AI-Based Book Recommendation System

Abhishek Degadwala, Parthiban Sundararaj, Vikram Chawla and Seshasai Srinivasan
W Booth School of Eng. Pract. & Tech., McMaster University, Canada

Abstract: Recommendation systems come up with user-specific service support by analyzing their previous activities and predicting their current preferences while looking for specific products. Artificial intelligence (AI) methodologies, specifically computational intelligence and machine learning models and algorithms have been extensively applied for the development of recommender systems to enhance prediction accuracy and counter data sparsity and cold start problems. By successfully implying these recommendation systems, the companies can follow a target-specific approach and set their production targets for a specific product. This also helps the companies analyze the current market trends.

1. Introduction

In this work, we aim to develop an Artificial Intelligence based Recommendation System for the Hamilton Public Library (HPL). The primary intent of this project is to develop a book recommendation system for library staff and users. In this project, AI is used to rank and tag items so that they can be matched with similar titles and content of the currently logged-in user, to provide personalized book recommendations from their inventory of books. In addition to this, the system will include their borrowing history and the user ratings, helping the library enhance their clients' library experience and provide them with a personalized environment. The users of Hamilton Public Library can use the library's website to search and check for the availability of books. HPL has a comprehensive online collection search system that is currently maintained by a third-party company. Through this system, the users can search using keywords, title, author, subject, and tag. Diverse ways of building a recommendation system usually fall into two major categories:

1.1 Content-based (CB) methods:

In content-based systems, the recommendation is based on the content present in an item's description, and the same is mapped with the user's preferences. Content-based systems aim to recommend products that are like the one in which the user has shown some interest. First, different item attributes are extracted from documents/descriptions. For instance, a shoe can be represented by attributes such as size, occasion, price range, brand, etc. These attributes can be obtained directly from organized/structured data, such as a table, or unstructured data, such as an article or news.

A content-based recommendation is based on product description and is thus user-independent. Therefore, this kind of system does not suffer from the data sparsity problem. Also, content-based recommendation systems can recommend new products to users, which act as a counter to the new product cold-start problem. Finally, content-based recommender systems end up with a more descriptive recommendation result.

1.2 Collaborative filtering (CF) methods:

CF-based recommendation systems analyze the utility of a product considering the ratings given by other users. Today, CF is one of the most popular techniques applied in recommendation systems. The basic assumption underlying the CF methodology is that users who have interests in common will utilize similar products. Therefore, a system following the CF mechanism relies on information provided by the users who map the preferences of the given user.

Model-based CF constructs a model to predict a user's rating on products using machine learning or data mining techniques rather than a heuristic approach that is used in memory-based CF. This method was originally developed to rectify the defects in memory-based CF, but it has been widely employed for solving problems in other domains. In addition to the user-product rating matrix, additional information such as location, tags, and reviews are used. The model-based CF technique is applied successfully if this supporting information is combined with the rating matrix. Matrix factorization was a product of the Netflix Prize competition of 2009, and it is

still one of the most popular algorithms in this field. It projects both, the user space, and the item space, onto the same latent factor space so that they are comparable.

2. Solution Design

In this work we have developed a hybrid model using both the content-based model and the CF model. For the Content-based model, the recommendation engine is built using book descriptions by first converting each book title and description into vectors using TF-IDF and bigram. The model recommends a similar book based on the description. Next, we calculate the similarity between all the books using cosine similarity. Finally, we define a function that takes the book title and genre as input and returns the top five similar recommended books based on the title and description. For the CF model, we will be using the Matrix factorization technique.

2.1 TF-IDF

Term Frequency-Inverse Document Frequency (TD-IDF) is a measure used in the fields of information retrieval (IR) and machine learning to quantify the importance or relevance of string representations in a document amongst a collection of documents TF-IDF can be broken down into two parts TF (term frequency) and IDF (inverse document frequency).

