Recommendation Systems

Data Science Dojo

datasciencedojo
unleash the data scientist in you

Overview

- Introduction
  - Collaborative vs Content-based
- How do they work?
  - Data structure
  - Ranking by similarity
  - Predicting
  - Evaluation
- Advantages/Disadvantages
- Example using Azure ML

Overview

- **Introduction**
  - Collaborative vs Content-based
- How do they work?
  - Data structure
  - Ranking by similarity
  - Predicting
  - Evaluation
- Advantages/Disadvantages
- Example using Azure ML

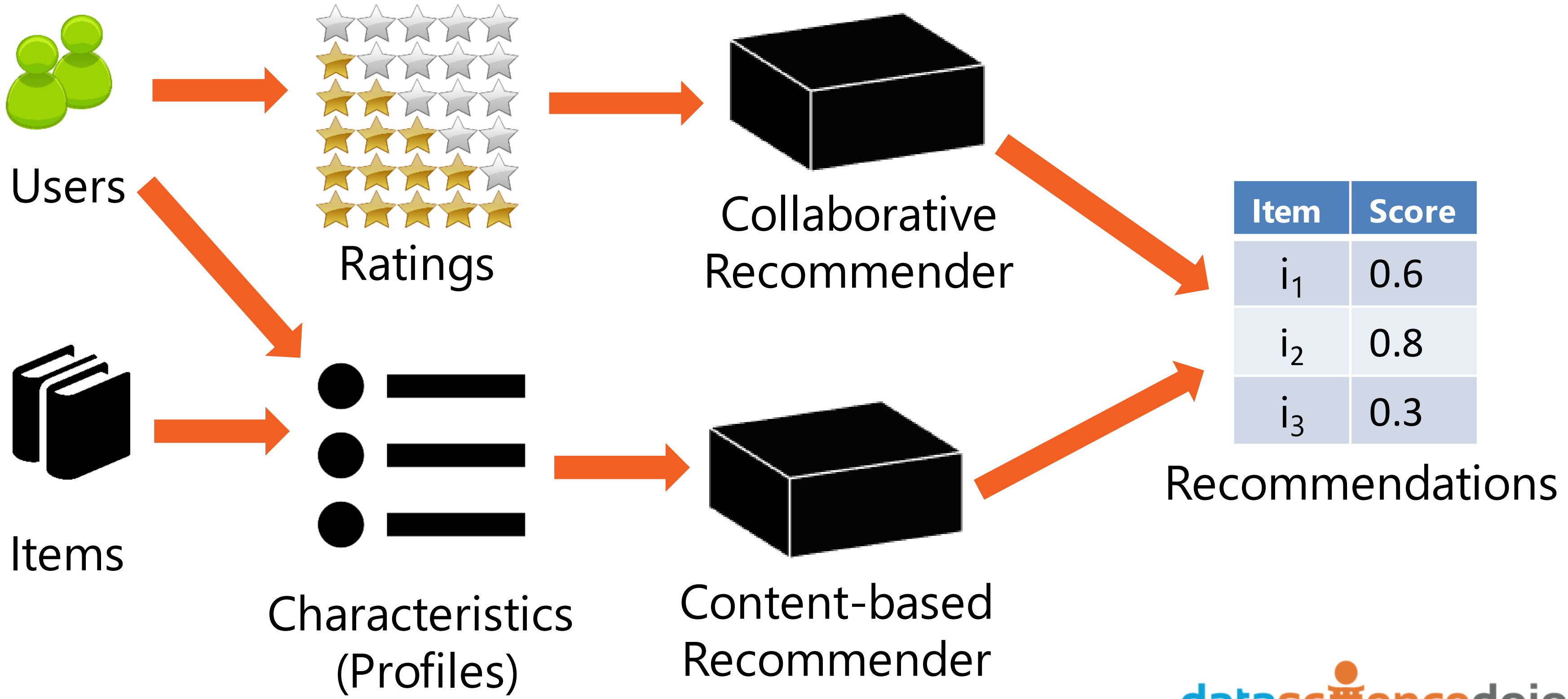
Recommendation Systems

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

Why recommendation systems?

- For Customers
  - Narrow down the set of choices
  - Discover new, interesting things
  - Save time
- For Business
  - Increase the number of items sold
  - Sell more diverse items
  - Better understand what the user wants

Two Types of Recommenders



Item	Score
i_1	0.6
i_2	0.8
i_3	0.3

datasciencedojo
unleash the data scientist in you

Two Types of Recommenders

<table>
<tr><th><u>Collaborative</u></th><th><u>Content-Based</u></th></tr>
<tr><td>

- 'Give me items that **people like me** enjoy'
- Wisdom of the crowds
- Widely applicable

</td><td>

- 'Give me items similar to **items I like**'
- Content analysis based
- Related to Information Retrieval

