COMP30120

Recommender Systems Collaborative Filtering Part 2

Derek Greene

School of Computer Science and Informatics

Autumn 2015

Overview

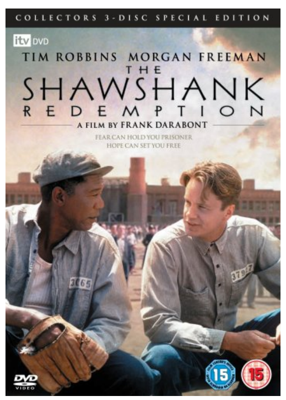
- Collaborative Filtering (CF)
- Evaluation
- Issues for Recommender Systems
    - Serendipity & Diversity
    - Explanation & Trust
    - Cold start problem
    - Attacks & Fraud
    - Recommenders with constraints
    - Group recommenders
- Short Tutorial

Reminder - Collaborative Filtering

Collaborative Filtering: Make predictions about the preferences of a user based on past activity of a community of other users.

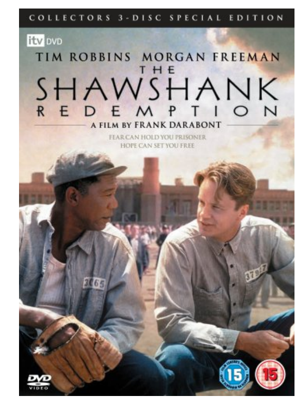
1. Predict Ratings

Ratings by Alice



★★★★★ ★★★★ ★★★

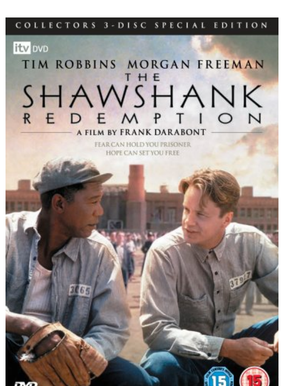
Ratings by Bob



★★★★★ ★★★★ ?????

2. Recommend Items

Bob likes…



Alice likes…

Bob may also like…

Explicit v Implicit Data Collection

- Explicit Data Collection
  - Actively ask users for explicit ratings for items.
  - Easy to interpret, as it directly represents a user's preferences.
  - Puts responsibility for data collection on to the user.
- Implicit Data Collection
  - Gather data directly based on user's activity - e.g. purchase history, click and browsing logs.
  - Easy to collect in large quantities, does not require extra effort from the user.
  - Can we reliably infer user preferences from implicit data? e.g. how do we interpret a partial play of a song in a music recommender?
- Both approaches can be combined - e.g. Apple iTunes.

Recommender System Evaluation

- Offline Evaluation
  - No actual users are involved in the evaluation.
  - An existing dataset is used which has been collected previously.
  - Advantage: Quick, cheap, can be easily repeated.
- Online Evaluation
  - Users interact with a running system in a “live experiment”, and receive actual recommendations.
  - Feedback from the users is collected by observing their online behaviour and/or explicitly collecting their feedback.
  - Advantage: Measures the true extent of customer satisfaction with the recommender.

Offline Evaluation in Collaborative Filtering

- Many ratings data sets are publicly available for benchmarking the performance of CF algorithms for rating prediction.
e.g http://grouplens.org/datasets/movielens/

- **Evaluation Methodology:**
  - Split the data into two subsets:
    1. The first subset acts as a *training set* that will be available for the algorithm to learn from.
    2. The second subset is the *test set*, with rating values that are not available to the algorithm.
  - Query the algorithm to make predictions for ratings on all the items in the test set.

- This process is repeated for many different splits of the data
i.e. *k*-fold cross validation.

Evaluation Measures

- Interested in error on unseen test set Q, not error on the training set.
- For each (u_f,i_g) in the test set, let r_{fg} = true rating, $\hat{r}_{fg}$ = predicted rating.
- Several popular approaches for evaluating rating predictions.

- Mean Absolute Error (MAE): Measures the average absolute deviation between a predicted rating and the user's true rating.

$$MAE = \frac{1}{|Q|} \sum_{(u_f, i_g) \in Q} |r_{fg} - \hat{r}_{fg}|$$

- Root Mean Square Error (RMSE): Related measure of difference between prediction and actual score. Punishes big mistakes more severely. Used in Netflix competition.

$$RMSE = \sqrt{\frac{1}{|Q|} \sum_{(u_f, i_g) \in Q} (r_{fg} - \hat{r}_{fg})^2}$$

➡ NB: Quantitative scores may not reflect the end user experience!

