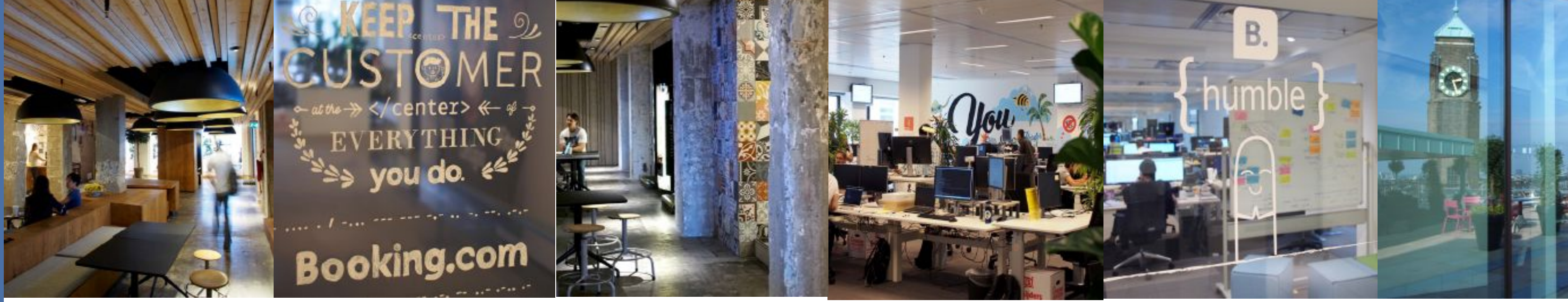


Scaling ML at B.com with Sparkling Water

Luca Falsina
Ben Teeuwen
Booking.com

“We don’t have better algorithms than anyone else; we just have more data.”

