Book Recommendation System

By Apoorva Banubakode, Purvi Misal, Shalabh Neema
CMPE 257 - Machine Learning
San Jose State University

Introduction

One of the most crucial issues, nowadays, is to provide a personalized experience to each user, based on their preferences to keep them engaged with the business. To achieve this goal, we need to generate relevant recommendations. A Recommendation Engine could be utilized as a tool to help the users in decision-making processes offering different items and options. It can be used to predict the likelihood that the user will like the product and then to recommend relevant items to users. In this case an item could be anything such as a document, a location, a movie, an article or even a user (friend suggestion). The main objective of the recommender systems is to suggest items which have great potential to be liked by users. In modern recommender systems, various methods are combined together with the aim of extracting patterns in available datasets.

How many times has it happened that you were suggested a book by a friend and you read it only to be disappointed later. The goal of this project is to solve this by building an Recommendation engine which would suggest books according to your liking/interested genres, previous read books and book ratings, of fellow readers depending on what people similar to you like to read - a totally personalized solution

The project uses the goodbooks-10k dataset, which was published under a Creative Commons license by Zygunt Zajac on the FastML website in 2017. The dataset includes about 6 million ratings of 10,000 books by more than fifty thousand users with the highest volume of ratings (Zajac). The project involves grouping readers based on their book tastes so we can recommend books that the users haven't read but their fellow group members (similar to them) have rated highly.

Data :-https://www.kaggle.com/zygmunt/goodbooks-10k

Literature Review/Theoretical background

The most common two problems recommendation systems can solve:

1. The Top N recommendation problem: (recommending top n books).
2. The Rating system problem: (predicting how the reader will rate books)

Recommendations Methods used up till now:

Content-based recommender system

Content-based recommender systems are systems that utilize items' descriptions to make recommendations to users. Here, the items and their metadata is of utmost importance to the model. The assumption of content-based systems is that if a user has reacted positively to items possessing certain characteristics, she must also be interested in similar items which are described as having the same attributes. The most common representations of items' characteristics are keywords and the keywords' weights, which denote how important the keywords are in describing the items.
The concepts of Term Frequency (TF) and Inverse Document Frequency (IDF) are used content based filtering mechanisms in this type of recommender. Content-based recommender systems are often useful in cold-start scenarios, where other users' ratings are of little use, due to the fact that these systems do not require other users' ratings to make recommendations. However, recommendations produced by content-based systems are typically not as diverse as that of other systems. The use of a content-centric similarity function for ranking the items results in recommended lists consisting of highly similar items content-wise, reducing the diversity of the recommendations.

Collaborative Filtering

In collaborative filtering, instead of comparing user interests with available items, the user is compared with the other users and similar users are found. Then in the next step based on the similar users, peer users, recommendations are provided for the active user. Additionally, in collaborative approach all users are involved in rating based on their interests. In this way, similar users who have the same tastes are linked together (put in the same group/community).
Therefore, it can be said that collaborative filtering does not have content-based filtering limitation since it does not depend on the contents; as a result, filtering information from any sources is possible. Moreover, using collaborative filtering makes complex and deep relationships between users and items, such as the needed quality or taste of a certain user. This feature makes it possible to make a difference between poor document and well written document by contrast of pure content-based approach. Lastly, collaborative filtering returns more accurate results since it utilizes real users ranking instead of pure machine made predictions. An item-based collaborative filtering (IBCF) recommender system assumes that item A is similar to item B if they receive similar ratings and if a user reacts positively to item A, she will like item B as well.

Popularity-based recommender system

A popularity-based recommender system assumes that users prefer popular items. The notion of popularity can be interpreted in different ways, and as a result, different methods of producing

recommendations can be formed. A possible interpretation of popularity is the number of ratings an item has. A popular item is intuitively known by many users and consequently is rated by many users. This interpretation assumes that the higher the number of ratings an item has, the more likely it will be reacted positively by users.

Hybrid Approach

Generally, a hybrid recommender system is a combination of multiple other recommender systems with the aim to achieve the best of all worlds. A hybrid system's assumption is a combination of different assumptions coming from the systems it integrates. As a result, a hybrid system has access to more resources, based on which it can provide more robust recommendations. However, an obvious downside to implementing a hybrid system is the extra computational complexity added for operating multiple recommender systems simultaneously. A common implementation of the hybrid approach is the weighted hybrid system, where the scores of several systems are combined into a single score using a set of weights, which are often determined through trial and error.

