



Scaling Machine Learning at Booking.com

Brammert Ottens, Booking.com

Luca Falsina, Booking.com

#ML8SAIS

Booking.com

Who are we?



Brammert Ottens
Senior Data Scientist



Luca Falsina
Senior Developer

Booking.com

- One of the **largest** travel e-commerce companies in the world
- **15.000** employees in 198 offices in 70 countries worldwide
- **133,577** destinations in 228 countries and territories worldwide
- Each day, more than **1,550,000** room nights are reserved on our platform

Outline

- Datascience @Booking.com
- The Machine Learning (ML) pipeline
- Offline ML
- Online model serving
- Model & features monitor and discoverability
- Future directions

Outline

- **Datascience @Booking.com**
- The Machine Learning (ML) pipeline
- Offline ML
- Online model serving
- Model & features monitor and discoverability
- Future directions

Datascience @ Booking.com

- We have a community of over 200 data scientists (DS)
- Each with different needs, expertise and preferred toolkits and methodologies

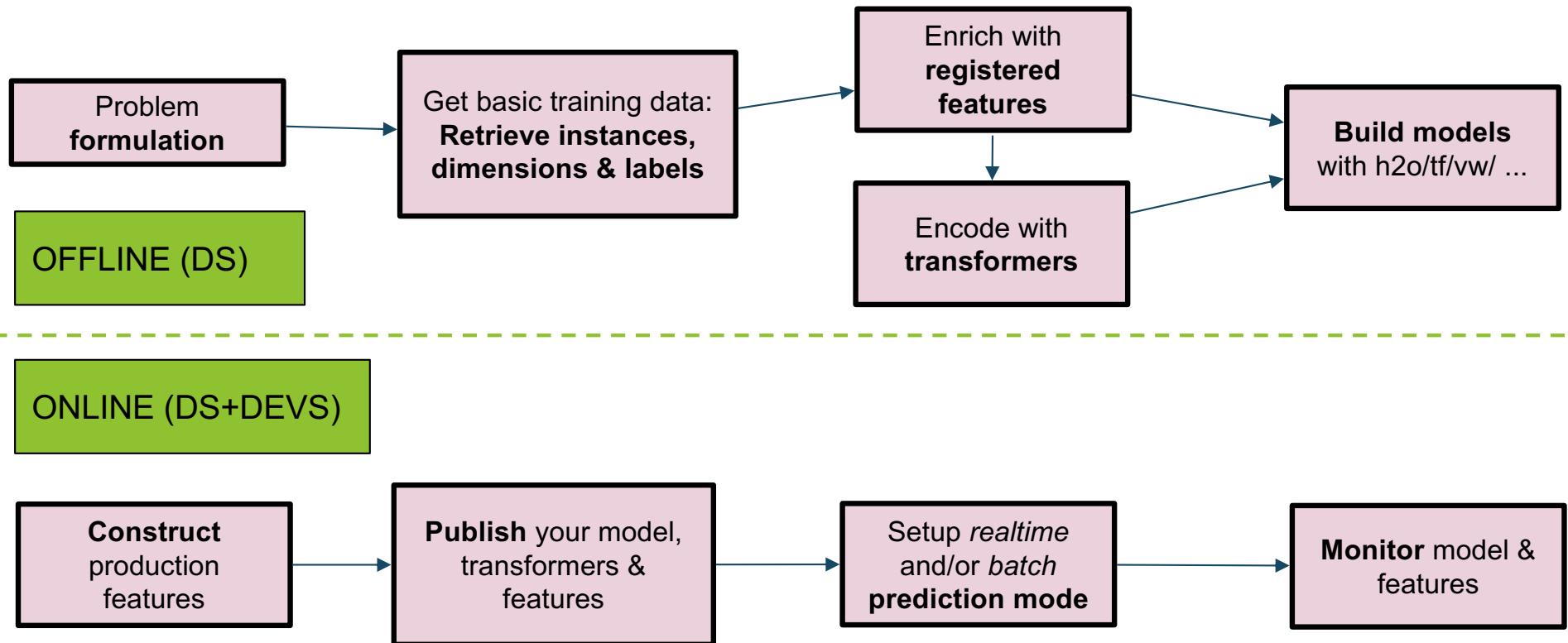
The mission of our track is to help scaling ML across our organization. How?

