JITET (Jurnal Informatika dan Teknik Elektro Terapan)
Vol. 12 No. 3, pISSN: 2303-0577 eISSN: 2830-7062 http://dx.doi.org/10.23960/jitet.v12i3.4738

KOMPARASI KINERJA METODE *COSINE* DAN *JACCARD SIMILARITY* DALAM *CONTENT-BASED RECOMMENDATION SYSTEMS* (CBRS) PADA APLIKASI EVENTHINGS

Arsya Amalia Ristias[1*], Eka Dyar Wahyuni[2], Seftin Fitri Ana Wati[3]

[1,2,3]Universitas Pembangunan Nasional Veteran Jawa Timur; Jl. Rungkut Madya No.1, Gn. Anyar, Kec. Gn. Anyar, Surabaya, Jawa Timur 60294; (031) 8706369

Received: 2 Juli 2024
Accepted: 31 Juli 2024
Published: 7 Agustus 2024

Keywords:
Content-Based Recommendation Systems;
Cosine Similarity;
Jaccard Similarity;
Term Weighting;
Eventhings.

Corespondent Email:
arsyaamalia1@gmail.com

Abstrak. Industri perencanaan acara di Indonesia mengalami pertumbuhan pesat, mendorong kebutuhan akan platform yang efisien untuk menghubungkan penyelenggara acara dengan vendor dan layanan yang tepat. Eventhings hadir sebagai platform yang menyederhanakan proses tersebut. Penelitian ini bertujuan untuk mengembangkan Content-Based Recommendation System (CBRS) untuk merekomendasikan vendor yang optimal berdasarkan kemiripan vendor yang telah dipilih sebelumnya. Dengan menggunakan dua metode kemiripan (similarity), yakni Cosine dan Jaccard Similarity, serta teknik-teknik pembobotan diimplementasikan dan dikomparasikan untuk menentukan metode dan teknik yang paling optimal dalam meningkatkan kualitas rekomendasi. Hasil penelitian dengan diversity evaluation metric menunjukkan bahwa teknik pembobotan TF-PDF dengan metode Cosine Similarity menghasilkan rekomendasi yang lebih akurat dan relevan (66%) dibandingkan Jaccard Similarity (49%).

Abstract. *The event planning industry in Indonesia is experiencing rapid growth, driving the need for an efficient platform to connect event organizers with the right vendors and services. Eventhings emerges as a platform to streamline this process. However, its current features are insufficient in enhancing user experience, particularly in the aspect of vendor recommendations for new users. This research aims to develop a Content-Based Recommendation System (CBRS) to recommend optimal vendors based on the similarity to previously selected vendors. By using two similarity methods, Cosine and Jaccard Similarity, as well as implementing and comparing various weighting techniques, the study seeks to determine the most effective method and technique for improving the quality of recommendations. The research results, evaluated using diversity evaluation metrics, indicate that the TF-PDF weighting technique combined with the Cosine Similarity method yields more accurate and relevant recommendations (66%) compared to Jaccard Similarity (49%).*

1. PENDAHULUAN

Pertumbuhan startup di Indonesia, khususnya dalam e-commerce, transportasi online, wisata, dan perjalanan, telah mendorong ekonomi digital [1]. Contohnya adalah kesuksesan Gojek, Bukalapak, Shopee, Tokopedia, dan Grab yang mengandalkan transformasi digital untuk memenuhi kebutuhan masyarakat. Transformasi digital ini mengubah kinerja bisnis organisasi dengan memanfaatkan teknologi, sumber daya manusia, dan proses bisnis [2].

Kemajuan teknologi informasi telah mengubah banyak industri menjadi digital [3], termasuk industri kreatif dan penyelenggaraan acara. Peningkatan gaya hidup masyarakat yang semakin canggih membuat industri ini harus mengikuti tren untuk terus berkembang. Pada tahun 2023, terdapat 3.000 acara di Indonesia dengan potensi ekonomi mencapai Rp 162 Triliun [4], dimana UMKM memainkan peran penting dalam menyediakan layanan untuk acara tersebut.

Pandemi COVID-19 mengganggu sektor pariwisata dan industri kreatif, menyebabkan penurunan kinerja bisnis dan banyak perusahaan mengalami kesulitan. Penyedia acara masih menggunakan media sosial untuk memasarkan produk mereka [5], namun menemukan dan memilih vendor yang tepat tetap menjadi tantangan.

Eventhings hadir sebagai solusi digital untuk mempermudah perencanaan acara dengan menghubungkan penyelenggara acara dengan vendor melalui aplikasi android. Meskipun telah berinovasi, masih ada peluang untuk meningkatkan rekomendasi vendor yang lebih personal dan efisien. Penerapan Content-Based Recommendation Systems (CBRS) dapat menjadi salah satu pendekatan untuk mengatasi masalah ini. CBRS menganalisis konten item untuk memberikan rekomendasi yang sesuai berdasarkan preferensi pengguna dan karakteristik item.

