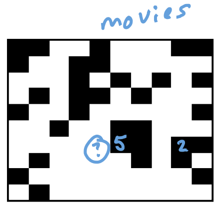
Case Study 4: Collaborative Filtering

Collaborative Filtering
Matrix Completion
Alternating Least Squares

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
May 7th, 2015

Matrix Completion Problem

X = 

X_{ij} known for black cells
X_{ij} unknown for white cells
Rows index users
Columns index movies

- Filling missing data?

well-approx. by rank k matrix

$$\underset{n \times m}{X} \simeq \underset{n \times k}{L}\ \underset{k \times m}{R'}$$

sparse

filled in matrix $\underline{\underline{\theta}}$

$n \cdot m$ params

soln for pred θ: learn low-dim user and movie vectors

$\underbrace{k \cdot n + k \cdot m}_{\text{learn this}} \ll \underbrace{n \cdot m}_{\text{predict this}}$

Matrix Completion via Rank Minimization

- Given observed values: $(u, v, r_{uv}) \in X$ some $r_{uv} = ?$
- Find matrix $\underset{n}{\overset{m}{\Theta}}$ ← filled in (not sparse)
- Such that: $\underbrace{\Theta_{uv} = r_{uv}}_{\text{fit } r_{uv} \neq ? \text{ perfectly}}$ $\forall r_{uv} \neq ?$ match ratings observed in X
- But… want low-rank Θ ☆
- Introduce bias: one possible objective $\left\{ \min_{\Theta} \operatorname{rank}(\Theta) \right.$ s.t. $\Theta_{uv} = r_{uv} \quad \forall r_{uv} \neq ?$
- Two issues:
  - NP-hard
  - you can't hope to get exact matching

Approximate Matrix Completion

- Minimize squared error: relax hard constraints
  - (Other loss functions are possible)

$$\min_{\Theta} \sum_{(u,v): r_{uv} \neq ?} (\Theta_{uv} - r_{uv})^2$$

allow for some error

- Choose rank *k:*

$${}_n\overset{m}{\Theta} = {}_n\overset{k}{L} \cdot \overset{m}{R'}$$ ← fix rank k

- Optimization problem:

$$\min_{L,R} \sum_{r_{uv} \neq ?} (L_u \cdot R_v - r_{uv})^2$$

non-convex opt. problem … local optima only

Coordinate Descent for Matrix Factorization

$$\min_{L,R} \sum_{(u,v):r_{uv}\neq ?} (L_u \cdot R_v - r_{uv})^2$$

- Fix movie factors R, optimize for user factors L
- First observation:

$$\min_{L_1,\dots,L_n} \sum_{(u,v):r_{uv}\neq ?} (L_u \cdot R_v - r_{uv})^2$$

$V_u \triangleq$ set of movies user u rated

$$= \min_{L_1,\dots,L_n} \sum_u \sum_{v\in V_u} (L_u \cdot R_v - r_{uv})^2$$

ind. opt. problem for each user

$$= \sum_u \min_{L_u} \sum_{v\in V_u} (L_u \cdot R_v - r_{uv})^2$$

data parallel problem

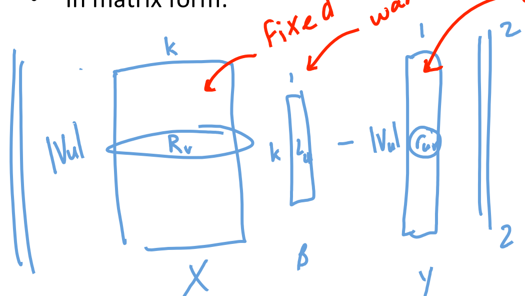
 5

Minimizing Over User Factors

- $\boxed{\text{For each user } u\text{:}}$ $\min_{L_u} \sum_{v\in V_u} (L_u \cdot R_v - r_{uv})^2$

want to learn in this iteration

- In matrix form:



