Recommender Systems

Data Science Dojo

datasciencedojo
unleash the data scientist in you

Overview

- What are Recommender Systems?
- How do they work?
  - Collaborative Recommendation
  - Content-Based Recommendation
- How do we evaluate them?
- Example using Azure ML

datasciencedojo
unleash the data scientist in you

Overview

- **What are Recommender Systems?**
- How do they work?
  - Collaborative Recommendation
  - Content-Based Recommendation
- How do we evaluate them?
- Example using Azure ML

Recommender Systems

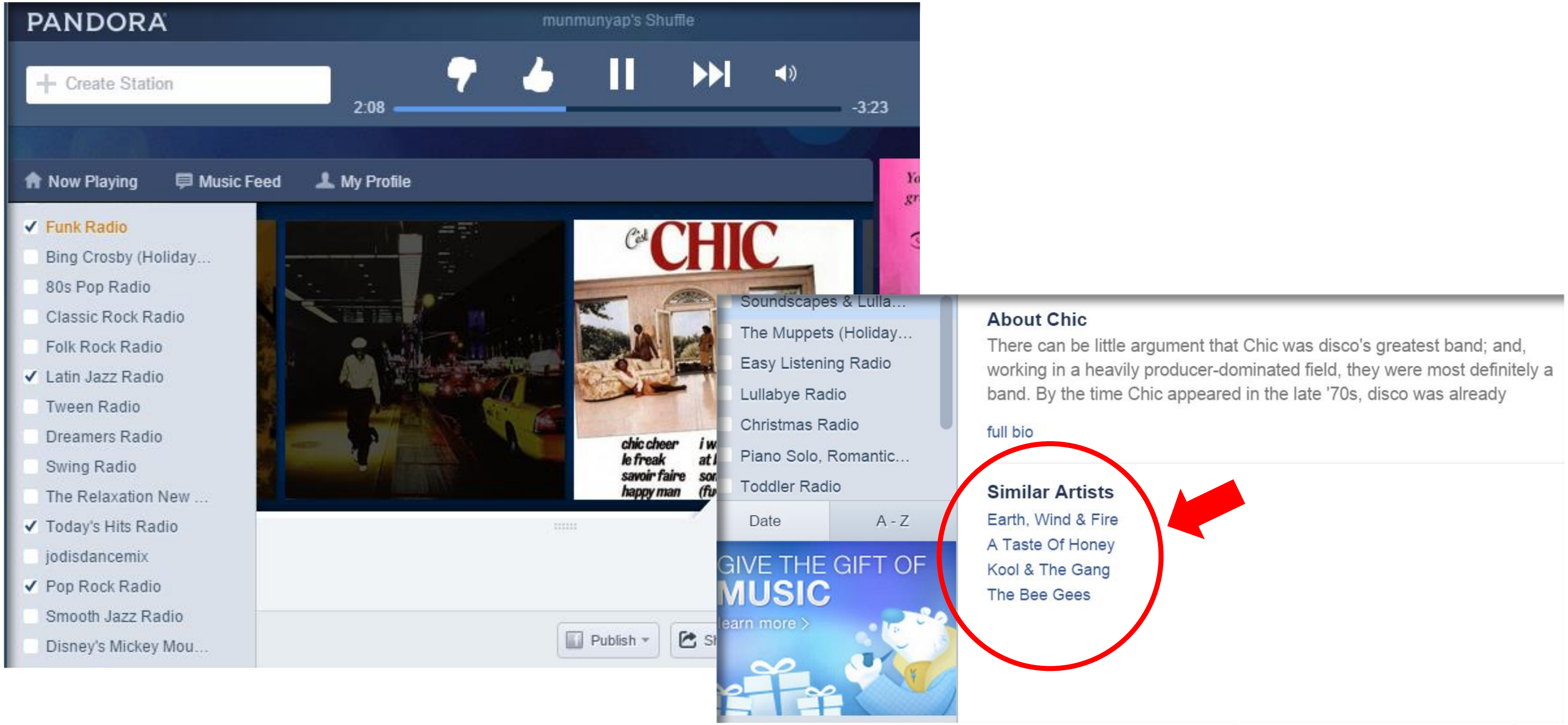
- What are Recommender Systems?
  - To solve information overload problem
  - Automated systems to filter and recommend products based on users' interest and taste.

