e-ISSN 2549-7472
Edumatic: Jurnal Pendidikan Informatika
Vol. 8 No. 2, Desember, 2024, **Hal. 595-604**
DOI: 10.29408/edumatic.v8i2.27793

Analisis Metode Collaborative Filtering menggunakan KNN dan SVD++ untuk Rekomendasi Produk E-commerce Tokopedia

Chalista Yulia Hazizah [1], Triyanna Widiyaningtyas [1,*]

[1] Program Studi Teknik Informatika, Universitas Negeri Malang, Indonesia
[*] Correspondence: triyannaw.ft@um.ac.id

Received: 13 Oktober 2024 | Revised: 19 Oktober 2024 | Accepted: 8 November 2024 | Published: 19 Desember 2024

Abstrak
Perkembangan pesat teknologi internet mendorong peningkatan adopsi *e-commerce*. Namun, perusahaan menghadapi tantangan dalam meningkatkan pengalaman belanja pengguna. Untuk membantu pengguna menemukan produk sesuai preferensi, diperlukan analisis rekomendasi yang relevan. Penelitian ini bertujuan membandingkan efektivitas algoritma *K-Nearest Neighbors* (KNN) dan *Singular Value Decomposition Plus Plus* (SVD++) dalam memberikan rekomendasi produk *e-commerce*. Data berasal dari dataset *Tokopedia Product Reviews* di Kaggle yang mencakup 40.893 ulasan produk dengan atribut ulasan, *rating*, *id_product*, dan *id_shop*. Penelitian melalui tahapan *data collection* untuk menyiapkan data ulasan dan *rating*. Tahapan *preprocessing* melibatkan penghapusan duplikat, penggantian nilai hilang menggunakan rata-rata, dan normalisasi rating menggunakan *MinMaxScaler*. Pada tahap *processing*, KNN dan SVD++ diterapkan untuk memprediksi *rating* menggunakan *cosine similarity* dan matriks faktor. Evaluasi dilakukan menggunakan *Mean Absolute Error* (MAE) dan *Root Mean Square Error* (RMSE). Hasil penelitian menunjukkan bahwa SVD++ unggul dengan MAE 0,161176 dan RMSE 0,185252 dibandingkan KNN yang memiliki MAE 0,163964 dan RMSE 0,197045. SVD++ terbukti lebih baik memberikan akurasi dan menangkap kompleksitas data karena kemampuannya memodelkan interaksi laten antara pengguna dan item. Penelitian ini menunjukkan potensi peningkatan efektivitas rekomendasi di *e-commerce* yang berkontribusi dalam meningkatkan kepuasan pengguna untuk menemukan produk sesuai preferensi mereka dengan efisien.

Kata kunci: algoritma knn; algoritma svd++; *collaborative filtering*; data mining; *e-commerce*

Abstract
The rapid development of internet technology has driven increased adoption of e-commerce, yet companies face challenges in enhancing users' shopping experiences. To assist users in finding products that match their preferences, relevant recommendation analysis is crucial. This research compares the effectiveness of K-Nearest Neighbors (KNN) and Singular Value Decomposition Plus Plus (SVD++) algorithms for e-commerce product recommendations using the Tokopedia Product Reviews dataset from Kaggle, which contains 40,893 reviews. The study includes data collection and preprocessing steps such as removing duplicates, replacing missing values with the average, and normalizing ratings. KNN and SVD++ are then applied to predict ratings using cosine similarity and factor matrices. Evaluation using Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) shows that SVD++ outperforms KNN, achieving a lower MAE of 0.161176 and RMSE of 0.185252, compared to KNN's MAE of 0.163964 and RMSE of 0.197045. This indicates that SVD++ is more effective in delivering accuracy and capturing data complexity. The findings highlight the potential to enhance recommendation effectiveness in e-commerce, improving user satisfaction by efficiently matching products to preferences.

Keywords: *knn algorithm; svd++ algorithm; collaborative filtering; data mining; e-commerce*

PENDAHULUAN

Perkembangan pesat teknologi internet memberikan dampak signifikan pada ekonomi dan mendorong adopsi *e-commerce* di masyarakat (Purba & Suendri, 2024). Proyeksi nilai pasar *e-commerce* di Indonesia diperkirakan mencapai USD $55-$65 miliar pada tahun 2022 dengan pertumbuhan transaksi sebesar 46% dibandingkan tahun sebelumnya (Muslim et al., 2021). Tingginya tingkat persaingan di sektor ini memaksa perusahaan untuk terus berinovasi dalam meningkatkan pengalaman belanja pengguna (Syah, 2020). Selain itu, banyak pengguna menghadapi kesulitan dalam menemukan produk yang sesuai dengan preferensi mereka di tengah banyaknya pilihan yang tersedia (Aisha & Kusumawati, 2022). Untuk mengatasi hal ini, analisis rekomendasi produk yang relevan dapat meningkatkan pengalaman belanja dengan menganalisis riwayat pembelian dan demografi pengguna (Rubangi & Rianto, 2022). Rekomendasi personal berpotensi mengurangi kebingungan serta meningkatkan tingkat konversi dan loyalitas pelanggan (Rustiyana et al., 2021). Sebagai contoh, Amazon dan Tokopedia telah berhasil mengimplementasikan sistem rekomendasi yang efektif untuk meningkatkan pengalaman pengguna dan meningkatkan penjualan (Februariyanti et al., 2021).

