Birla Institute of Technology and Science-Pilani, Hyderabad Campus

First Semester 2018-2019

Information Retrieval (CS F469)

Design Document – Recommender Systems

by

KULKARNI MOHITH 2016A7PS0109H

ASVS KARTHEEK AKELLA 2016A7PS0015H

Under the guidance of

Dr. Aruna Malapati

Recommender System

30th October 2018

OVERVIEW

The vast amount of data available on the Internet has led to the development of recommendation systems. This project and report addresses the limitations of current algorithms used to implement recommendation systems, evaluation of experimental results, and conclusion. This report provides a detailed summary of the project

GOALS

1. Collaborative Filtering
2. Collaborative Filtering with baseline approach.
3. Singular Value Decomposition
4. Singular Value Decomposition with 90% retained Energy
5. CUR Decomposition

Collaborative Filtering

ASSUMPTIONS

- We can predict the rating of item i by user x through set of other users whose ratings are "similar" to x's ratings or by set of other items whose ratings are "similar" to i's ratings.

FORMULATION

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

s_{ij}… similarity of items i and j
r_{xj}…rating of user u on item j
$N(i;x)$… set items rated by x similar to i

R_{xi} :Rating of item i by user x

PROS and CONS

Pros

- No feature selection needed.Recommendation are based on user-user or item-item similarity.
- Takes care of strict and lenient raters.

Cons

- Cold Start : Need enough users in the system to find a match
- Sparsity : The user/ratings matrix is sparse so it is hard to find users that have rated the same items
- First rater : Cannot recommend an item that has not been previously rated and have a hard time rating new items, Esoteric items
- Popularity bias : It tends to recommend popular items so it cannot recommend items to someone with unique taste .

Collaborative Filtering with baseline

ASSUMPTIONS

- We can predict the rating of item i by user x through set of other users whose ratings are "similar" to x's ratings or by set of other items whose ratings are "similar" to i's ratings.
- We also need to consider the deviation of that item i and user x from over-all mean of corpus.

Formulation

$$r_{xi} = b_{xi} + \frac{\sum_{j \in N(i;x)} s_{ij} \cdot (r_{xj} - b_{xj})}{\sum_{j \in N(i;x)} s_{ij}}$$

baseline estimate for r_{xi}

$$b_{xi} = \mu + b_x + b_i$$

- μ = overall mean movie rating
- b_x = rating deviation of user x = (*avg. rating of user* x) $- \mu$
- b_i = rating deviation of movie i

PROS and CONS

Pros

- No feature selection needed. Recommendation are based on user-user or item-item similarity.
- Takes care of strict and lenient raters.
- Demographics to deal with new user problem
- For any negative or zero predictions we are replacing it with its baseline estimator.

Cons

- Sparsity : The user/ratings matrix is sparse so it is hard to find users that have rated the same items
- First rater : Cannot recommend an item that has not been previously rated and have a hard time rating new items, Esoteric items
- Popularity bias : It tends to recommend popular items so it cannot recommend items to someone with unique taste .

Singular Value Decomposition

ASSUMPTIONS

- Matrix Decomposition are perfect i.e. there are no error due to floating point computation.
- Retaining 90% of energy allows us to retain enough information to approximately reconstruct the original matrix.

FORMULATION

$$\mathbf{A}_{[m \times n]} = \mathbf{U}_{[m \times r]} \Sigma_{[r \times r]} (\mathbf{V}_{[n \times r]})^T$$

A: **Input data matrix**

– m x n matrix (e.g., m users, n movies)

U: **Left singular vectors**

– m x r matrix (m users, r concepts)

Σ: **Singular values**

– r x r diagonal matrix (strength of each 'concept')
(r : rank of the matrix **A**)

V: **Right singular vectors**

– n x r matrix (n movies, r concepts)

$$\mathbf{A} \approx \mathbf{U}\Sigma\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^T$$

σ_i ... scalar
$\mathbf{u}_i$ ... vector
$\mathbf{v}_i$ ... vector

rows of $\mathbf{V}^t$ are eigenvectors of $\mathbf{D}^t\mathbf{D}$ = basis functions

Σ is diagonal, with δ_{ii} = **sqrt(λ_i)** (ith eigenvalue)

rows of **U** are coefficients for basis functions in **V**

(here we assumed that **m > n**, and **rank(D) = n**)

PROS and CONS

Pros

- Discover hidden correlations /topics : Correlation in two dimensions in 2 matrix gives us less information
- Remove redundant and noisy features : Features that are highly correlated to some other feature or feature that are giving noisy data can be removed by reducing the dimensions of sigma matrix.
- Interpretation and visualization : The new space between user and movie can give much more insight than previous space.
- Optical low-rank approximation in terms of Frobenius norm.
- Easier Storage and processing of the data.

Cons

- Interpretability Problem : A singular vector specifies a linear combination of all input columns or rows.
- Lack of Sparsity : Singular Matrix are dense.

CUR Decomposition

ASSUMPTIONS

- We can approximately reconstruct the original matrix multiplying selected columns matrix and selected rows matrix and Pseudo inverse of SVD of intersection of the former two matrix.

- We need to eliminate last 1% the eigen values to avoid predicted values beyond our range.

FORMULATION

$$\|A\text{-}CUR\|_F \leq \|A\text{-}A_k\|_F + \varepsilon\|A\|_F$$

Where A is Original Matrix CUR is Matrix obtained by CUR multiplication A $_k$ is matrix obtained by retaining k dimensions

$$P(x) = \sum_i \mathbf{A}(i,x)^2 / \boxed{\sum_{i,j} \mathbf{A}(i,j)^2}$$

Here the selected block represent the Frobenius norm of entire matrix

$$\mathbf{C}_d(:,i) = \mathbf{A}(:,j)/\sqrt{cP(j)}$$

$$W = X\,Z\,Y^T$$

$$U = W^+ = Y\,Z^+\,X^T$$

PROS and CONS

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.

Packages/Library Used

Every module is coded in python. Following packages or library are used-

1. Numpy
2. Scipy CSR Sparse Matrix
3. Linalg to compute eigen value decomposition
4. Math
5. csv to read input data