- Identify **common problems** (i.e. historical feature reconstruction)
- **Implement infra and tooling** to solve them
- **Train, educate and consult** internal users on these solutions

Outline

- DataScience @Booking.com
- **The Machine Learning (ML) pipeline**
- Offline ML
- Online model serving
- Model & features monitor and discoverability
- Future directions

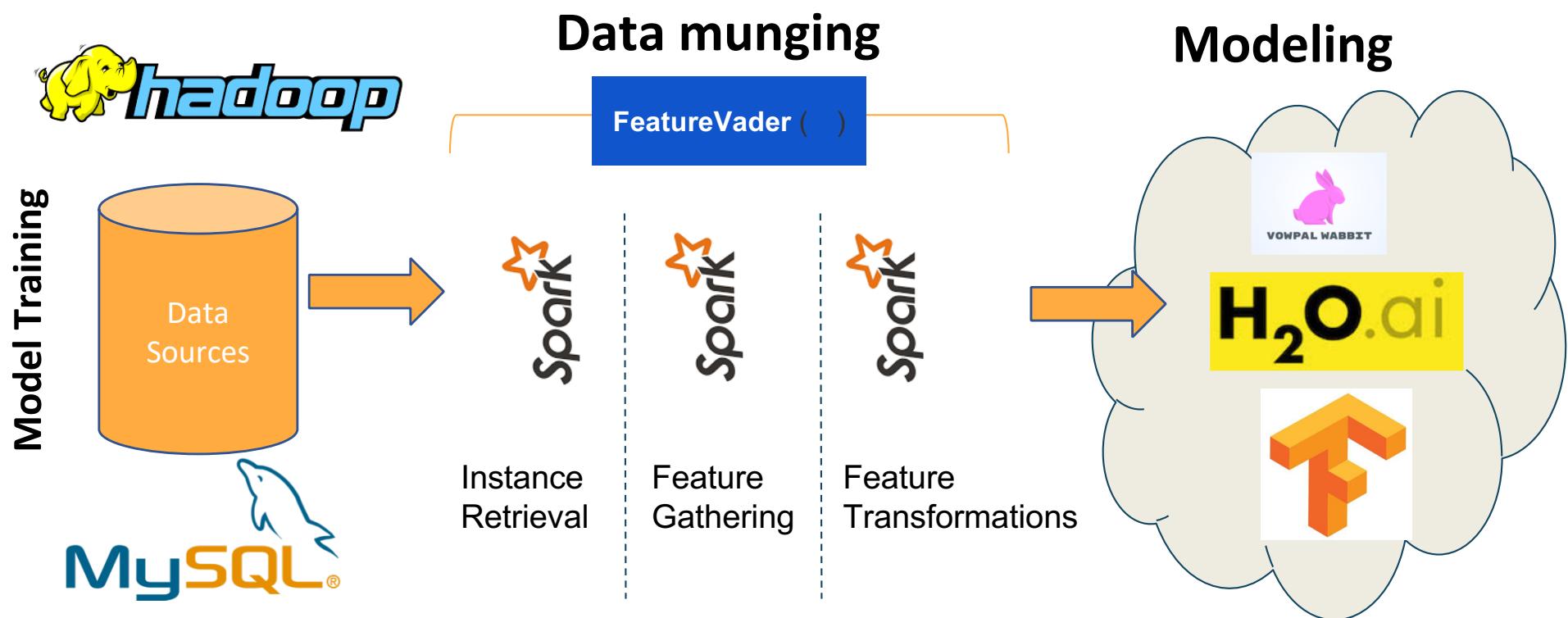
The “general” ML pipeline



Outline

- DataScience @Booking.com
- The Machine Learning (ML) pipeline
- **Offline ML**
- Online model serving
- Model & features monitor and discoverability
- Future directions