Metode rekomendasi dibagi menjadi *content-based filtering*, *collaborative filtering*, dan *hybrid recommender system* (Muliadi & Lestari, 2019). *content-based filtering* dibangun berdasarkan asumsi bahwa pengguna menyukai produk atau barang dengan fitur tersedia di masa lalu dan masa depan (Saifudin & Widiyaningtyas, 2024). Selain itu, *content-based filtering* juga dapat menggunakan data pengguna yang ada (Putri et al., 2018). *Collaborative filtering* mengevaluasi item berdasarkan preferensi pengguna lain (Sari et al., 2020). *hybrid recommender system* menggabungkan metode *collaborative filtering* dan *content-based filtering* untuk optimalisasi (Arfisko & Wibowo, 2022). Setiap metode memiliki kelebihan dan kekurangan seperti *content-based filtering* terbatas pada eksplorasi item baru (Tewari, 2020), sedangkan *collaborative filtering* menghadapi tantangan seperti *cold start* dan *sparsity* (Natarajan et al., 2020). *Hybrid recommender system* mencoba mengatasi kelemahan masing-masing metode dengan menggabungkan pendekatan yang berbeda (Biswas & Liu, 2022). Penelitian ini memilih *Collaborative Filtering* karena kemampuannya memanfaatkan interaksi antar pengguna untuk memberikan rekomendasi yang relevan berdasarkan *rating* yang diberikan oleh pelanggan pada *e-commerce* (Halim et al., 2022).

Banyak penelitian telah menerapkan algoritma KNN, yang merupakan metode klasifikasi mengandalkan jarak terdekat antara objek dalam dataset (Purwani et al., 2022). Algoritma ini sering dipadukan dengan variasi similaritas, seperti *cosine similarity*. Penelitian oleh Andra & Baizal (2021) menggunakan KNN yang menghasilkan RMSE 0,771806 dan MAE 0,66265, sehingga menunjukkan kinerja baik dalam mengidentifikasi preferensi pengguna. Sementara itu, Rubangi & Rianto (2022) juga mengimplementasikan KNN yang menghasilkan akurasi 73,53%, *precision* 73,64%, dan *recall* 99,62% yang menunjukkan efektivitas sistem dalam merekomendasikan produk yang relevan. Penelitian Zulvian et al. (2021) membandingkan KNN dengan *cosine similarity* dan *Mean Squared Deviation* (MSD), menemukan bahwa kedua pendekatan menghasilkan nilai MAE serupa tetapi MSD lebih efisien dalam waktu eksekusi yang menunjukkan keunggulan untuk dataset berskala besar. Penelitian oleh (Syah, 2020) menunjukkan KNN dengan *cosine similarity* pada sistem rekomendasi "*Tokopedia Product Reviews*" dengan hasil RMSE 0,713 dan MAE 0,488.

KNN memiliki keunggulan dalam analisis rekomendasi, tetapi algoritma ini menghadapi kelemahan signifikan dalam mengatasi *sparsity* dan akurasi rendah. Ketergantungan KNN pada jumlah tetangga terdekat berpotensi menyebabkan *overfitting* dan mengakibatkan *noise* dalam prediksi (Andra & Baizal, 2021). Sebagai solusi, SVD++ menawarkan optimasi prediksi dengan memanfaatkan umpan balik implisit. Algoritma ini tidak hanya mempertimbangkan rating eksplisit berupa penilaian angka yang diberikan pengguna terhadap item, seperti yang diterapkan pada algoritma KNN tetapi juga *rating* implisit seperti perilaku *browsing historis*

pengguna (Wang et al., 2020). SVD++ memperbaiki kelemahan KNN dalam menangani *sparsity*. SVD++ memberikan pendekatan yang lebih komprehensif dibandingkan KNN yang hanya memanfaatkan kesamaan *rating* eksplisit antar pengguna. SVD++ selain mempertimbangkan umpan balik eksplisit juga mempertimbangkan umpan balik implisit, seperti pola perilaku pengguna. Pendekatan ini memungkinkan SVD++ mampu menangkap preferensi laten secara lebih akurat sehingga bisa menghasilkan rekomendasi yang tepat dan relevan. Penelitian ini memberikan wawasan baru dengan analisis komprehensif mengenai performa SVD++. Analisis dilakukan untuk membandingkan KNN dengan *cosine similarity* dan SVD++ dalam pendekatan *collaborative filtering*.

