Recommender Systems

Krut Patel – 2017A7PS0184H
P Yedhu Tilak – 2017A7PS0021H
Akhil Agrawal – 2017A7PS0190H
Smit D Sheth – 2017A7PS1666H
Team: 26

Introduction

This application can predict ratings a user might give to a particular movie based on the previous history of ratings given by various users for various movies, and thus it can recommend new movies to the user. Ratings are given as a whole number in the range of 1 to 5.
Several techniques were used to achieve this-

a. Collaborative filtering
b. SVD
c. CUR
d. Latent factor modeling

along with enhancements. For each method used, RMSE and MAE were calculated using a validation/test set of ratings.

Dataset

In order to construct our Utility Matrix along with a test set of ratings, a dataset of 1,000,000 ratings from 6000 users on 3900 movies was taken from MovieLens. Every user has rated at least 20 movies hence we have a decent amount of history for each user. 80% of the 1 Million ratings were used to construct our Utility Matrix which then was used to train our models and help in predictions of future ratings. The remaining 20% of the ratings were used as a test set to evaluate the performance of our models.

Architecture

Each of the techniques used to build a recommender system resides in its own Python module. In addition, since the Dataset and Utility Matrix is required by each of these, they have been separated into their own modules, thus following Don't Repeat Yourself principle. We use the pandas library to handle the dataset, and sklearn to create the test and train datasets.

Techniques

Collaborative Filtering

In our first approach, collaborative filtering was used to predict user ratings; to be more specific item-item collaborative filtering was used. Item-based collaborative filtering is used, as a similarity between items is more accurate and meaningful than the similarity between users. To account for strict and lenient users, each row was subtracted by it's mean so as to make each row mean-centered at 0. To predict the rating of a particular movie given by a particular user, 15 movies that are most similar to the given movie and which are rated the user are used.

Collaborative filtering with baseline estimates

This approach was an enhanced version of the approach specified above. Here global and regional baseline estimates were also used as a part of computations. Strict and lenient raters were automatically handled as the baseline estimate accounts for the user's and the movie's mean ratings. The addition of these baseline estimates improved our predictions by more than 10%.

SVD Decomposition

Singular Value Decomposition is a factorization method to decompose or factorize a real valued matrix.
SVD factorizes a matrix M into U, Sigma and V* (V-transpose).
Columns of U represent the left-singular vectors of the matrix M.
Sigma is an m X n rectangular diagonal matrix, with each diagonal element as the singular values of the utility matrix.
Columns of V represent the right-singular vectors of the matrix M.
Dimensionality reduction is performed on each of the 3 matrices to bring the vectors to a lower-order dimension space.
The dot product U.Sigma.V* will generate a matrix which is a close approximation of the matrix M.

CUR Decomposition

CUR matrix decomposition is a low-rank matrix decomposition algorithm that uses a lesser number of columns and rows than the data matrix.This number is represented by the variable k.In our data, we have taken k=1000. The rows and columns are selected randomly based on their probability distributions. The probability distribution is based on a statistical leverage score which represents the row/column importance. Matrix C consists of the randomly picked columns and matrix R consists of the randomly selected rows.The intersection of R and C gives us the intermediate matrix W. On W, SVD is applied and U is obtained. Finally, the product CUR gives us approximations.

Latent Factor Model

The Latent Factor Model tries to map both users and movies to some number of hidden “concepts” or “factors”, and then use their similarity to make predictions. After accounting for strict and lenient users, it becomes an optimization problem parameterized by the baseline ratings of users and movies, and the Q and P matrices, which denote

the relation of features to items and users respectively. This problem can be solved using gradient descent to learn the parameters by minimizing the sum of RMSE and the L_2 regularization to prevent biases and coordinates in latent space from becoming too huge.

Results

Recommender System Technique	**Root Mean Squared Error (RMSE)**	**Mean Average Error (MAE)**	**Time taken for prediction**
Collaborative filtering	0.977	0.697	0.43s
Collaborative filtering with baseline	0.871	0.674	0.41s
SVD	0.21	0.0534	0.001s
SVD with 90% energy	0.211	0.0509	0.001s
CUR	0.231	0.044	0.001s
CUR with 90% energy	0.275	0.049	0.001s
Latent factor model (f=10)	0.877	0.690	0.37s
Latent factor model (f=40)	0.871	0.684	0.39s
Latent factor model (f=100)	0.876	0.687	0.42s
Latent factor model (f=1000)	0.892	0.702	0.49s