BookRecs

A python package to read and analyze data to generate book recommendations using different algorithms.

Charles Reinertson
Rebecca Klein
Urmika Kasi

Background

- Web developers need simplified ML solutions
- Similar to the Netflix model, book recommendations are a hot and growing industry
- A recommendation engine is a type of machine learning which offers relevant suggestions to customers.
- This package simplifies the process for web developers to create a website around a recommender system for books

Key Features

Dataset Handling

Eliminate the cleaning and data handling aspects for a user.

Built-in Options

Recommender system that retrieves books based on similar books or books highly rated by similar users. Provides different algorithms such as KNN and NMF to perform prediction

Scalable

Easy to add and implement new algorithms and datasets.

Data Used

Book-Crossing Dataset

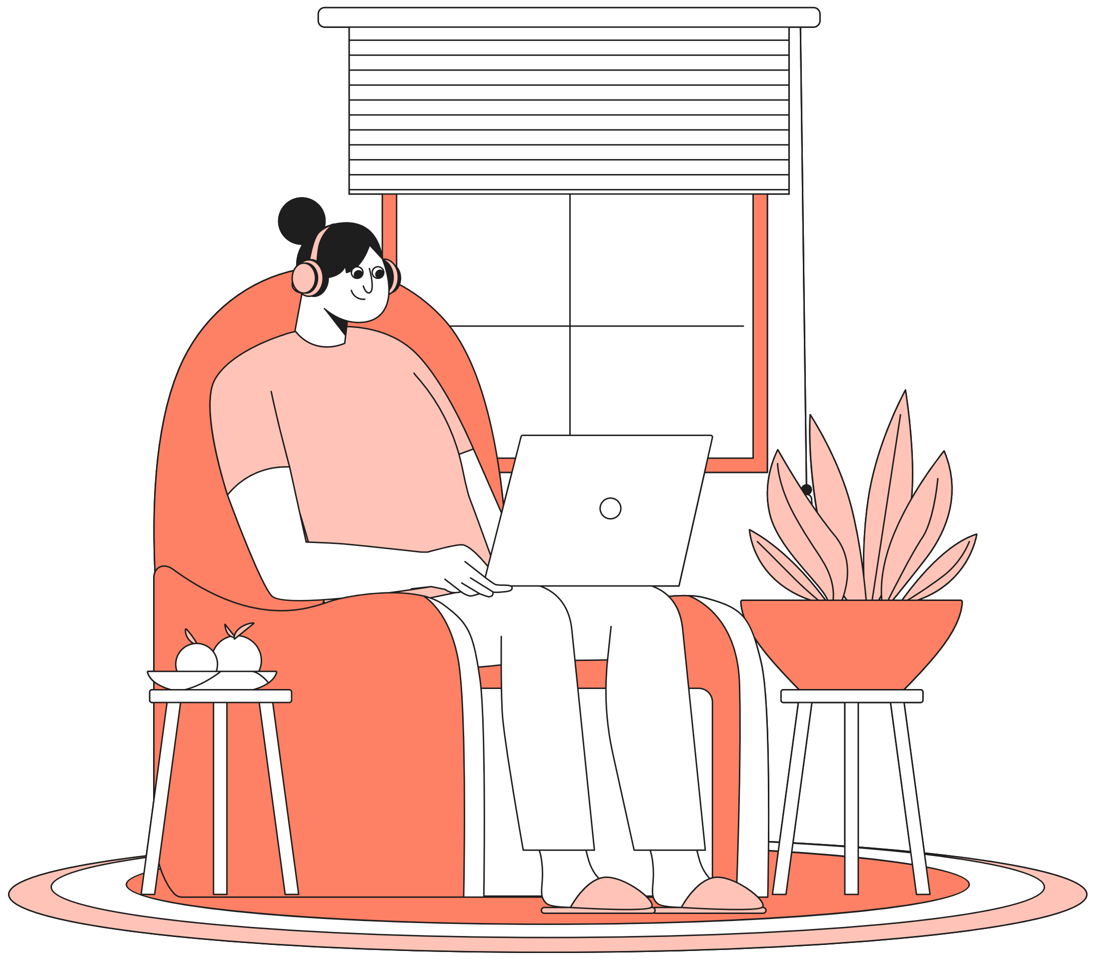
Source

Kaggle.
Link:
https://www.kaggle.com/arashnic/book-recommendation-dataset

Description

- Consists of 3 files:
  - Users
  - Books
  - Ratings
- Over 270k rows of data.

