Putting Deep Learning Models in Production

Sahil Dua


Let’s imagine!
Booking.com

But …

Booking.com

whoami

- Software Developer @ Booking.com
- Previously - Deep Learning Infrastructure
- Open Source Contributor (Git, Pandas, Kinto, go-github, etc.)
- Tech Speaker

Agenda

- ➔ Deep Learning at Booking.com
- ➔ Life-cycle of a model
- ➔ Training Models
- ➔ Serving Predictions

Deep Learning at Booking.com

Scale
highlights.

1,500,000+
room nights
booked
every 24 hours

1.4 million+
active properties
in **220+** countries

Deep Learning

- → Image understanding
- → Translations
- → Ads bidding
- → ...

Image Tagging






Booking.com





Image Tagging

Classes	Score
oceanfront	0.79
nature	0.79
beach house	0.62
building	0.62
penthouse	0.61
apartment	0.61
housing	0.61

Booking.com

Image Tagging

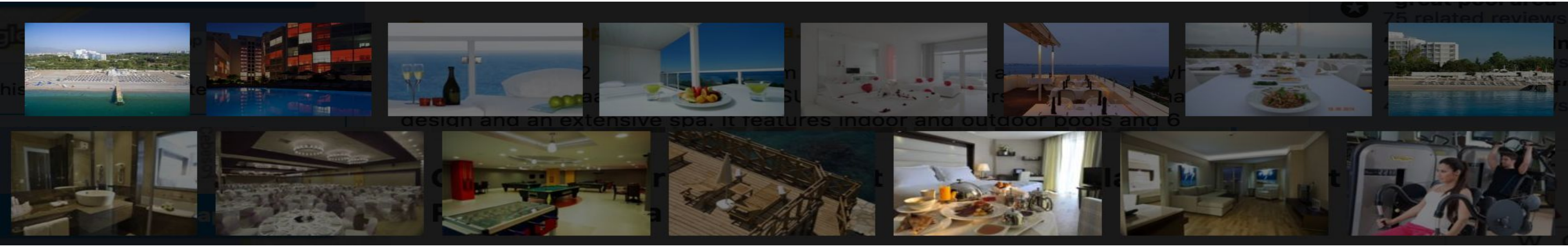
Classes	Score
oceanfront	0.79
nature	0.79
beach house	0.62
building	0.62
penthouse	0.61
apartment	0.61
housing	0.61

Sea view: 6.38
Balcony/Terrace: 4.82
Photo of the whole room: 4.21
Bed: 3.47
Decorative details: 3.15
Seating area: 2.70

Booking.com

Image Tagging

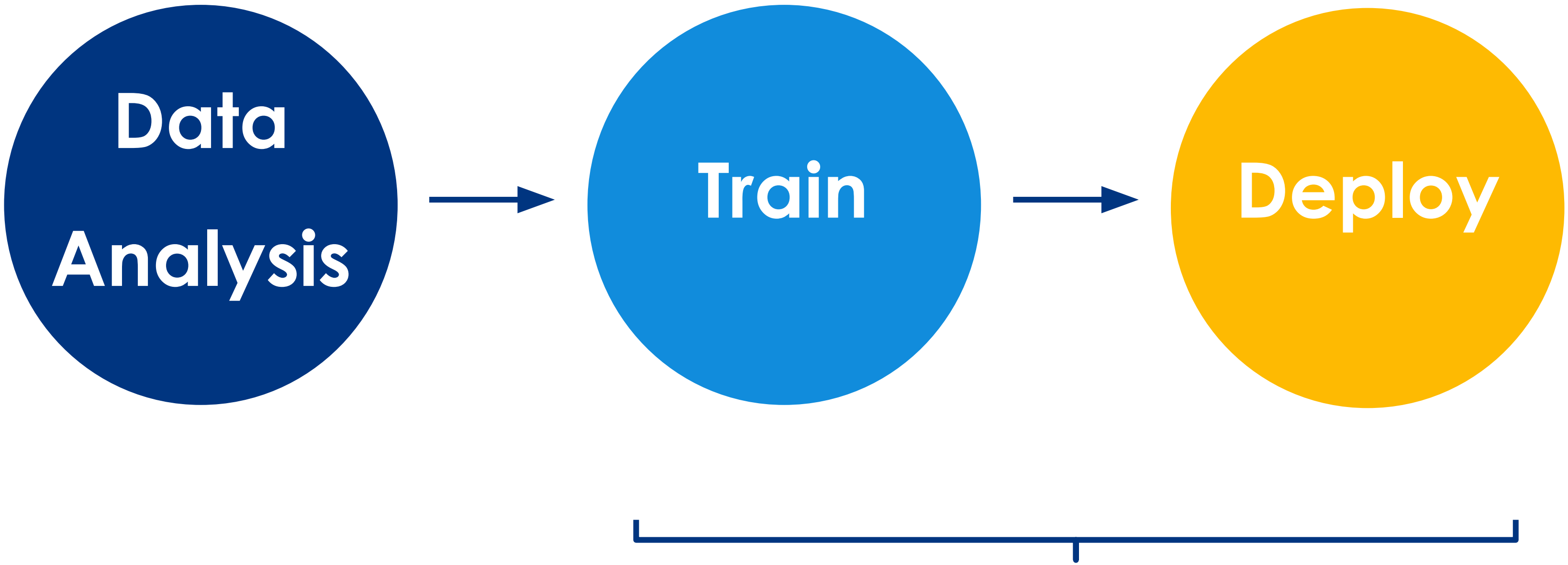
Using the image tag information in the right context
Swimming pool, Breakfast Buffet, etc.



