Computational Intelligence Laboratory

Matrix Approximation & Reconstruction

An Bian, Mikhail Karasikov

22-23 March 2018

ETH Zurich – cil.inf.ethz.ch

Overview

Yet Another Way for Collaborative Filtering

Given: rating matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ (e.g., the user-movie matrix)

$\mathbf{A}$ has only a subset of entries, indexed by $\mathcal{I}$

18,000 movies

480,000 users

x	1	1	x	...	x
x	x	x	5	...	x
x	x	3	x	...	x
x	4	3	x	...	2
...	x	x	x	...	x
x	5	x	1	...	x
x	x	3	3	...	x
x	1	x	x	...	2

Target: approximately decompose $\mathbf{A}$ into a product of two matrices $\mathbf{U}^\top$ and $\mathbf{V}$: $a_{ij} \approx \mathbf{u}_i^\top \mathbf{v}_j \ \forall (i,j) \in \mathcal{I}$

Why: to recover the entries missing in $\mathbf{A}$, $(i,j) \notin \mathcal{I}$

The Regularized Objective

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

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

Effect of regularization: Prevent entries of $\mathbf{u}_i^\top, \mathbf{v}_j$ to be too large

Remarks

- L is non-convex w.r.t. $(\mathbf{U}, \mathbf{V})$ even for $m = n = 1$
- However, convex w.r.t. each of $\mathbf{U}$ and $\mathbf{V}$

The ALS Algorithm

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

Algorithm 1: Alternating Least Squares (ALS)

How to derive the update rule?

- Differentiate the objective w.r.t. $\mathbf{u}_i$ holding $\mathbf{V}$ constant and set the gradient to zero.
- Symmetric for $\mathbf{v}_j$.

How to Utilize the Obtained $\mathbf{U}$ and $\mathbf{V}$?

1. Complete the missing entries $a_{pq} := \mathbf{u}_p^\top \mathbf{v}_q, \ (p,q) \notin \mathcal{I}$
2. Use as low-dimensional representations of users/items

Problem 1: Derivations

$$\frac{\partial L(\mathbf{U}, \mathbf{V})}{\partial \mathbf{u}_i} = -2 \sum_{j:(i,j)\in\mathcal{I}} (a_{ij} - \mathbf{u}_i^\top \mathbf{v}_j)\mathbf{v}_j + 2\lambda \mathbf{u}_i = 0$$

Therefore,

$$\begin{aligned}
\sum_{j:(i,j)\in\mathcal{I}} a_{ij}\mathbf{v}_j &= \sum_{j:(i,j)\in\mathcal{I}} (\mathbf{u}_i^\top \mathbf{v}_j)\mathbf{v}_j + \lambda \mathbf{u}_i \\
&= \sum_{j:(i,j)\in\mathcal{I}} \mathbf{v}_j(\mathbf{u}_i^\top \mathbf{v}_j) + \lambda \mathbf{u}_i \\
&= \sum_{j:(i,j)\in\mathcal{I}} \mathbf{v}_j(\mathbf{v}_j^\top \mathbf{u}_i) + \lambda \mathbf{u}_i \\
&= (\sum_{j:(i,j)\in\mathcal{I}} \mathbf{v}_j \mathbf{v}_j^\top)\mathbf{u}_i + \lambda \mathbf{u}_i \\
&= (\sum_{j:(i,j)\in\mathcal{I}} \mathbf{v}_j \mathbf{v}_j^\top + \lambda \mathbf{I}_k)\mathbf{u}_i.
\end{aligned}$$

Gradient Descent

Workhorse in Large-scale Learning Problems

Problem setup: a finite-sum

$$f(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} f_i(\mathbf{w}) \tag{1}$$

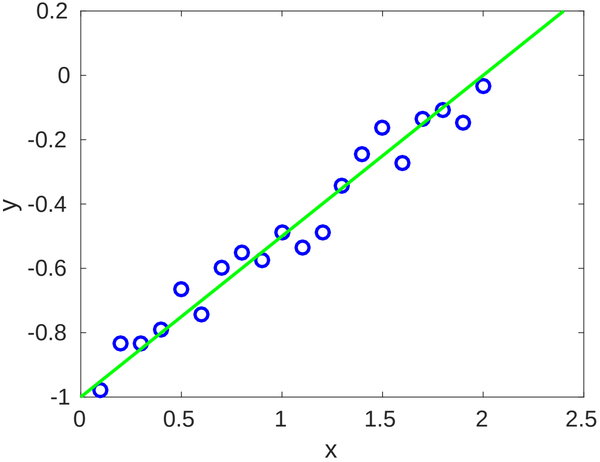
Example: Given n points $(x_i, y_i) \in \mathbb{R}^2$, $i = 1, \ldots, n$,

fit a line $y = w_0 + w_1 x$ to them by minimizing the quadratic loss

$$f_i(\mathbf{w}) = (w_0 + w_1 x_i - y_i)^2.$$

2D Line Fitting Example

The line: $y = 0.5 * x - 1, \ n = 20.$



Batch Gradient Descent

The gradient determines the ascending direction

$\Longrightarrow$ follow the negative gradient

Initialize $\mathbf{w}$;
while not convergent **do**
$\quad \mathbf{w} = \mathbf{w} - \eta \nabla f(\mathbf{w})$;

Notice that computing the gradient

$$\nabla f(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(\mathbf{w}) \tag{2}$$

requires the access to all data points.