Restricted boltzmann Machine

RBMs are a two-layered artificial neural network with generative capabilities. They have the ability to learn a probability distribution over its set of input. RBMs are a special class of Boltzmann Machines and they are restricted in terms of the connections between the visible and the hidden units. This makes it easy to implement them when compared to Boltzmann Machines. RBMs have the capability to learn latent factors/variables (variables that are not available directly but can be inferred from the available variables) from the input data. RBMs are unsupervised learning algorithms that have the capability to reconstruct input approximations from the data. They do this by trying to produce the probability distribution of the input data with a good approximation which helps in obtaining data points which did not previously exist in our data. They do this by learning a lower-dimensional representation of our data and later try to reconstruct the input using this representation.

Shortcomings of Recommender systems

Regardless of techniques to use, recommender systems have some shortcomings. In this section, we discuss the issues that are common in RS.

1. Cold Start

Recommender systems might suffer from the cold start problem. To generate recommendations a reasonable amount of information is needed. Therefore, there are some situations that lack of data causes RS to not make recommendations or the generated recommendations are poor. Cold start problems could be occurring due to adding a new item, or for a new user when not enough data is available or when launching a new system.

New user: When a user signs up for an account for the first time, the system is unable to suggest personalized recommendations to the user, since; the system has not had any information about the user. Additionally, the user has not rated any items yet. This means that the system is unable to provide accurate recommendations to users. The new user problem exists in both content-based and collaborative approaches.

New item: Adding a new item is when RS is not able to not make recommendations with new items. This is due to lack of enough ratings for the items or insufficient user reviews and feedback. This problem is more obvious in collaborative filtering techniques, specifically, those rely on item to item approach for making recommendations. Therefore, when a new item is added there is no data about that Item.

2. Data Sparsity

Data sparsity is a crucial problem in recommender systems. Data sparsity plays an important role in recommendation systems as their performance efficiency is affected by increased sparsity .Experiments have proven that using matrix factorization techniques such as SVD could reduce the side effects of data sparsity as it tries to bring down the utility matrix into dense smaller matrices with latent factors which represent features in a compact way.

3. Lack of Sufficient Context-Awareness

Even though some context such as location, time, season, date, and so on are taken into consideration to some extent, there are still various factors such as user emotion, mood, and other parameters should be catered since they influence user decisions in reality. User demand might change with the aforementioned context which cannot be satisfied thoroughly via available state of the art recommender systems.

4. Over-specialization

Another issue of recommender system is over-specialization especially in content-based filtering. In such systems, the main objective is to recommend items that highly match with user preferences, however, this could lead to suggesting items that the user already visited or items which are very similar, no cross category items are suggested by content based methods. As over-specialization problems are related with content-based filtering, collaborative filtering techniques can be used in order to eliminate the problem.

