Collaborative Filtering

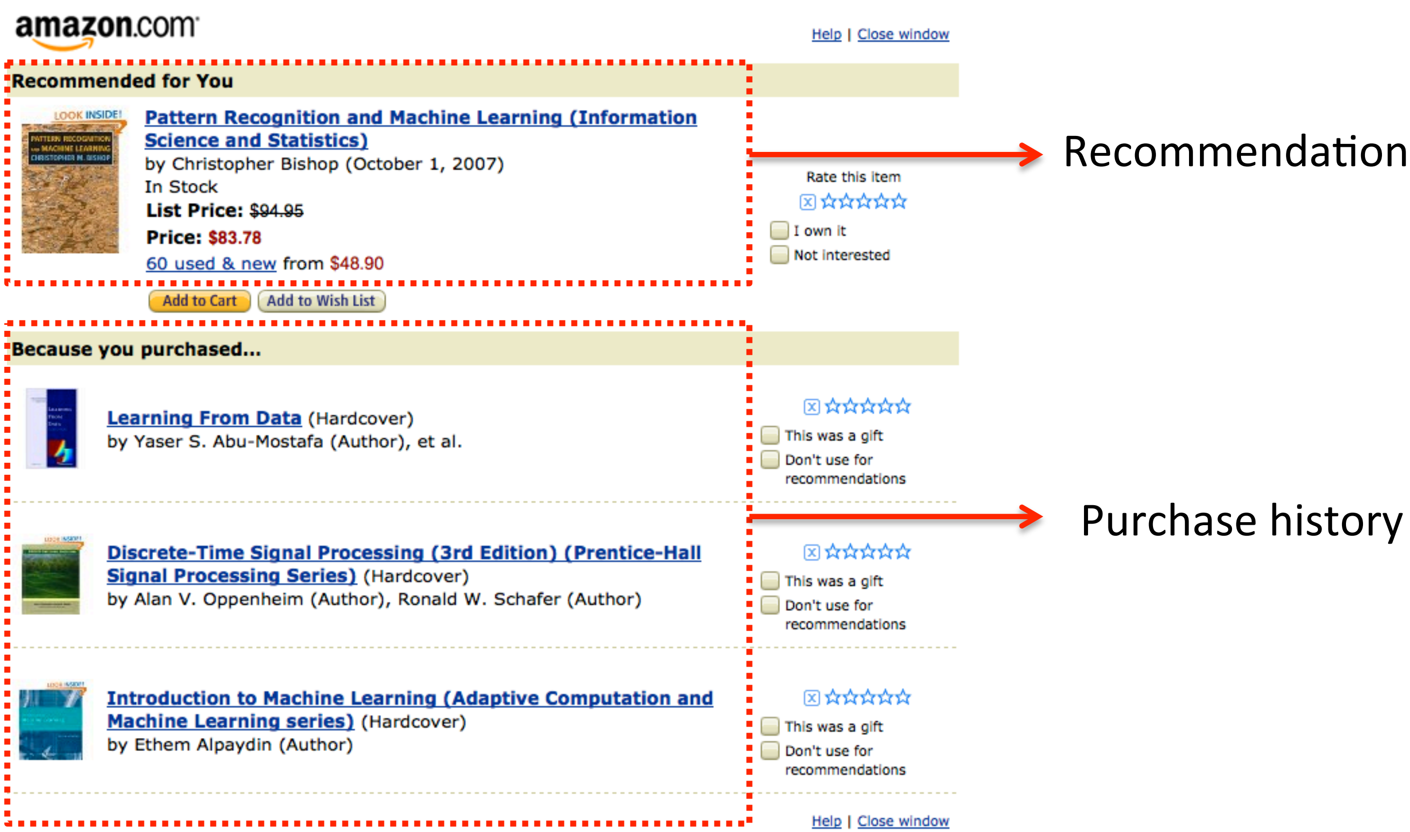
EECS 349 Machine Learning

Bongjun Kim

Fall, 2015

What is Collaborative Filtering?

