Introduction to Utility-based Preference Elicitation and Recommender Systems

Paolo Viappiani
CNRS and LIP6
UPMC, France

March, 20, 2013
Me

- Paolo Viappiani
- CNRS researcher
- LIP6 laboratory - UPMC
- Email: paolo.viappiani@lip6.fr
Summary

- Overview of preference-handling systems
- Conversational recommender systems and user interface issues
- Preferences in AI and foundations of Decision theory
- Utility-based recommendations
- Recommendations with subjective features
Preference-aware Systems

- Personalization technologies are becoming widespread (web applications and more)
  - Aiding the user in carrying out tasks
  - Recommender systems, conversational systems, personal agents, interface assistants,…

- Interactive artificial intelligence systems employ preferences in both reasoning and interaction
  - Personalization is a source of benefit and challenges
  - Designing successful preference-aware systems is challenging
Technical Challenges

1) **Degree of interactivity**: from fully manual through varying degrees of mixed initiative, to fully automated.

2) **Preference representation**: which way to effectively model preferences?

3) **Preference elicitation**: how to assess the user preference model

4) **Nature of the reasoning with preferences**: how to do inference, add new preferences, use the preference information

5) **Resulting output or behavior of the system**: information, decisions, actions
Ecommerce Applications
Recommender Systems

- **Conversational systems**
  - Aim to replace a «decision analyst»
  - The user is (at least partially) active
  - Mixed-initiative interaction, dialog systems, query-answering

- **Collaborative filtering**
  - Aim to provide automatic marketing
  - (Mostly) Passive observations of user preferences
  - Statistical analysis of user data

- The distinction is more and more blurred
Preference-based Search

- Product configuration
- Large collection of outcomes

- Users are not familiar with available items and features
- Users do not know their preferences: theory of preference construction [Payne]
- Biases in decision: framing, prominence, means-objectives [Gilovich, Kahneman]
SmartClient [Torrens, Pu & Faltings]
Teaching Salesman [Stolze]

- « need-based » conversational recommender system
- TS helps user choose an item based on his or her needs
- Options characterized at the user layer, abstract features and user attributes
- Preference discovery, preference optimization, preference debugging
Collaborative Filtering

Help users making choices, advertise interesting products (personalized marketing), suggest new products based on purchases (ratings) of similar users

- Basic idea: similar people like similar things
- Identify similarity between users and/or items
Personal Agents

- Examples: CAP (calendar apprentice) and PexA (project execution assistant)
- Cognitive assistants, Tutoring systems
- When to take action? When to prompt the user?
- How to expose the agent's assistance without degrading the user experience

[Myers et al., 2007]
Personalized Interfaces: Supple

[Gajos & Weld]

Adapt interfaces according to preferences and abilities

Can improve UI by either critiquing or by direct query-answering

http://www.eecs.harvard.edu/~kgajos/research/supple/
Ad Placement

Point of view of publisher: which bidding strategy?
Point of view of advertiser: which mechanism?
Social Networks & AI

- Rising interests in modeling social networks with mathematical models

- **Goal**: learn the structure of the networks
  - Who is friend with who?
  - *People you might know*...

- The **Contagion** effect
  - Product adoption can spread through the network
    - *Homophily*: people behave as their peers
    - Which nodes are most influential?

- **Applications**: targeted marketing campaign, recommender systems with social component
Preference-based Search

"If users cannot find a product, they cannot buy it", J. Nielsen

Slide from Pearl Pu
The user

User is passive

- Knows exactly what he is looking for
- Knows what is looking for, but unsure about preferences
- Does not know what he is looking for

User is active

- Database query
- Conversational Systems (example-based)
- Collaborative filtering

Interaction
Is form-filling a reasonable strategy for preference elicitation?

- Example: actual scenario with travel website (July 5\textsuperscript{th}, 2006)
- User wants to travel from Geneva to Dublin
- Return flight
- Preferences
  - Outbound flight, arrive by 5pm
  - Inbound flight, arrive by 3pm
  - (Cheapest)
To be there at 5pm, I should leave around noon.

To arrive back at 3pm, I should leave in the morning.
Leave out preference about SWISS

Still expensive but cheaper; does not arrive at the preferred time
Omit preference about departure time

Outbound arrives by 5pm; Return arrives by 3pm, as desired

Much Cheaper

Fare per person: 635 CHF (excl. taxes and fees)
Total for all passengers: 704 CHF (incl. taxes and fees)
Preference Construction

Users’ preferences are often constructed when considering specific examples

- Behavioral decision theory [Payne et al. ’93; Slovic’95; Tversky ’96]
ACE Framework \cite{Pu et al.}

- **Accuracy**: the system's ability to help users find their most preferred item
- **Confidence**: its ability to inspire trust in selecting the item displayed by the system
- **Effort**: amount of effort (cognitive, time) required

Since the user cannot observe accuracy directly, it is important that users feel confident about their choices!
Critiquing Conversational Model

Step 1: user specifies initial preferences

Step 2: the product search tool filters the options and displays recommended items based on users' stated preferences

K best items chosen as recommended set

Step 3: user revises preferences to receive more recommendations

Step 4: user picks the final choice
Guidelines

- **Any effort / any order / any preferences**
  - Many studies show that people cannot state their preferences right at the beginning!

- **Allow preference revision and partial satisfaction**

- **Tradeoff assistance**
  - Tweaking / critiquing
  - “I like this portable PC, can I find something lighter?”
### Partially Satisfied Preferences

#### Search Results

**There is NO apartment completely satisfying all your preferences, but these apartments are cheaper and bigger, although they are slightly farther**

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Price (Fs)</th>
<th>Area (m²)</th>
<th>Bathroom</th>
<th>Kitchen</th>
<th>Distance (mins)</th>
<th>Basket</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>shared apartment</td>
<td>450</td>
<td>25</td>
<td>private</td>
<td>private</td>
<td>20</td>
<td>Basket</td>
</tr>
<tr>
<td>30</td>
<td>room in a house</td>
<td>480</td>
<td>27</td>
<td>private</td>
<td>not available</td>
<td>20</td>
<td>Basket</td>
</tr>
</tbody>
</table>

**these apartments are closer and bigger, although they are slightly more expensive**

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Price (Fs)</th>
<th>Area (m²)</th>
<th>Bathroom</th>
<th>Kitchen</th>
<th>Distance (mins)</th>
<th>Basket</th>
</tr>
</thead>
<tbody>
<tr>
<td>77</td>
<td>shared apartment</td>
<td>550</td>
<td>25</td>
<td>private</td>
<td>not available</td>
<td>5</td>
<td>Basket</td>
</tr>
<tr>
<td>34</td>
<td>room in a house</td>
<td>600</td>
<td>30</td>
<td>shared</td>
<td>private</td>
<td>5</td>
<td>Basket</td>
</tr>
</tbody>
</table>

**these apartments provide private bathrooms, although they are slightly smaller**

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Price (Fs)</th>
<th>Area (m²)</th>
<th>Bathroom</th>
<th>Kitchen</th>
<th>Distance (mins)</th>
<th>Basket</th>
</tr>
</thead>
<tbody>
<tr>
<td>69</td>
<td>shared apartment</td>
<td>470</td>
<td>15</td>
<td>private</td>
<td>shared</td>
<td>10</td>
<td>Basket</td>
</tr>
<tr>
<td>72</td>
<td>shared apartment</td>
<td>500</td>
<td>12</td>
<td>private</td>
<td>shared</td>
<td>15</td>
<td>Basket</td>
</tr>
</tbody>
</table>

26
User-Initiated Critiquing

System-Suggested Critiquing

[Chen & Pu]
## Explanations

[Chen & Pu]

