Recommender Systems & Collective Intelligence

COMP47580

Dr. Michael O'Mahony

michael.omahony@ucd.ie

Overview

- Previously:
  - Non-personalised recommendation (ratings, making predictions, aggregated opinion, ad-hoc ranking metrics, product association)
- Today's topic:
  - Content-based recommender systems
  - Supports both personalised and non-pesonalised recommendation
  - Distinguish between:
    - Traditional content-based approaches (e.g. recommendating documents)
    - Case-based approaches (e.g. recommending items described by features)
  - Diversity enhancing approaches
  - Evaluation methodology and metrics

Considerations

- Personalised recommenders rely on learning users' interests
- Users' personal information is needed:
  - Privacy and trustworthiness
  - Who knows what about me?
- Recommendation output:
  - Biases built-in by operator – business priorities
  - Vulnerability to manipulation (robustness)
  - Transparency of recommenders, explanations
- Personalisation:
  - Ephemeral – matching current activity
  - Persistent – matching long-term interests

Recommender Systems – Benefits

- Turning Web Browsers into Buyers:
  - Due to the sheer volume of information online, people often spend only a small amount of time browsing individual web sites, and often leave without making purchases.
  - By ensuring good web site design and by helping customers to find suitable products, businesses can attract the attention of customers and increase sales and profits…

- Cross–Selling:
  - Recommender systems can promote sales by suggesting further products to customers that are likely to appeal to them. For example, during the check-out process, items that are similar/related to those already selected by the customer may be recommended…

Recommender Systems – Benefits

- Customer Loyalty:
  - Recommender systems can help to secure customer loyalty if customers are satisfied with the recommendations that are made to them.
  - Also, as a recommender system learns more about a customer’s preferences, the ability of the system to make better recommendations improves.
  - A relationship builds up over time between customers and specific sites, where the customer has invested time and effort in informing a recommender system about his or her own preferences, and in turn, the system reciprocates by continuously improving the quality of recommendations.
  - Following such investments, individual customers become less likely to switch allegiances to competitors, even if other companies provide a similar service.

Content-based Recommendation

- Items are recommended which are are similar in content to previously selected items
- Recommendations are based on a description of the content of items as opposed to what people actually thought about items
- A *more-like-this* form of recommendation – e.g. recommend items that are similar to a target item (non-personalised) or to those that the user has consumed (purchased/visited/read etc.) in the past (personalised)
- Distinguish between traditional content-based (unstructured) and case-based (structured) approaches
- Key concepts: item *representation* and item-item *similarity*

Content-based Recomendation

- Traditional content-based recommendation:
  - Recommending news articles, web pages etc.
  - Represent items using terms contained in documents (unstructured representation)
  - Use techniques from Information Retrieval (IR)
  - Generating recommendations:
    - Non-personalised – rank recommendation candidates by similarity to the target item
    - Personalised – rank recommendation candidates by similarity to the target user's profile (e.g. items previously purchased/visited/ watched etc. by the target user)

Content-based Recommendation

Typical scenario – recommending documents, e.g. news articles…

Independent.ie

Miss Panti: RTE pay out €85,000 in 'homophobe' row
Six people received compensation from the company following accusation

Ronan O'Gara and wife Jessica expecting another child
The Irish rugby great is to be father once again

Garda probe into crush that hurt girl at Copper Face Jacks
Up to 1,500 young people descended on club

FG's 'dirty dozen' gang up on Hogan at party meeting
Phil Hogan was confronted by angry backbenchers over Irish Water

'I just felt that I wasn't asking too much of him'
Kimmage resigned as writer of Brian O'Driscoll's autobiography

Gardai launch investigation after skull found on beach
Garda investigation after skull was found washed up on a beach in Galway.

Content-based Recomendation

- Items are represented by documents
- Use techniques from Information Retrieval (IR):
  - Vector Space Model (VSM) – represent documents as vectors in the multi-dimensional term space
- Term-document matrix:
  - Entries capture how frequently each term occurs in each document
  - For example, term t_3 occurs 7 times in document d_1

	d_1	d_2	d_3	...	...	...	d_m
t_1	2	0	10				1
t_2	3	1	4				4
t_3	7	3	0				0
t_4	0	0	9				2
...							
...							
t_n	...						...

- Term weighting – different approaches to weight the importance of terms in documents

Term Frequency

- Term frequency (TF) weighting:
  - Entries in the term-document matrix capture how frequently terms appear in documents

Term Frequency (TF)

	Doc 1	Doc 2	Doc 3	Doc 4
car	4	3	8	7
auto	3	7	10	0
insurance	0	14	0	5
best	2	0	0	0
bonus	5	0	7	8

Normalised Term Frequency

- Normalised term frequency weighting:
  - Expect longer documents to contain more occurrences of terms => normalise entries e.g. by the maximum TF in a document
  - Document-wise (i.e. column-wise) normalisation

Term Frequency (TF)

	Doc 1	Doc 2	Doc 3	Doc 4
car	4	3	8	7
auto	3	7	10	0
insurance	0	14	0	5
best	2	0	0	0
bonus	5	0	7	8

max(TF)	5	14	10	8

Normalised TF

	Doc 1	Doc 2	Doc 3	Doc 4
car	0.80	0.21	0.80	0.88
auto	0.60	0.50	1.00	0
insurance	0	1.00	0	0.63
best	0.40	0	0	0
bonus	1.00	0	0.70	1.00

$$\mathrm{nTF}(t_i, d_j) = \frac{f(t_i, d_j)}{\max\{f(w, d_j) : w \in d_j\}}$$

Inverse Document Frequency (IDF)

- Terms that appear in many documents do not help to discriminate between documents – idea is to reduce the weights of such terms
- Term-wise (i.e. row-wise) operation

Term Frequency (TF)

	Doc 1	Doc 2	Doc 3	Doc 4
car	4	3	8	7
auto	3	7	10	0
insurance	0	14	0	5
best	2	0	0	0
bonus	5	0	7	8

IDF Term Weights

IDF(car) = log(4/4) = 0

IDF(auto) = log(4/3) = 0.12

IDF(insurance) = log(4/2) = 0.30

IDF(best) = log(4/1) = 0.60

IDF(bonus) = log(4/3) = 0.12

$$\mathrm{IDF}(t_i, D) = \log\left(\frac{|D|}{|\{d \in D : t_i \in d\}|}\right)$$

