150 successful machine learning models: 6 lessons learned at Booking.com

OCTOBER 7, 2019

150 successful machine learning models: 6 lessons learned at Booking.com Bernadi et al., *KDD'19*

Here's a paper that will reward careful study for many organisations. We've previously looked at the deep penetration of machine learning models in the product stacks of leading companies, and also some of the pre-requisites for being successful with it. Today's paper choice is a wonderful summary of lessons learned integrating around 150 successful customer facing applications of machine learning at Booking.com. Oddly enough given the paper title, the six lessons are never explicitly listed or enumerated in the body of the paper, but they can be inferred from the division into sections. My interpretation of them is as follows:

1. Projects introducing machine learned models deliver strong business value
2. Model performance is not the same as business performance
3. Be clear about the problem you're trying to solve
4. Prediction serving latency matters
5. Get early feedback on model quality
6. Test the business impact of your models using randomised controlled trials (follows from #2)

There are way more than 6 good pieces of advice contained within the paper though!

> We found that driving true business impact is amazingly hard, plus it is difficult to isolate and understand the connection between efforts on modeling and the observed impact… Our main conclusion is that an iterative, hypothesis driven process, integrated with other disciplines was fundamental to build 150 successful products enabled by machine learning.

In case you're tempted to stop reading at this point, please don't interpret that quote as saying that investing in machine learning isn't worth it. On the contrary, developing an organisational capability to design, build, and deploy successful machine learned models in user-facing contexts is, in my opinion, as fundamental to an organisation's competitiveness as all the other characteristics of high-performing organisations highlighted in the State of DevOps reports. (And

by the way, wouldn't it be wonderful to see data confirming or denying that hypothesis in future reports!).

Context

You've probably heard of Booking.com, 'the world's largest online travel agent.' Delivering a great experience to their users is made challenging by a number of factors:

- The stakes are high for recommendations – booking a stay at the wrong place is much worse than streaming a movie you don't like!
- User's provide scant information about what they're really looking for when booking a trip
- The supply of accommodation is constrained, and changing prices impact guest preferences
- Guest preferences may change each time they use the platform (if e.g. booking only once or twice per year)
- There is a lot of rich information available regarding accommodations, which can be overwhelming for users

Different types of model

With around 150 models now in production, you won't be surprised to hear that machine learning has touched many parts of the Booking.com experience. Some models are very *specific*, focusing on a particular use case in a particular context (e.g. recommendations tailored for one point in the funnel), other models act as a *semantic layer*, modelling concepts that can be generally useful in many contexts. For example, a model indicating how flexible a user is with respect to the destination of their trip.

The models deployed at Booking.com can be grouped into six broad categories:

- **Traveller preferences models** operate in the semantic layer, and make broad predictions about user preferences (e.g., degree of flexibiilty).
- **Traveller context models**, also semantic, which predictions about the context in which a trip is taking place (e.g. with family, with friends, for business, …).
- **Item space navigation models** which track what a user browses to inform recommendations both the the user's history and the catalog as a whole.
- **User interface optimisation models** optimise elements of the UI such as background images, font sizes, buttons etc. Interestingly here, "*we found that it is hardly the case that one specific value is optimal across the board, so our models consider context and user information to decide the best user interface*."
- **Content curation models** curate human-generated content such as reviews to decide which ones to show
- **Content augmentation models** compute additional information about elements of a trip, such as which options are currently great value, or how prices in an area are trending.

Lesson 1: projects introducing machine learned models deliver strong business value

All of these families of models have provided business value at Booking.com. Moreover, compared to other *successful* projects that have been deployed but did not use machine learning, the machine learning based projects tend to deliver *higher returns*.

