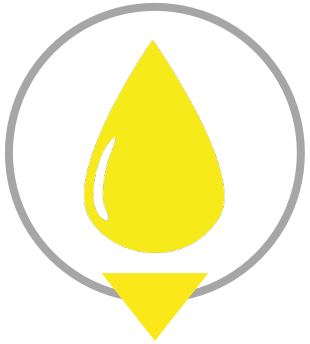
Scalable Machine Learning with H2O

Serafim Pinto.

H_2O.ai

Braga, 18 October 2017



Agenda

- What/Who is H2O.ai?
- H2O Machine Learning Software
- H2O Architecture
- H2O in R & Demo
- Ensemble
- Deep Water
- Steam for DevOps
- Auto ML
- Driverless AI

About Me

Serafim Pinto

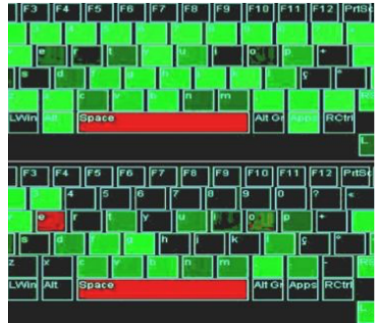
- 25 years old
- Software Engineering from University of Minho
- Postgraduate studies in **Artifical Intelligence**
- Startup
- Co-founder & CTO

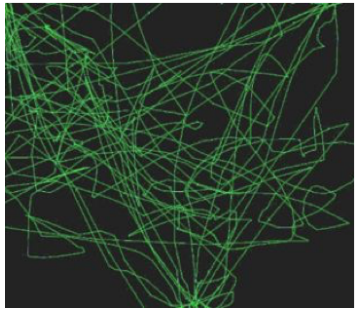
Full-time @ Performetric

---

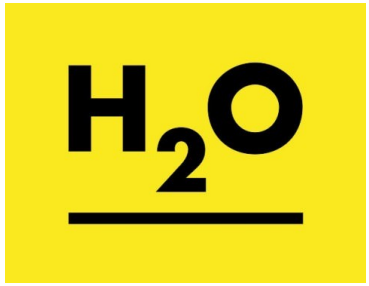
performetric

- Keyboard & Mouse
- Behavioral Analysis
- Machine Learning
- Mental Fatigue Classification





H2O Software



H2O is an open source, distributed, Java machine learning library.

APIs are available for:
R, Python, Scala & JSON

H2O Overview

Speed Matters!

- Time is valuable
- In-memory is faster
- Distributed is faster
- High speed AND accuracy

No Sampling

- Scale to big data
- Access data links
- Use all data without sampling

Interactive UI

- Web-based modeling with H2O Flow
- Model comparison

Cutting-Edge Algorithms

- Suite of cutting-edge machine learning algorithms
- Deep Learning & Ensembles
- NanoFast Scoring Engine

Current Algorithm Overview

Statistical Analysis

- Linear Models (GLM)
- Cox Proportional Hazards
- Naïve Bayes

Ensembles

- Random Forest
- Distributed Trees
- Gradient Boosting Machine
- R Package - Super Learner Ensembles

Deep Neural Networks

- Multi-layer Feed-Forward Neural Network
- Auto-encoder
- Anomaly Detection
- Deep Features

Clustering

- K-Means

Dimension Reduction

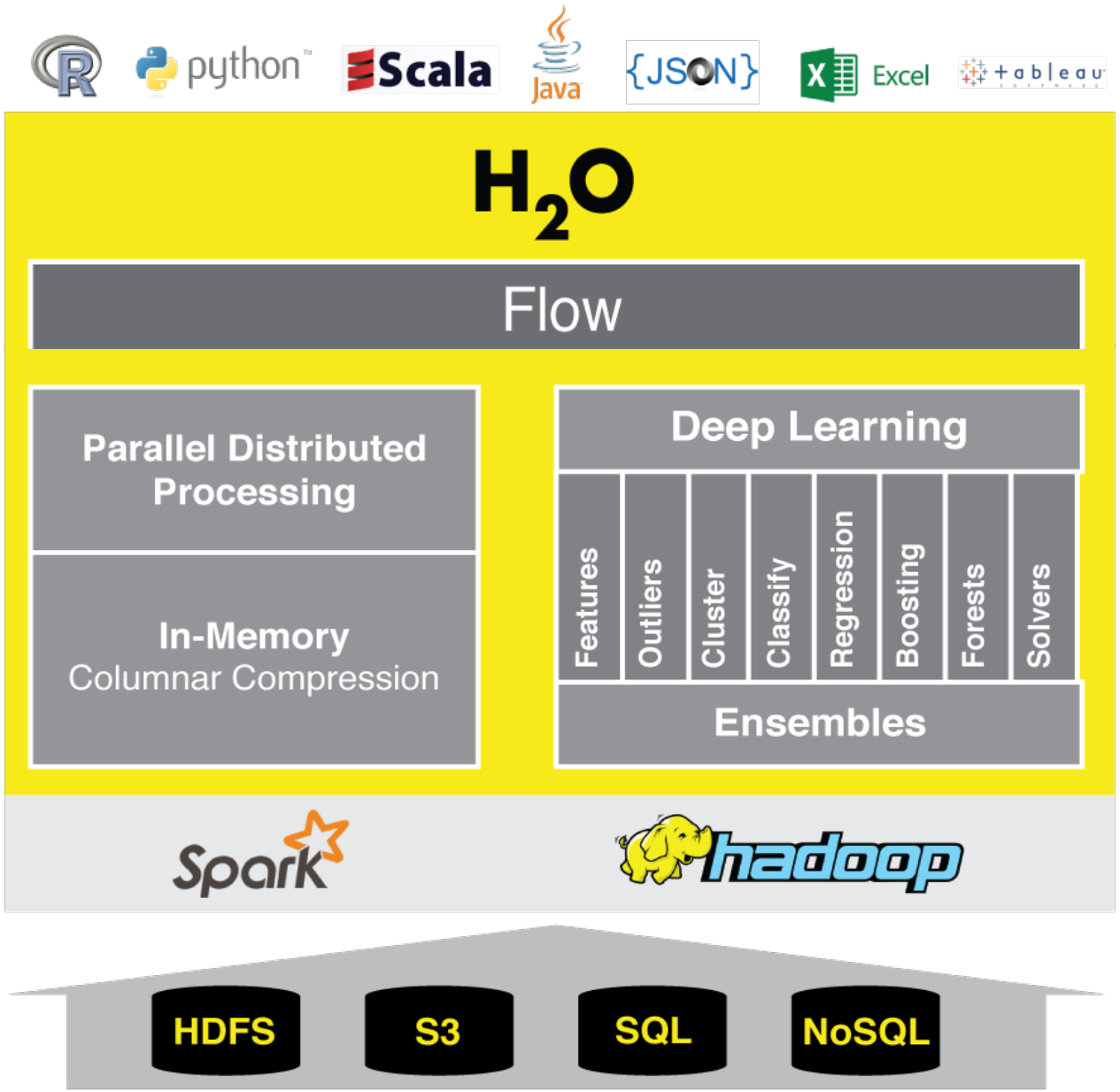
- Principal Component Analysis
- Generalized Low Rank Models

