Scalable Automatic Machine Learning with H2O

Santa Clara University
May 2018

Erin LeDell Ph.D.
@ledell

What is H2O?


H2O.ai, the company

- Founded in 2012
- Advised by Stanford Professors Hastie, Tibshirani & Boyd
- Headquarters: Mountain View, California, USA

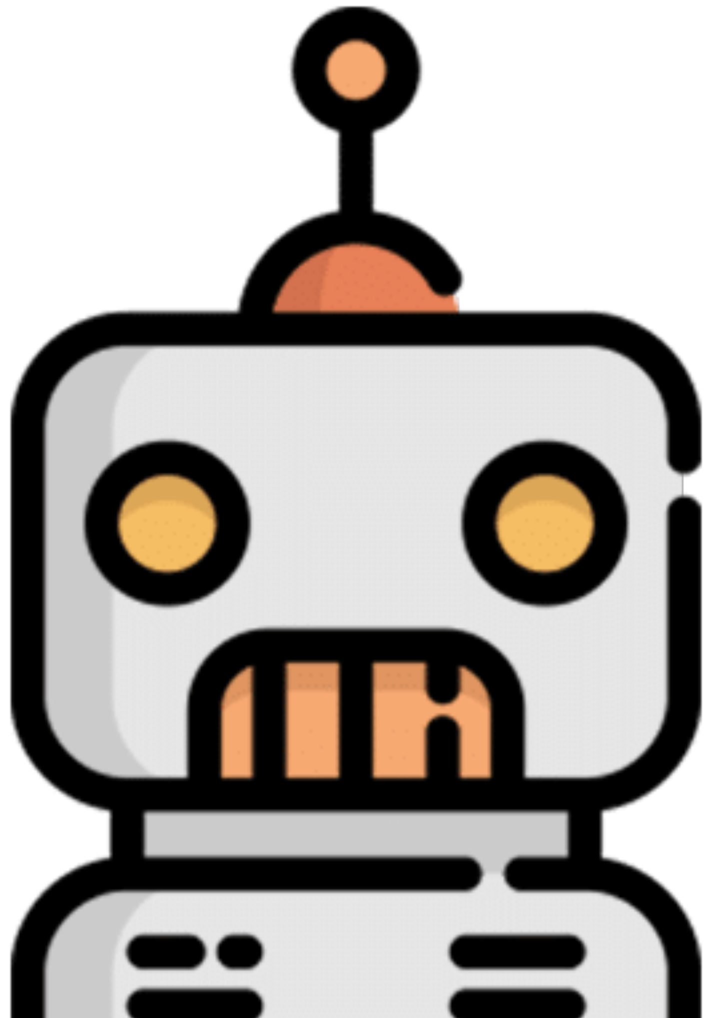
H2O, the platform

- Open Source Software (Apache 2.0 Licensed)
- R, Python, Scala, Java and Web Interfaces
- Distributed Machine Learning Algorithms for Big Data

Agenda

- H2O Platform
- Intro to Automatic Machine Learning (AutoML)
- H2O AutoML Overview
- Pro Tips
- Demo

