Capstone Project #1
Final Report

Justin Hughes-Coleman

Problem Statement

There aren't as many book recommendation services as there are for other forms of media; such as movies and music. The few recommendation services that are on offer really only lead to more recommendations by the same author or genre. This greatly limits the level of service that websites such as Goodreads provide to their community. With "this" recommendation system, it will increase the potential for cross-genre recommendations, which will engage their community more, which will drive more traffic to their site.

I will be using the Kaggle Dataset that was pulled using the Goodreads API that provides access to 6 million reviews across 10,000 titles(files can be accessed at https://github.com/jeydion/Capstone_Project_1). I will then filter down the data to users that have submitted at least twenty reviews in order to build "user profiles". The dataset has been previously used in Kaggle competition to make a book recommendation service; as such the dataset is relatively clean and easy to use. The dataset loads into pandas DataFrame with ease.

I will first run an exploratory analysis that will tell me more about the data. I will run two different models and see which is with the goodreads API I will filter down the data to users that have submitted at least twenty reviews in order to build "user profiles". The second model will be based strictly on the books and will contain their ratings, comment reviews, and associated genres. I plan to utilize the light.fm (https://github.com/jeydion/lightfm) recommendation algorithm in order to implement a machine learning system for book recommendation.

Deliverables

The deliverables for the project will be a recommender system that allows people to type in the name of a book they like and the system will then produce a list the length of their choosing of recommended books. Some books might be from the same genre (for instance if one enters Harry Potter and the Sorcerer's Stone, since this is part of a series, the system will most likely recommend other books in the same series) but most of the books that are recommended should be from a different genre.

Data Description

For my capstone project I am using the Goodreads 10k dataset. This dataset comes from a Kaggle competition in which you use the dataset to create a machine learning recommendation service. Thus, this data has been cleaned much more deliberately than other datasets. However, I still had some data cleaning that had to take place in order for me to get the data ready for analysis. I loaded the dataset from the csv file (https://raw.githubusercontent.com/jeydion/goodbooks-10k/master/books.csv') that is provided in the Goodreads dataset. I stored the csv file as a pandas dataframe in order to work more effectively with the data. The dataset has 22 columns and is indexed by book id (I use two different id's in my index, one relating to the goodreads book ID database and another to find the position of the book in this specific pandas dataframe. I then began to clean the data. I removed several columns from the dataset that I deemed either as duplicate information or not relevant. For example three of the columns used various ID's for each book that were duplicated from another dataset. I am not using

that data so these rows were omitted. There were two rows at the end that links to the cover images of each of the books. Since my analysis does not include image recognition, I omitted these as well (Though it would be an interesting side project if all the books had photo covers; creating an algorithm that would indeed judge a book by its cover).

Missing Values

I found there are 21 NAN values in the Year column. Since the amount is so low, I will manually replace the nan values with the correct years. I simply used Google to confirm the missing book year values.

There are 700 missing ISBNs from the DataFrame. Though there are methods for finding the missing values programmatically. I have decided to drop these two columns from the DataFrame. According the isbn.org, each 10 or 13 digit number assigned to every book does not correspond with the genre. (https://www.isbn.org/faqs_general_questions). The 10 to 13 digit number also isn't used globally, despite the I standing for international. For example, some books that are published in different languages but have an English translation, have an ISBN for the English version of the books but not the original. This creates an issue when trying to compare the average ratings per book because the ratings might span across different language versions of the same book.

I will reference the books by their goodreads_book_id. This will help reference each book in the other datasets that are attached to this dataframe as well. The other datasets include additional information for each book. Most of these datasets include duplicate or irrelevant information. For instance, one dataset links each book to each user who rated that book. This would be useful if we had more data on the users, such as review frequency, average review rating and review patterns. However, the user id does not necessarily associate with each user that rated a certain book. For all the books in the dataset, they all have a user_id = 1, 2, 3 and so on. This presents the problem of there not being unique user ids for each book.

There are 1084 books that have nan values for lang (book language) upon review of all the books, they are all in english and the nan value can be replaced with 'eng'.

There are 67 books that are in Arabic and Persian, in which there might not be English translations or said translations aren't the version of the book the reviews are referring. It would be exceedingly difficult to compare the data with the english or english translation books. This ties into the issue with using ISBN's in the data. There isn't a ISBN for these books and the data associated with English translations are not part of this dataset. Also the number of reviews in general (in Arabic or otherwise) is very low (average < 200), so the data gathered from these will be limited in scope.

