Machine Learning & Data Mining

CS/CNS/EE 155

Lecture 11:
Latent Factor Models &
Non-Negative Matrix Factorization
### Kaggle Results (Public)

<table>
<thead>
<tr>
<th>#</th>
<th>Team Name</th>
<th>Kernel</th>
<th>Team Members</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sarcastic Gradient Dissent</td>
<td></td>
<td></td>
<td>0.86480</td>
</tr>
<tr>
<td>2</td>
<td>KDT</td>
<td></td>
<td></td>
<td>0.86240</td>
</tr>
<tr>
<td>3</td>
<td>Yeet:)</td>
<td></td>
<td></td>
<td>0.86219</td>
</tr>
<tr>
<td>4</td>
<td>Jim Wants Internships</td>
<td></td>
<td></td>
<td>0.86160</td>
</tr>
<tr>
<td>5</td>
<td>岳教授萬歲萬歲萬歲</td>
<td></td>
<td></td>
<td>0.86140</td>
</tr>
<tr>
<td>6</td>
<td>======A+ LEVEL=====</td>
<td></td>
<td></td>
<td>0.86119</td>
</tr>
<tr>
<td>7</td>
<td>quackle</td>
<td></td>
<td></td>
<td>0.86060</td>
</tr>
<tr>
<td>8</td>
<td>Not Hotdog</td>
<td></td>
<td></td>
<td>0.86060</td>
</tr>
<tr>
<td>9</td>
<td>Average is Okay</td>
<td></td>
<td></td>
<td>0.86019</td>
</tr>
<tr>
<td>10</td>
<td>』</td>
<td></td>
<td></td>
<td>0.85999</td>
</tr>
<tr>
<td>11</td>
<td>Kar98k</td>
<td></td>
<td></td>
<td>0.85919</td>
</tr>
<tr>
<td>12</td>
<td>LeMao</td>
<td></td>
<td></td>
<td>0.85919</td>
</tr>
<tr>
<td>13</td>
<td>★★★★★</td>
<td></td>
<td></td>
<td>0.85899</td>
</tr>
<tr>
<td>14</td>
<td>TheMagicMarijuanaForest</td>
<td></td>
<td></td>
<td>0.85899</td>
</tr>
<tr>
<td>15</td>
<td>Training on the Test Data</td>
<td></td>
<td></td>
<td>0.85880</td>
</tr>
<tr>
<td>#</td>
<td>Δ pub</td>
<td>Team Name</td>
<td>Kernel</td>
<td>Team Members</td>
</tr>
<tr>
<td>----</td>
<td>-------</td>
<td>--------------------------------</td>
<td>------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>1</td>
<td>~20</td>
<td>A Clever Name is Hard</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>~2</td>
<td>Jim Wants Internships</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>~7</td>
<td>🤔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>~3</td>
<td>Sarcastic Gradient Dissent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>~2</td>
<td>Yeet;)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>~1</td>
<td>quackle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>~7</td>
<td>TheMagicMarijuanaForest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>~33</td>
<td>Transparent Learning Model</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>~3</td>
<td>======A+ LEVEL======</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>~47</td>
<td>The Best Ever CS155 Machine...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>~23</td>
<td>Numpy Dumpty</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>~7</td>
<td>trust the process</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>~11</td>
<td>KDT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>~6</td>
<td>Not Hotdog</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>~2</td>
<td>🌟🌟🌟🌟🌟</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Today

• Some useful matrix properties
  – Useful for Homework 5

• Latent Factor Models
  – Low-rank models with missing values

• Non-negative matrix factorization
Recap: Orthogonal Matrix

• A matrix $U$ is orthogonal if $UU^T = U^TU = I$
  – For any column $u$: $u^Tu = 1$
  – For any two columns $u$, $u'$: $u^Tu' = 0$
  – $U$ is a rotation matrix, and $U^T$ is the inverse rotation
  – If $x' = U^Tx$, then $x = Ux'$

Lecture 11: Latent Factor Models & Non-Negative Matrix Factorization
Recap: Orthogonal Matrix