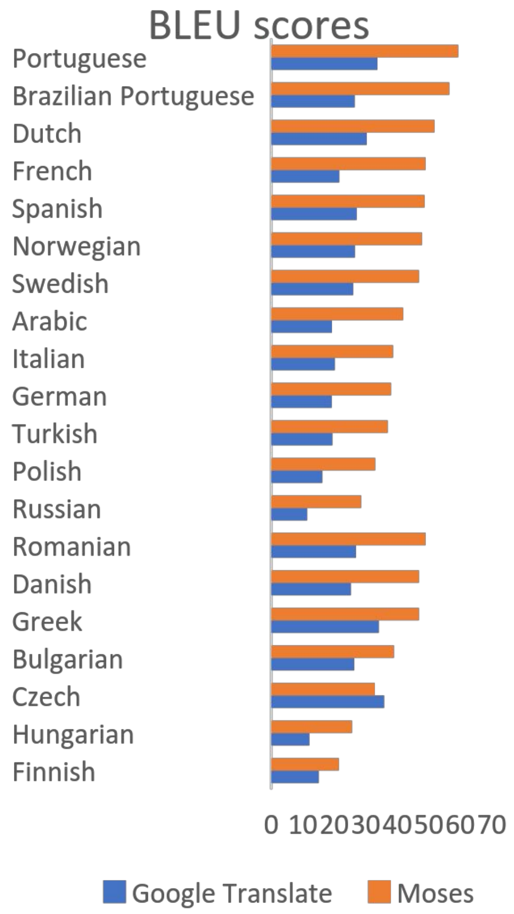
Peter Norvig,
Google’s Zeitgeist, 2011

Simple Algorithms for everyone

Machine Translating Hotel Descriptions

>1m partners, 43 languages
Efficient operations to human-translate by priority

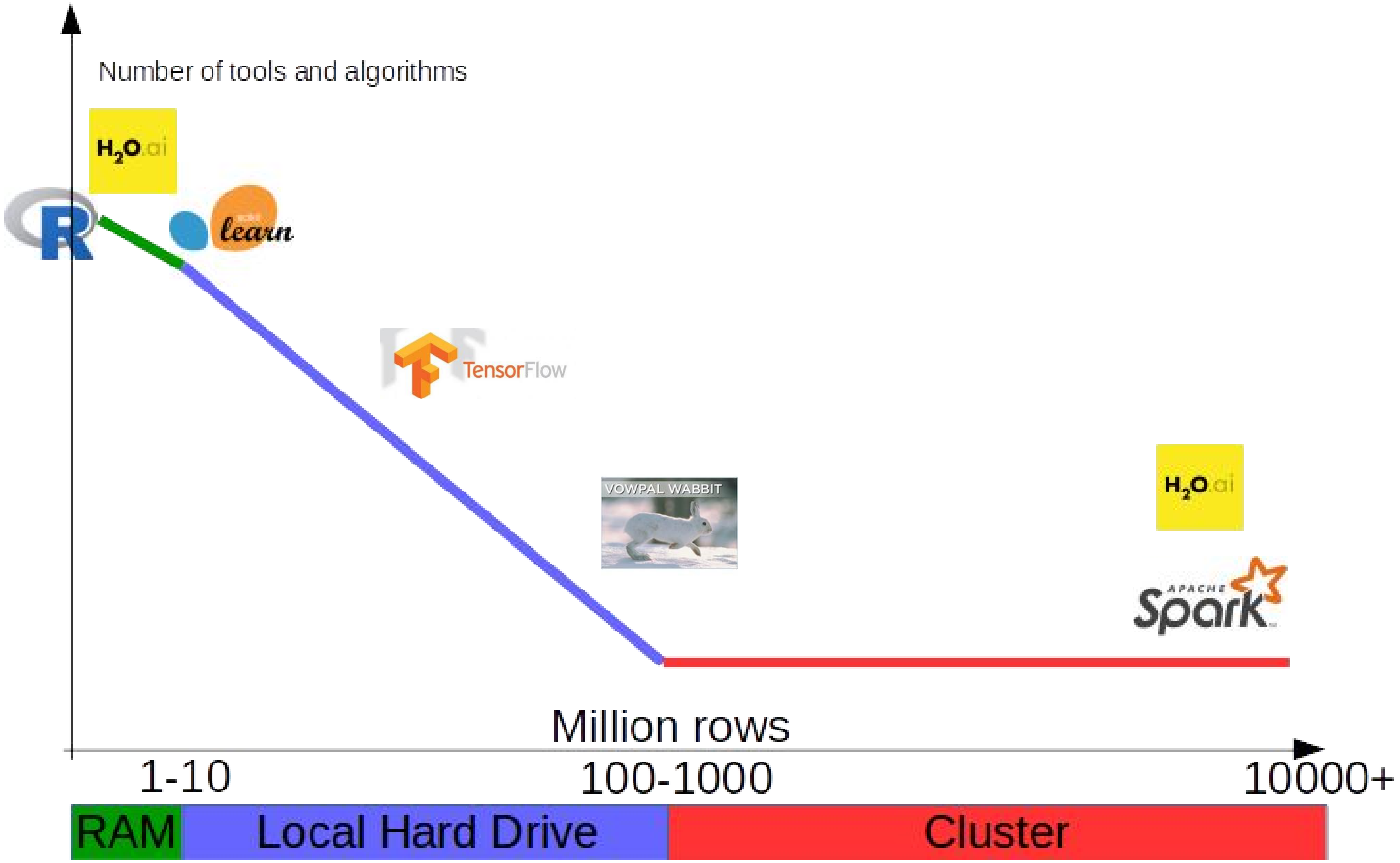
Company growth
€ millions to translate it all



>> Beats most famous Translation engine(2016)

Booking.com

Data Scale vs Tool Box



Marketing and Ranking

Marketing

Set bids daily for 1 billion different keywords

Recommendations

Score million hotels for hundreds of thousands of concurrent users

Email Marketing

Recommend 100k destinations for 80m users

Distributed ML requirements

- ❏ Large scale
- ❏ Easy to use
- ❏ Statistically sound
- ❏ Fast
- ❏ Reliable
- ❏ Easily productionizable

First try: ~ 2 years ago

Tried out Spark’s great data munging capabilities

Downsides:

- ML was unstable and slow
- Not many functionalities.
- Difficult to productize and slow in prediction

Next try: ~1 year ago

- Fast
- easy to use
- Scalable
- fast in prediction
- good algorithms

- Downside: YARN compatibility

(internal cluster mode)

