Lessons from the Netflix Prize

Yehuda Koren
The BellKor team
(with Robert Bell & Chris Volinsky)

We Know What You Ought To Be Watching This Summer

Netflix Prize

- Training data
  - 100 million ratings
  - 480,000 users
  - 17,770 movies
  - 6 years of data: 2000-2005
- Test data
  - Last few ratings of each user (2.8 million)
  - Evaluation criterion: root mean squared error (RMSE)
  - Netflix Cinematch RMSE: 0.9514
- Competition
  - 2700+ teams
  - $1 million grand prize for 10% improvement on Cinematch result
  - $50,000 2007 progress prize for 8.43% improvement

Overall rating distribution

- Third of ratings are 4s
- Average rating is 3.68

From TimelyDevelopment.com

#ratings per movie

- Avg #ratings/movie: 5627
#ratings per user

* Avg #ratings/user: 208

Average movie rating by movie count

- More ratings to better movies

## Most loved movies

<table>
<thead>
<tr>
<th>Title</th>
<th>Avg rating</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Shawshank Redemption</td>
<td>4.593</td>
<td>137812</td>
</tr>
<tr>
<td>Lord of the Rings: The Return of the King</td>
<td>4.545</td>
<td>133597</td>
</tr>
<tr>
<td>The Green Mile</td>
<td>4.306</td>
<td>180883</td>
</tr>
<tr>
<td>Lord of the Rings: The Two Towers</td>
<td>4.460</td>
<td>130876</td>
</tr>
<tr>
<td>Finding Nemo</td>
<td>4.415</td>
<td>139650</td>
</tr>
<tr>
<td>Raiders of the Lost Ark</td>
<td>4.504</td>
<td>117456</td>
</tr>
<tr>
<td>Forrest Gump</td>
<td>4.299</td>
<td>180736</td>
</tr>
<tr>
<td>Lord of the Rings: The Fellowship of the ring</td>
<td>4.433</td>
<td>147932</td>
</tr>
<tr>
<td>The Sixth Sense</td>
<td>4.325</td>
<td>149199</td>
</tr>
<tr>
<td>Indiana Jones and the Last Crusade</td>
<td>4.333</td>
<td>144027</td>
</tr>
</tbody>
</table>

## Important RMSEs

- **Global average**: 1.1296
- **User average**: 1.0651
- **Movie average**: 1.0533

### Personalization

- **Cinematch**: 0.9514; baseline
- **BellKor**: 0.8693; 8.63% improvement
- **Grand Prize**: 0.8563; 10% improvement
- **Inherent noise**: ???

## Challenges

- **Size of data**
  - Scalability
  - Keeping data in memory
- **Missing data**
  - 99 percent missing
  - Very imbalanced
- **Avoiding overfitting**
- **Test and training data differ significantly**

## The BellKor recommender system

- Use an ensemble of complementing predictors
- **Two, half tuned** models worth more than a **single, fully tuned** model
The BellKor recommender system

- Use an ensemble of complementing predictors
- **Two, half tuned** models worth more than a **single, fully tuned** model
- But: Many seemingly different models expose similar characteristics of the data, and won’t mix well
- Concentrate efforts along three axes...

The three dimensions of the BellKor system

- **Global effects**
  - Multi-scale modeling of the data
  - Combine top level, regional modeling of the data, with a refined, local view:
    - k-NN: Extracting local patterns
    - Factorization: Addressing regional effects

Multi-scale modeling – 1st tier

- **Global effects:**
  - Mean rating: 3.7 stars
  - The Sixth Sense is 0.5 stars above avg
  - Joe rates 0.2 stars below avg
  - Baseline estimation: Joe will rate The Sixth Sense 4 stars

Multi-scale modeling – 2nd tier

- **Factors model:**
  - Both The Sixth Sense and Joe are placed high on the “Supernatural Thrillers” scale
  - Adjusted estimate: Joe will rate The Sixth Sense 4.5 stars

Multi-scale modeling – 3rd tier

- **Neighborhood model:**
  - Joe didn’t like related movie Signs
  - Final estimate: Joe will rate The Sixth Sense 4.2 stars

Important technique: Shrinking

- Data is very sparse, and many parameters need to be estimated with small number of observations
- Shrinking: (in estimating s with n observations)
  \[
  s' = \frac{ns}{n + \beta}
  \]
- Can be justified using a Bayesian approach
First step: removing global effects

- Some users systematically give higher scores.
- Some movies have better ratings on average (simply good movies.)
- Take into account time effects (not covered in detail):
  - Movies may go out of fashion
  - People may change their tastes

Local level: k-NN

- Earliest and most popular collaborative filtering method
- Derive unknown ratings from those of “similar” items (movie-movie variant)
- A parallel user-user flavor: rely on ratings of like-minded users (not in this talk)

**k-NN**

```
users
<table>
<thead>
<tr>
<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>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**k-NN**

