

Personalize Expedia Hotel Searches –2nd Place

Jun Wang

Alexandros Kalousis



Outline

- Problem Setting
- Modelling
- Feature Engineering
- Conclusion and Future Work

Learning Problem

- Learning to rank hotels such that purchased and clicked hotels are ranked at top among all hotels associated with a search query.

Formally, let $\mathbf{x}_i \in \mathcal{X}$ be the i th search query,

$\mathcal{H}_i = \{h_{i1}, \dots, h_{im_i}\} \subset \mathcal{H}$ be the set of hotels associated with \mathbf{x}_i

$\mathbf{s}_i = (s_{i1}, \dots, s_{im_i})$ be their corresponding relevance scores.

We want to learn a ranking model

$$f : \mathbf{x}_i \times \mathcal{H}_i \rightarrow f(\mathbf{x}_i, \mathcal{H}_i) := \hat{\mathbf{s}}_i = (\hat{s}_{i1}, \dots, \hat{s}_{im_i})$$

such that the ranking order of predicted scores $\hat{\mathbf{s}}_i$ are exactly equal to the ranking order of \mathbf{s}_i .

$$x_i \in \mathcal{X}$$

$$\mathcal{H}_i = \{h_{i1}, \dots, h_{im_i}\}$$

The screenshot shows the Expedia website's hotel search interface. Key elements include:

- Navigation Bar:** Home, Vacation Packages, Hotels, Cars, Flights, Cruises, Things to Do, DEALS.
- PLAN YOUR TRIP ON EXPEDIA:** A section with radio buttons for Flight, Hotel (selected), Car, Activities, and Cruise. It also features a 'Flight + Hotel + Car' option and a 'Hotel + Car' option.
- Hotel Search Section:**
 - Find hotels near:** A dropdown menu with 'A city, airport or attraction' selected.
 - What City?:** A text input field with 'New York (and vicinity), New York, United States of America' entered.
 - Check-in:** A date picker showing '10/18/2013'.
 - Check-out:** A date picker showing '10/20/2013'.
 - Rooms:** A dropdown menu showing '1'.
 - Room 1:** A dropdown menu showing '2'.
 - Children:** A dropdown menu showing '0'.
 - Adults:** A dropdown menu showing '2'.
 - Children:** A dropdown menu showing '0'.
- Search Button:** A yellow button labeled 'SEARCH FOR HOTELS'.
- Best Price Guarantee:** A logo for the 'BEST PRICE GUARANTEE'.
- Ranking Model:** A large blue arrow pointing from the search interface to the hotel results, labeled 'Ranking Model'.

The screenshot shows three hotel listings from a search results page:

- Grand Hyatt New York** (★★★★★): 4.3 out of 5 (2700 reviews). Price: \$829 (avg/night) to \$479 (avg/night). Big Savings for Prime NYC Location. Book now for lowest rate. Located at Grand Central and walking distance to Times Square and 5th Ave shopping. **Sale • Advance Purchase Special - non-refundable**
- Wellington Hotel** (★★★): 3.7 out of 5 (3623 reviews). Price: \$349 (avg/night) to \$251 (avg/night). New York (Broadway - Times Square). 1-866-267-9053 • Expedia Rate. **Sale • Book Now & Save 20% Hurry! Offer ends in 35:25:50**
- New Yorker Hotel** (★★★★★): 4.1 out of 5 (5057 reviews). Price: \$369 (avg/night) to \$295 (avg/night). New York (Madison Square Garden). 1-866-272-4856 • Expedia Rate. **Free Cancellation**
- 4-Star Hotels from \$284 near New York**: Expedia Unpublished Rate Hotels. Deep discounts on quality hotels. Get the Deal now – and the Hotel Name after you book. Price: \$284 (4-Stars from).