Example: RMSE Evaluation

	True Rating (Stars)	Predicted Rating	Difference	Squared Difference
Test Item 1	**2**	**2.2**	-0.2	0.04
Test Item 2	**5**	**4.1**	0.9	0.81
Test Item 3	**4**	**4.7**	-0.7	0.49
Test Item 4	**1**	**2.1**	-1.1	1.21
Test Item 5	**4**	**3.6**	0.4	0.16
Total				2.71

Using RMSE to evaluate accuracy of predicted ratings (1 to 5 stars) for a test set of 5 items.

$$RMSE = \sqrt{\frac{1}{|Q|} \sum_{(u_f, i_g) \in Q} (r_{fg} - \hat{r}_{fg})^2}$$

$$RMSE = \sqrt{2.71/5}$$
$$= 0.736$$

Top-K Evaluation Measures

- Top-*k* recommendations: Focus on the ability of the algorithm to successfully identify *relevant* items among its top ranked recommendations.
- For ratings-based systems, we need to convert ratings to binary relevant/irrelevant classes.
  e.g. In a 5-point rating system, (1-3) = *irrelevant*, (4-5) = *relevant*

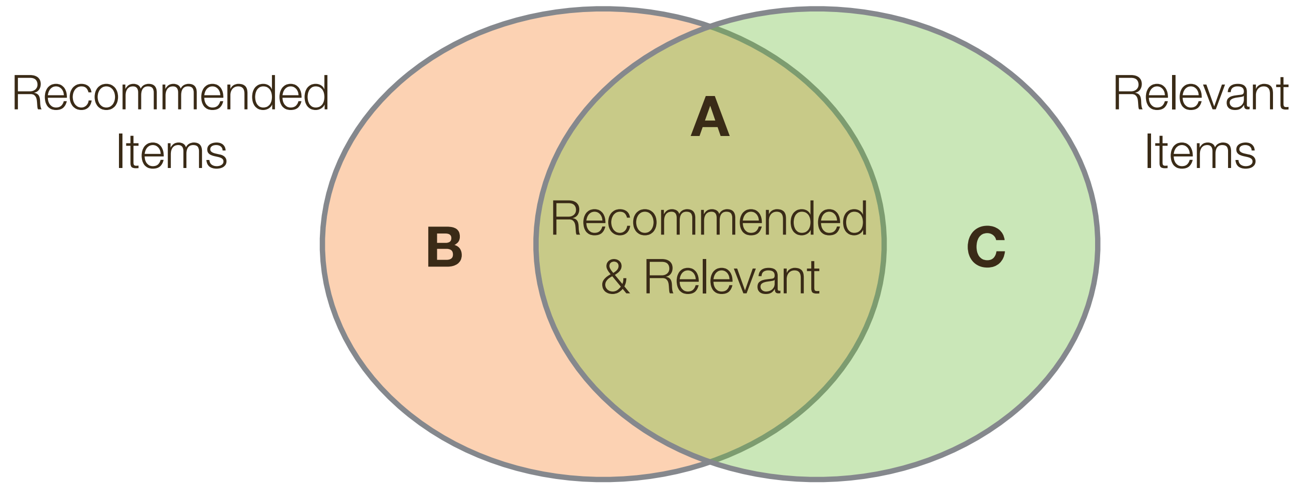
Rank	Recommended Item	Relevant?
1	Divergent	✗
2	Guardians of the Galaxy	✓
3	Gravity	✓
4	Thor	✗
5	Edge of Tomorrow	✓

Top 5: 3 relevant, 2 irrelevant

- We can then use accuracy metrics from classification and Information Retrieval to quantify the success of a recommendation task.
- Source of test data can be a held-out test subset (offline evaluation), or relevance judgements from users in a live trial (online evaluation).