Penelitian ini bertujuan untuk mengembangkan dan mengimplementasikan CBRS dalam aplikasi Eventhings dengan membandingkan metode Cosine Similarity dan Jaccard Similarity guna meningkatkan kualitas rekomendasi vendor.

2. TINJAUAN PUSTAKA

Penelitian ini berdasarkan teori-teori pendukung seperti Eventhings, Python, *scraping*, *crawling*, *text mining*, *pre-process*, *content - based recommendation systems* (CBRS), Cosine Similarity, Jaccard Similarity, TF-IDF, TF-RF, TF-ABS, TF-PDF, dan Diversity.

2.1. Eventhings

Eventhings adalah platform digital untuk menghubungkan penyelenggara acara dengan mitra media, sponsor, dan penyedia layanan lainnya di Indonesia. Fitur-fiturnya mencakup penemuan vendor, pemesanan peralatan, dan manajemen pembayaran, dengan fokus pada transparansi, kepercayaan, dan kualitas.

2.2. Python

Python adalah bahasa pemrograman yang kuat dan mudah digunakan, dengan dukungan untuk berbagai paradigma pemrograman seperti berorientasi objek dan fungsional (Matthes, 2019) [6].

2.3. *Scraping*

Scraping adalah proses ekstraksi data otomatis dari halaman web menggunakan perangkat lunak seperti BeautifulSoup atau Scrapy untuk mengurai data dari kode HTML atau XML (Jarmul & Lawson, 2017) [7].

2.4. *Crawling*

Crawling mengacu pada pengumpulan data dengan memasukkan URL yang mengarah pada situs web tujuan, kemudian melanjutkan ke link terkait menggunakan program atau API (Laraswati, 2022) [8].

2.5. *Text Mining*

Text mining adalah proses ekstraksi informasi atau pengetahuan dari teks yang tidak terstruktur, menggunakan teknik seperti clustering, klasifikasi, dan analisis sentimen untuk mengidentifikasi pola dan hubungan dalam data teks besar (Jo, 2018) [9].

2.6. *Pre-Processing*

Pre-processing adalah tahap penting dalam analisis data, mencakup penghapusan data yang hilang, normalisasi data, pengkodean ulang data kategorikal, dan penanganan outlier untuk memastikan data berkualitas tinggi (Garcia et al., 2014) [10].

2.7. *Content-Based Recommendation Systems* (CBRS)

CBRS memprediksi preferensi pengguna berdasarkan karakteristik atau konten item yang relevan, menggunakan teknik seperti analisis teks, pengindeksan & pengelompokan (Brusilovsky & Kobsa, 2007) [11].

2.8. *Cosine Similarity*

Cosine Similarity mengukur kesamaan antara dua vektor dengan menghitung sudut di antara keduanya, sering digunakan dalam

Information Retrieval dan NLP untuk membandingkan kesamaan dokumen atau fitur (Han et al., 2011) [12].

2.9. *Jaccard Similarity*

Jaccard Similarity mengukur kesamaan antara dua himpunan berdasarkan jumlah elemen yang sama dibandingkan dengan total elemen dalam himpunan, sering digunakan dalam analisis ekologi (Zuur et al., 2007) [13].

2.10. *Term Weighting*

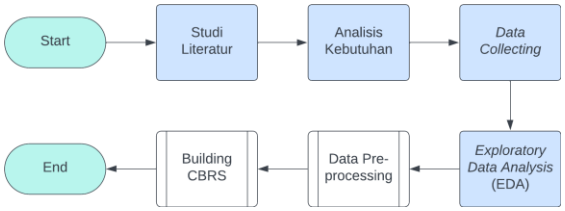
Pembobotan kata ini bertujuan untuk memperoleh nilai dari kata dasar yang telah berhasil diekstrak. Kata-kata dasar tersebut kemudian diubah menjadi vektor yang mencerminkan representasi kata-kata tersebut. Dalam penelitian ini, metode pembobotan Term Frequency (TF) digunakan untuk mengukur seberapa penting suatu kata dalam konteks analisis [14].

2.11. Diversity

Diversity dalam sistem rekomendasi mengacu pada upaya menghadirkan variasi dalam daftar rekomendasi, membantu pengguna menemukan item yang lebih sesuai dengan minat mereka (Kunaver & Požrl, 2017) [15].

3. METODE PENELITIAN

Pada Bab ini membahas tahapan-tahapan yang digunakan dalam penelitian ini agar terstruktur dengan baik. Tahapan - tahapan metodologi yang dilakukan dalam penelitian ini adalah seperti Gambar 1 berikut.



Gambar 1. Alur Metode Penelitian

3.1. Studi Literatur

Studi literatur merupakan tahap awal dalam penelitian ini yang melibatkan pengumpulan data dan kajian pustaka dari berbagai sumber. Sumber yang digunakan meliputi jurnal, buku, artikel ilmiah, dan penelitian terkait. Kegiatan ini mencakup membaca, mengolah, mencatat, dan mengelola bahan penelitian untuk memperoleh landasan teori yang kuat dan relevan dengan topik penelitian.