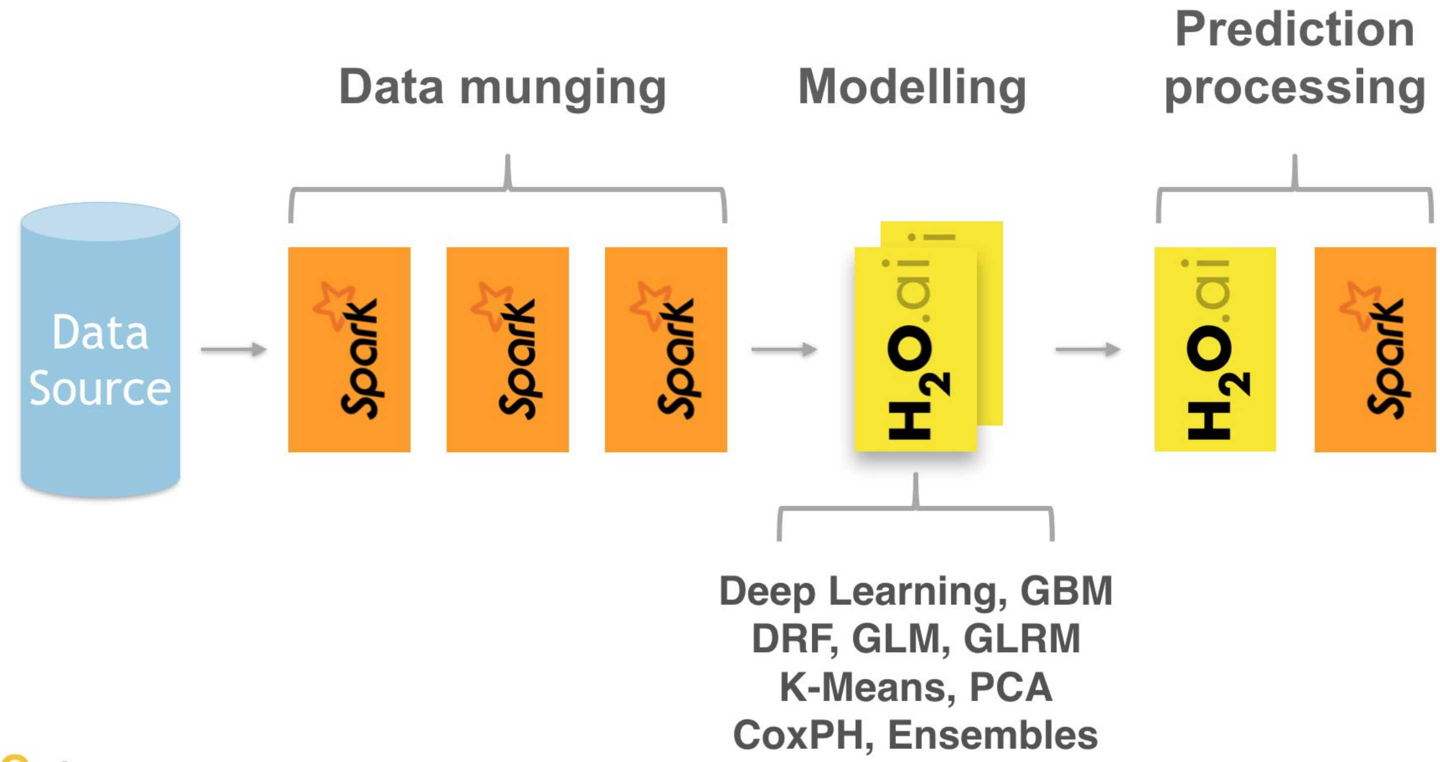
Third try: Team up with H2O

Developed external **Kluster** mode.

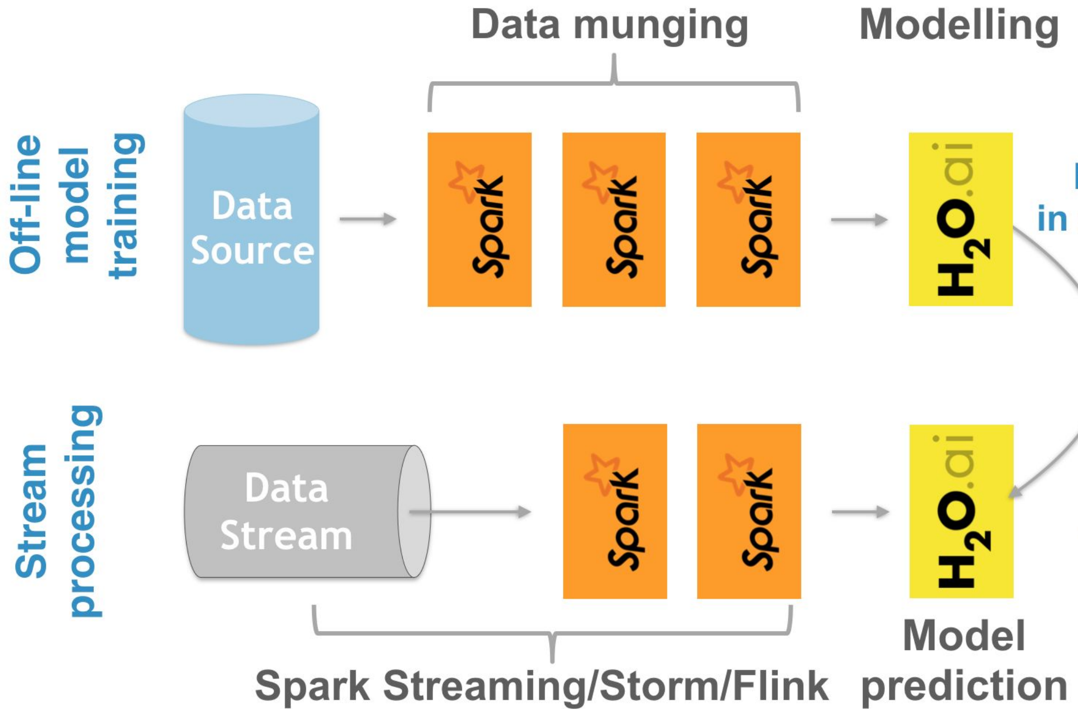
Sparkling Water

- Integration of H2O with Spark
  - H2O data structures and algorithms usable with Spark
- Boost Spark workflows with advanced ML algorithms