Lifecycle of a model

Booking.com

Lifecycle of a model



Training a Model - on laptop

Training a Model - on laptop

Machine Learning workload

- → Computationally intensive workload
- → Often not highly parallelizable algorithms
- → 10 to 100 GBs of data

Why Kubernetes (k8s)?

- Isolation
- Elasticity
- Flexibility

Why k8s – GPUs?

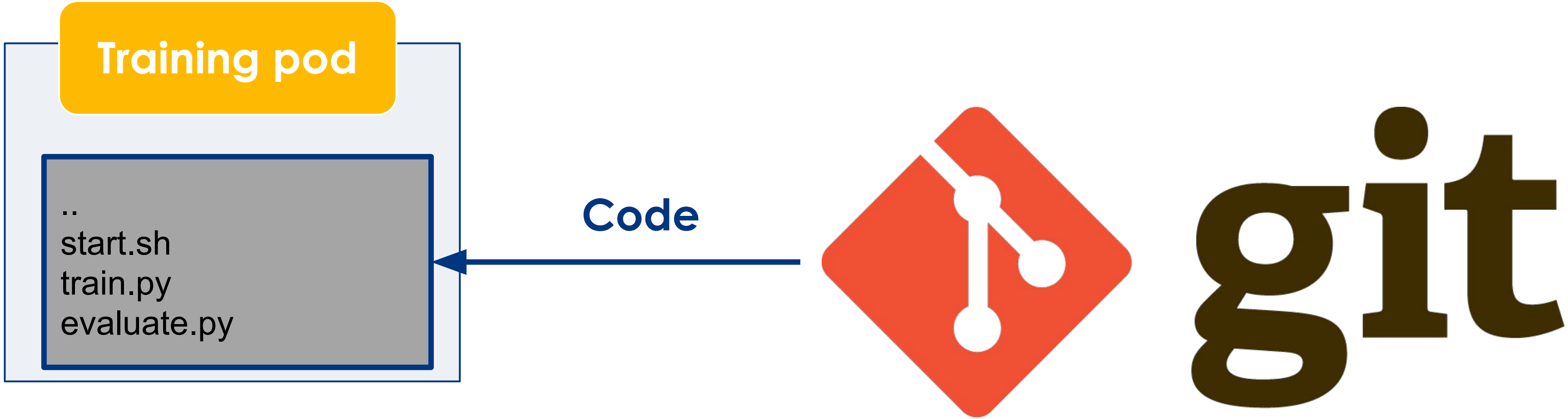
- ➔ In alpha since 1.3
- ➔ Speed up 20X-50X

```
resources:
  limits:
    alpha.kubernetes.io/nvidia-gpu: 1
```