Example: Retail

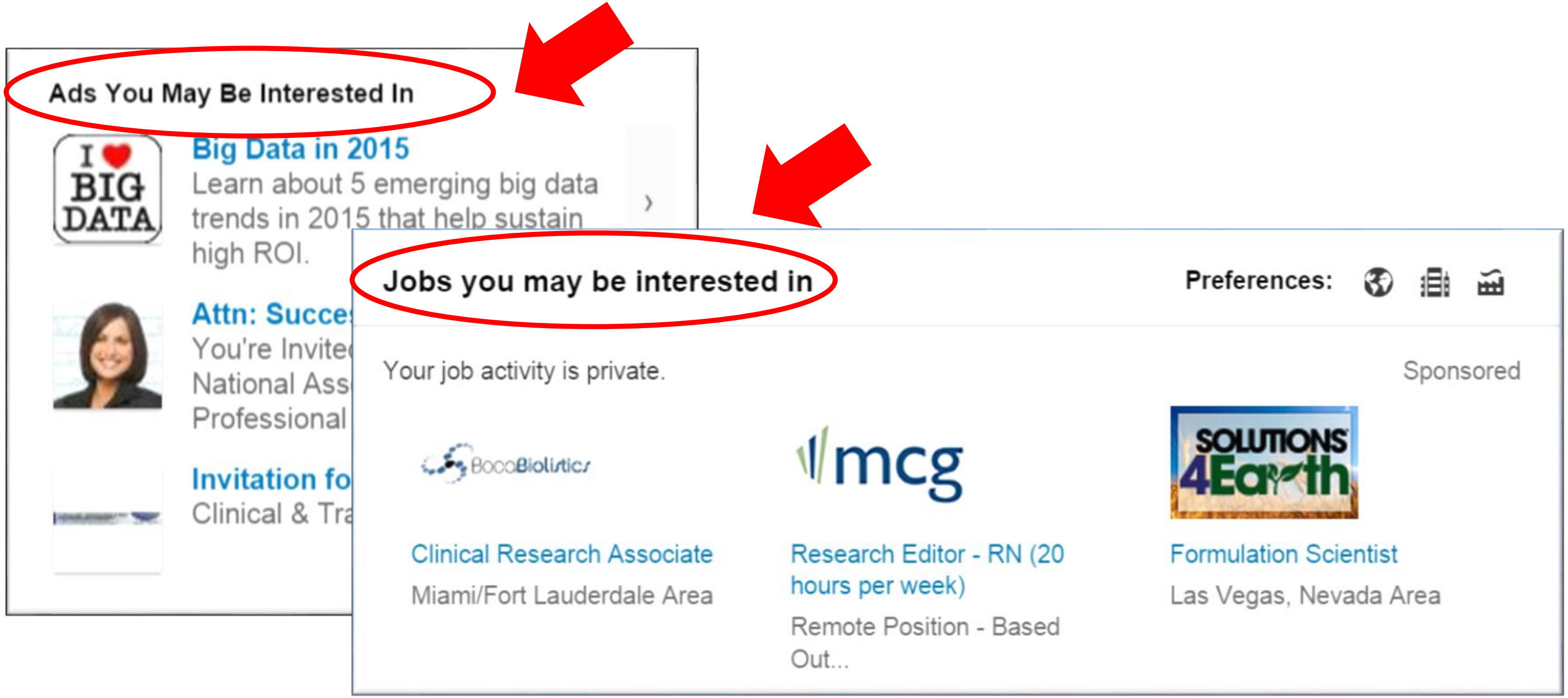
Related to Items You've Viewed See more

datasciencedojo
unleash the data scientist in you

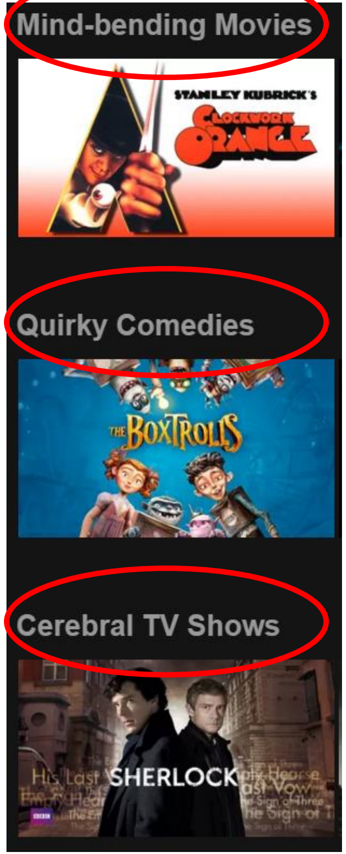
Example: Entertainment

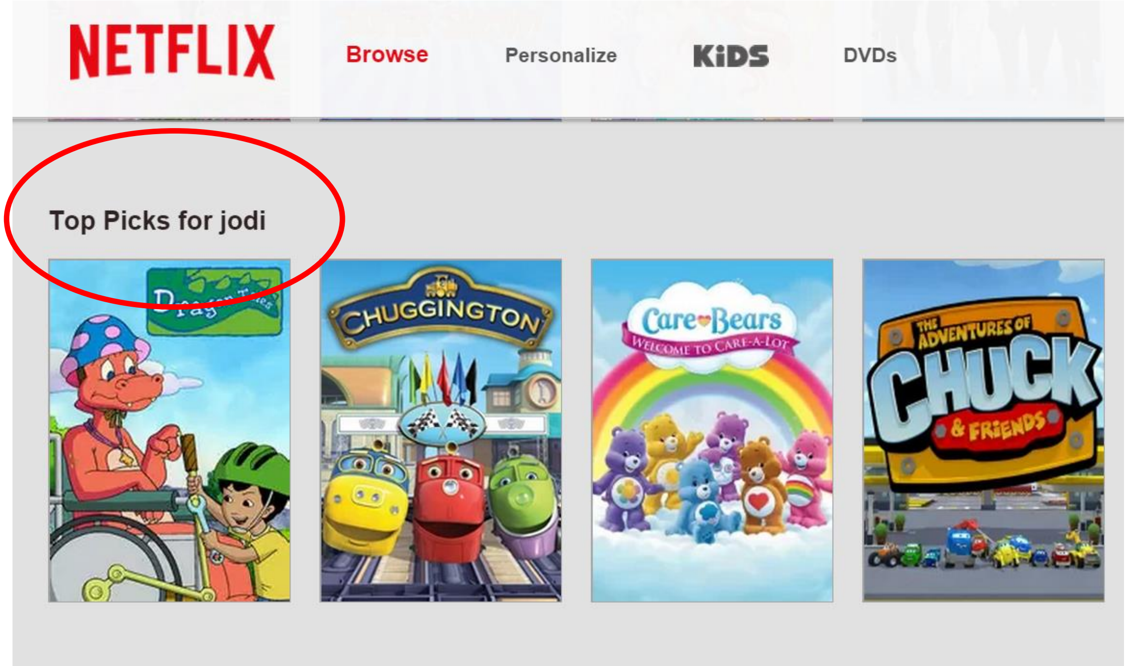


Example: Social Media



Example: Netflix





datasciencedojo
unleash the data scientist in you

Why recommendation systems?

For customer

- Narrow down the set of choices
- Discover new things
- Find things that are interesting