The “offline” world



OFFLINE DATA MUNGING

1. Extract relevant data
from events

2. Feature engineering

3. Time coordinate
matching

4. Scalability

Offline feature engineering

- **FeatureVader** (containing a feature registry):
 - Custom Spark ML Transformer:

```
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)
```

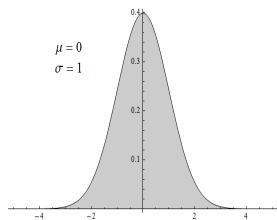
userId	time	label
1001	25	1
1002	36	0



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

Feature transformations

- Raw features are not enough.
- Transformations greatly increase model performance:



Feature transformations (WIP)

```
testimator = TargetEstimator()  
    .setFeature("feature1")  
    .setEncodedFeature("feature1_TE")  
    .setResponse("label")  
  
pipeline = new PipeLine(stages=[testimator])  
  
pipeline_model = pipeline.fit(train)  
  
pipelineWriter = PipelineWriter(pipelinemodel)  
  
pipelineWriter.writeTo("pipeline_model.bojo")
```

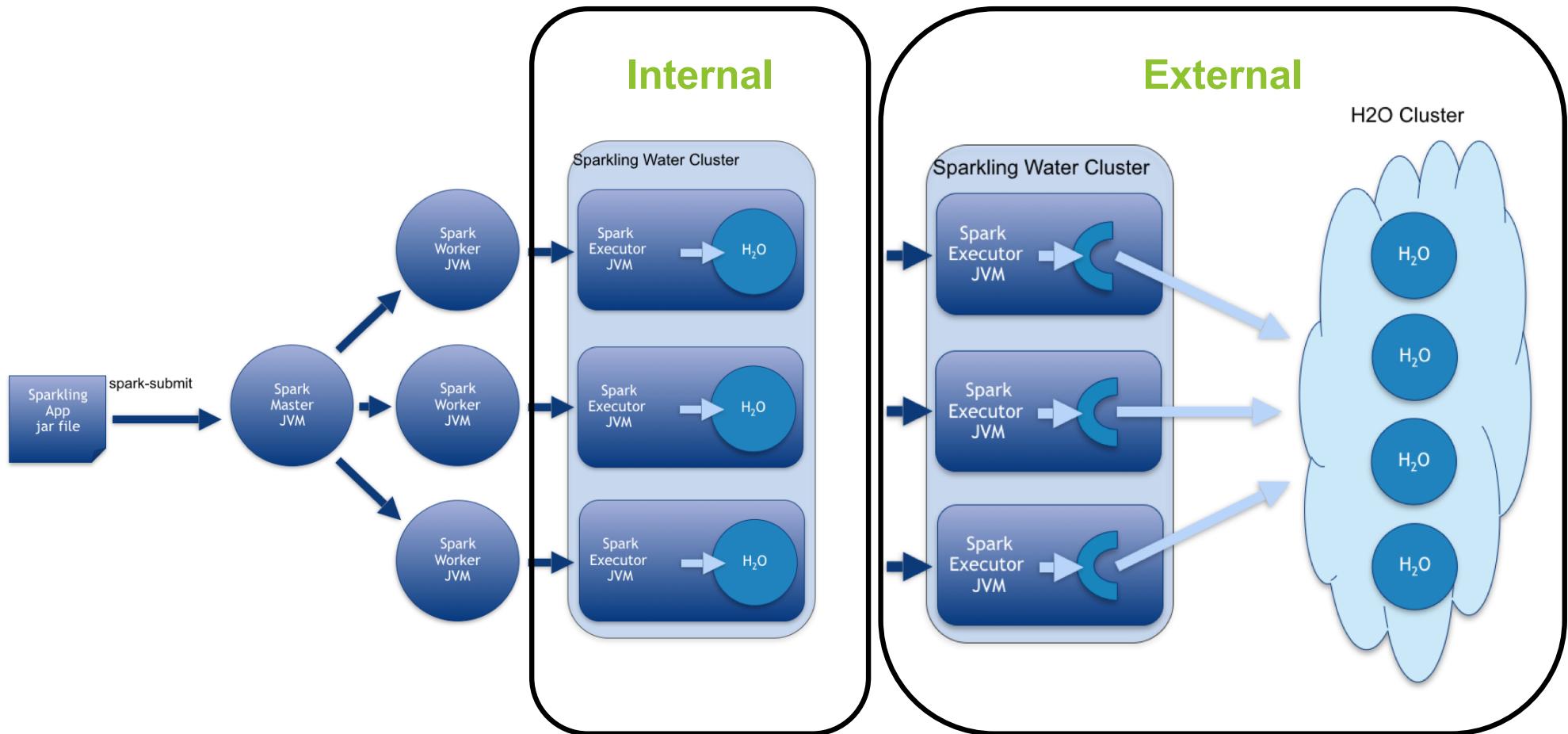
Modeling

Several libraries in use at Bookina.com



Modeling (H2O)

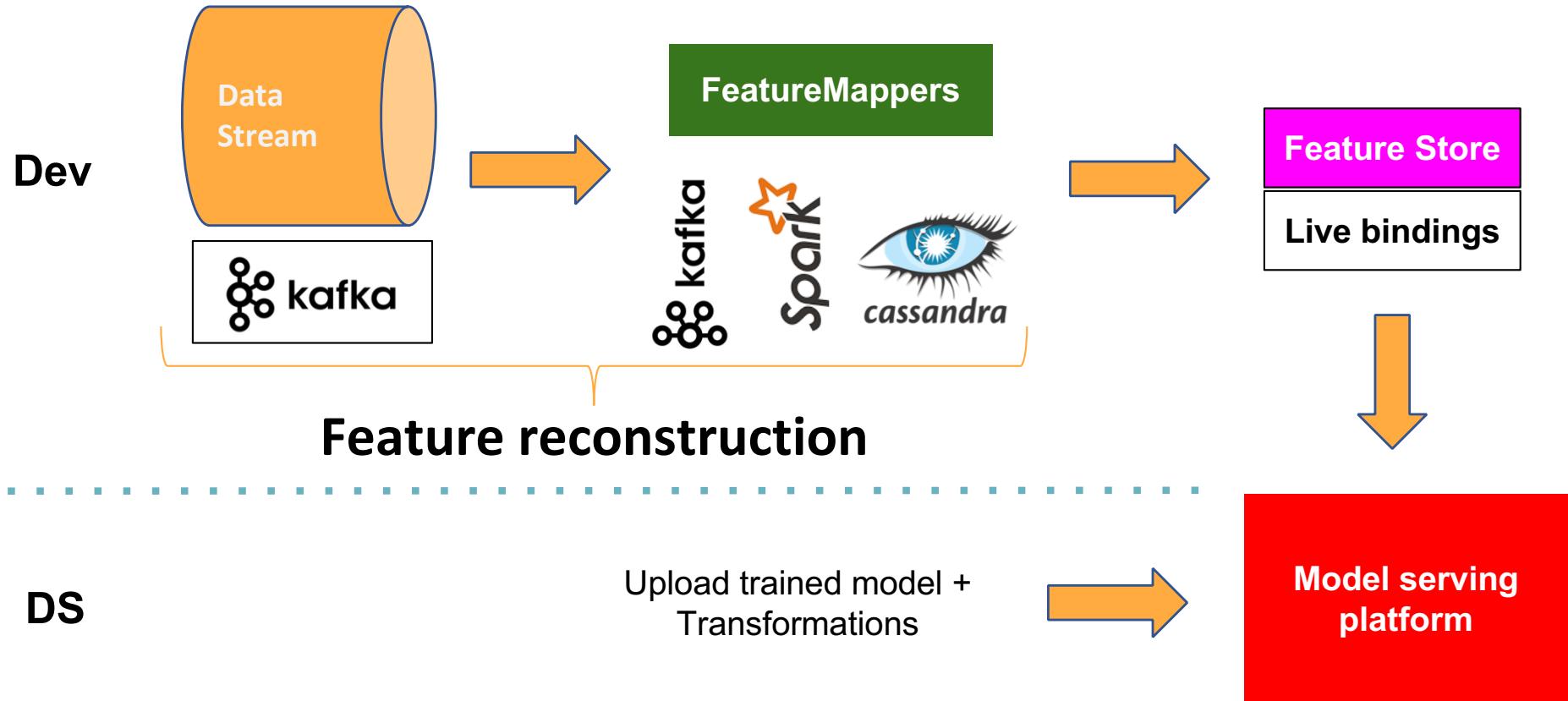
- Sparkling water
 - H2O integration with Spark
 - Best of both worlds: data munging in Spark + model training in H2O
 - Scalable to very large datasets (Billions of rows)
 - Distributed training
 - Internal vs external backend mode