</td></tr>
</table>

Two Types of Recommenders

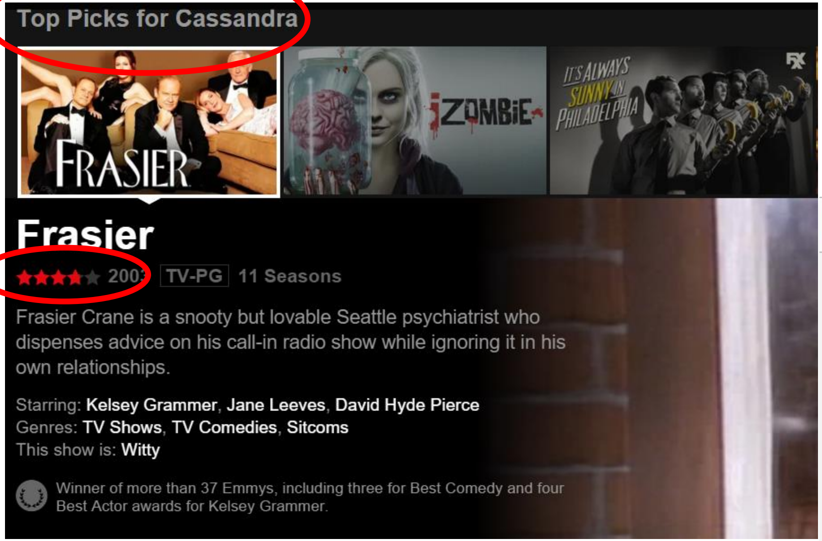
Collaborative

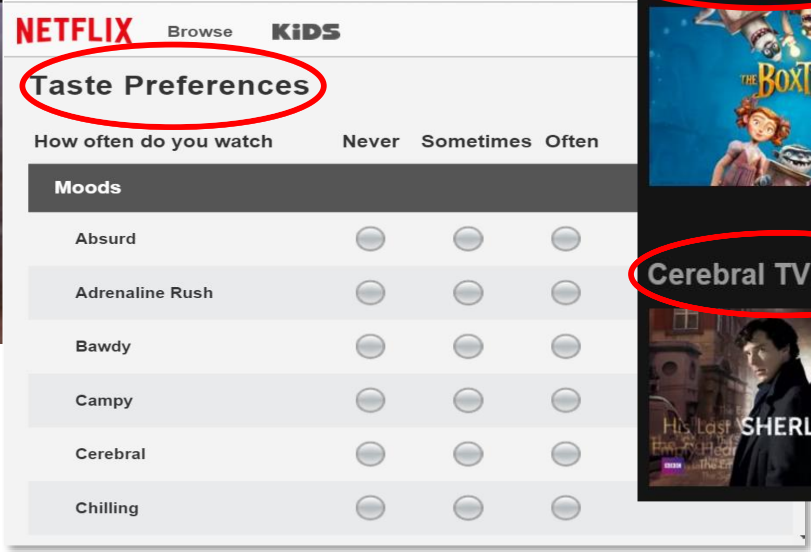
- Users, Items, & Ratings
- Use Ratings of similar Users to recommend unseen Items

Content-Based

- User & Item profiles
- Use overlap of User and Item characteristics to recommend unseen items