Think of as $\| X\beta - y \|_2^2$

normal LS problem

- Second observation: Solve by
  - matrix inversion
  - gradient methods

 6

Coordinate Descent for Matrix Factorization: Alternating Least-Squares

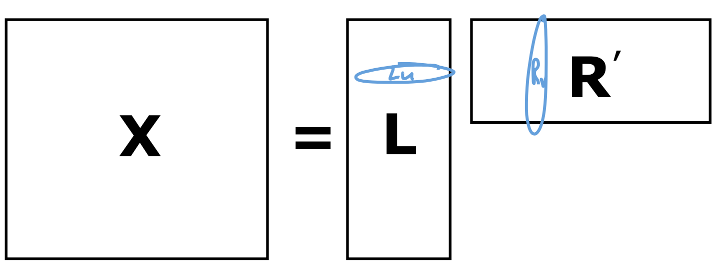
$$\min_{L,R} \sum_{(u,v):r_{uv}\neq ?} (L_u \cdot R_v - r_{uv})^2 + \lambda_u \|L\| + \lambda_v \|R\|$$

- $\boxed{\text{Fix movie factors}}$, optimize for user factors
  - Independent least-squares over users $\min_{L_u} \sum_{v \in V_u} (L_u \cdot R_v - r_{uv})^2 + \lambda_u \|L\|$
- $\boxed{\text{Fix user factors}}$, optimize for movie factors
  - Independent least-squares over movies $\min_{R_v} \sum_{u \in U_v} (L_u \cdot R_v - r_{uv})^2 + \lambda_v \|R\|$
- System may be underdetermined: use regularization
- Converges to local optima

Effect of Regularization

$\|A\|_F = \sqrt{\sum_{ij} A_{ij}^2}$

$$\min_{L,R} \sum_{(u,v):r_{uv}\neq ?} (L_u \cdot R_v - r_{uv})^2 + \lambda_u \|L\| + \lambda_v \|R\|$$



If $\|\cdot\| = \|\cdot\|_F^2$
each subproblem uses $\|L_u\|_2^2$ → ridge regression

If $\|\cdot\| = \|\cdot\|_1$,
each subproblem $\|L_u\|_1$
→ solved via LASSO methods

What you need to know…

- Matrix completion problem for collaborative filtering
- Over-determined -> low-rank approximation
- Rank minimization is NP-hard
- Minimize least-squares prediction for known values for given rank of matrix
  – Must use regularization
- Coordinate descent algorithm = "Alternating Least Squares"

Case Study 4: Collaborative Filtering

SGD for Matrix Completion
Matrix-norm Minimization

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
May 7th, 2015

Stochastic Gradient Descent

$$\min_{L,R} F(L,R) = \min_{L,R} \frac{1}{2} \sum_{r_{uv}} (\underbrace{L_u \cdot R_v - r_{uv}}_{\epsilon})^2 + \frac{\lambda_u}{2} ||L||_F^2 + \frac{\lambda_v}{2} ||R||_F^2$$

- Observe one rating at a time $r_{uv}^{(t)}$ $\quad \epsilon_t = L_u^{(t)} \cdot R_v^{(t)} - r_{uv}^{(t)}$
- Gradient observing r_{uv}:

$$\frac{\partial F}{\partial L_u} = \epsilon_t R_v + \lambda_u L_u$$

$$\frac{\partial F}{\partial R_v} = \epsilon_t L_u + \lambda_v R_v$$

$$\nabla F_t = \begin{bmatrix} \epsilon_t R_v^{(t)} + \lambda_u L_u^{(t)} \\ \epsilon_t L_u^{(t)} + \lambda_v R_v^{(t)} \end{bmatrix}$$

$$\begin{bmatrix} L \\ R \end{bmatrix} \leftarrow \begin{bmatrix} L \\ R \end{bmatrix} - \eta_t \nabla F_t$$ ← step size

- Updates:

$$\begin{bmatrix} L_u^{(t+1)} \\ R_v^{(t+1)} \end{bmatrix} \leftarrow \begin{bmatrix} (1-\eta_t \lambda_u) L_u^{(t)} - \eta_t \epsilon_t R_v^{(t)} \\ (1-\eta_t \lambda_v) R_v^{(t)} - \eta_t \epsilon_t L_u^{(t)} \end{bmatrix}$$ ← fast + easy to implement

Local Optima v. Global Optima

- We are solving:

$$\min_{L,R} \sum_{r_{uv}} (L_u \cdot R_v - r_{uv})^2 + \lambda_u ||L||_F^2 + \lambda_v ||R||_F^2$$

- We (kind of) wanted to solve:

$$\min_{\theta} \ \mathrm{rank}(\theta)$$
$$\theta_{uv} = r_{uv} \quad \forall (u,v,r_{uv}) \in X, \ r_{uv} \neq ?$$

?