Slides ⬇️ https://tinyurl.com/scu-automl



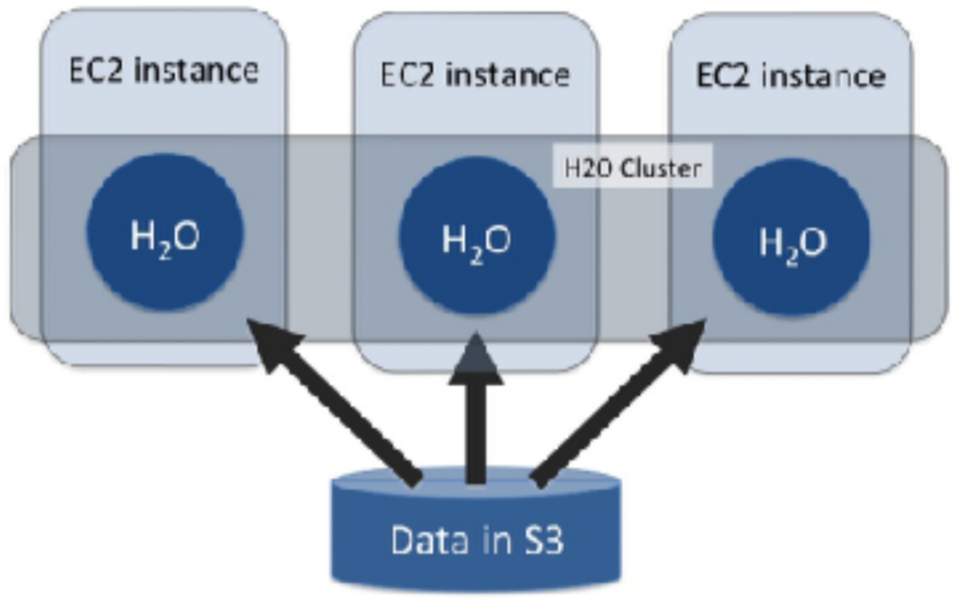
H2O Platform

H2O Machine Learning Platform

- Distributed (multi-core + multi-node) implementations of cutting edge ML algorithms.
- Core algorithms written in high performance Java.
- APIs available in R, Python, Scala; web GUI.
- Easily deploy models to production as pure Java code.
- Works on Hadoop, Spark, EC2, your laptop, etc.

H2O Distributed Computing

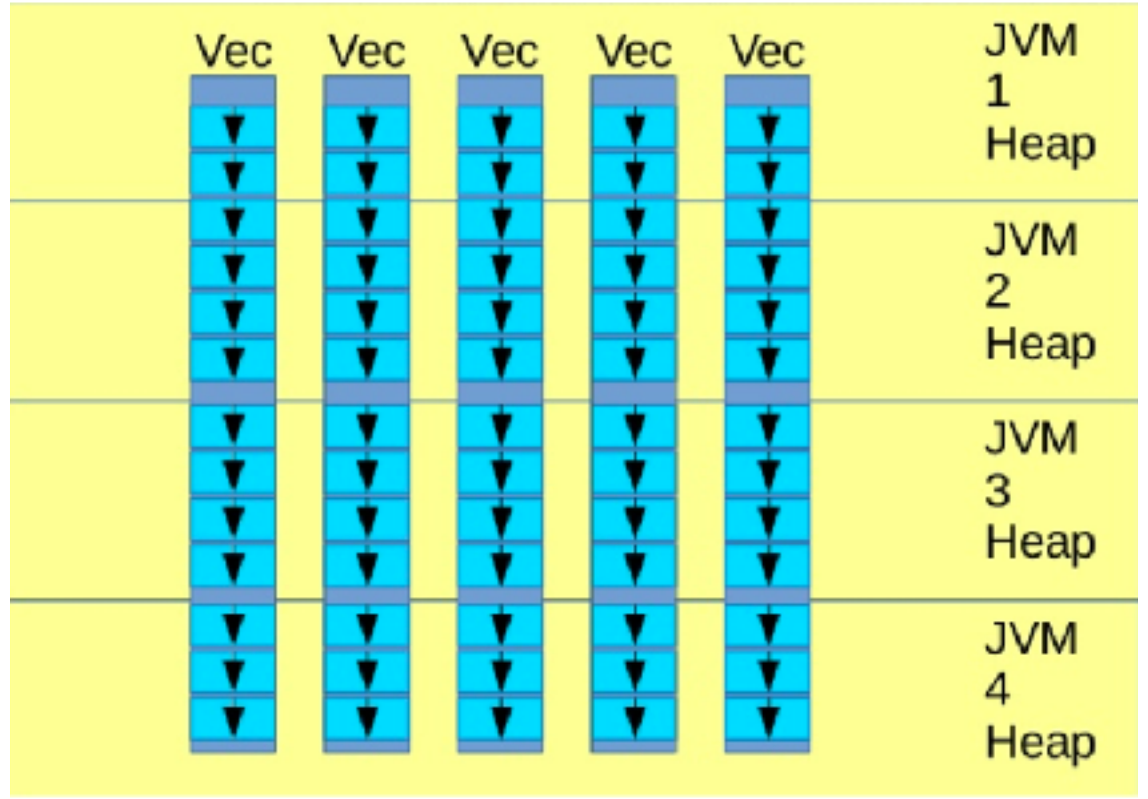
H2O Cluster



- 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.

