1. Dataset

Pada praktik UAS ini, kami menggunakan dataset mengenai books. Dataset yang kami bernama Goodreads yang bersumber dari www.kaggle.com. Dataset kami olah dengan metode Content Based Filtering (CBF).

Tujuan analisis dataset Goodreads adalah untuk mendapatkan gambaran tentang hubungan antara berbagai atribut yang mungkin dimiliki sebuah buku, seperti: peringkat agregat setiap buku, tren penulis selama bertahun-tahun, dan buku dengan banyak bahasa. Dengan lebih dari seratus ribu peringkat, ada buku-buku yang cenderung menjadi populer setiap hari.

Dengan metode CBF, diharapkan dapat melihat jenis buku apa yang benar-benar direkomendasikan dan mendorong orang untuk membaca di era modern ini
Dataset ini memiliki 12 kolom dan 11123 data, yaitu:

a. bookID → ID unik untuk setiap buku/seri
b. title → judul-judul buku
c. authors → penulis buku tertentu
d. average_rating → peringkat rata-rata buku, seperti yang diputuskan oleh pengguna
e. ISBN → informasi tentang buku - seperti edisi dan penerbit
f. ISBN 13 Format baru ISBN → diterapkan pada tahun 2007
g. language_code → bahasa untuk buku
h. num_pages → jumlah halaman buku
i. ratings_count → jumlah rating yang diberikan untuk buku tersebut
j. text_reviews_count → jumlah ulasan yang diberikan oleh pengguna
k. publication_date → tanggal di publish
l. publisher → pihak mana yang mem-publish

2. Coding

a. Import library

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
import warnings
warnings.filterwarnings('ignore')
import re
import string
from sklearn.metrics.pairwise import cosine_similarity
```

```
from sklearn.feature_extraction.text import CountVectorizer,
TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
```

- import pandas as pd: Mengimpor modul Pandas dan memberikan alias pd. Pandas adalah modul yang digunakan untuk analisis dan manipulasi data tabular.
- import numpy as np: Mengimpor modul NumPy dan memberikan alias np untuk digunakan dalam kode. NumPy adalah modul yang digunakan untuk komputasi numerik dalam Python
- import warnings: Mengimpor modul warnings dalam Python
- warnings.filterwarnings("ignore"): Mengatur perilaku modul warnings dengan mengabaikan peringatan yang muncul selama eksekusi program
- from sklearn.cluster import KMeans: Mengimpor kelas KMeans dari mod

b. Memuat dataset

```
df = pd.read_csv('books.csv', error_bad_lines = False)
df.head()
```

Output:

```
Skipping line 3350: expected 12 fields, saw 13
Skipping line 4704: expected 12 fields, saw 13
Skipping line 5879: expected 12 fields, saw 13
Skipping line 8981: expected 12 fields, saw 13
```

	bookID	title	authors	average_rating	isbn	isbn13	language_code	num_pages	ratings_count	text_reviews_count	publication_date	publisher
0	1	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPré	4.57	0439785960	9780439785969	eng	652	2095690	27591	9/16/2006	Scholastic Inc.
1	2	Harry Potter and the Order of the Phoenix (Har...	J.K. Rowling/Mary GrandPré	4.49	0439358078	9780439358071	eng	870	2153167	29221	9/1/2004	Scholastic Inc.
2	4	Harry Potter and the Chamber of Secrets (Harry...	J.K. Rowling	4.42	0439554896	9780439554893	eng	352	6333	244	11/1/2003	Scholastic
3	5	Harry Potter and the Prisoner of Azkaban (Harr...	J.K. Rowling/Mary GrandPré	4.56	043965548X	9780439655484	eng	435	2339585	36325	5/1/2004	Scholastic Inc.
4	8	Harry Potter Boxed Set Books 1-5 (Harry Potte...	J.K. Rowling/Mary GrandPré	4.78	0439682584	9780439682589	eng	2690	41428	164	9/13/2004	Scholastic

c. Data Prepocessing (Missing Value)
Mencari missing values dengan code:

```
missing_values = df.isnull().sum()
print(missing_values)
```

Menggunakan `df.isnull()`untuk membuat DataFrame baru dengan ukuran yang sama seperti `df`, di mana setiap elemen akan bernilai `True` jika elemen yang

sesuai di DataFrame asli `df` adalah nilai kosong (missing value), dan `False` jika tidak. `sum()`digunakan untuk menghitung jumlah nilai `True` dalam setiap kolom.

Output:

```
bookID               0
title                0
authors              0
average_rating       0
isbn                 0
isbn13               0
language_code        0
  num_pages          0
ratings_count        0
text_reviews_count   0
publication_date     0
publisher            0
dtype: int64
```

Tidak ditemukan missing value.

d. Pengambilan kolom

```
content_data = df[['title','authors','average_rating']]
content_data = content_data.astype(str)

content_data['content'] = content_data['title'] + ' ' +
content_data['authors'] + ' ' +
content_data['average_rating']
```

`content_data = df[['title','authors','average_rating']]` digunakan untuk mengambil kolom-kolom yang diinginkan dari dataframe `df` dan menyimpannya dalam variabel `content_data`. `content_data = content_data.astype(str)` mengubah tipe data dari kolom-kolom yang terpilih menjadi string. Fungsi `astype()` digunakan untuk mengubah tipe data kolom menjadi tipe data yang diinginkan, dalam hal ini menjadi tipe data string (`str`). `content_data['content'] = content_data['title'] + ' ' + content_data['authors'] + ' ' + content_data['average_rating']` digunakan untuk membuat kolom baru dengan nama 'content' di dalam dataframe `content_data`.

e. Menghubungkan buku dengan indeks numerik

```
content_data = content_data.reset_index()
indices = pd.Series(content_data.index,
index=content_data['title'])
```

content_data = content_data.reset_index()`, digunakan untuk me-reset indeks dari dataframe `content_data`. Operasi `reset_index()` digunakan untuk mengatur ulang indeks dataframe dengan indeks yang baru. Dalam hal ini, indeks asli di-reset menjadi indeks numerik default. `indices = pd.Series(content_data.index, index=content_data['title'])`, digunakan untuk membuat objek `pd.Series` yang disebut `indices`. Seri ini memiliki indeks yang terkait dengan kolom 'title' dalam dataframe `content_data` dan nilai yang sesuai adalah indeks numerik dari dataframe.

f. Inisiasi dan Membuat TF IDF

```
#removing stopwords
tfidf = TfidfVectorizer(stop_words='english')

#Construct the required TF-IDF matrix by fitting and
transforming the data
tfidf_matrix = tfidf.fit_transform(content_data['authors'])

#Output the shape of tfidf_matrix
tfidf_matrix.shape
```

Output:

```
(11123, 8448)
```

Dilihat dari Output, maka terdiri dari 11123 baris dan 8448 kolom

g. Menghitung kemiripan kosinus antara matriks TF-IDF tfidf_matrix dan dirinya sendiri menggunakan fungsi linear kernel

```
cosine_sim_author = linear_kernel(tfidf_matrix, tfidf_matrix)
```

Fungsi linear_kernel → sebuah fungsi utilitas dari library scikit-learn yang menghitung kernel linear antara dua matriks. Dalam kasus ini, fungsi tersebut digunakan untuk menghitung kemiripan kosinus antara tfidf_matrix dengan dirinya sendiri.

Perhitungan kemiripan kosinus:
Fungsi linear_kernel menggunakan tfidf_matrix sebagai kedua matriks input. Ia menghitung kemiripan kosinus pasangan antara setiap baris dari tfidf_matrix dengan setiap baris lainnya, menghasilkan sebuah matriks simetri. Variabel cosine_sim_author diberikan nilai matriks kemiripan kosinus yang dihasilkan.

h. Get recommendations books

```
def get_recommendations_books(title,
cosine_sim=cosine_sim_author):
    idx = indices[title]

    # Get the pairwsie similarity scores of all books with
that book
    sim_scores = list(enumerate(cosine_sim_author[idx]))