Assumption

- The relevance score reflects user's rational behaviour based on the quality of hotels provided by Expedia.

$$\left\{ \begin{array}{ll} s_{ij} > s_{ik} & h_j \text{ is better than } h_k \text{ for user } i \\ s_{ij} = s_{ik} & h_j \text{ is equally good as } h_k \text{ for user } i \\ s_{ij} < s_{ik} & h_j \text{ is worse than } h_k \text{ for user } i \end{array} \right.$$

- Any kind of irrational behaviour of user is treated as **random noise** in the data

Outline

- Problem Setting
- **Modelling**
- Feature Engineering
- Conclusion and Future Work

Modelling Methodology

- Linear – difficult to handle categorical (discrete) features
 - Linear Regression
 - linear learning-to-rank model (RankSVM and SGD approach)
- Nonlinear
 - Gradient Boosted Trees
 - RandomForest
 - LambdaMART

LambdaMART

- LambdaMART is a learning to rank algorithm based on Multiple Additive Regression Tree (MART).
- It is nonlinear and computationally efficient.

$$\min_{\hat{s}_{ij}, \hat{s}_{ik}} \sum_i \sum_{jk \in \mathcal{H}_i} |\Delta Z_i^{jk}| \log(1 + e^{\sigma y_{ijk}(\hat{s}_{ij} - \hat{s}_{ik})})$$

where $|\Delta Z_i^{jk}| = |NDCG(\hat{s}_i, s_i) - NDCG(\hat{s}_i^{jk}, s_i)|$

$$y_{ijk} = \begin{cases} -1 & s_{ij} > s_{ik} \\ 0 & s_{ij} = s_{ik} \\ 1 & s_{ij} < s_{ik} \end{cases}$$

NDCG score of \hat{s}_i

NDCG score of \hat{s}_i , with j, k elements swapped

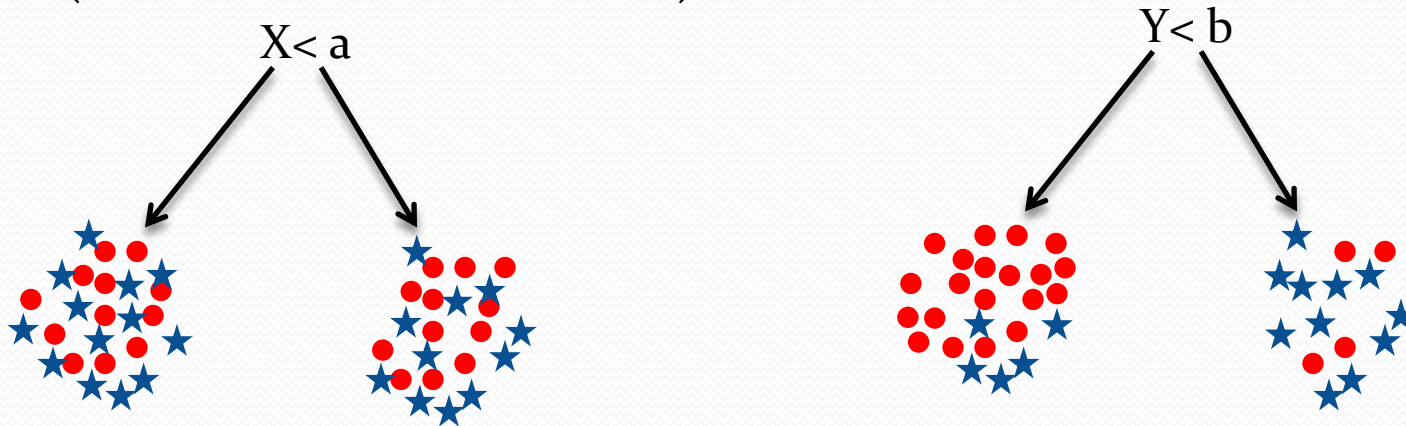
This objective is minimized by learning relevance score through MART.

Outline

- Problem Setting
- Modelling
- Feature Engineering
- Conclusion and Future Work

Feature Engineering (1)

