CS 391L **Machine Learning WB** **Spring 2025**

Homework 1

Lecturer: Inderjit Dhillon Date Due: Feb 14, 2025

Keywords: *Netflix, Regression, Least Squares*

1. **Million Dollar Question**

   For this problem, you will work on the Netflix dataset, where one needs to predict missing (movie, user) ratings. The problem of predicting missing values for a recommender system is formally known as Collaborative Filtering. Read more about the Netflix challenge and the dataset `https://www.kaggle.com/datasets/netflix-inc/netflix-prize-data`. The complete Netflix dataset has 480,189 users, 17,770 movies, and 100,480,507 ratings. So, the scale of the problem is huge. For this assignment we will work on a much smaller subset of 2000 users, 1821 movies and 220,845 ratings. You can download this subset dataset from Canvas $\rightarrow$ Files $\rightarrow$ HW1 $\rightarrow$ dataset.zip

   We will take an approach based on Matrix Factorization which was also used by a leading team for Netflix challenge. Let us first introduce some notation. Let there be u users and m movies and let $R \in \mathbb{R}^{u \times m}$ be the ratings matrix where element R_{ij} is the rating given by user i for movie j. Note that the matrix R has many missing values - in the entire Netflix dataset, $\sim 94\%$ of the entries are missing. Think of factoring the matrix $R \approx U^T M$, where $U \in \mathbb{R}^{k \times u}$ and $M \in \mathbb{R}^{k \times m}$ and $k \ll \min(u, m)$. Intuitively, you can think of column $\boldsymbol{u}_i$ of U as a low-dimensional feature vector for the i-th user, and the column $\boldsymbol{m}_j$ of M as a low-dimensional feature vector for the j-th movie.

   Using the above notation, the rating R_{ij} for a movie j by user i is approximated as $R_{ij} \approx \boldsymbol{u}_i^T \boldsymbol{m}_j$. Thus, the Matrix Factorization approach involves finding U and M such that $U^T M$ is as close to R as possible. Formally the problem can be stated as:

$$\min_{U,M} \ \|R - U^T M\|_F^2 + \lambda(\|U\|_F^2 + \|M\|_F^2), \tag{1}$$

   where $\|\cdot\|_F$ denotes the Frobenius norm, and where we have added regularization terms (as in ridge regression) and the regularization parameter $\lambda \in [0, \infty]$.

   This problem is in general hard to solve as both U and M need to be inferred and no known method guarantees the optimal solution. Instead, we use a heuristic which works well in practice. We use the method of Alternating Minimization, where in each iteration we fix U to compute M, and subsequently fix M to compute U, and so on. Now we describe the method to compute M assuming U to be fixed.

   Let U be fixed, then (1) reduces to:

$$\min_{M} \ \|R - U^T M\|_F^2 + \lambda \|M\|_F^2,$$

   This problem can be decoupled in terms of columns of M, $\boldsymbol{m}_j$, i.e.,

$$\min_{\{\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_m\}} \ \sum_j \|\boldsymbol{r}_j - U^T \boldsymbol{m}_j\|_2^2 + \lambda \sum_j \|\boldsymbol{m}_j\|_2^2,$$

   where $\boldsymbol{r}_j$ denote the j-th column of R. Thus, we can solve a separate ridge regression problem for each j, i.e.,

$$\min_{\boldsymbol{m}_j} \ \|\boldsymbol{r}_j - U^T \boldsymbol{m}_j\|_2^2 + \lambda \|\boldsymbol{m}_j\|_2^2,$$

However, the vector $\boldsymbol{r}_j$, which represents ratings for movie j, may have many missing values. Let there be n_j known ratings for the movie j. Then,the ridge regression should be solved using only n_j ratings, i.e. let $\boldsymbol{r}_j^{(\mathcal{K}_|)} \in \mathbb{R}^{n_j}$ denote the known ratings for movie j, and $U^{(\mathcal{K}_j)} \in \mathbb{R}^{k \times n_j}$ denote the submatrix of U for the corresponding users who rated movie j. Thus the problem can be written as:

$$\min_{\boldsymbol{m}_j} \ \|\boldsymbol{r}_j^{(\mathcal{K}_|)} - \left(U^{(\mathcal{K}_j)}\right)^T \boldsymbol{m}_j\|_2^2 + \lambda \|\boldsymbol{m}_j\|_2^2,$$

An analogous procedure can be used for computing U.

So, for each iteration compute U and M using the procedure given above. Repeat this procedure for a fixed number of iterations. Initialize U and M randomly, fix the number of iterations τ as 30 and fix $k = 10$.

(a) Select $\lambda = 0$ and compute U and M. What are the problems you face?

(b) Train your model on multiple values for λ and record the Root Mean Squared Error (RMSE) on both validation and train set for each run. Report the following: a) plot of λ vs RMSE on validation set and training set (both on separate plots); b) optimal RMSE on validation set along with the optimal λ.

(c) Now let us consider preprocessing the data in order to remove some inherent bias in the data. First center the complete data using global mean. Now, center each row of the ratings matrix R to mean 0, i.e. remove user mean for each row. Similarly center each coumn of the resulting matrix. Perform the above operations on known ratings only. After this preprocessing, repeat question 2 above and report the optimal RMSE on validation set. Note that after factorization, you need to add the global mean and the user/movie bias for prediction. Report RMSE obtained. How does RMSE change? Explain.

(d) What other initialization can be used? What is the problem with using the zero matrix as the initialization for U and M.

2. Consider a linear least squares regression problem where we are given the training data $\{X_i, Y_i\}_{i=1}^N, X_i \in \mathbb{R}^d$ and $Y_i \in \mathbb{R}$ where $d > N$ (i.e. the number of features d is more than the number of samples N). For example, we might have $N = 5$ data points, but each point has $d = 10$ features. What challenges do we face when using standard least squares regression in this "high-dimension, low-sample" scenario? How can we address these challenges using techniques covered in class?