Solvers & Optimization

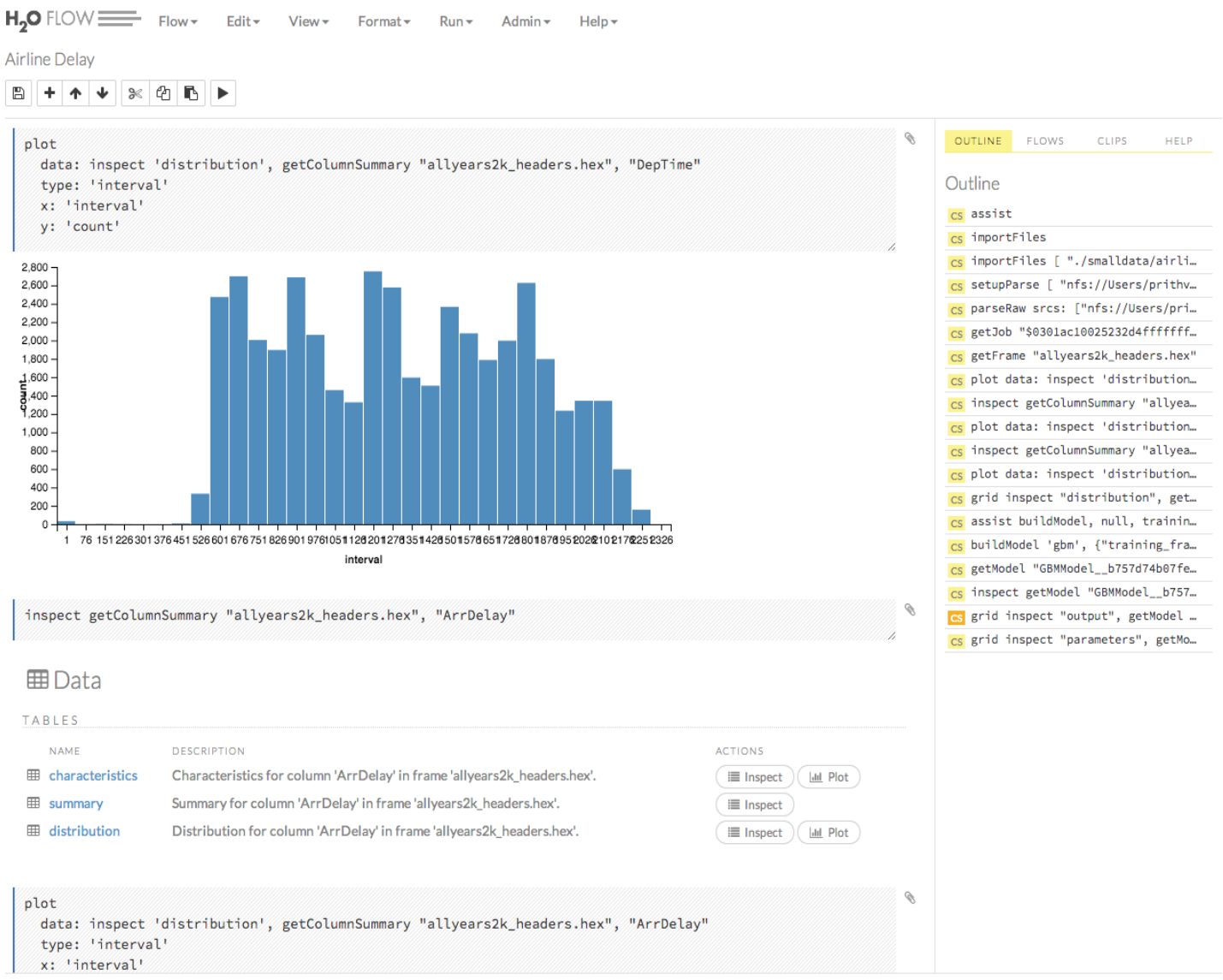
- Generalized ADMM Solver
- L-BFGS (Quasi Newton Method)
- Ordinary Least-Square Solver
- Stochastic Gradient Descent