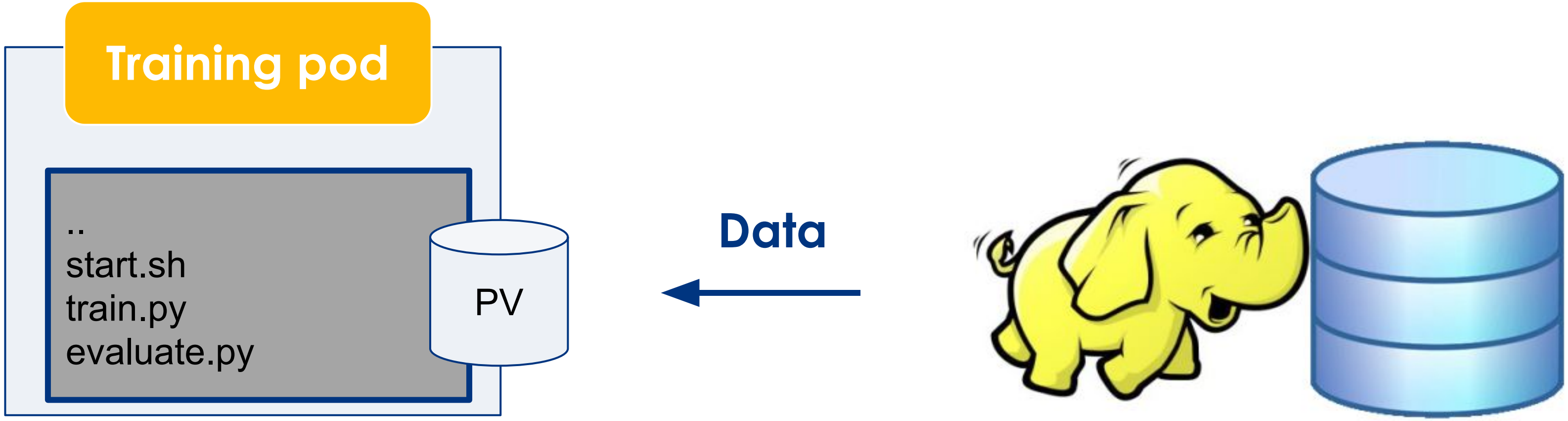
Training with k8s

- ➔ Base images with ML frameworks
  - ◆ TensorFlow, Torch, VowpalWabbit, etc.
- ➔ Training code is installed at start time
- ➔ Data access - Hadoop (or PVs)

