INFORMATIONSFABRIK

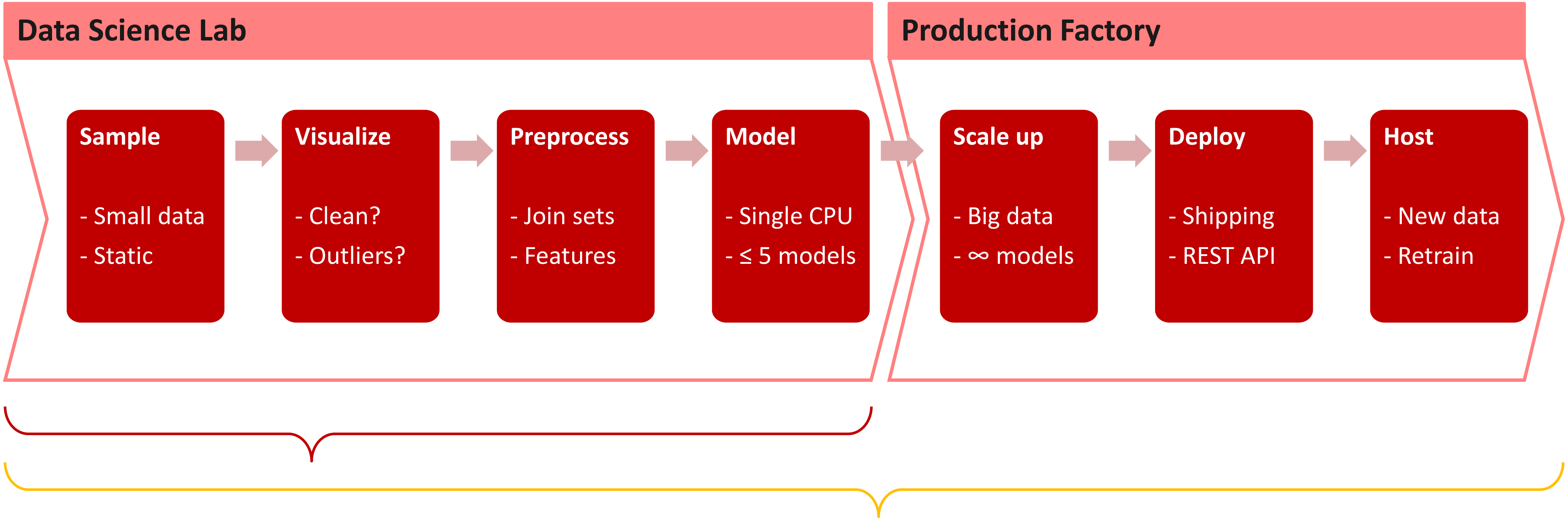
DATEN VERSTEHEN, ENTSCHEIDUNGEN TREFFEN

Beyond Notebooks: how to go to production with h2o

Dr. Thorben Jensen

The Problem:

Lab vs. Factory



Data Science Lab

Sample
- Small data
- Static

Visualize
- Clean?
- Outliers?