Precision & Recall

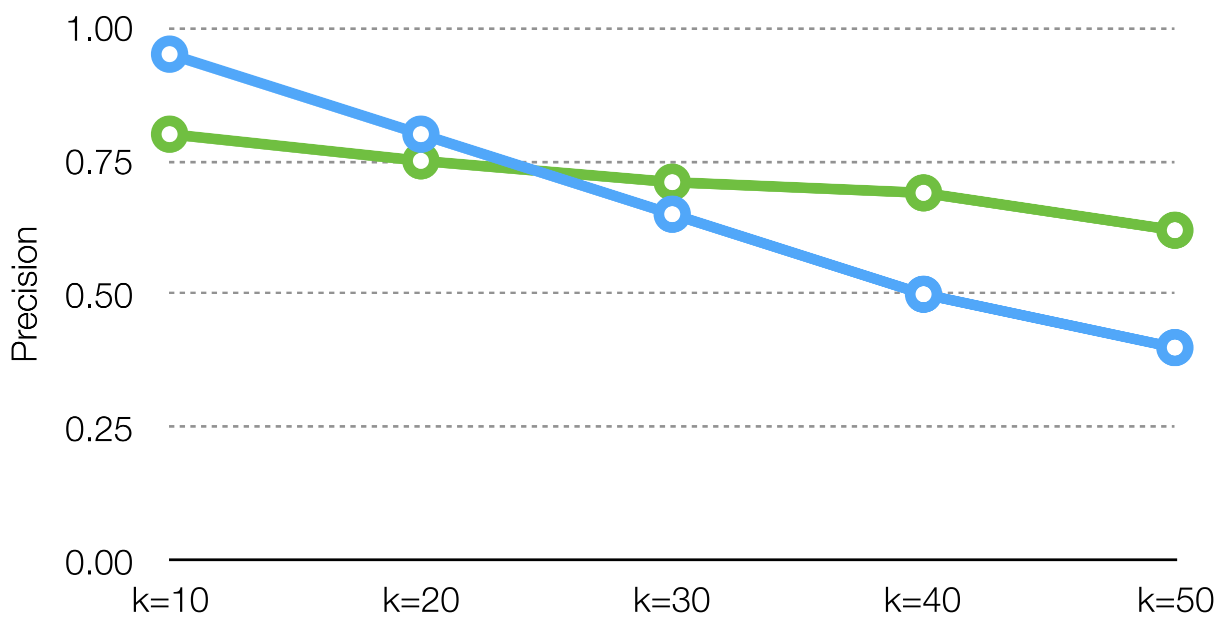
- Precision: proportion of recommended items that are relevant.
- Recall: proportion of relevant items that are recommended.



$$\text{Precision} = \frac{A}{A+B}$$

$$\text{Recall} = \frac{A}{A+C}$$

- Apply measures to the top *k* results from the recommender:
  - precision-at-*k*
  - recall-at-*k*



Serendipity & Diversity

- Serendipity: "the effect by which one accidentally discovers something fortunate, especially while looking for something entirely unrelated" - Wikipedia
- Often not helpful to recommend obvious items or lists of items that are too similar to one another.

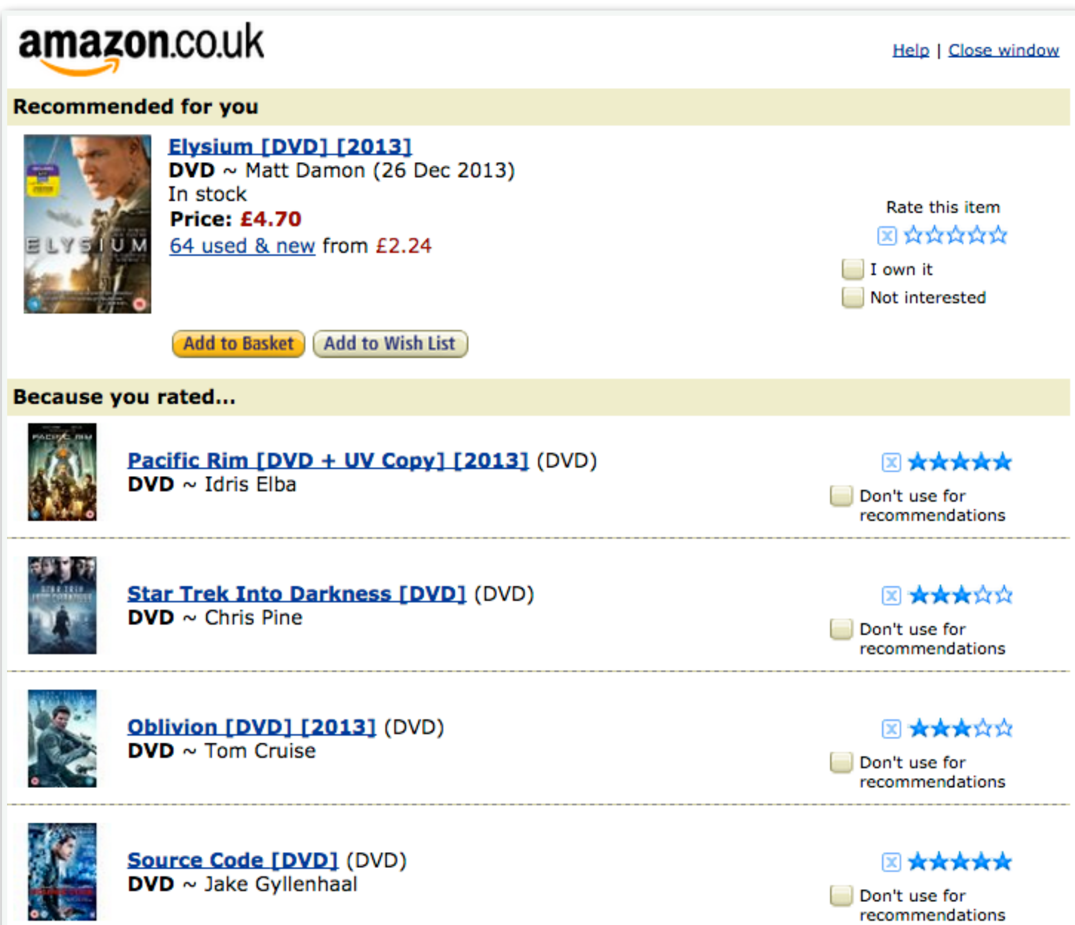
"If a user rates an item (e.g. rating an album by The Beatles), loading the user's recommendations with extremely similar items (i.e. all of the other albums by The Beatles) is often not helpful at all; the user has not been pointed towards new information, and is only inundated with recommendations towards content that is probably known already."
- Lathia, 2010.