Outline

- DataScience @Booking.com
- The Machine Learning (ML) pipeline
- Offline ML
- **Online model serving**
- Model & features monitor and discoverability
- Future directions

The “online” world



ONLINE DATA MUNGING

FeatureMappers

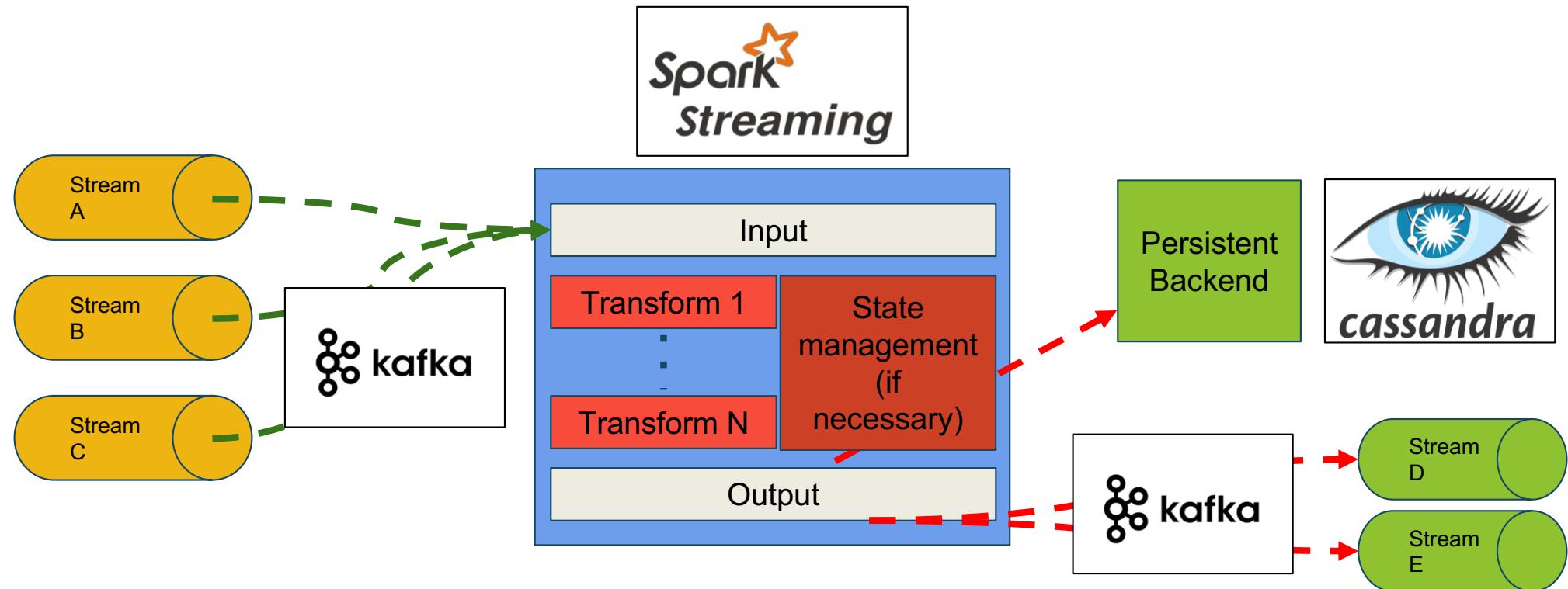
 SPARK+AI
SUMMIT 2018

1. Nearly real-time processing
2. State(less/full) transformations
3. Containerized infrastructure
4. Nearly real-time serving