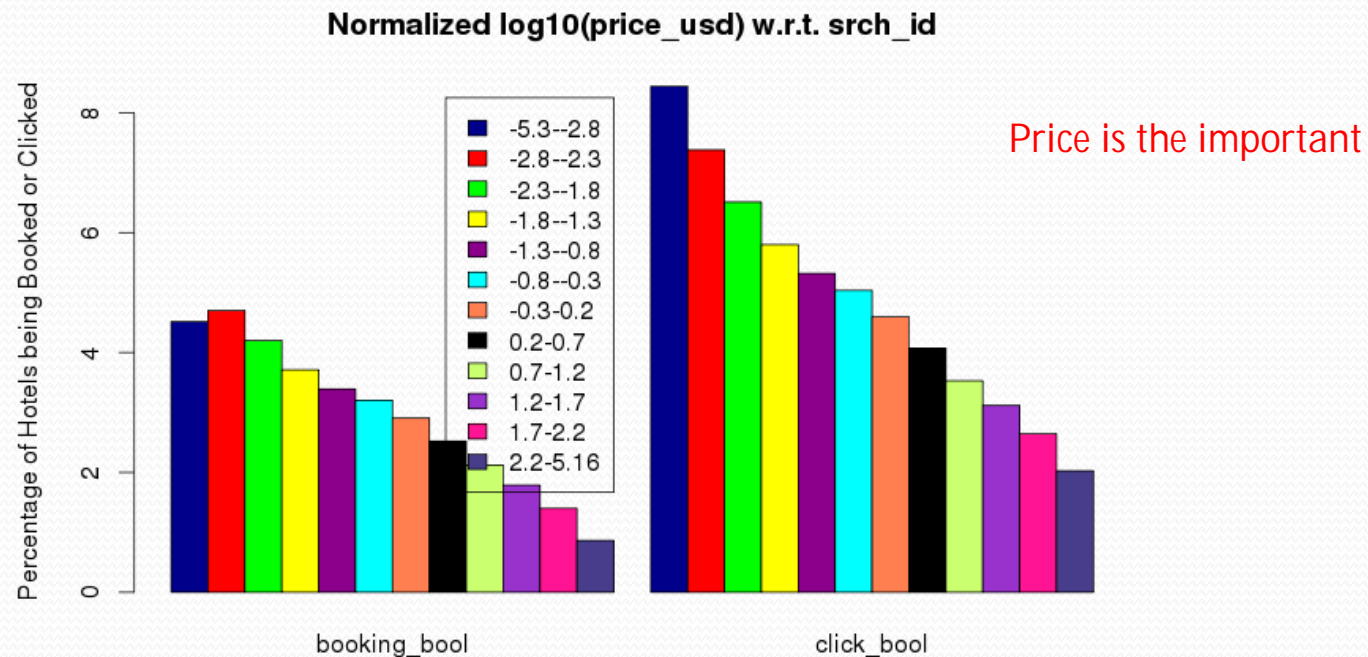
- What is a good representation for LambdaMART (tree-based classifier) ?



Feature Y is more discriminative than feature X for tree classifier

Feature Engineering (2)

- We want feature with **monotonic utility** w.r.t. target variable!



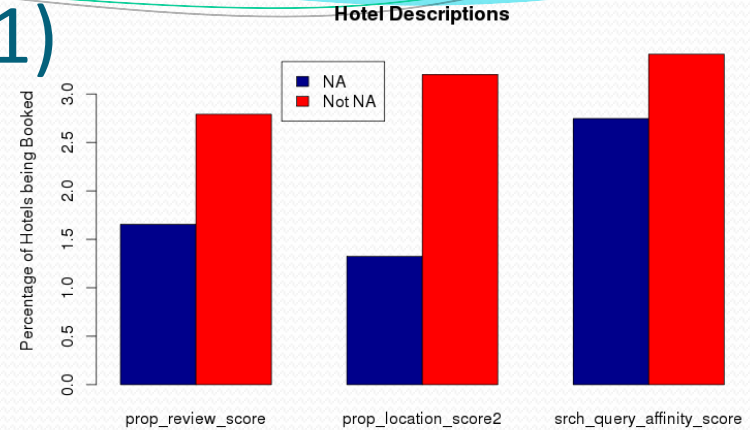
Feature Engineering Tasks

- Missing value estimation
 - On hotel descriptions
 - On user's historical data
 - On competitor descriptions
- Feature extraction
- Feature normalization

Missing Value Estimation (1)

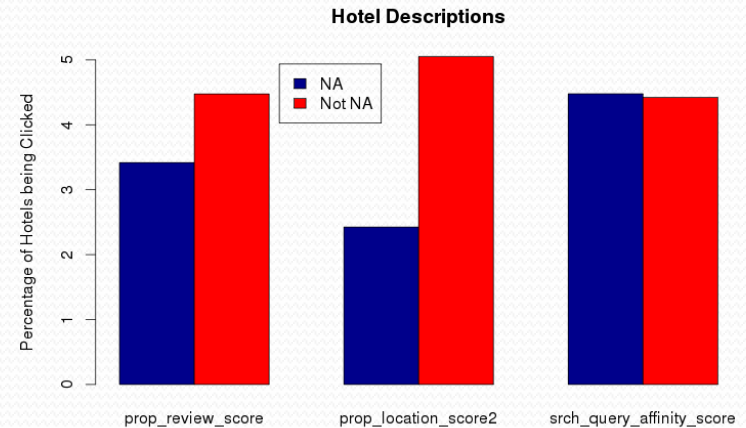
Hotel Descriptions

- User do not like to book and click hotels with missing values.



