Recommender Systems

Multi-armed bandits

http://de.fotolia.com/
Agenda

• Problems
  • Idea
  • Theory and Practice

• Algorithms
  • Ranked Explore and Commit
  • Ranked Bandits Algorithm

• Dueling Bandits Problem
Problem

Find algorithms for learning to rank web documents

Attention: Web search engines are no Recommender Systems. But they also use learning to rank algorithm and are simple examples to show how the following learning algorithms work.

Algorithms assume a document’s relevance is independent of other documents

In Recommender Systems “relevance” means “click rate” of a document
Problem

Goal: maximize the probability that a relevant document is found in the top k positions of a ranking
Idea

1. Learning from usage data

- click rate per document
- visiting time
- rank of clicked document

- larger quantities
- lower cost
- constantly updated
2. Online learning approach

- exploration vs. exploitation
- minimize the total number of poor rankings

http://de.fotolia.com/
3. Relevance of documents

• example: the query „jaguar“
Theory and Practice

Theory

• documents should be ranked by their probability of relevance to the query
• each document has a single relevance score

Practice

• given fixed query, the same document can have different relevance to different users
• users satisfied with finding a small number of relevant documents
• documents also depend on other documents ranked higher
Terms and definition

Query $q$

Document $d$

Learning parameter $\theta$

Scoring function $f(q, d_i, \theta)$
Algorithm 1

Ranked Explore and Commit
Algorithm 1: Ranked Explore and Commit

Goal: maximize the probability that a relevant document is found in the top $k$ positions of a ranking

Input: $k = 3$, $\varepsilon = \delta = 1.5$

$$x = \left\lfloor 2 \cdot \frac{k}{\varepsilon} \cdot \log \left( 2 \cdot \frac{k}{\delta} \right) \right\rfloor = 2$$
Algorithm 1: Ranked Explore and Commit

1. Choose some parameters $\varepsilon$, $\delta$ and an initial arbitrarily chosen set of $k$ documents

2. For each rank
   a) assign each document to that rank for specified interval and record clicks
   b) increment probability of assigning document that rank if it is chosen by user
   c) choose document with max probability and commit it to the rank

3. Display ordered set of $k$ documents

see: http://courses.cms.caltech.edu/cs101.2/slides/cs101.2-14-diverse-rankings.pdf
REC: Rank1, Iteration 1

\[ d = \{ \text{d1, B/b1} \} \]

\[ B = \{ \text{d1, B/b1} \} \]

\[ d_1 + B/b_1 \]
REC: Rank1, Iteration 1

d = \{ \text{monkey}, \text{frog}, \text{elephant}, \text{lion} \} \quad B = \{ \text{monkey}, \text{elephant}, \text{lion} \}

Click!
REC: Rank1, Iteration 1

\[ d = \{ \text{monkey, frog, elephant, lion} \} \]

\[ B = \{ \text{monkey, elephant, lion} \} \]

\[ d^2 + B/b1 \]
REC: Rank1, Iteration 1

d = \{ \text{monkey, frog, elephant, lion} \} \quad B = \{ \text{monkey, elephant, lion} \}

d_3 + B/b_1
REC: Rank1, Iteration 1

\[ d = \{ \text{monkey, frog, elephant, lion} \} \quad B = \{ \text{monkey, elephant, lion} \} \]

\[ d_4 + B/b_1 \]

Click!
REC: Rank1, Iteration 2

d = \{ \text{monkey}, \text{frog}, \text{elephant}, \text{lion} \} \quad B = \{ \text{monkey}, \text{elephant}, \text{lion} \}

d_1 + B/b_1
$d_2 + \frac{B}{b_1}$
REC: Rank1, Iteration 2

d = \{ \}
B = \{ \}

d3 + B/b1
REC: Rank1, Iteration 2

d = \{ \text{monkey, frog, elephant, lion} \} \quad B = \{ \text{monkey, elephant, lion} \}

d_4 + B/b1
REC: Rank1, Iteration 2

d = \{ \text{monkey, frog, elephant, lion} \} \quad B = \{ \text{highlighted monkey, elephant, lion} \}
REC: Rank2, Iteration 1

\[ d = \{ \text{monkey, frog, elephant, lion} \} \]

\[ B = \{ \text{red monkey, elephant, lion} \} \]

\[ d_1 + B/b_2 \]
REC: Rank2, Iteration 1

\[ d = \{ \text{monkey, frog, elephant} \} \quad B = \{ \text{monkey, elephant} \} \]

\[ d^2 + B/b^2 \]
REC: Rank2, Iteration 1

d = \{ \}

B = \{ \}

\[ d^3 + \frac{B}{b^2} \]
REC: Rank2, Iteration 1

d = \{ \text{monkey, frog, elephant, lion} \} \quad B = \{ \text{monkey, elephant, lion} \}

d_4 + B/b^2
REC: Rank2, Iteration 2

After 2. iteration
Disadvantages:

- After each document is selected, this decision is never revisited!
  - user interests do not change over time
  - documents do not change over time

Solution:
- Ranked Bandits Algorithm
Algorithm 2

Ranked Bandits Algorithm
RBA: Ranked Bandits Algorithm

User $U =$

Documents $d =$

$B_{run} = \{ \}$
Algorithm 2: Ranked Bandits Algorithm

1. Initialize the k ‘bandit algorithms’ MAB1, MAB2,...,MABk

2. For each of the k slots:
   a) select document according to the bandit algorithm.
   b) if already previously chosen, select arbitrary document.

3. Display ordered set of k documents
   a) Assign reward to document if user clicked it and chosen as per the algorithm
   b) Assign penalty otherwise
   c) Update algorithm for the rank

see: http://courses.cms.caltech.edu/cs101.2/slides/cs101.2-14-diverse-rankings.pdf
RBA: Run 1

MAB1

MAB2

MAB3

B_1: { { } }
RBA: Run 1

B_1: 

\{ \}  \{ \{ \{ \} \} \}
RBA: Run 1

MAB1

MAB2

MAB3

B_1:  \{\text{monkey} \} \quad \{\text{monkey, elephant} \}
RBA: Run 1

B_1: {MAB1} {MAB2} {MAB3}
RBA: Run 1
RBA: Run 1

MAB1

MAB2

MAB3
RBA: Run 1 vs. Run 6

- after run 1:
  - MAB1
  - MAB2
  - MAB3

- (maybe) after run 6:
  - MAB1
  - MAB2
  - MAB3
RBA: Run 1 vs. Run 6

- after run 1:
  - MAB1: 1 0 0 0

- (maybe) after run 6:
  - MAB1: 2 1 3 1
  - MAB2: 0 0 1 0
  - MAB3: 0 0 0 0

Rank list:

- MAB1: ?
- MAB2: ?
- MAB3: ?
REC vs. RBA

• REC:
  user interest and documents do not change over time
  Payoff: \( \left(1 - \frac{1}{e} - \varepsilon\right)OPT - O\left(k^3 \cdot \frac{n}{\varepsilon^2} \cdot \ln\left(\frac{k}{\delta}\right)\right) \)

• RBA:
  documents and user interests changes over time
  Payoff: \( \left(1 - \frac{1}{e}\right)OPT - O\left(k \sqrt{T \cdot n \cdot \log(n)}\right) \)

OPT: maximal payoff when the click probabilities for all users and documents is known
(1 - 1/e)OPT is the best obtainable polynomial time approximation
Dueling Bandits Problem
Dueling Bandits Problem

- Learning from implicit feedback (user’s clicks)

- Learning using pairwise comparisons

- Online Framework: Dueling Bandits Problem
  - find a sequence of comparisons that has low regret.
  - find a close to optimal retrieval function and never show clearly bad results in the process
Dueling Bandits Problem

\[ f_1(u, q) \rightarrow r_1 \quad \text{and} \quad f_2(u, q) \rightarrow r_2 \]

Rank 1
Rank 2
...

Rank 1
Rank 2
...

Interleaving \((r_1, r_2)\)

\[(r_2 > r_1) \leftrightarrow \text{clicks}(r_2) > \text{clicks}(r_1)\]
Dueling Bandits Problem

• Comparisons between two points \( w \) and \( w' \) within a space \( W \)
• \( W \) contains the origin, is compact and convex
• \( W \) contained in a \( d \)-dimensional ball of radius \( R \)

\[ P(w > w') = \frac{1}{2} + \varepsilon(w, w') \quad \varepsilon(w, w') \in \left[-\frac{1}{2}, \frac{1}{2}\right] \]

\[ \varepsilon(w, w') = \text{distinguishability between } w \text{ and } w' \]
Dueling Bandits Problem

\[ \delta - \text{explore step size} \]
\[ \gamma - \text{exploit step size} \]

Current point

Losing candidate

Winning candidate
Thank you for your attention!
References


• Dueling Bandits Problem, http://dev.videolectures.net/icml09_yue_ioirsdbp/