GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

RECOMMENDATION ENGINES

I. CONTENT-BASED FILTERING
II. COLLABORATIVE FILTERING

A recommendation system aims to match users to products/items/brand/etc that they likely haven’t experienced yet and/or predict a users preference based on past observations.

A recommendation system aims to match users to products/items/brand/etc that they likely haven't experienced yet and/or predict a users preference based on past observations.

A **ranking** or **prediction** is produced by analyzing other user/item ratings (and sometimes item characteristics) to provide personalized recommendations to users.

II. GENERAL DESIGN

There are two general approaches to the design:

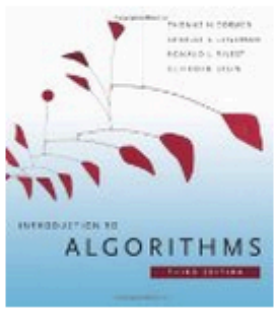
There are many approaches to the design, but these are commonly modeled techniques:

In **content-based filtering**, items are mapped into a feature space, and recommendations depend on *item characteristics*.

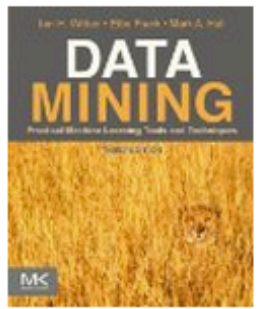
In contrast, an important assumption underlying all of **collaborative filtering**, is: *users who have similar preferences in the past are likely to have similar preferences in the future.*

Recommendations for You in Books

Cracking the Coding Interview: 150...
› Gayle Laakmann McDowell
Paperback
★★★★★ (166)
~~$39.95~~ $23.22
Why recommended?

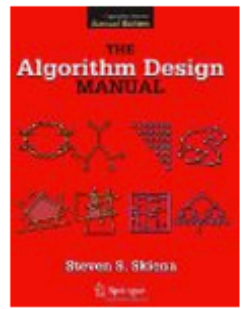


Introduction to Algorithms
Thomas H. Cormen, Charles E...
Hardcover
★★★★☆ (85)
~~$92.00~~ $80.00
Why recommended?



Data Mining: Practical Machine...
› Ian H. Witten, Eibe Frank, Mark A. Hall
Paperback
★★★★☆ (27)
~~$69.95~~ $42.09
Why recommended?

Elements of Programming Interviews...
› Amit Prakash, Adnan Aziz, Tsung-Hsien Lee
Paperback
★★★★½ (25)
~~$29.99~~ $26.18
Why recommended?



The Algorithm Design Manual
› Steve Skiena
Paperback
★★★★½ (47)
~~$89.95~~ $71.84
Why recommended?

Customers Who Bought This Item Also Bought

Pitch Dark (NYRB Classics)
› Renata Adler
Paperback
$11.54

How Literature Saved My Life
› David Shields
(60)
Hardcover
$18.08

Bleeding Edge
Thomas Pynchon
Hardcover
$18.05

The Flamethrowers: A Novel
› Rachel Kushner
(17)
Hardcover
$15.79

TV Shows

Your taste preferences created this row.

TV Shows.

As well as your interest in...

Because you watched 30 Rock

I. CONTENT-BASED FILTERING

Content-based filtering begins by mapping each item into a feature space. Both users and items are represented by vectors in this space.

Content-based filtering begins by mapping each item into a feature space. Both users and items are represented by vectors in this space.

Item vectors measure the degree to which the item is described by each feature, and ***user vectors*** measure a user's preferences for each feature.

Content-based filtering begins by mapping each item into a feature space. Both users and items are represented by vectors in this space.

Item vectors measure the degree to which the item is described by each feature, and ***user vectors*** measure a user's preferences for each feature.

Ratings are generated by taking **dot products** of user & item vectors.

features = (big box office, aimed at kids, famous actors)

Items (movies):
Finding Nemo = (5, 5, 2)
Mission Impossible = (3, -5, 5)
Jiro Dreams of Sushi = (-4, -5, -5)

features = (big box office, aimed at kids, famous actors)

Items (movies):
Finding Nemo = (5, 5, 2)
Mission Impossible = (3, -5, 5)
Jiro Dreams of Sushi = (-4, -5, -5)

Users:
Alice = (-3, 2, -2)
Bob = (4, -3, 5)

features = (big box office, aimed at kids, famous actors)

Items (movies):

Finding Nemo = (5, 5, 2)

Mission Impossible = (3, -5, 5)

Jiro Dreams of Sushi = (-4, -5, -5)

Prediction (for Alice)

5*-3 + 5*2 + 2*-2 = -9

User:

Alice = (-3, 2, -2)

features = (big box office, aimed at kids, famous actors)

Items (movies):

Finding Nemo = (5, 5, 2)

Mission Impossible = (3, -5, 5)

Jiro Dreams of Sushi = (-4, -5, -5)

Prediction (for Alice)

5*-3 + 5*2 + 2*-2 = -9

3*-3 + -5*2 + 5*-2 = -29

User:

Alice = (-3, 2, -2)

features = (big box office, aimed at kids, famous actors)

Items (movies):

