search ranking for "Experiences"

Search ranking for AirBNB experiences.

	2016	2017	2018
cities	500	5000	20000
experiences	12	60	1000

Different systems for different stages.

Stage 1
Offline ML model

- **Data size:** Small
- **Signals:**
  - Experience Features
- **Scoring**: Offline

Stage 2 Personalized
Offline ML model

- **Data size**: Medium
- **Signals:**
  - Experience Features
  - User Features
- **Scoring**: Offline

Stage 3 Personalized
Online ML model

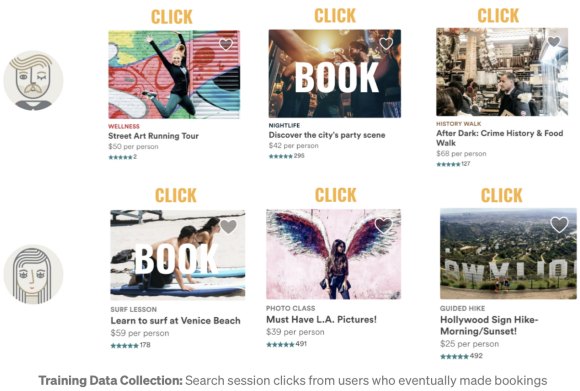
- **Data size**: Large
- **Signals:**
  - Experience Features
  - User Features
  - Query Features
- **Scoring**: Online

STAGE 1: Build a strong baseline

- When AirBNB launched, the amount of experiences to rank in the platform was very small
- The goal was to start collecting user interaction data on the experiences

Collecting data

- Best choice was to randomly re-rank until enough data is collected for building a first model.
- The collection was made for sessions where there was a booking.
- Labels in the dataset:
  - pos → Experiences that were clicked AND booked
  - neg → Experiences that were clicked BUT NOT booked



Training Data Collection: Search session clicks from users who eventually made bookings

Input features

- In stage 1, input features were purely based on the experiences.
  - Experience duration (1h, 2h, ...)
  - Price and price-per-hour
  - Category (cooking, music, surfing, etc)
  - Reviews (rating, # reviews)
  - Number of bookings (last 7, 30 days)
  - Occupancy past and future (eg. 60%)
  - Max number of seats (5 people)

→ careful → this is a metric that can rapidly change due to market conditions / seasonality
↳ better to use relative metrics (12 bookings per 1000 viewers)

Training and Testing the model

Training:

- GBDT for binary classification with log-loss function.

Testing:

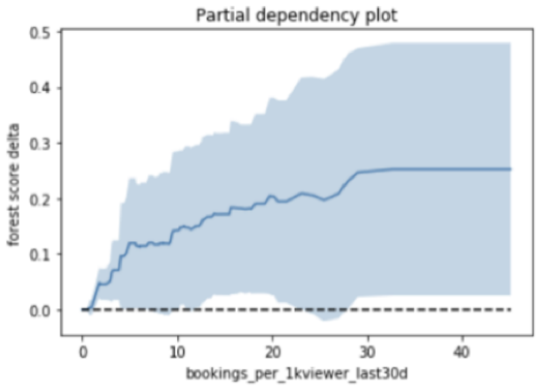
- AUC and NDCG
- Re-ranking items based on model scores (probability of ranking)
- Finally, checking where (position) would the items have landed
  - ↳ would they have been positioned higher compared to the position the user clicked on?

To understand the model

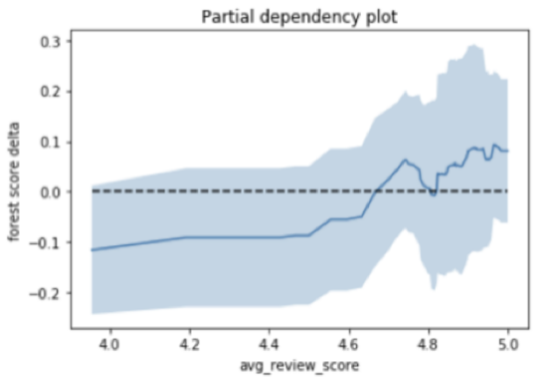
- ↳ Partial dependency plots
- ↳ Average input vs target
- ↳ SHAP

} these work well for a binary classification, but difficult to relate with nDCG.

more bookings per 1K views -> higher score



higher rating -> higher score



higher price -> lower score

Implementation

- Given that the model depended only on Experience features, reranking would be the same for all users.
  - ↳ therefore, this could be run offline on a daily basis and serve the ranking of the items are a lookup.