For our 2D fitting example

$$\nabla f_i(\mathbf{w}) = [2(w_0 + w_1 x_i - y_i),\ 2(w_0 + w_1 x_i - y_i) x_i]^\top .$$

Potential Problems of the Batch GD

- May have huge amount of data points, say billions $\rightarrow$ very time consuming of computing the full gradient $\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\mathbf{w})$
- The whole dataset may not fit into the memory of a single machine (think of the ImageNet dataset of 14,197,122 images)
- Sometimes the whole dataset is not available (think of the online training of a spam filter where the emails come in a streaming fashion)

Stochastic Gradient Descent (SGD)

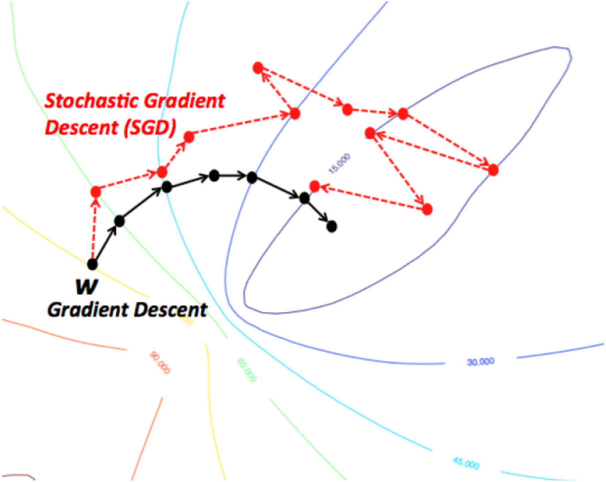
Initialize $\mathbf{w}$;
while not convergent **do**
  Randomly pick i from $\{1, \ldots, n\}$ (pick a data point);
  $\mathbf{w} = \mathbf{w} - \eta \nabla f_i(\mathbf{w})$;

- Needs to evaluate only one data point in each iteration.

For our 2D fitting example

$$\nabla f_i(\mathbf{w}) = [2(w_0 + w_1 x_i - y_i),\ 2(w_0 + w_1 x_i - y_i) x_i]^\top.$$

Typical Trajectories of SGD and Batch GD



SGD: A Versatile Algorithm

- Online setting/streaming setting
- Choice for large-scale setting
- Works well for non-convex models
  (e.g., training deep neural nets)
- Online setting/streaming setting
- Good statistical performance for learning problems

Solution to Problem 3

Consider the given objective function as a sum

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

where $\mathbf{U} \in \mathbb{R}^{D\times K}$, $\mathbf{Z} \in \mathbb{R}^{N\times K}$.

- **Stochastic Gradient:** For one fixed element (d, n) of the sum, we derive the gradient entry (d', k) of $\mathbf{U}$, that is $\frac{\partial}{\partial u_{d',k}} f_{d,n}(\mathbf{U}, \mathbf{Z})$, and analogously for the $\mathbf{Z}$ part.

$$\frac{\partial}{\partial u_{d',k}} f_{d,n}(\mathbf{U}, \mathbf{Z}) = \begin{cases} -\left[\mathbf{X}_{dn} - (\mathbf{U}\mathbf{Z}^T)_{dn}\right] z_{n,k} & \text{if } d' = d \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{\partial}{\partial z_{n',k}} f_{d,n}(\mathbf{U}, \mathbf{Z}) = \begin{cases} -\left[\mathbf{X}_{dn} - (\mathbf{U}\mathbf{Z}^T)_{dn}\right] u_{d,k} & \text{if } n' = n \\ 0 & \text{otherwise} \end{cases}$$

Recap: norms

Consider $\mathbf{A} \in \mathbb{R}^{m \times n}$

Common matrix norms

- $\|\mathbf{A}\|_F = \sqrt{\sum_i \sum_j |a_{ij}|^2} = \sqrt{\text{trace}(\mathbf{A}^\top \mathbf{A})} = \|\sigma(\mathbf{A})\|_2$
- $\|\mathbf{A}\|_* = \text{trace}(\sqrt{\mathbf{A}^\top \mathbf{A}}) = \|\sigma(\mathbf{A})\|_1$
- $\|\mathbf{A}\|_2 = \sigma_{\max}(\mathbf{A}) = \|\sigma(\mathbf{A})\|_\infty$