3.2. Analisis Kebutuhan Sistem

Tahap analisis kebutuhan sistem mencakup identifikasi kebutuhan data serta perangkat lunak dan keras yang diperlukan untuk mencapai tujuan penelitian.

3.3. Pengumpulan Data

Data untuk penelitian ini diperoleh melalui proses scraping dari website indonetwork (https://www.indonetwork.co.id/k/). Proses scraping dilakukan menggunakan Google Extension Web Scraper - Free Web Scraping, yang memungkinkan pengambilan data perusahaan di Indonesia dalam berbagai kategori. Data yang dikumpulkan termasuk nama perusahaan dan deskripsi, yang kemudian digunakan sebagai basis untuk merancang sistem rekomendasi berbasis konten (Content-Based Recommendation Systems, CBRS). Proses ini memastikan bahwa data yang digunakan relevan dan sesuai dengan kebutuhan penelitian.

3.4. *Exploratory Data Analysis* (EDA)

Exploratory Data Analysis (EDA) adalah tahap penting untuk memahami karakteristik data yang diperoleh dan mendeteksi adanya missing values. EDA membantu dalam mengoptimalkan pengetahuan mengenai data melalui berbagai metode, termasuk visualisasi grafis. Visualisasi ini memungkinkan peneliti untuk melihat pola, tren, dan anomali dalam data, sehingga dapat membuat keputusan yang lebih baik dalam tahap pemrosesan dan pemodelan data selanjutnya.

3.5. *Pre-Processing Data*

Sebelum data digunakan untuk pemodelan, perlu dilakukan pre-processing untuk memastikan bahwa data bersih dan terstruktur. Proses ini meliputi beberapa tahapan, yaitu data cleaning, case folding, tokenization, stopword removal, dan stemming. Data cleaning menghilangkan data yang tidak lengkap, duplikat, atau tidak relevan. Case folding menyeragamkan teks menjadi huruf kecil. Tokenization memisahkan teks menjadi token individu. Stopword removal menghapus kata-kata yang tidak signifikan menggunakan NLTK, dan stemming mengubah kata menjadi

bentuk dasar menggunakan library Sastrawi. Proses ini memastikan bahwa data siap digunakan untuk tahap pemodelan.

3.6. Pemodelan Sistem Rekomendasi Berbasis Konten

Tahap ini melibatkan pembangunan sistem rekomendasi berbasis konten melalui beberapa langkah. Pertama, data dibobot menggunakan teknik Term Frequency-Inverse Document Frequency (TF-IDF), Term Frequency-Reversed Frequency (TF-RF), Term Frequency-Absolute Frequency (TF-ABS), dan Term Frequency-Probability Density Function (TF-PDF) untuk mengukur pentingnya kata-kata dalam dokumen. Kemudian, nilai kesamaan dihitung menggunakan Cosine Similarity dan Jaccard Similarity. Hasilnya dievaluasi menggunakan metrik Diversity untuk mengukur keberagaman rekomendasi. Evaluasi ini penting untuk memastikan bahwa sistem memberikan rekomendasi yang akurat dan relevan bagi pengguna. Jika diperlukan, model akan disempurnakan berdasarkan hasil evaluasi sebelum diterapkan.

4. HASIL DAN PEMBAHASAN

Pada Bab Hasil dan Pembahasan ini membahas hasil dan evaluasi dari penelitian yang telah dilakukan tentang perbandingan kinerja metode *Cosine* dan *Jaccard Similarity* dalam *Content - Based Recommendation Systems* (CBRS) pada aplikasi Eventhings.

4.1. Implementasi Kebutuhan

Pada penelitian ini, data yang telah digunakan adalah data pemilik bisnis/usaha (vendor) seperti nama, deskripsi, kategori layanan, dan subkategori layanan. Selain itu, penelitian ini menggunakan perangkat keras berupa HP Laptop 14s-fq2xxx dengan spesifikasi tertentu serta perangkat lunak seperti Google Colab, Chrome, Python, Visual Studio Code, dan Microsoft Excel atau Google Spreadsheet untuk mendukung proses perancangan dan pelaksanaan sistem.

4.2. Pengumpulan Data

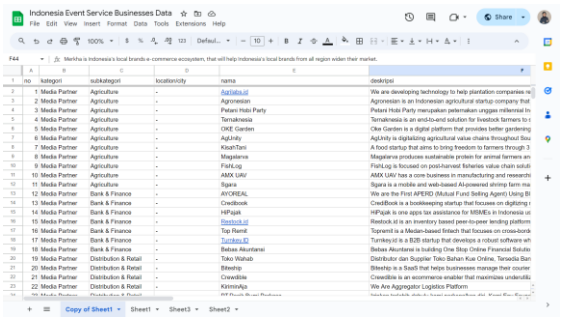
Tahap pengumpulan data melibatkan beberapa sumber. Pertama, data diperoleh dari hasil crawling Google Places API yang menampilkan data usaha/bisnis seperti wedding vendor, printing, lighting rental, tenda, buket, dan photo booth. Proses crawling menggunakan Python pada Google Colaboratory. Data yang dikumpulkan meliputi 9 kolom seperti Place ID, Name, Address, Contact, URL, Service/Business, District, Lat, dan Lng. Setelah penyaringan manual, terkumpul 2.800 data.

