Click here Slide(pdf))

Main dilemmas facing recommander systems

The long tail rule

lots of items have little popularity

Can't cater for niche preferences

Content-based Recommander systems

Main idea: evaluate main attributes of items to make recommandation

How to pick features?

We choose TF-IDF

(Term frequency * Inverse Doc Frequency)

define: f(i,j) means "frequency of item(feature)i in doc(item) j"

n(i) = number of docs that mention term i

N = total number of docs

first, compuye two variables:

TF(i,j) = f(i,j) / max(k f(k,j))

IDF(i) = log(N / n(i))

IF-IDF score, w(i,j) = TF(i,j) * IDF(i)

How to evalute similarity between user and item profiles?

use Cosine similarity.

upsides and downsides

upsides:

- not need other users' data
- can meet niche preferences

- able to recommand new items

downsides

- hard to find appropriate features
  In fact, we don't know how accurate the features we choose are.
- never command items out of the user profile

Collaborative Filtering

User-User collaborative filtering

Find other users that give similar ratings compared to the user

There we choose Pearson correlation.
Formula : See slides1 Page24.

Biased on other users' ratings

Item-Item collaborative filtering

find similar items to the unique item

Biased on unique user's ratings

Improvements in the Netflix game

The important idea: A three layer structure

Upper layer: Global

Consider overall deviations of users/movies

overall deviations of users

Add bias according to the mean of users/movies

Middle layer: Regional

Factorization: Addressing "regional" effects

choose SGD to do that

Lower layer: Local

CF: Extract local patterns
Made model modifications
See slides2Page10

Use weight to relace similarity

weight(i,j) -> minimize SSE on traing data
Use Convex optimization to do so