#ML8SAIS

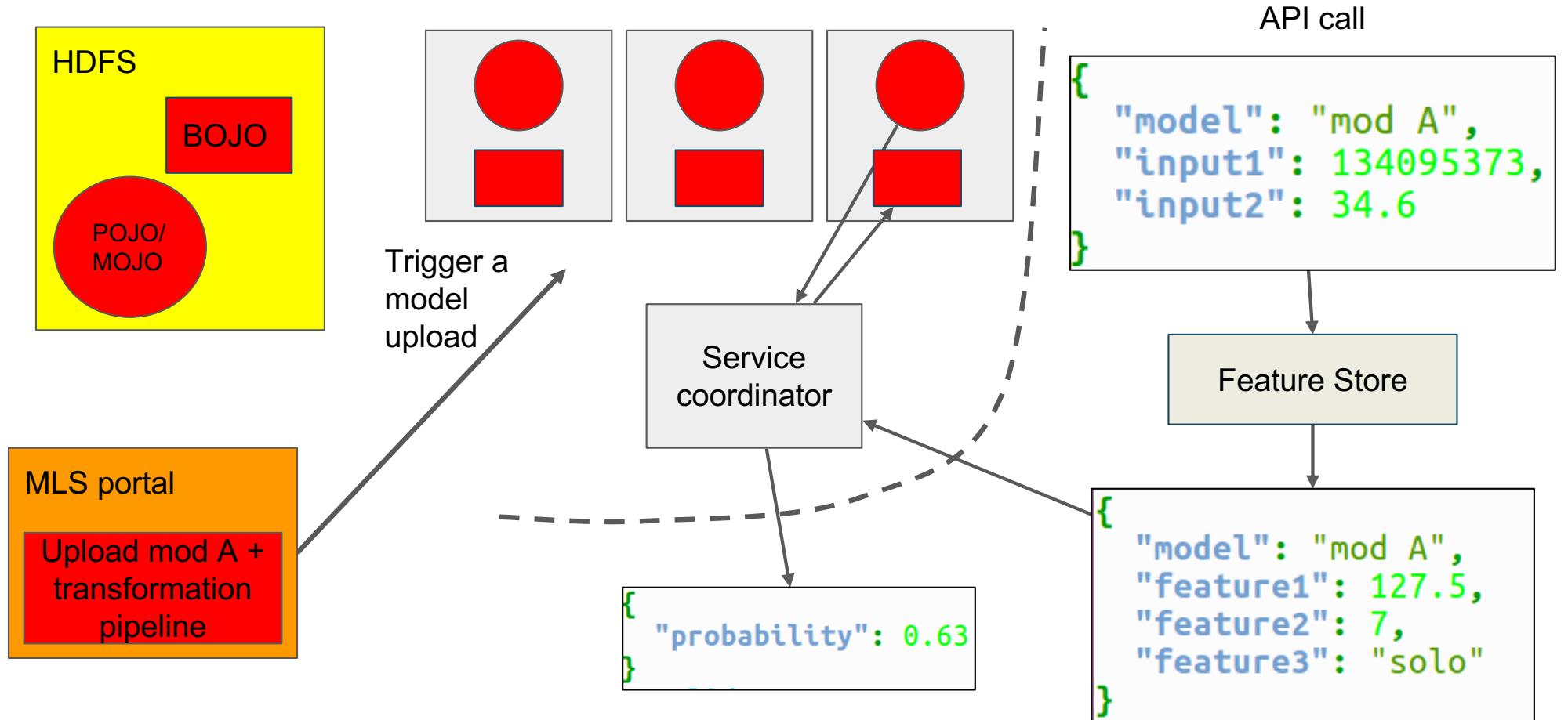
20

Online feature construction



Model serving platform

- Supports several model types (H2O, VW, TF, csv...)
- Allows both **single** and **batch** predictions
- Integration with *live feature bindings*