There weren't any outlier values other than publication year for some of the books not being in the last two hundred years. These books will remain in the dataset because they are still closely associated with many of the modern books on the list. Most of the books that predate 1800 are considered "classics" so they are commonly read in many languages and have many modern reviews.

Combined Datasets

I have combined several of the goodbooks 10k datasets into one. I am using the 'tags' and 'book tags' datasets and combined both of those to the 'books' in order to easily manipulate one dataset. All three datasets are linked by the "goodreads_book_id" column. This column doesn't serve any analytical or statistical purpose, it is only a label that goodreads assigns to the books in order to keep the information across the datasets consistent.

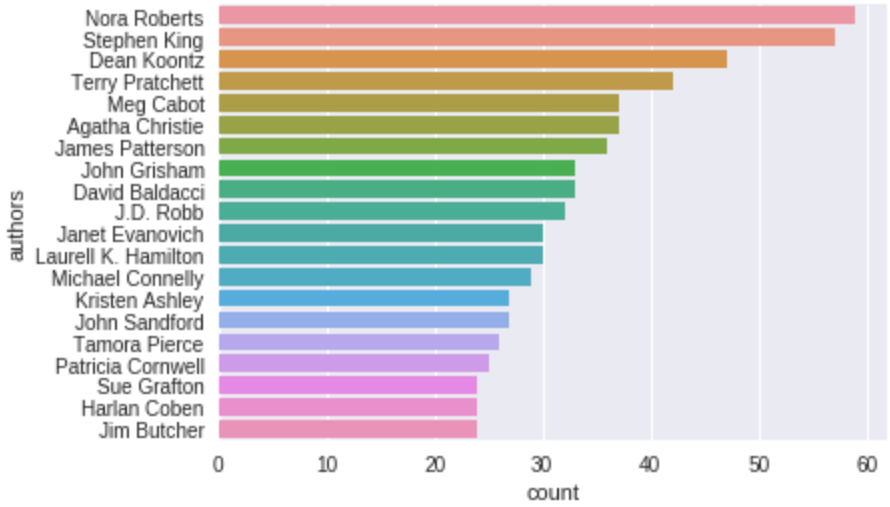
Initial Findings from Exploratory Data Analysis

Introduction

After cleaning the data and doing an initial visual exploration of the data, I am ready to showcase any findings that will help with the further analysis of the data. I will go through the variables I am using, their significance and if any of the statistical analysis showed further insights.

Data Insights

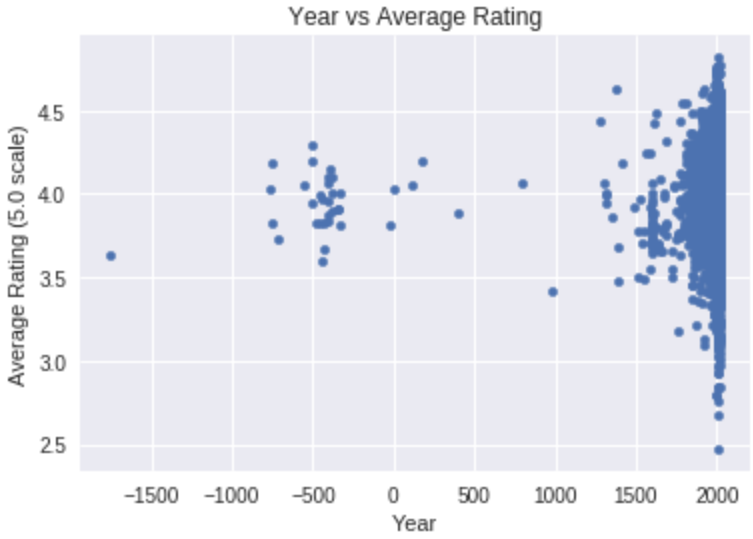
Starting with the variable "authors", some interesting insights were discovered. Firstly, of the nearly 10,000 books in the dataset there are only 4355 authors. This means that each author roughly wrote two books in this dataset. However, upon further inspection there is actually a great difference between the amount of books per author. Fig 1.



There are two explanations for this. One, some authors are very prolific writers such as Stephen King whereas others write a series or several series of books (A typical series is any sting of books longer than three).

