GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

RECOMMENDATION ENGINES

I. CONTENT-BASED FILTERING
II. COLLABORATIVE FILTERING

EXERCISE:
III. RECOMMENDING WITH PYTHON

IV. THE NETFLIX PRIZE

A recommendation system aims to match users to products/items/brand/etc that they likely haven’t experienced yet.

A recommendation system aims to match users to products/items/brand/etc that they likely haven’t experienced yet.

This rating is produced by analyzing other user/item ratings (and sometimes item characteristics) to provide personalized recommendations to users.

There are two general approaches to the design:

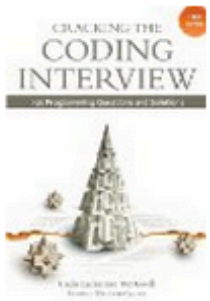
There are two general approaches to the design:

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

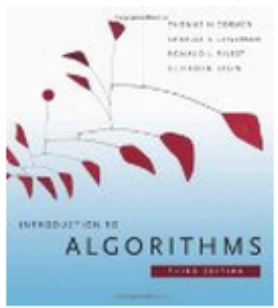
In contrast, the only data under consideration in **collaborative filtering** are user-item ratings, and recommendations depend on *user preferences*.

EXAMPLES – AMAZON CONTENT-BASED

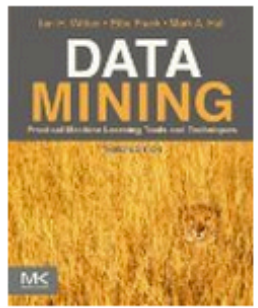
Recommendations for You in Books



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

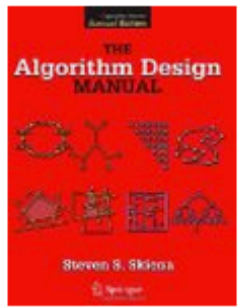


Introduction to Algorithms
Thomas H. Cormen, Charles E...
Hardcover
★★★★☆ (85)
~~$92.00~~ $80.00
Why recommended?



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?

EXAMPLES – AMAZON COLLABORATIVE FILTERING

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

EXAMPLES – NETFLIX

TV Shows

Your taste preferences created this row.

TV Shows.

As well as your interest in...

EXAMPLES – NETFLIX

Because you watched 30 Rock

Recommended for you because you watched
Sugar Minott - Oh Mr Dc (Studio One)

Mikey Dread - Roots and Culture

by klaxonklaxon · 1,164,133 views

Lyrics:
Now here comes a special request
To each and everyone

Recommended for you because you watched
Thelonious Monk Quartet - Monk In Denmark

Bill Evans Portrait in Jazz (Full Album)

by hansgy1 · 854,086 views

Bill Evans Portrait in Jazz 1960
1. Come Rain or Come Shine - 3.19 (0:00)
2. Autumn Leaves - 5.23 (3:24)

Recommended for you because you watched
Bob Marley One Drop

Bob Marley - She's gone

by Dionysios29 · 1,058,704 views

This is one of the eleven songs of album Kaya that Bob Marley and The Wailers creative in 1978.
Lyrics:

How can we find good recommendations?

- Manual Curation

- Manually Tag Attributes

content-based filtering

- Audio Content, Metadata, Text Analysis

- Collaborative Filtering

MOST E-MAILED | RECOMMENDED FOR YOU

1. **How Big Data Is Playing Recruiter for Specialized Workers**
2. SLIPSTREAM
   When Your Data Wanders to Places You've Never Been
3. MOTHERLODE
   The Play Date Gun Debate
4. **For Indonesian Atheists, a Community of Support Amid Constant Fear**
5. **Justice Breyer Has Shoulder Surgery**
6. BILL KELLER
   Erasing History

8. How do you determine my Most Read Topics? Back to top ▲

Each NYTimes.com article is assigned topic tags that reflect the content of the article. As you read articles, we use these tags to determine your most-read topics.

To search for additional articles on one of your most-read topics, click that topic on your personalized Recommendations page. To learn more about topic tags, visit Times Topics.

NOTE

Collaborative or Content based?

8. How do you determine my Most Read Topics? Back to top ▲

Each NYTimes.com article is assigned topic tags that reflect the content of the article. As you read articles, we use these tags to determine your most-read topics.

To search for additional articles on one of your most-read topics, click that topic on your personalized Recommendations page. To learn more about topic tags, visit Times Topics.

NOTE

Collaborative or Content based?

