INTRODUCTION TO MATRIX FACTORIZATION METHODS COLLABORATIVE FILTERING

USER RATINGS PREDICTION

Alex Lin
Senior Architect
Intelligent Mining
Outline

- Factor analysis
- Matrix decomposition
- Matrix Factorization Model
- Minimizing Cost Function
- Common Implementation
Factor Analysis

- A procedure can help identify the factors that might be used to explain the interrelationships among the variables
- Model based approach
Refresher: Matrix Decomposition

\[
X_{32} = (a, b, c) \cdot (x, y, z) = a \cdot x + b \cdot y + c \cdot z
\]

**R**
5 x 6 matrix

<table>
<thead>
<tr>
<th>X_{11}</th>
<th>X_{12}</th>
<th>X_{13}</th>
<th>X_{14}</th>
<th>X_{15}</th>
<th>X_{16}</th>
</tr>
</thead>
<tbody>
<tr>
<td>X_{21}</td>
<td>X_{22}</td>
<td>X_{12}</td>
<td>X_{24}</td>
<td>X_{25}</td>
<td>X_{26}</td>
</tr>
<tr>
<td>X_{31}</td>
<td>X_{32}</td>
<td>X_{33}</td>
<td>X_{34}</td>
<td>X_{35}</td>
<td>X_{36}</td>
</tr>
<tr>
<td>X_{41}</td>
<td>X_{42}</td>
<td>X_{43}</td>
<td>X_{44}</td>
<td>X_{45}</td>
<td>X_{46}</td>
</tr>
<tr>
<td>X_{51}</td>
<td>X_{52}</td>
<td>X_{53}</td>
<td>X_{54}</td>
<td>X_{55}</td>
<td>X_{56}</td>
</tr>
</tbody>
</table>

**q**
5 x 3 matrix

| a | b | c |

**p**
3 x 6 matrix

<table>
<thead>
<tr>
<th>x</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>z</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \hat{r}_{ui} = q_i^T p_u \]
Making Prediction as Filling Missing Value

$$\hat{r}_{ui} = q_i^T p_u$$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>...</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td>4</td>
<td>4</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

users

items

Movie Preference Factor Vector

User Preference Factor Vector

Rating Prediction

proprietary material
Learn Factor Vectors

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>...</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>users</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>?</td>
<td>5</td>
<td>?</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>items</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: only train on known entries

\[
\begin{align*}
2X + 3Y &= 5 \\
4X - 2Y &= 2
\end{align*}
\]

\[
\begin{align*}
2X + 3Y &= 5 \\
4X - 2Y &= 2
\end{align*}
\]
Why not use standard SVD?

- Standard SVD assumes all missing entries are zero. This leads to bad prediction accuracy, especially when dataset is extremely sparse. (98% - 99.9%)
- See Appendix for SVD
- In some published literatures, they call Matrix Factorization as SVD, but note it’s NOT the same kind of classical low-rank SVD produced by svdlibc.
How to Learn Factor Vectors

- How do we learn preference factor vectors \((a, b, c)\) and 
  \((x, y, z)\)?
- Minimize errors on the known ratings

\[
\min_{q^*,p^*} \sum_{(u,i)\in k} (r_{ui} - x_{ui})^2
\]

To learn the factor vectors \((p_u \text{ and } q_i)\)

Minimizing Cost Function (Least Squares Problem)

\(r_{ui}\) : actual rating for user \(u\) on item \(I\)
\(x_{ui}\) : predicted rating for user \(u\) on item \(I\)
Data Normalization

- Remove Global mean

<table>
<thead>
<tr>
<th>items</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>...</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.5</td>
<td>-0.9</td>
<td>-0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.79</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>0.39</td>
<td></td>
<td>0.82</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.52</td>
<td></td>
</tr>
</tbody>
</table>

proprietary material
Factorization Model

- Only Preference factors

\[
\min_{q^*, p^*} \sum_{(u,i) \in k} (r_{ui} - \mu - q_i^T p_u)^2
\]

To learn the factor vectors (\(p_u\) and \(q_i\))

- \(r_{ui}\): actual rating of user \(u\) on item \(I\)
- \(\mu\): training rating average
- \(b_u\): user \(u\) user bias
- \(b_i\): item \(i\) item bias
- \(q_i\): latent factor array of item \(i\)
- \(p_u\): later factor array of user \(u\)
Adding Item Bias and User Bias

- Add Item bias and User bias as parameters

\[
\min_{q^*, p^*} \sum_{(u,i) \in k} (r_{ui} - \mu - b_i - b_u - q_i^T p_u)^2
\]

To learn Item bias and User bias

- \(r_{ui}\): actual rating of user \(u\) on item \(I\)
- \(\mu\): training rating average
- \(b_u\): user \(u\) user bias
- \(b_i\): item \(i\) item bias
- \(q_i\): latent factor array of item \(i\)
- \(p_u\): later factor array of user \(u\)
Regularization

To prevent model overfitting

\[
\min_{q^*,p^*} \sum_{(u,i) \in k} (r_{ui} - \mu - b_i - b_u - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2 + b_i^2 + b_u^2)
\]

- \(r_{ui}\): actual rating of user \(u\) on item \(I\)
- \(\mu\): training rating average
- \(b_u\): user \(u\) user bias
- \(b_i\): item \(i\) item bias
- \(q_i\): latent factor array of item \(i\)
- \(p_u\): later factor array of user \(u\)
- \(\lambda\): regularization Parameters

Rating = 4

Global Mean
Preference Factor
Item Bias
User Bias

proprietary material
Optimize Factor Vectors

- Find optimal factor vectors - minimizing cost function
- Algorithms:
  - Stochastic gradient descent
  - Others: Alternating least squares etc..
- Most frequently use:
  - Stochastic gradient descent
Matrix Factorization Tuning

- Number of Factors in the Preference vectors
- Learning Rate of Gradient Descent
  - Best result usually coming from different learning rate for different parameter. Especially user/item bias terms.
- Parameters in Factorization Model
  - Time dependent parameters
  - Seasonality dependent parameters
- Many other considerations!
High-Level Implementation Steps

- Construct User-Item Matrix (sparse data structure!)
- Define factorization model - Cost function
- Take out global mean
- Decide what parameters in the model. (bias, preference factor, anything else? SVD++)
- Minimizing cost function - model fitting
  - Stochastic gradient descent
  - Alternating least squares
- Assemble the predictions
- Evaluate predictions (RMSE, MAE etc..)
- Continue to tune the model
Thank you

- Any question or comment?
Appendix

- Stochastic Gradient Descent
- Batch Gradient Descent
- Singular Value Decomposition (SVD)
Stochastic Gradient Descent

Repeat Until Convergence {
    for i=1 to m in random order {
        \[ \theta_j := \theta_j + \alpha (y^{(i)} - h_\theta (x^{(i)})) x_j^{(i)} \] (for every j)
    }
}

Your code Here:
Batch Gradient Descent

Repeat Until Convergence {

\[ \theta_j := \theta_j + \alpha \sum_{i=1}^{m} (y^{(i)} - h_{\theta}(x^{(i)})) x^{(i)}_j \] (for every j)

}

Your code Here:
Singular Value Decomposition (SVD)

\[ A = U \times S \times V^T \]

\[ A_k = U_k \times S_k \times V_k^T \]