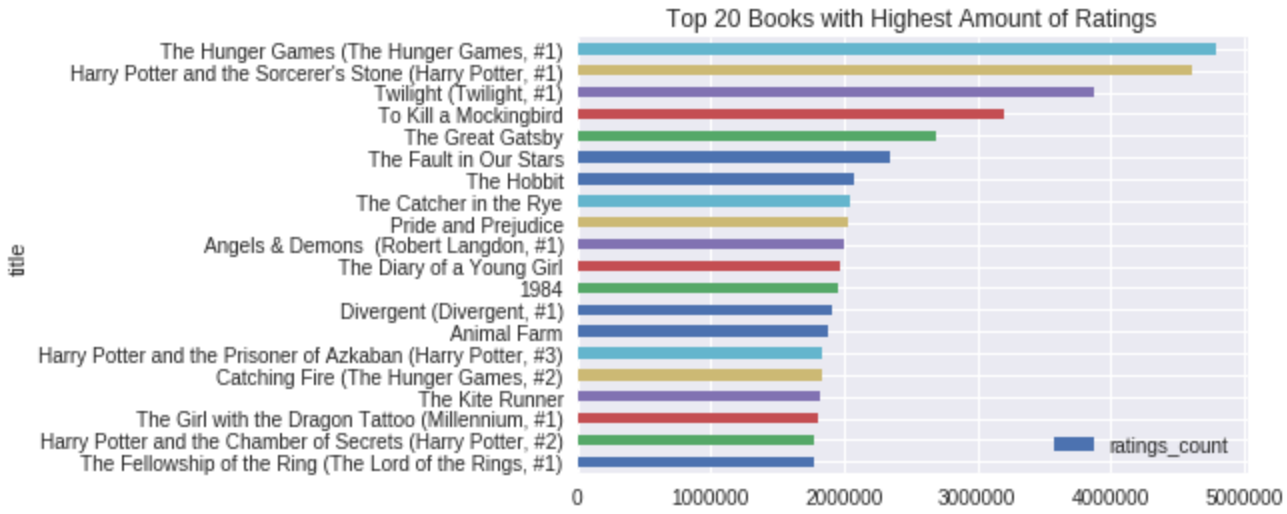
Next is "Books Count" which shows the amount of number of times a book has been tagged or labelled by individual users.

Book "Year" is also included but doesn't show any statistical significance. Though most books on the list are after 1750, there isn't any correlation between book ratings and book year. Fig 2.



The “Ratings Count” measures the number of ratings per book. There doesn’t seem to be much of a correlation between ratings count and book count. In fact, it seems that the books that have the most ratings are outliers. Most books have less than 150,000 ratings. Fig 3.

“Title” refers to the title of each book. This is useful with identifying individual books or series of books. Some books have a very high amount of ratings, much more than the average. Fig 4.