Penelitian ini bertujuan untuk menganalisis akurasi KNN dan SVD++ dalam merekomendasikan produk pada *e-commerce*. Selain itu, penelitian ini juga mengeksplorasi kemampuan SVD++ dalam memanfaatkan *feedback implisit* pengguna untuk meningkatkan kepuasan pengguna *e-commerce* dalam menemukan produk yang sesuai dengan preferensi mereka secara efisien. Diharapkan SVD++ akan memberikan akurasi yang lebih baik dibandingkan KNN untuk bisa memberikan pemahaman yang lebih mendalam tentang efektivitas kedua algoritma dan kontribusinya terhadap analisis rekomendasi *e-commerce.*

METODE

Penelitian ini menggunakan metode eksperimen dengan bahasa pemrograman *Python*. Pustaka yang digunakan adalah *pandas* untuk pembersihan data, *numpy* untuk perhitungan, dan *surprise* untuk mengevaluasi algoritma rekomendasi KNN serta SVD++. Penelitian ini dieksekusi pada *Google Colab*. Fokus penelitian adalah analisis rekomendasi produk di platform *e-commerce* dengan membandingkan performa KNN dan SVD++ dalam memberikan rekomendasi relevan. Penelitian terdiri dari empat tahap utama yaitu *data collection, data preprocessing, data processing,* dan *evaluation*. Pada tahap *data collection,* dataset yang digunakan adalah *Tokopedia Product Reviews* dari Kaggle, terdiri dari 40.893 data ulasan, *rating*, dan informasi produk seperti *id_product* dan *id_shop*. Atribut yang digunakan mencakup *rating* dan ulasan yang mencerminkan kepuasan pengguna, serta *id_product* dan *id_shop* sebagai identifikasi unik produk serta toko. Atribut tersebut berperan penting dalam menganalisis preferensi pengguna dan performa produk sehingga menjadi dasar analisis rekomendasi. Dataset tersebut dibuat pada tahun 2019 dan mencakup berbagai kategori produk. Dataset *Tokopedia Product Reviews* dipilih karena tersedia secara terbuka dan relevan dengan tren pengguna. *Tahap preprocessing* mencakup lima langkah yaitu menghapus duplikat untuk mencegah bias, mengganti *missing value* dengan rata-rata untuk menjaga stabilitas, menghapus baris yang mengandung NaN untuk integritas dataset, menormalkan rating ke skala 0-1 menggunakan *MinMaxScaler* pada persamaan (1) agar algoritma KNN dan SVD++ lebih optimal, serta menghapus outlier dengan metode *Interquartile Range* (IQR) untuk data tidak normal.

$$X' = \frac{\mathrm{X} \quad - X_{min}}{X_{max} - X_{min}} \tag{1}$$

Persamaan (1) *MinMaxScaler* digunakan untuk merubah skala data ke dalam rentang yang diinginkan dengan cara menormalkan nilai menggunakan nilai minimum dan maksimum dari setiap fitur. Dimana, X merupakan nilai asli dari data, X_{min} merupakan nilai minimum fitur pada data dan X_{max} merupakan nilai maksimum fitur pada data. Pada tahap *data processing*, algoritma KNN dengan *cosine similarity* dan SVD++ diterapkan untuk menghitung prediksi rating. Prediksi untuk KNN menggunakan persamaan (2), sedangkan SVD++ menggunakan persamaan (3). Dataset dibagi dengan rasio 80:20 menjadi *training set* dan

testing set untuk keseimbangan pelatihan dan evaluasi model. Teknik stratifikasi digunakan dalam pembagian dataset agar distribusi rating tetap seimbang. Algoritma KNN dikenal mudah diimplementasikan, namun memiliki keterbatasan pada dataset besar. Di sisi lain, SVD++ lebih kompleks dan unggul dalam mengatasi masalah sparsity. Kinerja model dinilai menggunakan metrik RMSE dan MAE. RMSE lebih sensitif terhadap kesalahan besar, sementara MAE memberikan rata-rata kesalahan. Kedua metrik ini saling melengkapi dalam mengevaluasi keakuratan model, sehingga dapat memberikan gambaran yang lebih komprehensif tentang efektivitas algoritma dalam memberikan rekomendasi.

$$\hat{r}_{ui} = \frac{\sum_{v \in N(u)} \mathrm{sim}(u, v) \cdot \Gamma_{v_i}}{\sum_{v \in N(u)} |\mathrm{sim}(u, v)|} \tag{2}$$

Persamaan (2) KNN menghitung prediksi rating sebagai rata-rata tertimbang dari rating tetangga terdekat untuk item i. Menggunakan *similarity* pengguna u dan pengguna tetangga v. Dimana, $N(u)$ merupakan kumpulan tetangga dari pengguna u , $sim(u, v)$ merupakan similaritas antara pengguna u dan v, dan Γ_{v_i} merupakan *rating* yang diberikan oleh pengguna v terhadap item i. Sedangkan, pada persamaan (3) SVD++ mengoptimalkan prediksi rating dengan mempertimbangkan faktor laten dari pengguna dan item. Dimana preferensi laten pengguna u merupakan P_u, sedangkan item i merupakan q_i. Selain itu, y_j untuk mempertimbangkan pengaruh item yang diinteraksi oleh pengguna u.

