan sehingga pembuatan jurnal ini bisa terlaksana. Selanjutnya kepada peneliti terdahulu yang telah menjadi referensi bagi penulis untuk melakukan penelitian ini.

REFERENSI

Musyrifah, Sulfayanti, I, Ap, Asmawati, N. Zulkarnain, “Sistem Rekomendasi Berbasis-Konten Untuk Pengembangan Web Smart Tourism”, Jurnal Komputer Terapan, Vol. 8, No 1, pp 143-150, Mei 2022.

A. R. Fitrianti, A. Rohmani dan Widjanarto, “Sistem Rekomendasi Film Berbasis Website dengan Metode Prototype Menggunakan Algoritma K-Nearest Negihbors (KNN)”, Jurnal of Information System, Vol. 5, No. 2, pp 278-287, Nopember 2020.

M. Yuliani dan D. D. Hutagalung, “Sistem Rekomendasi Webtoon dengan Menggunakan Metode Content-Based Filtering”, OKTAL : Jurnal Ilmu Komputer dan Science, Vol. 2, No. 11, pp 2959-2442, November 2023.

C. S. D. Prasetya, “Desain Smart Nutrition Monitoring System Teknik Budidaya Hidroponik Sistem Rekomendasi Pada E-Commerce Menggunakan K-Nearest Neighbor”, Jurnal Teknologi Informasi dan Ilmu Komputer , Vol. 4, No. 3 pp 194-200, September 2017.

F. B. A. Larasati dan H. Februariyanti, “Sistem Rekomendasi Product Emina Cosmetics Dengan Menggunakan Metode Content – Based Filtering”, Vol. 4, No. 1, pp 45-54, Januari 2021

N. D. Khairina, “Indentification of Pneumonia using The K-Nearest Neighbor Method using HOG Fitur Feature Extraction”, JITE (Journal of Informatics and Telemunication Engineering), pp 562-568, 2022.

R. D. Syah, “Performa Algoritma User K-Nearest Neighors pada Sistem Rekomendasi di TokoPedia”, Jurnal Informatika Universitas Pamulang, 302-306, 2020.

H. H. Afrisko, “Sistem Rekomendasi Film Menggunakan Metode Hybrid Collaborative Filtering dan Contens-Based Filtering”, eProceedings of Engineering, 2149-2159, 2022.

S. H. Angela, “Sistem Rekomendasi Pembelian Laptop Dengan K-Nearest Neighbor (KNN)”, INFRA, 1-7, 2022.

C. Fiarni dan H. Maharani, “Product Recommendation System Design Using Cosine Similarity and Content-Based Filtering Methods”, IJITEE (International Journal of Information Technology and Electrical Engineering), Vol.3, No. 2, pp 42-28. 2019.