Analysis

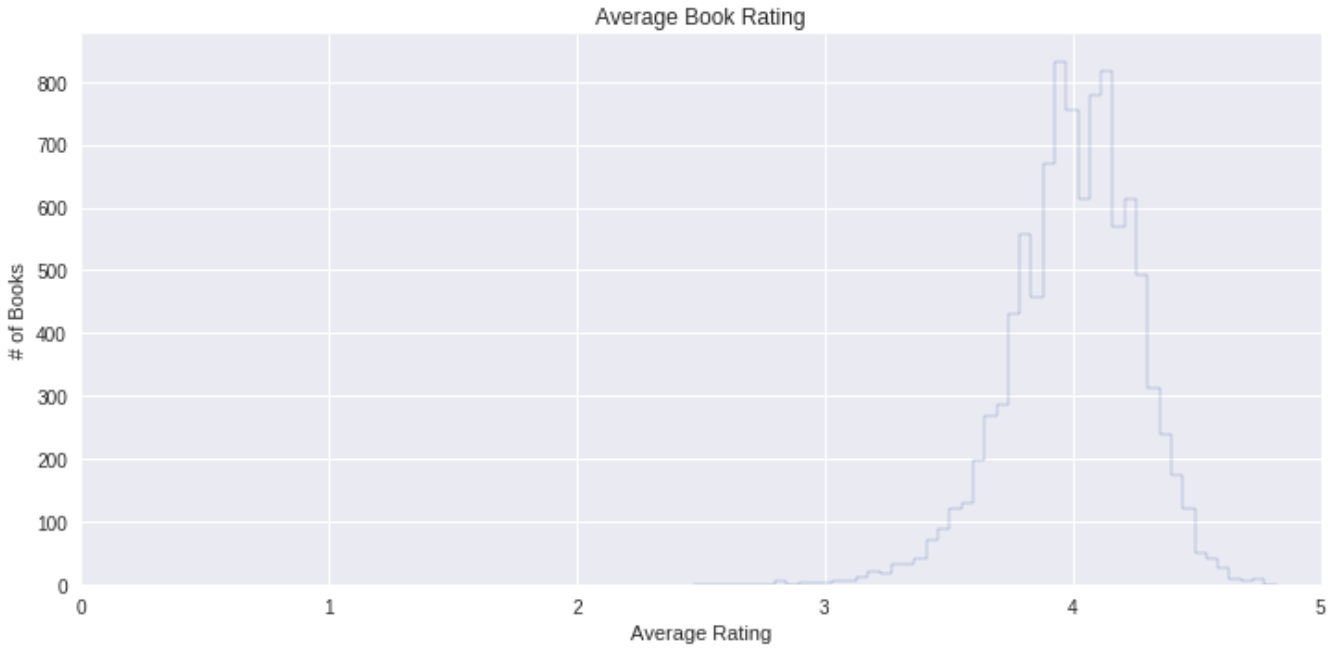
I tested for correlations between many of the numerical data in order to see if further analysis would be required. I first tested if there was any correlation between book counts and ratings counts. I used the Spearman correlation coefficient (Spearman) because both categories contain outliers. The null hypothesis is there isn't any correlation between the two groups. After completing the test and calculating a p-value of 0.0, it is within reason to say that the null hypothesis was failed to be rejected. There isn't a correlation between the two groups and further analysis isn't required. Stat 1

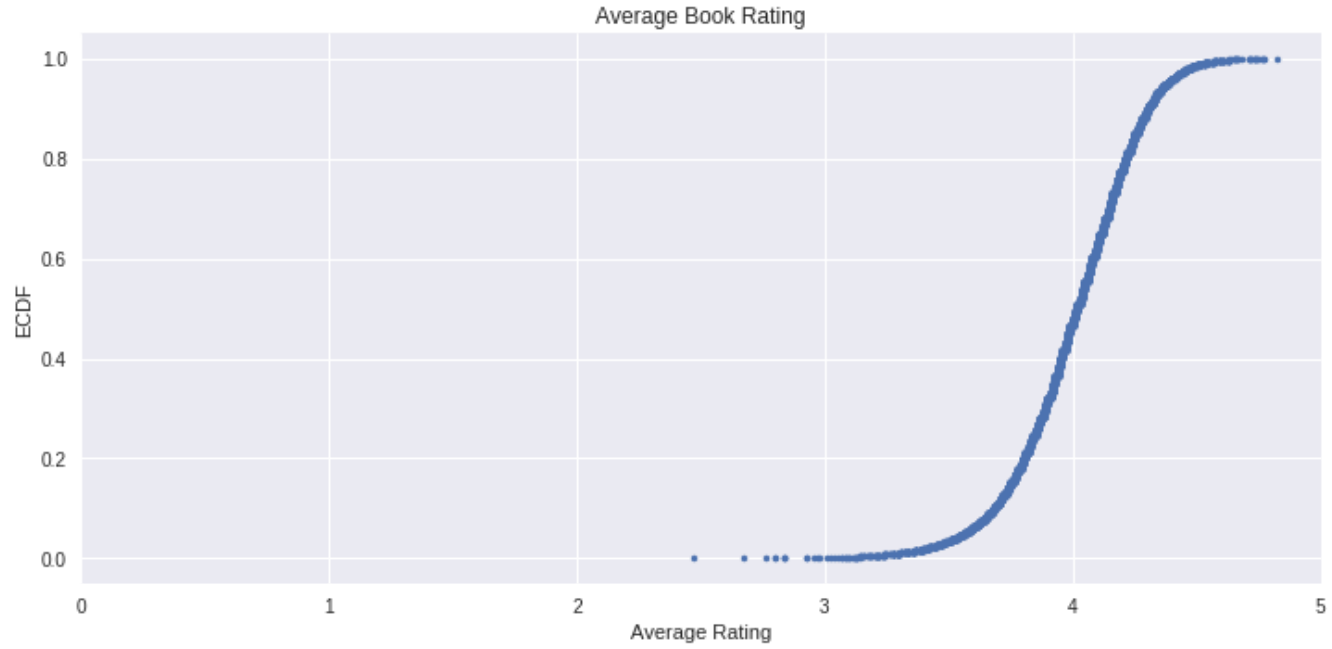
```
SpearmanrResult(correlation=0.3866475377144219, pvalue=0.0)
```