H2O Frame

- Distributed data frames (collection of vectors).
- Columns are distributed (across nodes) arrays.
- Works just like R's data.frame or Python Pandas DataFrame

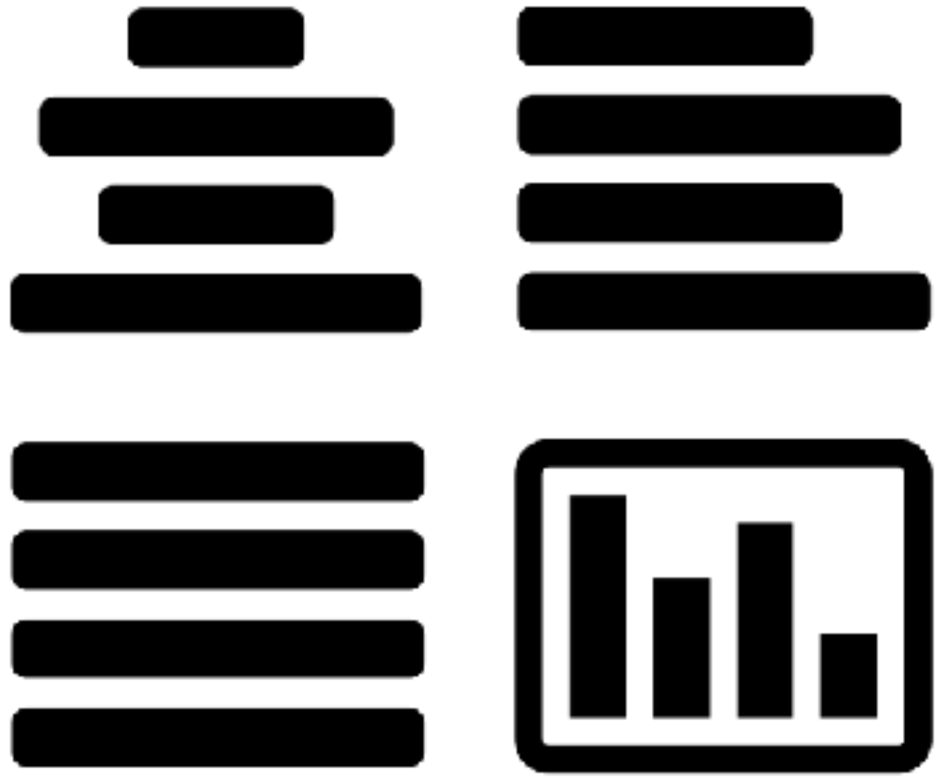


H2O Machine Learning Features

- Supervised & unsupervised machine learning algos (GBM, RF, DNN, GLM, Stacked Ensembles, etc.)
- Imputation, normalization & auto one-hot-encoding
- Automatic early stopping
- Cross-validation, grid search & random search
- Variable importance, model evaluation metrics, plots