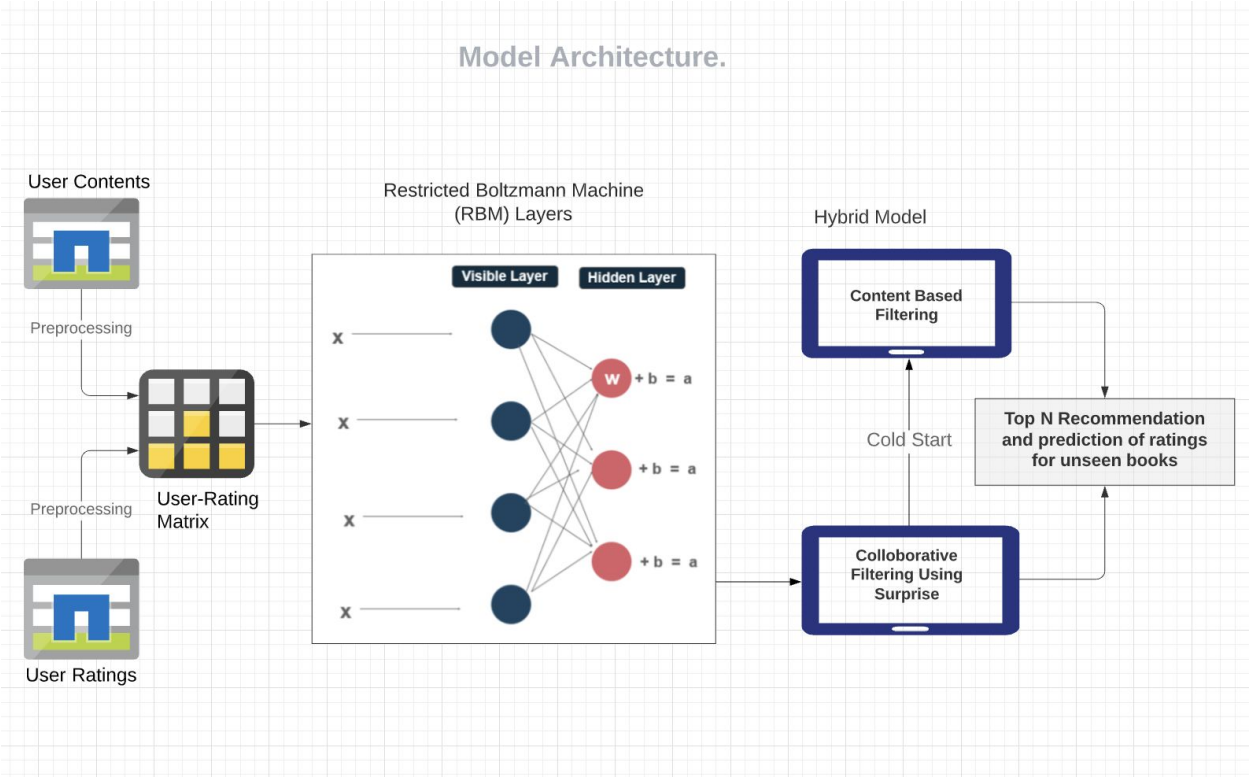
Model Architecture

We suggest an hybrid architecture consisting of deep learning model - Restricted Boltzmann Model, staggered with Collaborative filtering model and content based models (to cater to cold start problem).

Reason we don't directly use collaborative filtering is :

1. Due to the huge number of books, SVD/neighbour based methods take a very long time for computation.

2. And if we directly predict ratings using SVD, mostly all of the initial 100-200 items have a rating of 5. This is the second problem which can be catered if we filter the input to SVD; it's faster as well as recommendations are closer to 'to_read' data as initial input filters out users who haven't read the book, hence neighbour considerations are relevant as compared to initial user neighbours.



Implementation

Content Based Recommendation

Step 1: Load data from csv files of books, ratings, book_tags, tags
Step 2: Merge book_tags and tags dataframe to obtain a dataframe (joined_tags) with book id and all the tag id, the book has been tagged with
Step 3: Merge books and joined_tags dataframe to obtain a dataframe of books with all the tags it has been tagged with in one column named tag_name
Step 4: Now creating a corpus of text features including tag names and author name of each book
Step 5: Calculating TF-IDF using TfidfVectorizer on the corpus of attributes to get predictions
Step 6: Calculating the cosine similarity of tfidf corpus matrix to get the most similar books compared to the given book
Step 7: Recommend top N books according to the similarity values

Collaborative Filtering - Item-Item Based

Step 1: Dropping duplicates and removing the features that cannot contribute to the recommendation like ‘image_url’,’small_image_url’,’isbn’ and ’isbn13’’.
Step 2: Building the item-item similarity matrix.
Step 3: Encoding the data using DictVectorizer.
Step 4: Calculation of the cosine similarity between the item vectors.
Step 5: Recommending the top N books based on the similarity values.

Matrix Factorization using Singular Value Decomposition (CF)

Step 1: Load the books and ratings datasets
Step 2: Use pivot table function to create pivot table where users are the rows and books are the columns with respective rating values (having shape: number of users, number of books)
Step 3: Normalise the matrix by subtracting mean of ratings
Step 4: Compute SVD with numpy.linalg.svd function and use the 50 principal components
Step 5: Calculate the cosine similarity and sort from highest to lowest values of similarity
Step 6: Return the top N books to recommend to user

Restricted Boltzmann : Undirected model

Restricted boltzmann is an undirected deep learning model which belongs to a probabilistic Graphical model. It has an input layer in the form of visible nodes, hidden layers in the form of hidden nodes but no output layer as such and no direction unlike most deep learning models.
The major difference between RBM and Boltzmann MAchine is that visible nodes in RBM isnt interconnected with other visible nodes and hidden nodes are not interconnected with each other, basically : visible nodes connect to one or more hidden nodes and vice versa.

