Greedy Optimized Multileaving for Personalization
Kojiro Iizuka¹, Takeshi Yoneda¹, Seki Yoshifumi¹
¹Gunosy Inc.
(kojiro.iizuka, takeshi.yoneda, yoshifumi.seki)@gunosy.com

What is Multileaving?
- A method that evaluates multiple rankings efficiently using user click feedback (Schuth, 2014).

Challenges
- Achieving high sensitivity with low computation cost for the rankings, which differ for each user and each time.
- Ensuring stability over the number of rankers and the ranking length.

Contributions
- New Problems: Clarifies the challenges of applying the multileaving method to personalized rankings;
- Algorithm: Proposes the greedy optimized multileaving (GOM) method with the personalization credit function to solve these problems; and
- Stability and Sensitivity: Confirms the stability and sensitivity of GOM throughout offline and online experiments.

Greedy Optimized Multileaving

Optimized Multileaving (Radlinski 2013)
\[ \min_{\{\lambda_i\}} \sum_{i=1}^{n} \lambda_i + \sum_{i=1}^{n} \beta_i \]
subject to
\[ \sum_{i=1}^{n} \lambda_i = 1, \quad 0 \leq \beta_i \leq 1 \quad (k=1,\ldots,m). \]

Using the previous work’s formula has high computational cost to generate multileaved rankings.

Re-formulation for personalized settings

Greedy Optimized Multileaving (proposed)
\[ \arg \min_{k} \sum_{i=1}^{n} \lambda_i + \sigma_i^2 \]
subject to
\[ \sum_{i=1}^{n} \lambda_i = 1, \quad 0 \leq \beta_i \leq 1 \quad (k=1,\ldots,m). \]

- Personalized ranking is shown to the user only once,
- We can consider ranking output probability as a one-hot vector.

We can solve this problem by greedy strategy.

Inverse credit function
\[ \delta(O_{i,j}) = \frac{1}{\text{rank}(O_{i,j})}. \]

The deeper the click position, the smaller the credit.

Personalized credit function
\[ \delta(O_{i,j}) = -|\sum_{i=1}^{n} \lambda_i \cdot \text{rank}(O_{i,j})|. \]

The credit can be calculated without position noise.

Experiment

Do GOM’s evaluation coincide with A/B Testing? How is the efficiency?

Table 1: Differences between the sum of credits. The values of GOM-P and TDM were all positive, while some values of GOM-I (written in blue) were negative. The GOM-P and TDM’s results were consistent with previous CTR results of A/B testing (algo-A < algo-B < algo-C < algo-D < algo-E), but GOM-I was inconsistent. This means that the inverse credit function which was often used in previous studies is not appropriate for ranking.

<table>
<thead>
<tr>
<th></th>
<th>GOM-I</th>
<th>GOM-P</th>
<th>TDM</th>
</tr>
</thead>
<tbody>
<tr>
<td>algo-C</td>
<td>2.743</td>
<td>-3.072</td>
<td>-1.432</td>
</tr>
<tr>
<td>algo-D</td>
<td>2.743</td>
<td>-3.072</td>
<td>-1.432</td>
</tr>
<tr>
<td>algo-E</td>
<td>2.743</td>
<td>-3.072</td>
<td>-1.432</td>
</tr>
</tbody>
</table>

These experiments assumed a practical environment that requires a low computation cost to generate rankings in real-time; therefore, we did not use GOM.

Conclusion and Future Work

- We proposed GOM for personalized rankings that require a high degree of freshness, many hyperparameters, and a long ranking length for a rich user experience.
- We confirmed GOM’s stability and sensitivity by online and offline experiments.
- We will try to apply GOM’s feedback schema to speed up online learning to rank.