Preprocess
- Join sets
- Features

Model
- Single CPU
- ≤ 5 models

Production Factory

Scale up
- Big data
- ∞ models

Deploy
- Shipping
- REST API

Host
- New data
- Retrain

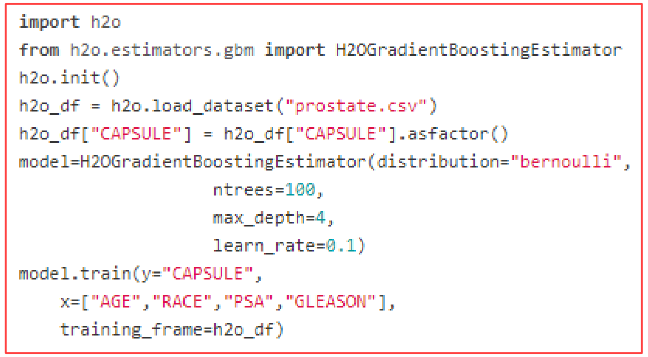
knitr

jupyter

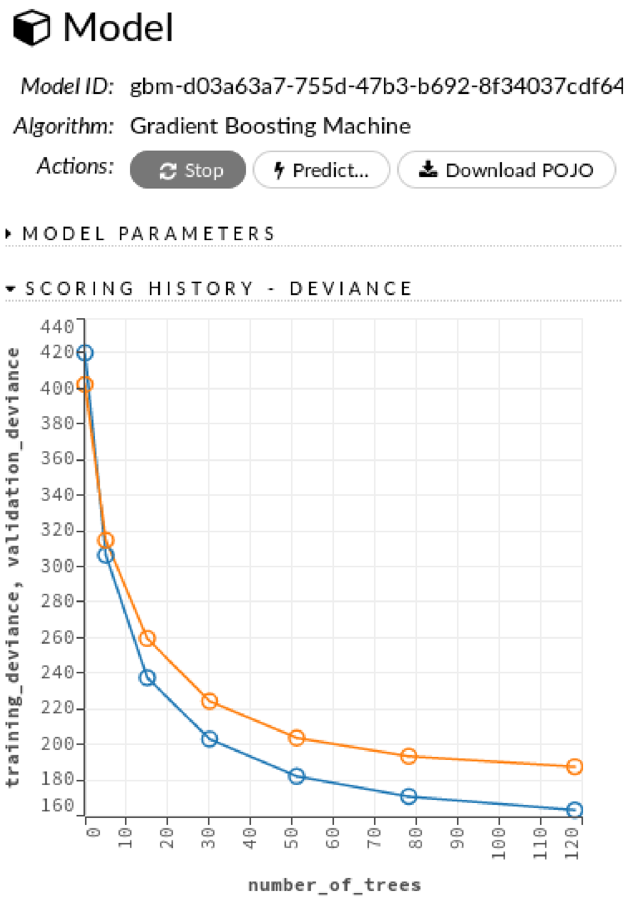
H₂O.ai

What‘s h2o?

- *“H2O is an open source, in-memory, distributed, … and scalable machine learning … platform … to build machine learning models … and easy productionalization …” (docs.h2o.ai)*



```
import h2o
from h2o.estimators.gbm import H2OGradientBoostingEstimator
h2o.init()
h2o_df = h2o.load_dataset("prostate.csv")
h2o_df["CAPSULE"] = h2o_df["CAPSULE"].asfactor()
model=H2OGradientBoostingEstimator(distribution="bernoulli",
                        ntrees=100,
                        max_depth=4,
                        learn_rate=0.1)
model.train(y="CAPSULE",
     x=["AGE","RACE","PSA","GLEASON"],
     training_frame=h2o_df)
```



Model

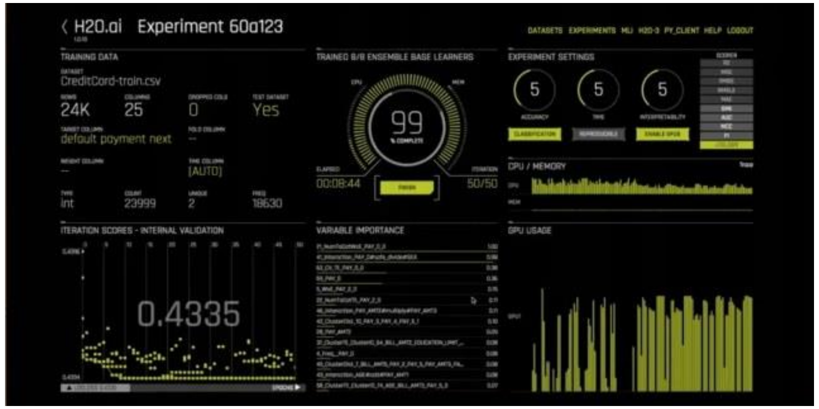
Model ID: gbm-d03a63a7-755d-47b3-b692-8f34037cdf64

Algorithm: Gradient Boosting Machine

Actions:

Spark + H2O

SPARKLING WATER



H2O.ai Experiment 60a123

TRAINING DATA

DATASET CreditCard-train.csv

ROWS 24K COLUMNS 25 DROPPED COLS 0 TEST DATASET Yes

TARGET COLUMN default payment next FOLD COLUMN —

WEIGHT COLUMN — TIME COLUMN [AUTO]