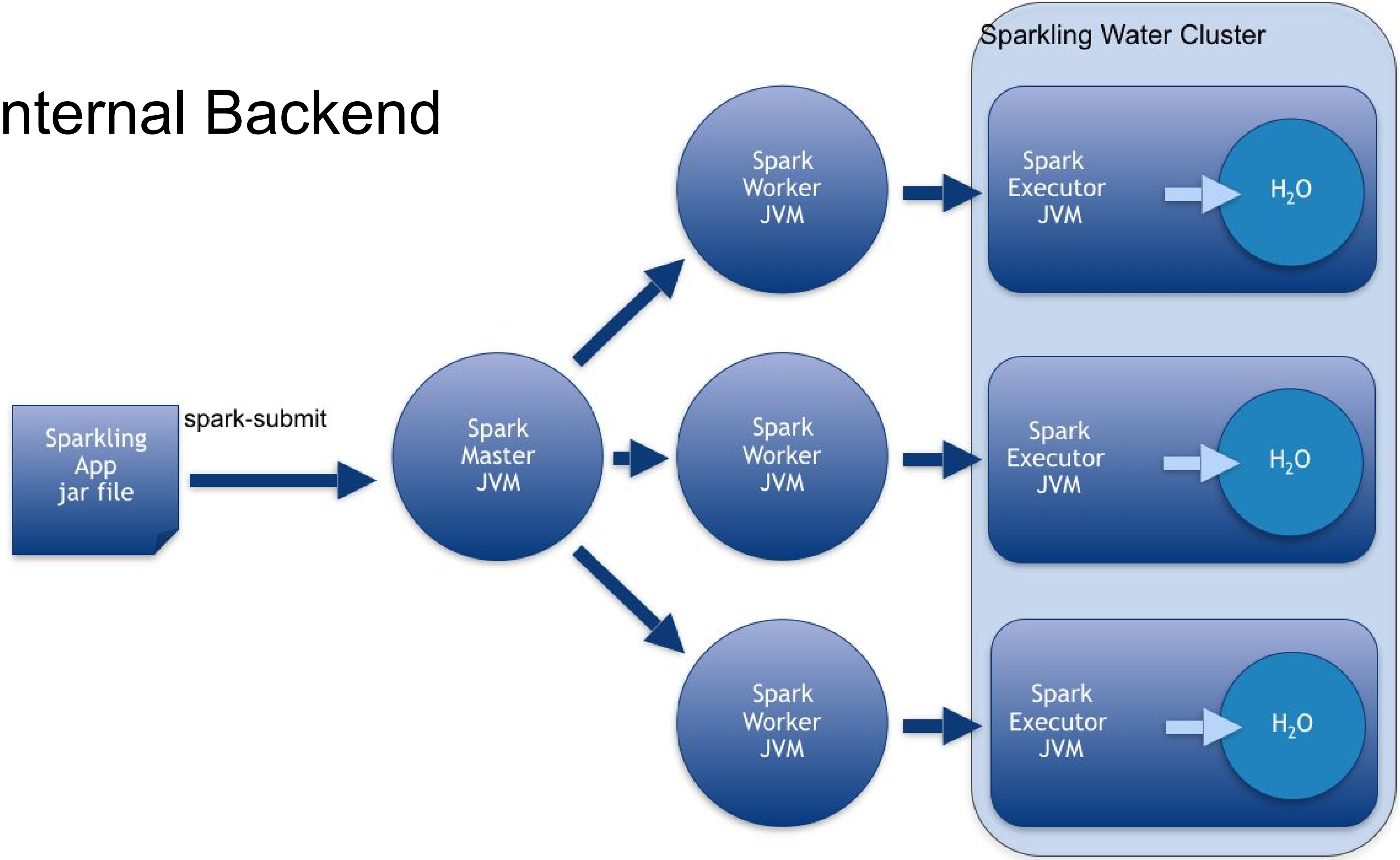
Model Building



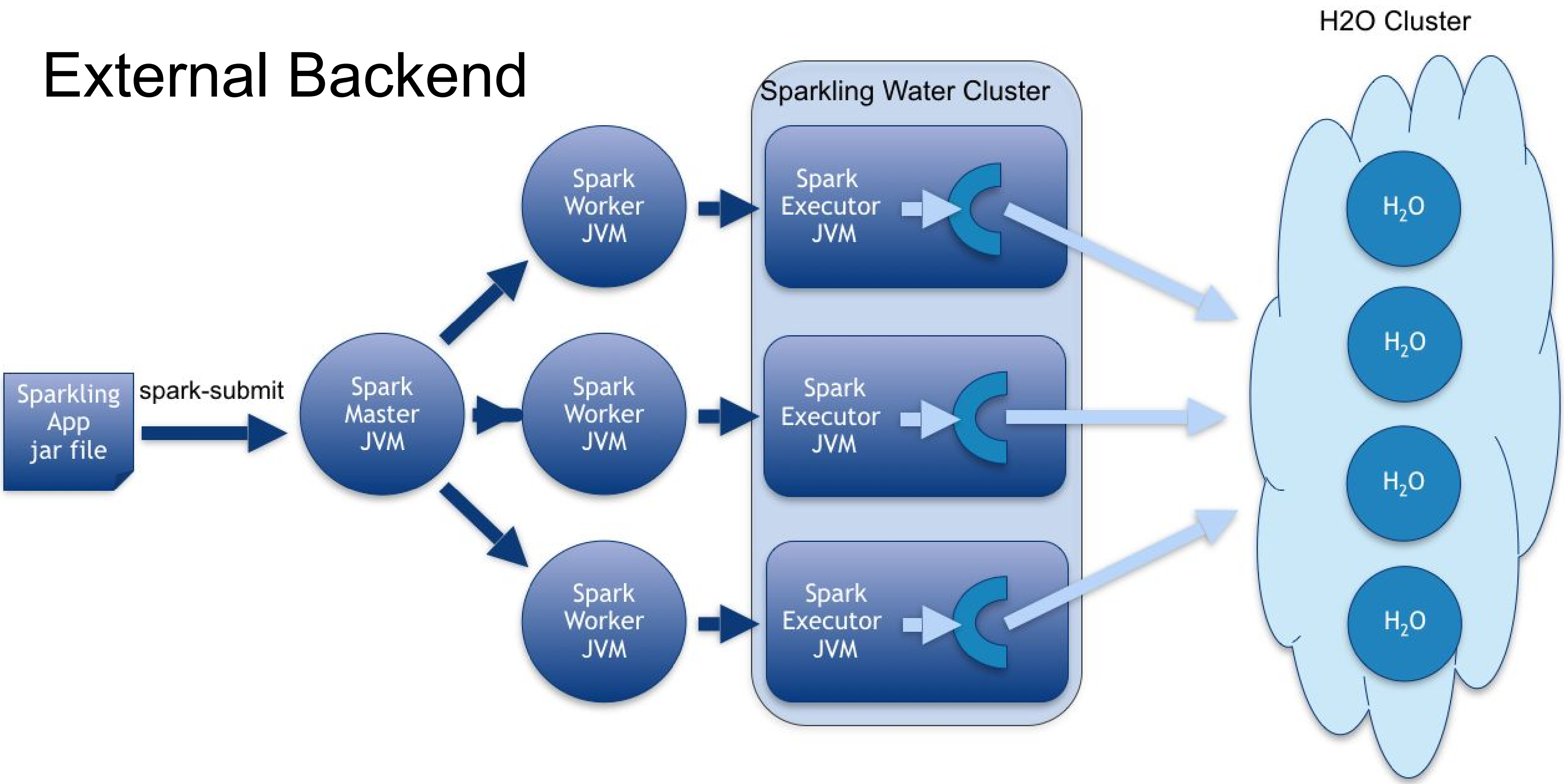
Offline & Stream Processing



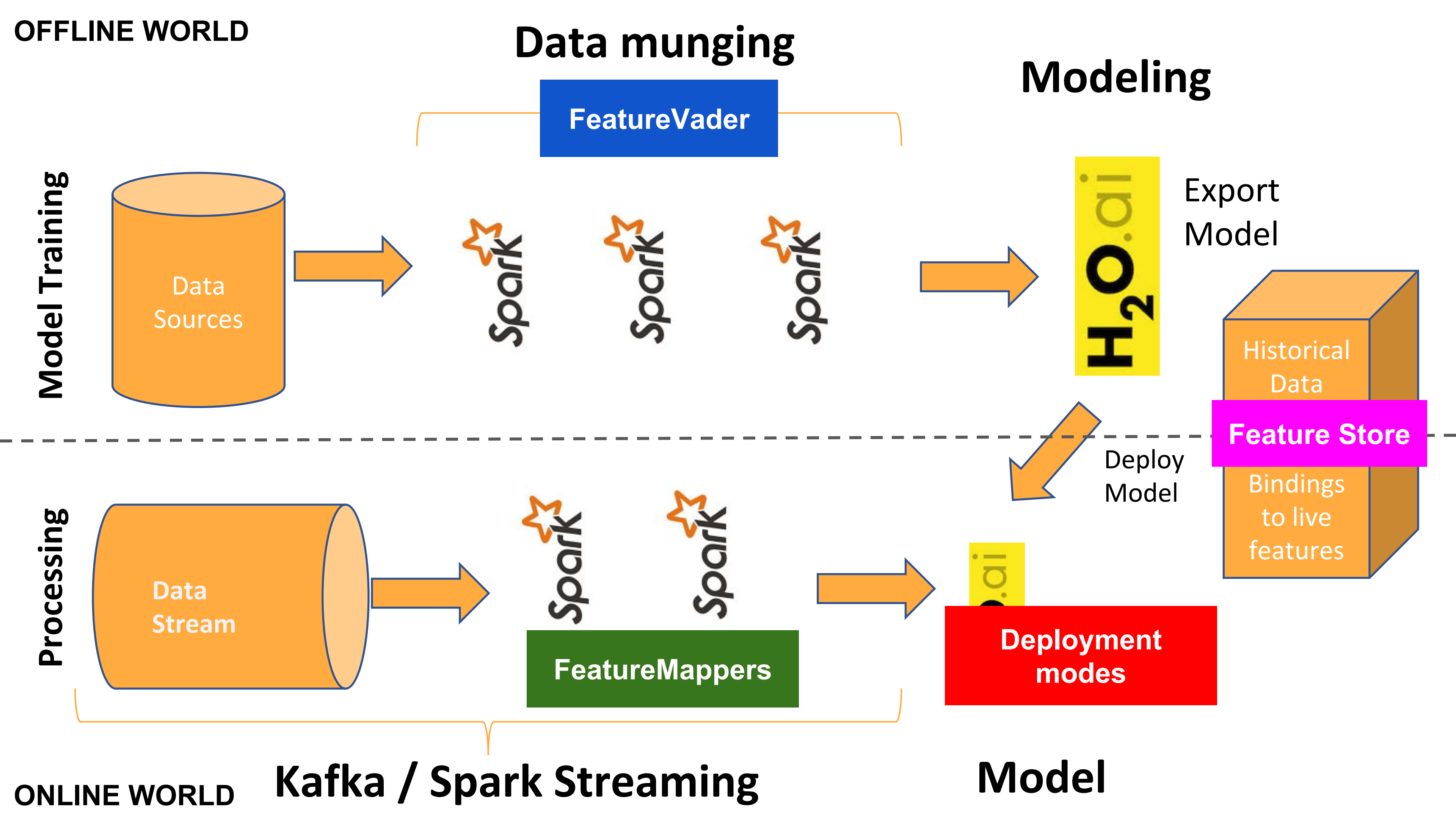
Internal Backend



External Backend



ML pipeline lift off


OFFLINE WORLD
Data munging
FeatureVader
Modeling
Model Training
Data Sources
Spark
Spark
Spark
Export Model
Historical Data
Feature Store
Bindings to live features
Deploy Model
Processing
Data Stream
Spark
Spark
FeatureMappers
Deployment modes
ONLINE WORLD
Kafka / Spark Streaming
Model

OFFLINE DATA MUNGING

Extract relevant data from events