Use Cases



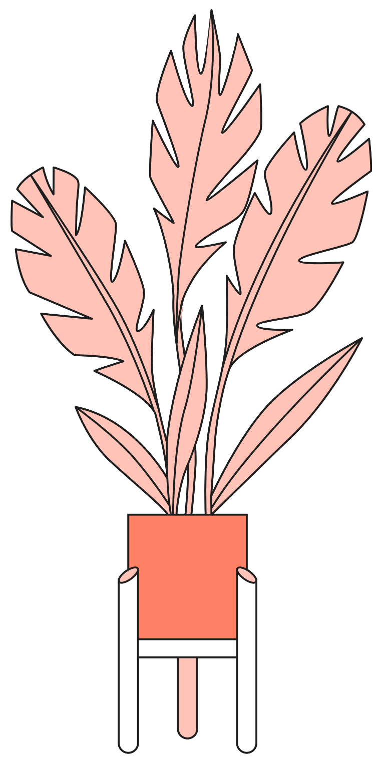
Website Development

Web developers of a book website can simply import the package and integrate necessary APIs to recommend book to a user on a website.

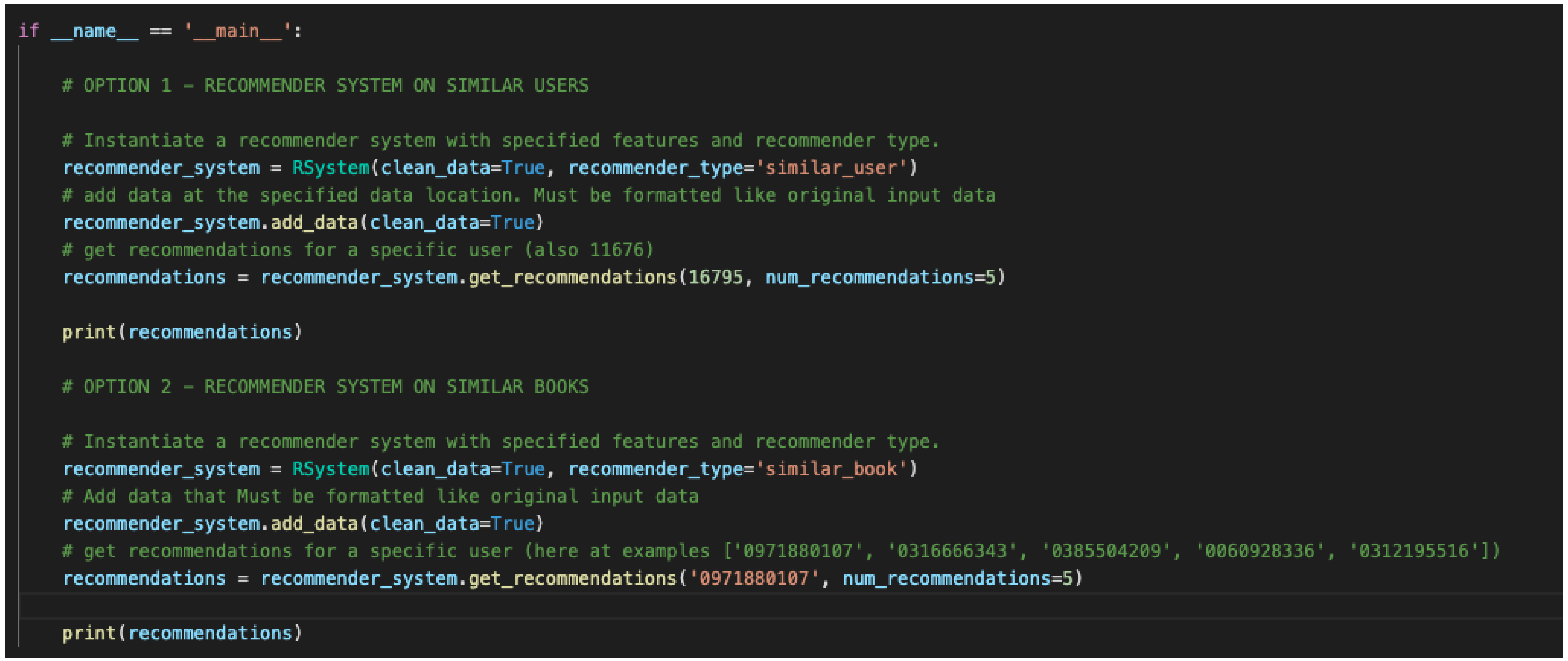
End User Research

This package can be imported for research purposes to test out the existing and new recommendation system algorithms.

