

Paolo Viappiani and Craig Boutilier

Optimal Bayesian Recommendation Sets and Myopically Optimal Choice Query Sets (W47)

- Bayesian approach to adaptive utility elicitation
 - Underlying decision problem with structured utility (as in MAUT)
 - Uncertain utility

- Probabilistic belief about utility parameters

- Ask queries, observe answers, update belief using Bayes

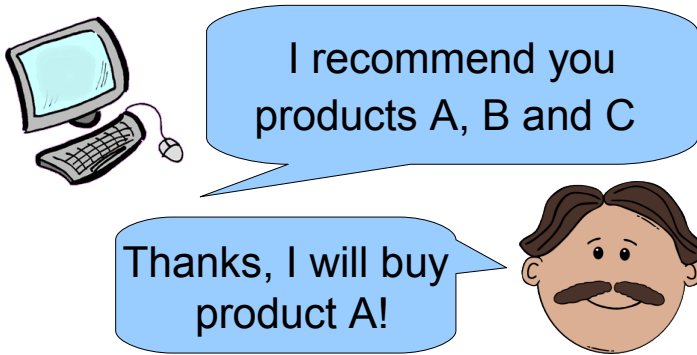


- Make a recommendation: show product that maximizes *expected utility*
- Natural criterion for queries: *Expected Value of Information (EVOI)*
- Because of complexity, most approaches use heuristics to select queries with no theoretical guarantees

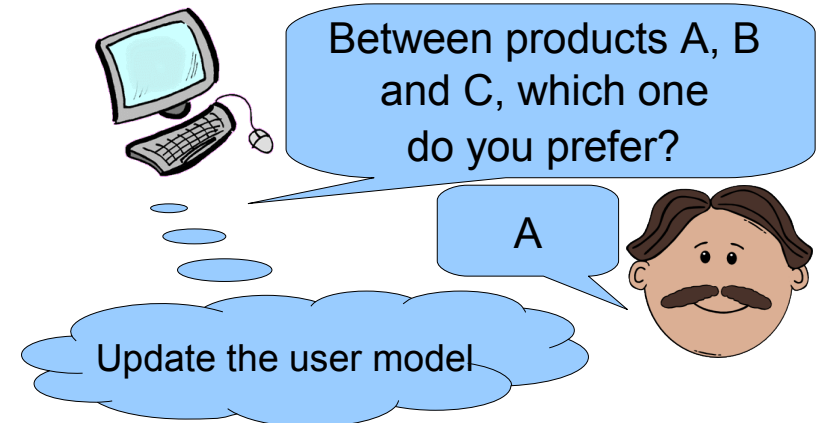
Paper contribution: Bayesian set-based recommendations and how they offer optimal or near-optimal EVOI choice queries

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

Online recommender systems:
options shown with dual goal of recommendation and elicitation

Laptop Features

Brand: Apple
ProcessorType: Core Duo
ProcessorSpeed(GHz): 1.83
ScreenSize(inches): 15.4
Memory(MB): 1024
HardDriveCapacity(GB): 100
Weight(lbs): 5.5
OperatingSystem: MacOS X 10
BatteryLife(hours): 5.6
Price(\$): 2199

We recommend this laptop for you

Price: 2199 USD
1799.2 EUR
2748.75 CHF

Main Features:

- ProcessorType Core Duo
- ProcessorSpeed(GHz) 1.83
- ScreenSize(inches) 15.4
- Memory(MB) 1024
- HardDriveCapacity(GB) 100
- Weight(lbs) 5.5
- OperatingSystem MacOS X 10.4
- BatteryLife(hours) 5.6

Product Description:

You've seen improvements in notebook performance before - but never on this scale. The Intel Core Duo powering MacBook Pro is actually two processors built into a single chip. This, combined with myriad other engineering leaps, boosts performance up to four times higher than the PowerBook G4. With this awesome power, it's a breeze to render complex 3D models, enjoy smooth playback of HD video, or hold a four-way video conference.