<table>
<thead>
<tr>
<th>Why?</th>
<th>-</th>
<th>$1'379.00</th>
<th>3.3 GHz</th>
<th>2 hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'179.00</td>
<td>3.2 GHz</td>
<td>2 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'529.00</td>
<td>1.7 GHz</td>
<td>6.5 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'599.00</td>
<td>1.7 GHz</td>
<td>6.5 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'425.00</td>
<td>1.6 GHz</td>
<td>5.5 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$2'235.00</td>
<td>1.8 GHz</td>
<td>2.5 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'190.00</td>
<td>3.2 GHz</td>
<td>1 hour</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'125.00</td>
<td>1.5 GHz</td>
<td>6 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$2'319.00</td>
<td>1.67 GHz</td>
<td>4.5 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'499.00</td>
<td>1.5 GHz</td>
<td>5 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'739.99</td>
<td>1.5 GHz</td>
<td>4.5 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'629.00</td>
<td>1.8 GHz</td>
<td>5.8 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'625.99</td>
<td>1.5 GHz</td>
<td>5 hours</td>
</tr>
<tr>
<td>Why?</td>
<td>-</td>
<td>$1'426.99</td>
<td>1.5 GHz</td>
<td>5 hours</td>
</tr>
</tbody>
</table>

*This product has higher processor speed and bigger hard drive capacity but is heavier.*
### The most popular product

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Price</th>
<th>Processor speed</th>
<th>Battery life</th>
<th>Installed memory</th>
<th>Hard drive capacity</th>
<th>Display size</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$2,095.00</td>
<td>1.67 GHz</td>
<td>4.5 hours(s)</td>
<td>512 MB</td>
<td>00 GB</td>
<td>36.6 cm</td>
<td>2.54 kg</td>
</tr>
</tbody>
</table>

### We also recommend the following products because they are cheaper and lighter, but have lower processor speed

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Price</th>
<th>Processor speed</th>
<th>Battery life</th>
<th>Installed memory</th>
<th>Hard drive capacity</th>
<th>Display size</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$1,499.00</td>
<td>1.5 GHz</td>
<td>5 hours(s)</td>
<td>512 MB</td>
<td>00 GB</td>
<td>33.9 cm</td>
<td>1.91 kg</td>
</tr>
<tr>
<td></td>
<td>$1,739.99</td>
<td>1.5 GHz</td>
<td>4.5 hours(s)</td>
<td>512 MB</td>
<td>00 GB</td>
<td>36.5 cm</td>
<td>2.49 kg</td>
</tr>
<tr>
<td></td>
<td>$1,825.99</td>
<td>1.5 GHz</td>
<td>5 hours(s)</td>
<td>512 MB</td>
<td>00 GB</td>
<td>30.7 cm</td>
<td>2.09 kg</td>
</tr>
<tr>
<td></td>
<td>$1,426.99</td>
<td>1.5 GHz</td>
<td>4 hours(s)</td>
<td>512 MB</td>
<td>60 GB</td>
<td>30.7 cm</td>
<td>2.09 kg</td>
</tr>
<tr>
<td></td>
<td>$1,929.00</td>
<td>1.2 GHz</td>
<td>4 hours(s)</td>
<td>512 MB</td>
<td>60 GB</td>
<td>26.9 cm</td>
<td>1.41 kg</td>
</tr>
<tr>
<td></td>
<td>$1,905.00</td>
<td>1 GHz</td>
<td>5.5 hours(s)</td>
<td>512 MB</td>
<td>40 GB</td>
<td>26.9 cm</td>
<td>1.41 kg</td>
</tr>
</tbody>
</table>

### They have higher processor speed and bigger hard drive capacity, but are heavier

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Price</th>
<th>Processor speed</th>
<th>Battery life</th>
<th>Installed memory</th>
<th>Hard drive capacity</th>
<th>Display size</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$1,220.49</td>
<td>1.8 GHz</td>
<td>5 hours(s)</td>
<td>1 GB</td>
<td>100 GB</td>
<td>36.1 cm</td>
<td>2.95 kg</td>
</tr>
<tr>
<td></td>
<td>$2,148.99</td>
<td>2 GHz</td>
<td>4 hours(s)</td>
<td>1 GB</td>
<td>100 GB</td>
<td>39.1 cm</td>
<td>2.9 kg</td>
</tr>
<tr>
<td></td>
<td>$1,379.00</td>
<td>3.3 GHz</td>
<td>2 hours(s)</td>
<td>512 MB</td>
<td>100 GB</td>
<td>45.2 cm</td>
<td>4.31 kg</td>
</tr>
<tr>
<td></td>
<td>$2,235.00</td>
<td>1.8 GHz</td>
<td>2.5 hours(s)</td>
<td>1 GB</td>
<td>100 GB</td>
<td>45.2 cm</td>
<td>3.99 kg</td>
</tr>
<tr>
<td></td>
<td>$2,319.00</td>
<td>1.7 GHz</td>
<td>4.5 hours(s)</td>
<td>512 MB</td>
<td>100 GB</td>
<td>43.2 cm</td>
<td>3.13 kg</td>
</tr>
<tr>
<td></td>
<td>$2,075.00</td>
<td>1.8 GHz</td>
<td>1.67 hours(s)</td>
<td>512 MB</td>
<td>100 GB</td>
<td>43.2 cm</td>
<td>4.4 kg</td>
</tr>
</tbody>
</table>
Anchoring effect

- Users are biased to what is shown to them (Tversky1974)

Example

- Three laptops that all weigh around 3-4 kg
- The user might never consider a lighter model

Metaphor: local optimum

- When all options look similar, motivation to state additional preference is low
Diverse Suggestions

- **Case-based Reasoning**
  - $Quality(x, S) = a \times Sim(target) + (1-a) \times Diversity(x, S)$
  - $Quality(x, S) = Sim(target) \times Diversity(x, S)$

- **Model-based Suggestions**
  - Probability that an option can become optimal

- **Utility-based “diversity”** (see later)
Form-filling is not Effective

- Incorrect means objectives: formulate the real goal by a “substitute” goal believed to lead to desired outcome
  - Users often state more preferences than necessary when prompted
  - *The preference model may be complete, but not accurate*
- Decision biases studied by *behavioral decision theory*
The lookahead principle

Suggestions should not be optimal under the current preference model, but should provide a high likelihood of optimality when an additional preference is added.
CritiqueShop [Reilly, Zhang, McGinty, Pu, Smyth]
Decision accuracy & user effort

Model-based suggestions leads to more accurate decisions
Question/answering leads to inaccurate decisions
How Many to Show?

- Number of preferences discovered according to the lookahead principle
- Adaptive model-based suggestions perform even better than simple model-based suggestions
Interview-based tool
Example-based Tool: Perceived Accuracy
The « Cold Start » Problem

- Users are rarely willing to spend time training the system, beyond stating few crude preferences
  - This is well known for collaborative filtering recommender systems, and it is known as the «cold start problem»
  - Common solution: use demographic data, « standard » recommendations, …

- For other systems: the importance of effective preference elicitation (or acquisition)
  - Aim at learning preferences with as little effort as possible
  - Focus learning on the most informative data
  - Consider implicit preferences
Decision-theoretic Preference Acquisition

- Heuristic methods sometimes work reasonably well.
- Can we do better by adopting a principled decision-theoretic view?
- What AI methods can offer to preference elicitation (learning) for real-time interactive systems?
Preference Elicitation in AI

- AI systems need to recommend decisions on behalf of individuals (groups)

- The *preference elicitation bottleneck*
  - What is the objective function?
  - User preferences (or utilities) are unknown
  - Elicitation of preferences is expensive!

- Challenging questions
  - What are sources of preference information?
  - What preference info is *relevant* the task at hand?
  - Is the elicitation effort *worth the improvement* it offers in terms of decision quality
Preference Modeling

- Give a precise meaning to « what it means to prefer something »
  - Assume a finite outcome set $X$ (states, items, products,...)