- Save time

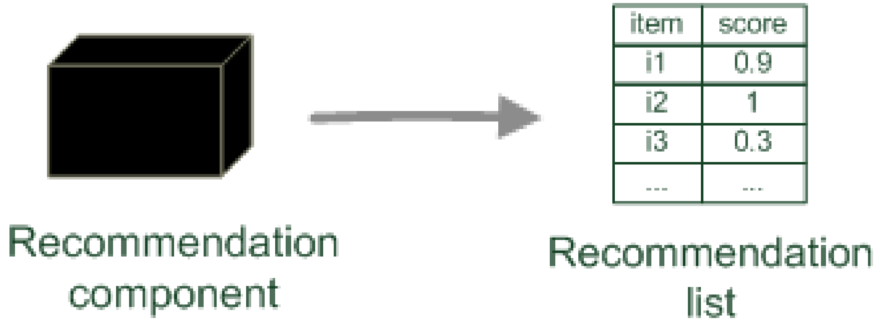
Why recommendation systems?

For businesses

- Increase the number of items sold
- Sell more diverse items
- Increase the user satisfaction
- Better understand what the user wants

Recommender Systems

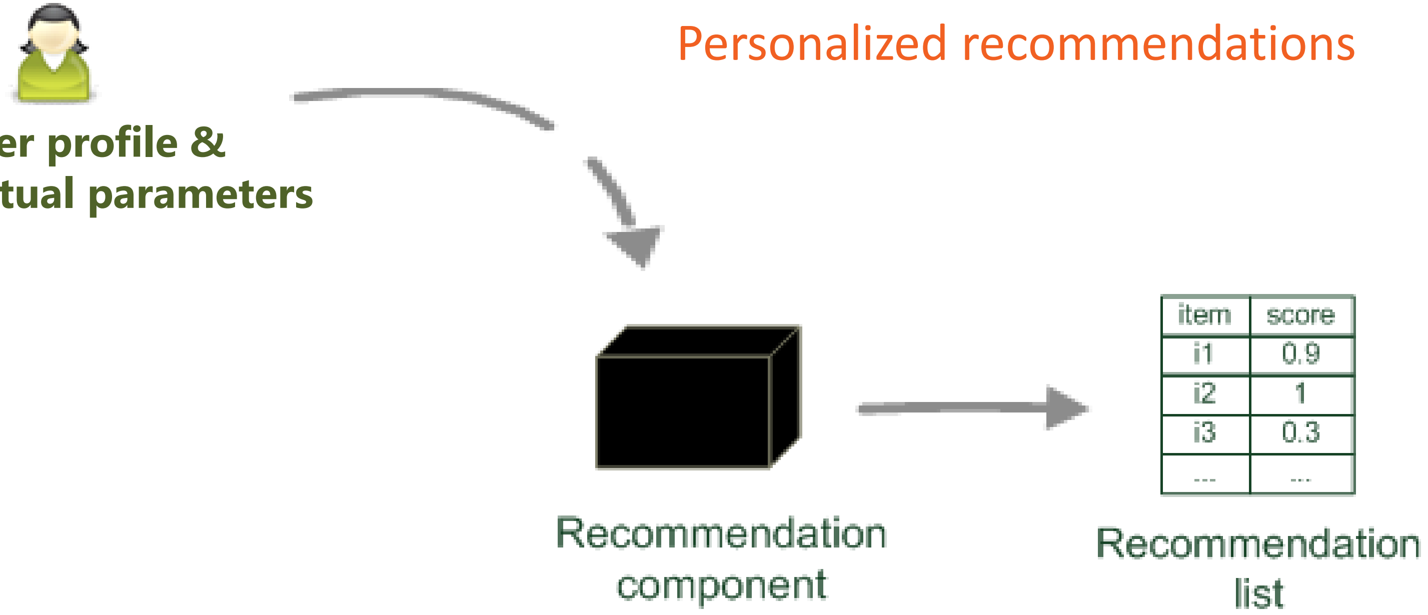
Recommender systems reduce information overload by estimating relevance



unleash the data scientist in you

Recommender Systems

Personalized recommendations



item	score
i1	0.9
i2	1
i3	0.3
...	...