• Any subset of columns of $U$ defines a subspace

\[ x' = U_{1:K}^T x \]

Transform into new coordinates
Treat $U_{1:K}$ as new axes

\[ \text{proj}_{U_{1:K}}(x) = U_{1:K} U_{1:K}^T x \]

Project $x$ onto $U_{1:K}$ in original space
“Low Rank” Subspace
Recap: Singular Value Decomposition

\[ X = \begin{bmatrix} x_1, \ldots, x_N \end{bmatrix} \in \mathbb{R}^{D \times N} \]

\[ X = U \Sigma V^T \]

Orthogonal Diagonal

\[ \sum_{i=1}^{N} \left\| x_i - U_{1:K} U_{1:K}^T x_i \right\|^2 \]

“Residual”

\[ U_{1:K} \] is the K-dim subspace with smallest residual
Recap: SVD & PCA

\[ XX^T = U \Lambda U^T \]

- **Orthogonal**
- **Diagonal**

**PCA**

\[ X = U \Sigma V^T \]

- **Orthogonal**

**SVD**

\[ XX^T = (U \Sigma V^T)(U \Sigma V^T)^T = U \Sigma V^T V \Sigma U^T = U \Sigma^2 U^T \]
Recap: Eigenfaces

- Each col of $U'$ is an “Eigenface”
- Each col of $V'^T = \text{coefficients of a student}$

$X' = U'$

Avg Face

225000-dimensional!
Matrix Norms

• Frobenius Norm

$$\|X\|_{Fro} = \sqrt{\sum_{ij} X_{ij}^2} = \sqrt{\sum_d \sigma_d^2}$$

• Trace Norm

$$\|X\|_* = \sum_d \sigma_d = \text{trace} \left( \sqrt{X^T X} \right)$$

Each $\sigma_d$ is guaranteed to be non-negative
By convention: $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_D \geq 0$
Properties of Matrix Norms

\[ \|X\|_{\text{Fro}}^2 = \text{trace}(X^T X) = \text{trace}\left( (U \Sigma V^T)^T U \Sigma V^T \right) \]
\[ = \text{trace}(V \Sigma^2 V^T) = \text{trace}(\Sigma^2 V^T V) \]
\[ = \text{trace}(\Sigma^2) = \sum_{d} \sigma_d^2 \]

Each \( \sigma_d \) is guaranteed to be non-negative
By convention: \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_D \geq 0 \)

\[ \text{trace}(ABC) = \text{trace}(BCA) = \text{trace}(CAB) \]
Properties of Matrix Norms

\[ \|X\|_* = \text{trace}\left( \sqrt{\left( U \Sigma V^T \right)^T U \Sigma V^T } \right) = \text{trace}\left( \sqrt{V \Sigma U^T U \Sigma V^T } \right) = \text{trace}\left( \sqrt{V \Sigma \Sigma V^T } \right) = \text{trace}\left( \sqrt{V \Sigma^2 V^T } \right) = \text{trace}\left( V \Sigma V^T \right) = \text{trace}(\Sigma) = \sum_d \sigma_d \]

Each \( \sigma_d \) is guaranteed to be non-negative
By convention: \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_D \geq 0 \)

\[ X = U \Sigma V^T \]

\[ \Sigma = \begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \ddots \\ & & & \sigma_D \end{bmatrix} \]

\[ \text{trace}(ABC) = \text{trace}(BCA) = \text{trace}(CAB) \]
Frobenius Norm = Squared Norm

• Matrix version of L2 Norm:

\[ \|X\|_{Fro}^2 = \sum_{ij} X_{ij}^2 = \sum_d \sigma_d^2 \]

• Useful for regularizing matrix models

\[ X = U\Sigma V^T \]

\[ \Sigma = \begin{bmatrix} \sigma_1 & \sigma_2 & \cdots & \sigma_D \end{bmatrix} \]
Recall: L1 & Sparsity