Data Munging

- Integrated R-Environment
- Slice, Log Transform


R
python
Scala
Java
{JSON}
Excel
tableau
H2O
Flow
Parallel Distributed Processing
In-Memory
Columnar Compression
Deep Learning
Features
Outliers
Cluster
Classify
Regression
Boosting
Forests
Solvers
Ensembles
Spark
hadoop
HDFS
S3
SQL
NoSQL

H2O Flow Interface



http://h2o.ai/download

H2O

gitter join chat

H2O is an in-memory platform for distributed, scalable machine learning. H2O uses familiar interfaces like R, Python, Scala, Java, JSON and the Flow notebook/web interface, and works seamlessly with big data technologies like Hadoop and Spark. H2O provides implementations of many popular algorithms such as GBM, Random Forest, Deep Neural Networks, Word2Vec and Stacked Ensembles. H2O is extensible so that developers can add data transformations and custom algorithms of their choice and access them through all of those clients.

Data collection is easy. Decision making is hard. H2O makes it fast and easy to derive insights from your data through faster and better predictive modeling. H2O allows online scoring and modeling in a single platform.

H2O-3 (this repository) is the third incarnation of H2O, and the successor to H2O-2.

Table of Contents

https://github.com/h2oai/h2o-3

H2O Architecture

H2O Components

H2O Cluster

- Multi-node cluster with shared memory model.
- All computations in memory.
- Each node sees only some rows of the data.
- No limit on cluster size.

Distributed Key Value Store

- Objects in the H2O cluster such as data frames, models and results are all referenced by key.
- Any node in the cluster can access any object in the cluster by key.
- The H2O K/V Store is a classic peer-to-peer distributed hash table.

H2O Frame

- Distributed data frames (collection of vectors).
- Columns are distributed (across nodes) arrays.

Data in H2O

Highly Compressed

- We read data fully parallelized from: HDFS, NFS, Amazon S3, URLs, URIs, CSV, SVMLight.
- Data is highly compressed (about 2-4 times smaller than gzip).

Speed

- Memory bound, not CPU bound.
- If data accessed linearly, as fast as C or Fortran.
- Speed = data volume / memory bandwidth
- ~50GB / sec (varies by hardware).

Distributed H2O Frame

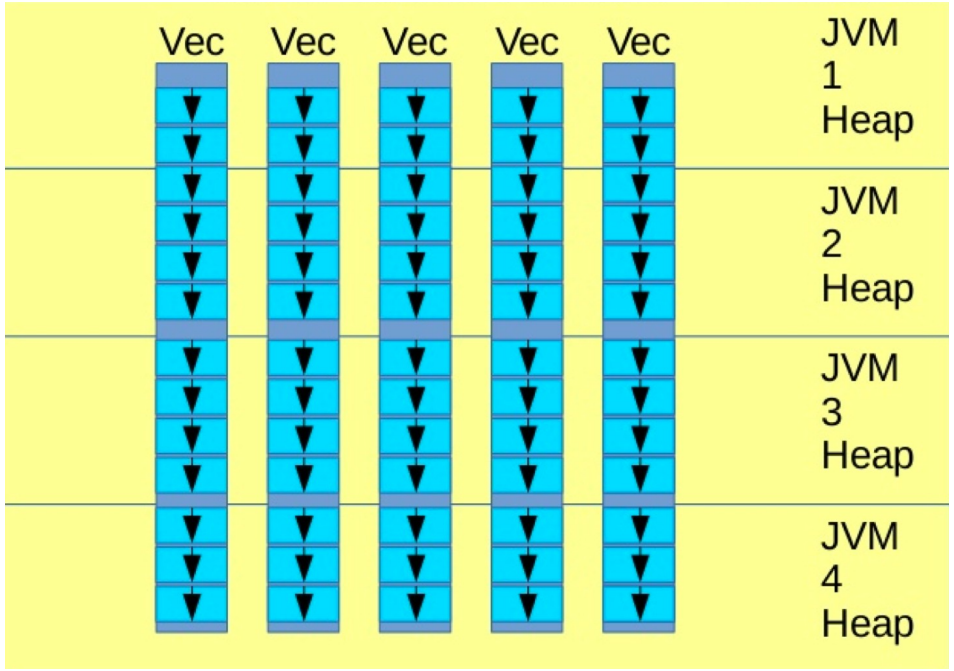
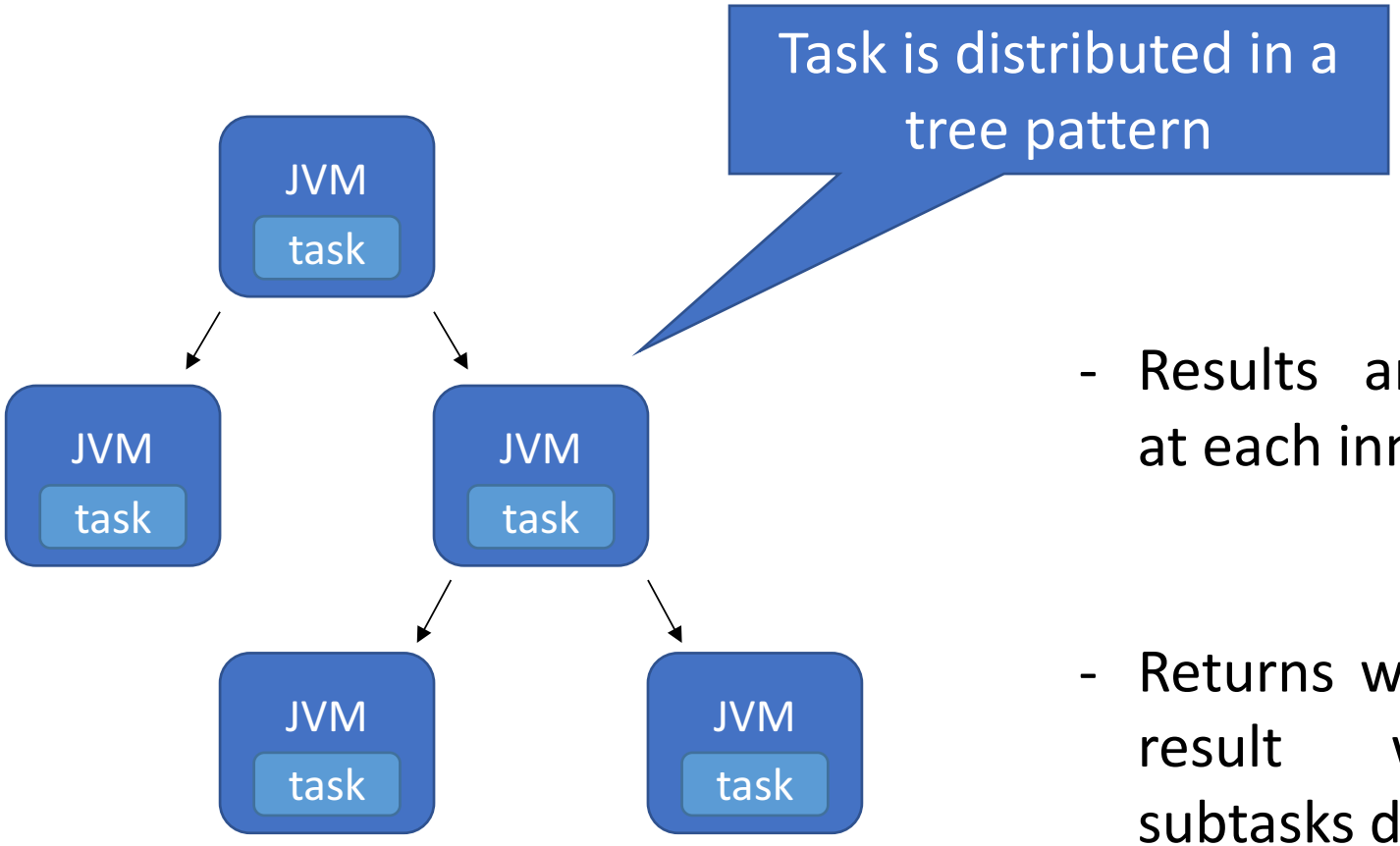