booking_bool

- Our solution: fill missing value of hotel description with worse case scenario.



click_bool

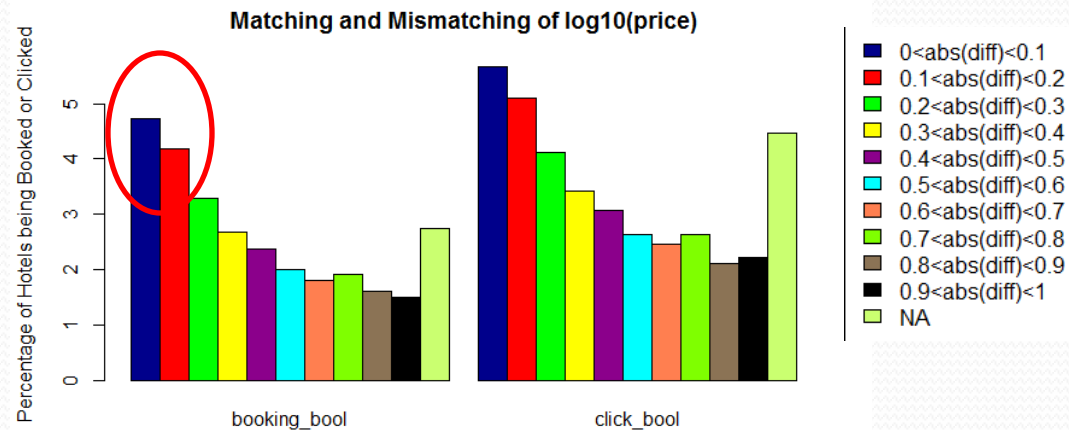
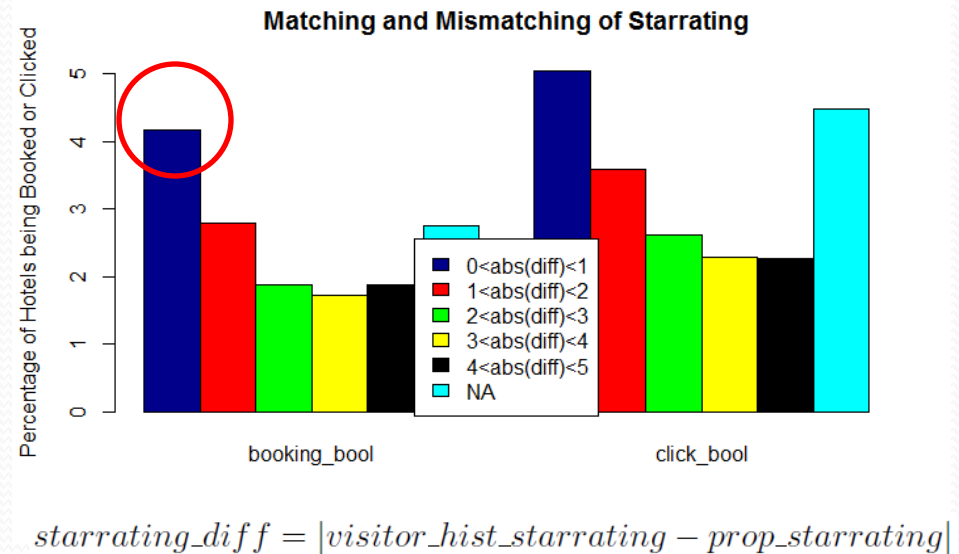
Missing Value Estimation (2)

User's Historical Data

- Approximately 95% of user's historical data are missing!!!
- No enough information to model user's historical data

Whether it's his favorite hotel??