➡ Alternative evaluation goal: Examine extent to which a collaborative filtering algorithm can generate diverse recommendations among its top *k* results.

Explanation & Trust

- A recommender is of little value if a user does not trust the system's recommendations.
- Trust can be gained by explaining how a system generates recommendations, and why it recommends items.

Explanations relative to other items



Social explanations

tripadvisor.com

Cold Start Problem

- Content-based recommenders try to find intrinsic similarities between items. So a user's past history is not really necessary.
- For collaborative filtering we face 2 problems:
  1. How to deal with “cold” users who have never rated any items?
  2. How to deal with “cold” items which have received no ratings?

	Book 1	Book 2	Book 3	Book 4	Book 5	Book 6	Book 7	Book 8	Book 9	Book 10
User 1				1	1					
User 2	1		1					1		
User 3		1			1		1			
User 4					1				1	
User 5	1	1					1			
User 6			1			1				
User 7									1	
User 8										

“Cold” item with no ratings

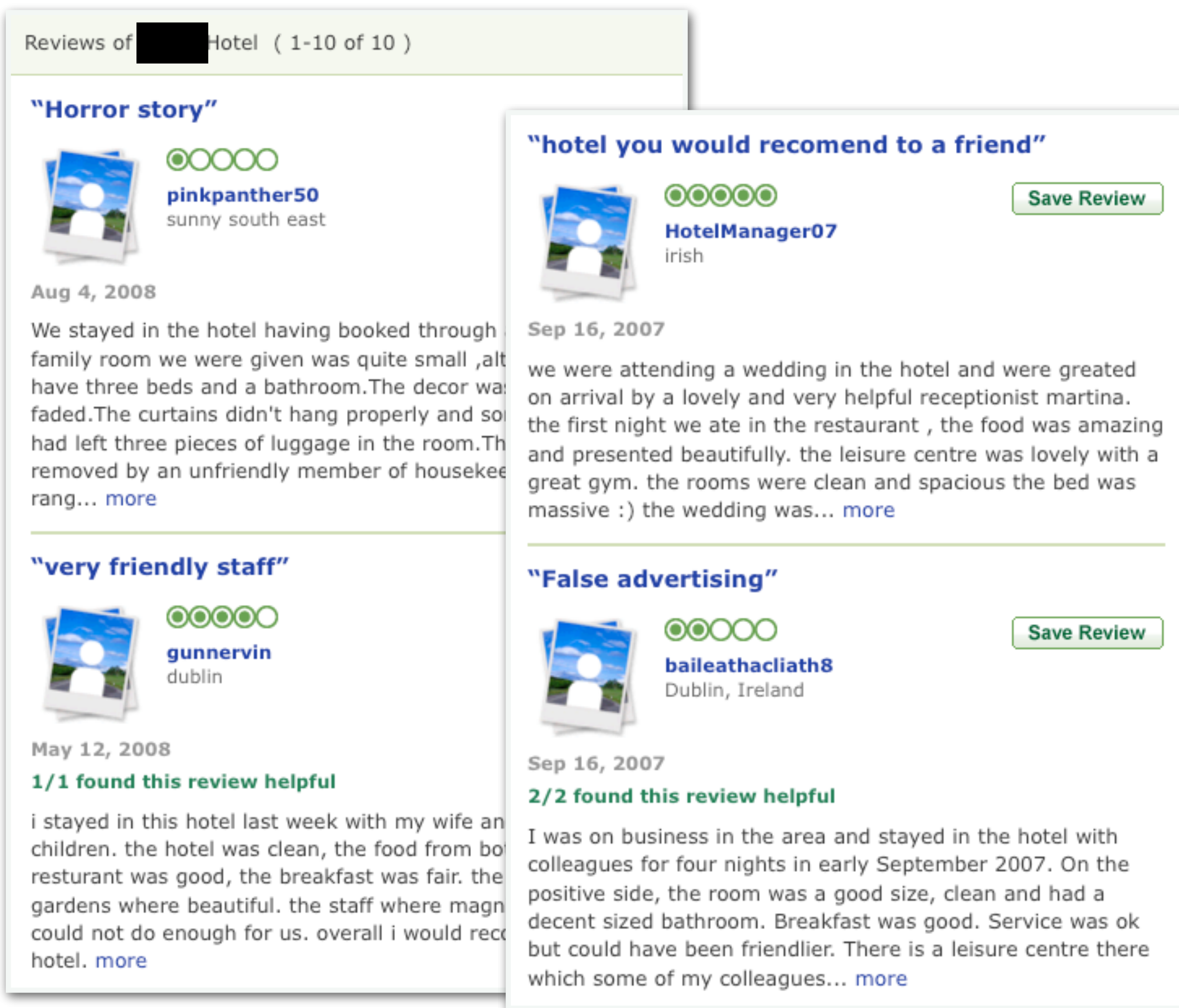
“Cold” user with no ratings

Cold Start Problem: Solutions

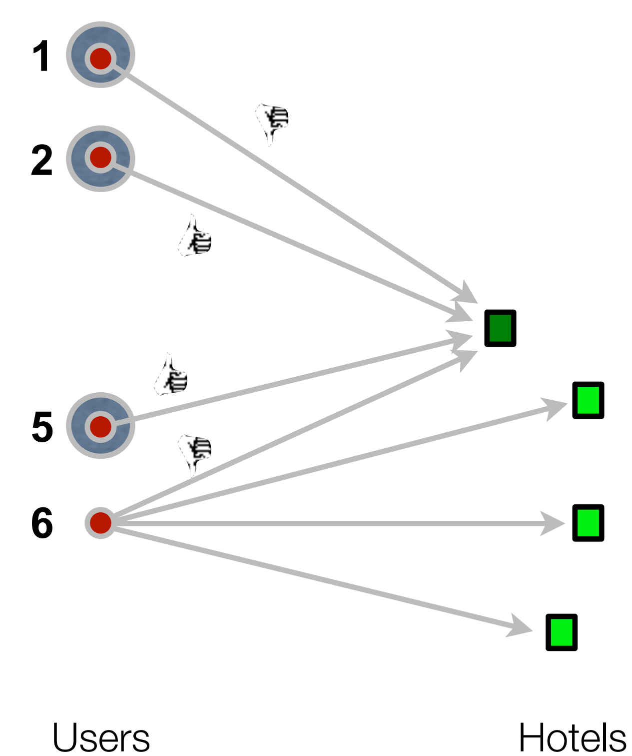
- Hybrid Recommenders: Combine CF with methods that do not suffer from cold start problem
- **Cold Item Problem**
  - Fall back on content-based recommendations for cold items.
  - Make inferences based on cold items by looking at historical data for other related items, sharing the same description or metadata.
- **Cold User Problem**
  - Naïve approach: Force new users to rate or select a set of items.
  - Recommend most popular items.
  - Turn to additional data about the users - e.g. demographic information, social data.

Attacks on Recommender Systems

- Shilling: Malicious users create fake profiles to distort the recommendation process.
  e.g. a vendor, motivated to promote ratings of their product, manually creates multiple user profiles that give their product 5★ ratings.