Sumber kedua berasal dari scraping website IndoNetwork yang menyediakan data perusahaan dari berbagai kategori di Indonesia. Proses scraping menggunakan Google Extension Web Scraper dengan sitemap. Data yang diperoleh didistribusikan ke dalam 8 kolom, seperti web-scraper-order, web-scraper-start-url, kategori, kategori-href, perusahaan, perusahaan-href, nama, dan deskripsi. Setelah penyaringan, terkumpul 1.158 data.

Sumber ketiga berasal dari scraping website Indonesia-Investments yang menyediakan data profil perusahaan di Indonesia. Proses scraping juga menggunakan Google Extension Web Scraper dengan sitemap. Data didistribusikan ke dalam 5 kolom: name, description, industry-sector, industry-subsector, dan detail-contact. Setelah penyaringan, terkumpul 335 data.

Sumber keempat berasal dari scraping website AirTable yang menyediakan database startup di Indonesia. Proses scraping menggunakan Airtable Extractor by Table Capture. Data didistribusikan ke dalam 3 kolom: Startup Name, Startup Description, dan Category. Setelah penyaringan, terkumpul 213 data.

Total data yang dikumpulkan dari empat sumber berbeda berjumlah 4.506, dikumpulkan dalam waktu 1 bulan mulai 4 Mei 2024 hingga 5 Juni 2024 yang dapat dilihat pada Gambar 2 berikut.



Gambar 2. Cuplikan Hasil Pengumpulan Dataset

4.3. *Exploratory Data Analysis* (EDA)

Pada tahap EDA, program diinisiasi di platform Google Colaboratory. Langkah pertama dalam Machine Learning menggunakan Python adalah memahami dan mengeksplorasi data dengan library yang diperlukan, seperti:

```
# Import needed modules
import numpy as np
import pandas as pd
import nltk
import re
import difflib
import time
import psutil
from sklearn.feature_extraction.text import
TfidfVectorizer
from sklearn.feature_extraction.text import
CountVectorizer
from sklearn.preprocessing import normalize
from sklearn.metrics.pairwise import
cosine_similarity
from sklearn.metrics import jaccard_score
from scipy.spatial.distance import pdist,
squareform
from sklearn.cluster import KMeans
```

Library Pandas digunakan untuk memuat data ke dalam DataFrame dari berbagai format file. Dengan read_csv(), data dapat dikonversi menjadi DataFrame:

```
# Read data
df =
pd.read_csv('/content/Indonesia_Event_Service_Bus
inesses_Indo.csv')
```

Untuk memahami data, perlu memeriksa jumlah baris dan kolom, tipe data, dan nilai yang hilang dengan fungsi info():

```
df.info()
```

Penting juga untuk mengetahui kata apa yang paling sering muncul dengan menggunakan word cloud. Word cloud adalah objek representasi visual yang sederhana namun kuat untuk pemrosesan teks. Word cloud digunakan untuk mengetahui kata yang paling sering muncul dalam kolom deskripsi:

```
import matplotlib.pyplot as plt
from wordcloud import WordCloud

text = open("output.txt", mode="r",
encoding="utf-8").read()
word_cloud = WordCloud(collocations=False,
background_color='white', height=700,
width=1200).generate(text)
plt.imshow(word_cloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

Gambar 3. Word Cloud EDA

Setelah itu, dilakukan pengecekan duplikat dengan nunique() dan pengecekan missing values dengan isnull().sum():

```
df.nunique()
df.isnull().sum()
```

Proses Data Reduction dilakukan dengan menghapus kolom yang tidak relevan:

```
df = df.drop(['no', 'address', 'contact', 'url',
'lat', 'lng'], axis=1)
df.info()
```

Feature Engineering dilakukan untuk menambahkan kolom baru yang relevan untuk analisis:

```
selected_features = ['kategori', 'subkategori',
'location/city', 'nama', 'deskripsi']
print(selected_features)
```

Kolom yang dipilih ini akan digunakan dalam proses Pre-Processing Data untuk membuat model prediksi machine learning.

4.4. *Pre-Processing* Data

Setelah data dieksplor dan dianalisis kemudian dilakukan proses yang sering disebut data preprocessing. Tahap data preprocessing berfungsi untuk menghilangkan noise pada data, karena sebelum diklasifikasikan data harus benar-benar bersih dan terstruktur.

4.4.1. Data Cleaning

Tahap ini mencakup pembersihan data untuk mengatasi data duplikat, data null, integrasi data, dan penghilangan karakter khusus. Langkah pertama adalah menghapus data

duplikat dengan menggabungkan kolom kategori dan nama menjadi 'kategori_nama' dan menghapus baris duplikat berdasarkan kolom tersebut.