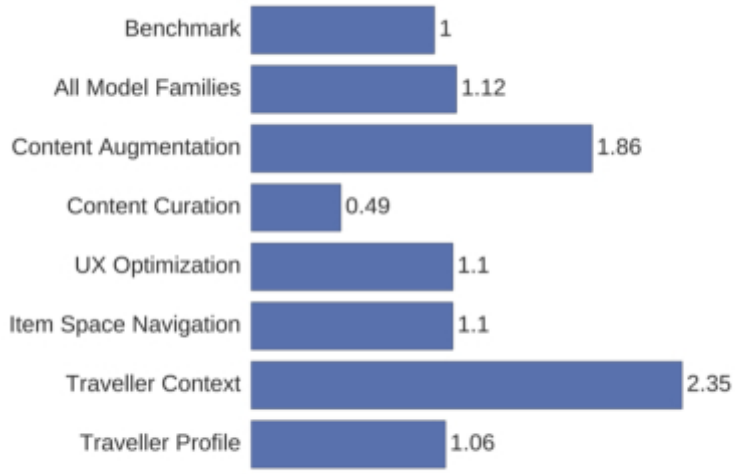

Figure 2: Model Families Business Impact relative to median impact.

Once deployed, beyond the immediate business benefit they often go on to become a foundation for further product development. The following figure shows the impact of a succession of deployments, each building on the original and further improving the business outcome.

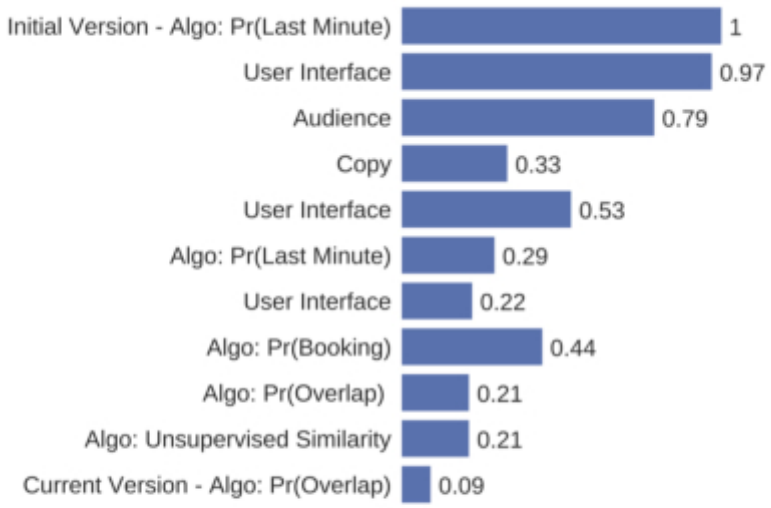

Figure 3: A sequence of experiments on a Recommendations Product. Each experiment tests a new version focusing on the indicated discipline or ML Problem Setup. The length of the bar is the observed impact relative to the first version (all statistically significant)

Lesson 2: model performance is not the same as business performance

Booking.com estimate the value delivered by a model through randomized controlled trials which measure the impact on business metrics.

> An interesting finding is that increasing the performance of a model does not necessarily translate into a gain in [business] value.

This could be for a number of reasons including saturation of business value (there's no more to extract, whatever you do); segment saturation due to smaller populations being exposed to a treatment (as the old and new models are largely in agreement); over-optimisation on a proxy metric (e.g. clicks) that fails to convert into the desired business metric (e.g. conversion); and the *uncanny valley* effect, which is best explained through the following picture:

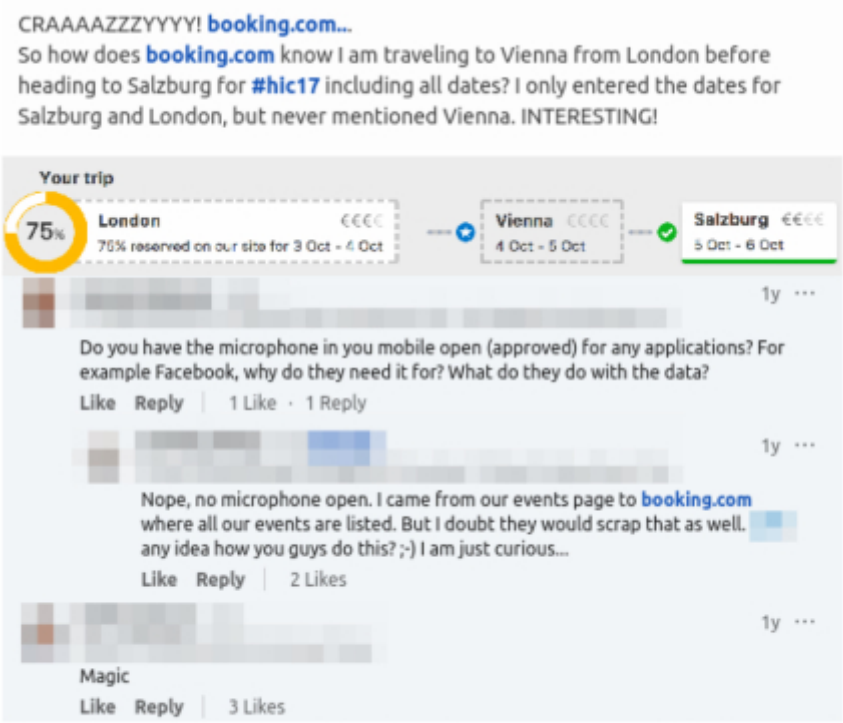

Figure 5: Uncanny valley: People not always react positively to accurate predictions (destination recommender using Markov chains).

Lesson 3: be clear about the problem you're trying to solve

Before you start building models, it's worth spending time carefully constructing a definition of the problem you are trying to solve.

> The Problem Construction Process takes as input a business case or concept and outputs a well-defined modeling problem (usually a supervised machine learning problem), such that a good solution effectively models the given business case or concept.

Some of the most powerful improvements come not from improving a model in the context of a given setup, but changing the setup itself. For example, changing a user preference model based on clink data to a natural language processing problem based on guest review data.

> In general we found that often the best problem is not the one that comes to mind immediately and that changing the set up is a very effective way to unlock value.

Lesson 4: prediction serving latency matters

Here we have yet another data point on the impact of performance on business metrics. In a experiment introducing synthetic latency, Booking.com found that an increase of about 30% in latency cost about 0.5% in conversion rates "*a relevant cost for our business*".

> This is particularly relevant for machine learned models since they require significant computational resources when making predictions. Even mathematically simple models have the potential of introducing relevant latency.

Booking.com go to some lengths to minimise the latency introduced by models, including horizontally scaled distributed copies of models, a in-house developed custom linear prediction engine, favouring models with fewer parameters, batching requests, and pre-computation and/or caching.

Lesson 5: get early feedback on model quality

> When models are serving requests, it is crucial to monitor the quality of their output but this poses at least two challenges…

- Incomplete feedback due to the difficulty of observing true labels
- Delayed feedback e.g. a prediction made at time of booking as to whether a user will leave a review cannot be assessed until after the trip has been made.

One tactic Booking. com have successfully deployed in these situations with respect to binary classifiers is to look at the distribution of responses generated by the model. "*Smooth bimodal distributions with one clear stable point are signs of a model that successfully distinguishes two classes*." Other shapes (see figure below) can be indicative of a model that is struggling.

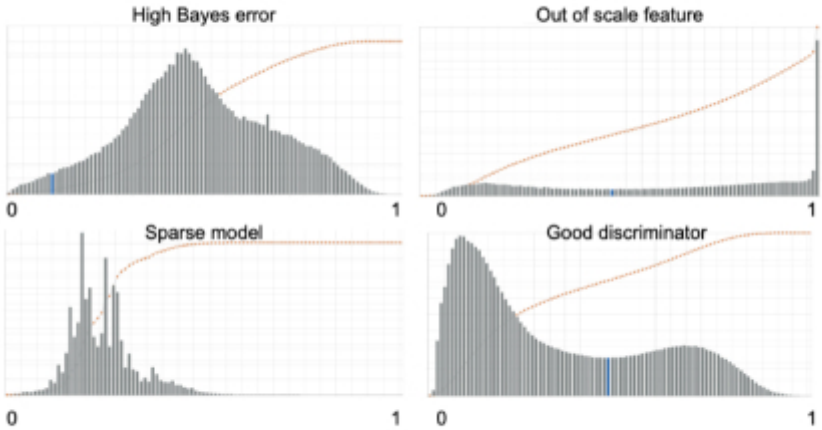

Figure 7: Examples of Response Distribution Charts

> …Response Distribution Analysis has proved to be a very useful tool that allows us to detect defects in the models very early.

Lesson 6: test the business impact of your models through randomised controlled trials

> The large majority of the successful use cases of machine learning studied in this work have been enabled by sophisticated experiment designs, either to guide the development process or in order to detect their impact.

The paper includes suggestions for how to set up the experiments under different circumstances.