Normalised Term Frequency - Inverse Document Frequency

- Apply nTF-IDF weighting to term-document matrix:

Normalised TF

	Doc 1	Doc 2	Doc 3	Doc 4
car	0.80	0.21	0.80	0.88
auto	0.60	0.50	1.00	0
insurance	0	1.00	0	0.63
best	0.40	0	0	0
bonus	1.00	0	0.70	1.00

Normalised TF-IDF

	Doc 1	Doc 2	Doc 3	Doc 4
car	0	0	0	0
auto	0.07	0.06	0.12	0
insurance	0	0.30	0	0.19
best	0.24	0	0	0
bonus	0.12	0	0.08	0.12

$$\text{nTF-IDF}(t_i, d_j, D) = \frac{f(t_i, d_j)}{\max\{f(w, d_j) : w \in d_j\}} \times \log\left(\frac{|D|}{|\{d \in D : t_i \in d\}|}\right)$$

Binary Weighting

- Binary term weighting:
  - Entries capture whether or not terms appear in documents {0, 1}
  - Often used for short-form documents (e.g. tweets)

Term Frequency (TF)

	Doc 1	Doc 2	Doc 3	Doc 4
car	4	3	8	7
auto	3	7	10	0
insurance	0	14	0	5
best	2	0	0	0
bonus	5	0	7	8

Binary

	Doc 1	Doc 2	Doc 3	Doc 4
car	1	1	1	1
auto	1	1	1	0
insurance	0	1	0	1
best	1	0	0	0
bonus	1	0	1	1

Useful Resources

Term Weighting:

- https://nlp.stanford.edu/IR-book/html/htmledition/inverse-document-frequency-1.html
- https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html
- Okapi BM25: https://nlp.stanford.edu/IR-book/html/htmledition/okapi-bm25-a-non-binary-model-1.html

Apache Lucene:

- High-performance, full-featured text search engine library written in Java (Python version available)
- http://lucene.apache.org/

Term Stemming

- Term stemming:
  - For grammatical reasons, documents use different forms of words, such as *organise*, *organises*, and *organising*.
  - Also, there are families of related words with similar meanings, such as *democracy*, *democratic*, and *democratisation*.
  - Can consider these terms as being the same for matching purposes.

- Term stemming is applied to all terms prior to construction of the term-document matrix:
  - Example: *computing*, *computer*, *compute* => *comput*
  - Can lead to incorrect matches between semantically unrelated terms…
  - Does not deal with synonyms (e.g. *car*, *automobile*…). See WordNet, lexical database for the English language (https://wordnet.princeton.edu/)

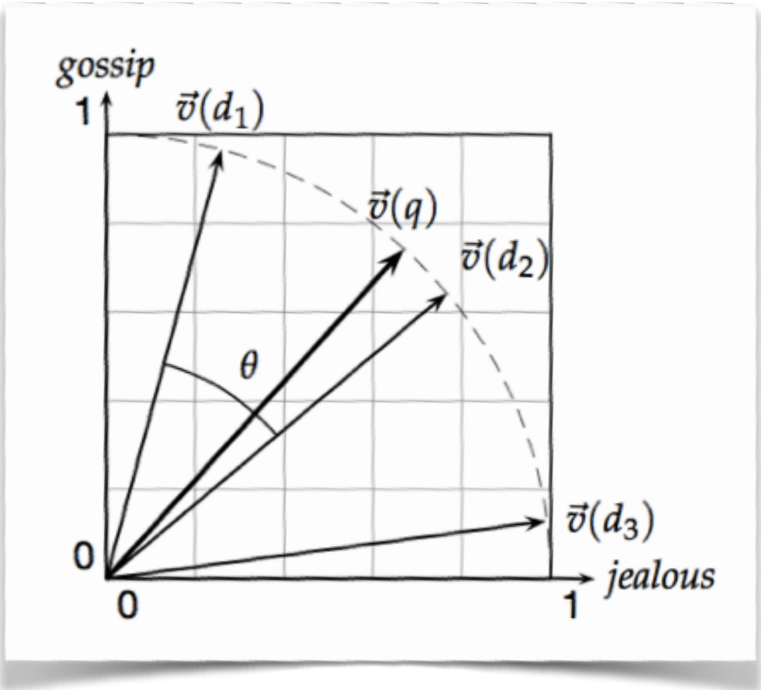
Stop Words

- Stop words:
  - Common words may be of little value when discriminating documents.
  - For example, some frequently occurring words in the English language are: *a, an, and, are, as, at, be, by, for, from*…

- Determining a stop word list:
  - The general aproach is to sort all terms by *collection frequency* – i.e. the total number of occurrences of a term in a document collection.
  - The most frequent terms (often hand-filtered for their semantic content relative to the collection domain) are added to the stop word list.
  - Stop words are omitted from the term-document matrix.

Document-document Similarity

- Since documents are represented as term vectors in a multi-dimensional vector-space, document-document similarity can be computed by the cosine of the angle between their two vectors.



$$sim(d_i, d_j) = \frac{\vec{V}(d_i) \cdot \vec{V}(d_j)}{|\vec{V}(d_i)|\,|\vec{V}(d_j)|}$$

Document-document Similarity Cosine Example

Normalised TF-IDF

	Doc 1	Doc 2	Doc 3	Doc 4
car	0	0	0	0
auto	0.07	0.06	0.12	0
insurance	0	0.30	0	0.19
best	0.24	0	0	0
bonus	0.12	0	0.08	0.12

$$sim(d_i, d_j) = \frac{\vec{V}(d_i) \cdot \vec{V}(d_j)}{|\vec{V}(d_i)|\,|\vec{V}(d_j)|}$$

$$sim(d_1, d_2) = \frac{0 \times 0 + 0.07 \times 0.06 + 0 \times 0.30 + 0.24 \times 0 + 0.12 \times 0}{\sqrt{(0^2 + 0.07^2 + 0^2 + 0.24^2 + 0.12^2)}\,\sqrt{(0^2 + 0.06^2 + 0.30^2 + 0^2 + 0^2)}}$$

Item-item Similarity Based on Ratings

Consider the following scenario in which users' ratings for items are available − construct a *user-item ratings matrix*

	Item 0	**Item 1**	**Item 2**	…	**Item *n***
User 0	9	8			
User 1		4	6		8
User 2			7		
…					
…					
User *m*		6	8		4

- Similarity is calculated by the cosine of the angle between the rating vectors:
  - Two items are considered similar if both have similar ratings patterns.
  - When calculating cosine using this approach, treat missing ratings as zeros and apply equation as shown on previous slide.