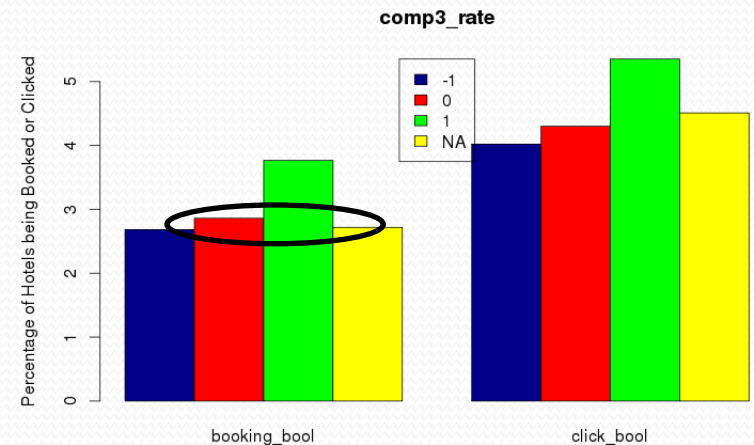
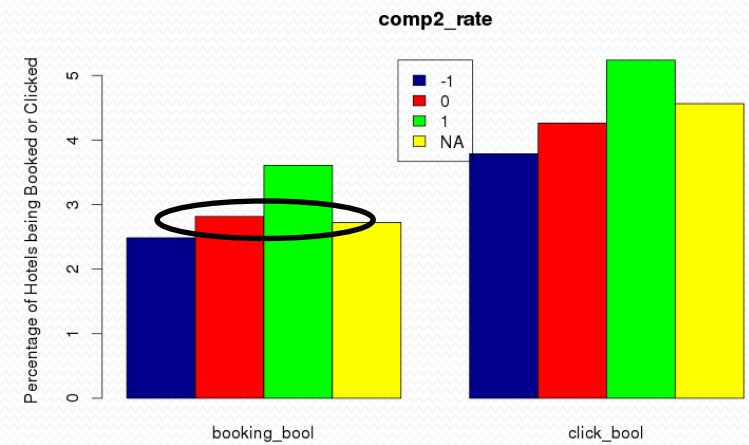
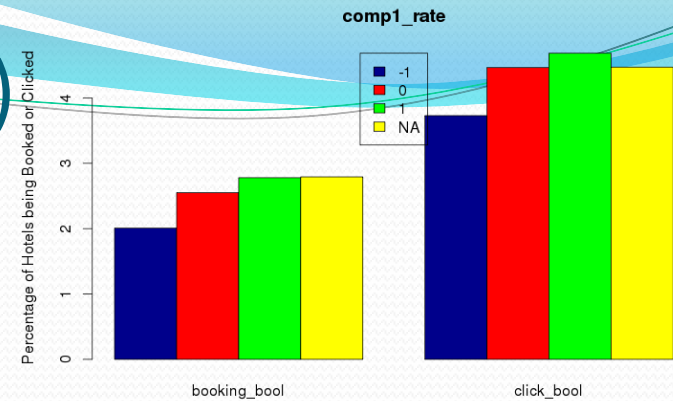
- Our solution: highlight the **matching** and (or) **mismatching** between historical data and the given hotel data



Missing Value Estimation (3)

Competitor Descriptions

- Missing value of competitor descriptions are all set to zero
- In total, there is no significant hotel price difference between Expedia and other competitors.



Feature Engineering Tasks

- Missing value estimation
- Feature extraction
 - Hotel quality estimation
 - Non-Monotonicity of Feature Utility
- Feature normalization

Feature Extraction (1)

- On user's historical data

$$starrating_diff = |visitor_hist_starrating - prop_starrating|$$

$$usd_diff = |visitor_hist_adr_usd - price_usd|$$

- On **hotel quality**

- The probability of each hotel being booked or clicked varies very much.
- This probability is estimated by

$$\frac{booking(prop_id)}{counting(prop_id)} \quad \text{and} \quad \frac{click(prop_id)}{counting(prop_id)}$$