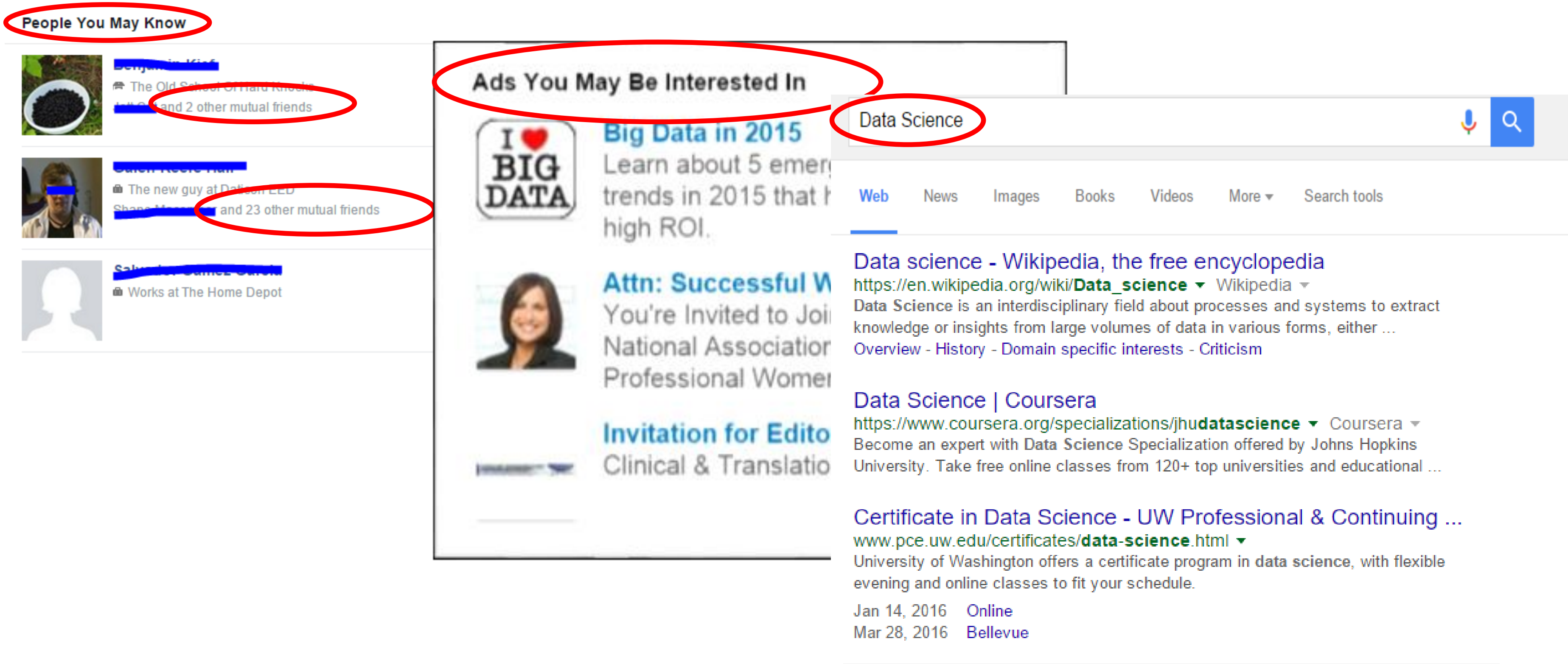
Example: Netflix





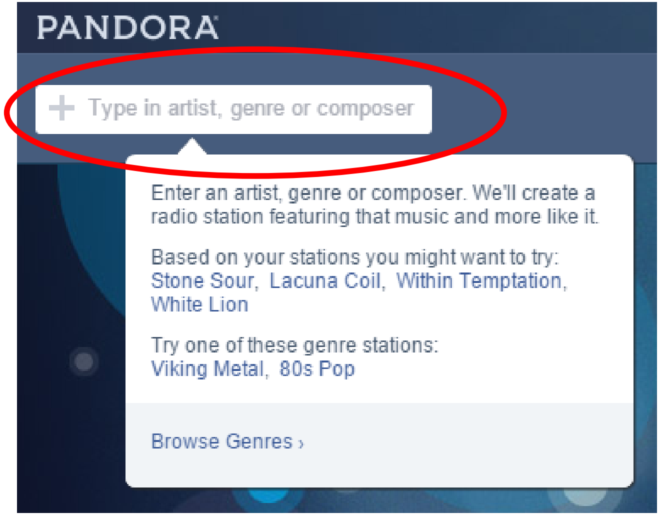
unleash the data scientist in you

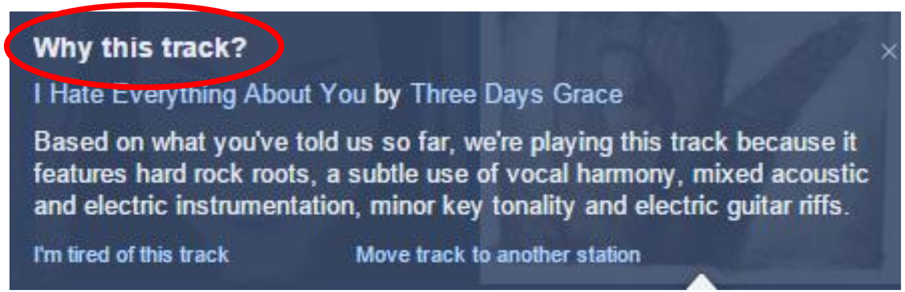
Example: Social Media & Search

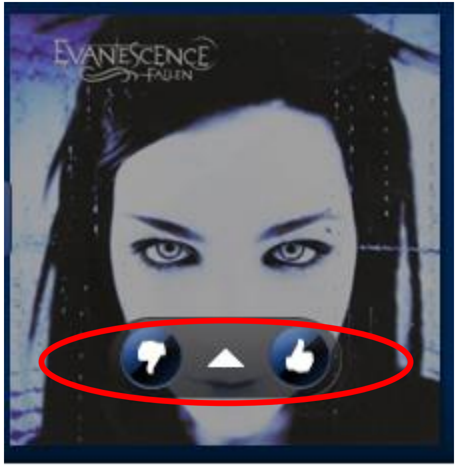


datasciencedojo
unleash the data scientist in you

Example: Pandora

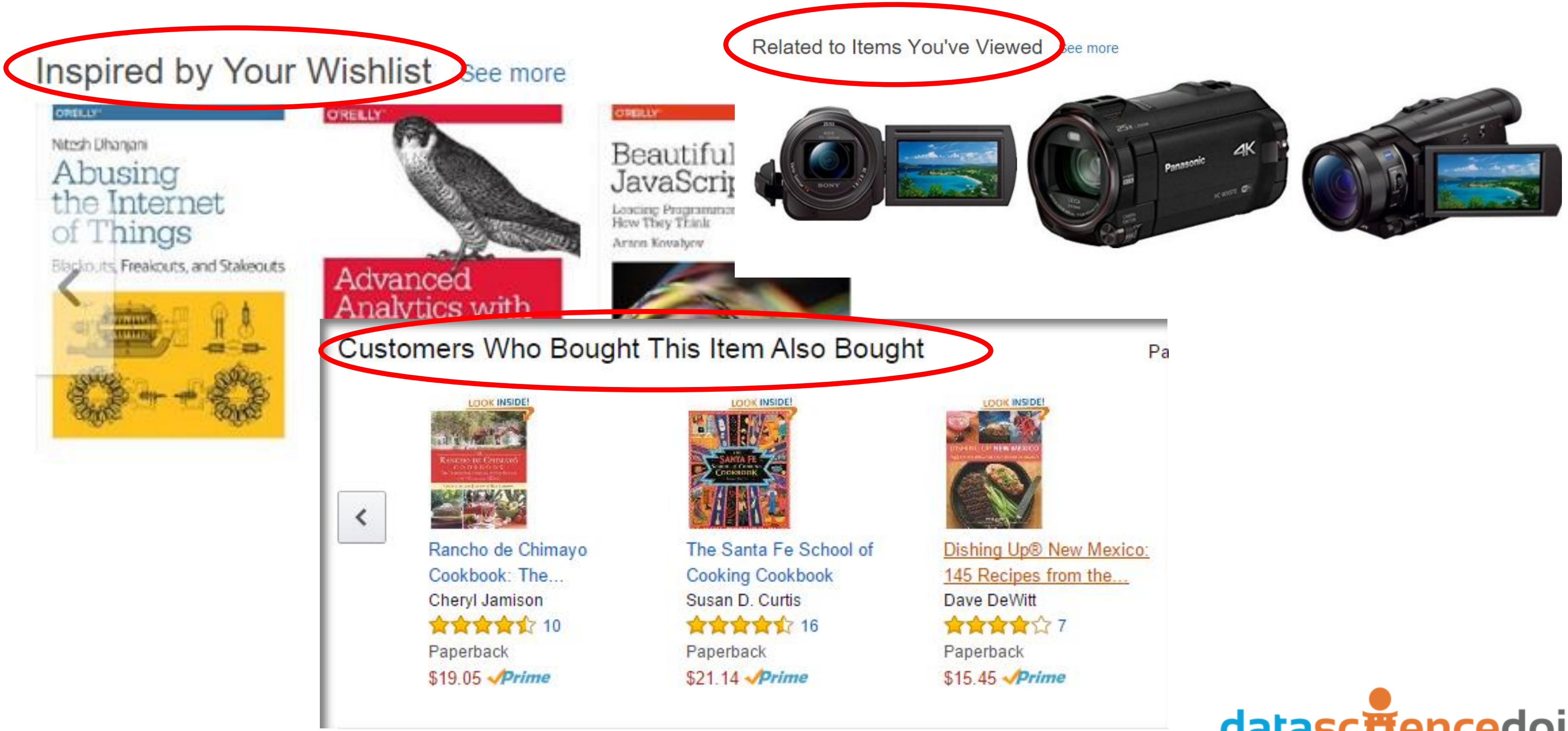






datasciencedojo
unleash the data scientist in you

Example: Amazon



datasciencedojo
unleash the data scientist in you

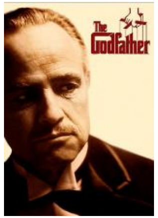
Overview

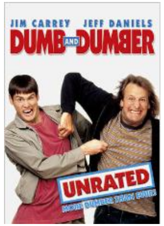
- Introduction
  - Collaborative vs Content-based
- **How do they work?**
  - **Data structure**
  - Ranking by similarity
  - Predicting
  - Evaluation
- Advantages/Disadvantages
- Example using Azure ML

