Comparision of SVD and CUR Decomposition

Design Document

Academic Project for the Course CS F469 Information Retrieval

Name of the Student	**ID Number**
Aniketh Janardhan Reddy	2014A7PS096H
Kartik Sethi	2014A7PS130H
Monica Adusumilli	2014A7PS005H
Vinay Datta Renigunta	2014A7PS509H

BITS Pilani Hyderabad Campus

ABSTRACT

Recommender systems are a subclass of information filtering system that seek to predict the "rating" or "preference" that a user would give to an item. Recommender systems have become extremely common in recent years, and are utilized in a variety of areas. There are recommender system applications for online purchases, books, jokes, social networking, garments, collaborators etc. In a recommender system application there are two classes of entities, users and items. Users have preferences for certain items, and these preferences must be teased out of the data. The data is represented as a utility matrix, giving each user-item pair, a value that represents what is known about the degree of preference of that user for that item. The utility matrix is usually sparse as typically the user would rate only a few items in a large database of items. The goal of the recommender system would be to predict the blanks in the utility matrix. This would make storing and comparing users extremely slow and memory intensive. Dimensionality is an approach through which the dimension of the utility matrix is reduced and this would improve the performance of the recommender system. In this project, two techniques, Singular Value Decomposition and CUR Decomposition are explored and compared on a dataset.

Singular Value Decomposition

Singular Value Decomposition is a factorization of a real or complex matrix. It is the generalization of the eigendecomposition of a positive semidefinite normal matrix to any m x n matrix via decomposition.

Theorem: Suppose M is a m × n matrix whose entries come from the field K, which is either the field of real numbers or the field of complex numbers. Then there exists a unique factorization, called a singular value decomposition of M, of the form

where $$\mathbf{M} = \mathbf{U\Sigma V}^*$$

- U is a m × r, unitary matrix,
- Σ is a diagonal r × r matrix with non-negative real numbers on the diagonal, and

•$V*$ is a $r \times n$, unitary matrix over K

Computing the SVD of a matrix

The SVD of a matrix M is strongly connected to the eigenvalues of the symmetric matrices $M^T M$ and $M M^T$. To begin the explanation, start with $M = U \Sigma V^T$, the expression for the SVD of M. Then

$$\mathbf{M^T = (U \Sigma V^T)^T = (V^T)^T \Sigma^T U^T = V \Sigma^T U^T}$$

Since Σ is a diagonal matrix, transposing it has no effect. Thus,
$\mathbf{M^T = V \Sigma U^T}$.

Now, $\mathbf{M^T M = V \Sigma U^T U \Sigma V^T}$
Since U an orthonormal matrix, $U^T U$ is a the identity matrix. Thus
$\mathbf{M^T M = V \Sigma^2 V^T}$
Finally, $\mathbf{M^T M V = V \Sigma^2}$
V is the matrix of eigenvectors of $M^T M$ and Σ^2 is the diagonal matrix whose entries are the corresponding eigenvalues.

Similarly, U is the matrix of eigenvectors of MM^T.

The Eigen Vectors and the Eigen values are calculated using the method of power iteration. Thus, the matrices, U, Σ and V^T are computed.

SVD Drawbacks

In large-data applications, it is normal for the matrix M being decomposed to be very sparse; that is, most entries are 0. For example, a matrix representing many documents (as rows) and the words they contain (as columns) will be sparse, because most words are not present in most documents.
We cannot deal with dense matrices that have millions or billions of rows and/or columns. However, with SVD, even if M is sparse, U and V will be dense. Since Σ is diagonal, it will be sparse, but Σ is usually much smaller than U and V, so its sparseness does not help.

CUR Decomposition

The merit of this approach lies in the fact that if M is sparse, then the two large matrices (called C and R for "columns" and "rows") analogous to U and V in SVD are also sparse. Only the matrix in the middle (analogous to Σ in SVD) is dense, but this matrix is small so the density does not hurt too much.