Term frequency works by looking at the frequency of a particular term relative to the document. Inverse document frequency looks at how common (or uncommon) a word is amongst the corpus. IDF is calculated as shown in Equation (1).

$$idf(t, D) = \log\left(\frac{N}{count(d \in D : t \in d)}\right) \quad (1)$$

where t is the term (word) we are looking to measure the commonness of, and N is the number of documents (d) in the corpus (D). The denominator is simply the number of documents in which the term, t, appears in. By multiplying these values together, we can get our final TF-IDF value as shown in Equation (2).

$$tf\ idf(t, d, D) = tf(t, d).idf(t, D) \quad (2)$$

2.2 Cosine Similarity

Cosine Similarity measures the cosine of the angle between two non-zero vectors of an inner product space. This similarity measurement is particularly concerned with orientation, rather than magnitude. In short, two cosine vectors that are aligned in the same orientation will have a similarity measurement of 1, whereas two vectors aligned perpendicularly will have a similarity of 0. If two vectors are opposed, meaning they are oriented in exactly opposite directions (i.e., back-to-back), then the similarity measurement is -1 (c.f. figure 1). Often, however, Cosine similarity is used in positive space, between the bounds 0 and 1.

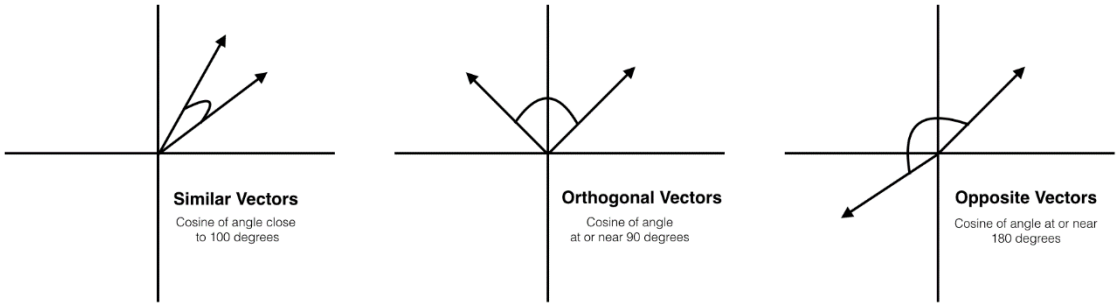

Figure 1: Word similarity measured via cosine similarity

The Cosine Similarity measurement begins by finding the cosine of the two non-zero vectors. This can be derived using the Euclidean dot product formula as shown in Equation (3):

$$A.B = ||A||\,.\,||B||cos\theta \quad (3)$$

Given the two vectors and the dot product, the cosine similarity is defined in Equation (4):

$$similarity = cos\theta = \frac{A.B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (4)$$

The output is a value ranging from -1 to 1, indicating similarity where -1 is non-similar, 0 is orthogonal (perpendicular), and 1 represents total similarity.

2.3 Matrix Factorization

Since sparsity and scalability are the two biggest challenges for the standard CF method, it comes to a more advanced method that decomposes the original sparse matrix into low-dimensional matrices with latent factors/features and less sparsity. That is Matrix Factorization.

To see how a matrix is being factorized, the first thing to understand is Singular Value Decomposition (SVD). Based on Linear Algebra, any real matrix R can be decomposed into 3 matrices U, Σ, and V as shown in Equation (5). Continuing using the book example, U is an n×r user-latent feature matrix, and V is an m×r book-latent feature matrix. Σ is an r×r diagonal matrix containing the singular values of the original matrix, simply representing how important a specific feature is to predict user preference.

$$R = U \sum V^T,\ U \in IR^{nXr}, \sum \in IR^{rXr}, V \in IR^{rXm} \quad (5)$$

To sort the values of Σ by decreasing the absolute value and truncating matrix Σ to the first k dimensions (k singular values), we can reconstruct the matrix as matrix A (figure 2). The selection of k should make sure that A can capture most of the variance within the original matrix R so that A is the approximation of R, i.e., A ≈ R. The difference between A and R is the error that is expected to be minimized.

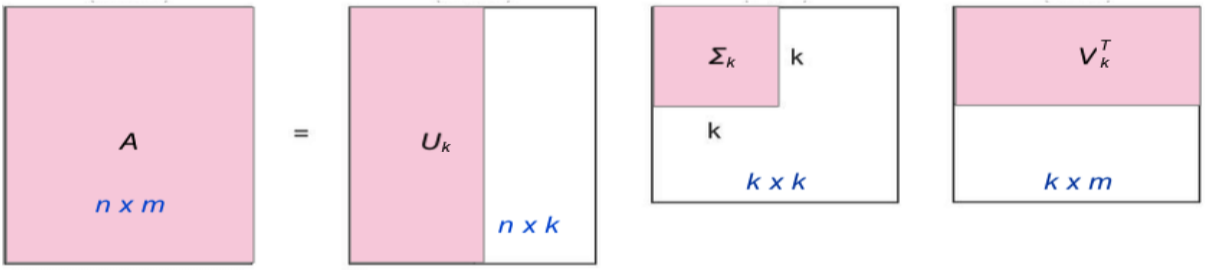

Figure 2: Singular Value Decomposition of Matrix A

When matrix R is dense, U and V could be easily factorized analytically. However, a matrix of book ratings is super sparse. Instead of factorizing R via SVD, we are trying to find U and V directly with the goal that when U and V are multiplied, the output matrix R' is the closest approximation of R and no more a sparse matrix. This numerical approximation is usually achieved with Non-Negative Matrix Factorization for recommender systems since there are no negative values in ratings.