Data Structure

- What kind of data?
  - Collaborative
    - Ratings of Items by Users
  - Content-based
    - Characteristic profiles of Users and Items

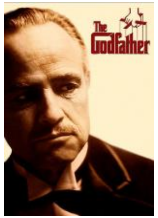
Data Structure - Collaborative

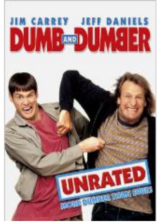




Alice	5	3	4	4	?
Bob	3	1	2	3	3
Chris	4	3	4	3	5
Donna	3	3	1	5	4
Evi	1	5	5	2	1

Data Structure – Content-based





Item/User	Drama?	Comedy?	Adventure?	Romance?
The Godfather	5	1	2	1
Titanic	4	3	2	5
Lord of the Rings	4	2	5	1
Dumb & Dumber	1	5	2	2
Spirited Away	5	3	5	2
Alice	5	4	1	4
Bob	3	1	1	1
Chris	4	2	5	2

Content-based: User Profiles

- **User Provided**
  - Ask for preferences
  - Needs accounts
  - Often low completion rates
- **Automated Generation**
  - Cookies follow behavior
  - No user persistence (often)
  - Loss in translation

Content-based: Item Profiles

- **Expert Labeling**
  - Assign keywords based on content
  - May be provided by creators/distributors
  - Crowd sourcing?
- **Automated Indexing**
  - Used for text documents
  - Based on word content of document set
  - No expert knowledge involved

Overview

- Introduction
  - Collaborative vs Content-based
- **How do they work?**
  - Data structure
  - **Similarity**
  - Predicting
  - Evaluation
- Advantages/Disadvantages
- Example using Azure ML

Similarity Measurements

- Given two vectors $\vec{x}$ and $\vec{y}$ with n components each
  - Ratings of User x and User y
  - Ratings for Item x and Item y
  - Profiles of User x and Item y
- How similar are the Users/Items?

Similarity Measurements

- Pearson's Correlation

$$sim(\vec{x},\vec{y}) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

- Cosine Similarity

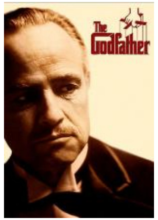
$$sim(\vec{x},\vec{y}) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}| * |\vec{y}|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

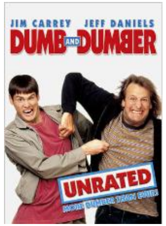
Collaborative: User-Based

- Goal: Predict User u's rating on a movie m they haven't seen
  - Find the N most similar Users to u who have seen m
  - Use their ratings to predict u's rating

Collaborative: User-based

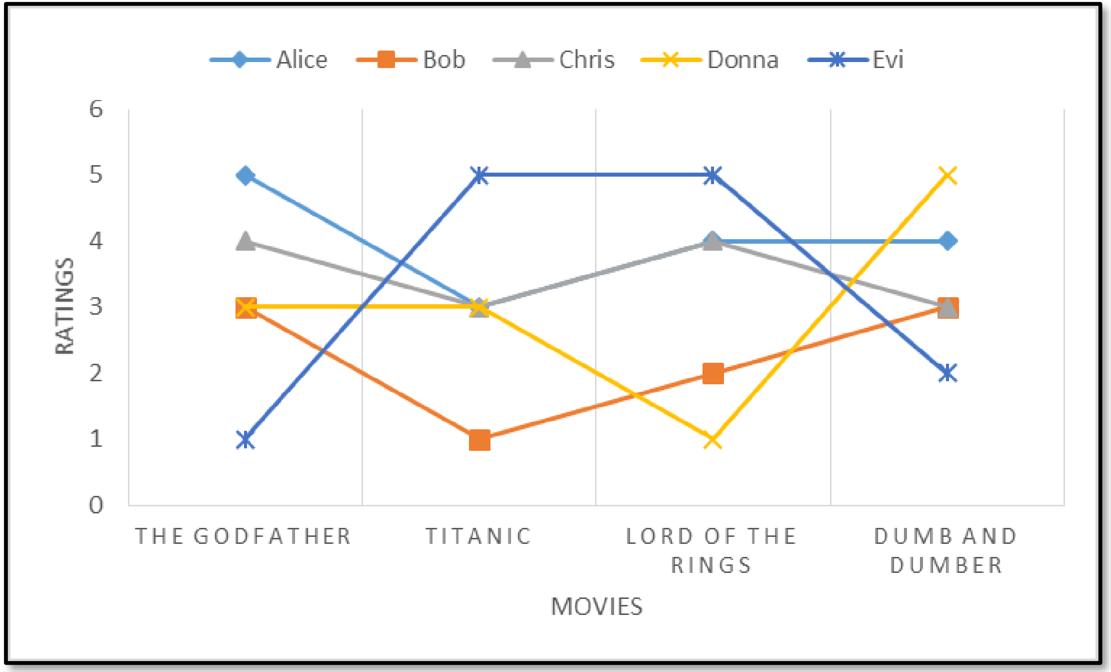
Which metric should we use?





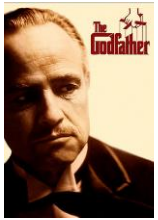
Alice	5	3	4	4	?	
Bob	3	1	2	3	3	sim = ?
Chris	4	3	4	3	5	sim = ?
Donna	3	3	1	5	4	sim = ?
Evi	1	5	5	2	1	sim = ?

