Homework 4, due Tuesday April 19 — COMS 4721 Spring 2016

Problem 1 (Collaborative filtering; 40 points)**.** In this problem, you will develop an alternating (regularized) least squares algorithm for a new collaborative filtering statistical model.

The statistical model has parameters

$$\boldsymbol{\theta} \;=\; (\boldsymbol{u}_1, \dots, \boldsymbol{u}_m, \boldsymbol{v}_1, \dots, \boldsymbol{v}_n, b_1, \dots, b_m, c_1, \dots, c_n, \mu)\,,$$

where

- $\boldsymbol{u}_i \in \mathbb{R}^k$ is the parameter vector for user i;
- $\boldsymbol{v}_j \in \mathbb{R}^k$ is the parameter vector for movie j;
- $b_i \in \mathbb{R}$ is the bias parameter for user i;
- $c_j \in \mathbb{R}$ is the bias parameter for movie j;
- $\mu \in \mathbb{R}$ is the global bias parameter.

Under the distribution with parameters $\boldsymbol{\theta}$ as above, the ratings $A_{i,j}$ for all user/movie pairs (i,j) are independent, and

$$A_{i,j} \;\sim\; \mathrm{N}\big(\langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle + b_i + c_j + \mu,\, 1\big)\,.$$

Let $\Omega \subseteq \{1, \dots, m\} \times \{1, \dots, n\}$ be a subset of user/movie pairs, and let $\mathcal{A} = \{a_{i,j} \in \mathbb{R} : (i,j) \in \Omega\}$ be a collection of ratings for these user/movie pairs. The log-likelihood $\boldsymbol{\theta}$ given $\mathcal{A}$ is (after dropping terms that do not depend on $\boldsymbol{\theta}$)

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{A}) \;:=\; -\frac{1}{2} \sum_{(i,j)\in\Omega} \big(\langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle + b_i + c_j + \mu - a_{i,j}\big)^2\,.$$

(a) Design an alternating least squares algorithm that attempts to maximize the log-likelihood objective. For simplicity, set μ to be the average rating in $\mathcal{A}$. Then, to set the remaining parameters, the algorithm should independently draw the $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ vectors from the $\mathrm{N}(\boldsymbol{0}, (1/k)\boldsymbol{I})$ distribution in $\mathbb{R}^k$, initialize the b_i and c_j to zero, and then perform the following iterative computation.

For $t = 1, 2, \dots, T$:

1. Holding all other parameters fixed, update the b_i for all $i = 1, 2, \dots, m$ so as to maximize the log-likelihood objective.
2. Holding all other parameters fixed, update the $\boldsymbol{u}_i$ for all $i = 1, 2, \dots, m$ so as to maximize the log-likelihood objective.
3. Holding all other parameters fixed, update the c_j for all $j = 1, 2, \dots, n$ so as to maximize the log-likelihood objective.
4. Holding all other parameters fixed, update the $\boldsymbol{v}_j$ for all $j = 1, 2, \dots, n$ so as to maximize the log-likelihood objective.

(Here, T is the total number of iterations.)

State *closed-form expressions* for all of the updates. (In this part, you may assume the invertibility of matrix you like, but state these assumptions explicitly.)

(b) Implement the alternating least squares algorithm from part (a); your code should compute the log-likelihood after each iteration. Download the "fake" movie ratings data set `ratings_fake.csv` from Courseworks. Each line is a triplet $(i, j, a_{i,j})$. Here, $m = 500$ and $n = 500$. Run your algorithm on this data with $k = 5$ for $T = 20$ iterations; the log-likelihood should be non-decreasing with each iteration. Submit a plot the log-likelihood as a function of the iteration number. Make sure the axes are clearly labeled.

(Do not submit this version of the code. You will submit code for a later part of this problem.)

(c) Download the "real" movie ratings (training) data set `ratings_train.csv` from Courseworks. Here, $m = 943$ and $n = 1682$. Run your algorithm on the training data with $k = 10$ for a few iterations. You should run in to some problem with the algorithm as is. What is the problem you encounter?

Note: Another issue that may come up is that some users or movies have no ratings in the training data. For such users i without any ratings, do not update the parameters $\boldsymbol{u}_i$ and b_i; and for such movies j without any ratings, do not update the parameters $\boldsymbol{v}_j$ and c_j.

(Again, do not submit this version of the code.)