$$\hat{r}_{ui} = \mu + b_u + b_i + {q_i}^T \left(P_u + \frac{1}{\sqrt{N(u)}} \sum_{j \in N(u)} y_j \right) \tag{3}$$

HASIL DAN PEMBAHASAN

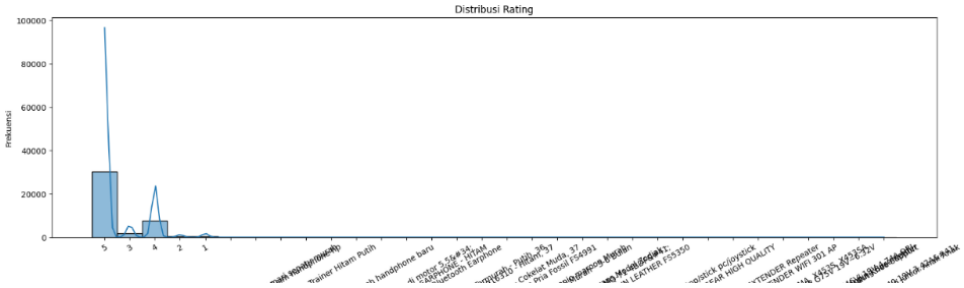
Hasil

Hasil pada tabel 1 merupakan representasi awal struktur data mentah yang digunakan dalam penelitian sebelum dilakukan proses pembersihan untuk mendukung analisis rekomendasi. Dataset awal terdiri dari 40.608 baris data dengan atribut *rating, product_id,* dan *shop_id*. Setiap entri data menunjukkan nilai *rating* yang diberikan oleh pengguna dengan skala 1 hingga 5 terhadap produk tertentu. *Product_id* digunakan sebagai identifikasi unik produk dan *shop_id* digunakan sebagai identifikasi unik toko. Beberapa atribut tambahan, seperti ulasan teks, kategori produk, dan *product_name* dihilangkan dalam tahap *preprocessing* untuk mengurangi kompleksitas komputasi. Beberapa atribut dihilangan karena tidak memberikan kontribusi signifikan terhadap performa model rekomendasi. Seluruh eksperimen dilakukan di *Google Colab* tanpa penggunaan GPU atau TPU, untuk menjaga fokus utama penelitian pada evaluasi metode analisis rekomendasi.

Gambar 1 menunjukkan distribusi *rating* yang tidak merata sebelum *preprocessing*, dengan beberapa nilai dalam dataset berupa *string*, seperti karakter non-numerik atau teks yang seharusnya berisi angka pada *rating*. Hal tersebut dapat mencemari kualitas data dan menyebabkan kesalahan dalam analisis, karena algoritma tidak dapat menginterpretasikan nilai-nilai yang tidak valid. Ketidakakuratan ini dapat memengaruhi hasil analisis dan menghasilkan prediksi yang tidak akurat, karena algoritma berusaha mempelajari pola dari data yang kurang valid. Selain itu, keragaman kategori produk dan toko menambah tantangan dalam pemrosesan. Kondisi ini menegaskan perlunya *preprocessing* yang baik untuk memperbaiki kualitas data melakukan analisis rekomendasi. Dengan demikian, penelitian ini menekankan pentingnya pengolahan data yang tepat untuk memastikan hasil yang akurat dan efektif.

Tabel 1. Data mentah sebelum dilakukan *preprocessing*

No	*Rating*	*Product_id*	*Shop_id*
1	5	418660637	1740837
2	5	416032545	1477109
3	5	416032545	1477109
4	5	102279869	771395
5	5	418660637	1740837
…..	…..	…..	…..
40607	1	201690705	648559
40608	1	78857201	648559



Gambar 1. Distribusi rating sebelum *preprocessing*

```
# Menghilangkan duplikat
df_cleaned = df.drop_duplicates()
print("Jumlah data setelah menghapus duplikat:", df_cleaned.shape[0])

# Konversi yang bukan numerik menjadi NaN
df_cleaned.loc[:, 'rating'] = pd.to_numeric(df_cleaned['rating'],
                                            errors='coerce')

# Ganti NaN dengan rata-rata rating
rating_mean = df_cleaned['rating'].mean()
df_cleaned.loc[:, 'rating'] = df_cleaned['rating'].fillna(rating_mean)

df_cleaned = df_cleaned.infer_objects()

# Ganti missing values di kolom lain dengan rata-rata
df_cleaned.loc[:, 'shop_id'] = pd.to_numeric(df_cleaned['shop_id'], errors='coerce').fillna(df_cleaned['shop_id'].mean())
df_cleaned.loc[:, 'product_id'] = pd.to_numeric(df_cleaned['product_id'], errors='coerce').fillna(df_cleaned['product_id'].mean())

print("Jumlah data setelah mengganti missing values dengan rata-rata:", df_cleaned.shape[0])

# Normalisasi rating dengan skala 0-1
scaler = MinMaxScaler()
df_cleaned.loc[:, 'rating'] = scaler.fit_transform(df_cleaned[['rating']])

print("Jumlah data setelah normalisasi rating:", df_cleaned.shape[0])

# Cek dan hapus outlier menggunakan IQR
Q1 = df_cleaned['rating'].quantile(0.25)
Q3 = df_cleaned['rating'].quantile(0.75)
IQR = Q3 - Q1
df_cleaned = df_cleaned[~((df_cleaned['rating'] < (Q1 - 1.5 * IQR)) | (df_cleaned['rating'] > (Q3 + 1.5 * IQR)))]
print("Jumlah data setelah menghapus outlier:", df_cleaned.shape[0])
```