Overview

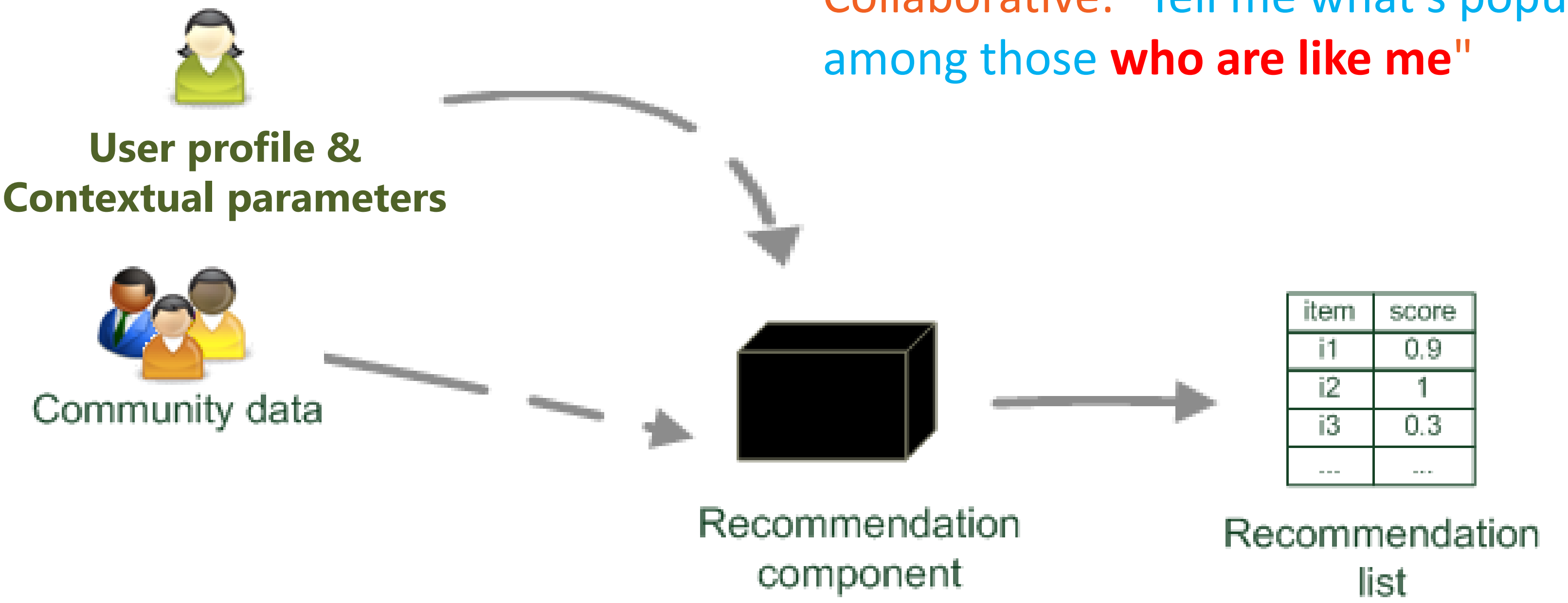
- What are Recommender Systems?
- How do they work?
    - **Collaborative Recommendation**
    - Content-Based Recommendation
- How do we evaluate them?
- Example using Azure ML

Collaborative Filtering

- Maintain a database of many users' ratings of a variety of items.
- For a given user, find other similar users whose ratings strongly correlate with the current user.
- Recommend items rated highly by these similar users, but not rated by the current user.

Collaborative Filtering (CF)

Collaborative: "Tell me what's popular among those **who are like me**"



Collaborative Filtering

- Most popular recommendation algorithm
  - Used by large, commercial e-commerce sites
  - Well-understood, variety of algorithms
  - Applicable to many domain (books, movies, songs,…)

- Approach: borrow the "wisdom of the crowd" to recommend items

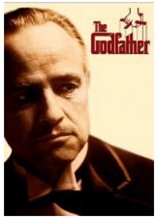
Collaborative Filtering

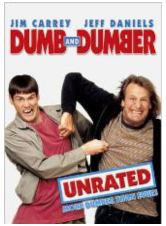
- Assumption:
  - Users give ratings to items
  - Users who has similar tastes in the past, have similar tastes in the future.
- User-based collaborative
- Item-based collaborative

Collaborative Filtering

- Assumption:
  - Users give ratings to items
  - Users who has similar tastes in the past, have similar tastes in the future.
- **User-based collaborative**
- Item-based collaborative

Movie Rating Example





Alice	5	3	4	4	?
Bob	3	1	2	3	3
Chris	4	3	4	3	5
Donna	3	3	1	5	4
Evi	1	5	5	2	1

Movie Rating Example

Goal: Given Alice is an "active" user, we want to predict the rating of movie *i* Alice hasn't seen before.