Examining historic behaviour of users can help to identify shills



Attacks on Recommender Systems

- Attacks on CF systems can be large-scale and coordinated.
- Profile injection: manipulate system by adding large number of fake profiles.
- Push attack: Aims to increase the popularity of an item.
  e.g. a vendor who wants to boost the reputation of their product.
- Nuke attack: Aims to decrease the popularity of an item.
  e.g. a vendor who wants to discredit a product from a competitor.

	Item1	Item2	Item3	Item4	Item5	Item6	Item7
User 1	4	3	4		3	4	4
User 2	5	5	1	4	1	3	4
User 3	1	5	2	5	4	2	1
User 4	5	1	5	3		5	2
User 5	3	5	4	4	1		2
User 6		5	5	4		2	3
User 7	1	2	3	2		2	4
Attacker 1	3	4	3	4	5	3	3
Attacker 2	2	5	3	4	5	4	3

Attack using 2 fake profiles to boost the average rating for target *Item 2*.

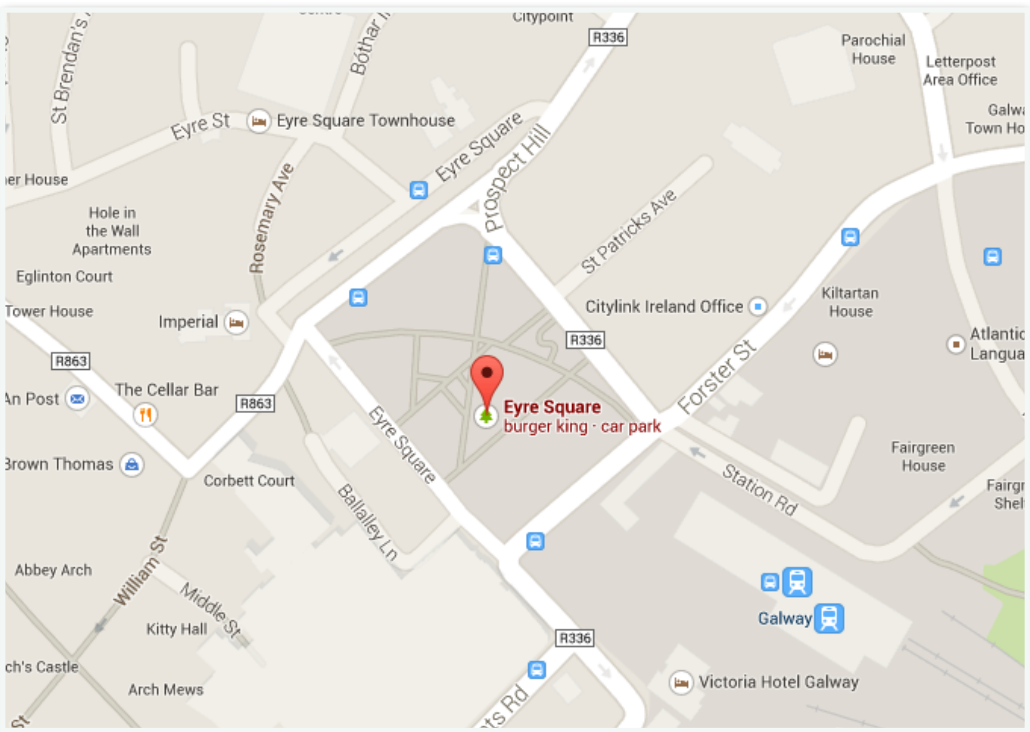
The item gets two new 5★ reviews. The other values in the profiles are “fillers”.

