Book Recommendation Engine

Team: Goal Diggers

Slide 1:

* Since we were not getting all the important features necessary to recommend a book(like authors, genre etc.) in any dataset, we decided to enhance the GoodReads book dataset by writing a script.
* We first cleaned the data by removing punctuations and unnecessary characters in various fields (like book title). This was done to ensure that analysing the dataset becomes less cumbersome and the values are more consistent with each other.
* The dataset did not have book genre as an attribute. So, the script added an entire column consisting of the genres of each book. The genres were extracted from the wikipedia links which are a part of the dataset. Every book has its genre mentioned explicitly in the table at the top of its wikipedia page.
* We reduced the number of genres to 27 by taking only the common genres and removing the rare ones.
* Following this, we formed a book title vs genre matrix (with dimensions 9626 x 27). A book can have multiple genres. So, a book will have 1 in the ith column if and only if the book has the ith genre as one of its genres.

Slide 2:

* The number of genres were reduced to 27 by choosing the most common genres in the previous step. For better analysis, we extracted 15 latent features out of the genres available.
* The 15 genres were decided so that the variance of the matrix is maximised and keeping in mind that every genre is orthogonal to all the other genres.
* We used Principal Component Analysis (PCA) library of python for doing so. The library uses Singular Value Decomposition(SVD) for finding out the latent features.
* PCA is used for dimensionality reduction.

Slide 3:

* To ensure novelty, we have used the epsilon greedy approach.
* If ϵ be the probability of exploration, the recommendation engine exploits the user preferences with a probability of (1-ϵ) and explores new genres and artists with a probability ϵ.

Slide 4:

* This slide is about changes in the exploration rate as time passes.
* At first, the recommendation engine has very little knowledge about the user’s taste and hence, there is a high exploration rate.
* As the learning process goes on, the exploration rate decreases and the engine exploits the user’s preferences more and more.
* As the process continues, the exploration value(ϵ) reduces to a certain minimum value after which it is kept constant.
* Once the books number of books that have not been suggested to the user and are according to the preferences of the user, falls below a threshold(say 20%), the exploration rate starts increasing slowly to ensure that the engine does not exhaust the data.

Slide 5:

* In this slide, the hyperparameters used in the model have been defined.

Slide 6:

* This slide is about the technique used to update the user vector. It also explains the formula used to calculate the similarity between a user vector and a book vector.
* The user vector gets updated according to the actions he chooses for the recommended books - Selected, Like, Dislike.
* To calculate the similarity, we use the cosine similarity, ensuring that the similarity doesn’t get scaled up in the beginning as the system is simply exploring.
* We also add the Upper Confidence Bound (UCB) of an item while calculating the similarity, which ensures that newer items get more chance of getting recommended than the items that have already been recommended.
* The items having maximum similarities with the user vector are recommended.