```
kategori_nama = df['kategori'] + ' ' + df['nama']
df = df.assign(kategori_nama=kategori_nama)
df.drop_duplicates(subset=['kategori_nama'],
keep="first", inplace=True)
df.reset_index(inplace=True)
df = df.drop(['index', 'kategori_nama'], axis=1)
```

Kemudian, nilai-nilai null diganti dengan string "null" untuk menangani missing values

```
df.fillna('', inplace=True)

def cleaning(Text):
    return re.sub('@\S+|https?:\S+|http?:\S|[^A-
Za-z0-9,.]+',' ', Text)
df['deskripsi'] = df['deskripsi'].apply(cleaning)
```

Selain itu, karakter spesial seperti :, $, @, / juga dihapus untuk memastikan teks bersih. Terakhir, dilakukan penggabungan beberapa kolom penting menjadi satu kolom 'combined_features' untuk mempermudah analisis selanjutnya.

```
df.to_csv('dataset.csv', sep=',', index=False)
combined_features = df['kategori'] + ' ' +
df['subkategori'] + ' ' + df['location/city'] + '
' + df['nama'] + ' ' + df['deskripsi']
df =
df.assign(combined_features=combined_features)
```

4.4.2. Case Folding

Pada tahap ini setiap kata pada data yang telah dikumpulkan diseragamkan menjadi huruf kecil semua menggunakan fungsi lower(). Karena tidak semua dokumen teks konsisten menggunakan huruf kapital atau huruf kecil.

```
df['case_folding'] =
df['combined_features'].str.lower()
```

4.4.3. Tokenization

Setelah data diseragamkan menjadi huruf kecil semua, maka selanjutnya dilakukan tokenization yang bertujuan untuk merepresentasikan teks dengan cara yang bermakna bagi mesin tanpa kehilangan konteksnya. Dengan mengubah teks menjadi token, algoritma dapat lebih mudah mengidentifikasi pola.

```
def tokenization(text):
    tokens = re.split('\W+', text)
    return tokens
df['tokenization'] =
df['case_folding'].apply(tokenization)
```

4.4.4. Stopword Removal

Setelah data ditokenisasi, maka selanjutnya dilakukan stopword removal yang bertujuan untuk menyaring kata-kata yang sering muncul dan tidak memiliki arti yang spesifik.

```
nltk.download('stopwords')
from nltk.corpus import stopwords
list_stopwords =
set(stopwords.words('indonesian'))

def stopwords_removal(Text):
    words = Text.split()
    return [word for word in words if word not in
list_stopwords]
df['stopword_removal'] =
df['case_folding'].apply(stopwords_removal)
```

Dalam source code tersebut digunakan kamus stopwords berbahasa Indonesia yang tersedia dalam library NLTK.

4.4.5. Stemming

Setelah proses menghilangkan stopwords, maka selanjutnya dilakukan proses stemming. Pada tahap ini akan mengubah kata yang memiliki imbuhan baik yang terdiri dari awalan, sisipan, akhiran, dan kombinasi dari awalan dan akhiran menjadi kata dasar dengan stemmer Bahasa Indonesia dari sastrawi.

```
!pip install swifter
!pip install Sastrawi
from Sastrawi.Stemmer.StemmerFactory import
StemmerFactory
import swifter

factory = StemmerFactory()
stemmer = factory.create_stemmer()

def apply_stemmed_term(Text):
    return [stemmer.stem(term) for term in Text]
df['stemming'] =
df['stopword_removal'].swifter.apply(apply_stemme
d_term)
```

Setelah proses stemming selesai, maka data sudah siap untuk dilanjutkan ke tahap pembangunan Sistem Rekomendasi Berbasis Konten.

4.5. Pemodelan Sistem Rekomendasi Berbasis Konten

Langkah pertama yang harus dilakukan adalah pembobotan. Data yang sudah bersih disimpan untuk digunakan selama masa

percobaan model. Pembobotan dilakukan dengan empat teknik term-weighting yakni, TF-IDF, TF-RF, TF-ABS, dan TF-PDF.

4.5.1. Pembobotan Kata

Dalam tahap pembobotan, terdapat empat teknik yang dilakukan, antara lain:

a. TF-IDF: Mengukur pentingnya kata dalam dokumen dengan mempertimbangkan frekuensi kemunculan kata (TF) dan kebalikannya di seluruh koleksi dokumen (IDF).

```
vectorizer_tfidf = TfidfVectorizer()
tfidf_matrix =
vectorizer_tfidf.fit_transform(dfclean['stemming'
])
print(tfidf_matrix)
```

b. TF-RF: Mengalikan frekuensi term (TF) dengan frekuensi relevansi (RF).

```
vectorizer_tfrf = CountVectorizer()
tf_matrix =
vectorizer_tfrf.fit_transform(dfclean['stemming']
)
tf_array = tf_matrix.toarray()
doc_freq = np.sum(tf_array > 0, axis=0)
N = tf_array.shape[0]
rf = np.log((N / (N - doc_freq + 1)))
tfrf = tf_array * rf
tfrf_matrix = csr_matrix(tfrf)
print(tfrf_matrix)
```

c. TF-ABS: Menggunakan frekuensi absolut dari term dalam dokumen untuk menghitung bobotnya.