• w is sparse if mostly 0’s:
  – Small L0 Norm

\[ \|w\|_0 = \sum_d 1[w_d \neq 0] \]

• Why not L0 Regularization?
  – Not continuous!

\[ \arg\min_w \lambda \|w\|_0 + \sum_{i=1}^{N} (y_i - w^T x_i)^2 \]

• L1 induces sparsity
  – And is continuous!

\[ \arg\min_w \lambda |w| + \sum_{i=1}^{N} (y_i - w^T x_i)^2 \]

Omitting b & for simplicity
Trace Norm = L1 of Eigenvalues

- A matrix $X$ is considered low rank if it has few non-zero singular values:

$$
\|X\|_{\text{Rank}} = \sum_d 1_{[\sigma_d > 0]}
$$

$$
\|X\|_* = \sum_d \sigma_d = \text{trace}\left(\sqrt{X^T X}\right)
$$

aka “spectral sparsity”

$$
X = U \Sigma V^T
$$

$$
\Sigma = \begin{bmatrix}
\sigma_1 \\
\sigma_2 \\
\vdots \\
\sigma_D
\end{bmatrix}
$$

Not continuous!
Other Useful Properties

• Cauchy Schwarz:

\[ \langle A, B \rangle^2 = \text{trace}(A^T B)^2 \leq \langle A, A \rangle \langle B, B \rangle = \text{trace}(A^T A) \text{trace}(B^T B) = \|A\|^2_F \|B\|^2_F. \]

• AM-GM Inequality:

\[ \|A\| \|B\| = \sqrt{\|A\|^2 \|B\|^2} \leq \frac{1}{2} \left( \|A\|^2 + \|B\|^2 \right) \quad \text{True for any norm} \]

• Orthogonal Transformation Invariance of Norms:

\[ \|UA\|_F = \|A\|_F \quad \|UA\|_* = \|A\|_* \quad \text{If U is a full-rank orthogonal matrix} \]

• Trace Norm of Diagonals

\[ \|A\|_* = \sum_i |A_{ii}| \quad \text{If A is a square diagonal matrix} \]
Recap: SVD & PCA

• SVD: \[ X = U \Sigma V^T \]

• PCA: \[ XX^T = U \Sigma^2 U^T \]

• The first K columns of U are the best rank-K subspace that minimizes the Frobenius norm residual:
  \[ \left\| X - U_{1:K} U_{1:K}^T X \right\|_{Fro}^2 \]
Latent Factor Models
Netflix Problem

- $Y_{ij} = \text{rating user } i\text{ gives to movie } j$
- Solve using SVD!

$Y = \begin{bmatrix} \mathbf{U} \\
\mathbf{V}^T \end{bmatrix}$

$\mathbf{U} \in \mathbb{R}^{M \times K}$

$\mathbf{V}^T \in \mathbb{R}^{K \times N}$

$y_{ij} \approx u^T_i v_j$
Example

\[ y_{ij} \approx u_i^T v_j \]

Actual Netflix Problem

- Many missing values!

\[ Y \] (missing values) = \[ UV \]

M Users \( \times \) N Movies = M \( \times \) K \( \times \) N

“Latent Factors”
Collaborative Filtering

• M Users, N Items
• Small subset of user/item pairs have ratings
• Most are missing

• Applicable to any user/item rating problem
  – Amazon, Pandora, etc.

• **Goal:** Predict the missing values.
Latent Factor Formulation

• Only labels, no features

\[ S = \{ y_{ij} \} \]

• Learn a **latent** representation over users U and movies V such that:

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \sum_{ij} \left( y_{ij} - u_i^T v_j \right)^2
\]
Connection to Trace Norm

• Suppose we consider all U,V that achieve perfect reconstruction: \( Y = UV^T \)

• Find U,V with lowest complexity:

\[
\arg\min_{Y=UV^T} \frac{1}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right)
\]

• Complexity equivalent to trace norm:

\[
\|Y\|_* = \min_{Y=UV^T} \frac{1}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right)
\]
Proof (One Direction)