- Recommendation system
  - Amazon recommends items based on your purchase history and ratings

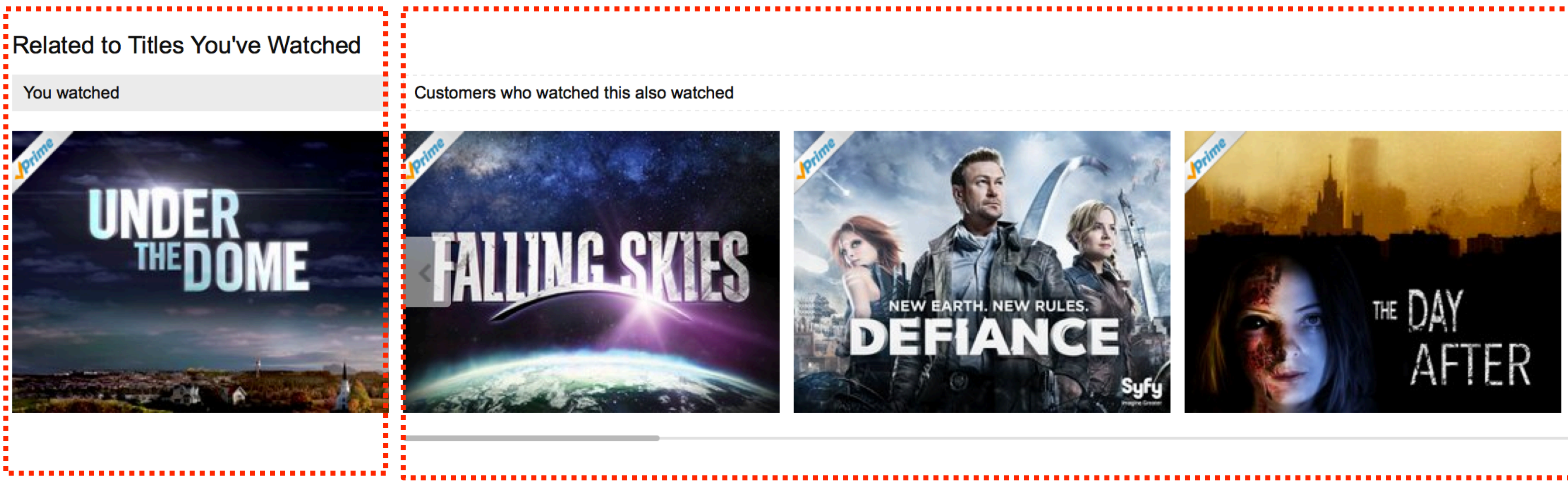


What is Collaborative Filtering?

- Recommendation system
  - Amazon recommends items based on your purchase history and ratings

View history

Recommendation

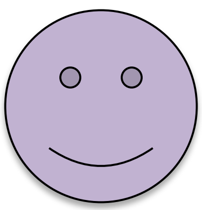


What is Collaborative Filtering?

- Task: How do I predict what you'll like?
- Two approaches
  - User-based: You will like *item A* because **<u>users</u>** who are similar to you like *item A*.
  - Item-based: You will like *item A* because you like **<u>items</u>** that are similar to *item A*.

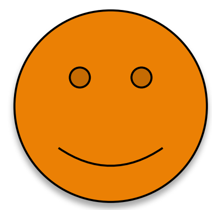
User-Based Collaborative Filtering

- Find users that is similar to you and you might like the item the user likes



A

I like..
- Star wars
- Star Trek
- Mission Impossible



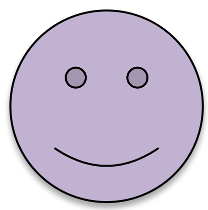
B

I like..
- Star wars
- Star Trek
- Mission Impossible
- X-men

B *is a user who has similar preference to* ***A****.*
So ***A*** *would like "X-men" too !!*

Item-Based Collaborative Filtering

- You might like items that are similar to items you already like



A

I like Star wars !

