Book Recommender

Sabina Alves, Ebrahim Moosa, Cody Gunter, Lyndah Mupfunya

Group 4

Our Data

- Our data is a kaggle dataset that includes:
  - The url to each book's page on the goodreads website, book title, author, book rating, number of reviews, genre, the book image, and the book's description, the book's ISBN, and recommendations.

- Initial cleaning included cutting the instances of multiple authors to just one, most prominent being included. The page count was converted to an integer by stripping 'pages' from the column. The original dataset had multiple genres listed, the most frequent being selected. The ISBN values were dropped due to null values, and the recommendations column was dropped. Non english titles were also dropped.

Questions/Goals

Our goal was to create two different recommenders so that users can get books based off of content or score.

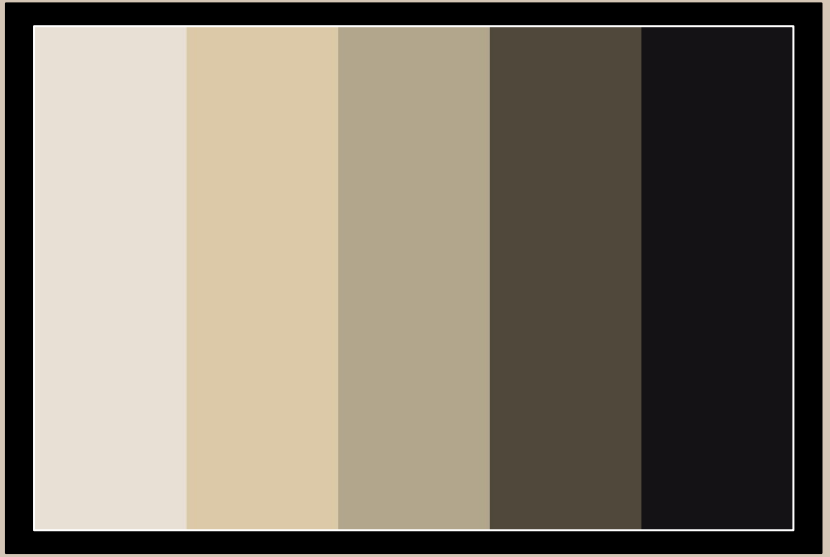
How would we utilize NLP?

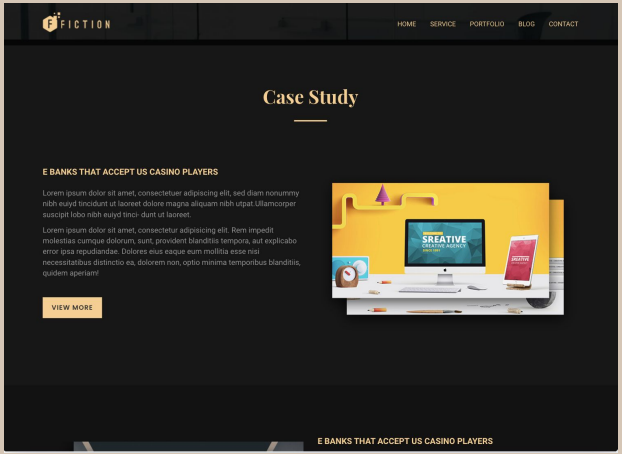
How do we recommend books without user data?

- What would our feature data be?