Diagram of distributed arrays. An “H2O Frame” is a collection of distributed arrays.

Distributed Fork/Join



- Results are reduced at each inner node
- Returns with a single result when all subtasks done

Data Processing in H2O

Map Reduce

- Map/Reduce is a nice way to write blatantly parallel code, and they support a particularly fast and efficient flavor.
- Distributed fork/join and parallel map: within each node, classic fork / join

Ease of Use

- H2O has overloaded all the basic data frame manipulation functions in R and Python.
- Tasks such as imputation and one-hot encoding of categoricals is performed inside the algorithms.

H2O in R

“h2o” R package on CRAN

Requirements

- The only requirement to run the “h2o” R package is R >=3.1.0 and Java 7 or later.
- Tested on many versions of Linux, OS X and Windows.

Installation

- The easiest way to install the “h2o” R package is to install directly from CRAN.
- Latest version: http://h2o.ai/download

Design

- No computation is ever performed in R.
- All computations are performed (in highly optimized Java code) in the H2O cluster and initiated by REST calls from R.

Start H2O Cluster from R

```
> library(h2o)
> localH2O <- h2o.init(nthreads = -1, max_mem_size = "8G")

H2O is not running yet, starting it now...

Note:  In case of errors look at the following log files:
    /var/folders/2j/jg4sl53d5q53tc2_nzm9fz5h0000gn/T//RtmpAXY9gj/h2o_me_started_from_r.out
    /var/folders/2j/jg4sl53d5q53tc2_nzm9fz5h0000gn/T//RtmpAXY9gj/h2o_me_started_from_r.err

java version "1.8.0_45"
Java(TM) SE Runtime Environment (build 1.8.0_45-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.45-b02, mixed mode)

.Successfully connected to http://127.0.0.1:54321/

R is connected to the H2O cluster:
    H2O cluster uptime:         1 seconds 96 milliseconds
    H2O cluster version:        3.3.0.99999
    H2O cluster name:           H2O_started_from_R_me_kfo618
    H2O cluster total nodes:    1
    H2O cluster total memory:   7.11 GB
    H2O cluster total cores:    8
    H2O cluster allowed cores:  8
    H2O cluster healthy:        TRUE

>
```

H2O in R: Load Data

Example

```
library(h2o)  # First install from CRAN
localH2O <- h2o.init()  # Initialize the H2O cluster

# Data directly into H2O cluster (avoids R)
train <- h2o.importFile(path = "train.csv")

# Data into H2O from R data.frame
train <- as.h2o(my_df)
```

