BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)

HYDERABAD CAMPUS

(November 21, 2020)

Design Document

Recommender System

SUBMITTED BY

ANUSHA AGARWAL 2018A3PS0032H

JUI PRADHAN 2018B3A70984H

KRITI JETHLIA 2018A7PS0223H

Under the supervision of

Prof. Aruna Malapati

1. Problem Statement

This project aims to analyse the results of the following recommender systems which are used to predict unknown ratings of movies.

1. Collaborative Filtering
2. Collaborative Filtering using Baseline approach.
3. Singular Value decomposition.
4. SVD with 90% energy.
5. CUR.
6. CUR with 90% energy.

2. Algorithm

Pre-processing:

The three dataset files users.dat, ratings.dat and movies.dat have been taken input as pandas dataframes. Using these dataframes, a new dataframe utility_matrix has been built. It contains userIds as rows, movieIds as columns and their ratings at their respective positions in the dataframe.

This matrix uses the userId, movieId and their ratings present in ratings.dat file to fill the rating values in the utility_matrix.

The data set is split into training and test set.training set is used to decompose into respective matrices and the reconstructed matrix is used to predict the values of the test set.

1. Collaborative Filtering:

Collaborative filtering offers a way of predicting the value(to be given) of an entity for an item based on the value given by other similar entities to the corresponding item.

The baseline approach has been implemented directly by applying the original formula:

baseline = (bias-movie) + (bias-user) + all_mean

= (movie_mean - all_mean) + (user_mean - all_mean) + all_mean

= movie_mean + user_mean + all_mean

2. Singular Value decomposition:

The goal of SVD is to decompose M as the product of three matrices U_{mxm}, S_{mxn} and V^T_{nxm} and reduce the re construction error.

The user ratings are predicted and error is calculated

3. **CUR:**

Aim of CUR decomposition is to reduce the density of SVD decomposition without having much loss of data. CUR decomposition aims to decompose the given matrix M into 3 matrices C, U, R, where, C is a matrix consisting of randomly selected columns and R is matrix consisting of randomly selected rows.

Number of rows selected is approximately equal to the rank of the utility matrix.

$$U = Y(\Sigma^{+})^{2}X^{\mathrm{T}}$$

where Σ^{+} is the pseudoinverse of the matrix formed by the intersection of C and R.

The original matrix is constructed by multiplying the C,U,R matrix

1. Data structured used

- **Sets-** A set is an unordered and unindexed data structure. You cannot access its elements directly, a loop is required.
- **Lists-** A list is a collection of ordered and mutable data. They are denoted by []. An item in a list can be accessed by index.
- **Arrays**- We have used a numpy array to implement our signature matrix. Numpy arrays are 50 times more efficient than python lists because they are stored at a continuous place in memory unlike lists, so processes can access and manipulate them very efficiently.
- **Dictionaries -** Dictionaries are used to store key and value pairs.

3. Results:

Recommender System Technique	Rmse	Precision top K	Spearman rank correlation	Time taken
Collaborative	1.0377	0.8515	0.99	162.20 secs
Collaborative with Baseline approach	1.51	0.36	0.9795	636.89 secs
SVD	1.1219	1.0	0.99	118.04 secs
SVD with 90% retained energy	1.129	1.0	0.99	109.5 secs
CUR	1.586	0.8806	0.99	60.37 secs
CUR with 90% retained energy	1.943	1.0	0.99	58.86 secs

4. Assumptions:

- For Precision at top k we assume k to be 5000
- The number of columns and rows for CUR calculations is 1000

5. Pros and Cons:

1. Collaborative Filtering :

Pros:

- No domain knowledge is necessary for collaborative filtering.
- The model can even help us discover new interests for users. In isolation, the system may not know the user is interested in a given item, but the model might still recommend it because similar users are interested in that item. In particular, the system doesn't need contextual features.

Cons :

- Collaborative filtering cannot handle fresh items.
- If an item is not seen during training, the system can't create an embedding for it and can't query the model with this item. This issue is often called the cold-start problem.
- Sparsity Problem - It is hard to find users who rated the same movies
- Popularity Bias - Cannot recommend items to someone with unique taste, tends to recommend popular items
- First rater problem
- Generally we are interested in only the higher values and not the lower rated values while recommending , however collaborative filtering approach considers both of them.

2. Singular Value decomposition :

Pros:

- Simplifies data
- removes noise
- may improve algorithm results.
- SVD is useful when there are a small number of concepts that connect the rows and columns of the original matrix.

Cons:

- Interpretability problem: Transformed data may be difficult to understand.
- Computational cost is cubic time in the size of data to compute.
- The results are dense vectors hence could take lots of space.

3. CUR :

Pros:

- Easy interpretation: Since the basis vectors are actual columns and rows
- Sparse basis: Since the basis vectors are actual columns and rows

Cons:

- Duplicate columns and rows: Columns of large norms will be sampled many times