Item-item Similarity Based on Ratings

Calculating similarity between items 1 and 2

Consider the following scenario in which users' ratings for items are available – construct a *user-item ratings matrix*

	Item 0	Item 1	Item 2	…	Item *n*
User 0	9	8	0		
User 1		4	6		8
User 2		0	7		
…					
…					
User *m*		6	8		4

- Similarity is calculated by the cosine of the angle between the rating vectors:
  - Two items are considered similar if both have similar ratings patterns.
  - When calculating cosine using this approach, treat missing ratings as zeros and apply equation as shown on previous slide.

Making Recommendations

- Non-personalised recommendations:
  - Rank recommendation candidates by similarity (i.e. using cosine similarity) to the target item
- Personalised recommendations:
  - Rank recommendation candidates by similarity to the target user's profile
  - Various approaches depending on how the profile is constructed

User Profiling

- Users could create their own profiles:
  - Tedious process… But facilitating users to edit their profiles can be useful
- Infer profile from user behaviour (implicit):
  - For example, based on reads, clicks, purchases, count the number of times the user chooses (or does not choose) items with certain keywords
- Infer profile from user ratings (explicit):
  - 5-star, binary rating scales etc.
- How to map from item preference to attribute preferences?
- Combination of implicit/explicit data
- Other approaches:
  - NLP techniques (e.g. POS tagging)
  - Document clustering
  - Topic modelling (e.g. LDA) – the user likes documents related to *20th century Irish politics…*

Case-based Recomendation

- A particular style of content-based recommendation:
  - Items are represented in a more structured manner using a well-defined set of features and feature values
  - Allows for sophisticated and fine-grained judgements about the similarity between items

- A powerful approach to recommendation:
  - Facilitates the search and navigation of complex information spaces (*conversational recommendation* – more in later lectures)
  - Provides for flexible user-feedback options
  - Well-suited to e-commerce applications

- Based on ideas from the area of Case-based Reasoning (CBR)

Example Cases

Epson Expression Home XP-235 All-in-One Inkjet Printer

£44.00 & **FREE Delivery** in the UK. Details | In stock. Dispatched from and sold by Amazon. Gift-wrap available.

	 Epson Expression Home XP-235 All-in-One Inkjet Printer	 Epson Home XP-335 Expression All-in-One Inkjet Printer
Customer Rating	(74)	(89)
Price	**£44.00**	**£44.99**
Delivery	FREE Delivery	FREE Delivery
Sold by	Amazon.co.uk	Amazon.co.uk
Connectivity Technology	WiFi	USB 2.0, Wireless LAN
Resolution	1200 Dots Per Inch	1200 Dots Per Inch
Ink Colour	Multicoloured	Multicoloured
Dimensions	14.5 cm x 30 cm x 39 cm	14.5 cm x 39 cm x 30 cm
Item Weight	Information not provided	4.2 kg
Maximum Printspeed Black White	26 ppm	33 ppm
Model Year	Information not provided	2015
Scanner Type	Flatbed	Flatbed
	Add to Basket	Add to Basket

Case-based Reasoning

- Case-based recommender systems:
  - Origin in Case-based Reasoning (CBR) systems…
- CBR systems:
  - Used for problem solving & classification tasks
  - Rely on concrete experiences (**cases**) to solve problems
  - Maintain a **casebase** (database) of past problem solving experiences
  - Differ from traditional problem solving techniques, e.g. rules-based systems
- Cases are comprised of two parts:
  - **Specification/problem part**: features that describe a problem
  - **Solution part**: solution to specified problem

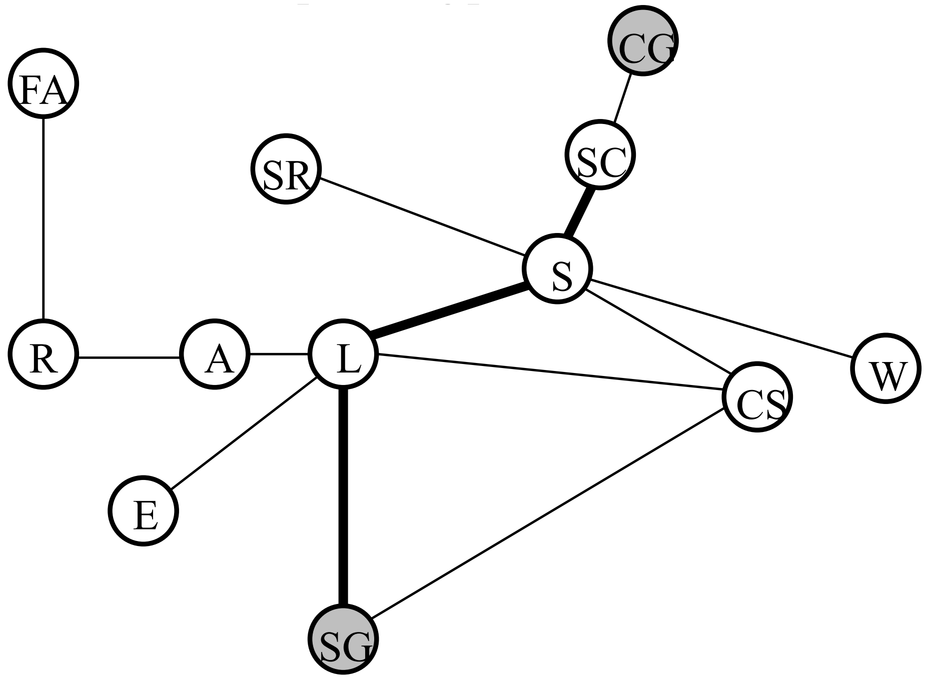
Case-based Reasoning

- Underlying assumptions:
  - The world is a **regular** place and similar problems tend to have similar solutions.
  - The world is a **repetitive** place and similar problems tend to recur.

- “Reasoning as remembering”:
  - Instead of solving problems from scratch using first-principles methods, the solutions to previous similar problems are recalled and reused.

Case-based Reasoning

- "Reasoning as remembering":
  - How can we reuse knowledge and past experiences to solve new problems – rather than start again from the beginning?