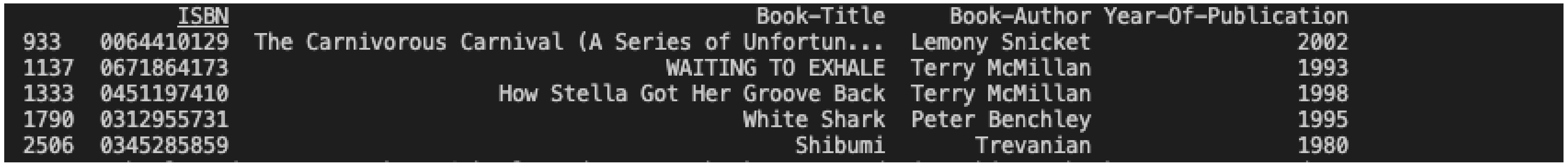
Demo



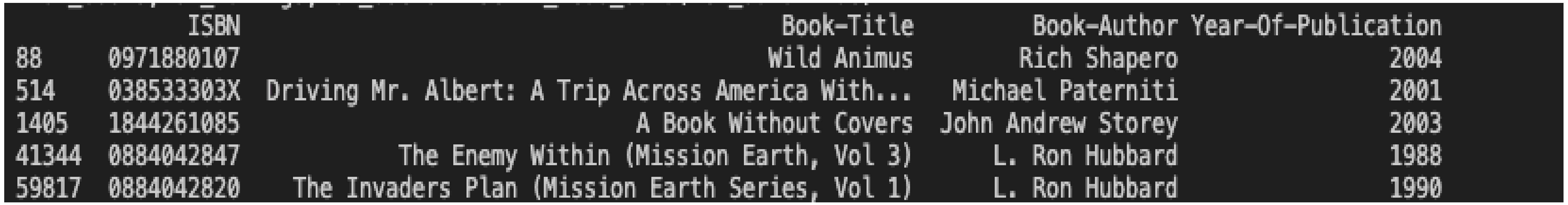
Screenshots



```python
if __name__ == '__main__':

    # OPTION 1 - RECOMMENDER SYSTEM ON SIMILAR USERS

    # Instantiate a recommender system with specified features and recommender type.
    recommender_system = RSystem(clean_data=True, recommender_type='similar_user')
    # add data at the specified data location. Must be formatted like original input data
    recommender_system.add_data(clean_data=True)
    # get recommendations for a specific user (also 11676)
    recommendations = recommender_system.get_recommendations(16795, num_recommendations=5)

    print(recommendations)

    # OPTION 2 - RECOMMENDER SYSTEM ON SIMILAR BOOKS

    # Instantiate a recommender system with specified features and recommender type.
    recommender_system = RSystem(clean_data=True, recommender_type='similar_book')
    # Add data that Must be formatted like original input data
    recommender_system.add_data(clean_data=True)
    # get recommendations for a specific user (here at examples ['0971880107', '0316666343', '0385504209', '0060928336', '0312195516'])
    recommendations = recommender_system.get_recommendations('0971880107', num_recommendations=5)

    print(recommendations)
```



```
            ISBN                                    Book-Title     Book-Author Year-Of-Publication
933   0064410129  The Carnivorous Carnival (A Series of Unfortun...  Lemony Snicket                2002
1137  0671864173                                WAITING TO EXHALE  Terry McMillan                1993
1333  0451197410                      How Stella Got Her Groove Back  Terry McMillan                1998
1790  0312955731                                      White Shark  Peter Benchley                1995
2506  0345285859                                          Shibumi       Trevanian                1980
```



```
             ISBN                                       Book-Title          Book-Author Year-Of-Publication
88     0971880107                                      Wild Animus         Rich Shapero                2004
514    038533303X  Driving Mr. Albert: A Trip Across America With...    Michael Paterniti                2001
1405   1844261085                              A Book Without Covers  John Andrew Storey                2003
41344  0884042847           The Enemy Within (Mission Earth, Vol 3)       L. Ron Hubbard                1988
59817  0884042820      The Invaders Plan (Mission Earth Series, Vol 1)       L. Ron Hubbard                1990
```

Design

01

Process Data

Creates a class (BookDataset) with methods to read, clean, split, and return the dataset.

02

Recommender

Includes wrapper classes to create, fit, and get recommendations for KNN and Matrix Factorization models.