The below is a diagram which shows how input nodes and hidden nodes are connected :

The blue nodes form the visible nodes - input nodes and red ones are hidden nodes.

Hidden nodes can be assumed to be a combination of features which define similar books, which are implicit in nature. Hidden nodes try to reconstruct visible nodes in an iterative way

such that it learns which hidden nodes affect which particular visible node and hence knows when to activate hidden nodes, to reconstruct and reduce errors as the training unfolds. Depending on which hidden nodes are activated and their probability value(if activation function is sigmoid,) we get a fair vote in case of binary outcome like ours.

Lets us see the steps we followed for this:

Step 1: Load the books, ratings datasets into dataframes.
Step 2: Sort the ratings data and remove duplicates ratings.
Step 3: Create an indexing variable which helps us uniquely identify each row after we group by user_id in ratings dataframe
Step 4: Set number of visible as the number of books , number of hidden nodes are dependent on the user and can be fine tuned later according to performance, we took it as 100.
Step 5: Split data into test and train and convert ratings>=3 as 1 (liked) and <3 as not liked.
Step 6: Convert df to torch tensors as we will be using the Pytorch library for this model building.
Step 7: In the RBM class, we write functions such as init for initializing weight and bias of visible and hidden nodes, sample_h and sample_ as functions to get probability that the hidden node will be activated depending on the value of activation value and vice versa for visible nodes too.
Step 8: Next we write the train function which takes input, visible nodes after k sampling, initial probability, probability after k steps.
Step 9: Train function is also where we update and adjust the weights as well as bias for hidden and visible nodes using contrastive divergence algorithms by Gibbs Sampling to adjust weights as undirected connections.(CD-1/3/5 pass)
Step 10: Next, Predict function takes in visible nodes and returns visible nodes after kth chaining, with binary values.
Step 11: This when trained with 10 epochs and batch size 100 in our case and k=10 for gibbs chaining, we get pretty decent results with test loss 0.14.

Drawback : The major drawback is that there is no ranking, rating to choose one item over another, hence we just get a comprehensive list of items that the user has not previously seen/read but either will like or not. Such an output though helps to narrow down results, is not a very good way to recommend items, hence we devise a hybrid approach taking in consideration the above drawbacks and strengths of all algorithms.

Our Hybrid Approach

We experimented with all the above methods, which have their own drawbacks as described, but the combination hybrid model is aimed at catering those to come up with a robust recommendation engine.

Step 1 : The utility rating matrix is converted to a binary format where ratings >3 are considered liked and are replaced by 1. Likewise ratings below 3 are considered disliked and are replaced

by 0. This is given as input to the Restricted Boltzmann model to decipher the books that the user will most likely read and which he might dislike from the set of books that he has not read.

Step 2 : This subset of books that we predict the user will like is given as input to the collaborative filtering model, which now has an utility matrix consisting of these books and users who have rated these books. (A smaller and denser utility matrix).
We experimented with K- Neighbour based collaborative filtering which is computationally very expensive Singular Dimensionality Reduction methods. SVD performed better in approximating ratings as compared to KNN- Baseline based algorithms which broke the kernel with complete data and less number of k as well.. After predicting ratings, we suggest top 10 items which were rated highest to the user.

Step 3: The scenarios where there is a lot of variance in predicted ratings, for instance 5 followed by an item with rating 3, and some scenarios where either the user is new (user has only 1-2 ratings we assume it to be cold start problem) where we use content based recommendations by suggesting the user books, similar to the 1-2 books that he has rated.

Eventually, we recommend a user a total of 15 books which he is most likely to enjoy reading.

Results from model:
Restricted Boltzmann Machine, trained for 10 epocs, has a train loss of 0.147 and test_loss of 0.1464.

