Exercises
Computational Intelligence Lab
SS 2013

Machine Learning Laboratory
Dept. of Computer Science, ETH Zürich
Prof. Dr. Joachim M. Buhmann
Web http://cil.inf.ethz.ch/

Series 2, March 7-8, 2013 (SVD)

A recommender system is concerned with presenting items (e.g. books on Amazon, movies at Movielens or music at lastFM) that are likely to interest the user. In collaborative filtering, we base our recommendations on the (known) preference of the user towards other items, and also take into account the preferences of other users.

One approach to collaborative filtering is to apply SVD to the data. The data comes as a matrix $\mathbf{A}$, where each row represents an item and each column represents a user. Each entry $\mathbf{A}_{i,j}$ indicates the rating of item i by user j.

Problem 1 (SVD Theory):

Viewers were asked to rate a list of movies (on a scale of 1-10). The results are presented in the following table, the * sign indicates a missing values (the user did not watch the movie).

	Ben	Tom	John	Fred	Jack
American pie	8	7	1	*	4
Shrek	9	7	2	5	6
Titanic	1	4	9	*	3
The godfather	3	*	8	5	4
Avatar	*	3	*	9	9
Star wars	5	1	4	10	*

We can think of this rating table as a matrix $\mathbf{A}$. In our case, $\mathbf{A} \in \mathbb{N}^{6,5}$.

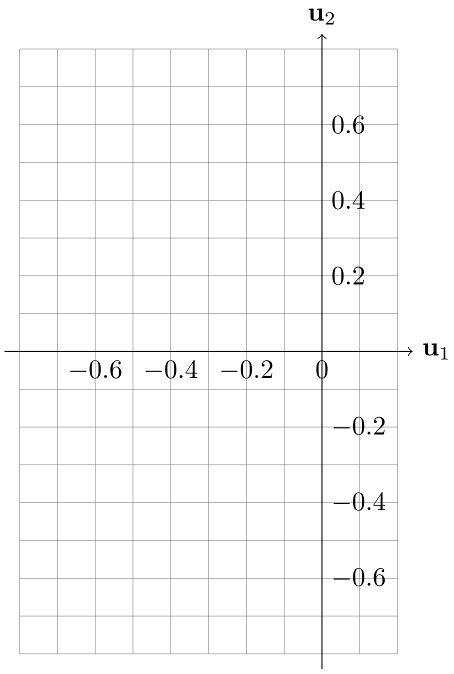
1. Consider the matrix $\mathbf{K} = \mathbf{A}\mathbf{A}^T$. What is the size of this matrix? What does $\mathbf{K}_{ij}$ tell us?
2. Consider the matrix $\mathbf{L} = \mathbf{A}^T\mathbf{A}$. What is the size of this matrix? What does $\mathbf{L}_{ij}$ tell us?
3. In SVD, we decompose $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$. What should be the sizes of $\mathbf{U}$, $\mathbf{D}$ and $\mathbf{V}$? (answer without looking at the matrices below).

   We used the MATLAB function `svd` to compute the decomposition. To do so we replaced each missing item with an average rating of 5.5