    # Sort the books based on the similarity scores
    sim_scores = sorted(sim_scores, key=lambda x: x[1],
reverse=True)

    # Get the scores of the 10 most similar books
    sim_scores = sim_scores[1:11]

    # Get the book indices
    book_indices = [i[0] for i in sim_scores]

    # Return the top 10 most similar books
    return list(content_data['title'].iloc[book_indices])
```

Fungsi `get_recommendations_books` mengambil judul buku sebagai argumen dan menghasilkan daftar judul buku yang paling mirip dengan buku yang diberikan berdasarkan skor kemiripan kosinus.

- `idx = indices[title]`: Mengambil indeks buku yang terkait dengan judul buku yang diberikan dari objek `indices`. Ini akan digunakan untuk mengambil skor kemiripan kosinus yang sesuai.
- `sim_scores = list(enumerate(cosine_sim_author[idx]))`: Membuat daftar pasangan indeks dan skor kemiripan kosinus antara buku yang diberikan dengan semua buku lainnya. `enumerate()` digunakan untuk menghasilkan

pasangan indeks dan nilai skor kemiripan kosinus dari matriks `cosine_sim_author`.

- `sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)`: Mengurutkan daftar skor kemiripan kosinus dalam urutan menurun. Fungsi `sorted()` digunakan dengan argumen `key=lambda x: x[1]` untuk mengurutkan daftar berdasarkan nilai skor (indeks 1 dalam setiap pasangan) dalam urutan menurun.
- `sim_scores = sim_scores[1:11]`: Mengambil sepuluh skor kemiripan teratas dari daftar skor yang diurutkan. Ini akan menghilangkan skor kemiripan buku itu sendiri yang memiliki skor 1.
- `book_indices = [i[0] for i in sim_scores]`: Membuat daftar indeks buku yang paling mirip berdasarkan skor kemiripan teratas.
- `return list(content_data['title'].iloc[book_indices])`: Mengembalikan daftar judul buku yang paling mirip dengan buku yang diberikan. Daftar ini diambil dari kolom 'title' dalam dataframe `content_data` menggunakan indeks buku yang telah diambil sebelumnya.

i. Mencetak daftar buku secara terpisah

```
def author_book_shows(book):
    for book in book:
        print(book)
```

`for book in book:`: Melakukan iterasi pada setiap elemen dalam daftar `book`. Dalam setiap iterasi, nilai `book` akan mewakili satu buku dalam daftar. `print(book)`: Mencetak nilai `book`. Fungsi `print()` digunakan untuk mencetak atau menampilkan nilai `book` ke output konsol.

j. Rekomendasi buku berdasarkan kemiripan penulis

```
books1 = get_recommendations_books('The Hobbit',
cosine_sim_author)
author_book_shows(books1)
```

Fungsi get_recommendations_books() → Syntax ini memanggil fungsi get_recommendations_books() dengan parameter pertama 'Shadow Kiss', yang merupakan judul buku yang ingin Anda cari rekomendasinya. Parameter kedua, cosine_sim_author, adalah matriks kemiripan kosinus antara penulis buku berdasarkan representasi TF-IDF mereka. Fungsi ini

akan menghasilkan daftar rekomendasi buku yang paling mirip dengan 'Shadow Kiss' berdasarkan kemiripan penulisnya.

Fungsi author_book_shows() → Syntax ini memanggil fungsi author_book_shows() dengan parameter books2. Fungsi ini bertujuan untuk menampilkan informasi tentang penulis dan judul buku dari daftar rekomendasi yang diberikan sebagai argumen. Namun, karena saya tidak memiliki informasi tentang implementasi dari fungsi-fungsi tersebut, saya tidak dapat memberikan penjelasan lebih lanjut mengenai fungsionalitas dan output yang dihasilkan.

Output :

```
The Lord of the Rings (The Lord of the Rings  #1-3)
The Fellowship of the Ring (The Lord of the Rings  #1)
The Lord of the Rings- 3 volumes set (The Lord of the Rings  #1-3)
The Lord of the Rings (The Lord of the Rings  #1-3)
The Hobbit  or There and Back Again
Poems From The Hobbit
The Hobbit: Or There and Back Again
The Hobbit
Tree and Leaf: Includes Mythopoeia and The Homecoming of Beorhtnoth
The Return of the King (The Lord of the Rings  #3)
```

k. Rekomendasi buku berdasarkan kemiripan penulis kedua

