Evaluation Methods and Challenges
Evaluation Methods

• Ideal method
  – Experimental Design: Run side-by-side experiments on a small fraction of randomly selected traffic with new method (treatment) and status quo (control)
  – Limitation
    • Often expensive and difficult to test large number of methods

• Problem: How do we evaluate methods offline on logged data?
  – Goal: To maximize clicks/revenue and not prediction accuracy on the entire system. Cost of predictive inaccuracy for different instances vary.
    • E.g. 100% error on a low CTR article may not matter much because it always co-occurs with a high CTR article that is predicted accurately
Usual Metrics

- Predictive accuracy
  - Root Mean Squared Error (RMSE)
  - Mean Absolute Error (MAE)
  - Area under the Curve, ROC

- Other rank based measures based on retrieval accuracy for top-k
  - Recall in test data
    - What Fraction of items that user actually liked in the test data were among the top-k recommended by the algorithm (fraction of hits, e.g. Karypsis, CIKM 2001)

- One flaw in several papers
  - Training and test split are not based on time.
    - Information leakage, results not valid
    - Even in Netflix, this is the case to some extent
      - Time split per user, not per event. For instance, information will leak if models are based on user-user similarity.
Metrics continued..

• Recall per event based on Replay-Match method
  – Fraction of clicked events where the top recommended item matches the clicked one.

• This is good if logged data collected from a randomized serving scheme, with biased data this will be a problem
  – We will be inventing algorithms that provide recommendations that are similar to the current one
    • No reward for novel recommendations
Details on Replay-Match method (Li, Langford, et al)

- x: feature vector for a visit
- r = [r_1, r_2, ..., r_K]: reward vector for the K items in inventory
- h(x): recommendation algorithm to be evaluated
- Goal: Estimate expected reward for h(x)

\[ E_{(x, r)} \sim P \left[ \sum_i \Pr(h(x) = i) \cdot r_i \right] \]

- s(x): recommendation scheme that generated logged-data
- x_1,..,x_T: visits in the logged data
- r_{ti}: reward for visit t, where i = s(x_t)
Replay-Match continued

- Estimator

\[
\frac{1}{T} \sum_t \sum_i I(h(x_t) = i \text{ and } s(x_t) = i) \cdot r_{ti} \cdot \alpha_t
\]

- If importance weights and \((x_t, r_t) \text{ iid } \sim \mathcal{P}\). 

  - It can be shown estimator is unbiased

- E.g. if \(s(x)\) is random serving scheme, importance weights are uniform over the item set

- If \(s(x)\) is not random, importance weights have to be estimated through a model
Challenges
Recall: Some examples

• Simple version
  – I have an important module on my page, content inventory is obtained from a third party source which is further refined through editorial oversight. Can I algorithmically recommend content on this module? I want to drive up total CTR on this module

• More advanced
  – I got X% lift in CTR. But I have additional information on other downstream utilities (e.g. dwell time). Can I increase downstream utility without losing too many clicks?

• Highly advanced
  – There are multiple modules running on my website. How do I take a holistic approach and perform a simultaneous optimization?
For the simple version

- Multi-position optimization
  - Explore/exploit, optimal subset selection

- Explore/Exploit strategies for large content pool and high dimensional problems
  - Some work on hierarchical bandits but more needs to be done

- Better offline evaluation strategies
  - This is important for progress in this area

- Constructing user profiles from multiple sources with less than full coverage

- Content understanding

- Metrics to measure user engagement (other than CTR)
Other problems

• Whole page optimization
  – Challenging, open area

• Content programming
  – How should we generate content to enhance our inventory?

• Incentivizing User generated content

• Incorporating Social information for better recommendation