In the predicted rating for a specific user and item in Equation (6), item i is denoted by vector q_i, and user u is denoted by a vector p_u such that the dot product of these two vectors is the predicted rating for user u on item i. This value is presented in the matrix R' at row u and column I (c.f. Figure 3).

$$Predicted\ Ratings: r'_{ui} = p_u^T q_i \quad (6)$$

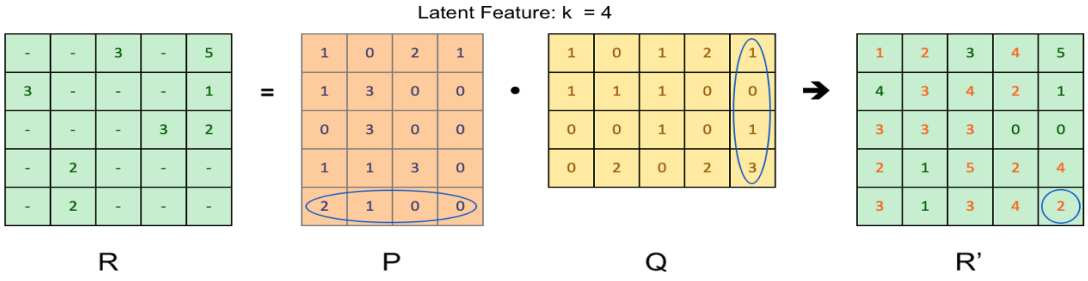

Figure 3: Calculating the predicted ratings matrix

Like most machine learning tasks, a loss function is defined to minimize the cost of errors as shown in Equation (7).

$$\min_{q,p} \sum (r_{ui} - p_u^T q_i)^2 + \lambda \left(||p_u||^2 + ||q_i||^2 \right) \quad (7)$$

r_{ui} is the true ratings from original user-item matrix. Optimization process is to find the optimal matrix P composed by vector p_u and matrix Q composed by vector q_i in order to minimize the sum square error between predicted ratings r_{ui}' and the true ratings r_{ui}.

3. Output

A sample recommendation from this system is shown in Figure 4 and 5.

```
cf_recommender("Harry Potter and the Sorcerer's Stone (Book 1)")
```

```
[['Harry Potter and the Chamber of Secrets (Book 2)', 'J. K. Rowling'],
 ['Harry Potter and the Prisoner of Azkaban (Book 3)', 'J. K. Rowling'],
 ['Harry Potter and the Goblet of Fire (Book 4)', 'J. K. Rowling'],
 ['Harry Potter and the Order of the Phoenix (Book 5)', 'J. K. Rowling'],
 ['The Hobbit: or There and Back Again', 'J.R.R. Tolkien']]
```

Figure 4: Recommendations from Collaborative Filtering Model

```
cm_recommender("Harry Potter and the sorcerer's stone", "audiobooks")
```

```
                                    title
56      Harry Potter and the goblet of fire
54  Harry Potter and the chamber of secrets
58       Alvin Ho collection. Books 3 and 4
55     Harry Potter and the Deathly Hallows
0                                    Monkey
```

Figure 5: Recommendations from Content Based Model

4. References

1. Francesco Ricci and Lior Rokach and Bracha Shapira, "Introduction to Recommender Systems Handbook," Recommender Systems, Handbook, Springer, 2011, pp. 1-35
2. Linden, et al., "Collaborative recommendations using item-to-item similarity mappings," United States Patent 6,266,649, July 24, 2001
3. Michael J. Pazzani and Daniel Billsus, "Content-Based Recommendation Systems," In "The Adaptive Web": P. Brusilovsky, A.
4. Kobsa, and W. Nejdl (Eds.), LNCS 4321, pp. 325 – 341, 2007. © Springer-Verlag Berlin Heidelberg 2007
5. Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for IDF. Journal of documentation.