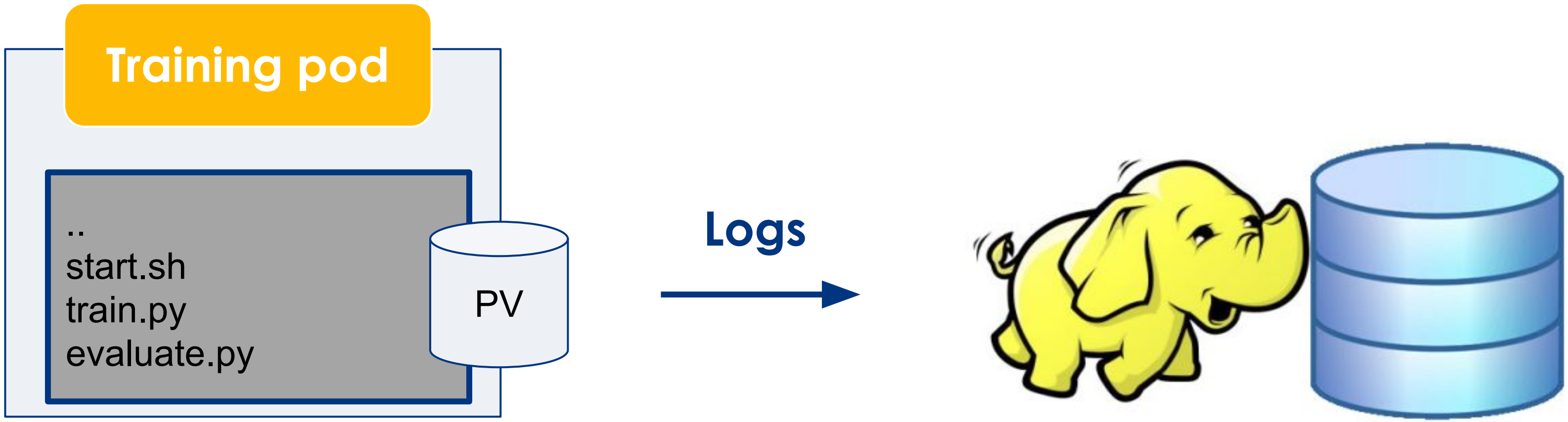
Startup



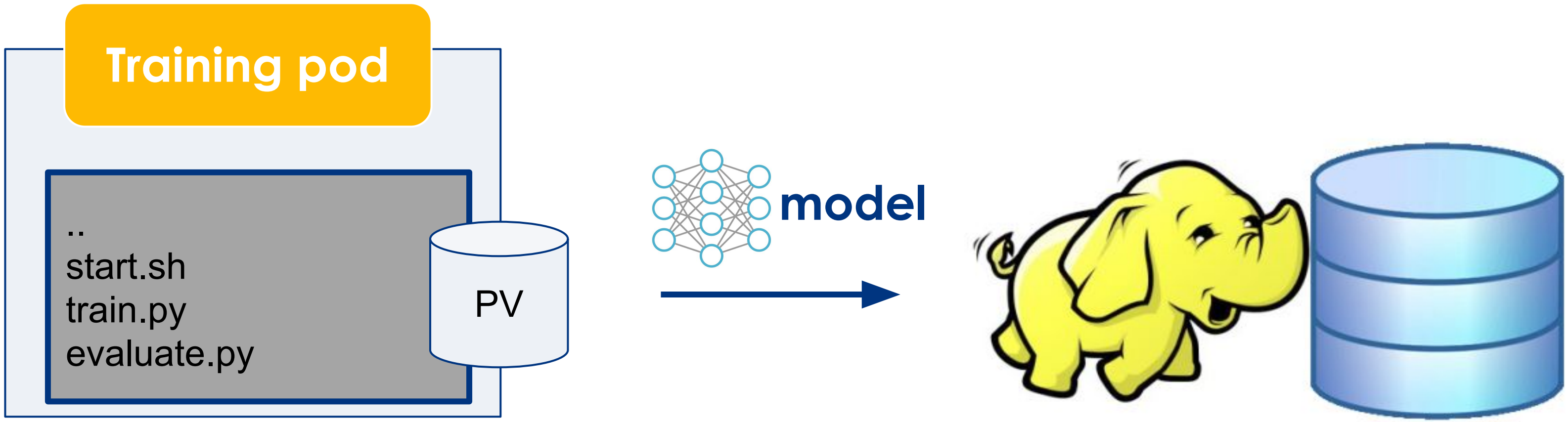
Startup



Streaming logs back

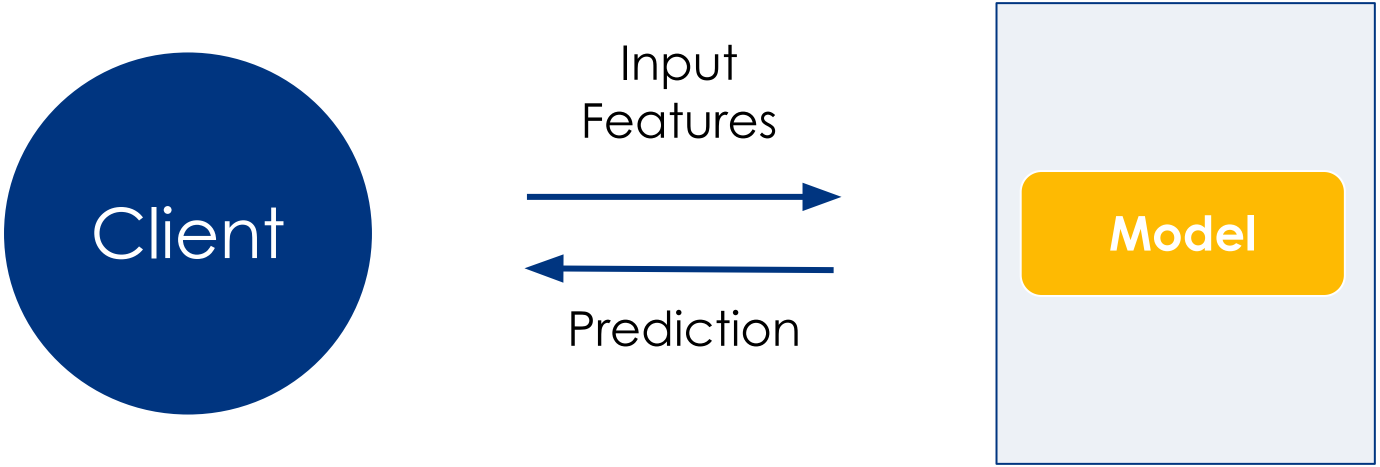