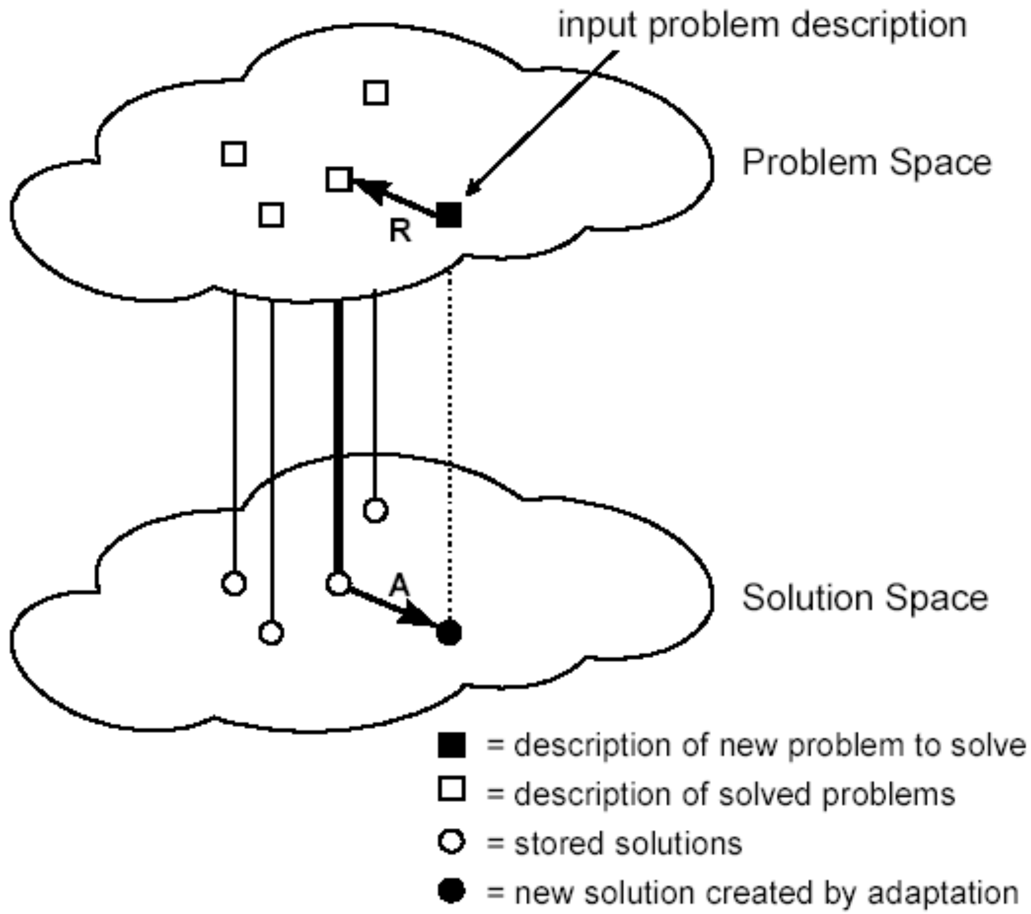
When trying to get from SG to CG would the fact that just yesterday you travelled from SG to SC affect your planning at all? It should!!!

CBR Cycle

- Solving new problems - CBR Cycle:
  - RETRIEVE: The target problem description is used to retrieve a case(s) from the casebase.
  - REUSE: The solutions of the retreived case(s) are reused (and possibly adapted) to develop a solution for the target problem.
  - REVISE: The new solution is tested for success (reviewed). A human expert may revise it if necessary.
  - RETAIN: The target problem along with its solution is then added to the casebase (learning).

Case-Based Reasoning

What happens when
a new problem is presented
to the system…



CBR Example…

Property Valuation:

Type:	Bungalow
Location:	Co. Cork
Bedrooms:	4
Rcpt Rooms:	2
Grounds:	1/3 Acre
Age:	New
Condition:	Excellent
PRICE:	**?**

- Task
  - Predict the selling/rental cost of residential properties.
- Challenges
  - Changing prediction rules due to market conditions.
- Solution
  - Maintain and reuse a case-base of recent properties as a basis for prediction.

CBR Example…

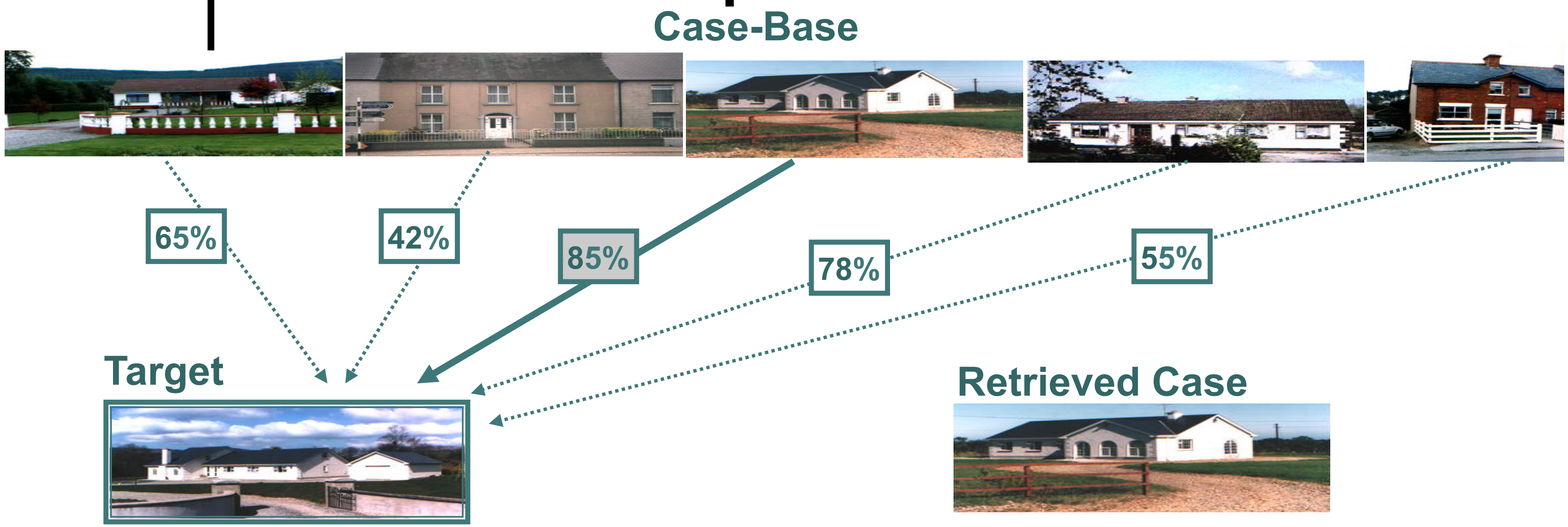
- Case Similarity
  - Based on a feature-wise comparison of cases…
  - Are all features equally important as predictors of price?