We will prove: \[ \|Y\|_* \geq \min_{Y=AB^T} \frac{1}{2} \left( \|A\|_{Fro}^2 + \|B\|_{Fro}^2 \right) \]

Choose: \[ A = U \sqrt{\Sigma}, \quad B = V \sqrt{\Sigma} \]

Then: \[ \min_{Y=AB^T} \frac{1}{2} \left( \|A\|_{Fro}^2 + \|B\|_{Fro}^2 \right) \leq \frac{1}{2} \left( \|U \sqrt{\Sigma}\|_{Fro}^2 + \|V \sqrt{\Sigma}\|_{Fro}^2 \right) \]

\[ = \frac{1}{2} \left( \text{trace} \left( \left( U \sqrt{\Sigma} \right)^T \left( U \sqrt{\Sigma} \right) \right) + \text{trace} \left( \left( V \sqrt{\Sigma} \right)^T \left( V \sqrt{\Sigma} \right) \right) \right) \]

\[ = \frac{1}{2} \left( \text{trace} \left( \sqrt{\Sigma} U^T U \sqrt{\Sigma} \right) + \text{trace} \left( \sqrt{\Sigma} V^T V \sqrt{\Sigma} \right) \right) \]

\[ = \frac{1}{2} \left( \text{trace} \left( \sqrt{\Sigma} \sqrt{\Sigma} \right) + \text{trace} \left( \sqrt{\Sigma} \sqrt{\Sigma} \right) \right) \]

\[ = \frac{1}{2} \left( \text{trace} \left( \Sigma \right) + \text{trace} \left( \Sigma \right) \right) = \text{trace} \left( \Sigma \right) = \|Y\|_* \]
Interpreting Model

• Latent-Factor Model Objective

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \sum_{ij} (y_{ij} - u_i^T v_j)^2
\]

• Related to:

\[
\arg\min_W \lambda \|W\|_* + \sum_{ij} (y_{ij} - w_{ij})^2
\]

Find the best low-rank approximation to Y!

\[
\|W\|_* = \min_{W=UVT} \frac{1}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right)
\]

Equivalent when U,V = rank of W
User/Movie Symmetry

\[
\text{argmin}_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \sum_{ij} \left( y_{ij} - u_i^T v_j \right)^2
\]

- If we knew V, then linear regression to learn U
  – Treat V as features

- If we knew U, then linear regression to learn V
  – Treat U as features
Optimization

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \sum_{ij} \omega_{ij} \left( y_{ij} - u_i^T v_j \right)^2 \quad \omega_{ij} \in \{0,1\}
\]

• Only train over observed \( y_{ij} \)

• Two ways to Optimize
  – Gradient Descent
  – Alternating optimization
    • Closed Form (for each sub-problem)
  – Homework question
Gradient Calculation

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_{ij} \omega_{ij} \left( y_{ij} - u_i^T v_j \right)^2
\]

\[
\partial_{u_i} = \lambda u_i - \sum_j \omega_{ij} v_j \left( y_{ij} - u_i^T v_j \right)^T
\]

Closed Form Solution (assuming V fixed):

\[
u_i = \left( \lambda I_K + \sum_j \omega_{ij} v_j v_j^T \right)^{-1} \left( \sum_j \omega_{ij} y_{ij} v_j \right)
\]
Gradient Descent Options

• Stochastic Gradient Descent
  – Update all model parameters for single data point

• Alternating SGD:
  – Update a single column of parameters at a time

\[ u_i = u_i - \eta \partial_{u_i} \]

\[ \partial_{u_i} = \lambda u_i - \sum_j \omega_{ij} v_j (y_{ij} - u_i^T v_j) \]
Alternating Optimization

• Initialize U & V randomly
• Loop
  – Choose next $u_i$ or $v_j$
  – Solve optimally:

\[
  u_i = \left( \lambda I_K + \sum_j \omega_{ij} v_j v_j^T \right)^{-1} \left( \sum_j \omega_{ij} y_{ij} v_j \right)
\]