- Find set of users who liked the same items as Alice in the past and also had rated movie *i*
- Predict Alice's rating on movie *i*
- Repeat for all items Alice has not seen and recommend the best rated.

User-Based collaborative filtering

- How do we define similarity?
- How many neighbor should we include?
- How to generate prediction from neighbors' ratings?

User-Based collaborative filtering

- **Nearest neighbors**
  - **Pearson correlation**

j : Alice
k: Bob
P: set of items, rated by Alice and Bob

j,k : users

$r_{j,p}$: *rating of user j for item p*

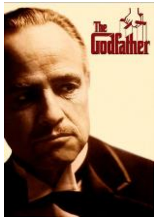
$\bar{r}_j$ *and* $\bar{r}_k$ *are the average ratings of user j and user k over all items*

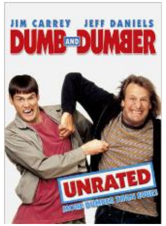
P: set of items, rated both by j and k

Possible similarity values between -1 and 1

$$\boldsymbol{sim(j,k)} = \frac{\sum_{p\,\in \boldsymbol{P}} (\boldsymbol{r}_{j,p} - \bar{\boldsymbol{r}}_j)(\boldsymbol{r}_{k,p} - \bar{\boldsymbol{r}}_k)}{\sqrt{\sum_{p\,\in \boldsymbol{P}} (\boldsymbol{r}_{j,p} - \bar{\boldsymbol{r}}_j)^2}\sqrt{\sum_{\boldsymbol{p}\,\in \boldsymbol{P}} (\boldsymbol{r}_{k,p} - \bar{\boldsymbol{r}}_k)^2}}$$

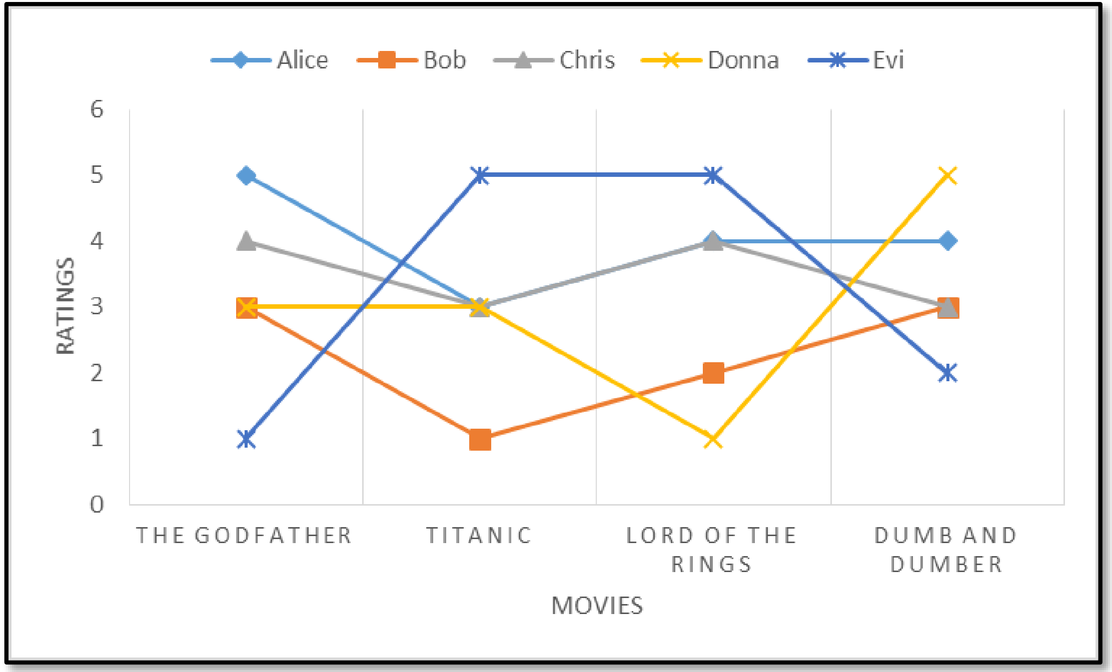
Pearson Correlation





Alice	5	3	4	4	?	
Bob	3	1	2	3	3	sim=0.85
Chris	4	3	4	3	5	sim=0.90
Donna	3	3	1	5	4	sim=0.70
Evi	1	5	5	2	1	sim=0.79

Pearson Correlation



datasciencedojo
unleash the data scientist in you

Making recommendations

- Making predictions is typically not the ultimate goal
- Usual approach
  - Rank items based on their predicted ratings
- However
  - This might lead to the inclusion of (only) niche items
- Better approach
  - Optimize according to a given rank evaluation metric

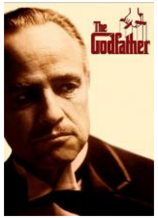
Collaborative Filtering

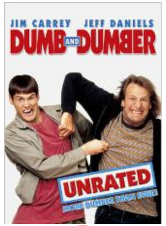
- Assumption:
  - Users give ratings to items
  - Users who has similar tastes in the past, have similar tastes in the future.
- User-based collaborative
- **Item-based collaborative**