- Billions of json payloads with unstructured data
- End up structured into data warehouse

Scalability

- Scale to billions of rows
- May need to process years of data to backfill features for a training

Feature engineering

- Raw values vs absolute / windowed aggregates
- Transformations (e.g. likelihood encoder)

Time coordinate matching

- Match instance epoch with historically correct feature value

FeatureVader

Construct offline features

- **FeatureVader** contains feature registry
  - Custom Spark ML Transformer for in notebooks:

```
labeledInstances = spark.sql("select userId, time, IF(x > 4, 1,0) label FROM data")
fv = FeatureVader()
fv.setTimeStamp("time").setDesiredFeatures(["feature1", "feature2"])
withFeatures = fv.transform(labeledInstances)
```

userId	time	label
1001	25	1
1002	36	0

userId	time	label	feature1	feature2
1001	25	1	213.5	2 kids
1002	36	0	123.7	0 kids

ONLINE DATA MUNGING

Nearly real-time processing

- Data from specialized streams on frontend
- Delay in generating features: seconds

Infrastructure

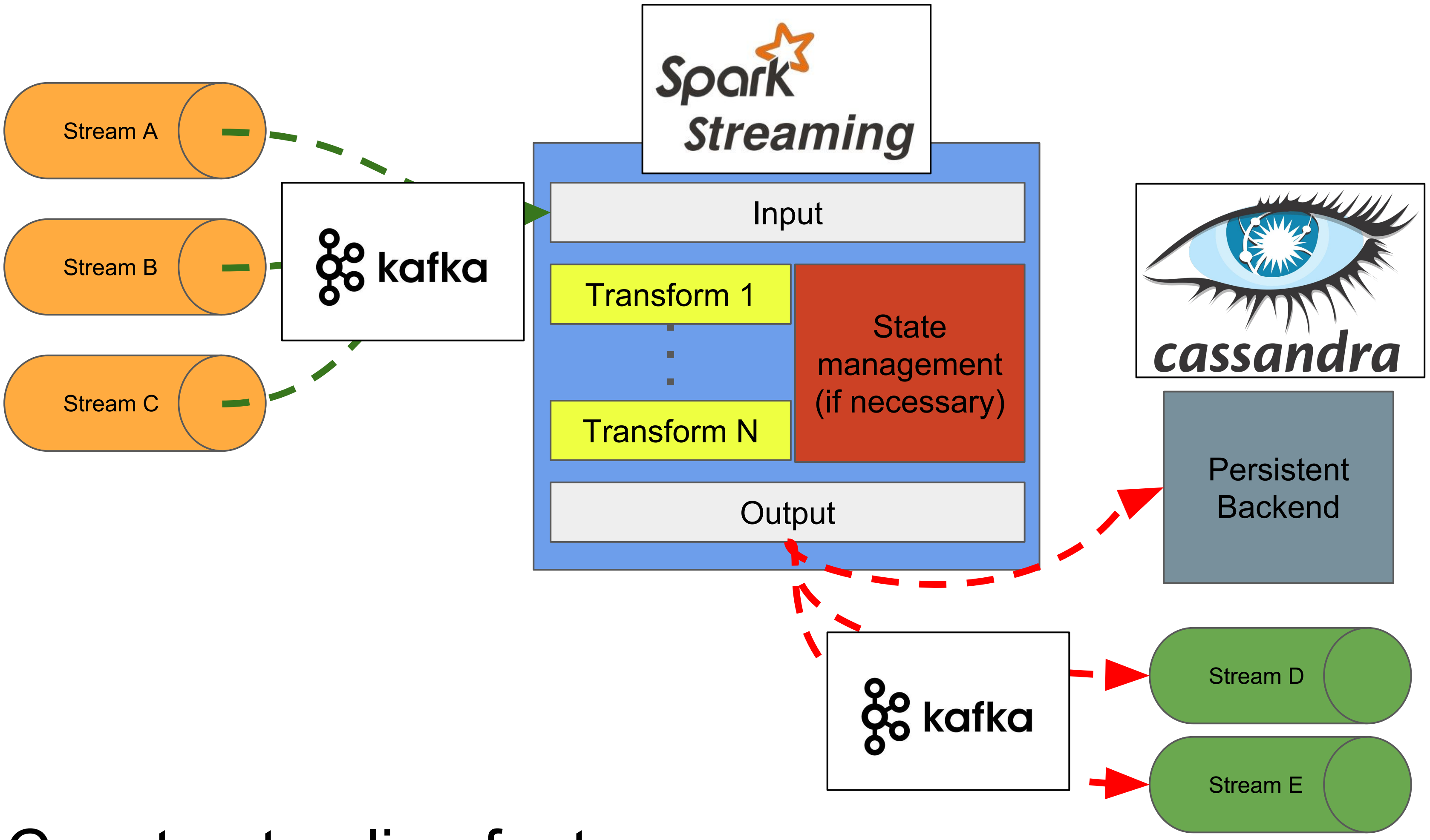
- Containers
- Health checks & automatic restarts