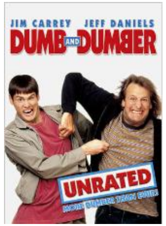
Collaborative: User-based



Collaborative: User-based

Pearson's correlation corrects for varied baselines





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

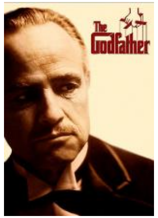
datasciencedojo
unleash the data scientist in you

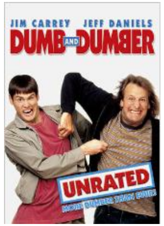
Collaborative: Item-based

- Alternate approach:
  - Use the similarity between items (and not users) to make predictions
  - Look for movies that are similar to movie m
  - Take **Alice**'s ratings for these items to predict the rating for movie m

Collaborative: Item-based

Which metric should we use?



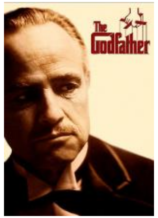


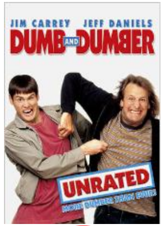
Alice	5	3	4	4	?
Bob	3	1	2	3	3
Chris	4	3	4	3	5
Donna	3	3	1	5	4
Evi	1	5	5	2	1

sim = ? sim = ? sim = ? sim = ?

Collaborative: Item-based

Cosine similarity allows for objective good/bad



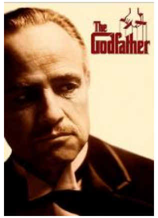


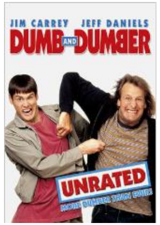
Alice	5	3	4	4	?
Bob	3	1	2	3	3
Chris	4	3	4	3	5
Donna	3	3	1	5	4
Evi	1	5	5	2	1
	sim=0.99	sim=0.74	sim=0.72	sim=0.93	

Content-based: Similarity

- Goal: Return a recommendation list of items for each user
  - Find similarity of each User to each Item
  - Order Items by similarity

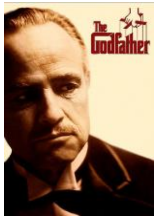
Content-based: Similarity

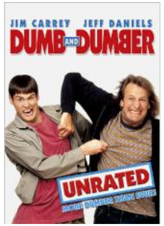




Item/User	Drama?	Comedy?	Adventure?	Romance?
The Godfather	5	1	2	1
Titanic	4	3	2	5
Lord of the Rings	4	2	5	1
Dumb & Dumber	1	5	2	2
Spirited Away	5	3	5	2
Alice	5	4	1	4
Bob	3	1	1	1
Chris	4	2	5	2

Content-based: Similarity





Alice	0.83	0.96	0.72	0.79	0.83
Bob	0.99	0.86	0.85	0.59	0.91
Chris	0.87	0.82	0.99	0.69	0.99

- Cosine similarity doesn't erase baselines
- Predict order, not exact rating

Overview

- Introduction
  - Collaborative vs Content-based
- **How do they work?**
  - Data structure
  - Similarity
  - **Predicting**
  - Evaluation
- Advantages/Disadvantages
- Example using Azure ML

Collaborative: Predictions

- Use "Aggregation Function"
- Choose N nearest neighbors to User u
- Combine each neighbor j's rating on Item i ($r_{j,i}$)
- Simple
  - $r_{u,i} = \frac{1}{N}\sum_{j=1}^{N} r_{j,i}$
- Weighted & Centered
  - $r_{u,i} = \overline{\boldsymbol{r}_u} + \alpha \sum_{j=1}^{N} sim(j,u)(r_{j,i} - \overline{r_j})$

datasciencedojo
unleash the data scientist in you

Content-based: Predictions

- Simple
  - Rank in order of similarity
- Information retrieval techniques
  - Well studied, wide diversity of models
  - Classification algorithms

Overview

- Introduction
  - Collaborative vs Content-based
- **How do they work?**
  - Data structure
  - Similarity
  - Predicting
  - **Evaluation**
- Advantages/Disadvantages
- Example using Azure ML

Evaluating Recommendation

- **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

Metrics

- Order matters, not exact rating value
- Graded Relevance
  - Have humans assign scores to possible results
  - Ideal results will be ordered by relevance, high to low
- Discounted cumulative gain (DCG)
  - Logarithmic reduction factor

$$DCG_N = rel_1 + \sum_{i=2}^{N} \frac{rel_i}{\log_2 i}$$

Where:

- N is the length of the recommendation list
- rel_i returns the relevance of recommendation at position i

Metrics

- **Ideal discounted cumulative gain (IDCG)**
  - DCG value when items are ordered perfectly

$$IDCG_N = rel_1 + \sum_{i=2}^{N} \frac{rel_i}{\log_2 i}$$

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

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

  - Normalized to the interval [0..1]

Overview

- Introduction
  - Collaborative vs Content-based
- How do they work?
  - Data structure
  - Similarity
  - Predicting
  - Evaluation
- **Advantages/Disadvantages**
- Example using Azure ML