TYPE int COUNT 23999 UNIQUE 2 FREQ 18630

TRAINED 8/8 ENSEMBLE BASE LEARNERS

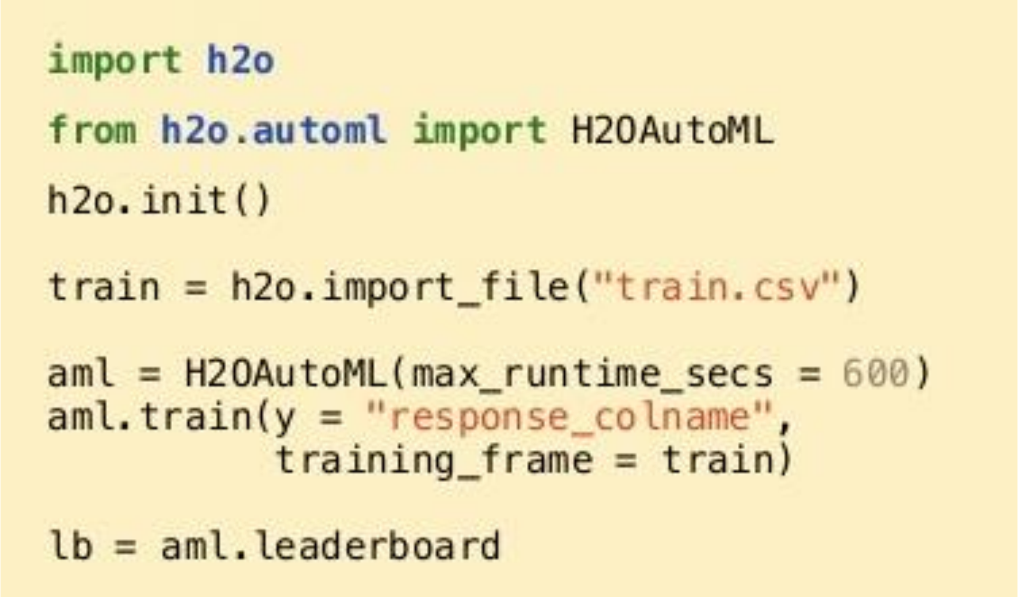
99 % COMPLETE

ELAPSED 00:08:44 ITERATION 50/50

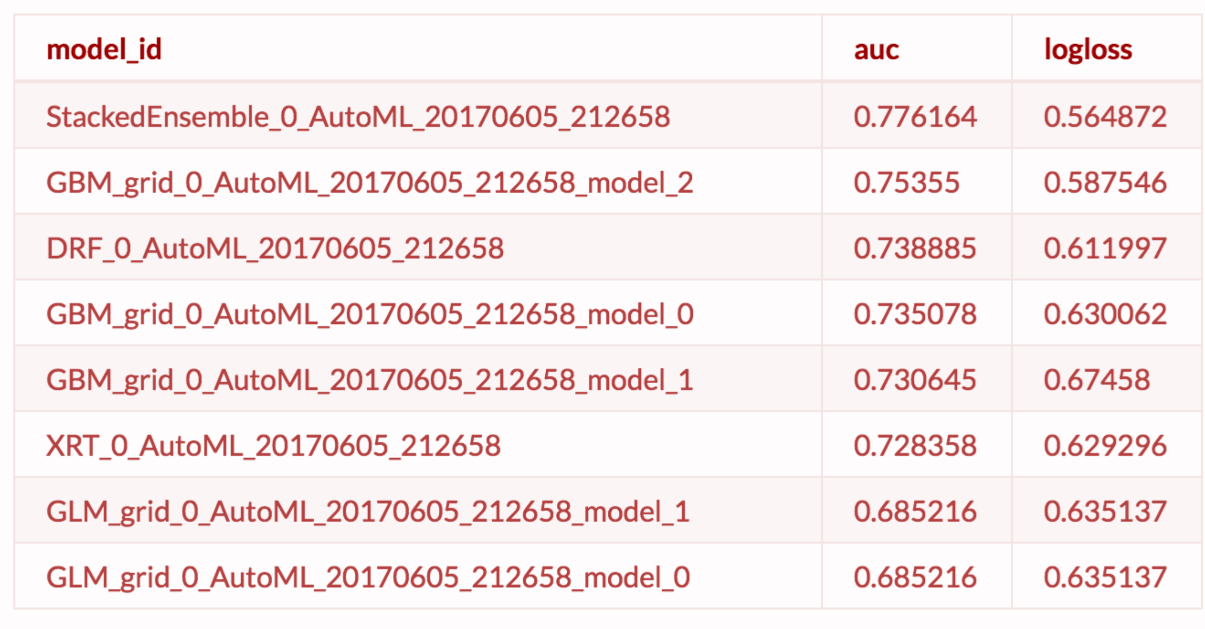
EXPERIMENT SETTINGS

5 ACCURACY 5 TIME 5 INTERPRETABILITY

Automated machine learning (AutoML)