Gambar 2. Data *preprocessing*

Tahap *preprocessing data* dilakukan untuk membersihkan dan menyiapkan data mentah sebelum diolah oleh model. Gambar 2 menunjukkan tahap *preprocessing* yang digunakan. Pertama, duplikasi data dihapus untuk mencegah bias dan *overfitting*. *Rating* yang tidak bernilai numerik diubah menjadi NaN menggunakan fungsi *pd.to_numeric*, kemudian nilai NaN diganti dengan rata-rata *rating* untuk menjaga stabilitas data, menghindari pengaruh negatif dari nilai ekstrem seperti 0. Selanjutnya, normalisasi dilakukan menggunakan *MinMaxScaler* untuk mengubah skala *rating* ke rentang 0-1, penting bagi algoritma seperti KNN dan SVD++ agar bekerja lebih baik dengan data yang konsisten. Penghapusan *outlier* dilakukan menggunakan metode IQR, yang efektif dalam distribusi tidak normal dan mengidentifikasi *outlier* di luar batas 1,5 kali jarak *interkuartil*. Penggunaan metode IQR dipakai karena metode lain seperti *z-score* karena IQR tidak terpengaruh oleh distribusi data yang tidak simetris atau data yang memiliki distribusi panjang ekor. Meskipun metode IQR bisa terlalu ketat pada distribusi sempit, teknik ini dianggap paling sesuai untuk menjaga integritas data.

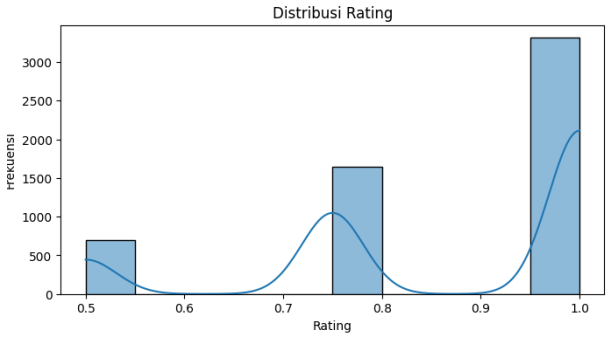
Sementara itu, gambar 3 menunjukkan perubahan jumlah data setiap *tahap preprocessing*. Awalnya terdapat 40.607 data, namun 34.414 dihapus karena duplikasi,

sehingga tersisa 6.193 data. Pada penggantian nilai hilang dengan rata-rata dan normalisasi rating menggunakan *MinMaxScaler*, jumlah data tetap sama karena langkah-langkah ini tidak mengeliminasi data, hanya menggantikan nilai hilang dan menyesuaikan skala. Namun, pada tahap penghapusan *outlier* menggunakan metode IQR, jumlah data berkurang menjadi 5.678. Pengurangan besar pada tahap awal disebabkan oleh banyaknya duplikasi, sementara langkah-langkah selanjutnya bertujuan meningkatkan kualitas data tanpa mengurangi jumlahnya, dan penghapusan *outlier* dilakukan untuk mengurangi anomali yang mengganggu analisis.

```
Jumlah data sebelum preprocessing: 40607
Jumlah data setelah menghapus duplikat: 6193
Jumlah data setelah mengganti missing values dengan rata-rata: 6193
Jumlah data setelah normalisasi rating: 6193
Jumlah data setelah menghapus outlier: 5678
```

Gambar 3. Jumlah data tiap tahap *preprocessing*

Selanjutnya hasil pada gambar 4 menunjukkan distribusi *rating* setelah normalisasi, di mana skala *rating* diubah dari 1-5 menjadi 0-1 menggunakan *MinMaxScaler*. Meskipun distribusi tetap serupa dengan data awal, skala yang lebih merata terlihat setelah normalisasi. Proses ini penting karena memungkinkan algoritma seperti KNN dan SVD++ bekerja lebih optimal dengan data dalam rentang yang sama, sehingga model dapat memahami variasi antar *rating* dengan lebih baik dan diharapkan dapat meningkatkan akurasi.