R code example: Load data

H2O in R: Train & Test

Example

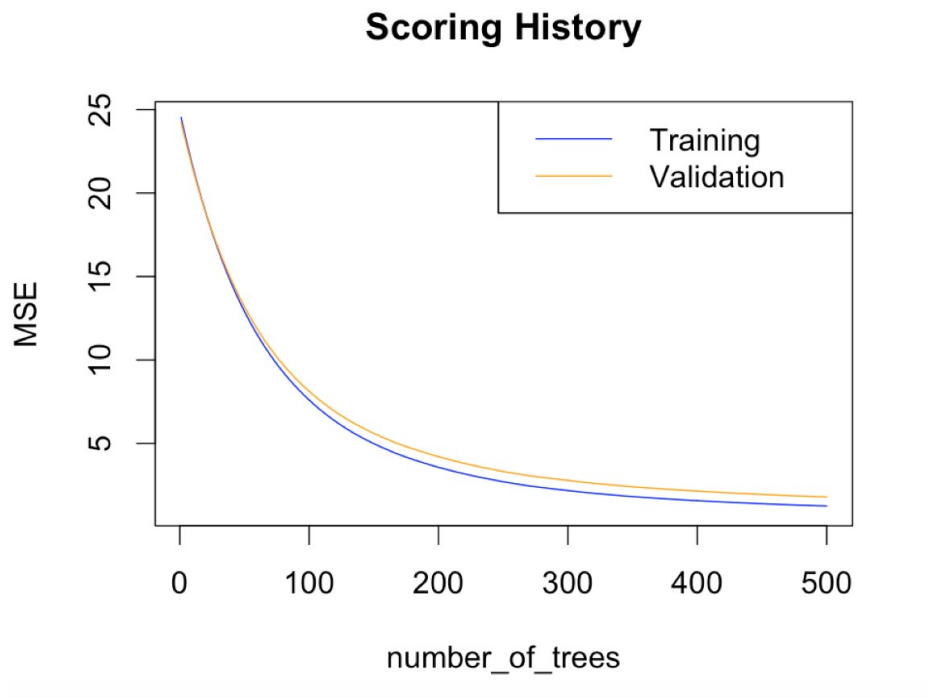
```
y <- "Class"
x <- setdiff(names(train), y)

fit <- h2o.gbm(x = x, y = y, training_frame = train)

pred <- h2o.predict(fit, test)
```

R code example: Train and Test a GBM

H2O in R: Plotting



`plot(fit)` plots scoring history over time.

Live H2O Demo!

FLOW!

Web interface

H2O Ensemble

What is Ensemble Learning?

What it is:

- “Ensemble methods use multiple learning algorithms to obtain better predictive performance that could be obtained from any of the constituent learning algorithms.” (Wikipedia)
- Random Forests and Gradient Boosting Machines (GBM) are both ensembles of decision trees.
- Stacking, or Super Learning, is technique for combining various learners into a single, powerful learner using a second-level metalearning algorithm.

What it’s not:

- Ensembles typically achieve superior model performance over singular methods. However, this comes at a price — computation time.

H2O Ensemble Overview

ML Tasks

- Regression
- Binary Classification / Ranking

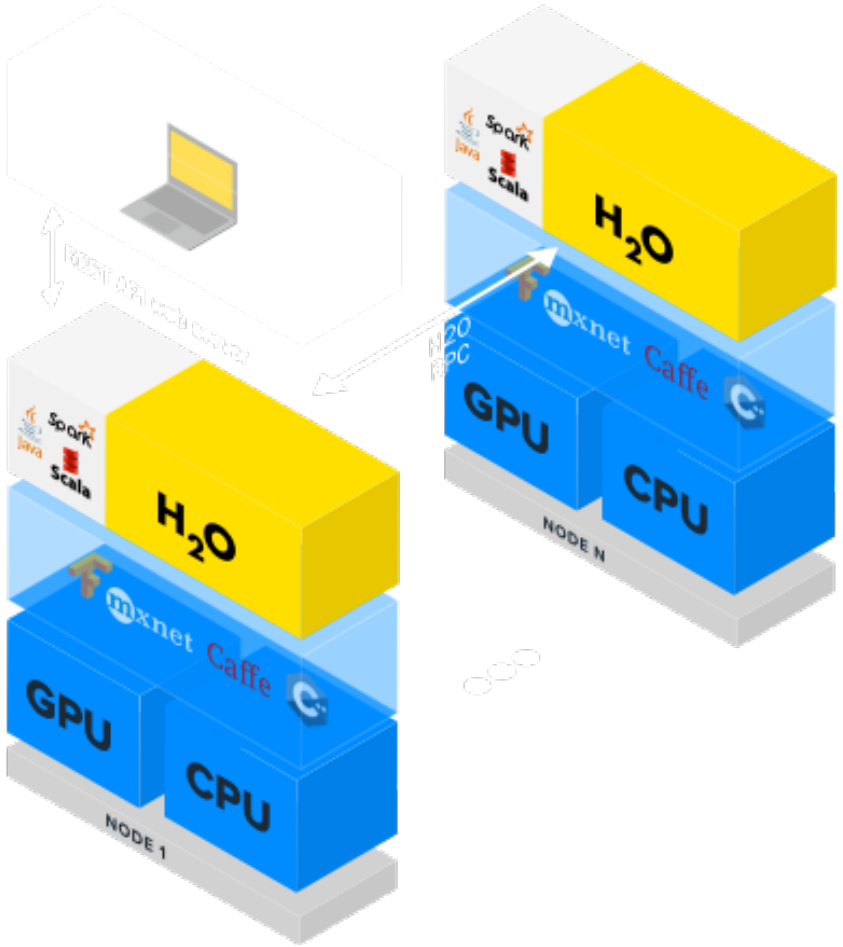
Super Learner

- H2O Ensemble implements the Super Learner algorithm.
- The Super Learner algorithm finds the optimal (based on defined loss) combination of a collection of base learning algorithms.

Why ensembles?

- When a single algorithm does not approximate the true prediction function well.
- When model performance is the most important factor (over training speed and interpretability).