- Basic ingredient: a preference relation $\succeq$
  - $a \succeq b$: « $a$ is at least as preferred as $b$ »
  - Strict preference relation $>$
    - $a > b$ iff $a \succeq b$ and not $b \succeq a$
  - Indifference relation $\approx$
    - $a \approx b$ iff $a \succeq b$ and $b \succeq a$
  - Incomparability relation $\equiv$
    - $a \equiv b$ iff neither $a \succeq b$ nor $b \succeq a$

Preference structure $(>,\approx,\equiv)$ induced by $\succeq$
Preference Modeling

Possible properties of preference relation $\succeq$:
- reflexive, transitive (partial order)
- reflexive, transitive, antysimmetric (preorder)
- reflexive, transitive, complete (total preorder)
- reflexive, transitive, antysimmetric, complete (total order)

Total preorders/orders are always reasonable?

What does it mean for two items to be incomperable?
- Lack of knowledge
- Lack of tradeoff (true incomparability)
In large domains it is almost impossible to specify the preference relation explicitly for all pairs

- Preference representation means defining a formalism to let express preferences in an efficient way

Different possibilities

- Qualitative approaches (mostly in AI)
- Quantitative approaches (utility-based)
Preference Languages

- Language to « express » preference
  - Logic formalism to take about preferences

- « I prefer A over B » where A and B are formulas
  - A and not B is preferred to having B and not A [Von Wright]

- Different semantics are possible
  - Strong: all items satisfying (A and not B) are preferred to all items satisfying (B and not A)
  - Ceteris Paribus: everything else being equal
  - Optimistic: the best item satisfying (A and not B) is preferred to the best item satisfying (B and not A) according to $\succ$
  - Pessimistic, Opportunistic, …

[Kaci, Working with Preferences, Springer 2011]
Preference Independence

- Two attributes $X_1$ and $X_2$ are preferentially independent (P.I.) of a third attribute $X_3$
  - if the preference between outcomes $<x_1,x_2,x_3>$ and $<x_1',x_2',x_3>$ does not depend on the particular value $x_3$ for attribute $X_3$

- Mutual preference independence: two attributes are preferentially independent of each other
  - If one assume all attributes are MPI → the decision can be characterized with an additive utility function (will return to that)
Representation for Qualitative Preferences

- CP-nets and follow-ups (TCP-nets,...)
  - Graphical model where a link connects a set of Parents to a Child node
  - Preference on the variable child can only depend to values of Parent nodes
AI: from Goals to Preferences

- Classic AI (planning, autonomous agents) focused on the notion of « goal »
  - *Example*: robot needs to navigate in a room and find exit, find and disarm a mine

- Notion of goal insufficient in many circumstances:
  - What if the goal cannot be met? Partially satisfied?
  - What if there are many possible ways to achieve the goal, how to differentiate between them?

- Preferences in AI
  - Preference-based planning, preference-based user interfaces,...
Utility Representations

- **Utility function** $u: X \rightarrow [0,1]$
- **$X$ can be combinatorial, sequential, etc.**
  - Representing, eliciting $u$ difficult in explicit form
  - Flat utility representation is often unrealistic

<table>
<thead>
<tr>
<th>Car</th>
<th>Model</th>
<th>Color</th>
<th>Motor</th>
<th>Consumption</th>
<th>...</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car 1</td>
<td>Toyota Prius</td>
<td>Silver</td>
<td>125hp</td>
<td>5.6/100k</td>
<td></td>
<td>0.82</td>
</tr>
<tr>
<td>Car 2</td>
<td>Acura TL</td>
<td>Black</td>
<td>286hp</td>
<td>8.9l/100k</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>Car 3</td>
<td>Acura TL</td>
<td>Blue</td>
<td>286hp</td>
<td>8.9l/100k</td>
<td></td>
<td>0.96</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Additive Utility Functions

- Additive representations (very common in MAUT)
  - Sum of local utility functions $u_i$ over attributes (or local value functions $v_i$ multiplied by scaling weights)
  - Exponential reduction in the number of needed parameters!

$$u(x) = \sum_{i=1}^{n} u_i(x_i) = \sum_{i=1}^{n} \lambda_i v_i(x_i).$$

$\lambda_1 = 0.2$

<table>
<thead>
<tr>
<th>Color</th>
<th>$v_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>red</td>
<td>1.0</td>
</tr>
<tr>
<td>blue</td>
<td>0.7</td>
</tr>
<tr>
<td>grey</td>
<td>0.0</td>
</tr>
</tbody>
</table>
Dominance & Pareto Optimality

- Idea: a rational decision maker will never picks something that is worse on every feature/attribute of interest.
  - Option a is dominated by b iff $v_i(b) \geq v_i(a)$ for all attributes $i$ and for at least one attribute $j$, $v_j(b) > v_j(a)$
  - Option a is Pareto-optimal if it is not dominated by any other option.
Generalized Additive Utility Models

- Generalized additive models
  - More flexible
  - Assume a set of subsets of variables (possibly overlapping!)
  - Use sum of local utility functions over attributes
  - Each local function concerns several attributes

\[ u(x) = \sum_{j=1}^{m} u_j(x_j) \]

\begin{array}{|c|c|c|}
\hline
\text{Color} & \text{Drs} & u_1 \\
\hline
\text{red} & 2 & 1.0 \\
\text{blue} & 4 & 0.9 \\
\text{red} & 4 & 0.6 \\
\text{blue} & 2 & 0.4 \\
\hline
\end{array}

\begin{array}{|c|c|c|}
\hline
\text{Pwr} & \text{Drs} & u_1 \\
\hline
350 & 2 & 1.0 \\
350 & 4 & 0.7 \\
280 & 2 & 0.65 \\
280 & 4 & 0.55 \\
\hline
\end{array}

\[ u(\text{red}, 2\text{dr}, 280\text{hp}) = 0.79 \]
Making Decision: Uncertainty

- Final outcome depends on our choice, but uncertainty is involved
- Example: decide how many days to study for an exam
  - Probabilistic Uncertainty: principle of Maximum Expected Utility
  - Strict Uncertainty: no way to quantify uncertainty, but each outcome has a different value
Utility of Money & Risk Attitudes

Money is a « special » attribute
- Normally: \( U(100\$) > 0.5 \ U(200\$) \)
- Concept of Certainty Equivalent (CE)

Risk attitudes
- Risk averse decision maker: \( U(\text{money}) \) concave
- Risk neutral: \( U(\text{money}) \) is linear
- Risk seeker: \( U(\text{money}) \) is convex

Utility is often expressed in a monetary scale (quasi-linear utility model)
Decision Criteria for Strict Uncertainty

- **Strict Uncertainty**: no way to quantify uncertainty, but each outcome has a different value

- **Which criteria?**
  - **Maximin**: choose action with best worst outcome
    - \( \max_a \min_\Theta v(f(a,\Theta)) \)
    - \( a \) with max security level \( s(a) \)
    - very pessimistic

  - **Maximax**: choose action with best best outcome
    - \( \max_a \max_\Theta v(f(a,\Theta)) \)
    - \( a \) with max optimism level \( o(a) \)

- **Hurwicz criterion**: set \( w \in (0,1) \)
  - \( \max w \, s(a) + (1-w) \, o(a) \)

<table>
<thead>
<tr>
<th></th>
<th>( \Theta_1 )</th>
<th>( \Theta_2 )</th>
<th>( \Theta_3 )</th>
<th>( \Theta_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_1 )</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( a_3 )</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( a_4 )</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Minimax Regret (Savage)

- Regret of action $a_i$ under nature choice $\Theta_j$:
  - $r_{ij} = \max \{v_{kj}\} - v_{ij}$