Advantages

Collaborative

- Wide applicability
- Serendipity
- Simple

Content-based

- No community needed
- Transparency
- Good cold start

Disadvantages

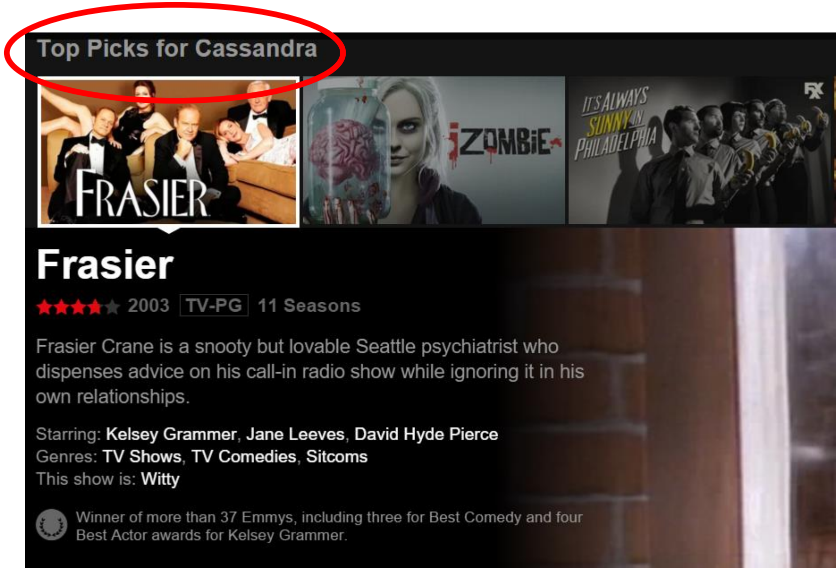
Collaborative

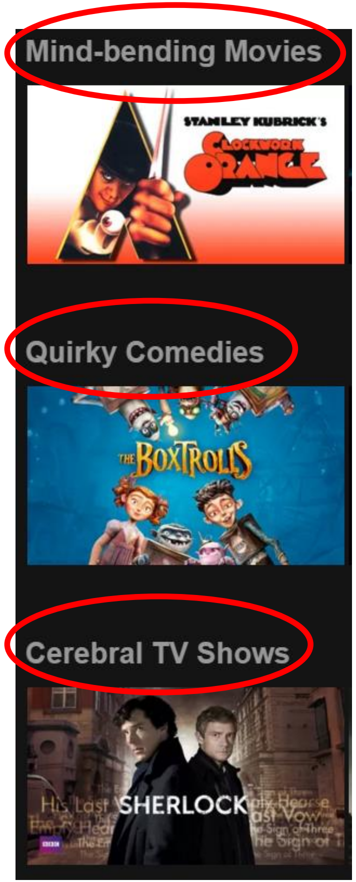
- Poor cold start
- Grey Sheep
  - Shared accounts
- Shilling
- Poor scaling

Content-based

- Limited profiles
  - New users
  - Cost of expert labeling
- Over-specialization
  - Lack of diversity

Back to Netflix





datasciencedojo
unleash the data scientist in you

QUESTIONS

Overview

- Introduction
  - Collaborative vs Content-based
- How do they work?
  - Data structure
  - Similarity
  - Predicting
  - Evaluation
- Advantages/Disadvantages
- **Example using Azure ML**

Collaborative Filtering Pros

- **Wide applicability**
  - Usable in wildly different domains
- **Well-understood**
  - Most well studied type of recommender
- **Simple**
  - No knowledge engineering required
- **Serendipity**
  - Odd recommendations that are very good

Collaborative Filtering Cons

- **Data sparsity & Cold Start**
  - New users need to indicate preferences for sufficient number of items before recommendations are good
  - Need initial customer/rating database
- **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 tend to get richer

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
- **Good 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**
  - Users will tend to be recommended items very similar to what they have enjoyed in the past
  - Very limited discoverability
- **New users**
  - Limited user information results in bad recommendations.

Similarity Measurement

- **Pearson correlation**

j,k : users

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

$\bar{\boldsymbol{r}}_j$ and $\bar{\boldsymbol{r}}_k$ 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

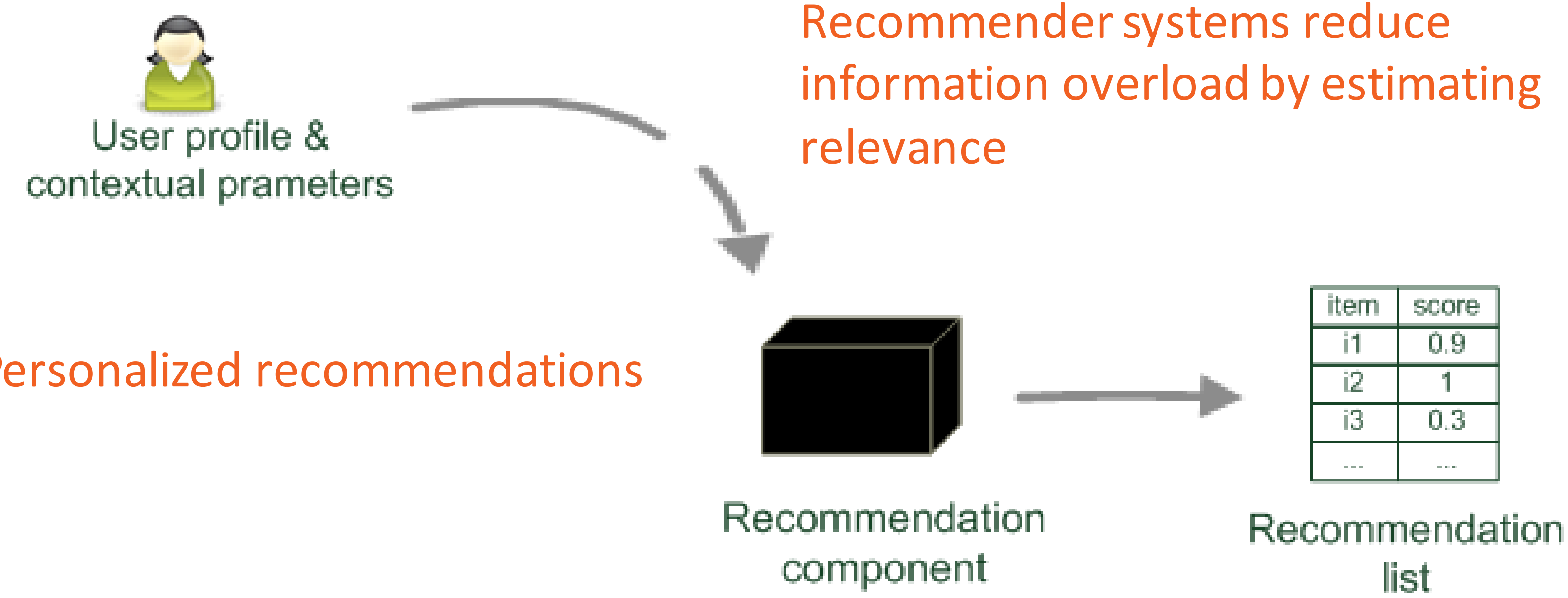
j : Alice
k: Bob
P: set of items, rated by Alice and Bob