These norms are called the Schatten p-norms

$$\|\mathbf{A}\|_p = \|\sigma(\mathbf{A})\|_p$$

and computed as the p-norm of the vector of singular values of $\mathbf{A}$

p-norm

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

Recap: SVD

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^\top,$$

- $\mathbf{U} \in \mathbb{R}^{m \times m}$ is orthogonal, $\mathbf{U}\mathbf{U}^\top = \mathbf{U}^\top\mathbf{U} = \mathbf{I}_m$
- $\mathbf{V} \in \mathbb{R}^{n \times n}$ is orthogonal, $\mathbf{V}\mathbf{V}^\top = \mathbf{V}^\top\mathbf{V} = \mathbf{I}_n$
- $\mathbf{D} \in \mathbb{R}^{m \times n}$ is rectangular diagonal,

$$\mathbf{D} = \begin{bmatrix} \sigma_1 & & & & 0 & \dots \\ & \sigma_2 & & & 0 & \dots \\ & & \ddots & & \vdots & \ddots \\ & & & \sigma_m & 0 & \dots \end{bmatrix}, \quad \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m.$$

$$\|\mathbf{A}\|_* = \|\mathbf{D}\|_*$$

Solution to Problem 2.1

You must know from linear algebra:

- $\text{rank}(\mathbf{XY}) \leq \text{rank}(\mathbf{X}) \quad \forall \mathbf{X} \in \mathbb{R}^{m \times n}, \mathbf{Y} \in \mathbb{R}^{n \times k}$
- $\text{rank}(\mathbf{XY}) = \text{rank}(\mathbf{X}) \quad \forall \mathbf{Y} \in \mathbb{R}^{n \times n},\ \text{rank}(\mathbf{Y}) = n$

The SVD decomposition: $\mathbf{A} = \mathbf{UDV}^\top$

$$\text{rank}(\mathbf{A}) = \text{rank}(\mathbf{UDV}^\top) = \text{rank}(\mathbf{D}) = \#\{\sigma_i > 0\}$$

On the other hand,

$$\|\mathbf{A}\|_2 = \sigma_{\max}(\mathbf{A}) = \sigma_1$$

Therefore, if $\|\mathbf{A}\|_2 \leq 1$ and hence $\forall i\ \sigma_i \leq 1$, we have

$$\text{rank}(\mathbf{A}) = \#\{\sigma_i > 0\} = \sum_{i:\ \sigma_i > 0} 1 \geq \sum_{i:\ \sigma_i > 0} \sigma_i = \sum_i \sigma_i = \|\mathbf{A}\|_*$$

Solution to Problem 2.2

Def. $f: X \to \mathbb{R}$ is convex if $\forall x, y \in X$

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y) \quad \forall \lambda \in [0,1].$$

Prove. $\forall \lambda \in [0,1],\ \forall \mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$

$$\|\lambda \mathbf{A} + (1-\lambda)\mathbf{B}\|_* \leq \lambda \|\mathbf{A}\|_* + (1-\lambda)\|\mathbf{B}\|_*.$$

SVD decomposition: $\lambda \mathbf{A} + (1-\lambda)\mathbf{B} = \mathbf{U}_\lambda \mathbf{D}_\lambda \mathbf{V}_\lambda^\top$

$$\begin{aligned}
&\|\lambda \mathbf{A} + (1-\lambda)\mathbf{B}\|_* = \mathrm{trace}\,(\mathbf{D}_\lambda) \\
&= \mathrm{trace}\left((\mathbf{U}_\lambda^\top \mathbf{U}_\lambda)\mathbf{D}_\lambda(\mathbf{V}_\lambda^\top \mathbf{V}_\lambda)\right) \\
&= \mathrm{trace}\left(\mathbf{U}_\lambda^\top(\mathbf{U}_\lambda \mathbf{D}_\lambda \mathbf{V}_\lambda^\top)\mathbf{V}_\lambda\right) \\
&= \mathrm{trace}\left(\mathbf{U}_\lambda^\top(\lambda \mathbf{A} + (1-\lambda)\mathbf{B})\mathbf{V}_\lambda\right) \\
&= \lambda\,\mathrm{trace}\left(\mathbf{U}_\lambda^\top \mathbf{A} \mathbf{V}_\lambda\right) + (1-\lambda)\,\mathrm{trace}\left(\mathbf{U}_\lambda^\top \mathbf{B} \mathbf{V}_\lambda\right)
\end{aligned}$$

Solution to Problem 2.2...