Exports the model

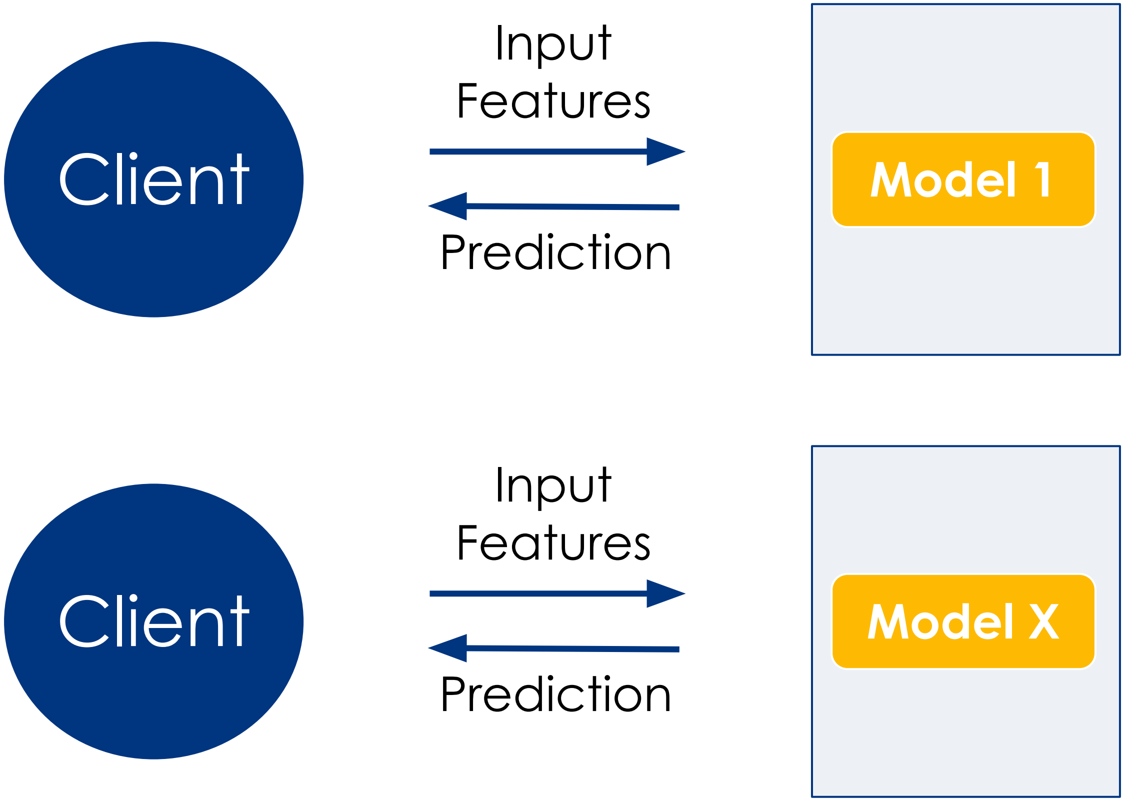


Serving predictions

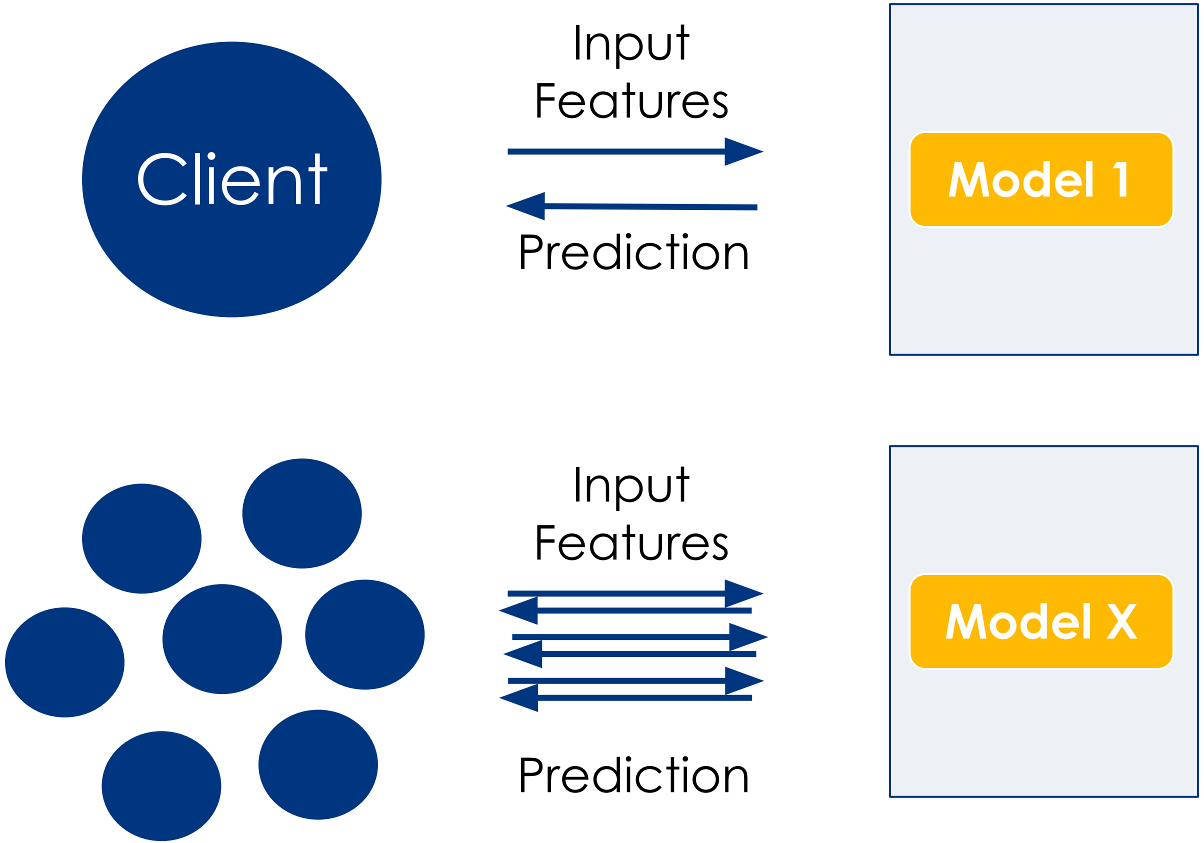
Serving Predictions



Serving Predictions