```
vectorizer_tfabs = CountVectorizer()
term_freq_matrix =
vectorizer_tfabs.fit_transform(dfclean['stemming'
])
tf_matrix = normalize(term_freq_matrix,
norm='l1', axis=1)
abs_freq =
np.asarray(term_freq_matrix.sum(axis=0)).flatten(
)
tf_abs_matrix = tf_matrix.multiply(abs_freq)
tfabs_matrix = csr_matrix(tf_abs_matrix)
print(tfabs_matrix)
```

d. TF-PDF: Mengalikan frekuensi term (TF) dengan fungsi kerapatan probabilitas (PDF).

```
vectorizer_tfpdf = CountVectorizer()
term_freq_matrix =
vectorizer_tfpdf.fit_transform(dfclean['stemming'
])
tf_matrix = normalize(term_freq_matrix,
norm='l1', axis=1)
doc_frequency = (term_freq_matrix >
0).sum(axis=0)
doc_frequency =
np.asarray(doc_frequency).flatten()
N = term_freq_matrix.shape[0]
pdf = np.log((N - doc_frequency + 0.5) /
(doc_frequency + 0.5))
tf_pdf_matrix = tf_matrix.multiply(pdf)
tfpdf_matrix = csr_matrix(tf_pdf_matrix)
print(tfpdf_matrix)
```

4.5.2. Skor Similaritas

Setelah data dibobot, maka selanjutnya dapat dilakukan perhitungan nilai kesamaannya dengan data konten lainnya. Perhitungan tersebut dilakukan dengan menggunakan Cosine Similarity dan Jaccard Similarity.

a. Cosine Similarity: Menghitung kesamaan kosinus antara vektor hasil pembobotan.

```
tfidf_cosine = cosine_similarity(tfidf_matrix,
tfidf_matrix)
print(tfidf_cosine)
tfrf_cosine = cosine_similarity(tfrf_matrix,
tfrf_matrix)
print(tfrf_cosine)
tfabs_cosine = cosine_similarity(tfabs_matrix,
tfabs_matrix)
print(tfabs_cosine)
tfpdf_cosine = cosine_similarity(tfpdf_matrix,
tfpdf_matrix)
print(tfpdf_cosine)
```

b. Jaccard Similarity: Menghitung kesamaan Jaccard antara dua set item.

```
tfidf_binary = (tfidf_matrix > 0).astype(int)
jaccard_distances = pdist(tfidf_binary.toarray(),
metric='jaccard')
tfidf_jaccard = 1 - squareform(jaccard_distances)
np.fill_diagonal(tfidf_jaccard, 1.0)
print(tfidf_jaccard)
```

4.5.3. Sistem Rekomendasi Berbasis Konten (CBRS)

Setelah scores similarity telah dihitung oleh masing-masing metode dan skenario atau teknik pembobotan, maka selanjutnya dibuatlah sistem rekomendasi berbasis konten.

```
def get_recommendations(index,
similarity_matrix):
    sim_services =
list(enumerate(similarity_matrix[index]))
    sim_services = sorted(sim_services,
key=lambda x: x[1], reverse=True)
```

```
    sim_services = sim_services[1:11]
    top_indices = [i[0] for i in sim_services]
    return df.iloc[top_indices][['kategori',
'subkategori', 'location/city', 'nama',
'deskripsi']]

scenarios = {
    'TF-IDF & Cosine': tfidf_cosine,
    'TF-RF & Cosine': tfrf_cosine,
    'TF-ABS & Cosine': tfabs_cosine,
    'TF-PDF & Cosine': tfpdf_cosine,
    'TF-IDF & Jaccard': tfidf_jaccard
}

index_input = 744
recommendations = {}

for name, sim_matrix in scenarios.items():
    recommendations[name] =
get_recommendations(index_input, sim_matrix)
    print(f"Scenario: {name}")
    print(f"Layanan yang dipilih:
{df.iloc[index_input]['nama']}")
    print("10 Rekomendasi:")
    print(recommendations[name])
    print("\n")
```

4.5.4. Evaluasi Sistem

Pada tahap ini, kinerja sistem rekomendasi dievaluasi menggunakan metrik Diversity. Diversity mengukur keberagaman rekomendasi yang diberikan kepada pengguna. Cara kerja metrik ini melibatkan pengukuran variasi antara item-item yang direkomendasikan, sering kali menggunakan metrik seperti jaccard index, dissimilarity measures, atau distance metrics antara fitur-fitur item. Proses ini dilakukan pada setiap skenario untuk mencari tahu kinerja terbaik untuk diterapkan pada aplikasi Eventhings.