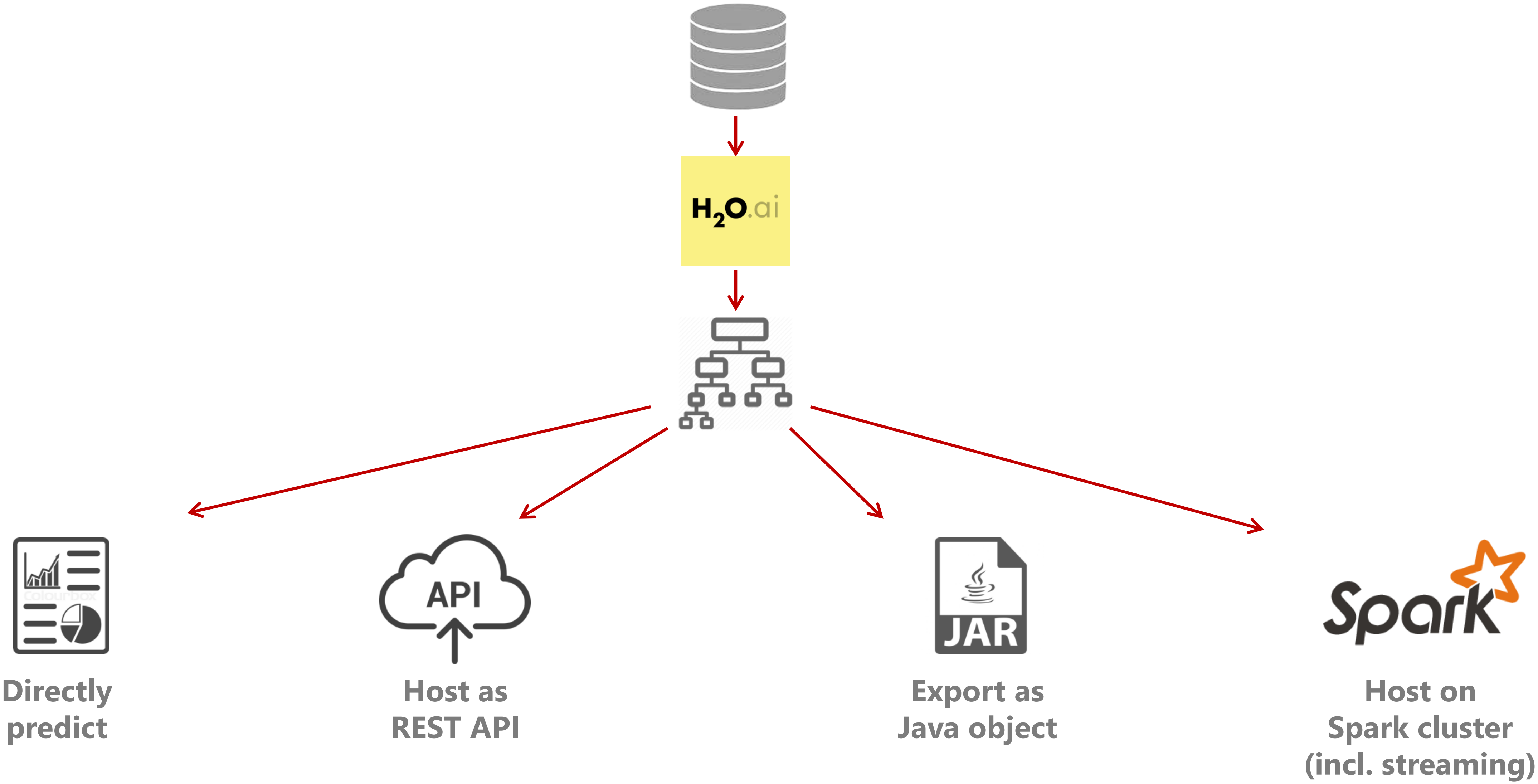


```
import h2o
from h2o.automl import H2OAutoML
h2o.init()

train = h2o.import_file("train.csv")

aml = H2OAutoML(max_runtime_secs = 600)
aml.train(y = "response_colname",
          training_frame = train)

lb = aml.leaderboard
```



model_id	auc	logloss
StackedEnsemble_0_AutoML_20170605_212658	0.776164	0.564872
GBM_grid_0_AutoML_20170605_212658_model_2	0.75355	0.587546
DRF_0_AutoML_20170605_212658	0.738885	0.611997
GBM_grid_0_AutoML_20170605_212658_model_0	0.735078	0.630062
GBM_grid_0_AutoML_20170605_212658_model_1	0.730645	0.67458
XRT_0_AutoML_20170605_212658	0.728358	0.629296
GLM_grid_0_AutoML_20170605_212658_model_1	0.685216	0.635137
GLM_grid_0_AutoML_20170605_212658_model_0	0.685216	0.635137