(d) Modify your algorithm so that it instead maximizes the regularized log-likelihood

$$\widetilde{\mathcal{L}}(\boldsymbol{\theta}; \mathcal{A}) \ := \ \underbrace{\mathcal{L}(\boldsymbol{\theta}; \mathcal{A})}_{\text{log-likelihood}} \ - \underbrace{\lambda \left(\sum_{i=1}^{m} \|\boldsymbol{u}_i\|_2^2 + \sum_{j=1}^{n} \|\boldsymbol{v}_j\|_2^2 \right)}_{\text{regularizer}} .$$

(Here, $\lambda > 0$ is the regularization parameter.) Also modify the algorithm so that it computes both the *regularized* log-likelihood (given the training data) as well as the "Test Root Mean Squared Error" on the test set `ratings_test.csv` (available from Courseworks):

$$\text{Test RMSE}(\boldsymbol{\theta}) \ := \ \sqrt{\frac{1}{|\Omega_{\text{test}}|} \sum_{(i,j)\in\Omega_{\text{test}}} \bigl(\text{clip}(\langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle + b_i + c_j + \mu) - a_{i,j}\bigr)^2} \, ,$$

where

$$\text{clip}(z) \ := \ \begin{cases} 1 & \text{if } z < 1 \, , \\ z & \text{if } z \in [1, 5] \, , \\ 5 & \text{if } z > 5 \, . \end{cases}$$

(The ratings here are always between 1 and 5.)

Run your algorithm on the real movie ratings data set with $k = 10$ and $\lambda = 10$ for $T = 40$ iterations. Again, make sure the *regularized* log-likelihood is non-decreasing with each iteration. Submit two plots: the *regularized* log-likelihood and the Test RMSE, each as a function of the iteration number. Make sure axes are clearly labeled.

You should also submit your code that implements your algorithm, as well as any other code needed to replicate the results and plots.

(e) (Optional.) If you are feeling up for it (perhaps for extra credit), use cross-validation to figure out a good setting of k, λ, and T—all just using the training data. What Test RMSE are you able to achieve?

(If you do this, please submit your code separately from the code needed for the previous part of this problem.)

Problem 2 (PCA; 20 points)**.** Suppose Alice and Bob each have the data matrix $\boldsymbol{A} \in \mathbb{R}^{n \times d}$. Let the eigenvalues of $\boldsymbol{A}^\top \boldsymbol{A}$ be denoted by $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$.

(a) Suppose Alice and Bob each computes three orthonormal eigenvectors of $\boldsymbol{A}^\top \boldsymbol{A}$ corresponding to λ_1, λ_2, and λ_3. Call them $\{\boldsymbol{v}_{1,\text{Alice}}, \boldsymbol{v}_{2,\text{Alice}}, \boldsymbol{v}_{3,\text{Alice}}\}$ and $\{\boldsymbol{v}_{1,\text{Bob}}, \boldsymbol{v}_{2,\text{Bob}}, \boldsymbol{v}_{3,\text{Bob}}\}$. Alice and Bob use different software with no numerical precision errors to compute these eigenvectors. They find that no matter how they flip the vectors (by multiplying by -1) or re-arrange the order of the vectors (with a permutation $\pi\colon \{1,2,3\} \to \{1,2,3\}$), they cannot seem to match up the eigenvectors. In other words,

$$\min_{\substack{\text{permutation} \\ \pi\colon \{1,2,3\} \to \{1,2,3\}}} \min_{\sigma_1,\sigma_2,\sigma_3 \in \{-1,+1\}} \sum_{i=1}^{3} \|\boldsymbol{v}_{i,\text{Alice}} - \sigma_i \boldsymbol{v}_{\pi(i),\text{Bob}}\|_2 \ > \ 0\,.$$

Give a simple reason for why this is does not actually indicate a problem with their software (i.e., explain how it can happen even with correct software).

(b) Now, using some different (and possibly less reliable) software, Alice and Bob each computes the orthogonal projection operator to the three-dimensional subspace spanned by eigenvectors corresponding to $\{\lambda_1, \lambda_2, \lambda_3\}$. Call them $\boldsymbol{\Pi}_{\text{Alice}}$ and $\boldsymbol{\Pi}_{\text{Bob}}$. Suppose $\lambda_1 = \lambda_2 = \lambda_3 = 1$ and $\lambda_i = 1/i$ for $i = 4, 5, \ldots, d$; and yet Alice and Bob find that $\boldsymbol{\Pi}_{\text{Alice}} \neq \boldsymbol{\Pi}_{\text{Bob}}$. Why does this indicate a problem with someone's (Alice's or Bob's) software?

(c) Same as (b), except suppose $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1$ and $\lambda_i = 1/i$ for $i = 5, 6, \ldots, d$. Does the fact $\boldsymbol{\Pi}_{\text{Alice}} \neq \boldsymbol{\Pi}_{\text{Bob}}$ indicate a problem? Why or why not?