Gambar 4. Distribusi *rating* setelah normalisasi

Tabel 2 menunjukkan hasil setelah proses *preprocessing*, *rating* telah dinormalisasi menggunakan skala 0-1 untuk memastikan data memiliki rentang yang seragam, sehingga memudahkan model dalam memproses informasi secara lebih akurat. Selain itu *product_id* dan *shop_id* yang tetap dipertahankan sebagai fitur penting untuk model rekomendasi. Jumlah data setelah seluruh tahapan *preprocessing*, termasuk penghapusan data duplikat, penanganan nilai yang hilang, normalisasi, dan penghapusan *outlier*, tersisa 5.678 data. Proses ini dilakukan untuk menghilangkan *noise* pada data, memastikan efisiensi dalam komputasi, serta meningkatkan akurasi. Data yang bersih dapat digunakan dalam tahap pemodelan lebih lanjut.

Dataset dibagi menjadi *training set* dan *testing set* dengan rasio 80:20. Pembagian ini bertujuan melatih model menggunakan 80% data dan menguji performa model dengan 20% untuk memastikan model memiliki cukup data untuk belajar sambil tetap mempertahankan sejumlah data yang cukup untuk menguji akurasi dan generalisasi model tersebut. Setelah pembagian, data dimasukkan ke dalam algoritma KNN dan SVD++. Gambar 5 menunjukkan tahap pemrosesan algoritma KNN dengan *similarity* berbasis *cosine*, yang menghitung kedekatan pengguna berdasarkan rating. Proses *cross-validation* digunakan untuk menghitung RMSE dan MAE. Gambar 6 menunjukkan model SVD++ yang juga diinisialisasi dan dievaluasi dengan *cross-validation* untuk RMSE dan MAE. Hasil evaluasi dicatat dalam variabel results, memberikan gambaran akurasi dan efektivitas model. Pendekatan ini memungkinkan perbandingan langsung kedua algoritma berdasarkan performa yang diukur

melalui metrik relevan, sehingga model yang paling sesuai untuk analisis rekomendasi dapat dipilih.

Tabel 2. Data setelah dilakukan *preprocessing*

No	*Rating*	*Product_id*	*Shop_id*
1	1,0	418660637	1740837,0
2	1,0	416032545	1477109,0
3	1,0	102279869	771395,0
4	1,0	190679689	969999,0
5	1,0	316217334	231740,0
…..	…..	…..	…..
5650	1,0	178350483	648559,0
5678	0,5	56759578	648559,0

```
# Menghitung Similaritas menggunakan Cosine
sim_options = {
    'name': 'cosine',
    'user_based': False
}

# Menggunakan KNN
algo = KNNBasic(sim_options=sim_options)

#perform cross validation
cv_results = cross_validate(algo, data, measures=['RMSE', 'MAE'],
                            cv=5, verbose=True)
results.append({
    'Algorithm': 'KNNBasic (cosine)',
    'MAE': cv_results['test_mae'].mean(),
    'RMSE': cv_results['test_rmse'].mean()
})
```

Gambar 5. *Data processing* knn

```
algo = SVDpp()

#perform cross validation
cv_results = cross_validate(algo, data,
                            measures=['RMSE', 'MAE'],
                            cv=5, verbose=True)
results.append({
    'Algorithm': 'SVDpp',
    'MAE': cv_results['test_mae'].mean(),
    'RMSE': cv_results['test_rmse'].mean()
})
```

Gambar 6. Data *processing* svd++

Tabel 3. Hasil Akurasi

Algoritma	*MAE*	*RMSE*
KNN	0,163964	0,197045
SVD++	0,161176	0,185252

Hasil akurasi pada tabel 3 menunjukkan kinerja algoritma KNN dan SVD++. Berdasarkan nilai MAE dan RMSE, SVD++ menunjukkan performa lebih baik dengan MAE 0,161176 dan RMSE 0,185252, disbanding dengan KNN yang mencatat MAE 0,163964 dan RMSE 0,197045. Selisih MAE 0,002788 dan RMSE 0,011793 menandakan SVD++ memberikan prediksi lebih akurat dan konsisten, yang berdampak pada peningkatan relevansi rekomendasi dan potensi konversi penjualan. KNN berisiko terpengaruh oleh *noise* dan *sparsity data*, mengurangi kualitas rekomendasi, terutama pada dataset besar, karena algoritma ini kesulitan menangkap pola kompleks tanpa fitur tambahan. Selanjutnya, penerapan teknik

hybrid yang menggabungkan kekuatan kedua algoritma serta eksplorasi algoritma lain seperti *matrix factorization* atau *deep learning* dapat meningkatkan akurasi dan efisiensi model rekomendasi.

Pembahasan

Hasil evaluasi menunjukkan bahwa algoritma SVD++ memiliki MAE sebesar 0,161176 dan RMSE 0,185252. Nilai ini sedikit lebih baik dibandingkan KNN dengan MAE 0,163964 dan RMSE 0,197045. Perbedaan ini menunjukkan bahwa SVD++ lebih unggul dalam memprediksi rating produk dengan akurasi yang lebih tinggi. Hasil ini berdampak positif pada relevansi rekomendasi dan pengalaman pengguna. SVD++ terbukti mampu memodelkan pola interaksi laten antara pengguna dan produk, sedangkan KNN hanya bergantung pada kesamaan tetangga. SVD++ juga lebih efektif menghadapi masalah *sparsity* dengan memanfaatkan umpan balik implisit dan memodelkan hubungan kompleks antara pengguna dan item.