STAGE 2: Personalise

- Remember that the Experiences as a product is very different to hotels and apartments
  - Hotels: there might be dozen of similar rooms with similar price
  - Experiences: the distinction between cooking and surf is big.
  - Therefore, personalization should play a big part and serving the right most interesting content at the top will be important.

1) Personalize for users who have already booked an apartment

- we can get lot's information about the user based on the rented property.
  - Booked apartment location / domestic or international
  - Trip dates & length & lead days
  - Number of guests
  - Trip type (family, business)
  - Trip price (below/above market)
  - First trip or returning trip.

calculate apartment vs experience:
- distance between them
- is experience available during trip

2) Personalise based on user clicks

- Based on short term searches or based on sessions:

1) What category is the user mostly clicking?

These intensity and recency can be calculated for other user actions (bookings, wishlisting, etc).

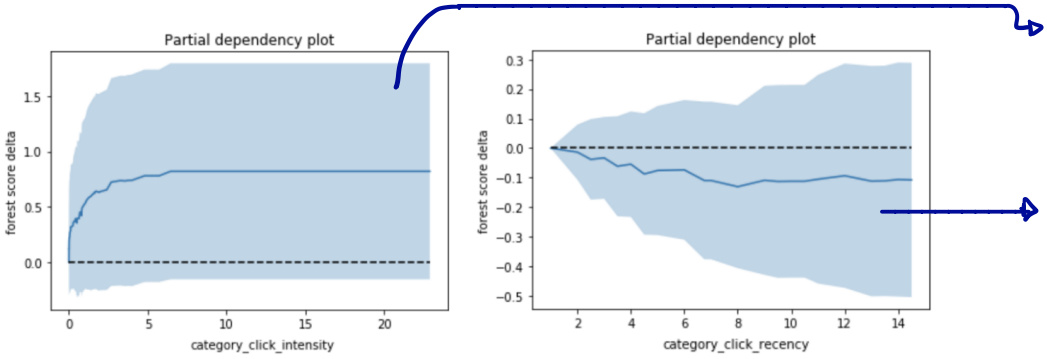
A) **Category intensity** → weighted sum of user clicks on Experiences that have that particular category where the sum is over the last 15 days.

$$\text{category intensity} = \sum_{d=d_0}^{d_{now}} \alpha^{(d-d_{now})} A_d$$

A_d → clicks

d_0 → 15 days.

B) **category recency** → number of days that passed since the user last clicked on an experience in that category.



→ Experiences where user has clicked more will rank higher

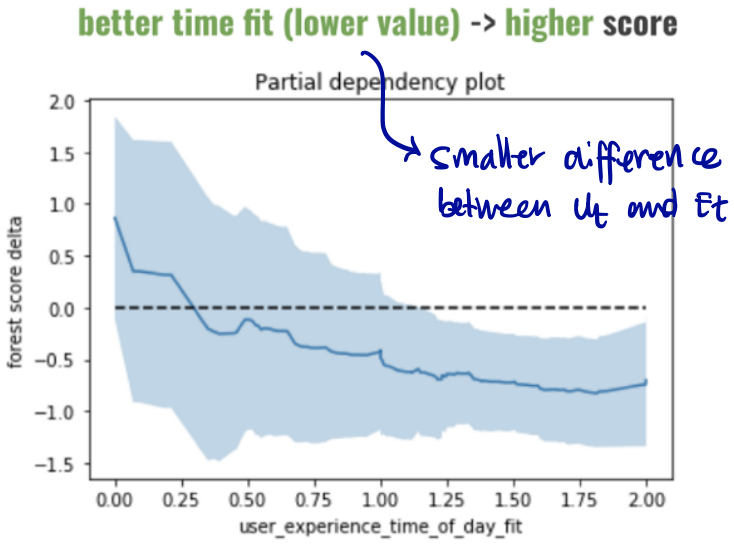
→ Experiences that haven't been visited in a long time, will be ranked lower.