Item-based collaborative filtering

- Basic idea:
  - Use the similarity between items (and not users) to make predictions
- Example:
  - Look for movies that are similar to movie 5
  - Take Alice's ratings for these items to predict the rating for movie 5

Movie Rating Example





Alice	5	3	4	4	?
Bob	3	1	2	3	3
Chris	4	3	4	3	5
Donna	3	3	1	5	4
Evi	1	5	5	2	1

Item-based Similarity Measurements

- cosine similarity

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

- Adjusted cosine similarity

$$sim(\vec{a},\vec{b}) = \frac{\sum_{u\in U}(r_{u,a}-\overline{r_u})(r_{u,b}-\overline{r_u})}{\sqrt{\sum_{u\in U}(r_{u,a}-\overline{r_u})^2}\sqrt{\sum_{u\in U}(r_{u,b}-\overline{r_u})^2}}$$

Collaborative Filtering Pros

- well-understood and proven
- works well in many domains
- no knowledge engineering required
- serendipity of results

Collaborative Filtering Cons

Data sparsity: New user needs to indicate preferences for sufficient number of items before getting recommendations

Scalability: Millions of customers (M) and millions of items (N).

Grey Sheep and Black Sheep: Grey sheep are users with inconsistent recommendations. Black sheep are the users with idiosyncratic preferences.

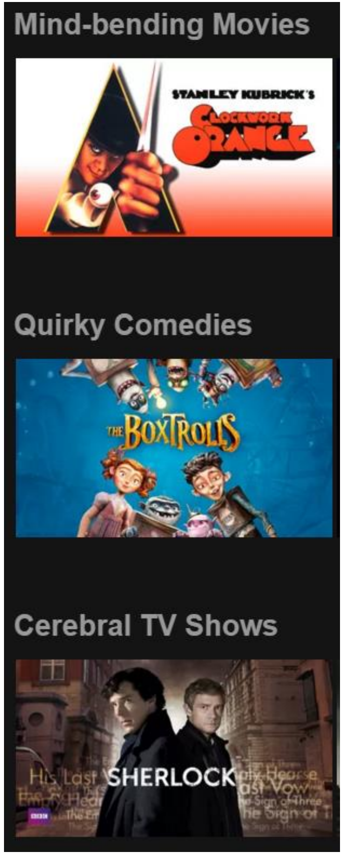
Collaborative Filtering Cons

Shilling: Intentional manipulation of ratings of your own products and competitors products

Diversity and Long Tail: Rich get richer.

Cold Start: Need initial customer/rating database

Example: Netflix



Overview

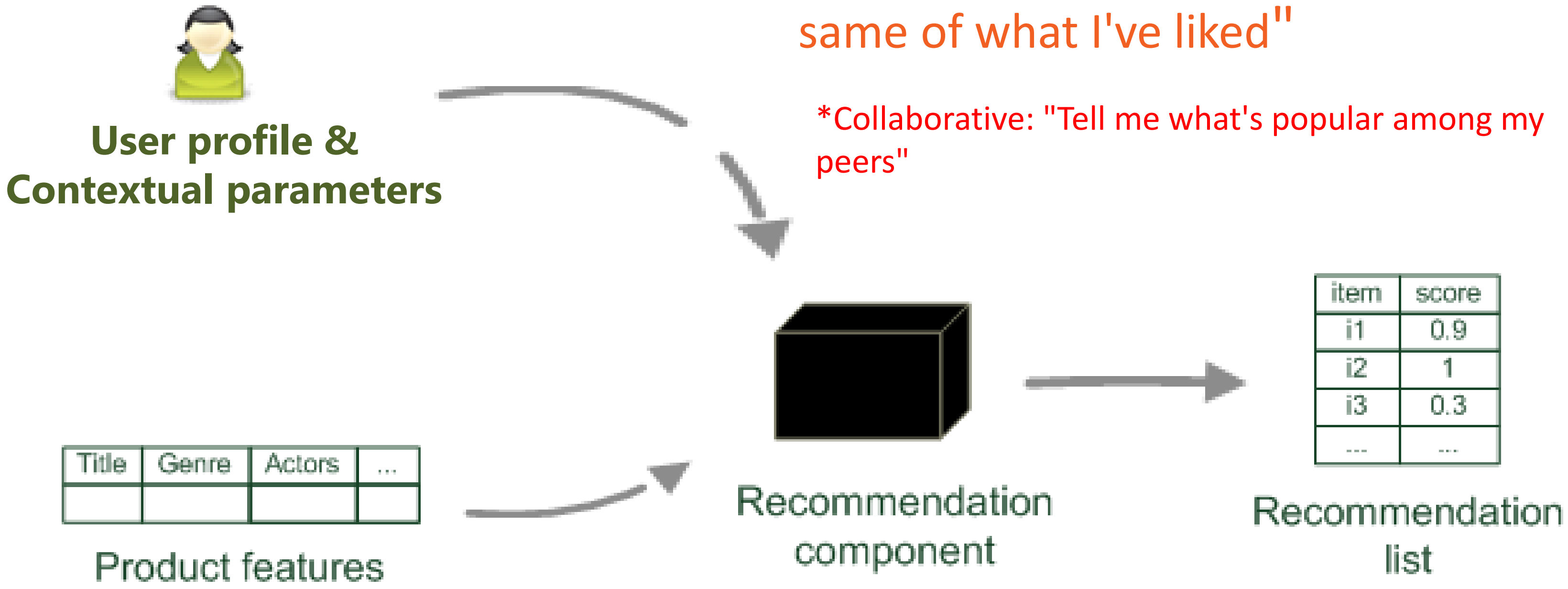
- What are Recommender Systems?
- How do they work?
  - Collaborative Recommendation
  - **Content-Based Recommendation**
- How do we evaluate them?
- Example using Azure ML