- Minimax regret chooses $\arg \min_a \max_j r_{ij}$

<table>
<thead>
<tr>
<th></th>
<th>$\Theta_1$</th>
<th>$\Theta_2$</th>
<th>$\Theta_3$</th>
<th>$\Theta_4$</th>
<th>Max Regret</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>2 / 0</td>
<td>2 / 2</td>
<td>0 / 1</td>
<td>1 / 0</td>
<td>2</td>
</tr>
<tr>
<td>$a_2$</td>
<td>1 / 1</td>
<td>1 / 3</td>
<td>1 / 0</td>
<td>1 / 0</td>
<td>3</td>
</tr>
<tr>
<td>$a_3$</td>
<td>0 / 2</td>
<td>4 / 0</td>
<td>0 / 1</td>
<td>0 / 1</td>
<td>2</td>
</tr>
<tr>
<td>$a_4$</td>
<td>1 / 1</td>
<td>3 / 1</td>
<td>0 / 1</td>
<td>0 / 1</td>
<td>1</td>
</tr>
</tbody>
</table>

Slide adapted from Craig Boutilier
Classic Utility Elicitation

- Assessment of multi attribute utility functions
  - Long list of queries (questions)
  - Focus on high risk decisions
  - Goal: learn weights up to small error
- Queries
  - Local: focus on an attribute in isolation
  - Global: compare different attributes

**STANDARD GAMBLE QUERIES**

- Choose between option $x_{12}$ for sure
- OR
- a gamble $< x^T, \ell, x^\perp, 1-\ell >$?

Semantic of SGB is equivalent to ask: “$U(x_{12}) > \ell$?”
Elicitation of Additive Models

- Consider an attribute (for example, Color)
  - Ask for best value of Color (say, red)
  - Ask for worst value (grey)
  - Ask local standard gamble for each remaining color to assess its local utility value

- «Bound» queries can be asked
  - Refine intervals on local utility
  - More understandable than SGB

- Assess weights of individual attributes (scaling factors)
  - Define reference outcome
  - Ask global queries in order to assess the difference in utility when, starting from the reference outcome, «moving» a particular attribute to the best / worst
Elicitation: Beyond the Classical View

- The classic view involving standard gambles difficult:
  - Large number of parameters to assess
  - Unreasonable precision required
  - Queries over full outcomes
  - Cost (cognitive, communication, computational, revelation) may outweigh benefit
    - Can often make optimal decisions without full utility information

- General approach to practical, automated elicitation
  - Cognitively plausible forms of interaction
  - Incremental elicitation until decision possible that is good enough
  - Collaborative/learning models to allow generalization across users
Preference Elicitation

- Decisions on behalf of individuals (groups)
  - Product configuration, recommender systems, personal assistants, ad auctions, ...

- The preference elicitation bottleneck
  - What is the objective function?
  - User preferences (or utilities) are unknown
  - Elicitation of preferences is expensive!

- Challenging questions
  - What are sources of preference information?
  - What preference info is relevant the task at hand?
  - Is the elicitation effort worth the improvement it offers in terms of decision quality
Utility Function Uncertainty

- General frameworks
  - Bayesian models
  - Strict uncertainty models

- Key components
  - decision criterion: what decision given the current knowledge about the user's utility function?
  - effective update (function of queries/interaction)
  - elicitation strategy: which query to ask next?
  - termination condition: when is decision “good enough”
A General Framework for Elicitation and Interactive Decision Making

- **Bel**: beliefs about user’s utility function $u$
- **Opt(Bel)**: “optimal” decision given incomplete, noisy, and/or imprecise beliefs about $u$

Repeat until $Bel$ meets some termination condition
- Ask user some query (propose some interaction) $q$
- Observe user response $r$
- Update $Bel$ given $r$

Return/recommend **Opt(Bel)**
Active Collaborative Filtering
The Basic Decision Problem

- Multiattribute outcome set defined over \( \{X_1 \ldots X_n\} \)
- Feasible set \( X \) defined by constraints, product DB, etc.
  - \( X \) might be very large in practical applications!
  - Some optimization process usually required

- Utility function \( u: X \rightarrow \mathbb{R} \)
  - Optimal decision \( x^* \) maximizes utility
  - Goal: find \( x^* \) in \( \arg\max_x \{u(x) : x \in X\} \)

- Utility representation critical to assessment
  - Some structural form usually assumed
  - \( u \) parameterized compactly (weight vector \( w \))
    - e.g., linear/additive, generalized additive models
    - \( u(x; w) = w_1 * f^1(x_1) + w_2 * f^2(x_2) \)

- However utility is not known with certainty \( \rightarrow \) ask queries
AI's Take on Utility Elicitation

- **Complete elicitation: too much burden**
  - Frequent approaches in RecSys: one-fits-all, product navigation based on similarity-measure, heuristic utility models

- **Adaptive utility elicitation**
  - Near-optimal decisions can be made with partial utility information
  - Two main frameworks to represent uncertainty:
    - constraints **vs** Bayesian
Online Recommendation Systems
Options shown with dual goal of recommendation and elicitation

- User might end the interaction unpredictably
- A set can be interpreted as a query or as a recommendation
  - Are we losing anything?
  - Computational advantages?
- Challenges
  - Limited number of interactions
  - Large product catalogs
  - Small display set
  - User responses are noisy

[Reilly, Zhang, McGinty, Pu, Smyth]
Exploitation vs Exploration?

- Natural tension between recommendation and elicitation

Goal: exploit current information

Note: since utility is uncertain, there can be value in recommending a set

Goal: acquire further information in order to make better recommendation

How to resolve this tension?

- Focus on choice queries
  - A set can be interpreted both as a query and as a recommendation

Thanks, I will buy product A!
How to Choose Recommendations and Queries?

- How to reason with an uncertain utility function?
- How to aggregate utility uncertainty?
- We extend classic decision criteria to sets

<table>
<thead>
<tr>
<th></th>
<th>Bayesian elicitation</th>
<th>Regret based elicitation (strict uncertainty)</th>
<th>Maximin elicitation (strict uncertainty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Recommendation</td>
<td>Expected Utility</td>
<td>Minimax Regret</td>
<td>Maximin Utility</td>
</tr>
<tr>
<td>Recommendation Set</td>
<td>Expected Utility of Selection</td>
<td>Setwise Regret</td>
<td>Setwise Maximin</td>
</tr>
<tr>
<td>Query Set</td>
<td>Expected Posterior Utility</td>
<td>Worst-case posterior Regret</td>
<td>Worst-case posterior maximin</td>
</tr>
</tbody>
</table>
Decision-making with Bayesian Uncertainty

Utility is parametric in $w$
For instance: $u(x; w) = w \cdot x$ (linear utility model)

$\theta$ is the current belief
$P(w; \theta)$ probability of parameter $w$

Each point represents a vector $w$ of parameters

Expected utility of an outcome wrt utility uncertainty
$EU(x; \theta) = \int u(x; w) P(w; \theta) \, dw$

$x^* = \text{argmax } EU(x; \theta)$ is the optimal outcome wrt $\theta$
$EU^* = \text{max } EU(x; \theta)$
Inference: Belief Update

- Preference statements (and query response) require the distribution to be updated → new belief
- Inference methods: MonteCarlo, Expectation Propagation
- Our main results are independent of the inference method

Importance Sampling

Prior $\theta$  

Sample

weight particles by likelihood of user responses

Posterior $\theta'$ (updated belief)

I prefer product A to B
Bayesian Elicitation: Criteria for Recommendation Sets and Queris

Expected Utility of Selection

\[ EUS_R(S; \theta) = \sum_{x \in S} P_R(S \sim x; \theta) \cdot EU(x; \theta | S \sim x) \]

I recommend you one of the following:

- (0,0,1,1,0)
- (1,0,0,1,0)
- (1,1,0,1,1)

Expected Posterior Utility

\[ EPUR_S(S; \theta) = \sum_{x \in S} P_R(S \sim x; \theta) \cdot EU^*(\theta | S \sim x) \]

Which one do you prefer?