- Which is NP-hard...
  - How do these things relate???

Eigenvalue Decompositions for PSD Matrices

- Given a (square) symmetric positive semidefinite matrix: $\underset{d\times d}{\Theta} \succeq 0$
  - Eigenvalues: $\lambda_1, \dots, \lambda_d \geq 0 \qquad \lambda = (\lambda_1, \dots, \lambda_d)$
- Thus rank is:

$$|\{\lambda_i : \lambda_i > 0\}| = \text{rank}(\Theta) = \|\lambda\|_0$$

- Approximation:

$$\|\lambda\|_0 \approx \|\lambda\|_1 = \sum_{i=1}^{d} |\lambda_i| \overset{PSD}{=} \sum_{i=1}^{d} \lambda_i$$

$\leftarrow$ L_1 norm is sum of eigvals

- Property of trace:

$$\text{trace}(\Theta) = \sum_{i=1}^{d} \lambda_i$$

- Thus, approximate rank minimization by:

$$\min_{\substack{\Theta \\ \Theta \succeq 0}} \text{rank}(\Theta) = \|\lambda\|_0 \quad \text{s.t. } \Theta_{uv} = r_{uv} \qquad \approx \qquad \min_{\substack{\Theta \\ \Theta \succeq 0}} \text{trace}(\Theta) = \|\lambda\|_1 \quad \text{s.t. } \Theta_{uv} = r_{uv}$$

Generalizing the Trace Trick

- Non-square matrices ain't got no trace
- For (square) positive semidefinite matrices, eigendecomposition:

$$\underset{d\times d}{\Theta} = P \Lambda P^{-1}$$

Λ = diag(λ)

- For rectangular matrices, singular value decomposition:

$$\underset{n\times m}{\Theta} = \underset{n\times n}{U}\; \underset{n\times m}{\Sigma}\; \underset{m\times m}{V'}$$

Σ: diagonal matrix w/ entries $\sigma_i(\Theta) \geq 0$ ← singular values

- Nuclear norm:

$$\|\Theta\|_* = \sum_{i}^{m} \sigma_i(\Theta)$$

$\uparrow$ nuclear norm

$$\min_{\Theta} \|\Theta\|_* \quad \text{s.t. } \Theta_{uv} = r_{uv}$$

convex problem!

Nuclear Norm Minimization

- Optimization problem:

new

$$\min_{\Theta} \|\Theta\|_* \quad \Theta_{uv} = r_{uv}$$

no feasible soln

- Possible to relax equality constraints: (relaxation of relaxation)

$$\min_{\Theta} \sum_{r_{uv}} (\Theta_{uv} - r_{uv})^2 + \lambda \|\Theta\|_*$$

- Both are convex problems!
(solved by semidefinite programming)

Analysis of Nuclear Norm

- Nuclear norm minimization = convex relaxation of rank minimization:

NP-hard   convex relaxation

$$\min_{\Theta} rank(\Theta) \quad \approx \quad \min_{\Theta} ||\Theta||_*$$

$$r_{uv} = \Theta_{uv}, \forall r_{uv} \in X, r_{uv} \neq ? \qquad r_{uv} = \Theta_{uv}, \forall r_{uv} \in X, r_{uv} \neq ?$$

- Theorem [Candes, Recht '08]:
  - If there is a true matrix of rank *k*,
  - And, we observe at least

$$C\ k\ \underbrace{n^{1.2} \log n}$$

original problem has n·m entries

assuming n ≥ m

random entries of true matrix

  - Then true matrix is recovered exactly with high probability via convex nuclear norm minimization!
    - Under certain conditions

Nuclear Norm Minimization vs. Direct (Bilinear) Low Rank Solutions

- Nuclear norm minimization: $\theta^* = \arg\min_{\Theta} \sum_{r_{uv}} (\Theta_{uv} - r_{uv})^2 + \lambda ||\Theta||_*$ (*)

  convex, global opt. close to truth

  - Annoying because:

    - θ very large (8B entries in Netflix)
    - SDP solvers are very slow (but poly time)

- Instead: $\min_{L,R} \sum_{r_{uv}} (L_u \cdot R_v - r_{uv})^2 + \lambda_u ||L||_F^2 + \lambda_v ||R||_F^2$ (**)

  ≈ 10-100 M params, but very fast solvers

  - Annoying because: - many local optima
  - But $||\Theta||_* = \inf \left\{ \min_{L,R} \frac{1}{2}||L||_F^2 + \frac{1}{2}||R||_F^2 : \Theta = LR' \right\}$
    - So (**) is a non-convex approx to (*)
    - And if we pick rank of L·R to be slightly larger than rank(θ^*), local optima of (**) are global optima of (*)
    - Under certain conditions [Burer, Monteiro '04]

What you need to know…

- Stochastic gradient descent for matrix factorization
- Norm minimization as convex relaxation of rank minimization
  - Trace norm for PSD matrices
  - Nuclear norm in general
- Intuitive relationship between nuclear norm minimization and direct (bilinear) minimization

Case Study 4: Collaborative Filtering

Nonnegative Matrix Factorization
Projected Gradient

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
May 6th, 2015

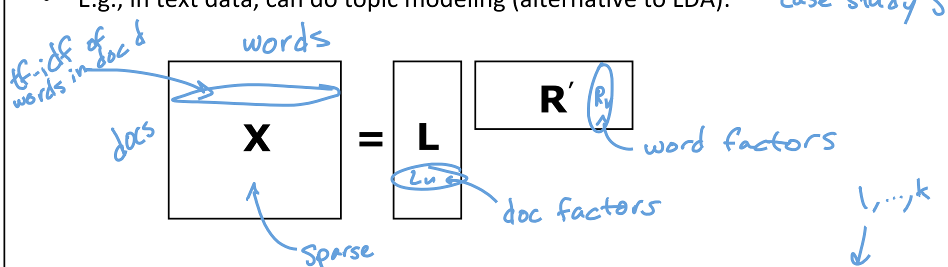
 19

Matrix factorization solutions can be unintuitive…

- Many, many, many applications of matrix factorization
- E.g., in text data, can do topic modeling (alternative to LDA):

every doc is a mixture of topics

will see in case study 5



- Would like:

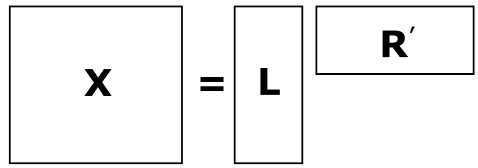
L_u: how much a doc is about each topic
R_v: how much a word contributes to a topic

- But…

Standard matrix factorization: L_u, R_v can be negative

 20

Nonnegative Matrix Factorization

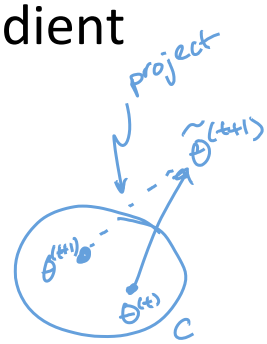


- Just like before, but

$$\underbrace{\min_{L\geq 0, R\geq 0}}_{\text{non-negative } L,\, R} \sum_{r_{uv}} (L_u \cdot R_v - r_{uv})^2 + \lambda_u ||L||_F^2 + \lambda_v ||R||_F^2$$

- Constrained optimization problem
  - Many, many, many, many solution methods… we'll check out a simple one

Recall: Projected Gradient



- Standard optimization:
  - Want to minimize: $\min_{\Theta} f(\Theta)$
  - Use, e.g., gradient updates:

$$\Theta^{(t+1)} \leftarrow \Theta^{(t)} - \eta_t \nabla f(\Theta^{(t)})$$

- Constrained optimization:
  - Given convex set *C* of feasible solutions
  - Want to find minima within *C*: $\min_{\Theta} f(\Theta)$ $\Theta \in \mathcal{C}$
- Projected gradient:
  - Take a gradient step (ignoring constraints):

$$\tilde{\theta}^{(t+1)} \leftarrow \theta^{(t)} - \eta_t \nabla f(\theta^{(t)})$$