• (assuming all other variables fixed)
Tradeoffs

• Alternating optimization much faster in terms of #iterations
  – But requires inverting a matrix:
    \[
    u_i = \left( \lambda I_K + \sum_j \omega_{ij} v_j v_j^T \right)^{-1} \left( \sum_j \omega_{ij} y_{ij} v_j \right)
    \]

• Gradient descent faster for high-dim problems
  – Also allows for streaming data
    \[
    u_i = u_i - \eta \partial_{u_i}
    \]
Recap: Collaborative Filtering

- **Goal**: predict every user/item rating

- **Challenge**: only a small subset observed

- **Assumption**: there exists a low-rank subspace that captures all the variability in describing different users and items
Aside: Multitask Learning

- M Tasks:

\[
S^m_m = \left\{ (x_i, y^m_{i}) \right\}_{i=1}^{N}
\]

\[
\arg\min \frac{\lambda}{2} R(W) + \frac{1}{2} \sum_m \sum_i (y_i - w^T_m x_i)^2
\]

Regularizer

- Example: personalized recommender system
  - One task per user:
How to Regularize?

\[
\arg\min_W \frac{\lambda}{2} R(W) + \frac{1}{2} \sum_m \sum_i \left( y_i - w_m^T x_i \right)^2 \quad S^m = \left\{ (x_i, y_i^m) \right\}_{i=1}^N
\]

• **Standard L2 Norm:**

\[
\arg\min_W \frac{\lambda}{2} \|W\|^2 + \sum_m \sum_i \left( y_i - w_m^T x_i \right)^2 = \sum_m \left[ \frac{\lambda}{2} \|w_m\|^2 + \sum_i \left( y_i - w_m^T x_i \right)^2 \right]
\]

• **Decomposes to independent tasks**
  – For each task, learn D parameters
How to Regularize?

\[
\arg\min_W \frac{\lambda}{2} R(W) + \frac{1}{2} \sum_m \sum_i \left( y_i - w_m^T x_i \right)^2
\]

\[
S^m = \left\{ (x_i, y_i^m) \right\}_{i=1}^N
\]

• Trace Norm:

\[
\arg\min_W \frac{\lambda}{2} \| W \|_* + \sum_m \sum_i \left( y_i - w_m^T x_i \right)^2
\]

• Induces W to have low rank across all task
Recall: Trace Norm & Latent Factor Models

• Suppose we consider all $U,V$ that achieve perfect reconstruction: $W=UV^T$

• Find $U,V$ with lowest complexity:

$$\arg\min_{W=UV^T} \frac{1}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right)$$

• Complexity equivalent to trace norm:

$$\|W\|_* = \min_{W=UV^T} \frac{1}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right)$$
How to Regularize?

\[
\text{argmin}_W \frac{\lambda}{2} R(W) + \frac{1}{2} \sum_m \sum_i (y_i - w_m^T x_i)^2 \quad S^m = \left\{ (x_i, y_i^m) \right\}_{i=1}^N
\]

- Latent Factor Approach

\[
\text{argmin}_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_m \sum_i (y_i - u_m^T V x_i)^2
\]

- Learns a feature projection \( x' = Vx \)
- Learns a K dimensional model per task
Tradeoff

• **D*N parameters:**

\[
\arg\min_{w} \sum_{m} \left[ \frac{\lambda}{2} \|w_m\|^2 + \frac{1}{2} \sum_{i} (y_i - w_m^T x_i)^2 \right]
\]

• **D*K + N*K parameters:**

\[
\arg\min_{U, V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_{m} \sum_{i} (y_i - u_m^T V x_i)^2
\]

– Statistically more efficient
– Great if low-rank assumption is a good one
Multitask Learning

- **M Tasks:**
  
  \[ S^m = \left\{ (x_i, y^m_i) \right\}_{i=1}^N \]
  
  \[
  \arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_m \sum_i \left( y^m_i - u^T_m V x_i \right)^2
  \]

- **Example:** personalized recommender system
  
  - One task per user:
  
  - If \( x \) is topic feature representation
    
    - \( V \) is subspace of correlated topics
    
    - Projects multiple topics together
Reduction to Collaborative Filtering

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_m \sum_i \left( y_{im} - u_m^T v_{ix} \right)^2
\]

\[
S^m = \{(x_i, y_{im})\}_{i=1}^N
\]

- Suppose each \(x_i\) is single indicator \(x_i = e_i\)
- Then: \(Vx_i = v_i\)
- Exactly Collaborative Filtering!