Example: Model upload & inference from frontend hosts

Outline

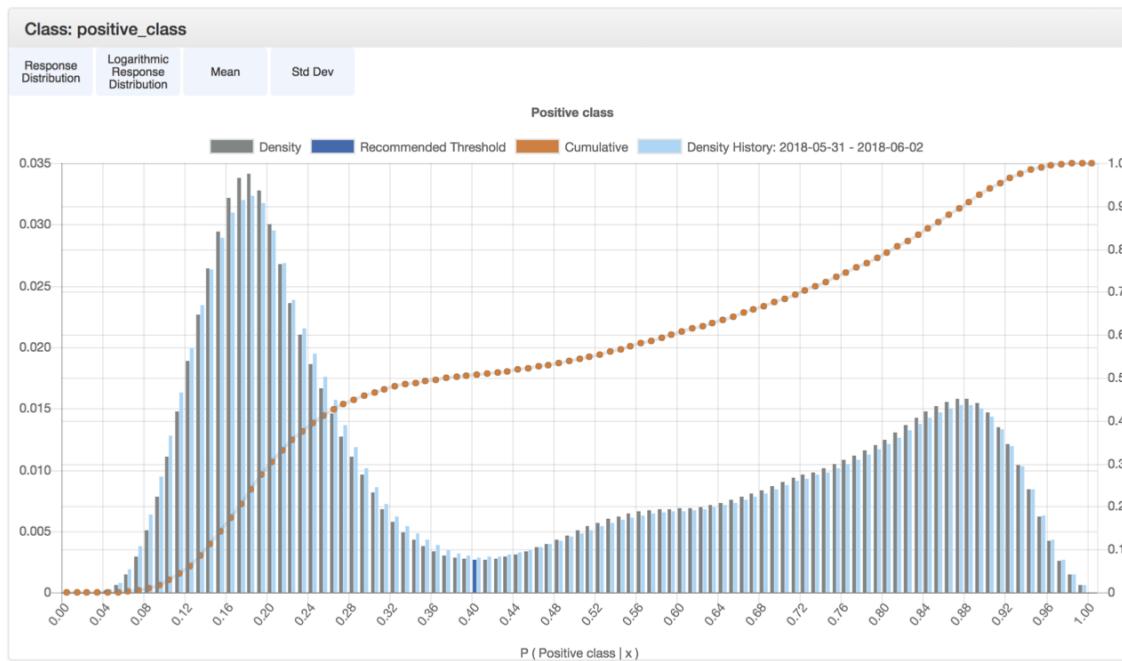
- DataScience @Booking.com
- The Machine Learning (ML) pipeline
- Offline ML
- Online model serving
- **Model & features monitor and discoverability**
- Future directions

Model portal

The screenshot shows a dark-themed web application interface. On the left, a sidebar menu lists 'RS', 'Models', 'Features', 'Upload Model', 'Feature mappers', and 'Create Placeholder'. The main area is titled 'RS (RIDERLESS SWAMP-FEVER) Upload Model'. It contains a 'HDFS path' input field containing '/tmp/mymodel/'. Below it is a note: 'Be sure to name folder nicely as it is going to be used as your model id. Proper names help your colleagues to discover and use your models. To get more info on formats check [this wiki page](#)'. A 'Hadoop Cluster' dropdown menu is set to 'LHR4'. Under 'Model type', the 'H2O mojo' option is selected (indicated by a blue radio button). Other options include 'CSV', 'TensorFlow (Saved Model)', and 'Lookup Tables'. At the bottom is a blue 'Upload' button.