03

System

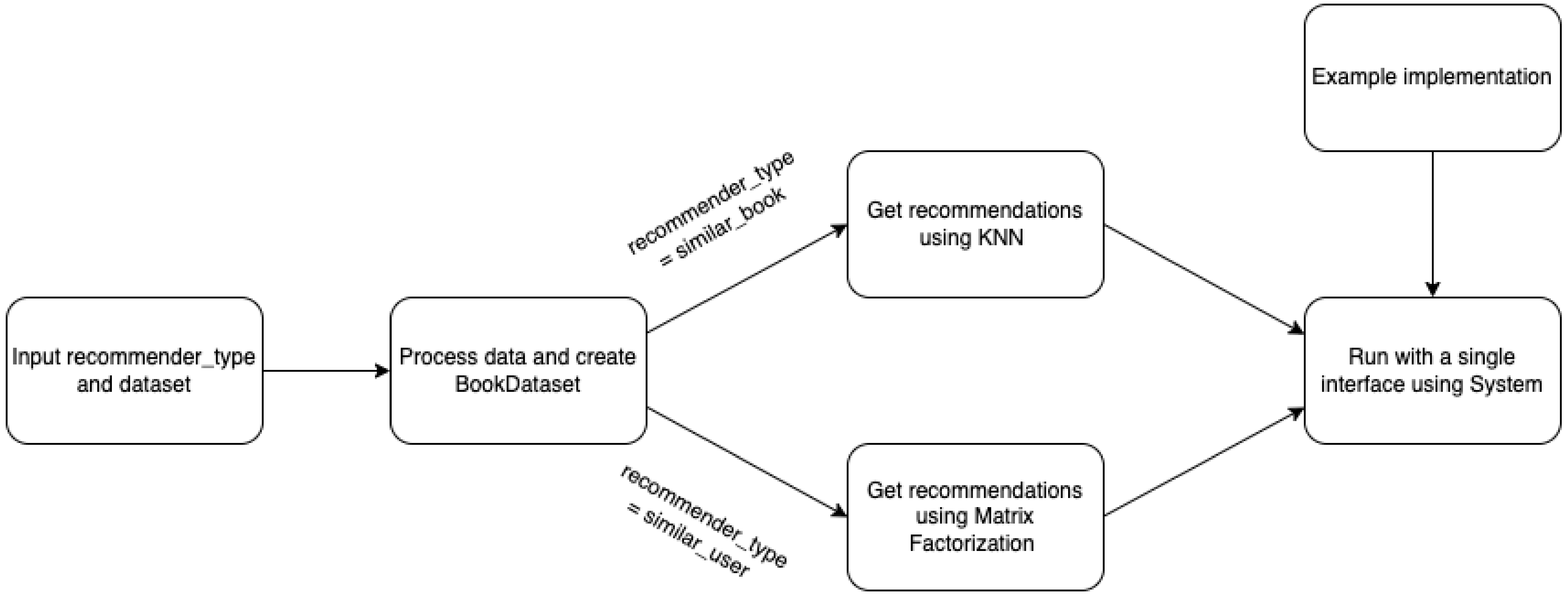
Integrates the recommenders and datasets into a single interface for operation (class RSystem).

04

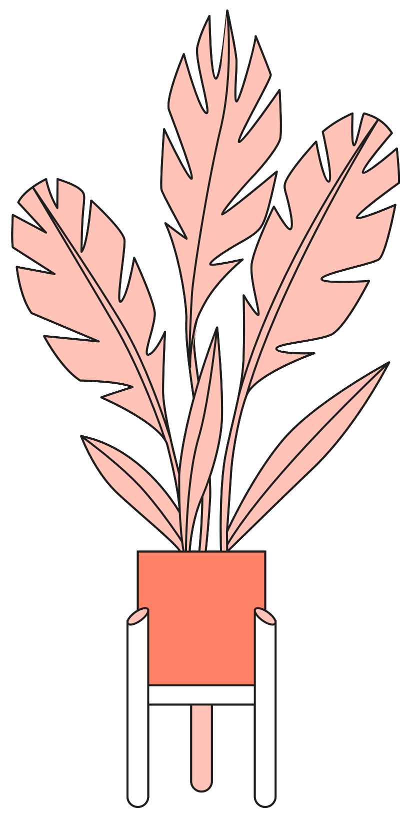
Sample Implementation

Includes code that loads data, builds models, and provides recommendations depending on recommendation preference (similar_user or similar_book)

Component Interaction


Input recommender_type and dataset
Process data and create BookDataset
recommender_type = similar_book
recommender_type = similar_user
Get recommendations using KNN
Get recommendations using Matrix Factorization
Example implementation
Run with a single interface using System

Project Structure



```
book_recommendations/
  |- bin/
     |- _mypath.py
     |- main.py
  |- book_recs/
     |- test
        |- _mypath.py
        |- test_knn.py
        |- test_matrix_factorization.py
        |- test_process_data.py
        |- test_system.py
        |- test_utils.py
     |- __init__.py
     |- process_data.py
     |- recommender.py
     |- system.py
     |- utils.py
  |- data/
     |- Books.csv
     |- Ratings.csv
     |- Users.csv
  |- docs/
     |- Design_Specification.md
     |- Final_Presentation.pdf
     |- Functional_Specification.md

 |- new_data/
    |- Books.csv
    |- Ratings.csv
    |- Users.csv
 |- README.md
 |- config.json
 |- setup.py
```

Lessons Learned

Continuous Integration

Understood integration with Travis CI.