Nearly real-time serving

- Features available as early as possible for online prediction (same session)
- Delays in retrieving features: milliseconds

Which transformations?

- Streaming world
- Stateless (maps, filters, reductions)
- Stateful (counters, windows)



Construct online features

FEATURE PORTAL

Feature discoverability

- See available features
- Understand which features are available online and/or offline
- Understand quality of a feature

Feature semantics & systems view

- Understand which data is available at which point in time and where
- Understand its meaning & semantic consistency over time

Feature reuse

- Reduce time from idea to experiment
- Reuse code between offline and online feature generation
- Reuse data sources/streams

Feature ownership

- Ensure quality of features
- Add monitoring to enforce quality
- Collect statistics (e.g., distribution) to gain insights

Feature Store

Models deployment

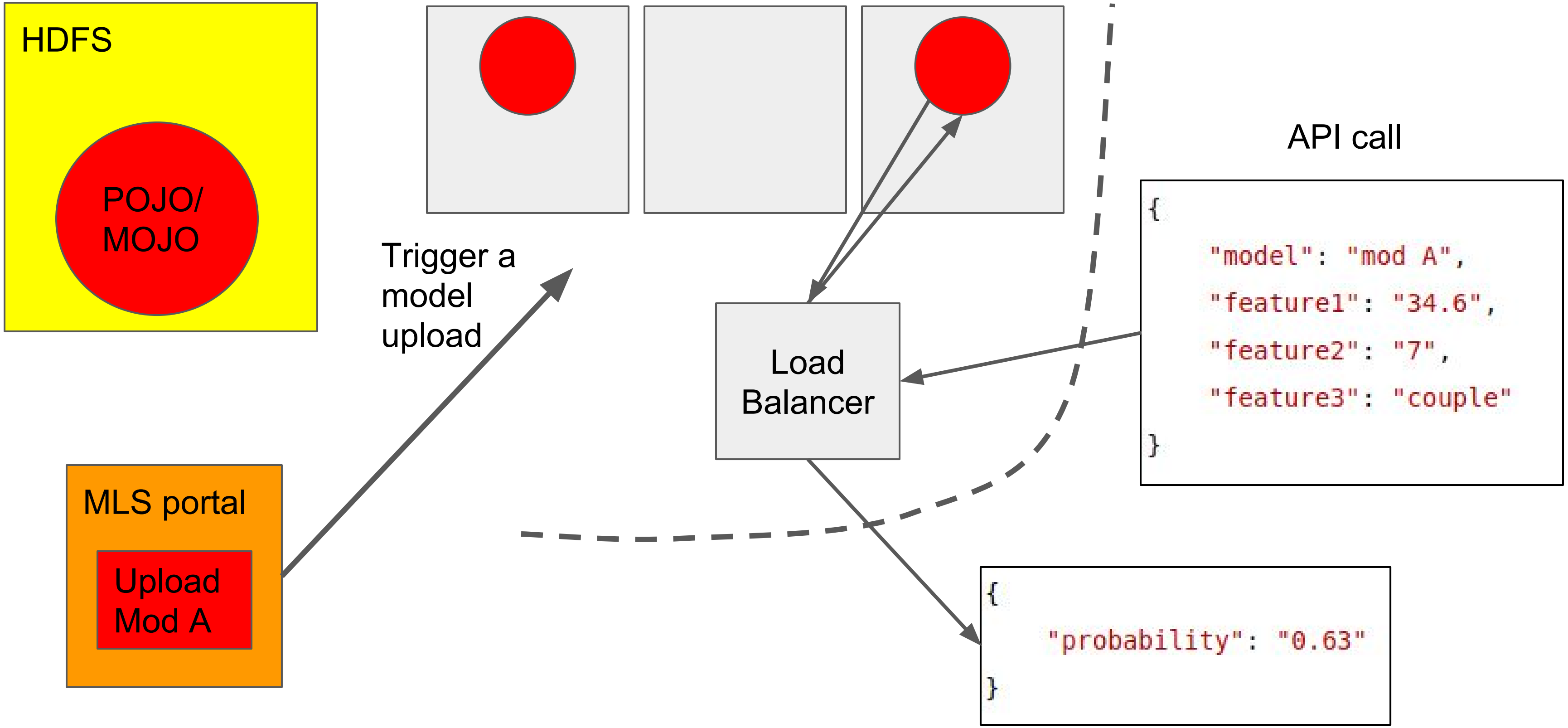
All models can be exported into a compiled set of Java classes (MOJO/POJO), saved to HDFS at the end of the training with Sparkling Water.