- (0,0,1,1,0)
- (1,0,0,1,0)
- (1,1,0,1,1)
Bayesian Elicitation: Criteria for Recommendation Sets and Queris

**Expected Utility of Selection**

I recommend you one of the following:

- \((0,0,1,1,0)\)
- \((1,0,0,1,0)\)
- \((1,1,0,1,1)\)

\[
EUS_{R}(S; \theta) = \sum_{x \in S} P_{R}(S \sim x; \theta) EU(x; \theta|S \sim x)
\]

\[\land\]

**Expected Posterior Utility**

Which one do you prefer?

- \((0,0,1,1,0)\)
- \((1,0,0,1,0)\)
- \((1,1,0,1,1)\)

\[
EPU_{R}(S; \theta) = \sum_{x \in S} P_{R}(S \sim x; \theta) EU^{*}(\theta|S \sim x)
\]
Response Model

- Selection probabilities $P_R$ depend on:
  - The underlying (true) utility
  - The user's cognitive ability to select the most preferred outcome

- Probability of selecting $x$ from $S$ given utility $w$
  - Response model gives $P_R(S \rightarrow x; w)$
  - Then, we can compute selection probability given belief $\theta$
    $$P_R(S \rightarrow x; \theta) = \int P_R(S \rightarrow x; w) P(w; \theta) \, dw$$

- Some common response models
  1) **Noiseless** responses
  2) **Constant**: probability $p$ of selecting a non-preferred outcome
  3) **Logistic** model (aka Luce-Sheppard, mixed-multinomial logit)
Logistic Noisy Response Model

Set A

$P_R \approx 0$

selection probability

$P_R \approx 1$

Set B

$P_R \approx 0.5$

selection probability

$P_R \approx 0.5$

Probability of selection

Logistic function with temperature parameter $\gamma$

$$P_R(S \sim x; w) = \frac{e^{\gamma u(x; w)}}{e^{\gamma u(x; w)} + e^{\gamma u(y; w)}}$$

utility difference
“Operator T”

Define an operator $T_\theta : S \rightarrow S$

Which one do you prefer?

set $S$

set $S' = T_\theta(S)$

EU* optimal in $\theta'$

EU* optimal in $\theta''$
“Operator T”

Define an operator $T_{\theta}: S \rightarrow S$

set $S$

 Which one do you prefer?

Lemma

The new set $T_{\theta}(S)$ is better than $S$ both as query and as recommendation

$EUS(\geq EPU(\in \theta$ for the noiseless and constant noise model

set $S' = T_{\theta}(S)$

EU$^*$ optimal in $\theta'$

EU$^*$ optimal in $\theta''$
Define an operator $T_\theta : S \rightarrow S$

$EPU(\cdot) \geq EUS(\cdot) \geq EPU(\cdot) \geq EUS(\cdot)$ in $\theta$

**Lemma** The new set $T_\theta(S)$ is better than $S$ both as query and as recommendation for the noiseless and constant noise model.

$\text{set } S' = T_\theta(S)$

$EU^* \text{ optimal in } \theta'$

$EU^* \text{ optimal in } \theta''$
Sets can be viewed as both recommendation and choice queries

- *Expected Utility of Selection*: value of a set as recommendation
- *Expected Value of Information*: value of a set as a choice query

Different response/selection models: *noiseless*, *constant noise*, *logistic* (aka mixed multinomial logit, Luce-Sheppard)

**Theorem:**

**Optimal recommendation sets are optimal choice queries**

- Assuming noiseless responses or a constant noise model
- No particular assumption about prior distribution, methods of Bayesian inference.

**Theorem:** *Optimal recommendation sets are near-optimal queries under the logistic noise model*

- We provide the expression for the worst-case loss $\Delta_{\text{max}}$ (surprisingly small)
- Also, the optimal query assuming *noiseless* responses is a near-optimal query under logistic *noise*
Expected Utility of a Selection
(Value of a set as recommendation)

Expected Posterior Utility
(Value of a set as query)
Expected Posterior Utility (Value of a set as query)

Expected Utility of a Selection
Expected Posterior Utility (Value of a set as query)

Value of a set as recommendation

Expected Utility of a Selection
Logistic Noise Model

- Expected Utility of Selection under a noisy model ≤ than the same selection under noiseless model
  - Loss = $\text{EUS}_{\text{NL}}(S; \theta) - \text{EUS}_{\text{L}}(S; \theta)$

- Can we derive bounds under the logistic model?

<table>
<thead>
<tr>
<th>Set A</th>
<th>Set B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection probability</td>
<td>Selection probability</td>
</tr>
<tr>
<td>$P_R \approx 0$</td>
<td>$P_R \approx 0.5$</td>
</tr>
<tr>
<td>select item with highest utility Loss --&gt; 0</td>
<td>both items have ~ same utility Loss --&gt; 0</td>
</tr>
</tbody>
</table>
Set C

utility = $d$

utility = 0

selection probability

$P_R = \text{Logistic}(d)$

$P_R = \text{Logistic}(-d)$

Probability of selecting the first item

$d$, difference in utility
Set C

utility = \(d\)  
utility = 0

selection probability

\[ P_R = \text{Logistic}(d) \]

\[ P_R = \text{Logistic}(-d) \]

\(d > 0 \rightarrow \text{first is best}\)

The loss is the difference \(d\) weighted by chance of wrong selection

\[ \text{Loss} = |d| \times \text{Logistic}(-|d|) \]

\(\Delta_{\text{max}}\) = max loss due to noise

- Function of \(\gamma\) and set size
- Loss \(\Delta_{\text{max}}\) ~ utility amount such that user identifies best item with 56%
Theorem (Logistic noise model)

1) The value of a recommendation set is at most $\Delta_{\text{max}}$ lower than the noiseless

- $E_{\text{US}}(S) \geq E_{\text{US}}(S) - \Delta_{\text{max}}$

- Consequence: we can optimize EUS without considering noise

2) The difference between the value of the optimal query and the optimal recommendation set is at most $\Delta_{\text{max}}$

- $E_{\text{PU}}(\mathcal{L})^* \geq E_{\text{US}}(\mathcal{L})^* - \Delta_{\text{max}}$

- Consequence: we can optimize a set wrt EUS and use it as a query (with bounded loss)

$\Delta_{\text{max}}$ can be expressed in function of the set size and the temperature parameter $\gamma$ (surprisingly low in practice)
Optimization Strategies

- Exact optimization of EPU is $O(n^{k+1})$
- Exact optimization of EUS is $O(n^k)$

Greedy maximization

- Complexity is $O(n)$ in the size of database
- **Submodularity** $\rightarrow$ gain is lower for larger sets
- Worst-case guarantee $\eta$ (73% of optimal value for comparisons) [Wolsey, Nemhauser & Fisher]
- $\eta - \frac{\Delta}{EUS^*_{NL}}$ of optimum (logistic noise)
- Practical speedup: lazy evaluation

Query Iteration (QI) is a local search technique

- $O(n)$ and with smaller constants than greedy
  - Start with an initialization set $S_0$
  - Apply the $T$ operator until no improvement in EUS
- However there might be several fixed point $T(S) = S$
Define operator T that maps a set S to S' 

- Example S= \{o_5, o_4\} 
  - Assume o_4 better than o_5 
    - Compute optimal EU*: this gives o_2 
  - Assume o_5 better than o_4 
    - Compute optimal EU*: this gives o_1 
- New set S'= \{o_1, o_2\}

**Lemma.** S' is both a better query and a better set than S. 

\[
EUS(S') \geq EPU(S) 
\]
Efficient Algorithms for Bayesian Elicitation

- Consequences of our theoretical results: efficient algorithms to generate choice queries
  - Optimizing a recommendation set is simpler and *submodular*
  - Approximated *greedy* strategies with *worst-case guarantees*
  - *Noiseless* optimization quite effective in *noisy* settings
  - Query Iteration strategy particularly efficient for large datasets

**Computation time**

<table>
<thead>
<tr>
<th></th>
<th>Dataset 1</th>
<th>Dataset 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact EPU</td>
<td>1815 s</td>
<td>~2 weeks</td>
</tr>
<tr>
<td>Exact EUS</td>
<td>405 s</td>
<td>~2 hours</td>
</tr>
<tr>
<td>Greedy with lazy evaluation</td>
<td>1.02 s</td>
<td>0.93 s</td>
</tr>
<tr>
<td>Query Iteration (local search)</td>
<td>0.15 s</td>
<td>0.05 s</td>
</tr>
</tbody>
</table>
How Many To Show?