```
books2 =get_recommendations_books('The Lost Continent: Travels in Small Town America', cosine_sim_author)

author_book_shows(books2)
```

Fungsi get_recommendations_books() → Syntax ini memanggil fungsi get_recommendations_books() dengan parameter pertama 'Shadow Kiss', yang merupakan judul buku yang ingin Anda cari rekomendasinya. Parameter kedua, cosine_sim_author, adalah matriks kemiripan kosinus antara penulis buku berdasarkan representasi TF-IDF mereka. Fungsi ini akan menghasilkan daftar rekomendasi buku yang paling mirip dengan 'Shadow Kiss' berdasarkan kemiripan penulisnya.

Fungsi author_book_shows() → Syntax ini memanggil fungsi author_book_shows() dengan parameter books2. Fungsi ini bertujuan untuk menampilkan informasi tentang penulis dan judul buku dari daftar rekomendasi yang diberikan sebagai argumen. Namun, karena saya tidak

memiliki informasi tentang implementasi dari fungsi-fungsi tersebut, saya tidak dapat memberikan penjelasan lebih lanjut mengenai fungsionalitas dan output yang dihasilkan.

Output :

```
Bill Bryson's African Diary
Bryson's Dictionary of Troublesome Words: A Writer's Guide to Getting It Right
In a Sunburned Country
I'm a Stranger Here Myself: Notes on Returning to America After Twenty Years Away
The Lost Continent: Travels in Small Town America
Neither Here nor There: Travels in Europe
Notes from a Small Island
The Mother Tongue: English and How It Got That Way
A Short History of Nearly Everything
A Short History of Nearly Everything (Illustrated Edition)
```

3. Evaluasi

- Rekomendasi pertama

Title	authors	average_rating
The Hobbit	**J.R.R. Tolkien**	**427**
The Lord of the Rings (The Lord of the Rings #1-3)	J.R.R. Tolkien/Rob Inglis	450
The Fellowship of the Ring (The Lord of the Rings #1)	J.R.R. Tolkien	436
The Lord of the Rings- 3 volumes set (The Lord of the Rings #1-3)	J.R.R. Tolkien	450
The Lord of the Rings (The Lord of the Rings #1-3)	J.R.R. Tolkien/Rob Inglis	450
The Hobbit or There and Back Again	J.R.R. Tolkien/Alan Lee	427
Poems From The Hobbit	J.R.R. Tolkien	430

The Hobbit: Or There and Back Again	J.R.R. Tolkien	427
The Hobbit	J.R.R. Tolkien	427
Tree and Leaf: Includes Mythopoeia and The Homecoming of Beorhtnoth	J.R.R. Tolkien	405
The Return of the King (The Lord of the Rings #3)	J.R.R. Tolkien	453

Didapatkan kesimpulan bahwa metode yang telah *Content-based Filtering* dengan menggunakan semua fitur untuk mendapatkan rekomendasi, walaupun hanya memiliki satu kesamaan dalam satu fitur, dia tetap masuk ke dalam metode CBF. Penjelasannya sebagai berikut:

(1) "*The Lord of the Rings (The Lord of the Rings #1-3)*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity.

(2) "*The Fellowship of the Ring (The Lord of the Rings #1)*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity.

(3) "*The Lord of the Rings -3 volumes set (The Lord of the Rings #1-3)*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity.

(4) "*The Lord of the Rings (The Lord of the Rings #1-3)*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity.

(5) "*The Hobbit or There and Back Again*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity yaitu 400 sekian.

(6) "*Poems From The Hobbit*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity yaitu 400 sekian.

(7) "*The Hobbit: Or There and Back Again*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity yaitu 400 sekian.

(8) "*The Hobbit*"memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity yaitu 400 sekian.

(9) "*Tree and Leaf: Includes Mythopeia and The Homecoming of Beorhtnoth*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity yaitu 400 sekian.

(10) "*The Return of the King (The Lord of the Rings #3)*" memiliki kesamaan pada author dan pada title sesuai dengan tema the hobbit dan pada average_rating memiliki similarity yaitu 400 sekian.

$$\text{Akurasi} = \frac{Total\ Data(n) - Data\ Missing}{Total\ Data} \times 100\% = \frac{10-0}{10} \times 100\% = 100\%$$