datasciencedojo
unleash the data scientist in you

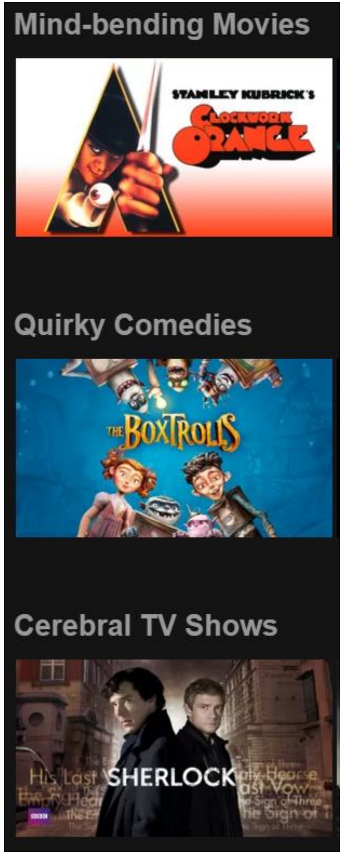
Content-based recommendation

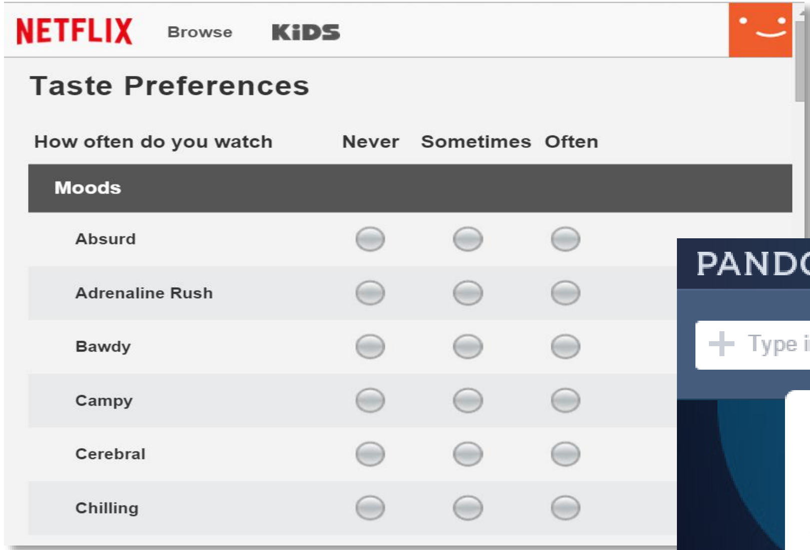
Content-based: "Show me more of the same of what I've liked"

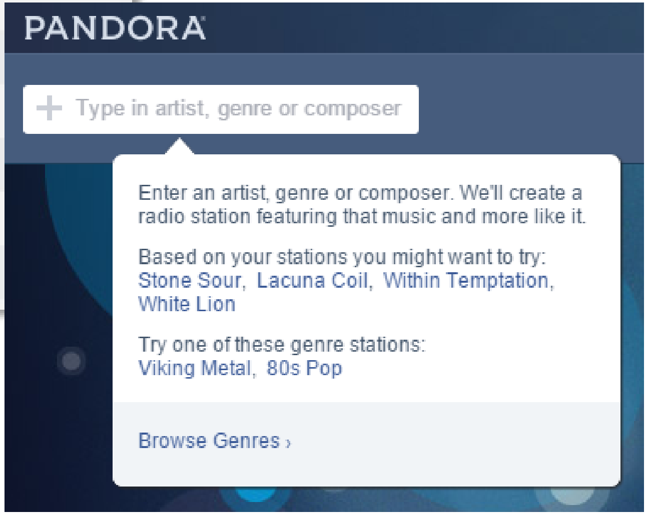
*Collaborative: "Tell me what's popular among my peers"



Content-based recommendation







datasciencedojo
unleash the data scientist in you

Content-based recommendation

Related to Items You've Viewed See more

Data Science

Web News Images Books Videos More Search tools

About 288,000,000 results (0.42 seconds)

Data science - Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/**Data_science** ▾ Wikipedia ▾
Data Science is an interdisciplinary field about processes and systems to extract knowledge or insights from large volumes of data in various forms, either ...
Overview - History - Domain specific interests - Criticism

Data Science | Coursera
https://www.coursera.org/specializations/jhu**datascience** ▾ Coursera ▾
Become an expert with **Data Science** Specialization offered by Johns Hopkins University. Take free online classes from 120+ top universities and educational ...