$$\|\lambda \mathbf{A} + (1-\lambda)\mathbf{B}\|_* = \lambda \,\text{trace}\left(\mathbf{U}_\lambda^\top \mathbf{A} \mathbf{V}_\lambda\right) + (1-\lambda)\,\text{trace}\left(\mathbf{U}_\lambda^\top \mathbf{B} \mathbf{V}_\lambda\right)$$

To conclude, $\mathbf{A} = \mathbf{U}_A \mathbf{D}_A \mathbf{V}_A^\top$ and

$$\begin{aligned}
\text{trace}\left(\mathbf{U}_\lambda^\top \mathbf{A} \mathbf{V}_\lambda\right) &= \sum_{i=1}^{\min(m,n)} \left[\mathbf{U}_\lambda^\top \mathbf{A} \mathbf{V}_\lambda\right]_i^i = \sum_{i=1}^{\min(m,n)} \left[\mathbf{U}_\lambda^\top \mathbf{U}_A \mathbf{D}_A \mathbf{V}_A^\top \mathbf{V}_\lambda\right]_i^i \\
&= \sum_{i=1}^{\min(m,n)} \sum_{j=1}^{\min(m,n)} \left[\mathbf{U}_\lambda^\top \mathbf{U}_A\right]_j^i \sigma_j(\mathbf{A}) \left[\mathbf{V}_A^\top \mathbf{V}_\lambda\right]_i^j \\
&= \sum_{j=1}^{\min(m,n)} \sigma_j(\mathbf{A}) \sum_{i=1}^{\min(m,n)} \left[\mathbf{U}_\lambda^\top \mathbf{U}_A\right]_j^i \left[\mathbf{V}_A^\top \mathbf{V}_\lambda\right]_i^j \\
&\leq \sum_{j=1}^{\min(m,n)} \sigma_j(\mathbf{A}) \left\| \left[\mathbf{U}_\lambda^\top \mathbf{U}_A\right]_j \right\|_2 \left\| \left[\mathbf{V}_A^\top \mathbf{V}_\lambda\right]^j \right\|_2 \\
&= \sum_{j=1}^{\min(m,n)} \sigma_j(\mathbf{A}) = \|\mathbf{A}\|_*.
\end{aligned}$$

Convex Optimization

Why convex optimization

- Local minimum $\implies$ global minimum
- The set of all minima is convex
- For each strictly convex function, if the function has a minimum, then the minimum is unique
- **Efficient numerical methods for solving**

Convex optimization:

$$\min_{x} f(x) \quad \text{s.t. } x \in U,$$

where

- f is a convex function,
- U is a convex set.

Minimize a convex function over a convex set.

Convex Relaxation

Often, optimization problems can be formulated as follows:

$$\min_x f(x) \quad \text{s.t. } x \in Q,$$

where f is a **convex** function and Q is a **non-convex** set.

The framework of convex relaxations works as follows:

1. Replace the non-convex set Q with its convex superset $U \supset Q$
2. Find x^*, the solution to the convex optimization problem

$$\min_x f(x) \quad \text{s.t. } x \in U$$

3. Project x^* back to Q (the rounding procedure)

$$\hat{x} := \Pi_Q(x^*) \in Q \quad x^* \in U$$

Some relaxations are exact: $\min_{x \in U} f(x) = \min_{x \in Q} f(x)$.

Relaxing the low-rank approximation problem

$$\min_{\mathbf{B}} \text{rank}(\mathbf{B}) \quad \text{s.t. } \|\mathbf{A}-\mathbf{B}\|_{\mathbf{G}} = 0,\ \|\mathbf{B}\|_2 \leq 1 \tag{3}$$

The low-rank approximation problem (3) is equivalent to

$$\min_{\mathbf{B},\, r} r \quad \text{s.t. } \text{rank}(\mathbf{B}) \leq r,\ \|\mathbf{A}-\mathbf{B}\|_{\mathbf{G}} = 0,\ \|\mathbf{B}\|_2 \leq 1,$$

which can be relaxed to

$$\min_{\mathbf{B},\, r} r \quad \text{s.t. } \|\mathbf{B}\|_* \leq r,\ \|\mathbf{A}-\mathbf{B}\|_{\mathbf{G}} = 0,\ \|\mathbf{B}\|_2 \leq 1,$$

since $\|\mathbf{B}\|_* \leq \text{rank}(\mathbf{B})$ if $\|\mathbf{B}\|_2 \leq 1$.

Therefore, the problem (3) can be relaxed as follows:

$$\min_{\mathbf{B}} \|\mathbf{B}\|_* \quad \text{s.t. } \|\mathbf{A}-\mathbf{B}\|_{\mathbf{G}} = 0,\ \|\mathbf{B}\|_2 \leq 1.$$