  - Projection into feasible set:

$$\Pi_C(\theta) = \arg\min_{\beta \in C} ||\theta - \beta||_2^2$$

$\leftarrow$ often easy to compute (always convex)

$$\theta^{(t+1)} = \Pi_C(\tilde{\theta}^{(t+1)})$$

Projected Stochastic Gradient Descent for Nonnegative Matrix Factorization

$$\min_{L\geq 0, R\geq 0} \frac{1}{2}\sum_{r_{uv}}(L_u \cdot R_v - r_{uv})^2 + \frac{\lambda_u}{2}||L||_F^2 + \frac{\lambda_v}{2}||R||_F^2$$

- Gradient step observing r_{uv} ignoring constraints:

don't necessarily satisfy pos. constraints

$$\begin{bmatrix} \tilde{L}_u^{(t+1)} \\ \tilde{R}_v^{(t+1)} \end{bmatrix} \leftarrow \begin{bmatrix} (1-\eta_t\lambda_u)L_u^{(t)} - \eta_t\epsilon_t R_v^{(t)} \\ (1-\eta_t\lambda_v)R_v^{(t)} - \eta_t\epsilon_t L_u^{(t)} \end{bmatrix}$$

- Convex set: $L_u \geq 0 \quad R_v \geq 0 \quad \forall u,v$
- Projection step:

$$\Pi_C(\theta) = \underset{\beta\in C}{\arg\min}\, ||\theta-\beta||_2^2 \longleftarrow$$ totally ind. problems per dimension

Single dim:

$$= \underset{\beta\geq 0}{\arg\min}\,(\theta-\beta)^2 = \begin{cases} \theta & \text{if } \theta \geq 0 \\ 0 & \text{if } \theta < 0 \end{cases} = (\theta)_+ \longrightarrow \begin{bmatrix} L_u^{(t+1)} \\ R_v^{(t+1)} \end{bmatrix} \xleftarrow{\text{threshold}} \begin{bmatrix} \tilde{L}_u^{(t+1)} \\ \tilde{R}_v^{(t+1)} \end{bmatrix}$$

set all neg. coord. to 0

easy

©Emily Fox 2015 23

What you need to know…

- In many applications, want factors to be nonnegative
- Corresponds to constrained optimization problem
- Many possible approaches to solve, e.g., projected gradient

©Emily Fox 2015 24

Case Study 4: Collaborative Filtering

Cold Start Problem

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
May 6th, 2015

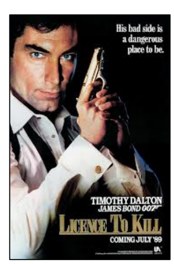
Cold-Start Problem

- **Challenge:** Cold-start problem (new movie or user)
- **Methods:** use features of movie/user

$$\phi_{skyfall} = \begin{pmatrix} 7 \\ 2 \\ 1 \\ \vdots \end{pmatrix}$$

IN THEATERS

$$\phi_{FRWL} = \begin{pmatrix} 8 \\ 1 \\ 6 \\ 0 \\ \vdots \end{pmatrix} \begin{array}{l} \text{action} \\ \text{romance} \\ \vdots \end{array}$$



Cold-Start Problem More Formally

- Consider a new user u' and predicting that user's ratings
  - No previous observations

    $r_{u'v} = ?\quad \forall v$

  - Objective considered so far:

    does not depend on $L_{u'}$ (no obs. ratings)

$$\min_{L,R} \frac{1}{2}\sum_{r_{uv}}(L_u \cdot R_v - r_{uv})^2 + \frac{\lambda_u}{2}||L||_F^2 + \frac{\lambda_v}{2}||R||_F^2$$

$L_{u'}$

only term that appears in ALS step

  - Optimal user factor:

    $L_{u'} = 0$ only penalty term is present

  - Predicted user ratings:

    always predict: $r_{u'v} = 0 \quad \forall v$ ... problem

27

An Alternative Formulation

- A simpler model for collaborative filtering
  - We would not have this issue if we assumed all users were identical