Design Principles

Understood how to make design decisions at every phase of software development.

Code Quality

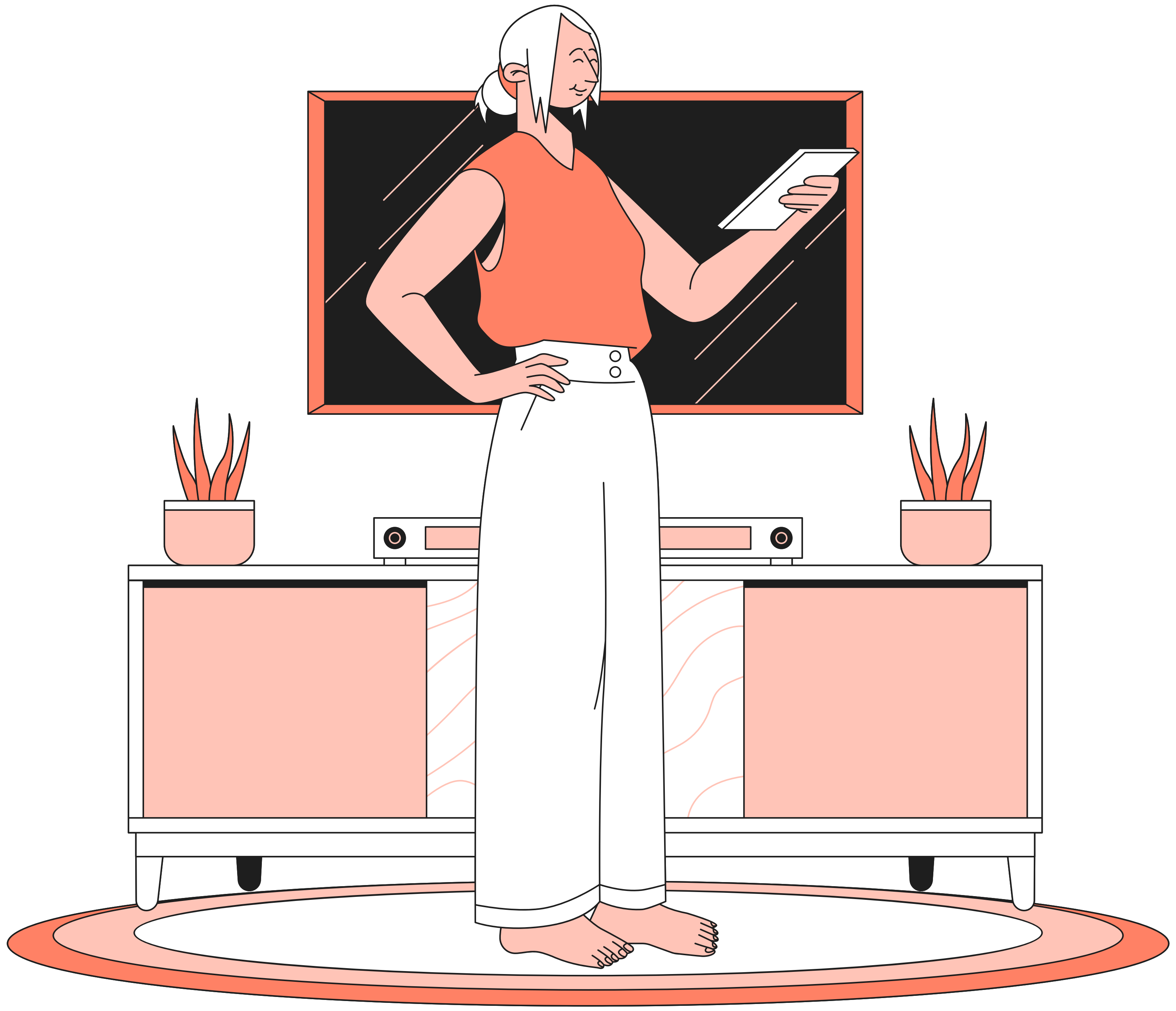
Understood code quality maintenance in compliance with flake8.

Testing

Understood how to make tests using pytest.

Future Work

- Deployment on PyPI
- More Algorithms
- Evaluation Metrics



Questions?