IN5526 - Web Intelligence

Recommender systems

Juan D. Velasquez
Gaspar Pizarro V.

Departamento de Ingeniería Industrial
Universidad de Chile

November 22, 2016

Contents

Beer and diapers

Case large US supermarket

- Customer purchase behaviour:
- Product linked with another
  - Bread $\rightarrow$ butter,
  - Beer $\rightarrow$ diapers

Beer and diapers

Case large US supermarket

- Customer purchase behaviour:
- Product linked with another
  - Bread $\rightarrow$ butter,
  - Beer $\rightarrow$ diapers *wait, what?*
- Market segment
  - Young men married in the last three years with small children.

Based on this information, we deduce:

- Place diaper and beer on the same place on Friday afternoons.

Beer and diapers

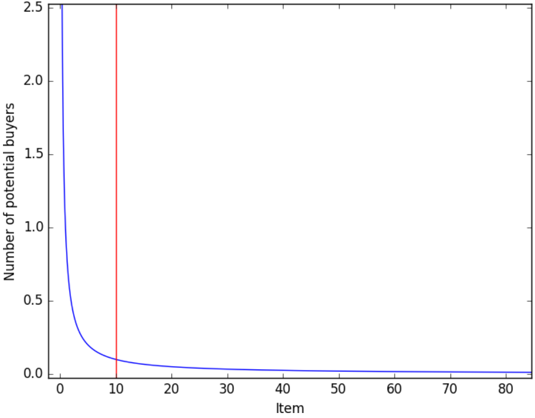
Wait, again?

- The new placement decision is made because it is profitable. Profit $>$ cost.
- But what if there were no cost to place beer with diapers or anything?

Recommender systems

- System that predicts user responses to options
- Options like:
  - Products in on-line retailers, like Amazon
  - Movies, like Netflix
  - News articles, like the New York Times *is it good?*

The tong tail



- In physical stores, decisions are taken on aggregates
- In online stores, decisions can be made for each user
  - Each user sees a different “store”

The utility matrix

Users vs items

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice	4			5	1		
Bob	5	5	4				
Carl				2	4	5	
Diego		3					3

Some things the matrix encodes

- *Alice prefers Twilight to Star Wars*
- *Carl has not watched any Harry Potter movie*

The problem to solve: *Will Alice like Star Wars 2?*

The utility matrix

Populating the matrix

How to populate the matrix?

- Explicit approach
  - *In a scale of 1 to 5, how much did you like this movie?*
  - But users usually do not want to give ratings
  - And the users who do bias the ratings
- Behavioral approach
  - Infer rating from user behavior
  - *User clicks link* $\Rightarrow$ *User likes item*
  - *User stays in link* $\Rightarrow$ *User loves item*

Contents

Content-based recommendations

Item profiles

- Recommend items similar to the ones user has liked before
- Item features can come from:
  - Text
  - Genres
  - Tags

Content-based recommendations

User profiles

- We want to get vector representations of users, the same way as for items
- Users can be caracterized with the features of the items they rate
- Users can be compared to other ones with these profiles
- Since we have item and user profiles, we can match a user with an item with those profiles
  - Example: Movie profile has Actor A, user profile has Actor A, better recommend it

Content-based recommendations

Classification algorithms

- We can predict the rating of a user with a regressor or classifier
- For each user we use some of their rating as training and test, and build a regressor or classifier
- Requires well-defined item features
- Requires (lots of) data
- Requires computing time (one classifier per user)

Content-based recommendations

Classification algorithms

- We can predict the rating of a user with a regressor or classifier
- For each user we use some of their rating as training and test, and build a regressor or classifier
- Requires well-defined item features
- Requires (lots of) data
- Requires computing time (one classifier per user)

Often the item representation is not available, so these approaches might not be possible

Contents

Collaborative filtering

We can use the user profiles to build item profiles and item profiles to build user profiles and recommend with them ...

Collaborative filtering

Recommendation model

This is one of many approaches to do collaborative filtering

- We model the rating of the user as a function of their *interests* in, let's say, genres, and the amount of genres a movie has
- $Interest_i = interestforgenre1 * amountofgenre1 + interestforgenre2 * amountofgenre2 + \cdots$

Collaborative filtering

Recommendation model

This is one of many approaches to do collaborative filtering

- We model the rating of the user as a function of their *interests* in, let's say, genres, and the amount of genres a movie has
- $Interest_i = interestforgenre1 * amountofgenre1 + interestforgenre2 * amountofgenre2 + \cdots$

$$y(i,j) = \theta^{(j)} \cdot x^{(i)}$$

- $r(i,j)$: 1 if User j has rated item i, 0 otherwise
- $y(i,j)$: Rating of item i by user j
- $\theta^{(j)}$: User profile for user j
- $x^{(i)}$: Item profile for item i

We assume the item profiles are not available from outside the system. We have to compute them

Collaborative filtering

Formal definition

- If we have item profiles $x^{(i)}$, we can estimate user profiles $\theta^{(j)}$:

$$\min_{\{\theta^{(j)}\}_{j=1}^{n_u}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}(\theta^{(j)}\cdot x^{(i)} - y(i,j))^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}||\theta^{(j)}||^2$$