*"Star Trek" is a movie similar to Star Wars because it has "star" in the name. Then, **A** would like "Star Trek" too!*

*Do you think **A** would also like "Dancing with the Star"?*

Feature Selection

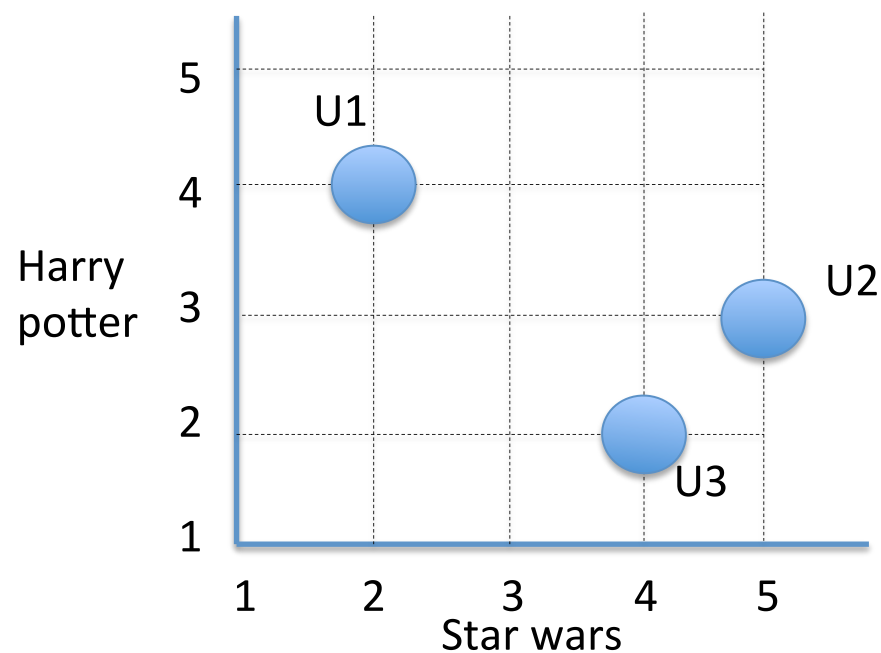
- Measuring similarity (of users or items) requires measuring their features.
- Which features should I measure?
- Are there features that are (relatively) insensitive to the particulars of the recommendation tasks?
- User ratings to items or their purchase history is one of the explicit features to measure user preference

USER-BASED COLLABORATIVE FILTERING

How do we find a user who is similar?

- Distance (or similarity) measure
  - N-dimensional space
- Example: movie ratings of 3 users
  - Ratings from 1 (dislike) to 5 (like)

	U1	U2	U3
Harry Potter	4	3	2
Star Wars	2	5	4



Which similarity measure to use?

- p-norm
  - Manhattan
  - Euclidian
- Pearson Correlation
- Cosine Similarity
- Etc..

Who is the most similar to John?

Example #1

	Inception	Begin again	Once
Brian	5	2	2
Bob	1	4	4
Cathy	2	3	3
John	5	1	2

- Manhattan Distance:

(John, Brian) = 0 + 1 + 0 =1
(John, Bob) = 4 + 3+ 2 =9
(John, Cathy) = 3 + 2 + 1 = 6

Q: Does Manhattan Distance measure similarities properly in this data set?

Who is the most similar to Adam?

Example #2

	Inception	Begin again	Once	Star wars
Bill	2	3	3	2
Brian	5	1	1	5
Adam	3	2	2	3

- Manhattan Distance:

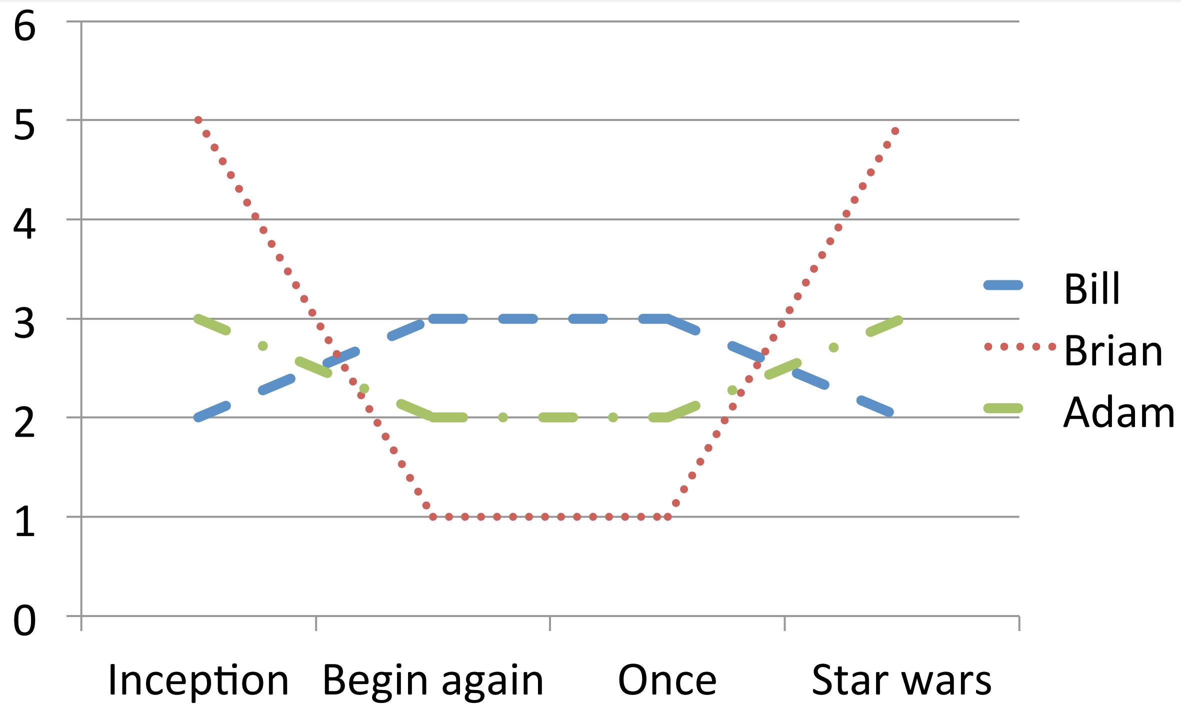
(Adam, Bill) = 1 + 1 +1 + 1 =4

(Adam, Brian) = 2 + 1+ 1 + 2 = 6

Q: Does Manhattan Distance measure similarities properly in this data set?

Different users may use different rating scales

Who is the most similar to Adam?



- Manhattan Distance:

(Adam, Bill) = 1 + 1 +1 + 1 =4

(Adam, Brian) = 2 + 1+ 1 + 2 = 6

Q: Does Manhattan Distance measure similarities properly in this data set?

Different users may use different rating scales

Pearson Correlation

- Measure of correlation between two variables
- Pearson correlation coefficient
  - Range (-1, 1)
  - A perfect positive correlation: 1
  - A perfect negative correlation: -1

$$sim(\mathbf{u}, \mathbf{v}) = \frac{\sum_{i \in C}(r_{\mathbf{u},i} - \bar{r}_{\mathbf{u}})(r_{\mathbf{v},i} - \bar{r}_{\mathbf{v}})}{\sqrt{\sum_{i \in C}(r_{\mathbf{u},i} - \bar{r}_{\mathbf{u}})^2}\sqrt{\sum_{i \in C}(r_{\mathbf{v},i} - \bar{r}_{\mathbf{v}})^2}},$$

In Python,

```
>> import scipy.stats
>> scipy.stats.pearsonr(array1, array2)
```

Cosine Similarity

- Measure of similarity between two vectors
  - Range from -1 (opposite) to 1 (same)

- Cosine similarity between vector *a* and *b:*

$$sim(a,b) = \frac{a \cdot b}{|a| * |b|}$$

Who is the most similar to Adam?

Example #2

	Inception	Begin again	Once	Star wars
Bill	2	3	3	2
Brian	5	1	1	5
Adam	3	2	2	3

- Pearson Correlation:

(Adam, Bill) = -1
(Adam, Brian) = 1

Q: Does Pearson Correlation measure similarities properly in this data set?

How to predict ratings to unrated items

- User-based K- Nearest Neighbor Collaborative Filtering
  1) Define a similarity measure
  2) Pick k users that had similar preferences to those of current user
  3) Compute a prediction from a weighted average of k nearest neighbors' ratings *(see the next slide)*

You need to do experiments to find optimal k value.

How to predict ratings to unrated items

- Prediction for the rating of user a for item p.

