Step-By-Step Instructions for Miniproject 2
Matrix Factorization with Missing Values

- **Goal #1**: Learn a Latent Factor Model $U \& V$
- **Goal #2**: Visualize & Interpret $U \& V$ (mostly $V$)

$Lecture 13$: Latent Factor Models & Non-Negative Matrix Factorization
Final Product: Create Something Like This

You need to create your own visualization (will have different projection of movies/users onto 2-dimensional plane than example above)

You need to interpret your dimensions and/or clusters of movies in your projection

Outline

• **Step 1: Learn U & V**
  – If you are not confident in implementation, use off-the-shelf software first
  – Then implement your own solver if you feel like it

• **Step 2: Project U & V down to 2 dimensions**
  – Basically SVD in Matlab or Python

• **Step 3: Plot projected U & V**
  – Give your own interpretation of the two projected dimensions
Step 1: Learning U & V

Choice of regularization doesn’t matter too much

\[
\arg\min_{U,V} \frac{\lambda}{2} \left( \|U\|^2 + \|V\|^2 \right) + \sum_{(i,j) \in S} \left( Y_{i,j} - u_i^T v_j \right)^2
\]

You don’t have to solve this exact objective. (many off-the-shelf solve something related.)

• Three options:
  
  – Stochastic Gradient Descent
    • Each step of SGD considers single index \((i,j)\) of \(S\)
  
  – Alternating Minimization
    • Each step completely solves \(U\) or \(V\) while holding the other fixed.
  
  – Use off-the-shelf software
    • Will only get 20/40 of this portion of the question
Off-the-Shelf Software

- Search for “Collaborative Filtering Matlab” or “Collaborative Filtering Python” or “Collaborative Filtering code”

  - http://spark.apache.org/docs/1.0.0/mllib-collaborative-filtering.html
  - http://select.cs.cmu.edu/code/graphlab/pmf.html
Step 1b: Learning U & V

(More Advanced)

Choice of regularization doesn’t matter too much

$$\arg\min_{U,V,a,b} \frac{\lambda}{2} (\|U\|^2 + \|V\|^2) + \sum_{(i,j) \in S} \left( Y_{i,j} - (u_i^T v_j + a_i + b_j) \right)^2$$

- Model the global tendency of a movie’s average rating
- Model the global tendency of how a user rates on average
- This keeps U & V more focused on variability between users and movies.
- Should be an option that you can turn on in many off-the-shelf implementations

$S =$ set of indices $(i,j)$ of observed ratings

Vector of bias/offset terms One for each user & movie
Step 1c: Learning U & V
(Even More Advanced)

Choice of regularization doesn’t matter too much

$$\arg\min_{U,V,a,b} \frac{\lambda}{2} (\|U\|^2 + \|V\|^2 + \|a\|^2 + \|b\|^2) + \sum_{(i,j) \in S} \left( (Y_{i,j} - \mu) - (u_i^T v_j + a_i + b_j) \right)^2$$

$S = \text{set of indices } (i,j) \text{ of observed ratings}$

$\mu$ is average of all observations in Y

Vector of bias/offset terms
One for each user & movie

• Model global bias $\mu$ as average over all observed $Y$
• Treat $a$ as user-specific deviation from global bias
• Treat $b$ as movie-specific deviation from global bias
• Should be an option that you can turn on in many off-the-shelf implementations
Step 1: Interpretation

- Common K-dimensional representation over users & movies
  - Rating defined by dot product (aka un-normalized cosine similarity):
    \[ Y_{i,j} \approx u_i^T v_j \] or \[ Y_{i,j} \approx u_i^T v_j + a_i + b_j \]

- Does our representation make sense? (i.e., is it interpretable?)
  - Need to visualize!
  - But can only (easily) visualize 2-dim points, not K-dim points!
Step 2: Projecting U & V to 2 Dimensions

• Step 2a:
  – (Optional) mean center V: each row of V has zero mean
  – Compute SVD of V: \[ V = A\Sigma B^T \]
  – The first two columns of A correspond to best 2-dimensional projection of movies V
Step 2: Projecting U & V to 2 Dimensions

• Step 2b:
  
  – Project every movie & user using $A_{1:2}$

  \[
  \tilde{V} = A_{1:2}^T V \in \mathbb{R}^{2 \times N}
  \]

  \[
  \tilde{U} = A_{1:2}^T U \in \mathbb{R}^{2 \times M}
  \]

  If you mean centered V, you need to shift U by same amount first

  – Now each user & movie is represented using a two dimensional point. Visualize and interpret!
Step 2: Projecting $U$ & $V$ to 2 Dimensions

- Step 2c (optional):
  - Do Steps 2a & 2b:  
    \[ \tilde{V} = A_{1:2}^T V \in \mathbb{R}^{2 \times N} \]
    \[ \tilde{U} = A_{1:2}^T U \in \mathbb{R}^{2 \times M} \]

- Then rescale dimensions:
  - E.g., each row of $\tilde{U}$ has unit variance.
  - Otherwise, visualization might look stretched:
Step 2: Interpretation

• The top D dimensions of matrix A defines a D-dim projection that best preserves the learned movie features V:

\[ \tilde{v}_j = A_{1:D}^T v_j \]

Minimizes loss of feature representation:

\[ \sum_j \| v_j - A_{1:D} \tilde{v}_j \|^2 \]

Projected representation

Preservation Loss of projection

• We want 2-dimensional projection for visualization purposes
  – So we take top 2 dimensions of SVD

• Now we can visualize movies in 2D plot
  – And see if close-by movies have similar sementics
  – E.g., horror, action, etc.
Step 2: Alternatives & Core Requirements

• You don’t have to do it the above way
  – Although the above method should always give you something reasonable to visualize

• Core requirement:
  – Projection should preserve as much of the original features as possible
  – A dot product in the 2-D representation should approximate the dot product in the K-D representation
Step 3: Plot U & V

• Plotting V is more important:
  – Pick a few movies and plot their projected 2D representation
  – Verify that distances/angles/axes in your plot can be interpreted

Example:

(Your visualization will probably not be as clean as this one, that is OK)

• Can also plot the genres provided:
  – E.g., where is the average horror movie?
  – E.g., compute the average v for all movies that belong to horror genre
My Own Example

Trained using
Step 1c (lambda=10)
Stochastic GD

SVD of Movie Matrix
Project top 2 bases

Picked a few popular
movies, and plotted them.

Then found a few extreme
points (e.g., Clockwork
Orange).

Removed most children’s
movies (didn’t seem to
project well using 1st two
SVD bases – maybe most
ratings are by adults).