Computing the CUR of a matrix

Let M be a matrix of m rows and n columns. Pick a target number of "concepts" r to be used in the decomposition. A CUR-decomposition of M is a randomly chosen set of r columns of M , which form the m × r matrix C, and a randomly chosen set of r rows of M , which form the r × n matrix R. There is also an r × r matrix U that is constructed from C and R as follows:

- Let W be the r × r matrix that is the intersection of the chosen columns of C and the chosen rows of R. That is, the element in row i and column j of W is the element of M whose column is the jth column of C and whose row is the ith row of R.
- Compute the SVD of W ; say $W = X\Sigma Y^T$.
- Compute Σ^+ , the Moore-Penrose pseudoinverse of the diagonal matrix Σ. That is, if the ith diagonal element of Σ is σ not equal to 0, then replace it by 1/σ. But if the ith element is 0, leave it as 0.
- Let $U = Y (\Sigma^+)^2 X^T$.

Case Study using Jester Dataset

A Jester database in xls format is taken and is used to compare the performance of SVD and CUR decompositions. **"Python"** language is used to implement both the algorithms. Numpy, Scipy, Pandas and Random libraries are used. As it is a sparse matrix, it is converted into a compressed sparse matrix using scipy package. The matrices U, Σ and V are then computed using SVD decomposition as described above. CUR decomposition of the matrix is also done using the method mentioned in the previous section. The reconstruction error, space consumed and the time taken for different sizes of the number of rows have been computed in both the cases and graphs are plotted. The three graphs are shown below. The blue plot represents SVD and the red plot represents CUR

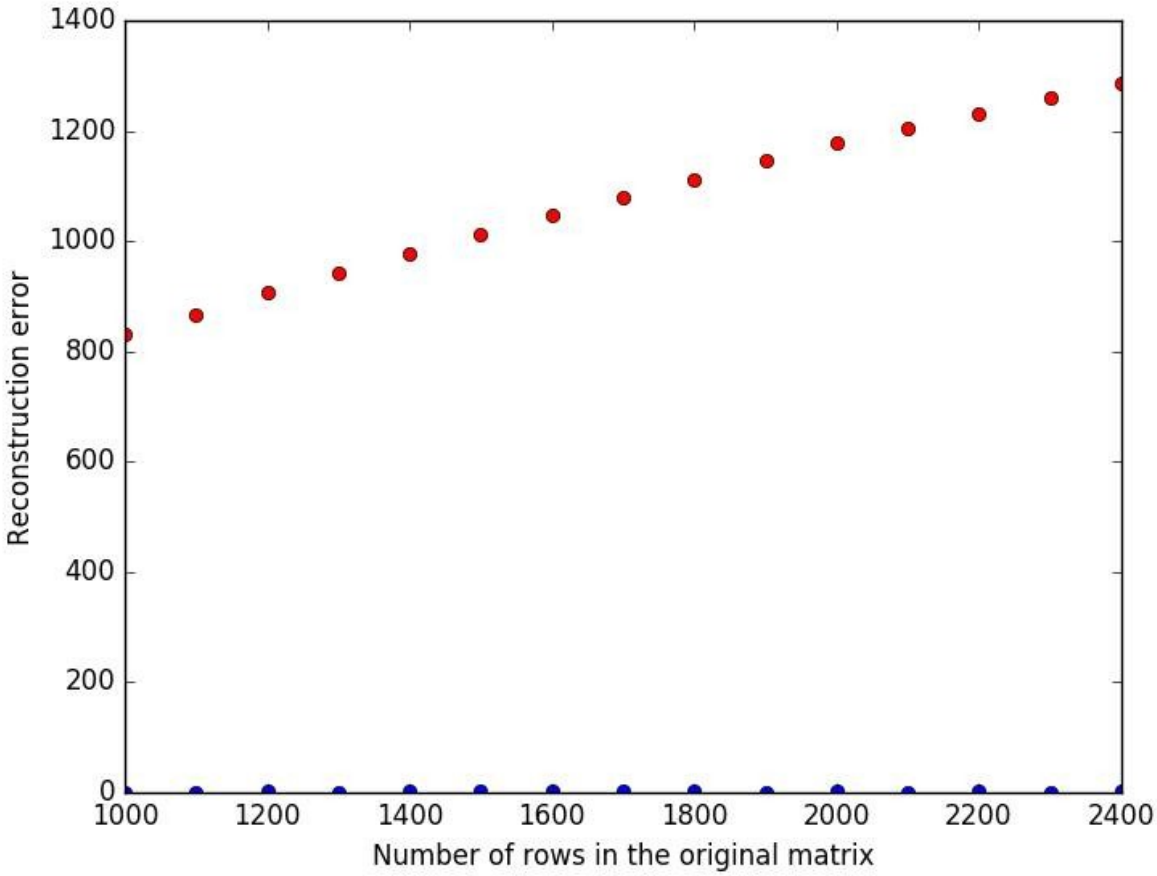
decomposition.