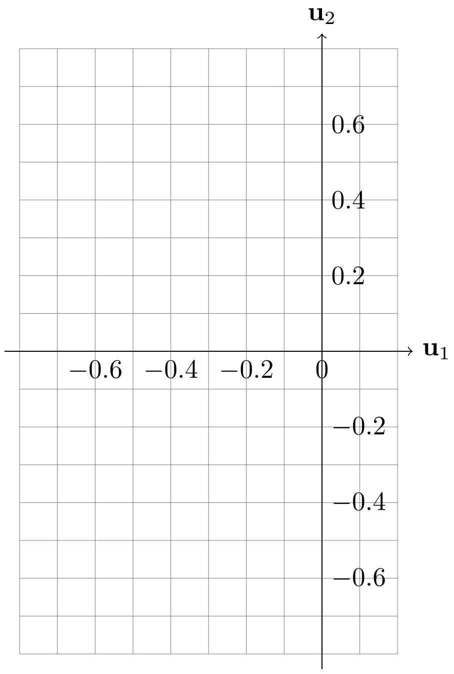
$$\mathbf{A} = \underbrace{\begin{pmatrix} -0.38 & -0.51 & 0.23 & -0.42 & 0.60 & 0.02 \\ -0.43 & -0.51 & 0.23 & 0.26 & -0.61 & -0.24 \\ -0.33 & 0.58 & 0.34 & -0.19 & 0.04 & -0.63 \\ -0.37 & 0.34 & 0.45 & 0.06 & -0.11 & 0.73 \\ -0.50 & 0.11 & -0.43 & 0.64 & 0.38 & -0.04 \\ -0.41 & 0.11 & -0.63 & -0.55 & -0.32 & 0.13 \end{pmatrix}}_{\mathbf{U}} \underbrace{\begin{pmatrix} 29.7 & 0 & 0 & 0 & 0 \\ 0 & 10.00 & 0 & 0 & 0 \\ 0 & 0 & 7.09 & 0 & 0 \\ 0 & 0 & 0 & 2.75 & 0 \\ 0 & 0 & 0 & 0 & 0.67 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}}_{\mathbf{D}} \underbrace{\begin{pmatrix} -0.44 & -0.60 & 0.02 & -0.11 & -0.66 \\ -0.37 & -0.25 & 0.73 & -0.07 & 0.51 \\ -0.39 & 0.74 & 0.35 & 0.06 & -0.41 \\ -0.56 & 0.16 & -0.50 & -0.56 & 0.31 \\ -0.45 & -0.04 & -0.30 & 0.81 & 0.20 \end{pmatrix}^T}_{\mathbf{V}}$$

4. What can we hope to gain by performing SVD on the rating matrix?
5. Consider the entries in $\mathbf{D}$, how many singular values is it reasonable for us to keep? Which ones would you keep, why?
6. What interpretation can you give for the matrix $\mathbf{U}$?
7. What interpretation can you give for the matrix $\mathbf{V}$?

8. If we keep the the first two columns of $\mathbf{U}$, we obtain a two dimensional representation. Plot this below and give an interpretation of the data.



9. If we keep the the first two columns of $\mathbf{V}$, we obtain another two dimensional representation. Plot this below and give an interpretation of the data.



10. After observing the plots (and applying some cinema knowledge), what interpretation can you give for the matrix $\mathbf{D}$?
11. Write down the approximation of the original matrix $\mathbf{A}$ using only the first three singular components.
12. What would be the approximation error under the euclidean and Frobenius norm?
13. Next, Bob wants to join the system. He shares a few of his movie ratings ([1,*,*,6,*,10]). How can we use the system to recommend movies for Bob?

   (a) Represent Bob in the new coordinate system and add this data to the plot.
   (b) Find the most suitable recommendation for Bob. (There is more than one way to do this.)

14. Does Bob's rating affect our prediction system?

Problem 2 (SVD for Collaborative Filtering):

In this assignment, we will apply singular value decomposition (SVD) to build our own Recommender system. We begin with first setting up the environment on your local machine for sending your Recommender system to the submission system, then we go step by step through the SVD collaborative filtering pipeline.

Submission system environment setup:

1. The web page for the collaborative filtering application can be found here:

   http://cil.inf.ethz.ch/applications/collaborative_filtering.

2. Download the provided templates for the function `PredictMissingValues.m`, the evaluation script `CollabFilteringEvaluation.m` and the data `data.mat` into a local folder.
3. Try running the evaluation script `CollabFilteringEvaluation.m` – it should output `5.3119`, which is the prediction error of the template function.

To submit your solution to the online evaluation system, we require you to prepare a ".zip" file containing:

1. A function file `PredictMissingValues.m` implementing

   `X_pred = PredictMissingValues(X, nil),`

   which takes a matrix with missing values (specified by the value `nil`) and outputs a predicted full matrix.

Your submission is evaluated according to the following criteria:

1. Root mean squared error of prediction.
2. Run time of `PredictMissingValues.m`.

Working with rating data: In the provided dataset, we observe that not all users rated all items, and hence, such ratings are not available. In this case, we have to deal with *missing values*. In collaborative filtering, the goal is to predict missing values from the observed ones.

First draft: A simple baseline algorithm One very simple way to compute the missing values is to replace each missing value by the average rating for the respective item. We suggest starting with this simple solution, later we will investigate more sophisticated approaches.

Below is a solution pipeline with an evaluation step:

1. **Load data:** Load the data matrix given in `Data.mat`. This loads a matrix $\mathbf{X}$ of 100 items by 7834 users.