```
def calculate_diversity(recommendation_indices,
similarity_matrix):
    n = len(recommendation_indices)
    if n <= 1:
        return 0
    dissimilarity_sum = 0
    for i in range(n):
        for j in range(i + 1, n):
            dissimilarity_sum += 1 -
similarity_matrix[recommendation_indices[i],
recommendation_indices[j]]
    diversity_score = dissimilarity_sum / (n / 2
* (n - 1))
    return diversity_score

# Menghitung nilai keragaman untuk semua skenario
diversity_values = {}
for name, sim_matrix in scenarios.items():
    recommended_items_list =
[get_recommendations(index, sim_matrix) for index
in range(len(dfclean))]
    diversity_values[name] =
[calculate_diversity(recommendations, sim_matrix)
for recommendations in recommended_items_list]
    average_diversity =
np.mean(diversity_values[name])
    print(f"Average Diversity value for {name}:
{average_diversity:.2f}")
```

Dari source code di atas yang menghitung nilai keragaman dari setiap skenario yang diterapkan, menghasilkan nilai yang berbeda-beda, seperti yang dapat dilihat dari tabel berikut ini.

Tabel 1. Hasil Perhitungan Nilai Keragaman

Skenario	Nilai Average Diversity Value
TF-IDF & Cosine	0.60
TF-RF & Cosine	0.03
TF-ABS & Cosine	0.04
TF-PDF & Cosine	0.66
TF-IDF & Jaccard	0.49

Berdasarkan penelitian sebelumnya [15][16], dapat disimpulkan bahwa skenario yang memiliki nilai yang bagus, baik untuk diversity maupun untuk relevancy, adalah rentang 50%-80% yaitu skenario 1 (60%) dan skenario 4 (66%). Hal tersebut dikarenakan nilai 50% menjadi titik tengah antara keragaman dan kemiripan sebuah konten. Sedangkan 80% menjadi titik tengah antara keragaman dan kerelevanan sebuah konten.

Setelah diketahui nilai keragaman, evaluasi dilanjutkan dengan menghitung CPU Usage dan Processing Time untuk mengetahui skenario terbaik yang memiliki diversity value yang bagus pada evaluasi metrik sebelumnya yakni skenario 1 dan skenario 4. Sebagai berikut, source code yang digunakan untuk menghitung CPU Usage dan Processing Time.

```
start_time = time.time()
cpu_percent_before = psutil.cpu_percent()

vectorizer_tf = TfVectorizer()
tf_matrix =
vectorizer_tf.fit_transform(dfclean['stemming'])
tf_cosine = cosine_similarity(tf_matrix,
tf_matrix)
top_10_recommendations =
df.iloc[get_recommendations(index_input,
```

```
tf_cosine)][['kategori', 'subkategori',
'location/city', 'nama', 'deskripsi']]

end_time = time.time()
cpu_percent_after = psutil.cpu_percent()
processing_time = end_time - start_time
cpu_usage = cpu_percent_after -
cpu_percent_before