Next I tested to see if there was a correlation between average rating and ratings count per book. My null hypothesis is there is not a correlation between the two variables. I chose the Spearman test again since I am testing correlation between an ordinal and interval variables, both of which contain outliers. My findings show there is a correlation between the two, although not the strongest. The null hypothesis is rejected. Stat 2

```
SpearmanrResult(correlation=0.07484136190471793, pvalue=1.6282945459948974e-13)
```

Based on the descriptive statistics and graphs, the average rating is very skewed to the right, with the vast majority of books rated above 3.5. This leads to speculation that there is a bias in reviews. That is to say, most people will review a book only if they really like it or really hate it. Since this dataset is from Goodreads Kaggle competition, it is more than likely that these books were already the most popular books on the site and as such their average ratings will be higher than the expected normal distribution. Fig 5





Without having the statistical tools at the moment, I concur there is a text correlation that can be used to further analyze the data. In the next section of the project, machine learning algorithms will be applied to further test this theory.

Initial Conclusions

Though only a few correlations were found within the statistical and graphical analysis of the data, the correlations that do appear will be instrumental with the machine learning algorithm in order to solve the original problem of producing a list of books based on the selection of a book. These

correlations along with the text data will still aid in finding additional correlations. After this exploratory data analysis, I will then split the data into training and test data.

In-Depth Analysis

In order to transform the text data to be utilized by machine learning algorithms, first the data must be transformed into numerical data. To do this, I will incorporate TF-IDF or Term Frequency-Inverse Document Frequency. This is one of the most common methods used in analyzing text information. Once the method has been implemented, I will then use cosine similarity in order to calculate a value that will represent the similarity between two books.

TF-IDF involved a two-part process in order to vectorize the text data. First, it counts the frequency of words within a document or body of text (commonly referred to as corpus when not dealing with an actual document). It stores the word frequency of document as a vector. Then, this is compared against how often a term appears across all documents or the corpus and is given a lower value the more frequently it appears. This gives less weight to words like "The", "and", "of" and the like in favor or proper nouns and verbs that better describe the text. In the case of this project, since the comparison is based on book titles, authors and genres, those proper nouns and verbs will be the primary factor in determining the similarity between books. This also helps solves the problem of the user-selected genre tags. Though the tags that are most common will be given less weight in the TD-IDF vector, the unique tags will still only have so much weight associated with them. For example, though many of the books will have the genre tag "fantasy" that will in turn place a lower value on the term "fantasy" itself, at least this will be reflected in all books with that tag. Tags such as "Best Book Ever" will have a higher weight but will not be similar to other tags so it will not be a major factor in the next step in the process, cosine similarity.

Cosine similarity takes the cosine of the angle between to two vectors and using the dot product of two vectors produces a number from 0 to 1 with 1 being the most similarity. Cosine similarity algorithm will be applied to the vectors produced using TD-IDF to determine the similarity between books. In order to get a wide range of options, the cosine similarity algorithm will be applied to different corpi from the dataset. I plan on applying it to the following columns individually; title, author, genre tags and then a corpus with all three included to see if the results for a particular set of books is consistent. In the long-term, when a user utilizes this algorithm to find a similar book to a book that is not listed in this dataset, all the algorithm will have is a the title of the book. This could be improved upon if the user has to use this recommendation system on the Goodreads platform where the book will have more data associated with it for the algorithm to implement in the similarity calculations