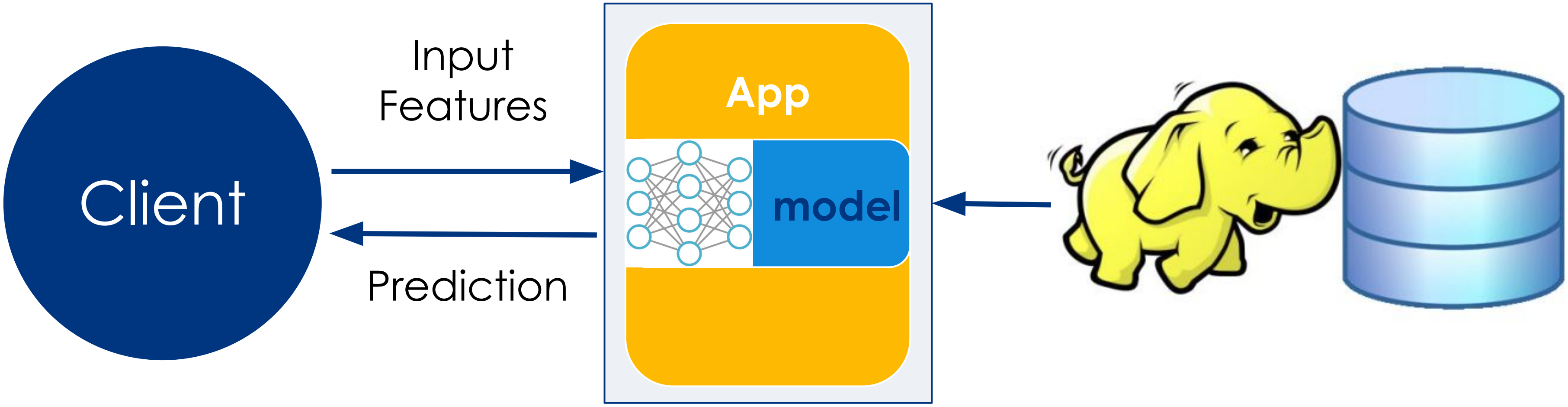
Serving Predictions



Serving Predictions

- ➔ Stateless app with common code
- ➔ Containerized
- ➔ No model in image
- ➔ REST API for predictions