- When not all subjects are eligible to be exposed to a change (e.g., they don't have a feature the model requires), create treatment and non-treatments groups from within the eligible subset.

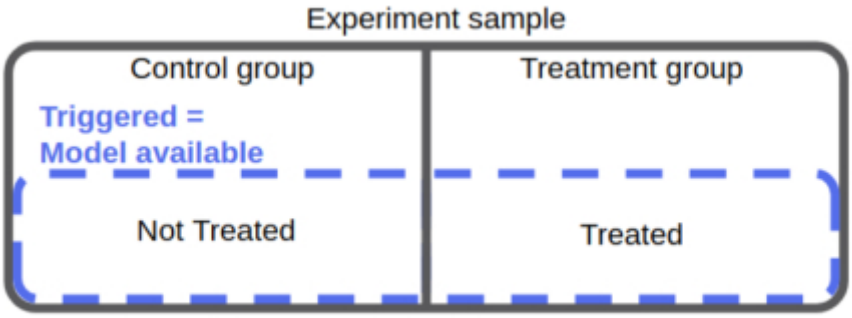

Figure 8: Experiment design for selective triggering.

- If the model only produces outputs that influence the user experience in a subset of cases, then further restrict the treatment and non-treatment groups to only those cases where the model produces a user-observable output (which won't of course be seen in the non-treatment group). To assess the impact of performance add a third control group where the model is not invoked at all.

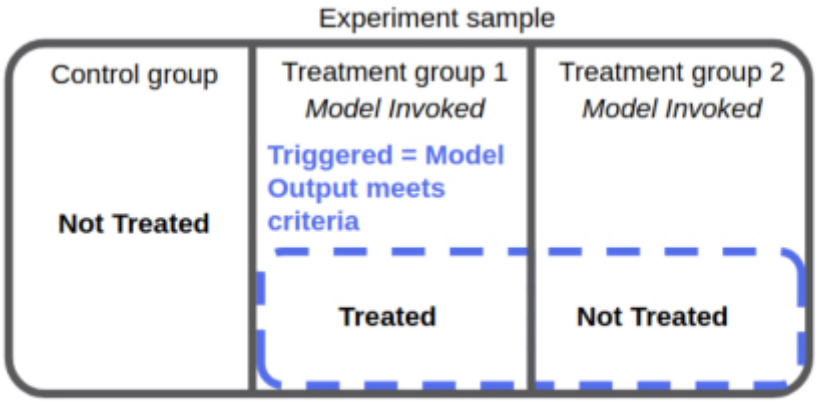

Figure 9: Experiment design for model-output dependent triggering and control for performance impact.

- When comparing models we are interested in situations where the two models *disagree*, and we use as a baseline a control group that invokes the current model (assuming we're testing a current model against a candidate improvement). That leads to an experiment design which looks like this:

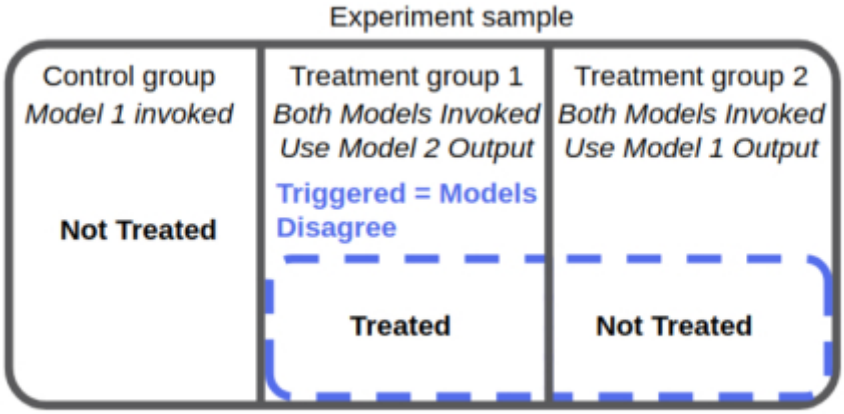

Figure 10: Experiment design for comparing models.

The last word:

> Hypothesis driven iteration and interdisciplinary integration are the core of our approach to deliver value with machine learning, and we wish this work to serve as guidance to other machine learning practitioners and spark further investigations on the topic.