Results from final model

Let's see what the hybrid models recommends for: **User Id : 5**

Some of the Books that the user has read in past :

1. 6703 - Hidden Fire -- Fantasy Fiction/ Mystery (According to Google)
2. 6646 - Silent Scream -- Fiction, Mystery, Thriller (According to Google)
3. 8072 - Wait till helen comes -- Fiction, Ghost Thriller (According to Google)

Books Recommended by Restricted boltzmann Machine are : 1,2,4,21,22,23,24,25,and many more (Total 922 out of 10,000) which are : mostly fiction, thriller, horror when checked manually, test loss was 0.146 meaning not the rmse but *87%* of successful predictions.

Top *15* recommendations after SVD are : (Categories of genres are cross checked with google) with Ratings ranging from *5 to 4.93*.

Recommendations	**Genres**

Transfer of Power	Thriller, Suspense, Political thriller, Political fiction
White Cat	Fantasy Fiction
Killing Patton	Mystery, Biography
The firm	Novel, Fiction, Mystery, Legal thriller, Legal Story
Mortal engines	Science Fiction
Waiting to exhale	Female Fiction
Post birthday world	Novel, Domestic Fiction
Other side of dawn	Drama, Fiction, war story
Billy buddy	Adventure fiction, Nautical fiction
Twilight watch	Fantasy Fiction, Young adult fiction
Honorable schoolboy	Novel, Thriller, Spy fiction
La tregua	Drama
Tennis shoes among Nephites	Fiction
The jester	Mystery, Thriller, Suspense, Historical Fiction
Lions of AL Rassan	Novel, Historical Fiction, Fantasy Fiction, Historical fantasy

	bookids	predictedrating	Title
0	2920	5.000000	[Transfer of Power (Mitch Rapp, #3)]
1	5345	5.000000	[White Cat (Curse Workers, #1)]
2	6231	5.000000	[Killing Patton: The Strange Death of World Wa...
3	7868	5.000000	[The Firm]
4	8643	5.000000	[Mortal Engines (The Hungry City Chronicles, #1)]
5	8779	5.000000	[Waiting to Exhale (Waiting To Exhale #1)]
6	9188	5.000000	[The Post-Birthday World]
7	9843	5.000000	[The Other Side of Dawn (Tomorrow, #7)]
8	9995	5.000000	[Billy Budd, Sailor]
9	7817	4.978427	[Twilight Watch (Watch #3)]
10	9233	4.976775	[The Honourable Schoolboy]
11	7913	4.967750	[La tregua]
12	9137	4.956284	[Tennis Shoes Among the Nephites (Tennis Shoes...
13	6728	4.936156	[The Jester]
14	8087	4.930667	[The Lions of Al-Rassan]

Now, as the user has read less than or equal to 5 books, we consider him to be a new user and would also generate content based recommendations, below are results for user 5.

```
Books similar to Id                                  Recommended Books
               6703                The Soulkeepers (The Soulkeepers, #1)
               6703                  The Mind Readers (Mind Readers, #1)
               6703                                    Long Time Coming
               6646     The Girl In The Ice (Detective Erika Foster, #1)
               6646                          Eeny Meeny (Helen Grace, #1)
               6646                            Birdman (Jack Caffery, #1)
               8072          Deep and Dark and Dangerous (A Ghost Story)
               8072  Scary Stories to Tell in the Dark (Scary Stori...
               8072                Harriet the Spy (Harriet the Spy #1)
               7487                            Fractured (Will Trent, #2)
               7487                               Broken (Will Trent, #4)
               7487                               Undone (Will Trent, #3)
               4829                                     The Good Neighbor
               4829                                              Wreckage
               4829                  Silent Scream (D.I. Kim Stone, #1)
```

Almost all are drama, fiction,horror/thriller related which the user has seen previously which validates our content based result too. Also we can see one of the results is Silent Scream but a different version which is in top recommendations.

FInally, we can suggest a mixed book results for users who have read less than 5 books, that is 10 from Collaborative filtering and 5 from content based results as final Suggestions.

Evaluation metrics

Testing methodology and Accuracy Testing

The metrics used for Restricted Boltzmann was the train loss and test loss discussed above, checked for 10 epochs with batch size of 100.

Metrics used for SVD was test rmse on 70 percent holout set, along with fit time, which was :

Algorithm	test_rmse	fit_time	test_time
SVD	0.841428	24.430082	3.197908

3 cross folds were used to improve performance along with experimentation with 50,100 factors. On the training set, when tested by creating an anti test set rmse was 0.77.