Serving Predictions

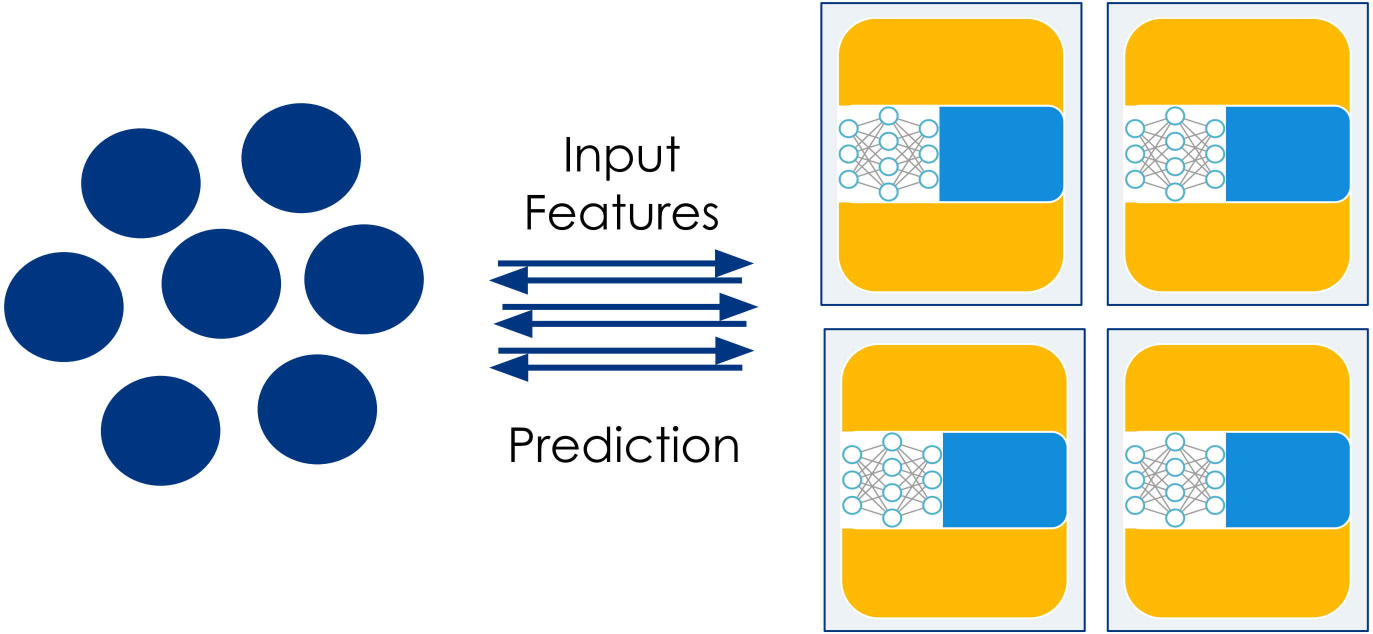


Booking.com

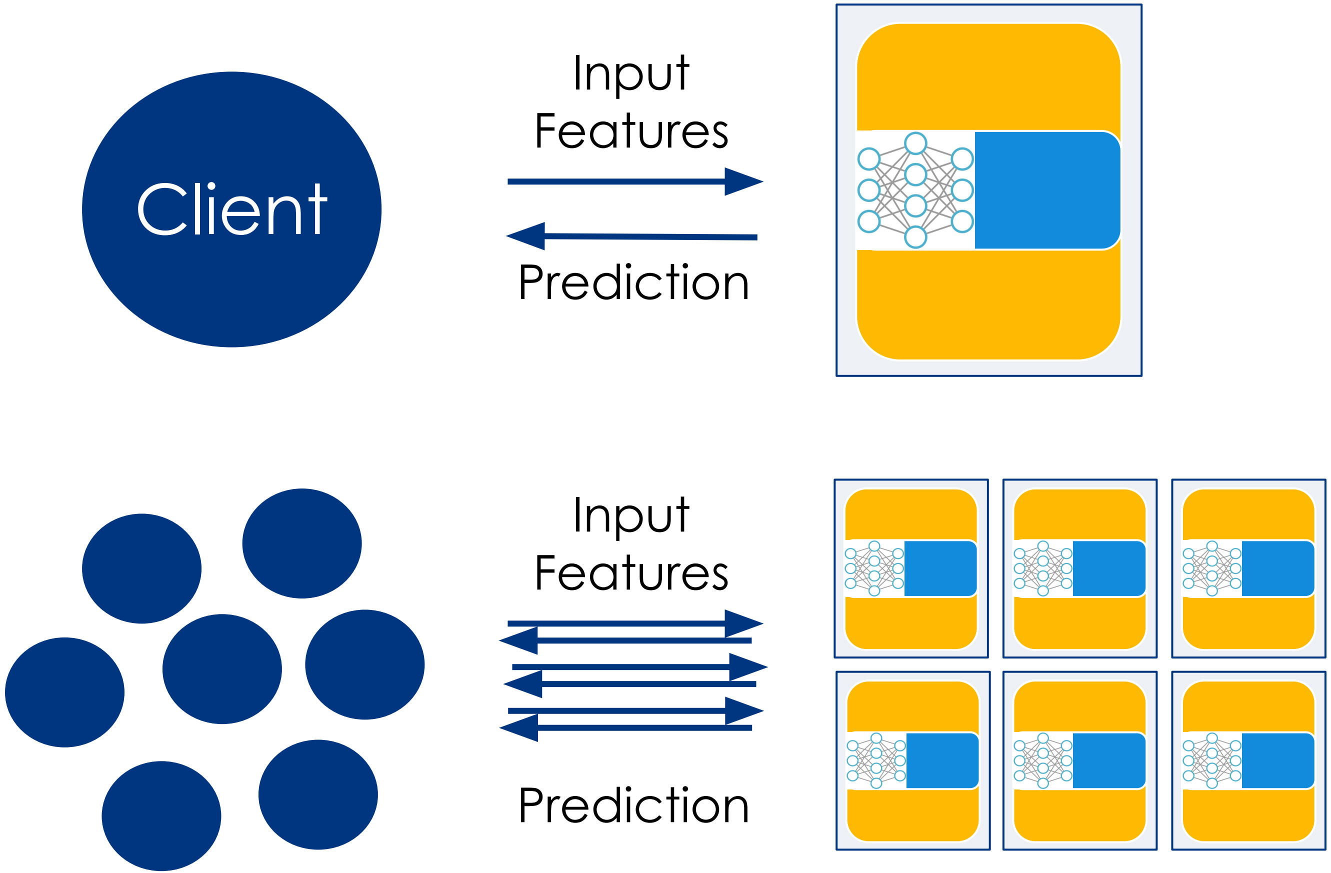
Serving Predictions

- ➔ Get trained model from Hadoop
- ➔ Load model in memory
- ➔ Warm it up
- ➔ Expose HTTP API
- ➔ Respond to the probes

Serving Predictions



Serving Predictions



Deploying a new model

- ➔ Create new Deployment
- ➔ Create new HTTP Route
- ➔ Wait for liveness/readiness probe

