Scoring

After candidate generation, another model scores and ranks the generated candidates to select the set of items to display. The recommendation system may have multiple candidate generators that use different sources, such as the following:

Examples

- Related items from a matrix factorization model.
- User features that account for personalization.
- "Local" vs "distant" items; that is, taking geographic information into account.
- Popular or trending items.
- A social graph; that is, items liked or recommended by friends.

The system combines these different sources into a common pool of candidates that are then scored by a single model and ranked according to that score. For example, the system can train a model to predict the probability of a user watching a video on YouTube given the following:

- query features (for example, user watch history, language, country, time)
- video features (for example, title, tags, video embedding)

The system can then rank the videos in the pool of candidates according to the prediction of the model.

Why Not Let the Candidate Generator Score?

Since candidate generators compute a score (such as the similarity measure in the embedding space), you might be tempted to use them to do ranking as well. However, you should avoid this practice for the following reasons:

- Some systems rely on multiple candidate generators. The scores of these different generators might not be comparable.
- With a smaller pool of candidates, the system can afford to use more features and a more complex model that may better capture context.

Choosing an Objective Function for Scoring

As you may remember from Introduction to ML Problem Framing (https://developers.google.com/machine-learning/problem-framing/), ML can act like a mischievous genie: very happy to learn the objective you provide, but you have to be careful what you wish for. This mischievous quality also applies to recommendation systems. The choice of scoring function can dramatically affect the ranking of items, and ultimately the quality of the recommendations.

For Example:

Click the plus icons to learn what happens as a result of using each objective.

Maximize Click Rate

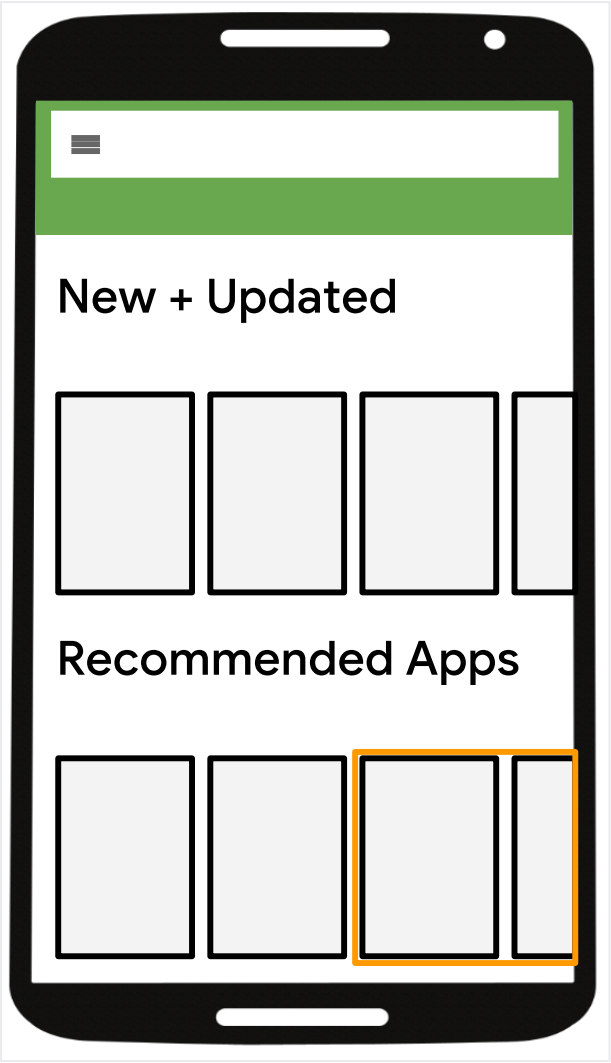
If the scoring function optimizes for clicks, the systems may recommend click-bait videos. This scoring function generates clicks but does not make a good user experience. Users' interest may quickly fade.

Maximize Watch Time

If the scoring function optimizes for watch time, the system might recommend very long videos, which might lead to a poor user experience. Note that multiple short watches can be just as good as one long watch.

Increase Diversity and Maximize Session Watch Time

Recommend shorter videos, but ones that are more likely to keep the user engaged.



Positional Bias in Scoring

Items that appear lower on the screen are less likely to be clicked than items appearing higher on the screen. However, when scoring videos, the system usually doesn't know where on the screen a link to that video will ultimately appear. Querying the model with all possible positions is too expensive. Even if querying multiple positions were feasible, the system still might not find a consistent ranking across multiple ranking scores.

Solutions

- Create position-independent rankings.
- Rank all the candidates as if they are in the top position on the screen.

Previous
← Retrieval (/machine-learning/recommendation/dnn/retrieval)

Next
Re-ranking (/machine-learning/recommendation/dnn/re-ranking) →

Last updated 2020-02-10 UTC.