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

Recommender Systems

Recommender systems reduce information overload by estimating relevance

Personalized recommendations

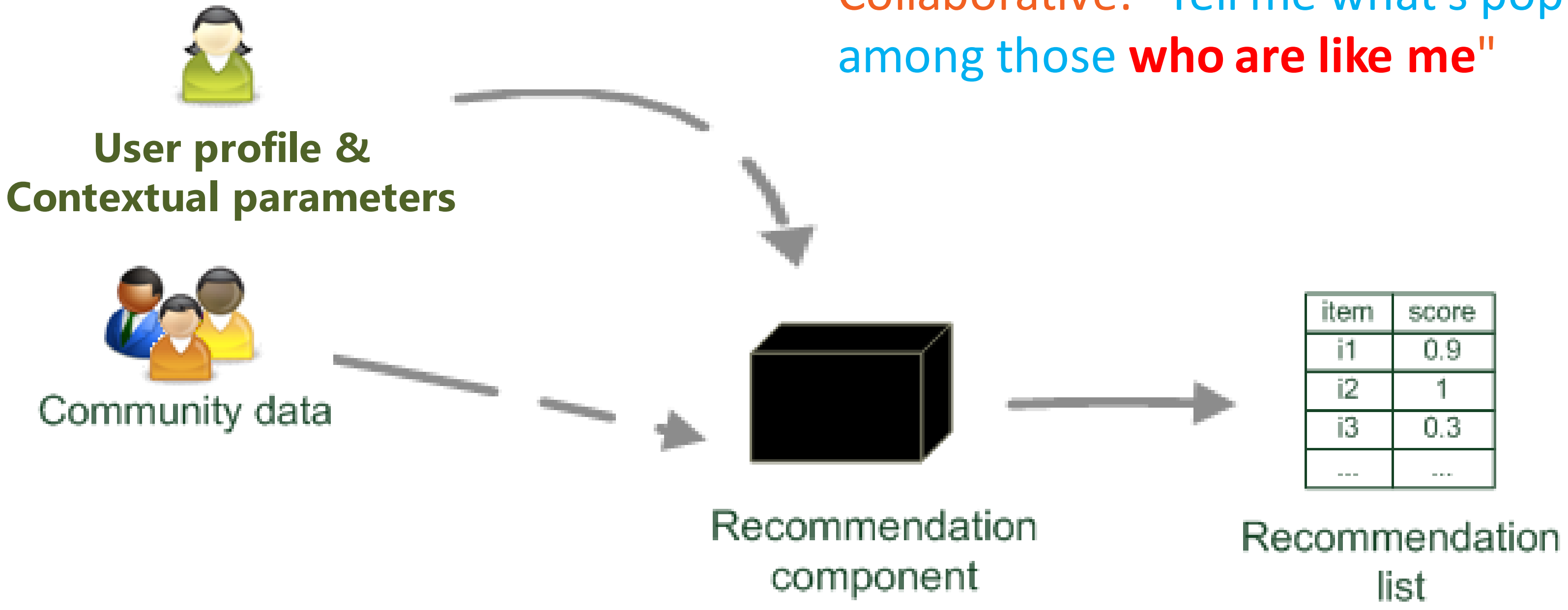


Overview

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

Collaborative Filtering (CF)

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



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

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

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

Collaborative Filtering

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

Collaborative Filtering

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

User-Based Collaborative Filtering

- How many neighbors should we include?
  - Choose a number – depends on size of data
- How do we define similarity?
- How to do we generate predictions from the neighbors' ratings?

Collaborative: User-based

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

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

Making prediction

$$pred(j,i) = \bar{r}_j + \frac{\sum_{k \in N} sim(j,k) * (r_{k,i} - \bar{r}_k)}{\sum_{k \in N} sim(j,k)}$$

j : Alice
k: Bob
i: movie Spirited Away

- 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 j user as a weight
- Add/subtract the neighbors' bias from the active user's average and use this as a prediction

datasciencedojo
unleash the data scientist in you

Making recommendations

- Prediction is typically not the ultimate goal
  - Rank items based on their predicted ratings
  - This might lead to the inclusion of (only) niche items
    - 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**

Making Predictions

- Sum over items rather than users
  - Simple
    - $r_{j,p} = \frac{1}{N} \sum_{q \in P} r_{j,q}$
  - Weighted & Centered
    - $r_{j,p} = \overline{\boldsymbol{r}_p} + \alpha \sum_{q \in P} simil(p, q\ )(r_{j,q} - \overline{\boldsymbol{r}_q})$