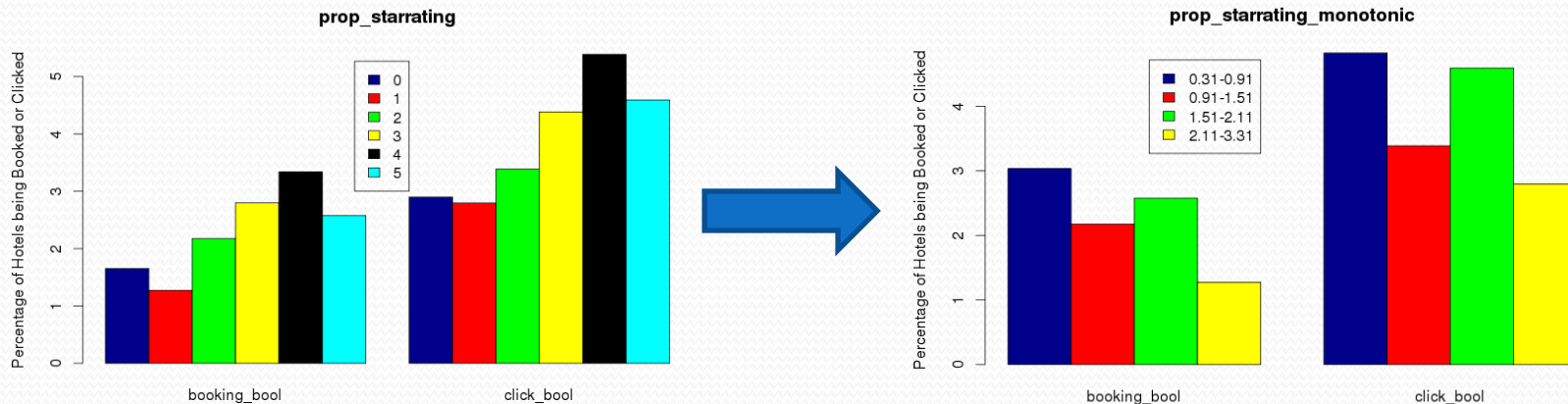
The number of times
that prop_id was booked

The number of times that prop_id
appeared in the data

Feature Extraction (2)

- On the **Non-Monotonicity of Feature Utility**
 - Some features have non-monotonicity utility functions.
 - E.g. `prop_starrating`

$$\text{prop_starrating_monotonic} = |\text{prop_starrating} - \text{mean}(\text{prop_starrating}[\text{booking_bool}])|$$



Feature Engineering Tasks

- Missing value estimation
- Feature extraction
- Feature normalization

Feature Normalization (1)

- Market, e.g. hotel price, varies in different cities, at different times.
- To build a good ranking model for all hotels worldwide, corresponding features should be normalized in an appropriate manner.
 - Removing scaling factors