Theme

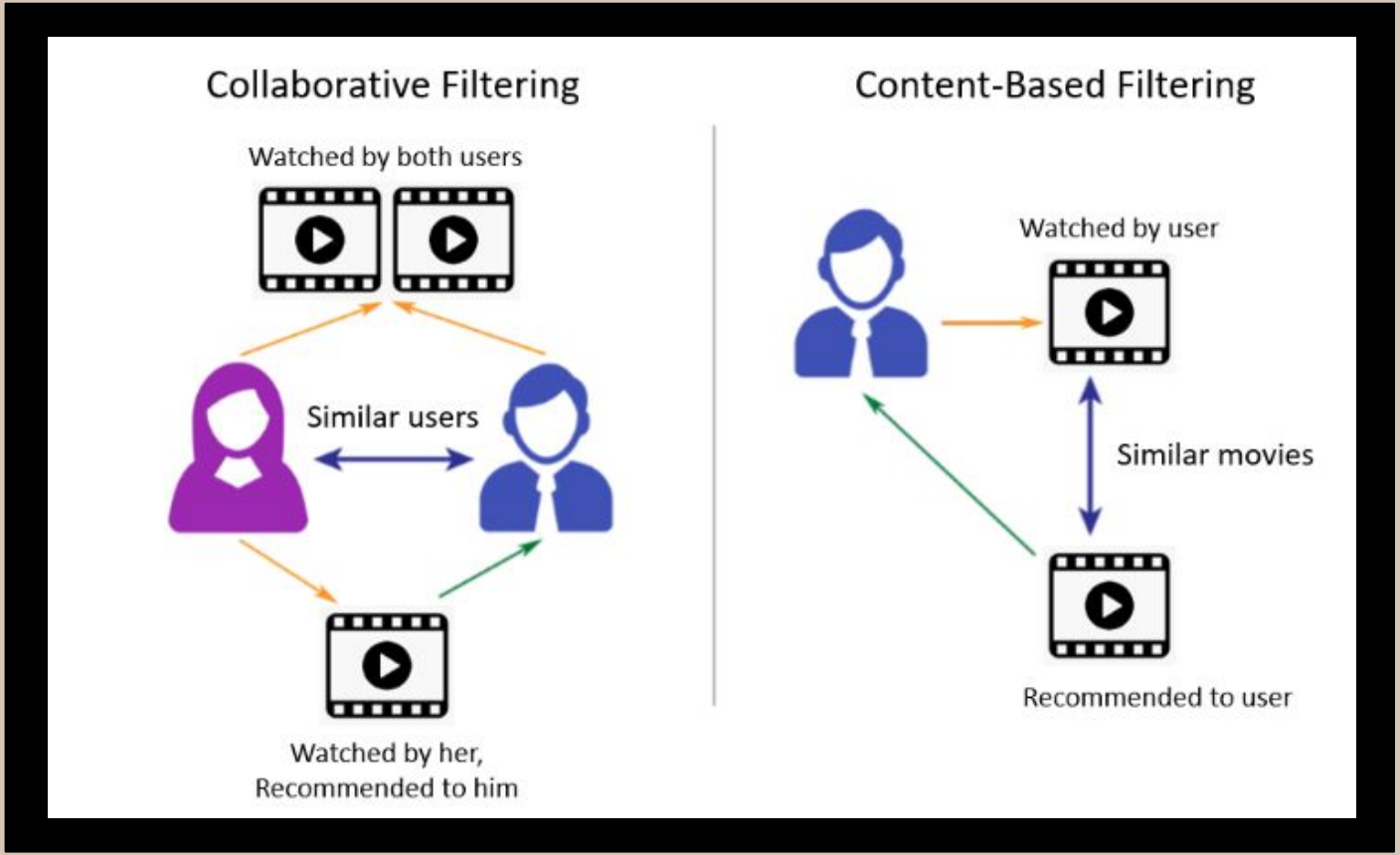
- Our color theme is based on book page color and shelving
- We used the fiction website template from ThemeFisher, heavily incorporating black and beige.





Recommender Models

Recommender Models



Recommender Models: Content Based Filtering

KNN Recommender

- Numerical Data
- Genre and Author
- Sentiment Scores (NLP)

NLP Recommender

- Book Description
- Genre

NLP: Data Processing

Get Key Words	Clean Up	Vectorized	Bag of Words
Keywords grabbed from book title and book description using the RAKE library. Stop words and punctuation dropped.	Additional symbols and numbers dropped. Keywords processed with PorterStemmer and WordNetLemmatizer. Making → make	Title, Genre, and Key words from book description made into lists.	Lists are combined into a bag of words for model.

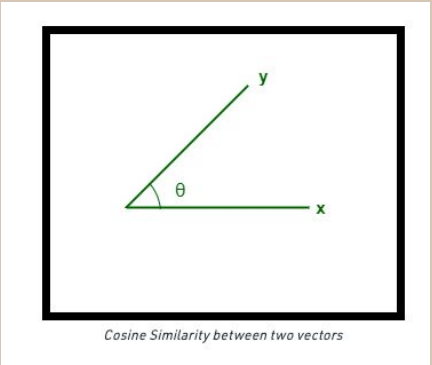
NLP: Model

TF-IDF

- Counts the frequency words and scores the terms based on rarity across the corpus

Cosine Similarity

- measures the angle between two points
- similarity scores generated between all books
- retrieved in flask app with AWS S3 (numpy array)



Cosine Similarity between two vectors

NLP: Sentiment Analysis

- Sentiment scores generated on the Book Description using Text Blob
  - Polarity: between -1 and 1, based on sentiment
  - Subjectivity: between 0 and 1, where 0 is objective and 1 is subjective

```
Sentiment(polarity=0.5, subjectivity=0.2666666666666666)
```

KNN

The KNN model was the simpler of the two models.

To get the model to work, we dropped non-numeric columns and one-hot encoded the genre and author columns.