2) Time of day personalization → is the user looking for experiences at night or in the morning?

user clicks
- morning → 70%
- afternoon → 25%
- evening → 5%

Experience (Et)
- morning → 100%
- afternoon → 0%
- evening → 0%

$$\sum_{\text{time of day}} \text{abs}(U_t - E_t)$$



Training and testing

- Careful with data leakage
  - For example, use only clicks that happened before booking
  - In addition, to avoid single interactions which can act as a proxy for future clicks, compute features if the user interacted with more than 1 feature.

- What happens to users who were not logged in?
  - Train 2 models → one with personalisation features for logged users
    - one without personalisation features for logged out traffic.

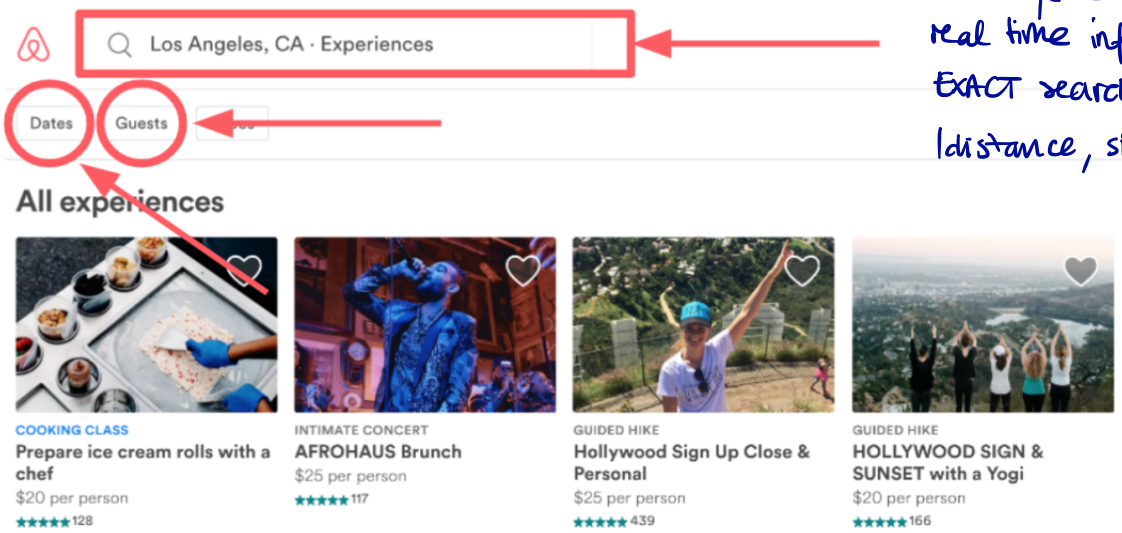
Implementation

- Created a lookup table keyed on user id that contained personalised rankings of all experiences
  - for those logged out, all share the same key (let's say 0)

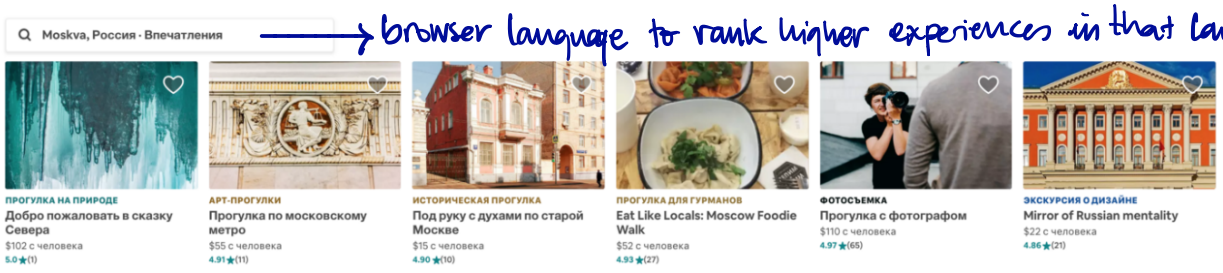
- Problem complexity → millions of users + thousands of experiences
  - solved by precomputing offline on a daily basis for top N million most active users.
    - now this has 1 day lag → this wouldn't be able to pick up a change in preference, number guests, etc
    - for the users that are not active, you would still rank without personalisation
  - however this can be used to proof the value of personalisation.