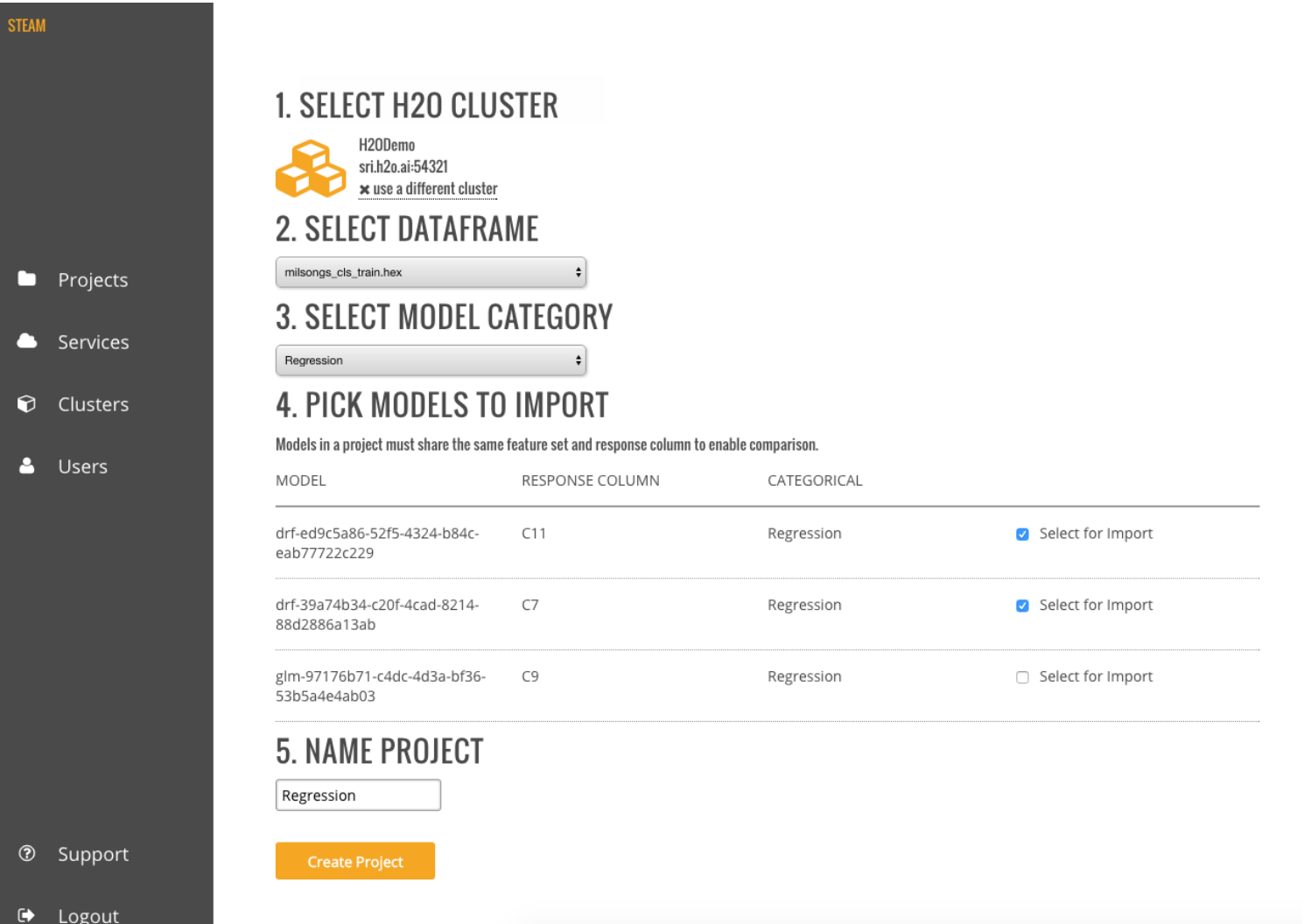
Deep Water



- Native implementation of Deep Learning models for GPU-optimized backends (MXNet, Caffe, TensorFlow, etc.)
- World Record Performance for AI
- H2O.ai is accelerating both machine learning and deep learning on GPUs, providing enterprises opportunities to build better models and enable new use cases.

Steam

DevOps for Data Scientists!



AutoML

- H2O has made it easy for **non-experts** to experiment with machine learning, but there is still a fair bit of knowledge and background in data science that is required to produce high-performing machine learning models.

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

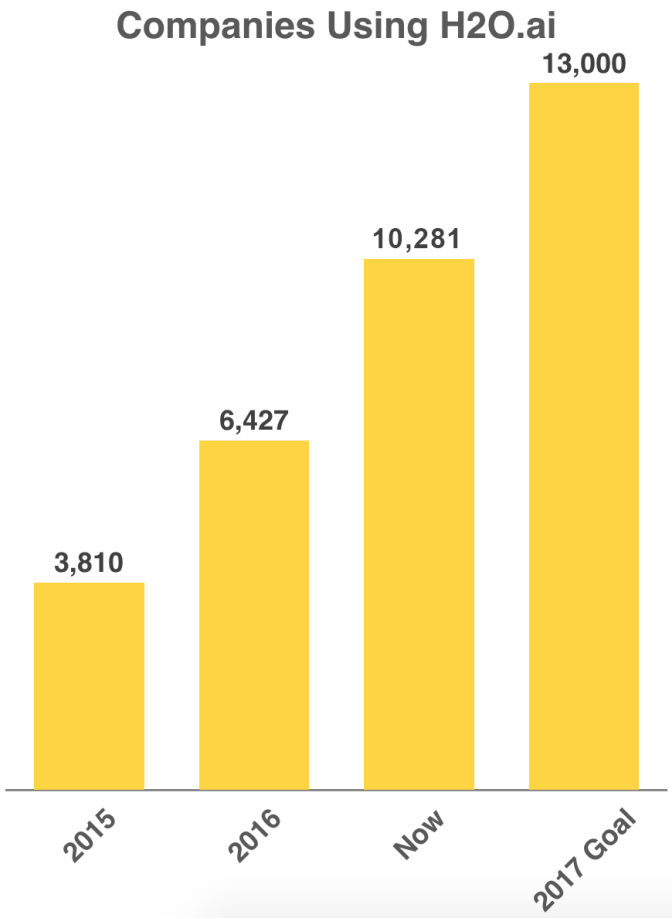
Driverless AI

- Driverless AI seeks to build the fastest artificial intelligence (AI) platform on graphical processing units (GPUs).