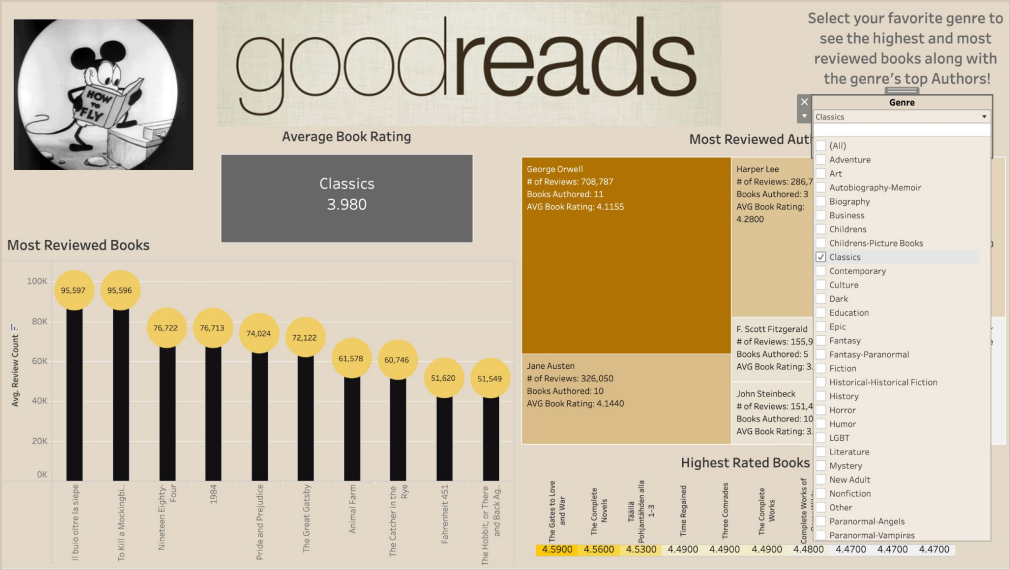
The numeric columns looking at book rating, number of rating, and number of reviews were scaled.

Finally we combined the scaled data, one-hot encoded data, and the sentiment scores to get our final dataframe

This dataframe was run through the KNN model with n-neighbors being set to 11. This let the model give us our book with 10 of its closest neighbors.

Tableau

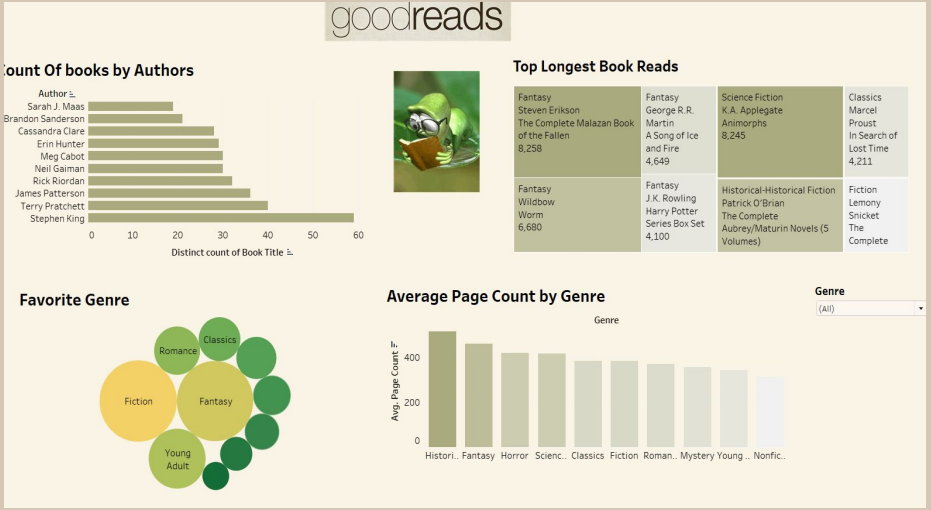
- Our first Dashboard has 4 visualizations responding to a genre filter.
- Lollipop chart, treemap and highlight bar along with a square chart.



- Top 10 lists are displayed of the selected genre(s) displaying:
  - The most reviewed books
  - Highest rated books
  - Most reviewed authors with books authored and average book rating included

Tableau

- Second first Dashboard has 3 visualizations responding to a genre filter.
- The Bubble chart reflects most Popular Genres



- Bar graph shows the top 10 average page count of longest book reads.
- Side bar graph showing distinct count of books by author.

LIVE DEMO

Limitations / Future Work

- Allow for more user inputs on recommendation engines to customize the responses
- Integrate Bootstrap with theme to have interactive table responses
- Find a different web host to allow for larger data files

Thank You

Happy Reading!