- If we have user interests $\theta^{(j)}$ we can estimate item profiles $x^{(i)}$:

$$\min_{\{x^{(i)}\}_{j=1}^{n_I}} \frac{1}{2}\sum_{i=1}^{n_I}\sum_{j:r(i,j)=1}(\theta^{(j)}\cdot x^{(i)} - y(i,j))^2 + \frac{\lambda}{2}\sum_{i=1}^{n_I}||x^{(i)}||^2$$

n_u: Number of users
n_I: Number of items

Collaborative filtering

Formal definition

If we have nothing, we can solve both problems *at the same time*

$$\min_{\{\theta^{(j)}\}_{j=1}^{n_u},\{x^{(i)}\}_{j=1}^{n_I}} \frac{1}{2} \sum_{(i,j):r(i,j)=1} (\theta^{(j)} \cdot x^{(i)} - y(i,j))^2 + \frac{\lambda}{2} \sum_{i=1}^{n_I} ||x^{(i)}||^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} ||\theta^{(j)}||^2$$

Collaborative filtering

Gradient descent

1. Initialize $x^{(i)}$, $\theta^{(j)}$ to small random values
2. Apply gradient descent

$$
\begin{aligned}
x_k^{(i)} &\leftarrow x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} (\theta^{(j)} \cdot x^{(i)} - y(i,j))\theta_k^{(j)} + \lambda x_k^{(i)} \right) \\
\theta_k^{(j)} &\leftarrow \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} (\theta^{(j)} \cdot x^{(i)} - y(i,j))x_k^{(i)} + \lambda \theta_k^{(j)} \right)
\end{aligned}
$$

3. Predict rating $y(i,j)$ as

$$y(i,j) = \theta^{(j)} \cdot x^{(i)}$$

Collaborative filtering

Mean normalization

- But, what to do with new users?
- New user $\Rightarrow$ row full of blanks
- Collaborative filtering algorithm ends up with $\theta^{(n_u+1)} = \vec{0} \Rightarrow$ Zero interest for anything

Collaborative filtering

Mean normalization

We can normalize ratings with their mean user ratings

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice	4			5	1		
Bob	5	5	4				
Carl				2	4	5	
Diego		3					3
μ	4.5	4	4	3.5	2.5	5	3

Collaborative filtering

Mean normalization

Normalized utility matrix

	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice	-0.5			1.5	-1.5		
Bob	0.5	1	0				
Carl				-1.5	1.5	0	
Diego		-1					0

- Do gradient descent, estimate θ, x
- Predict rating $y(i,j)$ as

$$y(i,j) = \theta^{(j)} \cdot x^{(i)} + \mu^{(i)}$$

- New users still get $\theta^{(n_u+1)} = \vec{0}$, but predictions are the mean of the other users ratings
- New users are recommended most popular items
- This is the better the system can do for a new user

Evaluation

- This is almost a standard Machine Learning, so standard techniques apply
  - With all users, some ratings are taken out, splitting the dataset in training and test
  - K-fold cross-validation with the training test, for choosing the regularization parameter
- Recommenders are evaluated by the Root Mean Squared Error

$$RMSE = \sqrt{\frac{1}{\sum_{i,j} r(i,j)} \sum_{(i,j):r(i,j)=1} (\theta^{(j)} \cdot x^{(i)} - y(i,j))^2} \qquad (1)$$

The Netflix Challenge

Throw some money at the problem

- In 2006, Netflix released a 100-million-rating dataset, and offered $\$10^6$ to anyone who could make a better recommendation engine than their own CineMatch
- CineMatch's RMSE ≈ 0.95

The Netflix Challenge

Some insights

- CineMatch was not a good algorithm
  - For $y(m,u)$, an algorithm that averages the average rating of user u over all their movies and the average rating of m given by other users was only 3% worse than CineMatch
- An algorithm like the one taught before, with other tricks, got a 7% improvement over CineMatch
- Some tried to fuse the information of the names with IMDB, and failed. Why?
  - The algorithms were capable to find the movie features anyway
  - It was not that easy to match IMDB and Netflix movie names or genres

The Netflix Challenge

Some insights

- The best algorithms were actually a combination of different algorithms
- Time of rating was useful

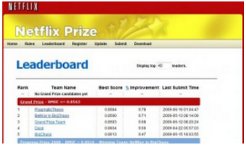
The Netflix Challenge

Sad ending

CASEY JOHNSTON, ARS TECHNICA BUSINESS 04.16.12 8:20 AM

NETFLIX NEVER USED ITS $1 MILLION ALGORITHM DUE TO ENGINEERING COSTS

Netflix awarded a $1 million prize to a developer team in 2009 for an algorithm that increased the accuracy of the company's recommendation engine by 10 percent. But it doesn't use the million-dollar code, and has no plans to



References

- Leskovec, Jure, Anand Rajaraman, and Jeffrey David Ullman. Mining of massive datasets. Cambridge University Press, 2014.
- Andrew Ng's Machine Learning course at coursera.org