```
users
<table>
<thead>
<tr>
<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>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**Neighbor selection:**
Identify movies similar to 1, rated by user 5

**Compute similarity weights:**
$s_{13}=0.2$, $s_{16}=0.3$
k-NN - Common practice
1. Define a similarity measure between items: $s_{ij}$
2. Select neighbors -- $N(i;u)$: items most similar to $i$, that were rated by $u$
3. Estimate unknown rating, $r_{ui}$, as the weighted average:

$$r_{ui} = b_u + \sum_{j \in N(i;u)} \left( s_{ij} \left( r_{uj} - b_j \right) \right) \sum_{j \in N(i;u)} s_{ij}$$

baseline estimate for $r_{ui}$

Problems:
1. Similarity measures are arbitrary; no fundamental justification
2. Pairwise similarities isolate each neighbor; neglect interdependencies among neighbors
3. Taking a weighted average is restricting; e.g., when neighborhood information is limited

Their approach: solving an optimization problem
- Use a weighted sum rather than a weighted average:

$$r_{ui} = b_u + \sum_{j \in N(i;u)} w_j \left( r_{uj} - b_j \right)$$

(We allow $\sum_{j \in N(i;u)} w_j = 1$)

- Model relationships between item $i$ and its neighbors
- Can be learnt through a least squares problem from all other users that rated $i$:

$$\min_w \sum_{i \in U} \left( r_{ui} - b_u - \sum_{j \in N(i;u)} w_j \left( r_{uj} - b_j \right) \right)^2$$

- Interpolation weights derived based on their role; no use of an arbitrary similarity measure
- Explicitly account for interrelationships among the neighbors

Challenges:
- Deal with missing values
- Avoid overfitting
- Efficient implementation

Interpolation weights
- Estimate inner-products among movie ratings

k-NN on the RMSE scale
- Global average: 1.1296
- User average: 1.0651
- Movie average: 1.0533
- Cinematch: 0.9514
- BellKor: 0.8693
- Grand Prize: 0.8563
- Inherent noise: ????

k-NN - Common practice
1. Define a similarity measure between items: $s_{ij}$
2. Select neighbors -- $N(i;u)$: items similar to $i$, rated by $u$
3. Estimate unknown rating, $r_{ui}$, as the weighted average:

$$r_{ui} = b_u + \sum_{j \in N(i;u)} \left( s_{ij} \left( r_{uj} - b_j \right) \right) \sum_{j \in N(i;u)} s_{ij}$$

Problems:
1. Similarity measures are arbitrary; no fundamental justification
2. Pairwise similarities isolate each neighbor; neglect interdependencies among neighbors
3. Taking a weighted average is restricting; e.g., when neighborhood information is limited

Estimate inner-products among movie ratings
Regional level: Latent factor models

Latent factor models

Estimate unknown ratings as inner-products of factors:

A rank-3 SVD approximation

Properties:
- SVD isn't defined when entries are unknown → use specialized methods
- Very powerful model → can easily overfit, sensitive to regularization
- Probably most popular model among contestants
  - 12/11/2006: Simon Funk describes an SVD based method
  - 12/29/2006: Free implementation at timelydevelopment.com
Factorization on the RMSE scale

Global average: 1.1296
User average: 1.0651
Movie average: 1.0533
Cinematch: 0.9514
BellKor: 0.8693
Grand Prize: 0.8563
Inherent noise: ???

Our approach

• User factors:
  Model a user \( u \) as a vector \( p_u \sim N(\mu, \Sigma) \)
• Movie factors:
  Model a movie \( i \) as a vector \( q_i \sim N(\gamma, \Lambda) \)
• Ratings:
  Measure “agreement” between \( u \) and \( i \):
  \[ r_{ui} \sim N(p_u \cdot q_i, \epsilon^2) \]
• Maximize model’s likelihood:
  – Alternate between recomputing user-factors, movie-factors and model parameters
  – Special cases:
    • Alternating Ridge regression
    • Nonnegative matrix factorization

Localized factorization model

• Standard factorization:
  User \( u \) is a linear function parameterized by \( p_u \)
  \[ r_u = p_u \cdot q_i \]
• Allow user factors – \( p_u \) – to depend on the item being predicted
  \[ r_u = p_u(i) \cdot q_i \]
• Vector \( p_u(i) \) models behavior of \( u \) on items like \( i \)

Results on Netflix Probe set

Combining multi-scale views

Combination using confidence scores

Confidence Scores

• All algorithms presented are based on optimizing some objective
  – Can use the objective value as measure of confidence
• Use these confidence scores when combining results of different models

Item-item suggestions are very good when confidence is high
Class comments

- Interesting that combining different algorithms improves performance.
- Add item content information?
- Good paper; learnt a lot
- Alternative approach to SVD: Gaussian process regression models to model non-linearity?
- Small improvement after a lot of work.