Book Management System: A Python Library for Organizing and Analyzing Books

Authors: Albert Yildirim, Rich Goodier, Suchita Sharma, Teja Ramana Modukuru

Repository: https://github.com/richgoodier/DS5010_library_project

Abstract - The Book Management System is a Python Library designed to efficiently organize and analyze collections of books. Utilizing classes for book representation and library management, it offers features like ISBN validation, text analysis, data visualization, and advanced search capabilities. This library serves as a tool for bibliophiles, librarians, and researchers alike, providing a programmable interface for managing and understanding book collections.

Index Terms - library, book, text analysis, data visualization

Introduction

In an era where digital libraries and information management have become crucial, the Book Management System emerges as a Python-based solution to simplify the organization, search, and analysis of book collections. This package addresses the need for a programmable interface to manage large collections of books, a task often cumbersome and time-consuming. It facilitates efficient library management, ranging from tracking individual book details to analyzing broader literary trends within collections.

Package Overview

The Book Management System, built in Python, is designed to provide an intuitive yet powerful tool for librarians, researchers, and bibliophiles. It is composed of two primary components:

Classes

1) **Book Class**: Represents a single book, encapsulating detailed attributes such as title, author, genre, and text content. It offers functionalities like:

- ISBN validation using isbnlib, ensuring reliable book data management.
- Text analysis capabilities, including word count and frequency
- Infer book themes, utilizing the nltk library for natural language processing and analysis.

2) **Library Class**: Manages collections of Book objects. This class is the backbone of the system, allowing users to:

- Add and remove books from the collection.
- Search through the library by various criteria, such as title and text content.
- Export book data in CSV format, enhancing data portability and external analysis capability.

Functions and Utilities

- **Text Analysis Tools**: Functions for analyzing and visualizing data, such as author frequency and genre distribution, using matplotlib.
- **Search Algorithms**: Efficient search algorithms to quickly find books by titles, authors, or content.
- **Data Import Capabilities**: Functionality to import library data from CSV.

Usage and Examples

The Book Management System is designed with ease of use in mind. Here are some examples demonstrating its functionality:

1) Initializing the Library:

```
from library_project import Library, Book
my_library = Library()
```

2) Adding a Book:

```
book = Book(isbn="9780590353427", genre="Fantasy",
text=["Page 1 text", "Page 2 text"])
my_library.add_book(book)
```

3) Searching for a Book by Title:

```
found_book = my_library.search_by_title("Book Title")
```

4) Retrieving Basic Info of a Book (title, author, genre) or page count:

```
print(book)
print(len(book))
```

5) Bookmarking pages of books and favoriting books:

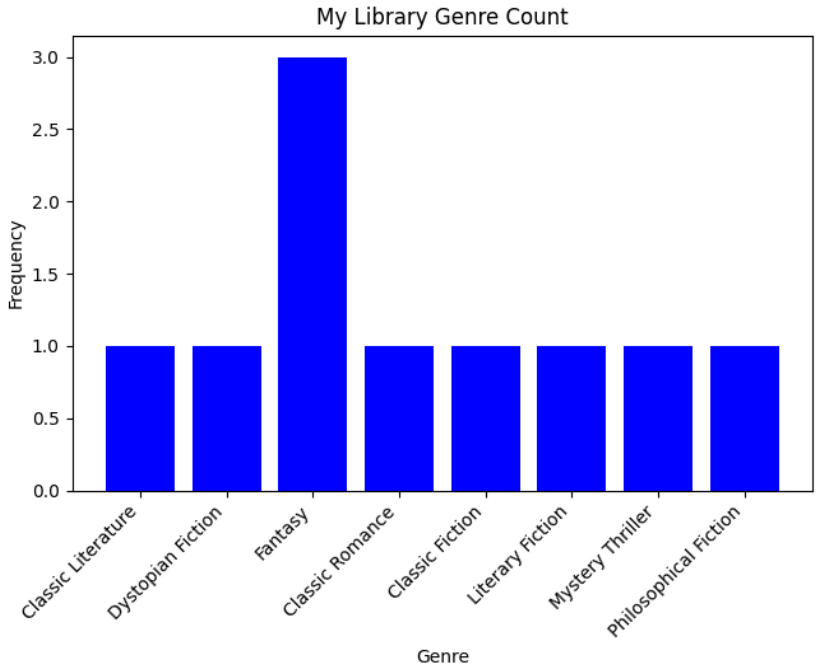
```
book.set_bookmark(2)
book.get_bookmark
my_library.search_by_title("Book Title").set_favorite()
```

6) Listing the inferred themes of a book based on word frequency:

```
book.themes(5)
```

7) Visualizing Genre Frequency:

```
my_library.freq_genre()
```



These examples showcase the practicality of the system, addressing common tasks in library management.

System Utility and Ecosystem Placement

The Book Management System fills a significant niche in digital library management. Its integration of text analysis and data visualization tools positions it as a unique solution for managing and understanding book collections. It complements existing tools in the Python ecosystem, such as **pandas** for data manipulation and **numpy** for numerical computations, by adding a layer of specialized functionality for book data.

Roadmap

One future improvement upon the package that should be prioritized would be to integrate it with well-known bibliographical conventions like the ONIX (ONline Information eXchange) and MARC (MAchine-Readable Cataloging) standards, both used to communicate bibliographical data across platforms.

To fully appreciate the functionality of this library, a user interface should be designed. This would enable the user to access the text of a book in a more readable format. Furthermore, some functions return book objects, which serve well for internal use but are less helpful for customer-facing utility.

Conclusion

The Book Management System stands as a testament to the versatility and power of Python for developing domain-specific tools. Its focus on library management and book analysis provides a valuable resource for those managing book collections, whether in academic, personal, or professional settings. The system's balance of simplicity and functionality makes it a noteworthy addition to the digital library management landscape.

Statement of Contributions

Albert Yildirim - developing and unit testing (UTests_2) the Library Class
Rich Goodier - developing the Book Class; writing report and README.md; assembling package
Suchita Sharma - developing and unit testing the Book Class
Teja Ramana Modukuru - developing and unit testing (UTests_1) the Library Class

References

Special thanks to those who contributed to creating the following Python libraries:

- isbnlib (https://pypi.org/project/isbnlib/)
- nltk (https://www.nltk.org/)
- matplotlib (https://matplotlib.org/)