   Each entry $\mathbf{X}_{i,j}$ contains a rating of item i by user j (scaled on the $[-10, 10]$ interval) or a special symbol (`nil` =99) for an absent rating (missing value). The goal is to predict the missing values (99) using the observed ones.

2. **Splitting data into training/testing sets:** Being a genuine dataset, some of the ratings are missing and therefore can not be used for evaluation. The remaining known values are then used for both training and evaluation. We artificially introduce new missing values by replacing some of the *known ratings* by the `nil` values. We store the original values for evaluation purposes. The artificially created dataset is denoted by $\mathbf{X}^{trn}$ and is used for learning. The omitted known values are stored in $\mathbf{X}^{tst}$ for evaluation.

   The evaluation script splits the data into Training/Test splits, by splitting the existing ratings in the data $\mathbf{X} \in \mathbb{R}^{N \times M}$ into two new matrices (of the same size) $\mathbf{X}^{trn}$ and $\mathbf{X}^{tst}$ which have no ratings in common. If a rating $\mathbf{X}_{i,j}$ is observed in $\mathbf{X}^{trn}$, then it is a missing in $\mathbf{X}^{tst}$.

3. **Prediction:** In this simple baseline algorithm, we impute missing values in the matrix $\mathbf{X}^{pred}$ by setting them to the average over all observed ratings for a particular item, in $\mathbf{X}^{trn}$. Since the matrices are of the same size, the index of an item (and user) in the train matrix is the same as in the predictions matrix.

4. **Evaluation:** Compare the predicted values in $\mathbf{X}^{pred}$ to the true observed values in $\mathbf{X}^{tst}$ using the root mean squared error:

$$J = \sqrt{\frac{\sum_{\mathbf{X}^{tst}_{i,j} \neq \mathtt{nil}} (\mathbf{X}^{pred}_{i,j} - \mathbf{X}^{tst}_{i,j})^2}{\sum_{\mathbf{X}^{tst}_{i,j} \neq \mathtt{nil}} 1}} \tag{1}$$

Second draft: Improving prediction Using SVD In this part we assume that the data has an underlying structure where each user and each item has an equivalent representation in some concept space. This kind of structure can be found using SVD. Performing SVD requires a full matrix, we can use the baseline solution as an initialization step to obtain missing values predictions. This initial step, although necessary, introduces noise which affects the SVD result. The hope is that the "true" underlying structure can be revealed by discarding the less informative part of the decomposition. The initial missing values can then be inferred using the approximated representation.

1. **Impute missing values:** Repeat previous steps.
2. **SVD decomposition:** Compute the SVD decomposition of the training matrix with imputed values:

$$\tilde{\mathbf{X}}^{trn} = \mathbf{U}\mathbf{D}\mathbf{V}^\top.$$

For simplicity, we recommend multiplying $\mathbf{U}$ and $\mathbf{V}$ by the square root of the eigenvalues:

$$\mathbf{U}' = \mathbf{U}\sqrt{\mathbf{D}},$$

$$\mathbf{V}' = \sqrt{\mathbf{D}}\mathbf{V}.$$

3. **Prediction:** The prediction for any missing value $\mathbf{X}_{i,j} = \mathtt{nil}$ (in our case $\mathtt{nil} = 99$) can now be computed as the inner product of the i-th row in $\mathbf{U}'$ and the j-th column in $\mathbf{V}'^\top$.
4. **Model Selection:** Select a number k of eigenvalues to be used and truncate $\mathbf{U}$ and $\mathbf{V}$ accordingly. Compute predictions for test matrix using the truncated matrices and calculate the error (Eq. 1). Try different values of k, and observe what happens to your error rates.
5. **Recommender System:** Write a function to predict the (unknown) ratings:

```
X_pred = PredictMissingValues(X, nil),
```

where `X` contains the training matrix, and `X_pred` contains the same matrix with the missing `nil` entries replaced with the hypothesized ratings.

Extensions: Naturally there are many ways to improve your SVD solution. More advanced techniques can be found in the following publications:

- Koren Y., Bell R., Volinsky B., "MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS" IEEE Computer, Volume 42, Issue 8, p.30-37 (2009); http://research.yahoo.com/files/ieeecomputer.pdf
- A. Paterek, "Improving Regularized Singular Value Decomposition for Collaborative Filtering," Proc. KDD Cup and Workshop, ACM Press, 2007, pp. 39-42.