Target Problem

Candidate Case

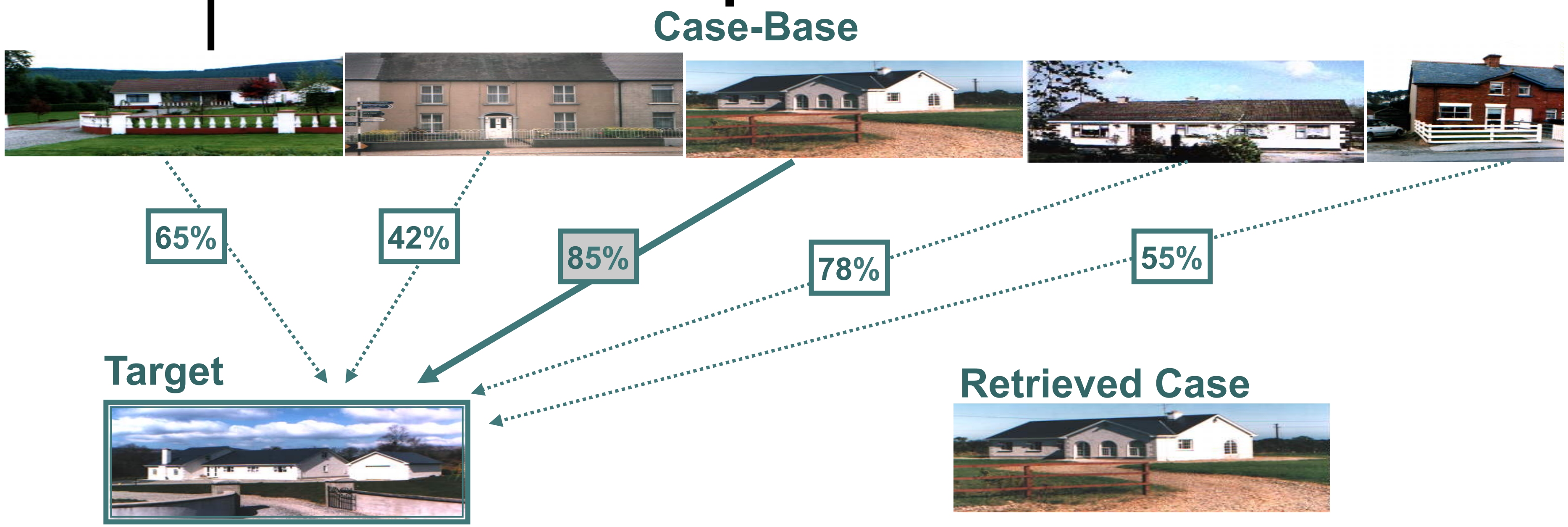
Type:	Bungalow	⟷	Type:	Bungalow
Location:	Co.Cork	⟷	Location:	Co.Cork
Bedrooms:	**4**	⟷	**Bedrooms:**	**3**
Rcpt Rooms:	2	⟷	Rcpt Rooms:	2
Grounds:	**1/3 Acre**	⟷	**Grounds:**	**1/4 Acre**
Age:	New	⟷	Age:	New
Condition:	Excellent	⟷	Condition:	Excellent
			Price:	**€ 270,000**

CBR Example



Type:	Bungalow		Type:	Bungalow
Location:	Co. Cork		Location:	Co. Cork
Bedrooms:	4	+£20k	Bedrooms:	3
Rcpt Rms:	2		Rcpt Rms:	2
Grounds:	1/3 Acre	+£15k	Grounds:	1/4 Acre
Age:	New		Age:	New
Condition:	Excellent		Condition:	Excellent
Price:			Price:	£270k

CBR Example



Type:	Bungalow		Type:	Bungalow
Location:	Co. Cork		Location:	Co. Cork
Bedrooms:	4	+£20k	Bedrooms:	3
Rcpt Rms:	2		Rcpt Rms:	2
Grounds:	1/3 Acre	+£15k	Grounds:	1/4 Acre
Age:	New		Age:	New
Condition:	Excellent		Condition:	Excellent
Price:	£305k		Price:	£270k

CBR Example – Call Centres

- Problem – how to better handle technical support:
  - Customers demand better tech support (costly)
  - Staff must be knowledgeable – training is expensive, high staff turnover
  - Tech support often deal with previously unencountered problems

- Automated solution?
  - A KBS cannot deal with previously unencountered problems
  - People describe problems differently – noisy, incomplete information
  - KBS: need continual maintenance to keep up with new products / problems

CBR Example – Call Centres

- Introduce a CBR System
  - Can deal with noisy, incomplete problem discriptions – is less “brittle” compared to KBS
  - Acquires new problems & their solutions (learning) – much easier maintenance compared to a KBS
- Deployment experience:
  - Increased problem resolution from 50% to 95%
  - Less than 2 minutes on average to solve a problem
- CBR tools are used in call centres and included in software products:
  - People can diagnose many problems themselves before calling tech support…

Case-based Recommendation

- Borrow core concepts of retrieval and similarity in CBR:
  - Items represented as cases
  - Retrieve cases most similar to user's profile or target item
- Key differences from traditional content-based systems:
  - **Case representation**: manner in which items are represented
  - **Similarity assessment**: how similarity is computed between items
- Well suited to e-commerce domains where detailed feature-based product descriptions are often available

Case Representation

- Cases (items) are representated using a set of well-defined features and feature values rather than free-form text

- Example…

Epson Expression Home XP-235 All-in-One Inkjet Printer

£44.00 & **FREE Delivery** in the UK. Details | In stock. Dispatched from and sold by Amazon. Gift-wrap available.