STAGE 3: move to online scoring.

- Remember that in Stage 2, we could infer aspects of user if they had previously logged in and infer aspects of location based on users who had booked.



In stage 3 we can leverage real time information about the EXACT searching parameters. (distance, singles/couples/groups)



→ browser language to rank higher experiences in that language

(flag is experience language = browser language)

- IP address → where is the user searching from?
  - ↳ for example, Experiences in Paris
    - → Japanese travellers prefer cooking
    - → US travellers prefer wine tasting.
    - → French prefer history.

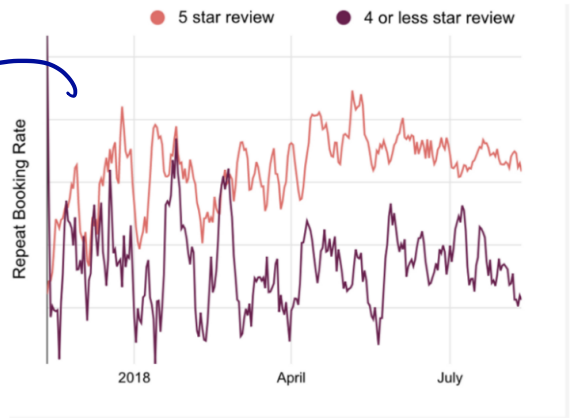
Training and testing

Still 2 models
- Model logged in users
  - item features
  - search query features
  - personalisation features
  - } still only top active users
- Model logged out users
  - item features
  - search query features.
  - } but now we can pseudo personalise for everyone else!

STAGE 4 : Handle Business rules

- up to this point the different stages have shown how to grow from zero a ranking system and have been steps to show value in each iteration.
- however, aside from Bookings, there might be secondary scores/goals to achieve → for example, quality.

users who provide higher ratings tend to repeat booking experiences



Experience Bookers ✓ target metric	▼ 0.36% ± 1.8
Experience Quality = high quality	2.5% ± 3.1
Experience Quality = medium quality	▼ 4.1% ± 3.2
Experience Quality = low quality	▼ 1.3% ± 4.5
Experience Quality = very low quality	▼ 5.7% ± 5.5
Experience Quality = very high quality	7.5% ± 6.3

→ Trigger a change in the relevance definition:

- ✗ booked = +1, clicked & not booked = -1
- ✓ weights given for quality ratings
  - → +5 ⇒ +50 reviews, rating 4.9, 55% user say unique experience

- Other secondary metrics:
  - → Discover new potential hits for new experiences
  - → Enforce diversity in top 8 results
  - → Optimise search without location → if the user doesn't specify a location, choose top 20 from all locations and then re-rank based on clickthrough rate.

STAGE 5 : monitoring

Hosts for experiences want to know what positively/negatively affects make sure the models are doing what we expect.

Example explaining why the ranking of a particular Experience improved over time

STAGE 6: Future iterations

- Change loss function → using pairwise loss to compare items that were booked vs non-booked
- Change relevance target →
  - 0 → impression
  - 0.1 → click
  - 0.2 → click with selected date & time
  - 1 → booking
  - 1.2 → booking & high quality
- Training data construction → log the features when used by the model to always reconstruct what happened.
- Real-time down to events that happened 10 mins ago.
- Tackling position bias

SUMMARY

Experience Bookings as Guests (target metric)	Model Type	Data Size
13% ± 9.3	Offline Model **Experience** features	500 Experiences 50K Training Data Size
7.9% ± 5.7	Offline Model **Experience** & **User** features	4.000 Experiences 250K Training Data Size
5.1% ± 3.4	**Online** Model **Experience** & **User** & **Query** features	16.000 Experiences 2M Training Data Size
2.3% ± 1.8	**Enforcing Diversity in Top 8**	20.000 Experiences
2.2% ± 1.7	**Low Intent Anywhere Search** Re-rank **Top 18** based on **CTR**	20.000 Experiences