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_m \sum_i \left( y_{im} - u_m^T v_i \right)^2
\]

\[
x_i = \begin{bmatrix}
\vdots \\
0 \\
1 \\
0 \\
\vdots
\end{bmatrix}
\]
Latent Factor Multitask Learning vs Collaborative Filtering

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_m \sum_i \left( y_i^m - u_m^T V x_i \right)^2
\]

- Projects \( x \) into low-dimensional subspace \( V x \)
- Learns low-dimensional model per task

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_m \sum_i \left( y_i^m - u_m^T V v_i \right)^2
\]

- Creates low dimensional feature for each movie
- Learns low-dimensional model per user
General Bilinear Models

\[ \arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \sum_i \left( y_i - z_i^T U^T V x_i \right)^2 \quad S = \{(x_i, z_i, y_i)\} \]

- Users described by features \( z \)
- Items described by features \( x \)

- Learn a projection of \( z \) and \( x \) into common low-dimensional space
  - Linear model in low dimensional space
Why are Bilinear Models Useful?

\[
\begin{align*}
\text{argmin}_{U,V} & \quad \frac{\lambda}{2} \left( \|U\|_F^2 + \|V\|_F^2 \right) + \frac{1}{2} \sum_m \sum_i (y_i - u_m^T v_i)^2 \\
U & : \text{MxK} \\
V & : \text{NxK}
\end{align*}
\]

\[
\begin{align*}
\text{argmin}_{U,V} & \quad \frac{\lambda}{2} \left( \|U\|_F^2 + \|V\|_F^2 \right) + \frac{1}{2} \sum_m \sum_i (y_i - u_m^T V x_i)^2 \\
U & : \text{MxK} \\
V & : \text{DxK}
\end{align*}
\]

\[
\begin{align*}
\text{argmin}_{U,V} & \quad \frac{\lambda}{2} \left( \|U\|_F^2 + \|V\|_F^2 \right) + \frac{1}{2} \sum_i (y_i - z_i^T U^T V x_i)^2 \\
U & : \text{FxK} \\
V & : \text{DxK}
\end{align*}
\]

\[S = \{(x_i, z_i, y_i)\}\]
Story So Far: Latent Factor Models

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \frac{1}{2} \sum_i \left( y_i - z_i U^T V x_i \right)^2 \quad S = \{(x_i, z_i, y_i)\}
\]

- **Simplest Case:** reduces to SVD of matrix Y
  - No missing values
  - \((z,x)\) indicator features
- **General Case:** projects high-dimensional feature representation into low-dimensional linear model
Non-Negative Matrix Factorization
Limitations of PCA & SVD

All features non-negative

PCA/SVD Solution

Better Solution?
Non-Negative Matrix Factorization

• Assume $Y$ is non-negative
• Find non-negative $U$ & $V$

$Lecture 11: Latent Factor Models & Non-Negative Matrix Factorization$
CS 155 Non-Negative Face Basis

Lecture 11: Latent Factor Models & Non-Negative Matrix Factorization
CS 155 Eigenfaces

Lecture 10: Clustering & Dimensionality Reduction
Aside: Non-Orthogonal Projections

• If columns of $A$ are not orthogonal, $A^T A \neq I$
  
  – How to reverse transformation $x' = A^T x$?
  