Meskipun penelitian sebelumnya menunjukkan keunggulan KNN, hasil penelitian ini menunjukkan bahwa SVD++ lebih efektif dalam menangani kompleksitas data. Penelitian oleh Syah (2020) menghasilkan RMSE 0,713 dan MAE 0,488, Andra & Baizal (2021) menghasilkan RMSE 0,771806 dan MAE 0,66265, sementara Rubangi & Rianto (2022) menghasilkan akurasi 73,53%, *precision* 73,64%, dan *recall* 99,62%. Meskipun baik dalam mengidentifikasi preferensi pengguna, penelitian-penelitian tersebut menghadapi keterbatasan dalam menangani *sparsity* dan sering terpengaruh oleh *noise*. Dalam penelitian ini, SVD++ diutamakan karena kemampuannya memodelkan interaksi laten dengan mempertimbangkan umpan balik implisit, yang berdampak positif pada pengalaman pengguna dalam menemukan produk yang sesuai. *Preprocessing* seperti penghapusan duplikat, penanganan missing values dengan rata-rata, normalisasi *rating*, dan penghapusan *outlier* sangat penting untuk meningkatkan kinerja model. Pendekatan ini mengurangi risiko kehilangan informasi penting meskipun data dikurangi dari 40.893 menjadi 5.651.

Perbedaan MAE dan RMSE menunjukkan bahwa meskipun SVD++ lebih baik juga memiliki kelemahan, seperti ketergantungan pada umpan balik implisit yang dapat menyebabkan bias jika informasi tidak seimbang. Penelitian lebih lanjut diperlukan untuk mengatasi dampak bias pada rekomendasi. Eksplorasi *teknik ensemble* yang menggabungkan model *content-based filtering* dan *collaborative filtering*, terbukti efektif dalam mengatasi ketergantungan pada umpan balik implisit (Raghavendra & Srikantaiah, 2022). Hasil penelitian ini diharapkan dapat memberikan kontribusi signifikan dalam analisis rekomendasi *e-commerce* serta membuka peluang eksplorasi dalam pengembangan algoritma rekomendasi yang lebih kompleks, sekaligus memberikan wawasan bagi praktisi dalam merumuskan strategi pemasaran yang lebih efektif dan responsif terhadap perilaku konsumen.

SIMPULAN

Hasil penelitian menunjukkan bahwa algoritma SVD++ terbukti lebih efektif dibandingkan KNN dalam rekomendasi *e-commerce*. Hasil menunjukkan bahwa SVD++ memiliki nilai MAE 0,161176 dan RMSE 0,185252 lebih rendah daripada KNN yang mencapai MAE 0,163964 dan RMSE 0,197045. SVD++ lebih baik dalam menangkap interaksi kompleks antara pengguna dan produk, sehingga meningkatkan relevansi rekomendasi. Penelitian ini berkontribusi bahwa SVD++ lebih sesuai untuk rekomendasi *e-commerce* yang membutuhkan akurasi tinggi dan personalisasi mendalam. Namun, ketergantungan SVD++ pada umpan balik implisit dapat menimbulkan bias. Oleh karena itu, penelitian selanjutnya disarankan untuk mengeksplorasi *teknik ensemble* dan menggunakan dataset yang lebih besar untuk menguji skalabilitas dan akurasi model prediksi.

REFERENSI

Aisha, D., & Kusumawati, R. (2022). Implementasi Metode Algoritma Collaborative Filtering dan K-Nearest Neighbor pada Sistem Rekomendasi E-Commerce. *Juisikjurnal Ilmiah Sistem Informasi Dan Ilmu Komputer*, *2*(3), 25–38. https://doi.org/10.55606/juisik.v2i3.314

Andra, D., & Baizal, A. B. (2022). E-commerce Recommender System Using PCA and K-Means Clustering. *Jurnal Resti (Rekayasa Sistem Dan Teknologi Informasi)*, *5*(2), 57–63. https://doi.org/10.29207/resti.v6i1.3782

Arfisko, H. H., & Wibowo, A. T. (2022). Sistem Rekomendasi Film Menggunakan Metode Hybrid Collaborative Filtering Dan Content-Based Filtering. *E-Proceeding of Engineering*, *9*(3), 2149–2159.

Biswas, P. K., & Liu, S. (2022). A Hybrid Recommender System for Recommending Smartphones to Prospective Customers. *Expert Systems with Applications*, *208*. https://doi.org/10.1016/j.eswa.2022.118058

Februariyanti, H., Laksono, A. D., Wibowo, J. S., & Utomo, M. S. (2021). Implementasi Metode Collaborative Filtering untuk Sistem Rekomendasi Penjualan pada Toko Mebel. *Jurnal Khatulistiwa Informatika*, *9*(1), 43–50.