	 Epson Expression Home XP-235 All-in-One Inkjet Printer	 Epson Home XP-335 Expression All-in-One Inkjet Printer
Customer Rating	(74)	(89)
Price	**£44.00**	**£44.99**
Delivery	FREE Delivery	FREE Delivery
Sold by	Amazon.co.uk	Amazon.co.uk
Connectivity Technology	WiFi	USB 2.0, Wireless LAN
Resolution	1200 Dots Per Inch	1200 Dots Per Inch
Ink Colour	Multicoloured	Multicoloured
Dimensions	14.5 cm x 30 cm x 39 cm	14.5 cm x 39 cm x 30 cm
Item Weight	Information not provided	4.2 kg
Maximum Printspeed Black White	26 ppm	33 ppm
Model Year	Information not provided	2015
Scanner Type	Flatbed	Flatbed

Add to Basket | Add to Basket

Similarity Assessment

- Determines which cases to retrieve in response to user profile or target item
- Content-based recommenders (documents): VSM, cosine similarity
- Case-based recommenders:
  - Structured case representations => more sophisticated similarity assessment
  - Case-level similarity – based on explicit mappings between case features:

$$\mathrm{Sim}(T, C) = \frac{\sum_{i=1}^{n} w_i \times \mathrm{Sim}(v_{C,i}, v_{T,i})}{\sum_{i=1}^{n} w_i}$$

- Sim(T, C) is the similarity between the target case T and candidate case C
- $v_{C,i}$ is the value of feature i in case C
- $\mathrm{Sim}(v_{C,i}, v_{T,i})$ is a similarity function for case feature i
- Weight w_i encodes the relative importance of feature i

Numeric Feature Similarity

Different approaches are possible:

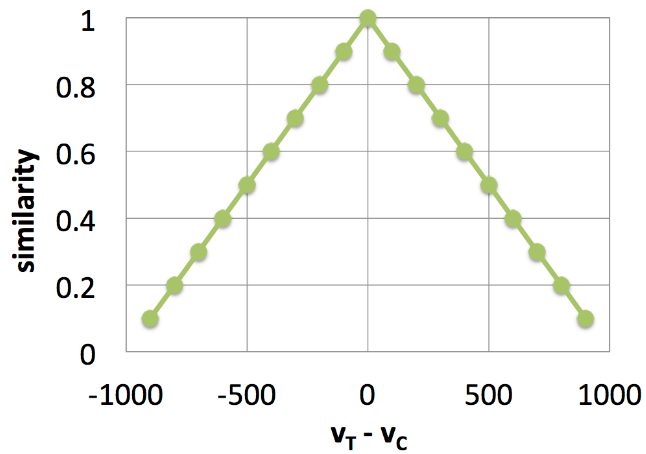
- Here consider symmetric and asymmetric similarity metrics

Symmetric similarity metrics:

- Consider feature *size*
- Maximum similarity achieved when feature *size* of candidate matches target case
- Symmetric: assume no user bias in favour of either higher/lower *sized* candidate cases

$$\mathrm{Sim}(v_C, v_T) = 1 - \frac{|v_T - v_C|}{v_T}$$

	v_C	v_T	$v_T - v_C$
candidate case feature value is lower	100	1000	900
	200	1000	800
	300	1000	700
	400	1000	600
	500	1000	500
	600	1000	400
	700	1000	300
	800	1000	200
	900	1000	100
	1000	1000	0
candidate case feature value is higher	1100	1000	-100
	1200	1000	-200
	1300	1000	-300
	1400	1000	-400
	1500	1000	-500
	1600	1000	-600
	1700	1000	-700
	1800	1000	-800
	1900	1000	-900

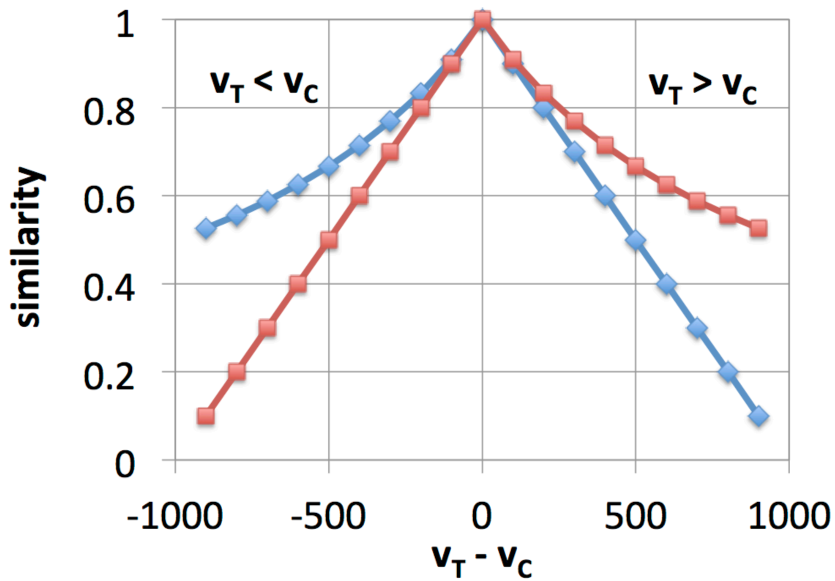


Numeric Feature Similarity

Asymmetric similarity metrics:

- Consider feature *memory* – assume user is interested in candidates with memory (v_C) **higher** than target memory (v_T)

- Consider feature *price* – assume user favours candidates with price (v_C) **lower** than target price (v_T)



$$\mathrm{Sim\_1}(v_C, v_T) = 1 - \frac{|v_T - v_C|}{\max(v_T, v_C)}$$

$$\mathrm{Sim\_2}(v_C, v_T) = 1 - \frac{|v_T - v_C|}{v_T + \max(0, (v_T - v_C))}$$

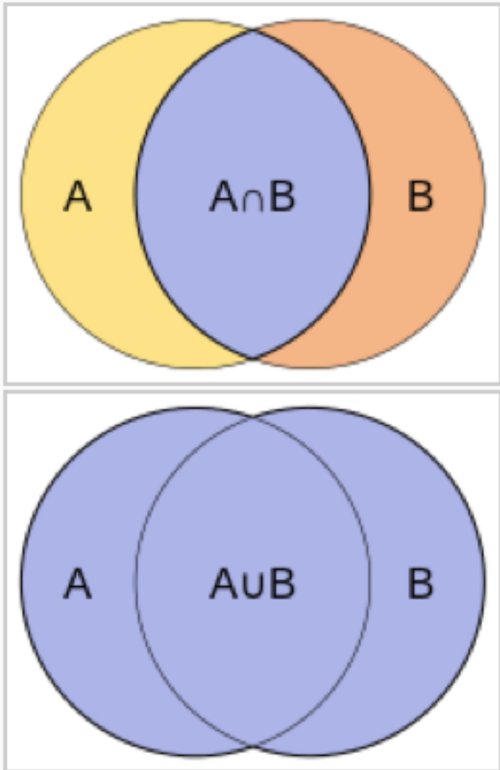
Use **Sim_1** (**Sim_2**) if **higher** (**lower**) candidate feature value is desired