Not satisfied with the result? you may select other recommendations listed below

- Faster CPU**
Get with More Expensive
[See product details](#) [I like this](#)
- Faster CPU and Cheaper**
Get with Less Memory and Smaller Hard-Disk
[See product details](#) [I like this](#)
- Lighter and Cheaper**
Get with Different Type of CPU, Slower CPU, Smaller Screen, Less Memory, Smaller Hard-Disk and Shorter Battery Life
[See product details](#) [I like this](#)
- Larger Screen and Larger Hard-Disk**
Get with Different Type of CPU, Slower CPU, Less Memory, Heavier, Shorter Battery Life and More Expensive
[See product details](#) [I like this](#)
- Lighter and Longer Battery Life**
Get with Different Brand, Slower CPU, Smaller Screen, Different OS and More Expensive
[See product details](#) [I like this](#)

[Next Step](#)

[Pu et. al.,08]

- 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.

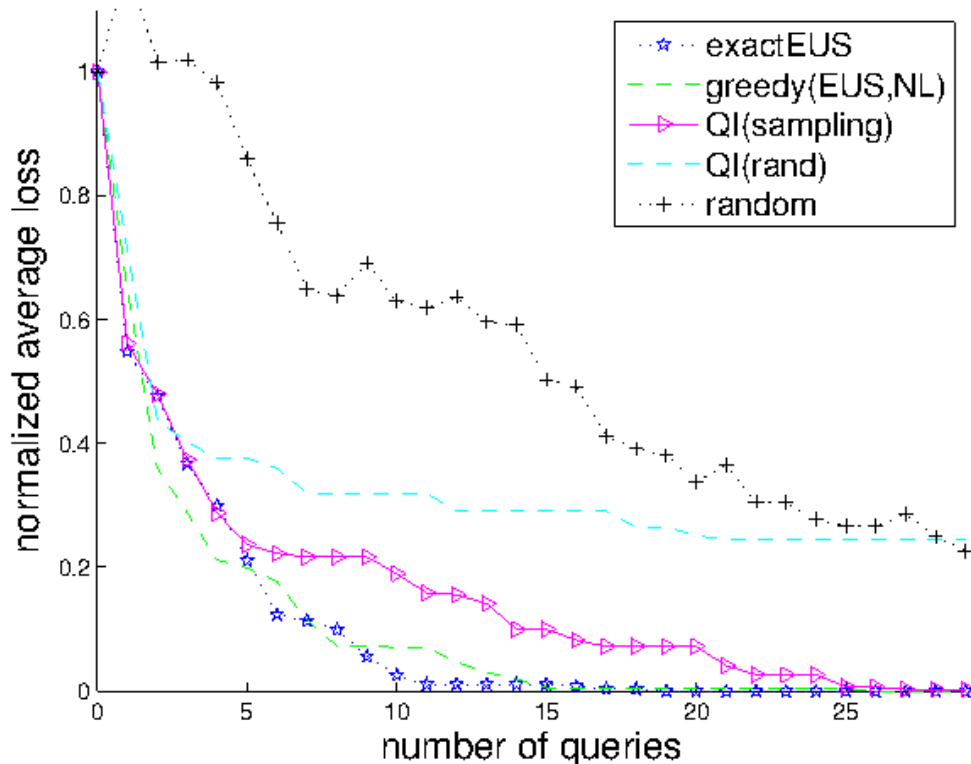
Theorem: Optimal recommendation sets are near-optimal queries under the logistic noise model

- We provide the expression for the worst-case loss Δ_{\max} (surprisingly small)
- Also, the optimal query assuming *noiseless* responses is a near-optimal query under logistic *noise*

Algorithms

Consequences of our theoretical results: efficient algorithms to generate choice queries

- Optimizing a recommendation set is simpler and *submodular*
- Approximated strategies with worst-case guarantees
- *Noiseless* optimization quite effective in *noisy* settings
- Query Iteration strategy particularly efficient for large datasets



| Computation time | Dataset 1 Size=187 | Dataset 2 Size=506 |
|--------------------------------|-----------------------|-----------------------|
| Exact EVOI | 1815s | ~2 weeks |
| Exact EUS | 405s | ~2 hours |
| Greedy with lazy evaluation | 1.02s | 0.93 s |
| Query Iteration (local search) | 0.15s | 0.05 s |