- EVOI in function of the number of items in the query set
- Dataset
  - 506 items
- Logistic noise model
- Monte Carlo methods for Bayesian inference
How to Choose Recommendations and Queries?

- How to reason with an uncertain utility function?
- How to aggregate utility uncertainty?
- We extend classic decision criteria to sets

<table>
<thead>
<tr>
<th></th>
<th>Bayesian elicitation</th>
<th>Regret based elicitation (strict uncertainty)</th>
<th>Maximin elicitation (strict uncertainty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Recommendation</td>
<td>Expected Utility</td>
<td>Minimax Regret</td>
<td>Maximin Utility</td>
</tr>
<tr>
<td>Recommendation Set</td>
<td>Expected Utility of Selection</td>
<td>Setwise Regret</td>
<td>Setwise Maximin</td>
</tr>
<tr>
<td>Query Set</td>
<td>Expected Posterior Utility</td>
<td>Worst-case posterior Regret</td>
<td>Worst-case posterior maximin</td>
</tr>
</tbody>
</table>
Minimax Regret

W = set of feasible utility parameters
Recommend product $x^*$ associated with minimax regret

Max regret:
$$MR(x; W) = \max_{y \in X} \max_{w \in W} u(x^a; w) - u(x; w)$$

Minimax regret:
$$MMR(W) = \min_{x \in X} MR(x, W); \quad x^*_W = \arg\min_{x \in X} MR(x, W)$$
Computation of Minimax Regret

Benders' decomposition + constraint generation techniques

"Master" problem

\[
\begin{align*}
\min_{x,R} & \quad R \\
\text{s.t.} & \quad R > wx^*_w - wx \\
& \quad \forall w \in GEN
\end{align*}
\]

Constraint Generation
(calculates max regret of the master's solution)

"Slave" problem

\[
\begin{align*}
\max_{w,y,Y} & \quad \sum_i Y_i - wx \\
\text{s.t.} & \quad \text{constraints}(W) \\
& \quad Y \text{ "encode" } w^*y
\end{align*}
\]

add the \( w \) and \( x^*_w \) to \( GEN \)
Regret-based Elicitation: Criteria for Recommendation Sets and Queries

- **Setwise Regret**

  \[ SMR(S; W) = \max_{x^a \in X} \max_{w \in W} u(x^a; w) - \max_{x \in S} u(x; w) \]

  adversary chooses \( w \) and \( y \)

  Regret is calculated wrt item in \( S \) with highest utility

- **Worstcase Regret**

  \[ WR(Z) = \max \left[ MMR(W^{Z \rightarrow 1}), \ldots, MMR(W^{Z \rightarrow k}) \right] \]

  minimax regret in the updated belief

  updated belief (add constraint)
Computation of Setwise Minimax Regret

New “master” problem; MIP

“Master” problem

\[
\begin{align*}
\min_{M, I^j_w, x^j, V^j_w} & \quad M \\
\text{s.t.} & \quad M \geq \sum_{1 \leq j \leq k} V^j_w \quad \forall w \in \text{Vert} \\
& \quad V^j_w \geq w \cdot (x^*_w - x^j) + (I^j_w - 1)m_{big} \\
& \quad \forall j \in [1, k] \land \forall w \in \text{Vert} \\
& \quad \sum_{1 \leq j \leq k} I^j_w = 1 \quad \forall w \in \text{Vert} \\
& \quad I^j_w \in \{0, 1\} \\
& \quad V^j_w \geq 0 \quad \forall j \in [1, k], \forall w \in \text{Vert}
\end{align*}
\]

Constraint Generation

(\text{calculate max regret of for each item in the set retrieved by the “master”})

“Slave” problems
Sets can be viewed as both recommendation and choice queries

- Setwise Regret: value of a set as recommendation
- Worstcase Regret: value of a set as a choice query

**Theorem:**

*Optimal recommendation sets are optimal choice queries*

- In the regret-based elicitation model
Empirical Results

- Randomly generated *quasilinear* utility functions
- Real dataset (~200 options)
- User iteratively picks preferred option in a pair \(k=2\)
- Measure regret reduction
- SMMR recommendations are significantly better than CSS
- Hillclimbing (HCT) is as good as SMMR
How to Choose Recommendations and Queries?

- How to reason with an uncertain utility function?
- How to aggregate utility uncertainty?
- We extend classic decision criteria to sets

<table>
<thead>
<tr>
<th></th>
<th>Bayesian elicitation</th>
<th>Regret based elicitation (strict uncertainty)</th>
<th>Maximin elicitation (strict uncertainty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Recommendation</td>
<td>Expected Utility</td>
<td>Minimax Regret</td>
<td>Maximin Utility</td>
</tr>
<tr>
<td>Recommendation Set</td>
<td>Expected Utility of Selection</td>
<td>Setwise Regret</td>
<td>Setwise Maximin</td>
</tr>
<tr>
<td>Query Set</td>
<td>Expected Posterior Utility</td>
<td>Worst-case posterior Regret</td>
<td>Worst-case posterior maximin</td>
</tr>
</tbody>
</table>
Maximin Utility Definition

\( W = \) set of feasible utility parameters (such as importance weights)
\( X = \) set of products
\( x = \) recommendation

**Minimum utility**

\[ \text{MU}(x; W) = \min_{w \in W} u(x; w) \]

**Maximin utility and maximin optimal** \( x^*_w : \)

\[ \text{MMU}(W) = \max \text{MU}(x, W) ; \quad x^*_w = \arg\max_{x \in X} \text{MU}(x, W) \]

\[ x \in X \]
Maximin Recommendation Sets

- **Setwise MMU**

  \[ SMU(S; W) = \min_{w \in W} \max_{x \in S} u(x; w) \]

  adversary chooses \( w \)

  utility is calculated wrt item in \( S \) with highest utility

  I recommend you the following:
  
  \[
  (0,0,1,1,0) \quad (1,0,0,1,0) \quad (1,1,0,1,1)
  \]

- **Optimal recommendation set**

  \[ SMMU(W) = \max_{S : |S|=k} \min_{w \in W} \max_{x \in S} u(x; w) \]

  Which one do you prefer?

  \[
  (0,0,1,1,0) \quad (1,0,0,1,0) \quad (1,1,0,1,1)
  \]
Computation of Setwise Maximin Utility

"Master" problem

\[
\begin{align*}
\max & \quad \delta \\
\text{s.t.} & \quad \delta \leq \sum_{1 \leq j \leq k} v^j_w \\
& \quad \forall w \in \text{GEN} \\
& \quad v^j_w \leq w \cdot x^j \\
& \quad \forall j \leq k, w \in \text{GEN} \\
& \quad v^j_w \leq w^T I^j_w \\
& \quad \forall j \leq k, w \in \text{GEN} \\
& \quad \sum_{1 \leq j \leq k} I^j_w = 1 \\
& \quad \forall w \in \text{GEN} \\
& \quad I^j_w \in \{0, 1\} \\
& \quad \forall j \leq k, w \in \text{GEN}
\end{align*}
\]

Constraint Generation

(add the w to GEN)

(calculate minimum utility of every item in current best set)