Analysis and Results

The implementation details are as follows:

Number of rows in the input matrix=5000
Number of columns in the input matrix=100
Number of rows sampled by CUR=1000
Number of columns sampled by CUR=75
Number of eigen vectors taken into consideration in SVD=98

Figure 1

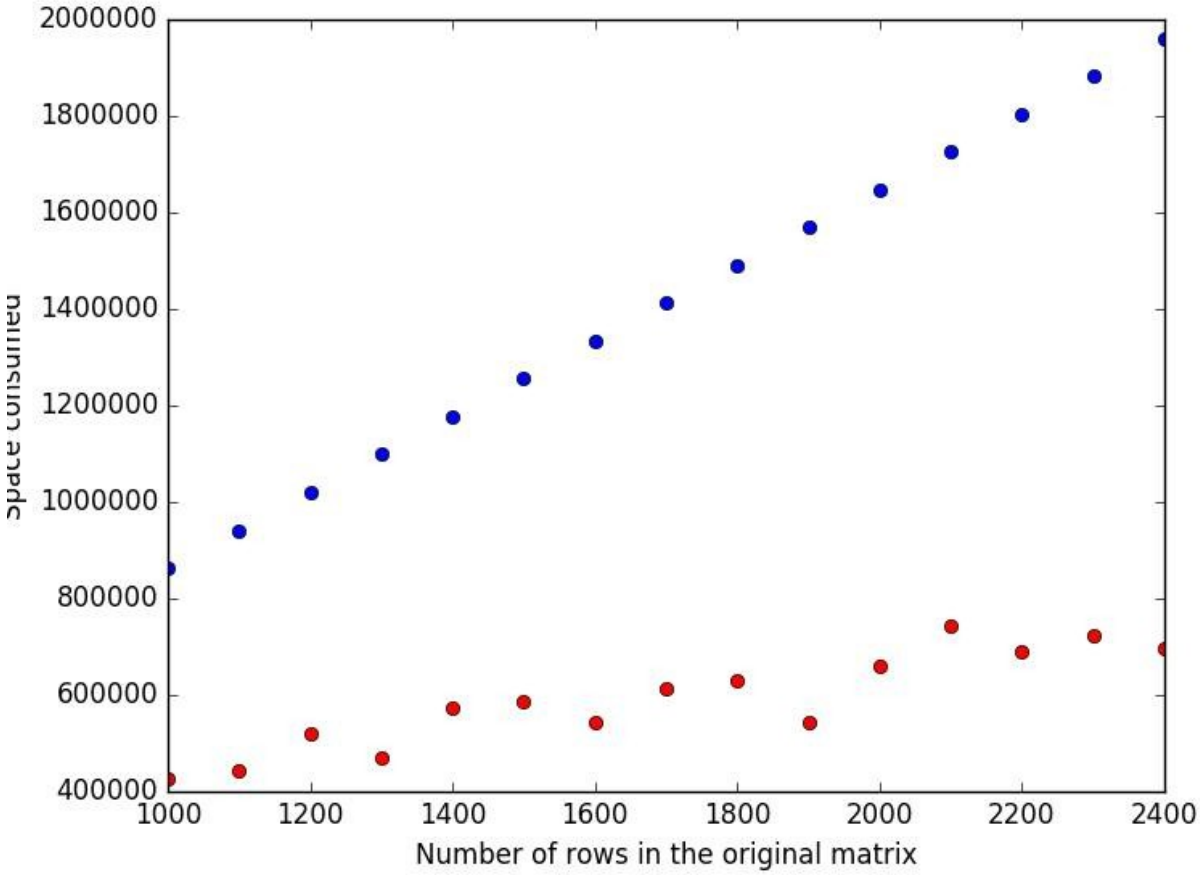


Blue dots- SVD
Red dots-CUR

From the above graph it is clear that the reconstruction error in the case of CUR is much higher when compared to the reconstruction error in the case

of SVD.