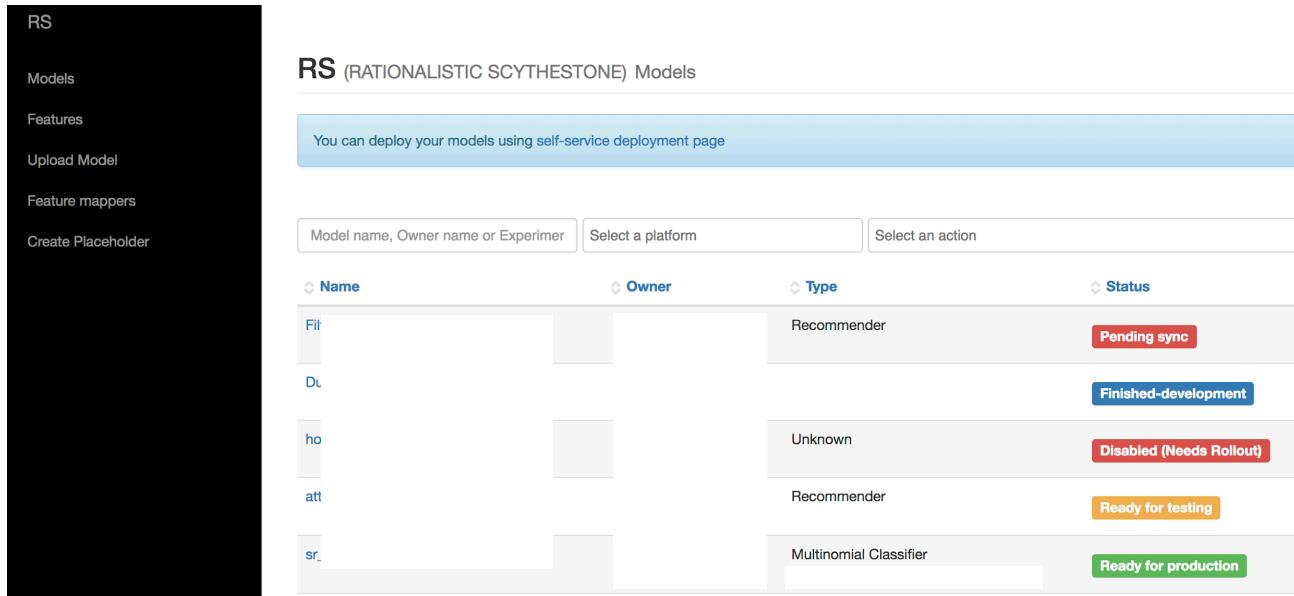
- Allow to define models and upload new versions

Model portal



- Provide monitoring

Model portal



The screenshot shows a user interface for a model portal. On the left, there is a dark sidebar with the acronym 'RS' at the top and a list of menu items: Models, Features, Upload Model, Feature mappers, and Create Placeholder. The main area has a header 'RS (RATIONALISTIC SCYTHESTONE) Models'. Below the header is a message: 'You can deploy your models using self-service deployment page'. There are three input fields: 'Model name, Owner name or Experimenter', 'Select a platform', and 'Select an action'. A table lists five models:

Name	Owner	Type	Status
Fil		Recommender	Pending sync
Du			Finished-development
ho		Unknown	Disabled (Needs Rollout)
att		Recommender	Ready for testing
sr.		Multinomial Classifier	Ready for production

- Allow to discover and reuse existing models

Feature store portal

The screenshot shows a search interface for feature names. The search bar contains 'outstanding'. The results list three entries under 'Feature names': 'ad' (repeated twice) and 'af'. Each entry has a small icon to its right. Below the results, sections for 'Available offline implementations in Vader:' and 'Available online implementations in RS:' are shown, each listing a single item. At the bottom, a section titled 'Used in models:' is partially visible.

RS

What if I had already named my features differently? ▾

Q outstanding Search Dimension and Required Keys

Feature names

ad

ad

af

Available offline implementations in Vader:

- Dimension(required key): affiliate_id
Owner:
Description:
Source table in Hive: feature_store.factlog_sherlock_hose_simulated

Available online implementations in RS:

- Required keys: affiliate_id_features(affiliate_id)
Description: affiliate_id outstanding count
Data Type: Integer
Source: Frontend
Value distribution across all models: [here](#)

Used in models:

- Allow to discover and reuse features

Future directions

- Make the full pipeline smoother for faster iterations
- Limit involvement of devs (scarce resource) for ML work
 - Automatic reconstruction of features online from offline



Thank you for joining our talk!

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

<https://workingatbooking.com>

Luca Falsina
Brammert Ottens

[luca.falsina@
brammert.ottens@](mailto:luca.falsina@booking.com)

Booking.com