Finding Nemo = (5, 5, 2)
Mission Impossible = (3, -5, 5)
Jiro Dreams of Sushi = (-4, -5, -5)

Prediction (for Alice)

5*-3 + 5*2 + 2*-2 = -9
3*-3 + -5*2 + 5*-2 = -29
-4*-3 + -5*2 + -5*-2 = +12

User:

Alice = (-3, 2, -2)

features = (big box office, aimed at kids, famous actors)

Items (movies):

Finding Nemo = (5, 5, 2)

Mission Impossible = (3, -5, 5)

Jiro Dreams of Sushi = (-4, -5, -5)

Prediction (for Alice)

5*-3 + 5*2 + 2*-2 = -9

3*-3 + -5*2 + 5*-2 = -29

-4*-3 + -5*2 + -5*-2 = +12

User:

Alice = (-3, 2, -2)

features = (big box office, aimed at kids, famous actors)

Items (movies):

Finding Nemo = (5, 5, 2)
Mission Impossible = (3, -5, 5)
Jiro Dreams of Sushi = (-4, -5, -5)

Prediction (for Bob)

User:

Bob = (4, -3, 5)

features = (big box office, aimed at kids, famous actors)

Items (movies):	**Prediction (for Bob)**	
Finding Nemo = (5, 5, 2)	5*4 + 5*-3 + 2*5	= +15
Mission Impossible = (3, -5, 5)	3*4 + -5*-3 + 5*5	= +52
Jiro Dreams of Sushi = (-4, -5, -5)	-4*4 + -5*-3 + -5*5	= -26

User:

Bob = (4, -3, 5)

features = (big box office, aimed at kids, famous actors)

Items (movies):

Finding Nemo = (5, 5, 2)

Mission Impossible = (3, -5, 5)

Jiro Dreams of Sushi = (-4, -5, -5)

Prediction (for Bob)

5*4 + 5*-3 + 2*5 = +15

3*4 + -5*-3 + 5*5 = +52

-4*4 + -5*-3 + -5*5 = -26

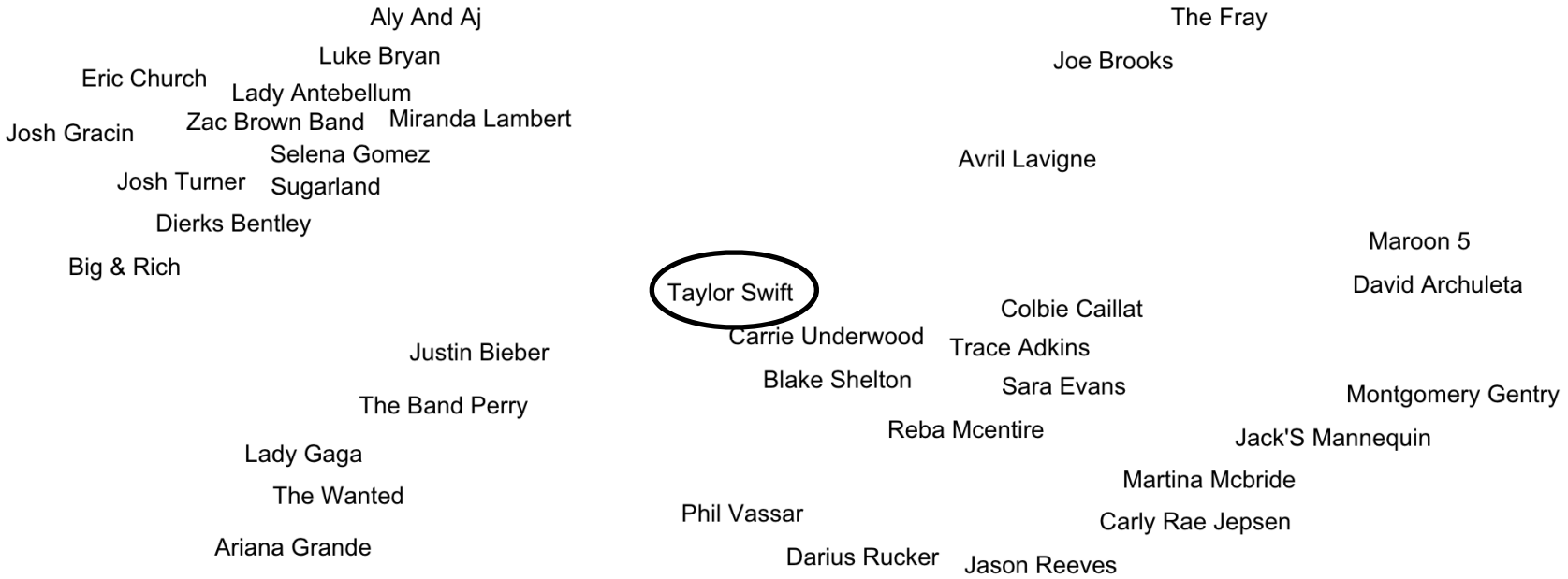
User:

Bob = (4, -3, 5)

One notable example of content-based filtering is Pandora, which maps songs into a feature space using features (or “genes”) designed by the Music Genome Project.