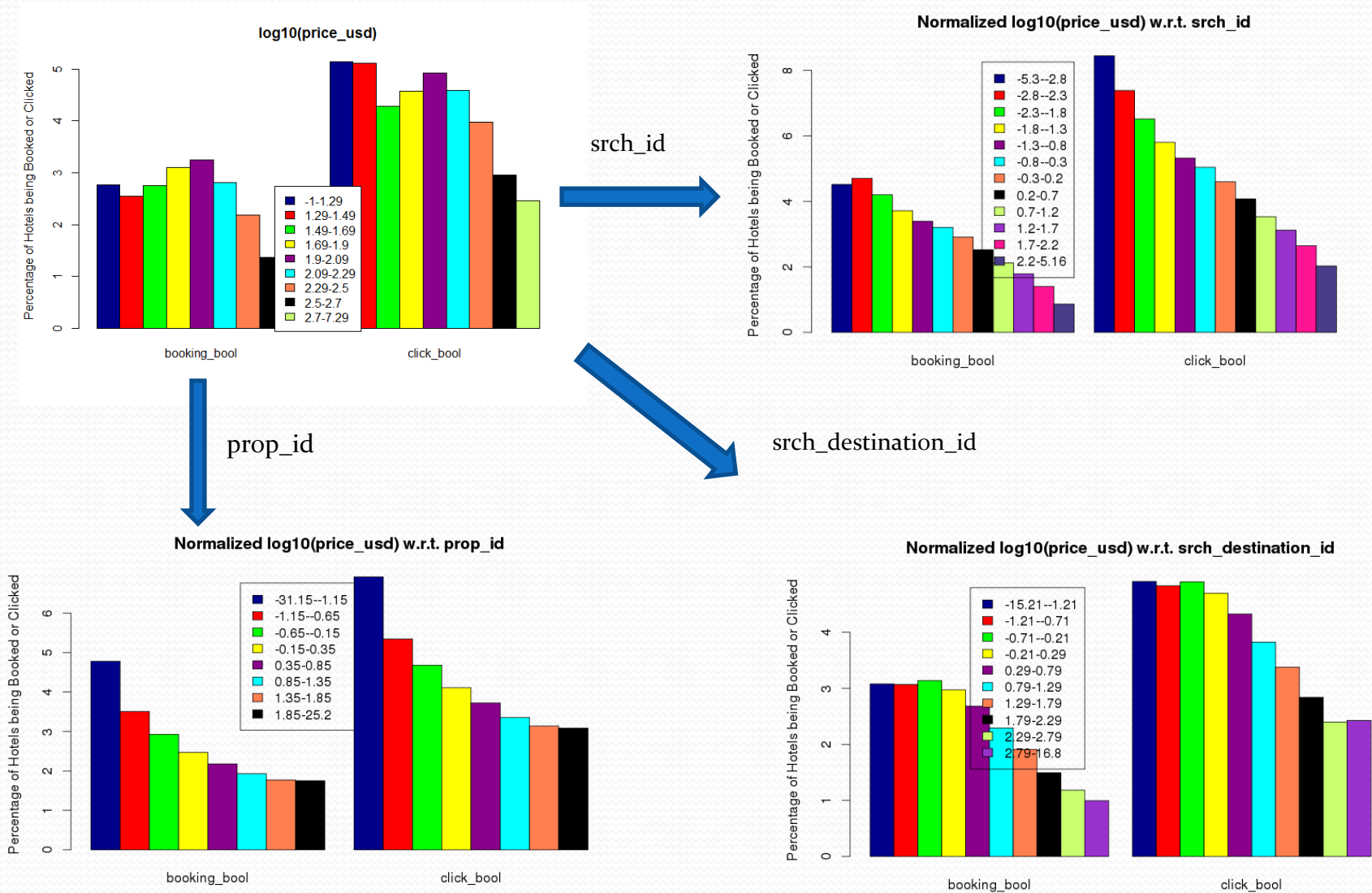
Feature Normalization (2)

- Our solution: normalize hotel and competitor descriptions with respect to different indicators

`srch_id, prop_id, month, srch_booking_window, srch_destination_id, prop_country_id`

- `srch_id`: compare the quality of hotels in the impression list
- `prop_id`: compare the quality of hotels in their own history
- `srch_des_id`: compare the quality of hotels in a given region
- `month`: compare the quality of hotels in a given time
- `srch_book_window`: compare the quality of hotels for a given booking window
- ...