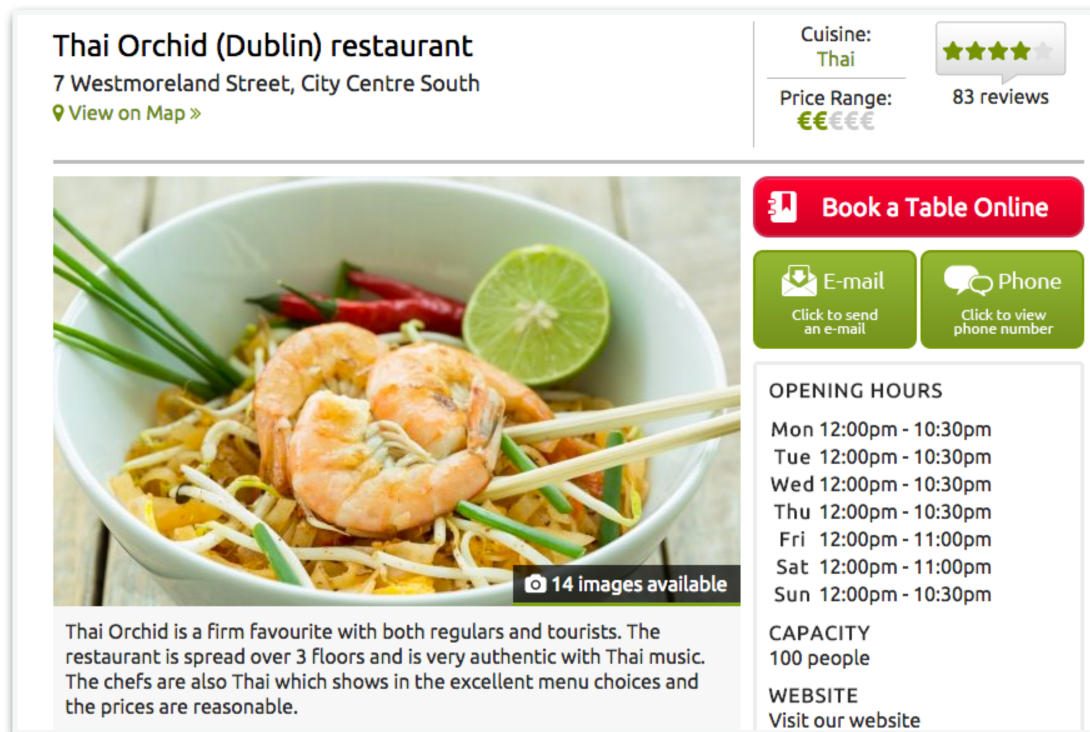
Example from Neil Hurley

Recommender Systems with Constraints

- The "best" item to recommend may not be the highest ranked item globally. May need to take into account the user's current context as well as historic preferences.
  - e.g. user's location, time of day, mode of transport.
- Particularly relevant for recommenders in mobile applications.