  – Solution: Pseudoinverse!

\[
A = U \Sigma V^T \\
\text{SVD}
\]

\[
A^+ = V \Sigma^+ U^T \\
\text{Pseudoinverse}
\]

\[
\Sigma^+ = \begin{bmatrix} \sigma_1 & 0 & \ldots & 0 \\ 0 & \sigma_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & \sigma_p \end{bmatrix}, \\
\sigma^+ = \begin{cases} 
1/\sigma & \text{if } \sigma > 0 \\
0 & \text{otherwise}
\end{cases}
\]

Intuition: use the rank-K orthogonal basis that spans $A$.

\[
A^+ A^T x = U \Sigma^+ V^T V \Sigma U^T x \\
= U_{1:K} U_{1:K}^T x
\]
Objective Function

\[
\text{argmin}_{U \geq 0, V \geq 0} \sum_{ij} \ell(y_{ij}, u_i^T v_j)
\]

• Squared Loss:
  – Penalizes squared distance

\[
\ell(a, b) = (a - b)^2
\]

• Generalized Relative Entropy
  – Aka, unnormalized KL divergence
  – Penalizes ratio

\[
\ell(a, b) = a \log \frac{a}{b} - a + b
\]

• Train using gradient descent

SVD/PCA vs NNMF

• **SVD/PCA:**
  
  – Finds the best orthogonal basis faces
  
  • Basis faces can be neg.
  
  – Coeffs can be negative
  
  – Often trickier to visualize
  
  – Better reconstructions with fewer basis faces
    
    • Basis faces capture the most variations

• **NNMF:**
  
  – Finds best set of non-negative basis faces
  
  – Non-negative coeffs
    
    • Often non-overlapping
  
  – Easier to visualize
  
  – Requires more basis faces for good reconstructions
Non-Negative Latent Factor Models

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 \right) + \sum_i \ell(y_i, z_i^T U^T V x_i) 
\]

\[S = \{(x_i, z_i, y_i)\}\]

- **Simplest Case:** reduces to NNMF of matrix Y
  - No missing values
  - \((z,x)\) indicator features

- **General Case:** projects high-dimensional non-negative features into low-dimensional non-negative linear model
Modeling NBA Gameplay Using Non-Negative Spatial Latent Factor Models
Fine-Grained Spatial Models

- Discretize court
  - 1x1 foot cells
  - 2000 cells
- 1 weight per cell
  - 2000 weights

\[ F_s(x) : \]

Lecture 11: Latent Factor Models & Non-Negative Matrix Factorization
Fine-Grained Spatial Models

- Discretize court
  - 1x1 foot cells
  - 2000 weights
- 1 weight per cell
  - 2000 cells

But most players haven’t played that much!

$$F_s(x)$$:
When across the basketball court. We see that the spatial latent factor offers significant improvement. The gain from adding a pass event. These results are depicted in the second column. We evaluated using both ROC curves as well as average log loss, and we observe that our latent factor learns by our model, we also evaluate predictive performance on a pass event – about 70% of frames have no action events forthcoming in the next 1 second. To further tease out the interesting effects, we comparing only passing events (bottom row), we observe a significant error reduction in the two “low post” locations. We comparing all events (top row), we observe a significant error reduction around the basket area. When comparing action events (middle row), we observe a significant error reduction in the two “low post” locations.

Section VI will shed light on how components of our model contribute to the various reductions in error.

We now provide a detailed inspection our learned model. Each player’s conditional shooting probabilities can be represented as a combination of these factors. This type of analysis is applicable to a wide range of domains.

Figure 4(b) depicts how predictive performance varies with fine-grained spatial patterns. This degree angles away from the basket.