- Rekomendasi kedua

Title	authors	average_rating
The Lost Continent: Travels in Small Town America	**Bill Bryson**	**383**
Bill Bryson's African Diary	Bill Bryson	344
Bryson's Dictionary of Troublesome Words: A Writer's Guide to Getting It Right	Bill Bryson	387
In a Sunburned Country	Bill Bryson	407
I'm a Stranger Here Myself: Notes on Returning to America After Twenty Years Away	Bill Bryson	390
The Lost Continent: Travels in Small Town America	Bill Bryson	383
Neither Here nor There: Travels in Europe	Bill Bryson	386

Notes from a Small Island	Bill Bryson	391
The Mother Tongue: English and How It Got That Way	Bill Bryson	393
A Short History of Nearly Everything	Bill Bryson	421
A Short History of Nearly Everything (Illustrated Edition)	Bill Bryson	421

(1) "*Bill Bryson's African Diary*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(2) "*Bryson's Dictionary of Troublesome Words: A Writer's Guide to Getting It Right*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(3) "*In a Sunburned Country*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(4) "*I'm a Stranger Here Myself: Notes on Returning to America After Twenty Years Away*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(5) "*The Lost Continent: Travels in Small Town America*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(6) "*Neither Here nor There: Travels in Europe*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(7) "*Notes from a Small Island*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(8) "*The Mother Tongue:* memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(9) "A *Short History of Nearly Everything*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

(10) "*A Short History of Nearly Everything (Illustrated Edition)*" memiliki kesamaan pada author dan pada title buku sesuai dengan temanya.

$$\text{Akurasi} = \frac{Total\ Data(n) - Data\ Missing}{Total\ Data} \times 100\% = \frac{10 - 0}{10} \times 100\% = 100\%$$

4. Kesimpulan

Dalam proyek ini, metode Content-Based Filtering (CBF) diterapkan untuk membangun sistem rekomendasi berdasarkan konten item pada dataset e-commerce. Dalam proyek ini, kami menerapkan metode Content-Based Filtering (CBF) untuk membangun sistem rekomendasi berdasarkan konten item pada dataset buku. Kami mengikuti beberapa langkah yang mencakup pengumpulan data, representasi item dan pengguna, perhitungan similaritas, perankingan, dan rekomendasi. Dalam proses pengumpulan data, kami memperoleh preferensi pengguna dan riwayat buku yang telah dinikmati. Kami menggunakan metode cosine similarity untuk menghitung tingkat kesamaan antara representasi pengguna dan buku, yang menghasilkan skor kesamaan. Buku dengan skor kesamaan tertinggi dipilih sebagai rekomendasi Kami menggunakan library seperti pandas, sklearn, dan untuk melakukan manipulasi data, perhitungan similaritas, dan visualisasi. Sebelumnya, kami melakukan preprocessing data dengan mendeteksi dan menangani missing value. Kami menggunakan metode TF-IDF untuk vektorisasi teks dan menghasilkan matriks representasi item, yang digunakan dalam perhitungan similaritas menggunakan metode cosine similarity. Kami juga melakukan pengujian akurasi untuk mengevaluasi kinerja sistem rekomendasi.

Berdasarkan hasil akurasi dari kedua perbandingan sistem rekomendasi diperoleh hasil bahwa perbandingan pertama menunjukkan nilai 100% dan perbandingan kedua menunjukkan nilai 100%. Sehingga dari kedua perbandingan tersebut diperoleh rata - rata nilai akurasi sebesar 100% untuk sistem rekomendasi yang telah kami buat. Nilai tersebut membuktikan bahwa hasil sistem rekomendasi yang telah dibuat dengan akurat. Proyek ini berhasil mengimplementasikan metode CBF dalam membangun sistem rekomendasi berdasarkan konten author pada dataset buku. Sistem rekomendasi ini menghasilkan akurasi yang baik dalam memberikan rekomendasi kepada pengguna