Certificate in Data Science - UW Professional & Continuing ...
www.pce.uw.edu/certificates/**data-science**.html ▾
University of Washington offers a certificate program in **data science**, with flexible evening and online classes to fit your schedule.
Jan 14, 2016 Online
Mar 28, 2016 Bellevue

Content-based recommendation

Recommend items that are “similar” to the user preferences

What do we need:

- Item Profiles: content of the items
- User profiles: preferences of the user.
  - User specified or based on item ratings

Item Profile Strategies

- **Expert Labeling**
  - Assign keywords based on content
  - Good for songs, movies, etc
  - May be provided by creators/distributors
  - Crowd sourcing?

Content-based recommendation

- **Information Retrieval (IR)**
  - Used for text documents (web pages, books, tweets)
  - Based on word content of document set
  - No expert knowledge involved
  - Can be keyword or full dictionary based

Content-based recommendation

- **Prediction: Simple approach**
  - Compute the similarity of an item and user profile based on keyword overlap
  - $\text{sim}(b_i, b_j) = \dfrac{2 * |keywords(b_i) \cap keywords(b_j)|}{|keywords(b_i)| + |keywords(b_j)|}$

Simple approach: drawbacks

- Not every word has similar importance
- Longer documents have a higher chance to have an overlap with the user profile
- Automated extraction particularly problematic

TF-IDF

- Common Solution: TF-IDF
  - **Term Frequency:** Measures, how often a term appears (density in a document)
    - Assuming that important terms appear more often
    - Normalization has to be done in order to take document length into account
  - **Inverse Document Frequency:** Aims to reduce the weight of terms that appear in all documents

Term Frequency

- **Term frequency (TF)**
  - Let *freq(t,d)* number of occurrences of keyword *t* in document *d*
  - Let *max{freq(w,d)}* denote the highest number of occurrences of another keyword of *d*
  - $TF(t,d) = \frac{freq(t,d)}{\max\{freq(w,d) : w \in d\}}$

datasciencedojo
unleash the data scientist in you

Inverse Document Frequency

- **Inverse Document Frequency (IDF)**
  - N: number of all recommendable documents
  - n(t): number of documents in which keyword t appears
  - $IDF(t) = log\frac{N}{n(t)}$

TF-IDF

- Compute the overall importance of keywords
  - Given a keyword *t* and a document *d*

$$TF\text{-}IDF\,(t,d) = TF(t,d) * IDF(t)$$

datasciencedojo
unleash the data scientist in you

TF-IDF Exercise

- http://lsirwww.epfl.ch/courses/dis/2006ws/exercises/IR/Exercise8.htm
- http://lsirwww.epfl.ch/courses/dis/2006ws/exercises/IR/Exercise%208%20solution%202007.pdf

Recommending items

- Simple method: nearest neighbors
  - Given a set of documents D already rated by the user (like/dislike, ratings)
    - Find the n nearest neighbors of a not-yet-seen item *i* in D
    - Take these ratings to predict a rating/vote for *i*

Example taken from http://informationretrieval.org

Recommending items

- Query-based retrieval: Rocchio's method
- Probabilistic methods
- linear classification/regression algorithms
- etc

Example taken from http://informationretrieval.org

Content-based recommenders

Advantages

- No community required: Only need the items and a single user profile for recommendation.
- Transparency: CB models can tell you why they recommend an item, not subject to vagaries of human taste
- No cold start: new items can be suggested before being rated by a substantial number of users.

Content-based recommenders

Disadvantages

- Limited content analysis: requires well annotated content for good recommendations.
- Over-specialization
- New users: limited user information results in bad recommendations.

Overview

- What are Recommender Systems?
- How do they work?
  - Collaborative Recommendation
  - Content-Based Recommendation
- **How do we evaluate them?**
- Example using Azure ML

Evaluating Recommendation

- Metrics measure error rate
  - **Mean Absolute Error (*MAE*)** computes the deviation between predicted ratings and actual ratings

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | p_i - r_i |$$

  - **Root Mean Square Error (*RMSE*)** is similar to *MAE*, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (p_i - r_i)^2}$$

datasciencedojo
unleash the data scientist in you

datasciencedojo
unleash the data scientist in you

Metrics

- Order matters, not exact ranking value
- Discounted cumulative gain (DCG)
  - Logarithmic reduction factor

$$DCG_{pos} = rel_1 + \sum_{i=2}^{pos} \frac{rel_i}{\log_2 i}$$

Where:

- *pos* denotes the position up to which relevance is accumulated
- rel_i returns the relevance of recommendation at position *i*

Metrics

- **Ideal discounted cumulative gain (IDCG)**
  - Assumption that items are ordered by decreasing relevance

$$IDCG_{pos} = rel_1 + \sum_{i=2}^{|h|-1} \frac{rel_i}{\log_2 i}$$

- **Normalized discounted cumulative gain (nDCG)**

$$nDCG_{pos} = \frac{DCG_{pos}}{IDCG_{pos}}$$

  - Normalized to the interval [0..1]

QUESTIONS

Overview

- What are Recommender Systems?
- How do they work?
  - Collaborative Recommendation
  - Content-Based Recommendation
- How do we evaluate them?
- **Example using Azure ML**