Exercises
Computational Intelligence Lab
SS 2020

Machine Learning Institute
Dept. of Computer Science, ETH Zürich
Prof. Dr. Thomas Hofmann
Web http://da.inf.ethz.ch/cil

Series 4, March 9, 2020
(Matrix Approximation & Reconstruction)

Problem 1 (Constrained Optimization with Lagrange multipliers):

The method of Lagrange multipliers can be used to find local maxima and minima associated with a function subject to equality constraints. It is widely applied in several different scientific fields and plays an important role in a number of key derivations in machine learning.

We are going to use it to provide an alternative derivation of a well-known result in linear algebra: if $\boldsymbol{A}$ is an $n \times n$ real symmetric matrix, then there exists an orthonormal basis of $\mathbb{R}^n$ consisting of eigenvectors of $\boldsymbol{A}$. Let's consider the quadratic form $f(\mathbf{x}) = \langle \mathbf{x}, \boldsymbol{A}\mathbf{x} \rangle$ and suppose we want to optimize f on the unit sphere $S^{n-1} = \left\{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\|^2 = 1\right\}$. That is, with $g(\mathbf{x}) := \|\mathbf{x}\|^2 - 1$, the constraint is given by $g(\mathbf{x}) = 0$.

1. Let $\lambda_1 = \max f|_{S^{n-1}}$ and $\mathbf{v}_1 \in S^{n-1}$ a point maximizing f, i.e., $\lambda_1 = f(\mathbf{v}_1)$. Prove that $\boldsymbol{A}\mathbf{v}_1 = \lambda_1 \mathbf{v}_1$.
2. Now, maximize f on the set $S^{n-2} = \left\{\mathbf{x} \in S^{n-1} : \langle \mathbf{x}, \mathbf{v}_1 \rangle = 0\right\}$. More specifically, with $g(\mathbf{x})$ as before and $h(\mathbf{x}) := \langle \mathbf{x}, \mathbf{v}_1 \rangle$, consider $g(\mathbf{x}) = 0$ and $h(\mathbf{x}) = 0$ as the new constraints. Assuming that $\lambda_2 = \max f|_{S^{n-2}}$ and $\mathbf{v}_2 \in S^{n-2}$ is a point maximizing f, prove that $\boldsymbol{A}\mathbf{v}_2 = \lambda_2 \mathbf{v}_2$.

   Hint: Show that if $(\mathbf{x}, \lambda, \mu)$ satisfies

   $$\begin{cases} \nabla(f - \lambda g - \mu h)(\mathbf{x}) = 0 \\ g(\mathbf{x}) = 0 \\ h(\mathbf{x}) = 0, \end{cases}$$

   then $\mu = 0$, $\boldsymbol{A}\mathbf{x} = \lambda \mathbf{x}$ and $\lambda = f(\mathbf{x})$.
3. Applying the same rationale as above, prove that $\boldsymbol{A}\mathbf{v}_3 = \lambda_3 \mathbf{v}_3$, where $\lambda_3 = \max f|_{S^{n-3}} = f(\mathbf{v}_3)$ and $S^{n-3} = \left\{\mathbf{x} \in S^{n-1} : \langle \mathbf{x}, \mathbf{v}_1 \rangle = 0, \langle \mathbf{x}, \mathbf{v}_2 \rangle = 0\right\}$.
4. By iterating the above procedure, conclude that $\{\mathbf{v_k}\}_{k=1}^n$ forms an orthonormal basis of $\mathbb{R}^n$, with $\boldsymbol{A}\mathbf{v}_k = \lambda_k \mathbf{v}_k$, $\lambda_k = \max f|_{S^{n-k}} = f(\mathbf{v}_k)$, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$.
5. Recap in a few words how the Lagrange multiplier method is used as part of PCA.

Problem 2 (Alternating Least Squares for Collaborative Filtering):

Suppose we have a user-movie rating matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ which contains the ratings from m users for n movies on Netflix. Let $\mathcal{I}$ contain the indexes of the entries observed in $\mathbf{A}$.

To build a recommender system, we decompose the rating matrix $\mathbf{A}$ into a product of two matrices $\mathbf{UV}$, where $\mathbf{U} \in \mathbb{R}^{m \times k}$ and $\mathbf{V} \in \mathbb{R}^{k \times n}$ are the user and item matrices correspondingly, and the number of factors $k \ll \max(n, m)$ is relatively small. Let the i-th row of $\mathbf{U}$ be $\mathbf{u}_i^\top$ ($\mathbf{u}_i \in \mathbb{R}^k$), and the j-th column of $\mathbf{V}$ be $\mathbf{v}_j \in \mathbb{R}^k$. Let $\mathbf{I}_k$ denote the $k \times k$ identity matrix.

We are going to perform approximate matrix factorization by minimizing the regularized Frobenius loss

$$L(\mathbf{U}, \mathbf{V}) = \sum_{(i,j) \in \mathcal{I}} (a_{ij} - \mathbf{u}_i^\top \mathbf{v}_j)^2 + \lambda \sum_{i=1}^{m} \|\mathbf{u}_i\|^2 + \lambda \sum_{j=1}^{n} \|\mathbf{v}_j\|^2, \tag{1}$$

where $\lambda > 0$ is the regularization strength.

The Alternating Least Squares algorithm aims at minimizing the loss function $L(\mathbf{U}, \mathbf{V})$ as follows.