Visualizing location factors L

Visualizing players $B_b L$

Training Data

STATS SportsVU
2012/2013 Season, 630 Games,
80K Possessions, 380 frames per possession
Prediction

- **Game state: \( x \)**
  - Coordinates of all players
  - Who is the ball handler

- **Event: \( y \)**
  - Ball handler will shoot
  - Ball handler will pass (to whom?)
  - Ball handler will hold onto the ball
  - 6 possibilities

- **Goal:** Learn \( P(y|x) \)

Logistic Regression
(Simple Version: Just for Shooting)

\[ P(y \mid x) = \frac{\exp\{F(y \mid x)\}}{Z(x \mid F)} \]

\[ Z(x \mid F) = \sum_{y' \in \{s, \bot\}} \exp\{F(y' \mid x)\} \]

\[ F(y' \mid x) = \begin{cases} 
F_s(x) & y' = s \\
F_\bot & y' = \bot 
\end{cases} \]

- Shot
- Hold on to ball

Offset or bias

\[ P(y = s \mid x) = \frac{1}{1 + \exp\{-F_s(x) + F_\bot\}} \]

Lecture 11: Latent Factor Models & Non-Negative Matrix Factorization
Learning the Model

• Given training data:

\[ S = \{(x, y)\} \]

• Learn parameters of model:

\[
\arg\min_{F_s, F_\perp} \frac{\lambda}{2} \|F_s\|^2 + \sum_{(x, y) \in S} \ell(y, F_s(x) - F_\perp)
\]

\[
P(y = s | x) = \frac{1}{1 + \exp\{-F_s(x) + F_\perp\}}
\]
Optimization via Gradient Descent

\[
\arg\min_{B \geq 0, L \geq 0, F_L} \frac{\lambda}{2} \left( \|B\|^2 + \|L\|^2 \right) + \sum_{(x,y)} \ell \left( y, B_b(x) L_l(x) - F_L \right)
\]

\[
\partial L_i = \lambda_i L_i - \sum_{(x,y)} \frac{\partial \log P(y \mid x)}{\partial L_i}
\]

\[
\frac{\partial \log P(y \mid x)}{\partial L_i} = \left( 1_{[y=s]} - P(s \mid x) \right) B_b(x)
\]


Lecture 11: Latent Factor Models & Non-Negative Matrix Factorization
Where are Players Likely to Receive Passes?

![Diagram of latent factor models and visualization of location factors](http://www.yisongyue.com/publications/icdm2014_bball_predict.pdf)

--

Visualization of Location Factors $M$

- Tony Parker
- Dirk Nowitzki
- LeBron James
- Monta Ellis
- Manu Ginobili
- David Lee
- Jose Calderon
- Chandler Parsons
- Goran Dragic
- Joachim Noah

Enforce Non-Negativity
(Accuracy Worse)
(More Interpretable)
How do passes tend to flow?


Lecture 11: Latent Factor Models & Non-Negative Matrix Factorization
How do passes tend to flow?

- The diagram shows a visual representation of how passes flow from one location to another.
- The top row of diagrams depicts the spatial coefficients of players which have high affinity for each of the factors.
- The bottom row shows the spatial coefficients of players behind the basket, which is a region of large error.
- The first factor corresponds to players that receive passes near the basket.
- The next three factors depict various midrange regions.
- The seventh factor corresponds to players who shoot near the basket.
- The eighth factor relates to players who shoot while attacking the basket from the baseline.
- The final two factors correspond to players who shoot midrange shots.


Lecture 11: Latent Factor Models & Non-Negative Matrix Factorization
Tensor Latent Factor Models
Tensor Factorization

\[ Y \times U = (\text{Missing Values}) \]
Tri-Linear Model

\[
\arg\min_{U,V,W} \frac{\lambda}{2} \left( \|U\|_{Fro}^2 + \|V\|_{Fro}^2 + \|W\|_{Fro}^2 \right) + \sum_i \ell(y_i, \langle U^T z_i, V^T x_i, W^T q_i \rangle)
\]

- Prediction via 3-way dot product: \[\langle a, b, c \rangle = \sum_{k} a_k b_k c_k\]
  - Related to Hadamard Product

- Example: online advertising
  - User profile \(z\)
  - Item description \(x\)
  - Query \(q\)

Solve using Gradient Descent
Summary: Latent Factor Models

• Learns a low-rank model of a matrix of observations $Y$
  – Dimensions of $Y$ can have various semantics

• Can tolerate missing values in $Y$

• Can also use features

• Widely used in industry
Next Lecture

• Embeddings

• Word2Vec