Task: Recommend a good Thai restaurant

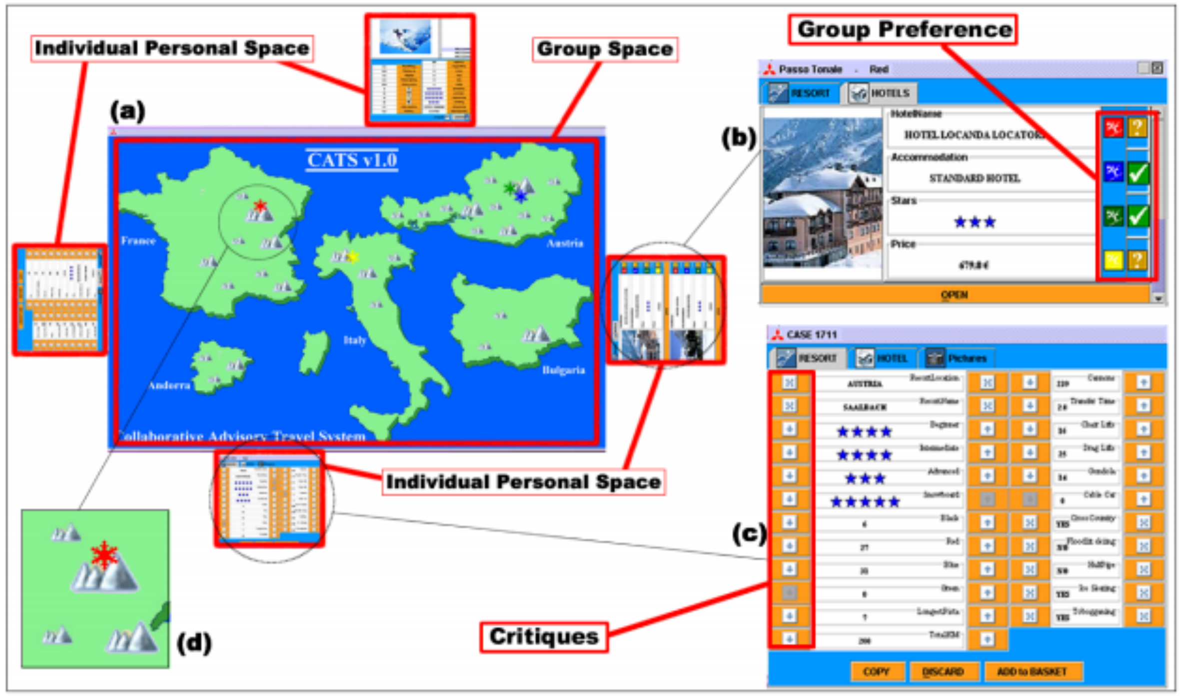




Group Recommender Systems

- Usually recommenders focus on personalisation for an individual.
- Some scenarios need to take into account several related users:
  - Multiple people sharing the same account.
    e.g. a family all using the same Netflix account.
  - Two or more people generating recommendations collaboratively.
    e.g. a group of friends planning a ski holiday together.

- Introduces new challenges for recommenders:
  - Capture individual activity.
  - Capture group interactions.
  - Combine preferences of different users, who may have competing interests.



CATS system. McCarthy et al (2006)

Tutorial Question 1

Q. The user-item matrix below shows the purchasing history of 5 users with respect to 9 different books in a user-based CF system.
Who will be *U3*'s nearest neighbour in the data?
Calculate similarities with the Jaccard Index.

	Book1	Book2	Book3	Book4	Book5	Book6	Book7	Book8	Book9
U1		1		1		1	1		
U2			1		1			1	
U3	1	1		1			1		1
U4	1	1				1		1	
U5			1	1	1				

Jaccard Index

$$sim(p,q) = \frac{|B_p \cap B_q|}{|B_p \cup B_q|}$$

$B_p = \text{Books purchased by } p$

$B_q = \text{Books purchased by } q$

Tutorial Question 1 - Solution

	Book1	Book2	Book3	Book4	Book5	Book6	Book7	Book8	Book9
U1		1		1		1	1		
U2			1		1			1	
U3	1	1		1			1		1
U4	1	1				1		1	
U5			1	1	1				

Target	Other	Intersection	Union	Jaccard
U3	U1	3	6	**0.50**
U3	U2	0	8	0.00
U3	U3	5	5	1.00
U3	U4	2	7	0.29
U3	U5	1	7	0.14

$$sim(p,q) = \frac{|B_p \cap B_q|}{|B_p \cup B_q|}$$

$$B_p = \text{Books purchased by } p$$
$$B_q = \text{Books purchased by } q$$

Based on past book purchases, U3's nearest neighbour is U1

Tutorial Question 2

Q. The user-item matrix below shows the purchasing history of 6 users for 10 different products in a user-based CF system.
Who will be *U1*'s nearest neighbour in the data?
Calculate similarities with the Jaccard Index.

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
U1		1		1		1		1		1
U2	1		1				1			
U3				1				1		
U4		1		1						
U5				1		1		1		1
U6	1	1	1							

Tutorial Question 2 - Solution

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
U1		1		1		1		1		1
U2	1		1				1			
U3				1				1		
U4		1		1						
U5				1		1		1		1
U6	1	1	1							

$$sim(p,q) = \frac{|B_p \cap B_q|}{|B_p \cup B_q|}$$

Target	Other	Intersection	Union	Jaccard
U1	U1	5	5	1.00
U1	U2	0	8	0.00
U1	U3	2	5	0.40
U1	U4	2	5	0.40
U1	U5	4	5	**0.80**
U1	U6	1	7	0.14

Based on past purchase history, U1's nearest neighbour is U5

Summary

- Part 1
  - User-based Collaborative Filtering
  - Item-based Collaborative Filtering
  - Content-based Recommendation
- Part 2
  - Evaluation
  - Issues for Recommender Systems
    - Serendipity & Diversity
    - Explanation & Trust
    - Cold start problem
    - Attacks
    - Constrained & Group recommenders

References

- J. Leskovec, A. Rajaraman, J. Ullman “Mining of Massive Datasets”. Cambridge Press. http://www.mmds.org
- J Wang, AP De Vries, MJT Reinders. "Unifying user-based and item-based collaborative filtering approaches by similarity fusion". Proc. SIGIR 2006.
- B. Sarwar, G. Karypis, J. Konstan, J. Riedl. “Item-based Collaborative Filtering Recommendation Algorithms”, Proc. WWW 2010.
- N. Lathia. “Evaluating Collaborative Filtering Over Time”. PhD Thesis, University College London, 2010
- K. McCarthy et al. “Group Recommender Systems: A Critiquing Based Approach”, Proc. IUI, 2006.