How to deploy:

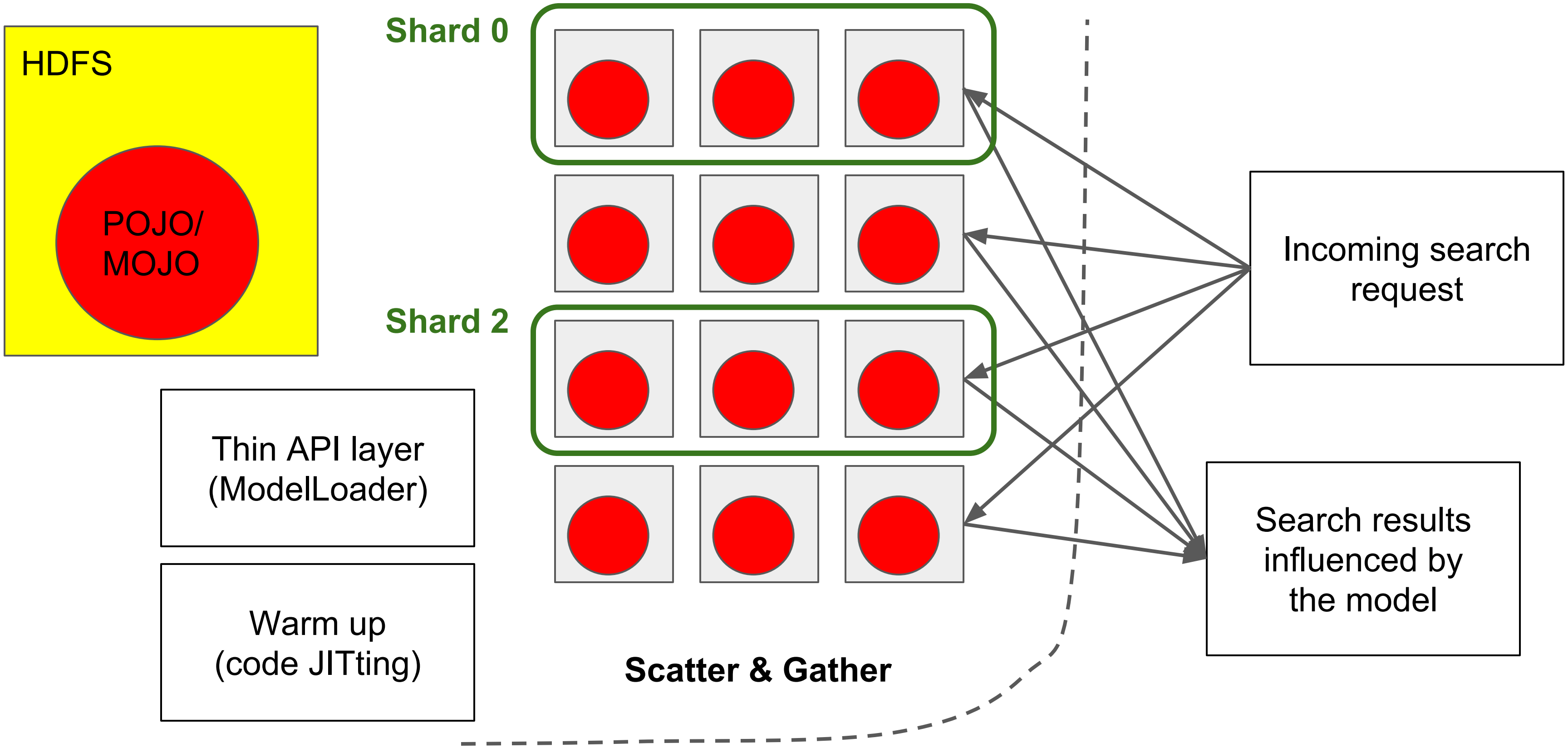
1. Dedicated service for model deploying/serving in B.com
2. Embed H2O model in some of our services

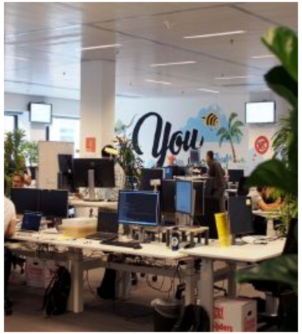
Deployment modes

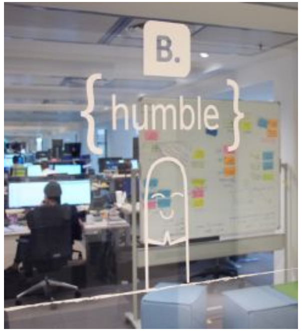
1. Dedicated model service



2. Embed model in service







Thank you all for joining our talk!

We are hiring: Join the world's #1 website for booking hotels and other accommodations

https://workingatbooking.com

Luca Falsina - luca.falsina@booking.com
Ben Teeuwen - ben.teeuwen@booking.com

Booking.com