CONTENT BASED ☺

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):	**Prediction (for Alice)**	
Finding Nemo = (5, 5, 2)	5*-3 + 5*2 + 2*-2	= -9
Mission Impossible = (3, -5, 5)	3*-3 + -5*2 + 5*-2	= -29
Jiro Dreams of Sushi = (-4, -5, -5)	-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):	**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

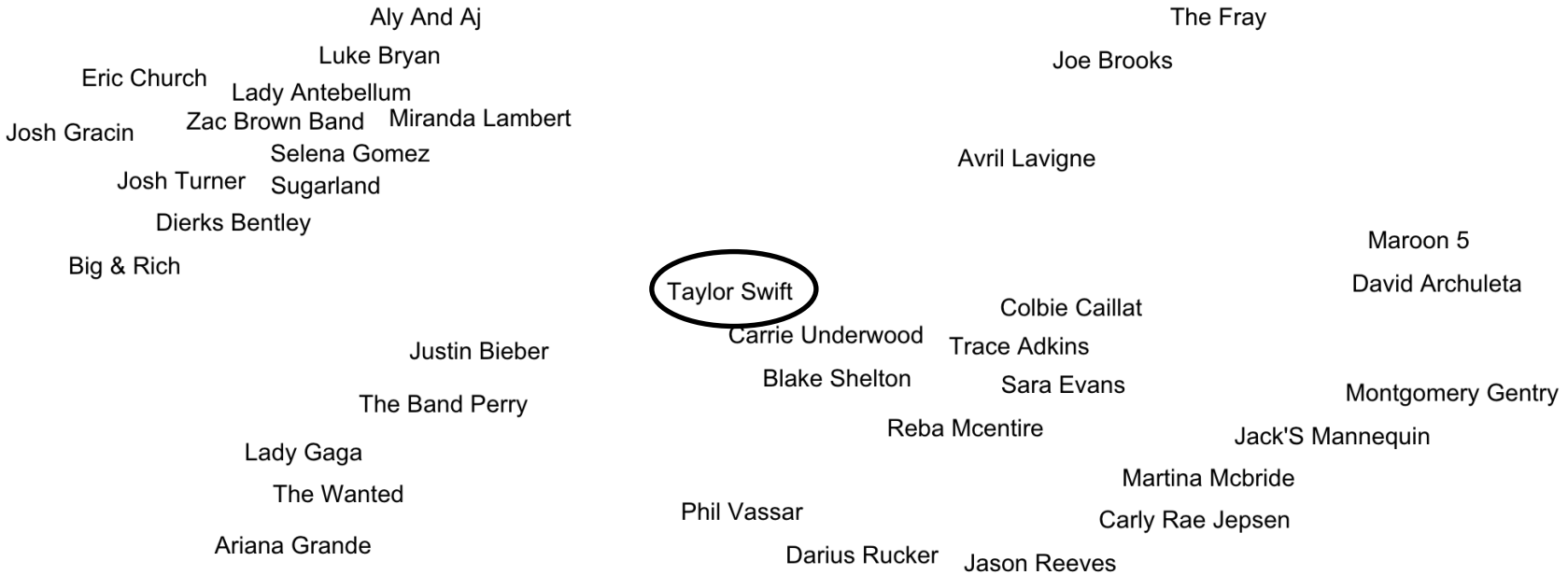
User:

Bob = (4, -3, 5)

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.

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.

In this case, our dataset is a *ratings matrix* whose columns correspond to items, and whose rows correspond to users.

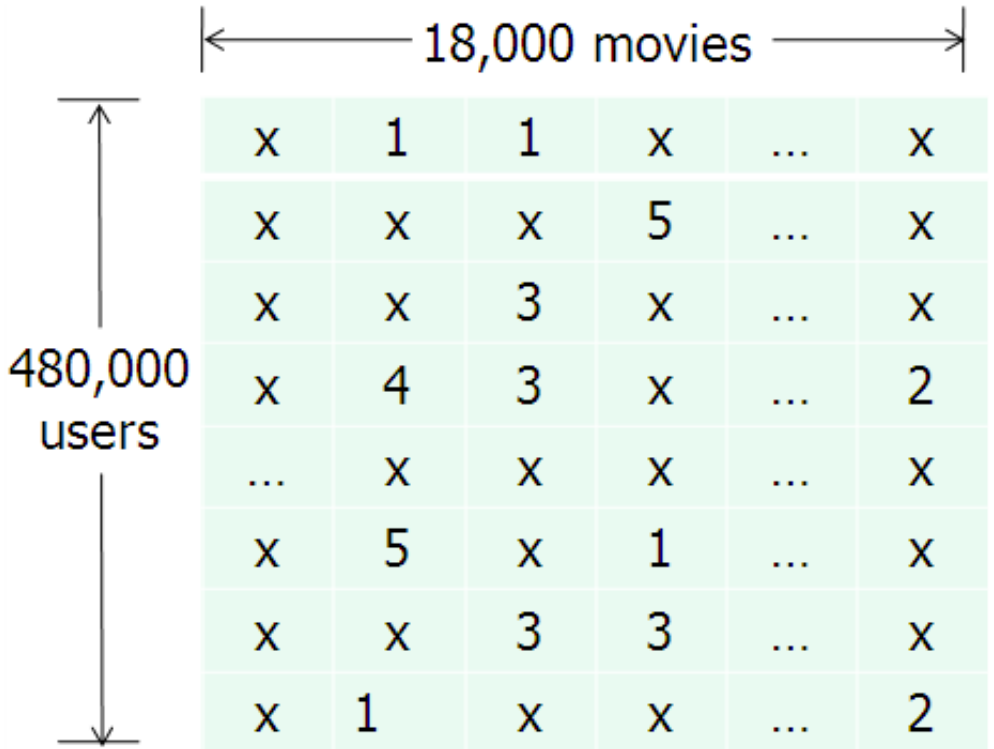
Collaborative filtering refers to a family of methods for predicting ratings where instead of thinking about users and items in terms of a feature space, we are *only* interested in the existing user-item ratings themselves.