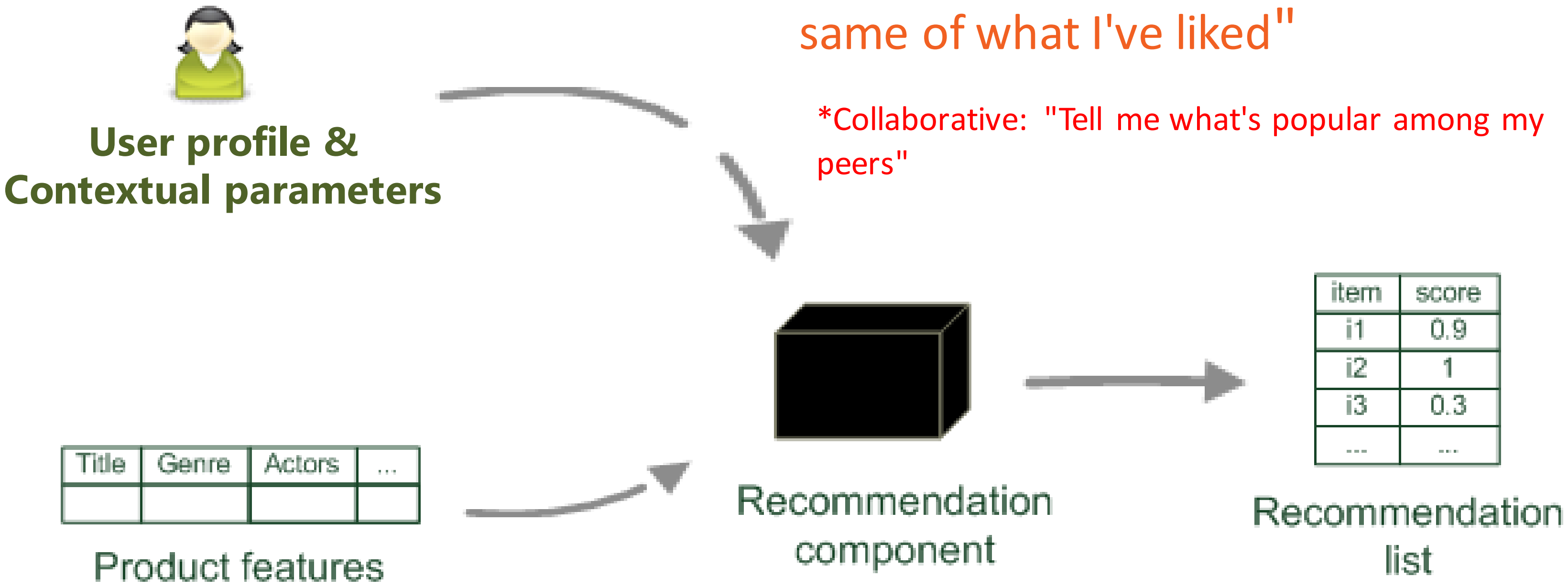
Overview

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

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

Recommend items that are “similar” to the user preferences

What do we need?

- Item Profiles: list of content-based keywords
- User profiles: preferences of the user.
  - User specified or based on past behavior

Content-based recommendation

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

Simple approach: drawbacks

- Not every word has similar importance
- Longer documents have a higher chance to have an overlap with the user profile
- **Solution:** TF-IDF

TF-IDF Search Exercise

- **D1** = "If it walks like a duck and quacks like a duck, it must be a duck."
- **D2** = "Beijing Duck is mostly prized for the thin, crispy duck skin with authentic versions of the dish serving mostly the skin."
- **D3** = "Bugs' ascension to stardom also prompted the Warner animators to recast Daffy Duck as the rabbit's rival, intensely jealous and determined to steal back the spotlight while Bugs remained indifferent to the duck's jealousy, or used it to his advantage. This turned out to be the recipe for the success of the duo."
- **D4** = "6:25 PM 1/7/2007 blog entry: I found this great recipe for Rabbit Braised in Wine on cookingforengineers.com."
- **D5** = "Last week Li has shown you how to make the Sechuan duck. Today we'll be making Chinese dumplings (Jiaozi), a popular dish that I had a chance to try last summer in Beijing. There are many recipies for Jiaozi."

- **Dictionary:** {beijing, dish, duck, rabbit, recipe}

Query: "Beijing duck recipe"

Document Matrix

Query: "Beijing duck recipe"

Word	IDF
Beijing	.398
Dish	.398
Duck	.097
Rabbit	.398
Recipe	.222

	Beijing	**Dish**	**Duck**	**Rabbit**	**Recipe**
D1	0	0	0.097	0	0
D2	0.199	0.199	0.097	0	0
D3	0	0	0.097	0.199	0.111
D4	0	0	0	0.398	0.222
D5	0.398	0.398	0.097	0	0.222
Query	**1*.398**	**0**	**1*.097**	**0**	**1*.222**

TF-IDF Search Exercise

- Cosine similarity of query and each doc
- $D1 = [0, 0, 0.097, 0, 0]$
- $Q = [0.398, 0, 0.097, 0, 0.222]$
- $\cos(D1, Q) = \frac{0*0.398+0*0+0.097*0.097+0*0+0*0.222}{\sqrt{0.097^2}*\sqrt{0.398^2+0.097^2+0.222^2}}$
- $\cos(D1, Q) = \frac{0.00941}{0.0452} = 0.208$

Cosine similarities

	Beijing	Dish	Duck	Rabbit	Recipe	Cos(D,Q)
D1	0	0	0.097	0	0	0.208
D2	0.199	0.199	0.097	0	0	0.639
D3	0	0	0.097	0.199	0.111	0.256
D4	0	0	0	0.398	0.222	0.232
D5	0.398	0.398	0.097	0	0.222	0.760
Query	**.398**	**0**	**.097**	**0**	**.222**	1

Final ordered list

- **D5** = "Last week Li has shown you how to make the Sechuan duck. Today we'll be making Chinese dumplings (Jiaozi), a popular dish that I had a chance to try last summer in Beijing. There are many recipies for Jiaozi."
- **D2** = "Beijing Duck is mostly prized for the thin, crispy duck skin with authentic versions of the dish serving mostly the skin."
- **D3** = "Bugs' ascension to stardom also prompted the Warner animators to recast Daffy Duck as the rabbit's rival, intensely jealous and determined to steal back the spotlight while Bugs remained indifferent to the duck's jealousy, or used it to his advantage. This turned out to be the recipe for the success of the duo."
- **D4** = "6:25 PM 1/7/2007 blog entry: I found this great recipe for Rabbit Braised in Wine on cookingforengineers.com."
- **D1** = "If it walks like a duck and quacks like a duck, it must be a duck."

datasciencedojo
unleash the data scientist in you

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*
    - Same principle as collaborative ranking

Recommending items

- Advanced Methods
  - Classification algorithms
    - Predict either ratings or like/dislike
  - Information retrieval techniques
    - Well studied field, wide diversity of models

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

- Among many techniques
  - Which one is the best in a given application domain?
  - What are the success factors of different techniques?
  - Comparative analysis based on an optimality criterion?

Evaluating Recommendation

- Research questions are:
  - Is a RS efficient with respect to a specific criteria like accuracy, user satisfaction, response time, serendipity, online conversion, ramp-up efforts, ...
  - Do customers like/buy recommended items?
  - Do customers buy items they otherwise would have not?
  - Are they satisfied with a recommendation after purchase?