Algorithm 1: Alternating Least Squares (ALS)

Initialize $\mathbf{U}, \mathbf{V}$
while *not convergent* **do**
  for $i = 1, \ldots, m$ **do**
    $\mathbf{u}_i = (\sum_{j:(i,j) \in \mathcal{I}} \mathbf{v}_j \mathbf{v}_j^\top + \lambda \mathbf{I}_k)^{-1} \sum_{j:(i,j) \in \mathcal{I}} a_{ij} \mathbf{v}_j$
  for $j = 1, \ldots, n$ **do**
    $\mathbf{v}_j = (\sum_{i:(i,j) \in \mathcal{I}} \mathbf{u}_i \mathbf{u}_i^\top + \lambda \mathbf{I}_k)^{-1} \sum_{i:(i,j) \in \mathcal{I}} a_{ij} \mathbf{u}_i$

1. Is the objective function (1) convex with respect to the pair $(\mathbf{U}, \mathbf{V})$? If not, prove it.
2. Is the objective (1) convex with respect to $\mathbf{U}$?
3. Derive the update rule for $\mathbf{u}_i$. Note that the update rule for $\mathbf{v}_j$ is symmetric to that for $\mathbf{u}_i$.

   Hint: differentiate the objective (1) with respect to $\mathbf{u}_i$ holding $\mathbf{V}$ constant and set the gradient to zero.
4. Suppose the computational complexity of inverting a $k \times k$ matrix is $O(k^3)$, let n_i be the number of items rated by user i. Find the computational complexity of the step

$$\mathbf{u}_i = \Big(\sum_{j:(i,j) \in \mathcal{I}} \mathbf{v}_j \mathbf{v}_j^\top + \lambda \mathbf{I}_k \Big)^{-1} \sum_{j:(i,j) \in \mathcal{I}} a_{ij} \mathbf{v}_j$$

   in the ALS algorithm above. Use big O notation.
5. For a recommender system, $\mathbf{u}_i$ and $\mathbf{v}_j$ can be interpreted as the low-dimensional representations of user i and item j correspondingly. Interpret the update steps of the ALS algorithm in terms of obtaining low-dimensional representations for a recommender system.

Problem 3 (Stochastic Gradient Descent for Collaborative Filtering):

We have seen matrix completion already in Exercise 2, where we approximated a full matrix by an SVD.

In this exercise, we will apply *optimization techniques* to directly minimize the training error for the (unconstrained) matrix factorization formulation $\min_{\mathbf{U}\in Q_1,\mathbf{Z}\in Q_2} f(\mathbf{U},\mathbf{Z})$, with the objective function being the mean squared error,

$$f(\mathbf{U},\mathbf{Z}) = \frac{1}{|\Omega|}\sum_{(d,n)\in\Omega}\frac{1}{2}\left[\mathbf{X}_{dn} - (\mathbf{U}\mathbf{Z}^T)_{dn}\right]^2, \tag{2}$$

and $\mathbf{U}\in Q_1 := \mathbb{R}^{D\times K}$, $\mathbf{Z}\in Q_2 := \mathbb{R}^{N\times K}$. Here, $\Omega \subseteq [D]\times[N]$ is the set of indices of the observed ratings in the input matrix $\mathbf{X}$.

Environment setup. Please use the same setup and data as in Exercise 2. This is also explained on the web page for the collaborative filtering project, https://www.kaggle.com/c/cil-collab-filtering-2019.

The Task. Implement Stochastic Gradient Descent:

1. Derive the full gradient $\nabla_{(\mathbf{U},\mathbf{Z})} f(\mathbf{U},\mathbf{Z})$. Note that since we have $(D+N)\times K$ variables, the gradient here can be seen as a $(D+N)\times K$ matrix.
2. Derive a stochastic gradient $\mathbf{G}$ using the sum structure of f over the Ω elements. We want to do this in such a way that $\mathbf{G}$ only depends on a single observed rating $(d,n)\in\Omega$.
3. Implement the Stochastic Gradient Descent algorithm[1] for our objective function given in (2).
4. Experimentally find the best stepsize γ to obtain the lowest training error value.
5. Does the test error also decrease monotonically during optimization, or does it increase again after some time?
6. (Optional) Can you speed up your code, for example by maintaining the set of values $(\mathbf{U}\mathbf{Z}^\top)_{dn}$ for the few observed values $(d,n)\in\Omega$, and thereby avoiding the computation of the matrix multiplication $\mathbf{U}\mathbf{Z}^\top$ in every step?

Extensions. Naturally there are many ways to improve your solution. One of them is to use regularization term to avoid over-fitting. Such techniques and other extensions can be found, e.g., in the following publications:

[1] Webb, B. (2006). *Netflix Update: Try This at Home*. Simon Funk's Personal Blog. `http://sifter.org/~simon/journal/20061211.html`

[2] Koren Y., Bell R., Volinsky B. (2009). *Matrix Factorization Techniques for Recommender Systems*. IEEE Computer, Volume 42, Issue 8, pp. 30-37. `http://research.yahoo.com/files/ieeecomputer.pdf`

[3] A. Paterek (2007). *Improving Regularized Singular Value Decomposition for Collaborative Filtering*. Proc. KDD Cup and Workshop, ACM Press, pp. 39-42.

[1] `https://en.wikipedia.org/wiki/Stochastic_gradient_descent`