In this case, our dataset is a *ratings matrix* whose columns correspond to items, and whose rows correspond to users.

NOTE

The idea here is that users get value from recommendations based on other users with similar *tastes*.

RATINGS MATRIX



NOTE

This matrix will always be *sparse*!

source: http://www.eecs.berkeley.edu/~zhanghao/main/publications/subfolder/netflix.png

Main difference between content and collaborative filtering:

Content Based:

maps items and users into a feature space

Collaborative:

relies on previous user-item ratings

We will look at collaborative filtering in a user-user sense.

We will look at collaborative filtering in a user-user sense.

We will take a given user, and find the K most similar users, and then recommend brands from the similar users!

We will look at collaborative filtering in a user-user sense.

We will take a given user, and find the K most similar users, and then recommend brands from the similar users!

NOTE

Sound familiar? It's similar to KNN!

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 system cannot draw inferences because it hasn’t gathered enough information yet.

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

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).

III. PYTHON EXAMPLE

Our data:

- CSV of two columns, user ID and Brand
- Each row represents a user liking a brand

Example:

User ID	Brand
86509	H&M
86509	Target
86509	Old Navy
86510	H&M
86510	Lowe's
86510	Home Depot
86511	Banana Republic
86511	Kohl's
86511	Old Navy

How do we define “similarity” of users?

How do we define “similarity” of users?

This is required if we want to do user-based collaborative filtering

MATH ALERT!!

How do we define “similarity” of users?

Jaccard Similarity:

Defines similarity between two sets of objects

How do we define “similarity” of users?

Jaccard Similarity:

Defines similarity between two sets of objects

$$JS(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|}$$

How do we define “similarity” of users?

Jaccard Similarity:

Defines similarity between two sets of objects

$$JS(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|}$$

Number of similar elements

Number of distinct elements

$$JS(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|}$$

JS ({1, 2, 3}, {2,3,4}) = $\frac{\{2, 3\}}{\{1, 2, 3, 4\}} = \frac{2}{4}$

$$JS(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|}$$

Exercise:

User one: {“Target”, “Banana Republic”, “Old Navy”}
User two: {“Banana Republic”, “Gap”, “Kohl’s”}

JS (User one, User two) =

$$JS(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|}$$

Exercise:

User one: {“Target”, “Banana Republic”, “Old Navy”}
User two: {“Banana Republic”, “Gap”, “Kohl’s”}

JS (User one, User two) = 1 / 5 **= .2**

PYTHON ALGORITHM STEPS

1. Get list of known users in a dictionary where the key is the user ID, an the value is a list of brands they like

   Example: { ‘83065’ : ["Kohl's", 'Target’] }

2. For a given user, we will calculate their closeness to every user in csv
3. We will choose the K most similar users
4. Recommend brands liked by similar users

PYTHON ALGORITHM STEPS

1. Get list of known users in a dictionary where the key is the user ID, an the value is a list of brands they like

   Example: { ‘83065’ : ["Kohl's", 'Target’] }

2. For a given user, we will calculate their closeness to every user in csv
3. We will choose the K most similar users
4. Recommend brands liked by similar users

Consider this a kind of KNN but instead of Euclidean Distance, we are using the Jaccard Similarity

IV. THE NETFLIX PRIZE

The Netflix prize was a competition to see if anyone could make a 10% improvement to Netflix's recommendation system (accuracy measured by RMSE).

The Netflix prize was a competition to see if anyone could make a 10% improvement to Netflix's recommendation system (accuracy measured by RMSE).

The grand prize was $1m dollars

The Netflix prize was a competition to see if anyone could make a 10% improvement to Netflix's recommendation system (accuracy measured by RMSE).

The grand prize was $1m dollars

The ratings matrix contained >100mm numerical entries (1-5 stars) from ~500k users across ~17k movies. The data was split into train/quiz/test sets to prevent overfitting on the test data by answer submission (this was a clever idea!)

The competition began in 2006, and the grand prize was eventually awarded in 2009. The winning entry was a stacked ensemble of 100's of models (including neighborhood & matrix factorization models) that were blended using boosted decision trees.

Ultimately, the competition ended in a photo finish. The winning strategy came down to last-minute team mergers & creative blending schemes to shave 3rd & 4th decimals off RMSE (concerns that would not be important in practice).

The competition began in 2006, and the grand prize was eventually awarded in 2009. The winning entry was a stacked ensemble of 100's of models (including neighborhood & matrix factorization models) that were blended using boosted decision trees.

Ultimately, the competition ended in a photo finish. The winning strategy came down to last-minute team mergers & creative blending schemes to shave 3rd & 4th decimals off RMSE (concerns that would not be important in practice).

Though they adopted some of the modeling techniques that emerged from the competition, Netflix never actually implemented the prizewinning solution.

Why do you think that's true?