In such scenarios, rmse don't usually make sense a lot of times as high predicted rating doesn't necessarily mean the user will read the book as parameters like human mood, interest, time of the day, how occupied is the user with work etc affect decisions implicitly.

Hence we did a manual check for some users and found results were pretty similar to genres (which were not fed to the model/ fetched for google) which validates the results. A sample for user 5 is shown above.

Conclusion

Results and Future Work

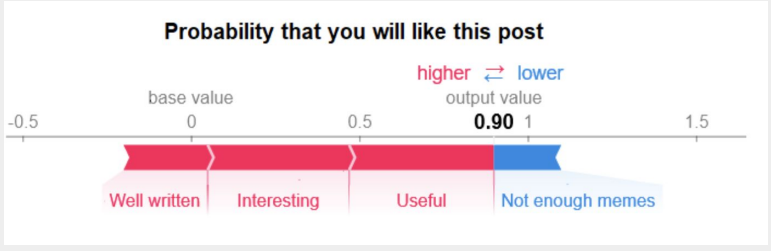
Restricted Boltzmann machines even after being probabilistic graph models are successful in building a recommender system.The RBM recommender system can learn the probability distribution of ratings of items for users given their previous inclination. The the RBM recommender system uses the learned probability distribution and activation of hidden nodes to to predict ratings on never-before-seen items. RBM based recommendation systems can only predict binary outcomes in simple machines. For example, in our dataset our model was predicting the value between (0,1) in which 0 suggested disliked book and 1 suggested liked.

Now as this isn't sufficient to recommend a small subset to the user, there is a need to map and rank values predicted as 1, probable of liking and suggest a much smaller subset to the user.

The hybrid model of content based and collaborative based filtering has been used to evaluate the ratings in the original forms. For the user having Cold start problem, our model can evaluate using Content based filtering and find similar users based on the predicted values generated by the RBM. The recommended user's original rating will be retrieved based on user_id. Based on the given best ratings from the users , our model will recommend those movies which users have not seen yet. Similarly , Collaborating filtering will be used to recommend the movies if there is sufficient data available for the user.

However, RBM as it is a deep learning model and depends on graph reconstruction,it is not an explainable model and unable to explain the reason for predicting those ratings and their recommendations.

There are many libraries available that can trace the computation performed in the deep learning model. We have tried **SHAP** python library to visualize the results which can give us the reasoning analysis for our prediction just like the below pic which has been generated for a classification problem if the user will like the post or not.



Unfortunately, **SHAP doesn't support Pytorch as yet,** as change was rolled back sometime ago and hence we couldn't get the explanations for our RBM model in our case.

This opens up a good future possible directions for us to further explore - Descriptive analysis of the predictions made by our model. As this can explain many unusual predictions made by a model.

There are many types of deep learning based recommendation systems like Perceptron based systems, Autoencoders, Convolutional Neural Networks which can be developed and compare their performance with our model. The possibilities are endless and a lot can be tweaked within this ever increasing domain of personalization.

References

1. Seyednezhad, s. M. Mahdi & Cozart, Kailey & Bowllan, John & Smith, Anthony. (2018). A Review on Recommendation Systems: Context-aware to Social-based.
2. Salakhutdinov, Ruslan & Mnih, Andriy & Hinton, Geoffrey. (2007). Restricted Boltzmann machines for collaborative filtering. ACM International Conference Proceeding Series. 227. 791-798. 10.1145/1273496.1273596.
3. Madadipouya, Kasra, AND Chelliah, Sivananthan. "A Literature Review on Recommender Systems Algorithms, Techniques and Evaluations" BRAIN. Broad Research in Artificial Intelligence and Neuroscience [Online], Volume 8 Number 2 (24 July 2017)
4. Restricted Boltzmann Machine Tutorial: Deep Learning Concepts https://www.edureka.co/blog/restricted-boltzmann-machine-tutorial/
5. Welcome To Pytorch Tutorials https://pytorch.org/tutorials/

Data and Libraries:

1. PyTorch https://pytorch.org/
2. SciKitLearn, Pandas, Numpy
3. Dataset: https://github.com/zygmuntz/goodbooks-10k