"Slave" problems

\[
\begin{align*}
\min & \quad w \cdot x = \sum_{1 \leq i \leq n} x_i \cdot w_i \\
\text{s.t.} & \quad \text{Constraints}(W) \\
& \quad w_i \leq w_i \leq w_i^T \\
& \quad w_i \leq w_i \leq w_i \\
& \quad w_i \leq w_i \leq w_i
\end{align*}
\]
Theorem:
Optimal recommendation sets are myopically optimal choice queries

- In the maximin utility model

- However, this result is a bit less useful than in the other frameworks: the pessimistic fashion of maximin means that myopic elicitation is often not enough
The image shows a preference elicitation interface with two apartments, A and B, each with different features and rental prices. Apartment A is located in Toronto Central, is an apartment, has 1 bedroom, is unfurnished, has laundry available, parking available, a dishwasher, storage room, and is air-conditioned. Apartment B is located in Scarborough, is a house, has 1 bedroom, is unfurnished, has laundry available, parking available, a dishwasher, no storage room, and is air-conditioned. The rent for Apartment A is $900, and for Apartment B is $750. The text states: "Please carefully consider the two apartments and indicate which of the two you like more by clicking on it. Note that the features in black are the same for both apartments."
You are asked to decide whether the apartment on the left is "closer" in value to the TOP apartment or the BOTTOM apartment.

Features that are not shown (including price) are the same for all three apartments. Note that any features shown in grey are also the same for all apartments.

You have previously indicated that BOTTOM has the worst combination of features, and TOP has the best combination of features. On the scale from 0 to 100 (shown on the right of the bins) BOTTOM is at 0, and TOP is at 100. You should consider where the apartment in question falls on this scale. If its value is between 0 and the tip of the slider, please drag it to the bottom bin; otherwise, drag it to the top bin.
Please carefully consider the two apartments and indicate which of the two you like more by clicking on it.

Note that the features in black are the same for both apartments.

---

**CURRENT SOLUTION**

Current minimax regret: $666

---

**Rent: $900**

- **Toronto Central**
  - Apartment
  - 1 bedroom
  - Unfurnished
  - Laundry available
  - Parking available
  - Dishwasher
  - Storage room
  - Air-conditioned

**Recommended choice (ID 95)**

Utility $1,975

---

**Rent: $850**

- **Toronto West**
  - House
  - 1 bedroom
  - Unfurnished
  - Laundry not available
  - Parking available
  - No dishwasher
  - Storage room
  - Not air-conditioned

**"Adversary" choice (ID 54)**

Utility $1,700

---

Value regret $1,700 - $1,981 = $116

Regret is $116 + $900 - $850 = $666
Discussion: Utility-based Recommendations

- Utility-based Preference Elicitation for Conversational Systems
  - Sets can be used as both recommendation and elicitation
- **Theorem**: Optimal Recommendation sets are Myopically Optimal Choice Query sets
  - Bayesian / regret-based / maximin frameworks
  - Different response models in the probabilistic setting
- Efficient Algorithms suitable for large outcome spaces
- Other issues / Current and future works
  - Query optimization in very large configuration spaces
  - Optimization and elicitation with non-linear utility models
  - Structure learning of utility functions

Example 1

Only 2 possible utility functions and 3 available items.

<table>
<thead>
<tr>
<th></th>
<th>$U^1$</th>
<th>$U^2$</th>
<th>EU</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>B</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

| p(U)  | 0.5   | 0.5   |

**EUS**

<table>
<thead>
<tr>
<th></th>
<th>EUS noisless</th>
<th>EUS Constant 10%</th>
<th>EUS logistc</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>0.8</td>
<td>0.75</td>
<td>0.61</td>
</tr>
<tr>
<td>{A, C}</td>
<td>1</td>
<td>0.9</td>
<td>0.73</td>
</tr>
<tr>
<td>{B, C}</td>
<td>0.8</td>
<td>0.75</td>
<td>0.61</td>
</tr>
</tbody>
</table>
### Example 2

<table>
<thead>
<tr>
<th>set</th>
<th>EUS noisless</th>
<th>EPU noisless</th>
<th>T(.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>0.76</td>
<td>0.76</td>
<td>{A, B}</td>
</tr>
<tr>
<td>{A, C}</td>
<td>0.86</td>
<td>0.86</td>
<td>{A, C}</td>
</tr>
<tr>
<td>{B, C}</td>
<td>0.76</td>
<td>0.86</td>
<td>{A, C}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$U^1$</th>
<th>$U^2$</th>
<th>$U^3$</th>
<th>$EU$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>0</td>
<td>0.6</td>
<td>0.53</td>
</tr>
<tr>
<td>B</td>
<td>0.6</td>
<td>0.6</td>
<td>0.7</td>
<td>0.63</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.33</td>
</tr>
</tbody>
</table>

$p(U)$: 0.33 0.33 0.33
Preference Elicitation with Subjective Features

- Usually focuses on “catalog” attributes
  - Fix vocabulary of features: e.g., engine, size, color, fuel economy, ...

- Preferences are often most naturally expressed in terms of **subjective features**
  - Safe, cool, sporty, big, trendy, light, modern...
  - User-specific
A subjective feature is a user-defined concept

E.g. **SAFE CAR**

- I care about **Safety** and **power**
  - For me **SAFETY** := HasSideAirbags $\text{AND}$ isSUV

- I care about **Safety** and **Luggage space**
  - For me **SAFETY** := CrashTestRatings $> \text{Good}$ $\text{AND}$ easy to park

- I care about **safety with speed**, and **cabrio**
  - For me **SAFETY** := size=$\text{big}$ $\text{AND}$ brand=$\text{volvo}$ $\text{AND NOT (3 doors)}$
Feature Elicitation

- Subjective definition means we have to elicit them
- Feature elicitation vs. classical concept learning
  - *Learn just enough* about a concept in order to make a *good decision*
  - Near optimal recommendation with weak concept knowledge
  - Minimize user queries
Feature Elicitation

- Subjective definition means we have to elicit them

- Feature elicitation vs. classical concept learning
  - *Learn just enough* about a concept in order to make a *good decision*
  - Near optimal recommendation with weak concept knowledge
  - Minimize user queries

*Example:* preference for safe cars, BUT fuel economy more important.
Feature Elicitation

- Subjective definition means we have to elicit them
- Feature elicitation vs. classical concept learning
  - Learn just enough about a concept in order to make a good decision
  - Near optimal recommendation with weak concept knowledge
  - Minimize user queries

Example: preference for safe cars, BUT fuel economy more important.

If all “safe cars” have poor fuel economy, it is not worth continuing to learn more about safety!
Joint Elicitation with Minimax Regret

Simultaneous elicitation of user **features** and **utility**

- Doing one “completely” followed by other is wasteful
- Optimal or near-optimal recommendation is possible with little concept and utility information

Contributions

1) Define a model that allows **simultaneous** elicitation of user features and utility
2) **Minimax regret optimization** in presence of both utility and feature uncertainty, providing robust recommendations
3) Several heuristic techniques for eliciting concepts and utility that reduce **regret** quickly
Version Space Example

Most specific concepts

Most generic concepts

A

B

Not A

Not B

A and B

not A and B

A and not B

not A and not B

nil
Abstract Model

- Product space $X \subseteq \text{Dom}\{X_1 \ldots X_n\}$
  - Utility $u(x; w, c)$ function of unknown weights and feature; weight $w$ reflects tradeoffs between features
  - Concept $c(x)$ drawn from some hypothesis space $H$
  - Bonus weight $w_b$: additional utility for an $x$ satisfying $c(x)$
  - Goal: recommend product with highest utility

$$u(x; w, c) = w x + w_b c(x) \quad \text{with } w \text{ in } W, \ c \text{ in } V \ (\text{unknown})$$

$(V, W)$ consistent with prior knowledge
**Query Types**

Which one do you prefer?