- “AI to do AI”

- Software

Customers

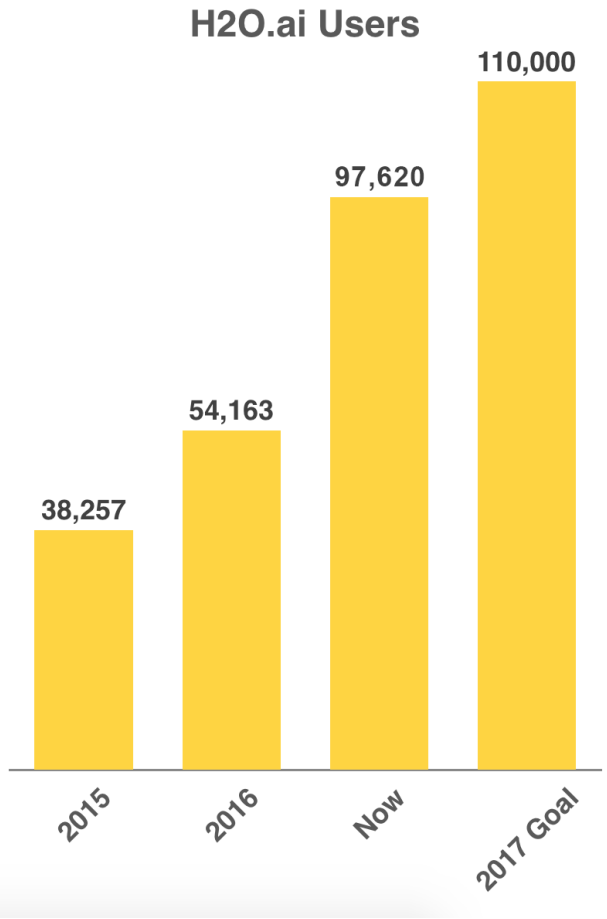


169 OF THE FORTUNE 500

8 OF TOP 10 BANKS

7 OF TOP 10 INSURANCE COMPANIES

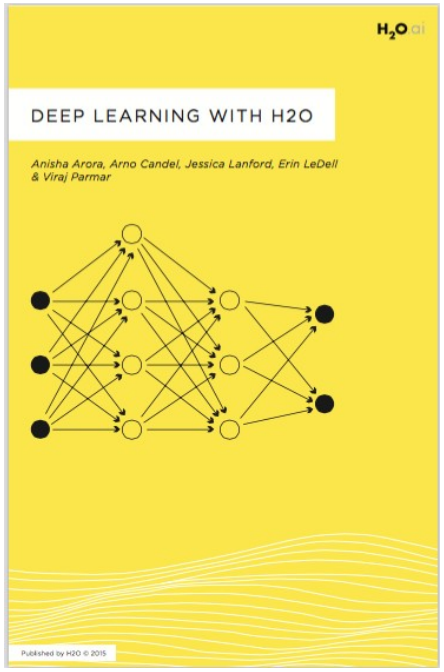
4 OF TOP 10 HEALTHCARE COMPANIES

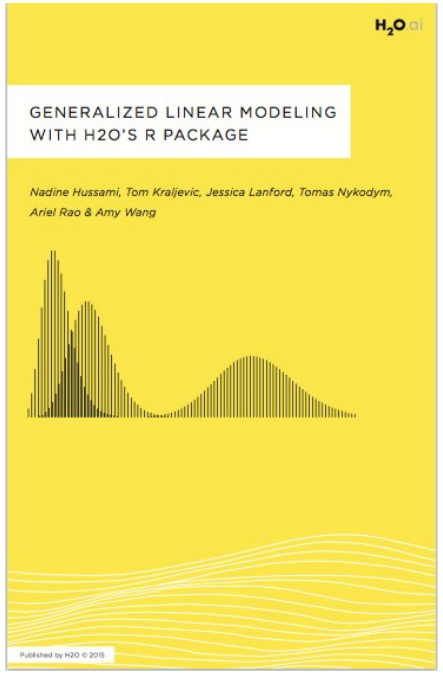


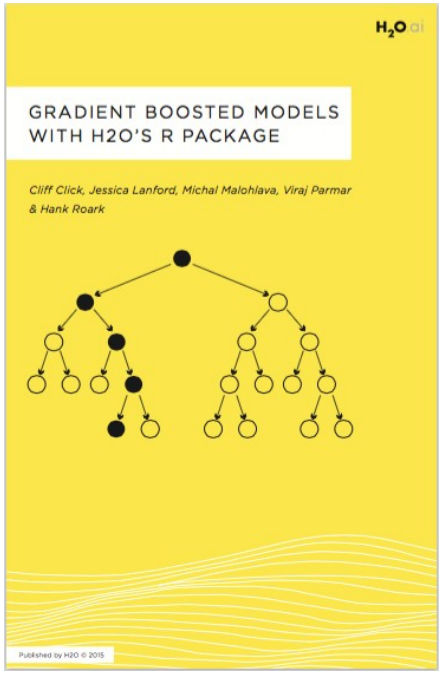
Where to learn more?