print(f"CPU Usage Skenario: {cpu_usage}%")
print(f"Processing Time Skenario:
{processing_time} detik")
```

Dari source code di atas yang menghitung CPU Usage dan Processing Time dari skenario 1 dan skenario 4, mengeluarkan hasil yang berbeda, seperti yang dapat dilihat dari tabel berikut ini.

Tabel 2. Hasil Perhitungan CPU Usage dan Processing Time

Skenario	CPU Usage dan
TF-IDF & Cosine	23% dan 1.866028 milliseconds
TF-PDF & Cosine	21% dan 0.447288 milliseconds

Berdasarkan Tabel 2, dapat dilihat bahwa skenario 4 memiliki processing time yang lebih cepat dibandingkan dengan skenario 1. Selain itu, penggunaan CPU pada skenario 4 lebih rendah dibandingkan dengan skenario 1. Sehingga dapat disimpulkan bahwa Skenario 4 yakni teknik pembobotan *Term Frequency-Probability Density Function* (TF-PDF) dengan metode *Cosine Similarity* memiliki kinerja yang paling baik di antara seluruh skenario percobaan.

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dengan membandingkan kinerja metode cosine dan jaccard similarity dalam pembangunan Content - Based Recommendation Systems (CBRS) dapat disimpulkan bahwa skenario 4 dengan teknik pembobotan TF-PDF dan metode Cosine Similarity memiliki kinerja yang paling baik, dengan nilai diversity yang cukup tinggi (66%). Selain itu, evaluasi CPU usage (21%) dan processing time (0.45 ms) juga menunjukkan bahwa skenario ini memiliki performa yang paling optimal. Hal itu disebabkan oleh pembobotan yang dilakukan sebelum melakukan perhitungan similarity score serta kecocokannya dalam metode perhitungan cosine similarity dibandingkan dengan jaccard similarity.

5.2. Saran

Untuk penelitian selanjutnya, beberapa saran dapat dipertimbangkan. Pertama, adalah eksplorasi lebih lanjut terhadap preferensi user untuk memastikan bahwa sistem rekomendasi yang diberikan relevan atau sesuai dengan yang user inginkan. Penelitian ini dapat memberikan wawasan baru dan memperluas cakupan pemahaman dalam pembangunan Content-Based Recommendation Systems (CBRS). Selanjutnya, penggunaan metrik evaluasi tambahan seperti Precision, Recall, atau F1-Score juga perlu dipertimbangkan. Integrasi metrik evaluasi ini akan memberikan pemahaman yang lebih holistik tentang kinerja sistem rekomendasi, memungkinkan evaluasi yang lebih komprehensif.

Kemudian, perlu dilakukan optimasi kinerja lebih lanjut pada implementasi sistem, baik dari segi algoritma maupun infrastruktur, untuk memastikan penggunaan sumber daya yang efisien dan responsif. Dengan mempertimbangkan saran-saran ini, diharapkan penelitian selanjutnya dapat meningkatkan pemahaman dan kinerja dari sistem rekomendasi berbasis konten yang dikembangkan.

UCAPAN TERIMA KASIH

Dengan mengucap puji syukur kehadirat Allah SWT atas rahmat dan hidayah-Nya, penulis dapat menyelesaikan skripsi berjudul "Komparasi Kinerja Metode Cosine dan Jaccard Similarity dalam Content-Based Recommendation Systems (CBRS) pada Aplikasi Eventhings". Skripsi ini disusun sebagai syarat memperoleh gelar sarjana pada Program Studi Sistem Informasi Universitas Pembangunan Nasional “Veteran” Jawa Timur. Dalam penyusunan skripsi ini, penulis menerima banyak dukungan dan bantuan. Terima kasih kepada seluruh pihak yang telah memberikan bantuan dan dukungan dalam penyelesaian skripsi ini. Penulis memohon maaf atas segala kekurangan dalam laporan ini

dan berharap skripsi ini bermanfaat bagi pembaca dan penulis.

DAFTAR PUSTAKA

[1] Ghifary, R. A. (2018). Analisis kualitas layanan pada perusahaan e-commerce traveloka.

[2] Hadiono, K., & Santi, R. C. N. (2020). Menyongsong Transformasi Digital. Proceeding SENDI_U, 81–84.

[3] Hidayatullah, S., Waris, A., & Devianti, R. C. (2018). Perilaku Generasi Milenial dalam Menggunakan Aplikasi Go-Food. *Jurnal Manajemen Dan Kewirausahaan*, *6*(2). https://doi.org/10.26905/jmdk.v6i2.2560

[4] Mediana. (2023). *Ada 3.000 Acara Tahun Ini, POTENSI Ekonominya RP 162 Triliun*. kompas.id. https://www.kompas.id/baca/ekonomi/2023/05/22/3000-event-hadir-di-indonesia-sepanjang-2023

[5] Lukitaningtyas, I., Andreswari, R., & Al Anshary, F. M. (2018). Rancang Bangun E-marketplace" dyland" Bagi Penyedia Jasa Event Organizer-Party Planner Menggunakan Metode Iterative Incremental (modul Transaksi) Studi Kasus Event Organizer Kota Bandung. eProceedings of Engineering, 5(3).

[6] Matthes, E. (2019). Python Crash Course, 2nd Edition: A Hands-On, Project-Based Introduction to Programming. United States: No Starch Press.

[7] Jarmul, K., Lawson, R. (2017). *Python Web Scraping*. United Kingdom: Packt Publishing.

[8] Laraswati, B. D. (2022, August 15). *Web Scraping vs Web Crawling, Apa Bedanya?* Algoritma Data Science School. https://blog.algorit.ma/web-scraping/

[9] Jo, T. (2018). *Text Mining: Concepts, Implementation, and Big Data Challenge*. Germany: Springer International Publishing.

[10] García, S., Luengo, J., Herrera, F. (2014). *Data Preprocessing in Data Mining*. Germany: Springer International Publishing.

[11] Brusilovsky, P., Kobsa, A. (2007). *The Adaptive Web: Methods and Strategies of Web Personalization*. Germany: Springer.

[12] Han, J., Kamber, M., Pei, J. (2011). Data Mining: Concepts and Techniques. Netherlands: Elsevier Science.

[13] Zuur, A., Ieno, E. N., Smith, G. M. (2007). Analyzing Ecological Data. Germany: Springer New York.

[14] Harnelia, H. (2024). ANALISIS SENTIMEN REVIEW SKINCARE SKINTIFIC DENGAN ALGORITMA SUPPORT VECTOR MACHINE (SVM). Jurnal Informatika Dan Teknik Elektro Terapan, 12(2). https://doi.org/10.23960/jitet.v12i2.4095.

[15] Kunaver, M., & Požrl, T. (2017). Diversity in recommender systems – A survey. Knowledge-based Systems, 123, 154–162. https://doi.org/10.1016/j.knosys.2017.02.009.

[16] Wu, W., Chen, L., & Zhao, Y. (2018). Personalizing recommendation diversity based on user personality. *User Modeling and User-adapted Interaction*, 28(3), 237–276. https://doi.org/10.1007/s11257-018-9205-x