- $X_1 \land \neg X_2 \land ..$
- $\neg X_1 \land \neg X_2 \land ..$

Is this car SAFE?

- $X_1 \land \neg X_2 \land ..$
- $\neg X_1 \land \neg X_2 \land ..$

**Comparison queries**

Answers impose constraints on $W$ and $V$

\[
u( X_1 \land \neg X_2 \land ..) > u( \neg X_1 \land \neg X_2 \land ..)
\]

**Membership queries**

Answers impose constraints on $V$ only

\[
c( X_1 \land \neg X_2 \land ..) == True
\]

\[
c(X_1 \text{ and not } X_2)=0
\]
Conditional Constraints

- When user prefers configuration $x$ to configuration $y$
  - Relate the difference in reward to whether either or both satisfy the unknown concept
  - No feature uncertainty: linear constraint $wx > wy$
  - Unknown membership: conditional constraints

$$
wx - wy > 0 \text{ if } c(x), c(y)
$$

$$
wx + p - wy > 0 \text{ if } c(x), \neg c(y)
$$

$$
wz - wy - p > 0 \text{ if } \neg c(x), c(y)
$$

$$
wz - wy > 0 \text{ if } \neg c(x), \neg c(y)
$$

- can be linearized and encoded in a MIP; for example:

$$
wx + b - wy > \left[ \sum_{j \leq n} I(\neg x[j]) + (1 - I(\neg y[k])) \right] \Delta \quad \forall k \leq n
$$
Minimax Regret over Concepts and Utility

Let \((W,V)\) be current utility and version space

\(x \text{ in } X\)
Minimax Regret over Concepts and Utility

Let \((W, V)\) be current utility and version space.
Minimax Regret over Concepts and Utility

Let \((W, V)\) be current utility and version space

\[
MR(x; W, V) = \max_{w \in W} \max_{c \in V} \max_{x^a \in X} u(x^a; w, c) - u(x; w, c)
\]

\[
MMR(W, V) = \min_{x \in X} MR(x; W, V)
\]

Current solution \((x^*, x^a, w, c)\)
Computing MMR

\[ \min \delta \]

\[ \delta > \text{actual regret for any possible adversary's choice} \]

s.t. constraints encoding concept memberships
Computing MMR

\[
\min \delta \\
\delta > \text{actual regret for any adversary's choice in } \text{GEN}
\]

s.t. constraints encoding concept memberships
Constraint Generation

- **REPEAT**
  - Solve *minimax regret* problem with a subset $GEN$ of $(W, V)$
    - The adversary's hands are tied to choose $(w, c)$ from $GEN$
    - $LB$ of minimax regret
  - Solve *max-regret* $MR(x)$ subproblem
    - $UB$ of minimax regret
    - Add the adversarial choice to $GEN$
  - Terminate **WHEN** $UB = LB$
# Max Regret

- Maximization sub-problem \( MR(x^*; V, W) \)

<table>
<thead>
<tr>
<th>Max</th>
<th>utility(adv) – utility(player's choice)</th>
</tr>
</thead>
<tbody>
<tr>
<td>s.t.</td>
<td>Adversary get <em>utility bonus</em> when concept is satisfied</td>
</tr>
<tr>
<td></td>
<td>Player gets <em>utility bonus</em> when concept is satisfied</td>
</tr>
<tr>
<td></td>
<td>Concept must be consistent with <em>positive</em> and <em>negative</em> examples (query answers)</td>
</tr>
<tr>
<td></td>
<td>Constraints between variables representing utilities and integer variables representing choices</td>
</tr>
<tr>
<td></td>
<td><em>Feasible</em> outcome and <em>feasible</em> utility weight</td>
</tr>
</tbody>
</table>
Elicitation

**Aim**: reduce *minimax regret* quickly
- Empirically, *actual regret* also reduces quickly

**Strategies**
1) Which type of query to ask
2) What to ask
CSS Comparisons

- Current solution \((x^*, x^a, w, c)\) immediately suggest query
  - Ask to compare \(x^*\) and \(x^a\)

C-CSS
(Comparison based on Current Solution Strategy)

Which one do you prefer?
Strategies for Concept Learning

```
Is this a safe car?''
```

Several strategies using membership queries:

1. **Halving**: aims to learn concept directly
   - "Random" query $x$ until positive response; then refine (unique) most specific concept in $V$ (negate one literal at a time)

2. **Current Solution (M-CSS)**: tackle regret directly
   - If $c(x^*)$ and $c(x^a)$ → query $x^a$ (unless certain)
   - If $c(x^a)$, $\neg c(x^*)$ → query $x^*$ (unless certain)
   - If $\neg c(x^*), \neg c(x^a)$ → query $x^a$ (unless certain)

Several variants show modest improvements
Which Type of Query to Ask?

- Given the current solution \((x^*, x^a, w, c)\)
- MaxRegret = RewardRegret (RR) + ConceptRegret (CR)

\[
\begin{align*}
\text{Interleaved strategies (I)} & \text{ asks comparison query when } RR > CR \\
\text{Phased Strategies (Ph)} & \text{ always ask membership when } CR > 0 \\
\text{Combined comparison-membership query (CCM):} & \text{ asks both comparison and membership queries about } x^* \text{ and } x^a \\
& \text{In general, counts as 3 queries}
\end{align*}
\]
Large concepts (defined over 10 attributes)

*Interleaved* elicitation strategies are better off than *phased* strategies

Our CSS-based heuristics better than *Halving-based* strategies
20 binary attributes, ~2100 feasible configurations

- Randomly generated utility functions and a subjective feature
- Conjunctive concepts defined over 5 attributes (avg 3.33 literals)

Interleaved elicitation strategies better off than myopic optimal

- True optimal product is found ~30 queries
Discussion On Feature Elicitation

- Feature elicitation under utility uncertainty
- Minimax regret optimization using mixed-integer programming and constraint generation
- Our interleaved approach is especially effective
  - Regret is used in order to choose among the different queries
  - Near-optimal recommendations with partial knowledge

Relations to Active Learning

Future Works
- Real valued domains, multivalued features and concepts
- Other concept/hypothesis spaces
- Noisy responses and probabilistic approaches
- UI, user studies
Conclusions

- Preference-handling systems: design issues
- Psychological biases in decision making and preference elicitation
- Preference elicitation with principled decision theory techniques
- Recommendation sets and optimal queries: bayesian framework, regret-based and maximin frameworks
- Subjective features

Acknowledgements: works done in collaboration with Craig Boutilier (utility-based recommendations), Boi Faltings, Pearl Pu (critiquing systems) and others
References 1

General on Preferences in AI


SmartClient


Teaching Salesman


Supple


PexA

Biases in Decision-making

- Kahneman, Daniel; Paul Slovic, Amos Tversky (1982), Judgment under Uncertainty: Heuristics and Biases, Cambridge University Press,
- Gilovich, Thomas; Dale Griffin, Daniel Kahneman (2002), Heuristics and biases: The psychology of intuitive judgment, Cambridge University Press

Foundations of Preference Reasoning

Example-critiquing

References 4

Minimax Regret

- Darius Braziunas, Craig Boutilier: Minimax regret based elicitation of generalized additive utilities. UAI 2007: 25-32

UTPREF (Decision-support system based on Regret)

- Darius Braziunas, Craig Boutilier: Assessing regret-based preference elicitation with the UTPREF recommendation system. ACM Conference on Electronic Commerce 2010: 219-228

Bayesian elicitation

References 5

Recommender Sets


Feature Elicitation

- Craig Boutilier, Kevin Regan, Paolo Viappiani: Online feature elicitation in interactive optimization. ICML 2009
- Craig Boutilier, Kevin Regan, Paolo Viappiani: Simultaneous Elicitation of Preference Features and Utility. AAAI 2010