Using song vectors that depend on these features, Pandora can create a station with music having similar properties to a song the user selects.

VISUALIZATION OF SIMILAR ARTISTS



http://www.music-map.com/

Content-based filtering has some difficulties:

Content-based filtering has some difficulties:

- Must map items into a feature space (usually by hand!)
- Recommendations are limited in scope (items must be similar to each other)
- Hard to create cross-content recommendations (eg books/ music films…this would require comparing elements from different feature spaces!)

II. COLLABORATIVE FILTERING

Collaborative filtering refers to a family of methods for predicting ratings where instead of thinking about users and items in terms of a feature space, we are *only* interested in the existing user-item ratings themselves.

Main difference between content and collaborative filtering:

Content Based:

maps items and users into a feature space

Collaborative:

relies on previous user-item ratings

User-based nearest-neighbor collaborative filtering (2)

- **Example**
  - A database of ratings of the current user, Alice, and some other users is given:

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

  - Determine whether Alice will like or dislike *Item5*, which Alice has not yet rated or seen

User-based nearest-neighbor collaborative filtering (3)

- **Some first questions**
  - How do we measure similarity?
  - How many neighbors should we consider?
  - How do we generate a prediction from the neighbors' ratings?

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Measuring user similarity (1)

- **A popular similarity measure in user-based CF: Pearson correlation**
  - a, b : users
  - $r_{a,p}$: rating of user a for item p
  - P : set of items, rated both by a and b
  - Possible similarity values between -1 and 1

$$sim(a,b) = \frac{\sum_{p \in P}(r_{a,p} - \bar{r}_a)(r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P}(r_{a,p} - \bar{r}_a)^2}\sqrt{\sum_{p \in P}(r_{b,p} - \bar{r}_b)^2}}$$

Measuring user similarity (2)

- **A popular similarity measure in user-based CF: Pearson correlation**
  - a, b : users
  - $r_{a,p}$: rating of user a for item p
  - P : set of items, rated both by a and b
  - Possible similarity values between -1 and 1

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

sim = 0,85

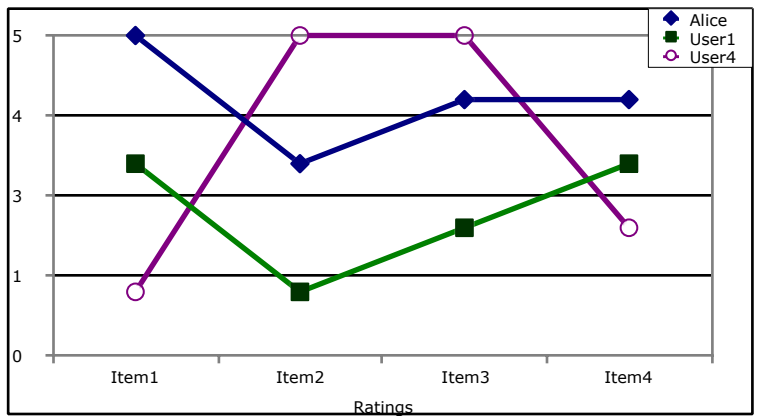
sim = 0,00

sim = 0,70

sim = -0,79

Pearson correlation

- **Takes differences in rating behavior into account**



- **Works well in usual domains, compared with alternative measures**
  - such as cosine similarity

Making predictions

- A common prediction function:

$$\boldsymbol{pred(a,p)} = \overline{\boldsymbol{r_a}} + \frac{\sum_{\boldsymbol{b} \in \boldsymbol{N}} \boldsymbol{sim(a,b)} * (\boldsymbol{r_{b,p}} - \overline{\boldsymbol{r_b}})}{\sum_{\boldsymbol{b} \in \boldsymbol{N}} \boldsymbol{sim(a,b)}}$$

- Calculate, whether the neighbors' ratings for the unseen item i are higher or lower than their average
- Combine the rating differences – use the similarity with a as a weight
- Add/subtract the neighbors' bias from the active user's average and use this as a prediction

Customers Who Bought This Item Also Bought

Pitch Dark (NYRB Classics)
› Renata Adler
Paperback
$11.54

How Literature Saved My Life
› David Shields
(60)
Hardcover
$18.08

Bleeding Edge
Thomas Pynchon
Hardcover
$18.05

The Flamethrowers: A Novel
› Rachel Kushner
(17)
Hardcover
$15.79

The cold start problem arises because we've been relying only on ratings data, or on explicit feedback from users.

Until users rate several items, we don't know anything about their preferences!

The cold start problem arises because we've been relying only on ratings data, or on explicit feedback from users.

Until users rate several items, we don't know anything about their preferences!

We can get around this by enhancing our recommendations using implicit feedback, which may include things like item browsing behavior, search patterns, purchase history, etc.

While explicit feedback (ratings, likes, purchases) leads to high quality ratings, the data is sparse and cold starts are problematic.

While explicit feedback (ratings, likes, purchases) leads to high quality ratings, the data is sparse and cold starts are problematic.

Meanwhile implicit feedback (browsing behavior, etc.) leads to less accurate ratings, but the data is much more dense (and less invasive to collect).