Performance

PredictionTime = RequestOverhead + N*ComputationTime

N is the number of instances to predict on

Optimizing for Latency

- Do not predict if you can precompute
- Reduce Request Overhead
- Predict for one instance
- Quantization (float 32 => fixed 8)
- TensorFlow specific: freeze network & optimize for inference

Optimizing for Throughput

- ➔ Do not predict if you can precompute
- ➔ Batch requests
- ➔ Parallelize requests

Summary

→ Training models in pods

→ Serving models

→ Optimizing serving for latency/throughput

Next steps

- ➔ Tooling to control hundred deployments
- ➔ Autoscale prediction service
- ➔ Hyper parameter tuning for training

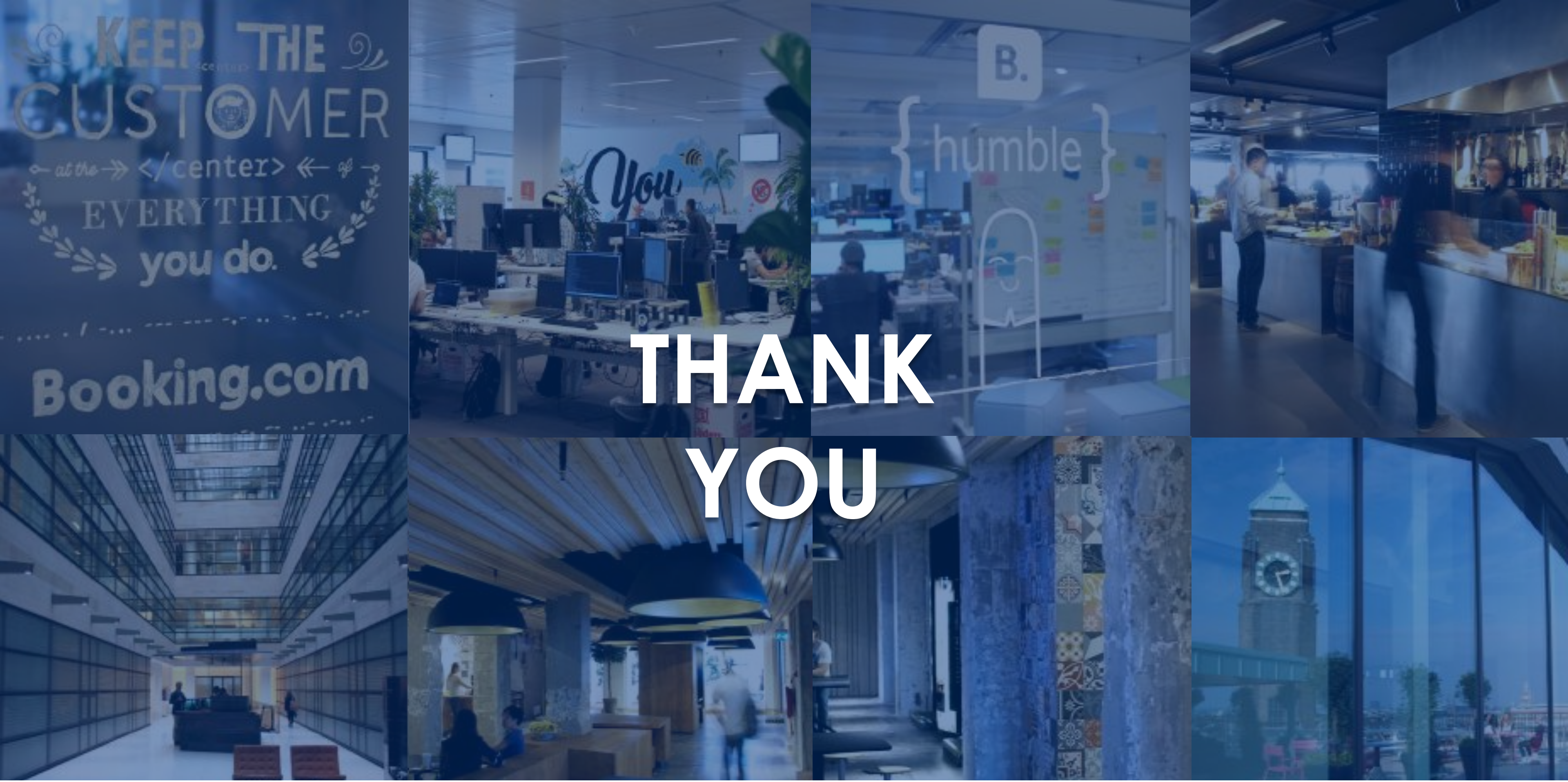
Want to get in touch?

LinkedIn / Twitter / GitHub

@sahildua2305

Website

www.sahildua.com



THANK YOU

@sahildua2305