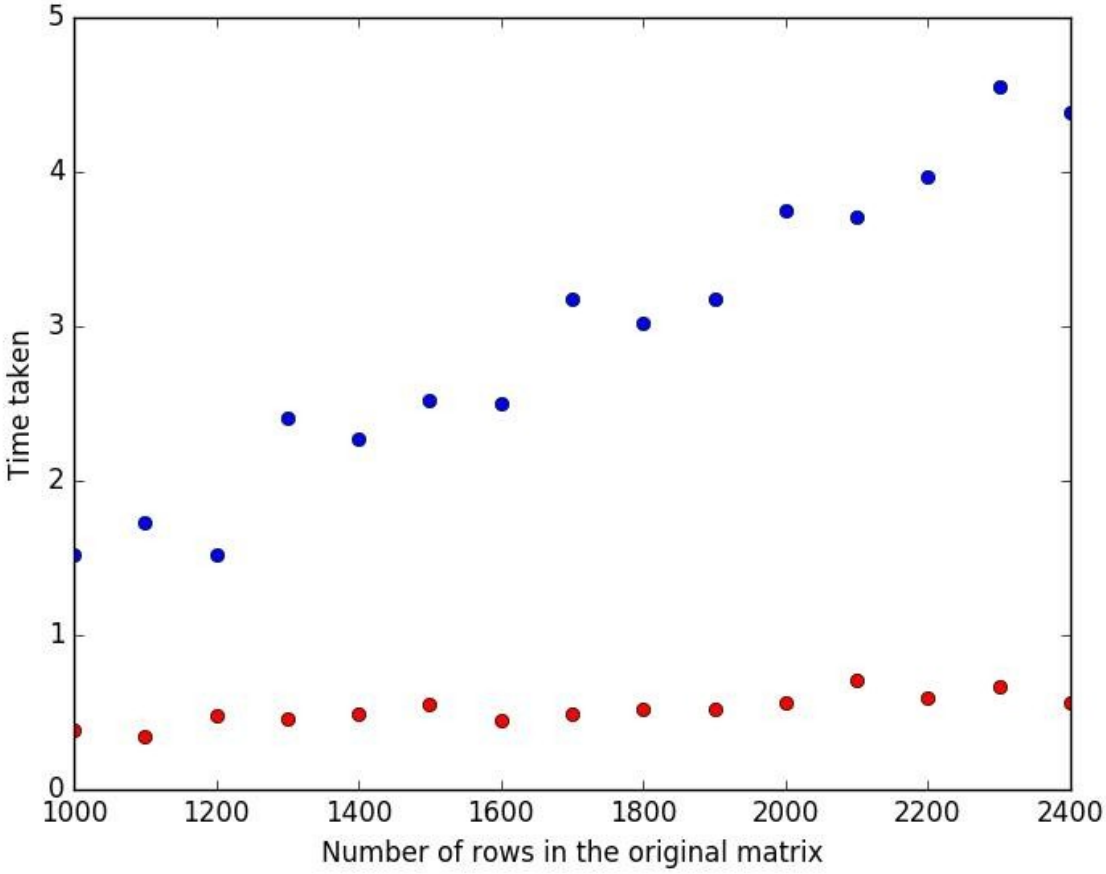
Figure 2



Blue dots- SVD
Red dots-CUR

From the above graph it is observed that the space consumed in the case of CUR is very low while the space consumed by SVD increases as the number of rows in the original matrix increases.

Figure 3



Blue dots- SVD
Red dots-CUR

From the graph, it is observed that the time taken to calculate the decomposition matrices in SVD is higher than the time taken to calculate the decomposition matrices in CUR.

Conclusion

It is evident from the graphs that SVD has a lesser reconstruction error than CUR but the time taken and space occupied by SVD is far more than that of CUR.So,we can say that both the techniques are useful but the choice of the technique that is deem fit for a scenario depends on the

characteristics of the dataset.For a larger dataset,it is better to go ahead with CUR so as to reduce the space and time overhead.But in the cases where every recommendation should be accurate,it’s better to use SVD since the reconstruction error is minimum.