Options for „productionalization“



Directly predict

Host as REST API

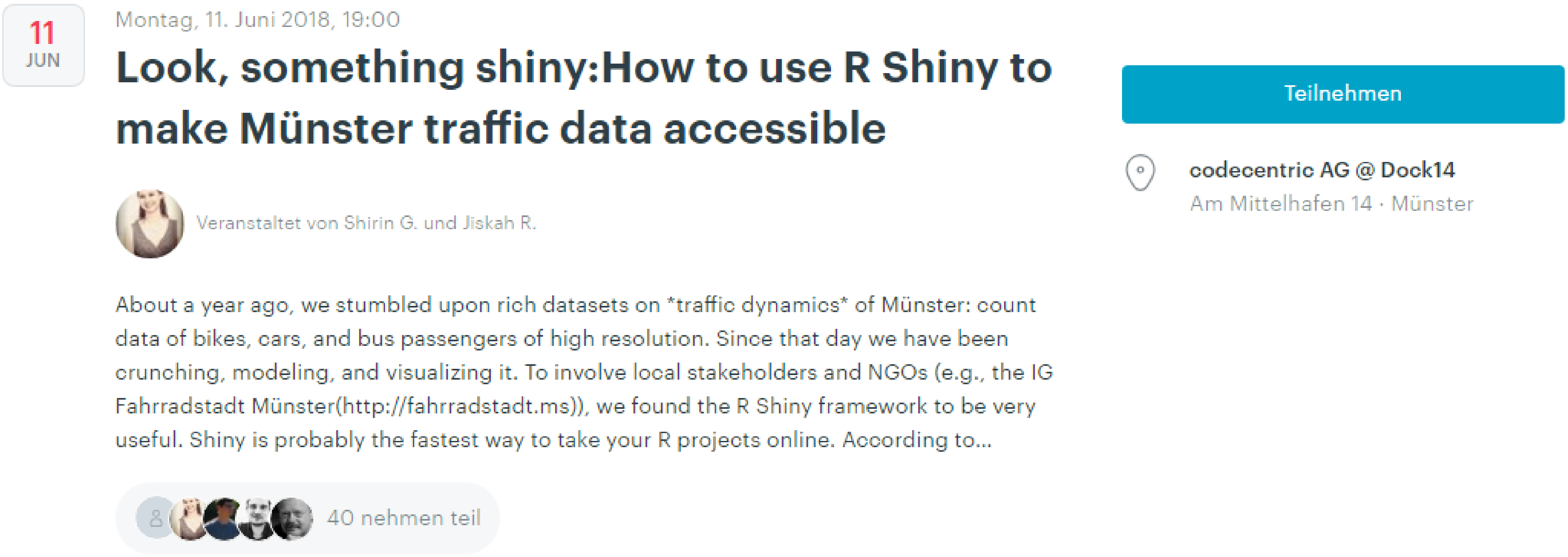
Export as Java object

Host on Spark cluster (incl. streaming)

Checklist: how to get ready for production

- ✓ **Stop using notebooks, asap**
- ✓ **Use modern DevOps** (git, test automation, containers)
- ✓ **Automate data and modeling pipelines** (Airflow, Luigi)
- ✓ ***„Never get high on your own CPU supply“*** (AWS, Azure)
- ✓ **Beat benchmark with *H2OAutoML* and move on**
- ✓ ***„Never trust no training sample“*** (automated retraining)

Coming up: **talk on Shiny dashboards with R**



Montag, 11. Juni 2018, 19:00

Look, something shiny:How to use R Shiny to make Münster traffic data accessible

Veranstaltet von Shirin G. und Jiskah R.

codecentric AG @ Dock14
Am Mittelhafen 14 · Münster

About a year ago, we stumbled upon rich datasets on *traffic dynamics* of Münster: count data of bikes, cars, and bus passengers of high resolution. Since that day we have been crunching, modeling, and visualizing it. To involve local stakeholders and NGOs (e.g., the IG Fahrradstadt Münster(http://fahrradstadt.ms)), we found the R Shiny framework to be very useful. Shiny is probably the fastest way to take your R projects online. According to...

40 nehmen teil

Further reading

- http://docs.h2o.ai/h2o/latest-stable/h2o-docs/booklets/SparklingWaterBooklet.pdf
- http://rstudio.github.io/sparklyr/articles/images/deployment/amazon-emr/emrArchitecture.png
- https://www.r-bloggers.com/the-10-data-science-crack-commandments/