Rating of user *b* for item *p*

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

User *a*'s average rating

Similarity between user *a* and user *b*

Let's practice user-based k-NN CF

- In this practice and our homework, we will use much simpler way to compute a prediction of rating
  1) Define a similarity measure
  2) Pick k users that had similar preferences to those of current user
  3) **Pick the mode of the top k nearest neighbors as the predicted rating**

     - ex) If you pick 3 neighbors and their ratings to the target item are (2, 2, 3), then the prediction will be 2.

Practice: User-based k-NN CF (k=1)

Example #1: How would John rate Star wars?

	Inception	Begin again	Once	Star wars
Brian	5	2	2	4
Bob	1	4	4	2
Cathy	2	3	3	1
John	5	1	2	?

Manhattan Distance:

(John, Brian) = 0 + 1 + 0 =1

(John, Bob) = 4 + 3+ 2 =9

(John, Cathy) = 3 + 2 + 1 = 6

The nearest neighbor: Brian

John's rating to Star wars: 4

Practice: User-based k-NN CF (k=1)

Example #2: How would John rate Avatar?

	Inception	Begin again	Once	Star wars	Avatar
Brian	2	3	3	1	4
Bob	5	1	1	5	2
Cathy	5	1	2	4	1
John	3	2	2	3	?

Manhattan Distance:
(John, Brian) = 1 + 1 +1 + 2 =5
(John, Bob) = 2 + 1+ 1 + 2 = 6
(John, Cathy) = 1 + 1 + 1 + 1= 4

The nearest neighbor: Cathy
John's rating to Avatar: 1

Pearson Correlation Coefficient
(John, Brian) = -0.90
(John, Bob) = 1.0
(John, Cathy) = 0.95

The nearest neighbor: Bob
John's rating to Avatar: 2

ITEM-BASED COLLABORATIVE FILTERING

How to predict ratings to unrated items

- **Item-based** K- Nearest Neighbor Collaborative Filtering

  1) Define a similarity measure between **items**

  2) Pick k items rated by the current user similar to the target item

  3) Compute a prediction from a weighted average of the k similar items' ratings

Let's practice **item-based** k-NN CF

- In this practice and our homework, we will use much simpler way to compute a prediction of rating
  1) Define a similarity measure between **items**
  2) Pick k items rated by the current user similar to the target item
  3) **Pick the mode of the top k nearest neighbors as the predicted rating**

  - ex) If you picked 3 items and current user's ratings to the 3 items are (2, 2, 3), then the prediction will be 2.

Practice: Item-based k-NN CF (k=1)

Example #1

	Inception	Begin again	Once	Star wars
Brian	5	2	2	4
Bob	1	4	4	2
Cathy	2	3	3	1
John	5	1	2	?

Manhattan Distance:

(Star wars, Inception) = 1 + 1 + 1 =3
(Star wars, Begin again) = 1 + 2+ 2 =5
(Star wars, Once) = 2+2+2 = 6

The most similar item to Star wars: Inception
John's rating to Star wars: 5

The Cold Start Problem

- What if this user has never rated anything before?
- What if nobody has rated this item before?
- Additional information. For example,
  – Ask users to rate some initial items
  – Demographic information for users
  – Content analysis or metadata for items

Missing values

- Missing values in user-rating matrix
  - What if two users have rated different sets of things? How do we compare them?
  - What if two items have been rated by disjoint sets of users? How do we compare them?

Dealing with missing values

Example

	Inception	Begin again	Once	Star wars	Avatar
Brian	2	?	3	?	4
Bob	5	1	1	5	2
Cathy	5	?	2	2	1
John	5	?	2	3	?

Dealing with missing values

Example

	Inception	Begin again	Once	Star wars	Avatar
Brian	2	0	3	0	4
Bob	5	1	1	5	2
Cathy	5	0	2	2	1
John	5	0	2	3	?

Dealing with missing values

- Discarding the person/item from comparison?
  - It does not solve cold start problem
  - What if the data set is so sparse?
- Putting in a crazy number (-1000) for missing values?
- Putting in a random number?
- Putting in a mean (median) value?
  - Mean value of what set?
- Other advanced imputation technique?

Make a decision

- Which similarity (or distance) measure to use?
- How many neighbors to pick?
- How to weight neighbors chosen?
- User-based or item-based?
- How to deal with missing values?