    - If all users shared a feature vector w, with w informed by all ratings, then can use w for new user

  - What about for new movies? What if we had side information?

    create movie feature vector

    $\phi(v) = ($ 'action' genre, 1994 year, Tarantino director ... $)$

  - What dimension should w be? same length as movie feature vector
  - Fit linear model:

    For all users u, $r_{uv} \approx w \cdot \phi(v)$

    fixed

  - Minimize:

    only param $\rightarrow \min_{w} \sum_{r_{uv}} (w \cdot \phi(v) - r_{uv})^2 + \lambda_w ||w||$ ← LS, Lasso, ridge

28

Personalization

- If we don't have any observations about a user, use wisdom of the crowd
  - **Address cold-start problem**

For user u', predict $r_{u'v} \approx w \cdot \phi(v)$

- Clearly, not all users are the same
- Just as in personalized click prediction, consider model with global and user-specific parameters

Consider user-specific deviations w_u from the crowd w

$$r_{uv} \approx (w + \underbrace{w_u}_{\text{init. to } 0}) \cdot \phi(v)$$

- As we gain more information about the user, forget the crowd

w_u more informed

User Features…

- In addition to movie features, may have information about the user:

$$\phi(u) = (\underbrace{25}_{\text{age}}, \underbrace{F}_{\text{gender}}, \underbrace{MSc}_{\text{education}}, \underbrace{A^+}_{\text{big data grade}}, \ldots)$$

- Combine with features of movie:

$$\phi(u,v) = (\cdots \phi(u) \cdots \phi(v) \cdots\cdots \text{cross features})$$

- Unified linear model:

$$r_{uv} = (w + w_u) \cdot \phi(u,v)$$

Feature-Based Approach vs. Matrix Factorization

- Feature-based approach:
  - Feature representation of user and movies fixed — important side info $\phi(u,v)$
  - Can address cold-start problem
- Matrix factorization approach:
  - Suffers from cold-start problem
  - User & movie features are learned from data — L_u, R_v
- A unified model: combine both ideas

$$r_{uv} \approx L_u \cdot R_v + (w + w_u) \cdot \phi(u,v)$$

solve via ALS
SGD
⋮

Unified Collaborative Filtering via SGD

$$\min_{L,R,w,\{w_u\}_u} \frac{1}{2} \sum_{r_{uv}} (L_u \cdot R_v + (w + w_u) \cdot \phi(u,v) - r_{uv})^2$$
$$+ \frac{\lambda_u}{2}||L||_F^2 + \frac{\lambda_v}{2}||R||_F^2 + \frac{\lambda_w}{2}||w||_2^2 + \frac{\lambda_{wu}}{2}\sum_u ||w_u||_2^2$$

- Gradient step observing $r_{uv}^{(t)}$ — $\epsilon_t = L_u^{(t)} \cdot R_u^{(t)} + (w^{(t)} + w_u^{(t)}) \cdot \phi(u,v) - r_{uv}^{(t)}$
  - For L,R

$$\begin{bmatrix} L_u^{(t+1)} \\ R_v^{(t+1)} \end{bmatrix} \leftarrow \begin{bmatrix} (1-\eta_t\lambda_u)L_u^{(t)} - \eta_t\epsilon_t R_v^{(t)} \\ (1-\eta_t\lambda_v)R_v^{(t)} - \eta_t\epsilon_t L_u^{(t)} \end{bmatrix}$$

  - For w and w_u: $\nabla_w F^{(t)} = \epsilon_t \, \phi(u,v) + \lambda_w w^{(t)}$

$$\begin{bmatrix} w^{(t+1)} \\ w_u^{(t+1)} \end{bmatrix} \leftarrow \begin{bmatrix} (1-\eta_t\lambda_w)w^{(t)} - \eta_t\epsilon_t\phi(u,v) \\ (1-\eta_t\lambda_{w_u})w_u^{(t)} - \eta_t\epsilon_t\phi(u,v) \end{bmatrix}$$

only update w_u for user u in $r_{uv}^{(t)}$

What you need to know…

- Cold-start problem
- Feature-based methods for collaborative filtering
  - Help address cold-start problem
- Unified approach