Non-numeric Feature Similarity

- Consider features such as movie genres, actors…
- Feature values represented as sets
- Example – movie genres:
  - *Star Trek* = {sci-fi, action, adventure, thriller}
  - *2001 A Space Odessey* = {sci-fi, avdenture, mystery}
- Feature similarity functions:
  - Overlap coefficient, Jaccard index, Dice coefficient…

$$Jaccard(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

$$overlap(A,B) = \frac{|A \cap B|}{\min(|A|,|B|)}$$

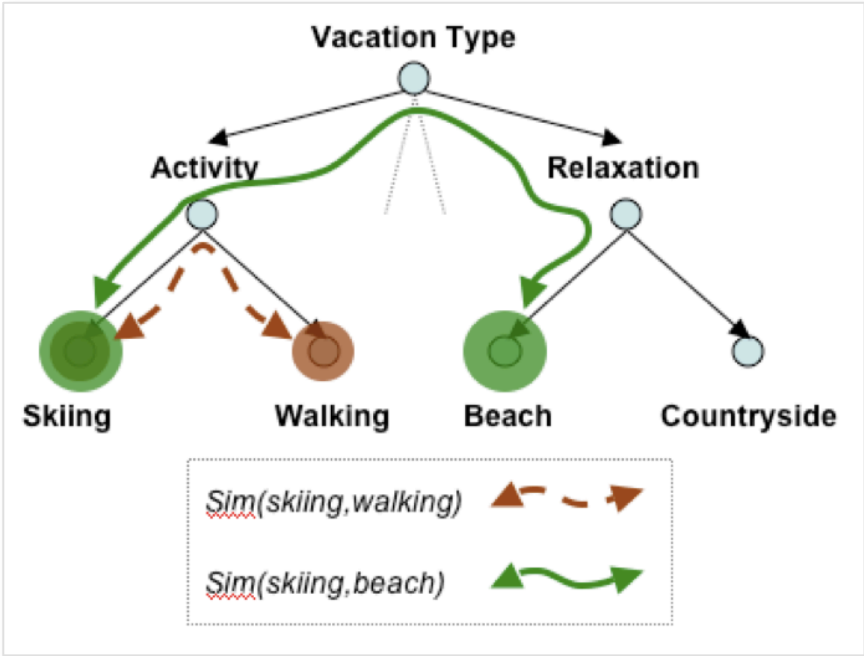


Intersection and union of two sets A and B

Non-numeric Feature Similarity

- Require additional domain knowledge to capture similarity of certain non-numeric features

- Consider a vacation recommender:
  - What is the similarity between different vacation types (important!)?
  - E.g. is a *skiing* holiday more similar to a *walking* holiday than to a *beach* holiday?
  - Create ontology of vacation types: different feature values represented as nodes
  - Define similarity as inverse function of distance between two nodes



Acquiring Similarity Knowledge

- Key issue for case-based recommenders
- Case-level similarity function:

$$\mathrm{Sim}(\mathrm{T},\mathrm{C}) = \frac{\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{w_i} \times \mathrm{Sim}(\mathrm{v_{C,i}}\ ,\mathrm{v_{T,i}})}{\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{w_i}}$$

- Need to develop:
  - Feature-based similarity metrics
  - Weighting functions – the relative importance of features in terms of overall case similarity (i.e. is feature *x* more important than *y* – consider the feature *location* in the property domain…)
- Such knowledge can be hard-coded by domain experts (cost applies)
- Automated approaches:
  - E.g. learn feature weights by re-ranking recommendation lists via user feedback to minimise average ordering error …

Making Recommendations

- Non-personalised recommendations:
  - Rank recommendation candidates by similarity to the target item

- Personalised recommendations:
  - Rank recommendation candidates by similarity to the target user's profile
  - Various approaches depending on how the profile is constructed

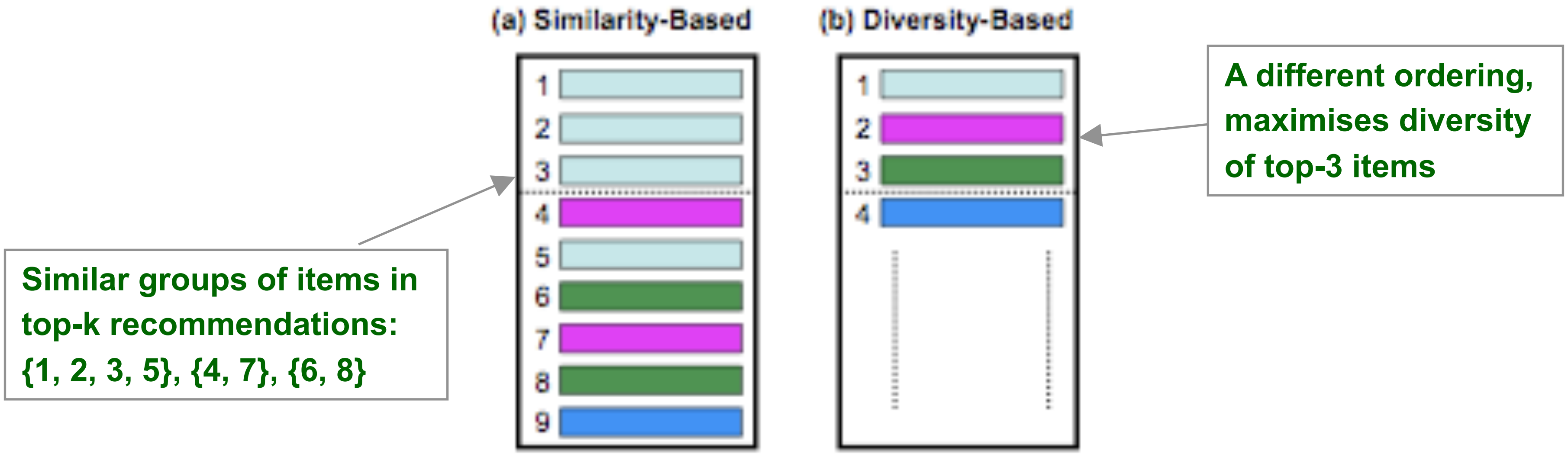
Similarity-based Recommendation

- Similarity-based recommendation often result in recommendations that lack diversity – i.e. the top recommendations are similar to both the target item/ user profile (✔) and each other (✗)