Below are examples of the cosine similarity algorithm recommendation engine with different vectorized corpora. The book used to demonstrate these results is ‘1984’ by George Orwell.

```
In [33]: authors_recommendations("1984")
```

```
Out[33]: 13                                          Animal Farm
         2499                    Down and Out in Paris and London
         6531                                         Burmese Days
         9343                           Keep the Aspidistra Flying
         2841                                    The Art of Loving
         820                                   Animal Farm / 1984
         3883                                 Homage to Catalonia
         2326                         Love Story (Love Story, #1)
         7103                                             Doctors
         5877                                       Three Comrades
         7047    Arch of Triumph: A Novel of a Man Without a Co...
         35         A Game of Thrones (A Song of Ice and Fire, #1)
         104        A Clash of Kings  (A Song of Ice and Fire, #2)
         128        A Storm of Swords (A Song of Ice and Fire, #3)
         157        A Feast for Crows (A Song of Ice and Fire, #4)
         177     A Dance with Dragons (A Song of Ice and Fire, #5)
         1444    A Storm of Swords: Blood and Gold (A Song of I...
         1924    A Storm of Swords: Steel and Snow (A Song of I...
         2070    A Song of Ice and Fire (A Song of Ice and Fire...
         2443    A Song of Ice and Fire (A Song of Ice and Fire...
         Name: title, dtype: object
```

Table 1: Book recommendations based on authors’ names.

```
In [37]: tags_recommendations('1984')
```

```
Out[37]: 50                                  Brave New World
         44                                   Fahrenheit 451
         13                                      Animal Farm
         27                                Lord of the Flies
         66                                     Frankenstein
         60                               Slaughterhouse-Five
         7                              The Catcher in the Rye
         784     Brave New World / Brave New World Revisited
         89                         The Picture of Dorian Gray
         164                              A Clockwork Orange
         123                 One Flew Over the Cuckoo's Nest
         820                              Animal Farm / 1984
         152                              Great Expectations
         78                             A Tale of Two Cities
         374     The Strange Case of Dr. Jekyll and Mr. Hyde
         58                                Wuthering Heights
         6130                                      High-Rise
         678                                   Mrs. Dalloway
         193                               A Christmas Carol
         107                                       Catch-22
         Name: title, dtype: object
```

Table 2: Book recommendations based on tag-names of each book.

```
In [40]: corpus_recommendations("1984")

Out[40]: 50                                  Brave New World
         13                                      Animal Farm
         44                                   Fahrenheit 451
         27                                Lord of the Flies
         820                              Animal Farm / 1984
         7                             The Catcher in the Rye
         9343                       Keep the Aspidistra Flying
         123                  One Flew Over the Cuckoo's Nest
         784     Brave New World / Brave New World Revisited
         164                               A Clockwork Orange
         60                               Slaughterhouse-Five
         66                                     Frankenstein
         89                         The Picture of Dorian Gray
         152                              Great Expectations
         107                                        Catch-22
         1163                              The Invisible Man
         58                                Wuthering Heights
         257                            Flowers for Algernon
         6130                                     High-Rise
         758                                The Awakening
         Name: title, dtype: object
```

Table 3: Book recommendations based on both authors' names and tag-names associated with each book.

Results and Insights

The recommendation engines that were developed showed similarities in books associated with a given title. There are a few areas where the engine can be improved. As previously stated, the engine only works with this specific Goodreads dataset. Though this dataset has the top 10,000 books read on goodreads, it is still not every book on the platform. Thus, this engine needs to be improved to include all titles that are in the Goodreads database. If this recommendation engine is ever implemented on a site such as Goodreads, the corpus recommendation would be the top choice since it uses the most pertinent information about the books in tandem.

Another issue with the engine is that it may be too good at recommending similar books. The original thesis was to create an engine that would recommend books that were similar to the selected book but from different genres. Since authors and tag-name are the primary data used for this engine, it is difficult to see a solution that would purposely and accurately choose different tag-name books that are also similar. The original goal was the vectorized corpus would find similarities between different tag-names and authors that are not apparently to the naked eye. Further development in this area would lead to a more customizable recommendation engine that could potentially utilize filters.

This engine is easy to use but an app or web app integration would take the engine to the next level. By drastically improving the user experience this could lead to more use and in turn more data for the engine to consider. Goodreads could have this recommendation engine integrated into their site on the User Account section that way people can get recommendations to books they have already read.

Conclusion

The original problem facing companies such as Goodreads and companies that aggregate user data is the lack of versatile recommendation systems. Netflix and other media companies develop algorithms for their platforms that allow users to rate movies or select similar movies based on movies that have been previously watched. The goal of this project was to develop such a system for Goodreads to use. This project succeeded in using a similarity algorithm in finding books that are similar to each other on more than one criteria. Though the algorithm works a little too well and won't recommend books outside the genre, it will still find different titles outside of a series or by a different author. This is an excellent starting place for further development in building recommendation systems.