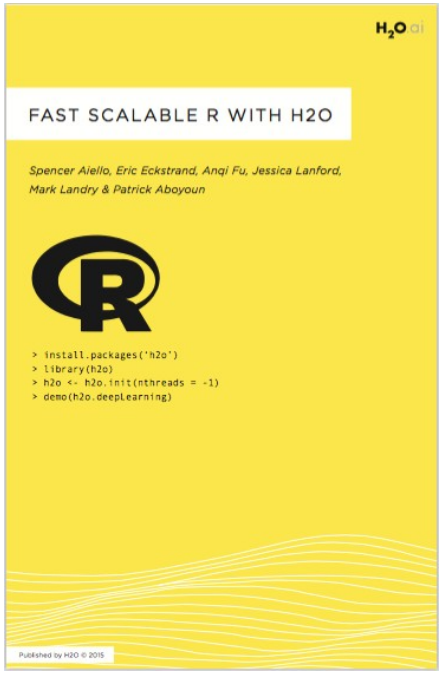
- H2O Online Training (free): http://learn.h2o.ai
- H2O Slidedecks: http://www.slideshare.net/0xdata
- H2O Video Presentations: https://www.youtube.com/user/0xdata
- H2O Community Events & Meetups: http://h2o.ai/events
- Machine Learning & Data Science courses: http://coursebuffet.com

H2O Booklets









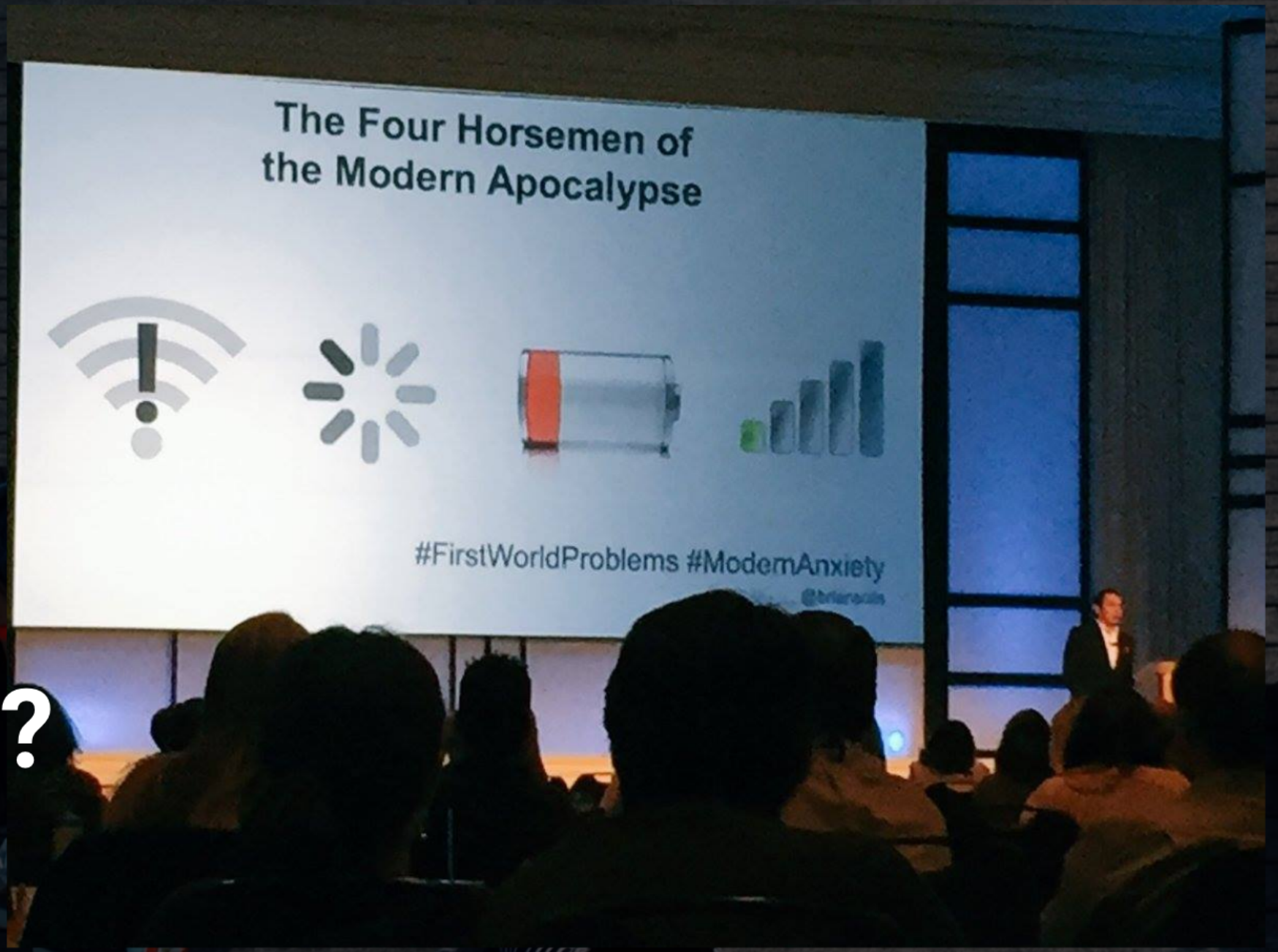
http://docs.h2o.ai

Thanks!

@serafimpinto on
LinkedIn, GitHub

spinto@performetric.net

#ModernSociety



Questions?

spinto@performetric.net