- Explore retrieval approaches that maintain high similarities with target item/user profile while also promoting recommendation diversity

Similarity vs. Diversity

- Top-k retrieved items may be similar to the target item/user profile in similar ways
- Ideally we want the top-k retrieved items to be equally similar to the target item/user profile but in different ways



- Consider two diversity enhancing algorithms – *Bounded Greedy Selection* and *Shimazu's Algorithm*

Bounded Greedy Selection

```
define BoundedGreedySelection(t, C, k, b)
begin
  C′:= bk cases in C that are most similar to t
  R := {}
  for i = 1 to k
    Sort C′ by Quality(t, c, R) for each c in C′
    R := R + First(C′)
    C′:= C′ - First(C′)
  end
  return R
end
```

t – target case(item); C – case base; k – size of recommended set;
b – bound for initial similarity-based retrieval; R – recommendation set

$$Quality(t, c, R) = \alpha \times Similarity(t, c) + (1 - \alpha) \times RelDiversity(c, R)$$

$$RelDiversity(c, R) = \begin{cases} 1 & \text{if } R = \{\} \\ \frac{1}{|R|} \sum_{r \in R} \left(1 - Similarity(c, r)\right) & \text{otherwise} \end{cases}$$

Bounded Greedy Selection

- Bounded Greedy Selection algorithm – key results:
  - Significant improvements in recommendation diversity: 50% improvement in relative diversity for top-3 lists
  - Minor reductions in target case similarity: < 10% loss in similarity between top-3 recommendations and target case compared to standard similarity-based retrival

Shimazu's Algorithm

- Consider recommendation list sizes of 3 (approach generalises to larger lists)

- Select 3 recommendations c_1, c_2 and c_3 for target case q such that:
  - c_1 is maximally similar to q
  - c_2 is maximally dissimilar to c_1
  - c_3 is maximally dissimilar to c_1 and c_2

Shimazu's Algorithm

- Consider recommendation list sizes of 3 (approach generalises to larger lists)

- Select 3 recommendations c_1, c_2 and c_3 for target item q such that:
  - c_1 is maximally similar to q
  - c_2 is maximally dissimilar to c_1
  - c_3 is maximally dissimilar to c_1 and c_2

- But – similarity of recommendations to target item can be compromised:
  - Can limit to where c_1, c_2 and c_3 are drawn from a set of items that are all sufficiently similar to the target item/profile to begin with

Content-based Recommendation

- Items are recommended which are are similar in content to previously selected items
- Recommendations are based on a description of the content of items as opposed to what people actually thought about the items
- Advantages: early recommendations can be made – once a user has selected a single item, new recommendations can be provided (not the case for collaborative filtering recommenders)
- However:
  - Feature identification and extraction can be problematic in some domains – e.g. items may not have a readily available content description (e.g. art, jokes…)
  - Content-based filters, unlike people, cannot distinguish between low and high quality items
  - Users are recommended similar items to those selected previously – a “more like this” approach, low diversity or serendipity

Performance Evaluation

- So far – we have considered a number of possible approaches (heuristically motivated) to content-based recommendation
- Need to perform rigorous evaluation using systematic scientific experiments

Evaluating Recommender Systems

- Live-user Trials:
  - Analysis of real usage and recommendation feedback. A/B testing to evaluate different algorithms. Facilitates a holistic approach to system evaluation. Preferred approach but expensive.

- Offline Evaluations:
  - Best suited to testing core recommendation algorithm components by using existing datasets.

- Methodology & Metrics
  - Repeated random sub-sampling, cross-fold validation
  - Relevance, coverage, diversity…

Evaluation Methodology

- Repeated Random Sub-Sampling:
  - Randomly split the data set into training and test data; e.g. 5x 80/20 splits.
  - For each split evaluate the test data based on recommendations from the training data.
  - Average evaluation results across the splits.

- K-Fold Cross Validation:
  - Randomly partition the data set into K subsamples.
  - Of the K subsamples, a single subsample is used as the test data, with the remaining K-1 subsamples as training data.
  - Repeat K times (the folds), using each subsample in turn as the test set.
  - Average evaluation results across the folds.

- Leave-One-Out
  - Select a single observation from the data set as the test data with the remaining data as the training set.
  - Repeat for each of the observations and average over the individual tests.

Evaluation Metrics

- Precision and Recall:
  - Precision can be seen as a measure of exactness or fidelity, whereas Recall is a measure of completeness.
  - Precision (P) represents the probability that a recommended item is relevant.
  - Recall (R) represents the probability that a relevant item is recommended.
- F_1 Measure:
  - Precision and recall are often conflicting metrics. For example, increasing the number of recommendations is likely to improve recall, but reduce precision.
  - To resolve this conflict, the F_1 measure, which combines the precision and recall metrics, can be used.

$$P = \frac{|T \cap R|}{|R|} \qquad R = \frac{|T \cap R|}{|T|} \qquad F_1 = \frac{2 \times P \times R}{P + R}$$

(where, for a given user, T is the test set and R is the recommended set)

Evaluation Metrics

- Diversity:
  - Mean pairwise dis-similarity between recommended items ($r_1,\ldots,r_N$)
  - Prefer recommendation sets containing a broad set of diverse items

$$Diversity(r_1,...,r_N) = \frac{\sum_{i=1...N} \sum_{j=1...N \wedge i \neq j} (1 - sim(r_i, r_j))}{N(N-1)}$$

- Coverage – different variations – one variation:
  - The percentage of users for whom recommendations can be made

This Topic…

- Content-based recommender systems
- Distinguished between traditional content-based recommendation scenarios (e.g. recommendating documents) and case-based recommendation (e.g. recommending items described by features)
- Key differences – item representation and item similarity (unstructured vs. structured)
- Diversity enhancing approaches
- Evaluation methodology and metrics

Next Topics…

- Collaborative filtering and conversational recommender systems…