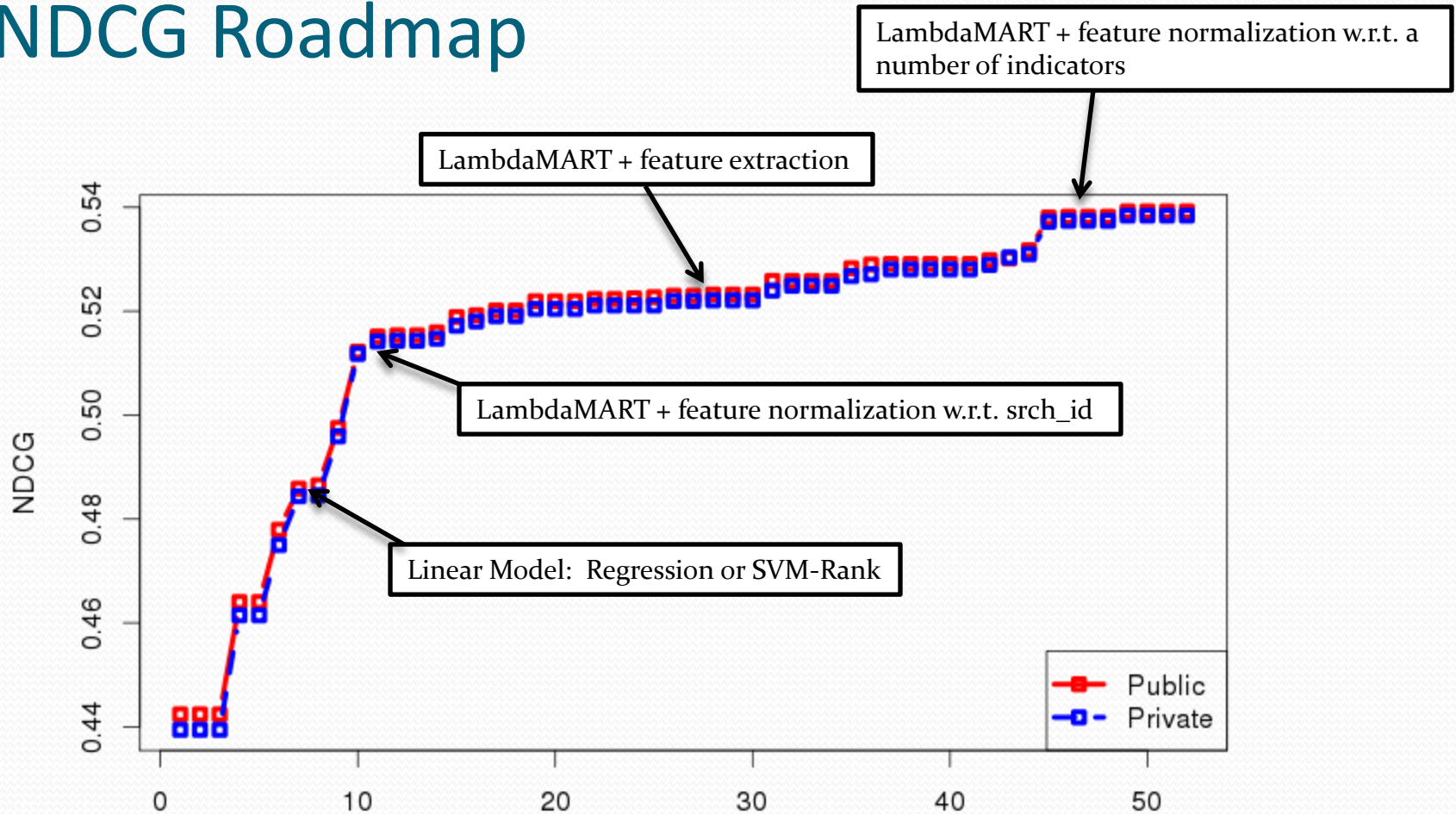
Example -- Hotel Price



Learning

- Training Data
 - Num. of instance: 9M
 - Training data 80% and Validation data 20%
 - Num. of feature: 300 (after feature engineering)
- Methodology
 - Linear model: regression or SVM-Rank
 - Nonlinear model: LambdaMART

NDCG Roadmap

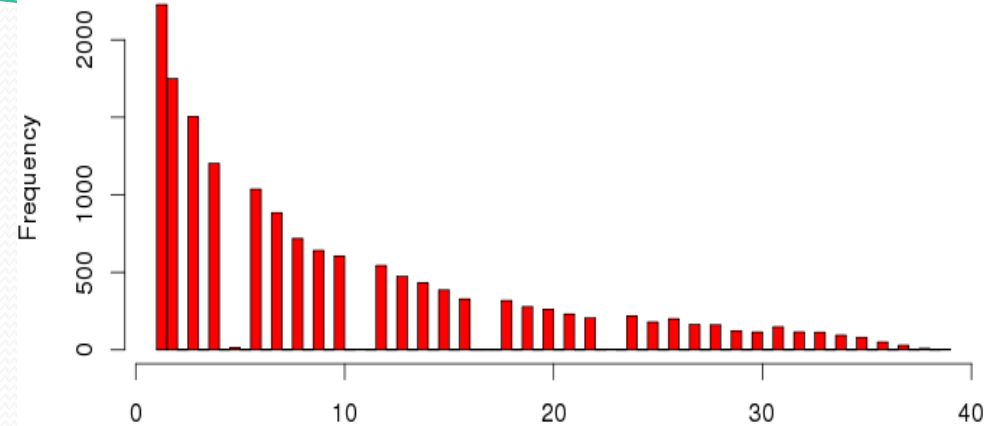


What matters?

- Feature Engineering
- Linear vs. Nonlinear
- Number of learning instances

Future works

Histogram of Position of Hotels Being Booked (Random Data)



- User's behaviour is not rational
 - Modelling the position bias
- Feature engineering
 - Missing value estimation in a more principle way
 - Normalizing features with multiple indicators
 - Remove redundant features
- Modelling Methodology
 - Improving LambdaMART
 - New modelling formulation based on MART