Halim, F., Wijaya, A. H., & Wiyono. (2022). Analisis dan Perancangan E-Commerce Berbasis Web Dengan Penerapan Sistem Perekomendasian Menggunakan Metode Collaborative Filtering Serta Metode Up, Down, Cross Selling. *Jurnal Algor*, *4*(1). https://doi.org/10.31253/algor.v4i1.1516

Muliadi, K. H., & Lestari, C. C. (2019). Rancang Bangun Sistem Rekomendasi Tempat Makan Menggunakan Algoritma Typicality Based Collaborative Filtering Engineering of a Dining Place Recommendation System Using Typicality Based Collaborative Filtering Algorithm. *Jurnal Teknologi Informasi Techno.Com*, *18*(4), 275–287. https://doi.org/10.33633/tc.v18i4.2515

Muslim, I. A., Purnandi, H., Hazna, C. R., Atmaja, S. A., & Putra, I. E. (2021). Penggunaan Sistem E-Commerce Dalam Meningkatkan Daya Saing Pelaku Bisnis Dalam Perkembangan Dunia Usaha Studi Kasus Aplikasi Onlineshop Tokomobile. *Jurnal Teknik Informatika Dan Sistem Informasi*, *8*(3), 1651–1664. https://doi.org/10.35957/jatisi.v8i3.1077

Natarajan, S., Vairavasundaram, S., Natarajan, S., & Gandomi, A. H. (2020). Resolving data sparsity and cold start problem in collaborative filtering recommender system using Linked Open Data. *Expert Systems with Applications*, *149*. https://doi.org/10.1016/j.eswa.2020.113248

Purba, P. M., & Suendri, S. (2024). Aplikasi E-Comerce Produk UMKM menggunakan Metode Filtrasi Kolaboratif berbasis Mobile. *Edumatic: Jurnal Pendidikan Informatika*, *8*(1), 300–309. https://doi.org/10.29408/edumatic.v8i1.25880

Purwani, F., Wahyudi, R. T., & Jaya, I. D. (2022). Penerapan Algoritma K-Nearest Neighbor dengan Euclidean Distance untuk Menentukan Kelompok Uang Kuliah Tunggal Mahasiswa. *Edumatic: Jurnal Pendidikan Informatika*, *6*(2), 344–353. https://doi.org/10.29408/edumatic.v6i2.6547

Putri, M. W., Muchayan, A., & Kamisutara, M. (2018). Sistem Rekomendasi Produk Pena Eksklusif Menggunakan Metode Content-Based Filtering dan TF-IDF. *Jointecs (Journal of Information Technology and Computer Science)*, *3*(1), 229–236.

Raghavendra, C. K., & Srikantaiah, K. C. (2022). Weighted Hybrid Model for Improving Predictive Performance of Recommendation Systems Using Ensemble Learning. *Indian Journal of Computer Science and Engineering*, *13*(2), 513–524. https://doi.org/10.21817/indjcse/2022/v13i2/221302133

Rubangi, & Rianto. (2022). Sistem Rekomendasi Pada Tokopedia Menggunakan Algoritma K-Nearest Neighbor. *Jurnal Teknik Komputer Amik Bsi*, *8*(1), 103–107. https://doi.org/10.31294/jtk.v4i2

Saifudin, I., & Widiyaningtyas, T. (2024). Systematic Literature Review on Recommender System: Approach, Problem, Evaluation Techniques, Datasets. *IEEE*, *12*, 19827–19847. https://doi.org/10.1109/ACCESS.2024.3359274

Sari, R. K., Suharso, W., & Azhar, Y. (2020). Pembuatan Sistem Rekomendasi Film dengan Menggunakan Metode Item Based Collaborative Filtering pada Apache Mahout. *Repositor*, *2*(6), 767–774. https://doi.org/10.22219/repositor.v2i6.30715

Syah, R. D. (2020). Performa Algoritma User K-Nearest Neighbors pada Sistem Rekomendasi di Tokopedia. *Jurnal Informatika Universitas Pamulang*, *5*(3), 302–306. https://doi.org/10.32493/informatika.v5i3.6312

Tewari, A. S. (2020). Generating Items Recommendations by Fusing Content and User-Item based Collaborative Filtering. *Procedia Computer Science*, *167*, 1934–1940. https://doi.org/10.1016/j.procs.2020.03.215

Wang, S., Sun, G., & Li, Y. (2020). SVD++ Recommendation Algorithm Based on Backtracking. *Information (Switzerland)*, *11*(7). https://doi.org/10.3390/info11070369

Zulvian, S. A., Prihandani, K., & Ridha, A. A. (2021). Perbandingan Metode MSD dan Cosine Similarity pada Sistem Rekomendasi Item-Based Collaborative Filtering. *Journal of Information Technology and Computer Science (Intecoms)*, *4*(2), 340–347. https://doi.org/10.31539/intecoms.v4i2.2781