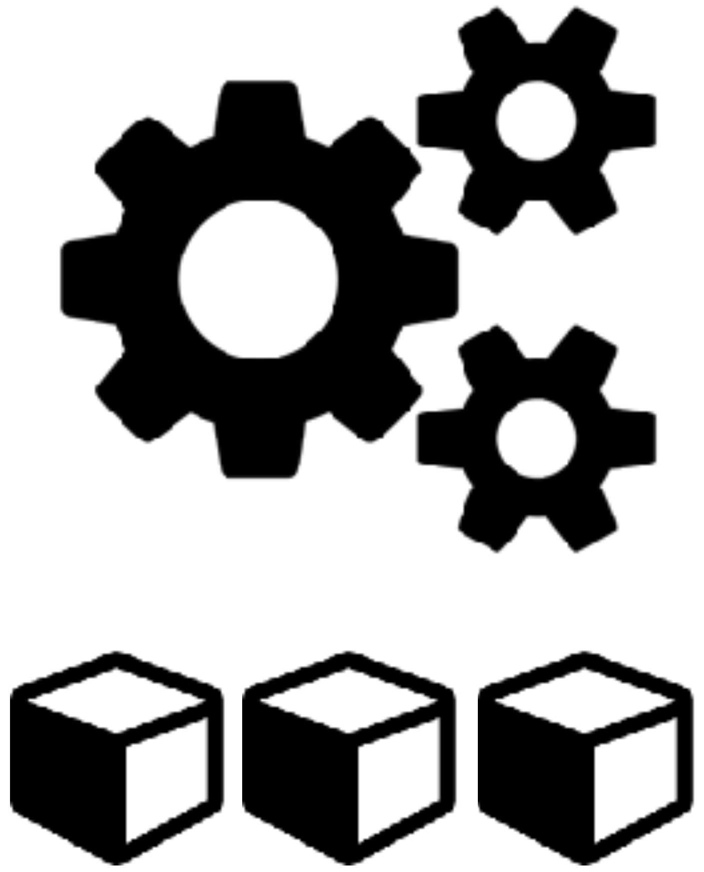
Intro to Automatic Machine Learning

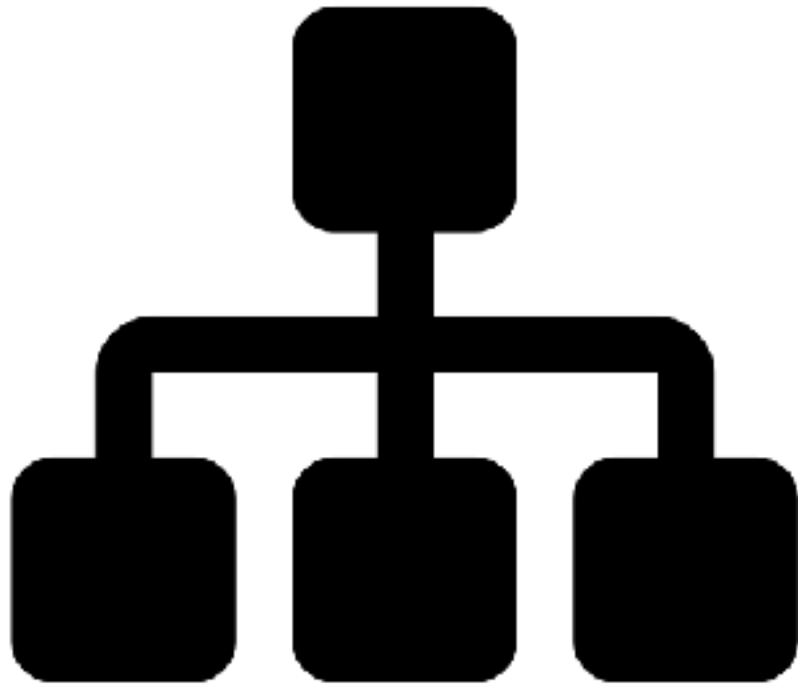
Aspects of Automatic Machine Learning



Data Prep

Model Generation





Ensembles

Aspects of Automatic Machine Learning

Data Preprocessing	• Imputation, one-hot encoding, standardization • Feature selection and/or feature extraction (e.g. PCA) • Count/Label/Target encoding of categorical features
Model Generation	• Cartesian grid search or random grid search • Bayesian Hyperparameter Optimization • Individual models can be tuned using a validation set
Ensembles	• Ensembles often out-perform individual models • Stacking / Super Learning (Wolpert, Breiman) • Ensemble Selection (Caruana)

H2O’s AutoML

H2O Machine Learning Platform

- Open source, distributed (multi-core + multi-node) implementations of cutting edge ML algorithms.
- Core algorithms written in high performance Java.
- APIs available in R, Python, Scala; web GUI.
- Easily deploy models to production as pure Java code.
- Works on Hadoop, Spark, AWS, your laptop, etc.

H2O AutoML (current release)

Data Preprocessing

- Imputation, one-hot encoding, standardization
- Feature selection and/or feature extraction (e.g. PCA)
- Count/Label/Target encoding of categorical features

Model Generation

- Cartesian grid search or random grid search
- Bayesian Hyperparameter Optimization
- Individual models can be tuned using a validation set

Ensembles

- Ensembles often out-perform individual models:
- Stacking / Super Learning (Wolpert, Breiman)
- Ensemble Selection (Caruana)

Random Grid Search & Stacking

- Random Grid Search combined with Stacked Ensembles is a powerful combination.
- Ensembles perform particularly well if the models they are based on (1) are individually strong, and (2) make uncorrelated errors.
- Stacking uses a second-level metalearning algorithm to find the optimal combination of base learners.

Stacking (aka Super Learner Algorithm)

$$n\left\{\overbrace{\begin{bmatrix} \quad & & \\ & X & \\ & & \quad \end{bmatrix}}^{m}\right. \quad \begin{bmatrix} \\ y \\ \\ \end{bmatrix}$$

"Level-zero" data

- Start with design matrix, X, and response, y
- Specify L base learners (with model params)
- Specify a metalearner (just another algorithm)
- Perform k-fold CV on each of the L learners

Stacking (aka Super Learner Algorithm)

$$
n\left\{ \begin{bmatrix} \\ p_1 \\ \\ \end{bmatrix} \cdots \begin{bmatrix} \\ p_L \\ \\ \end{bmatrix} \begin{bmatrix} \\ y \\ \\ \end{bmatrix} \right. \rightarrow n\left\{ \overbrace{\begin{bmatrix} & & \\ & Z & \\ & & \end{bmatrix}}^{L} \begin{bmatrix} \\ y \\ \\ \end{bmatrix} \right.
$$

"Level-one" data

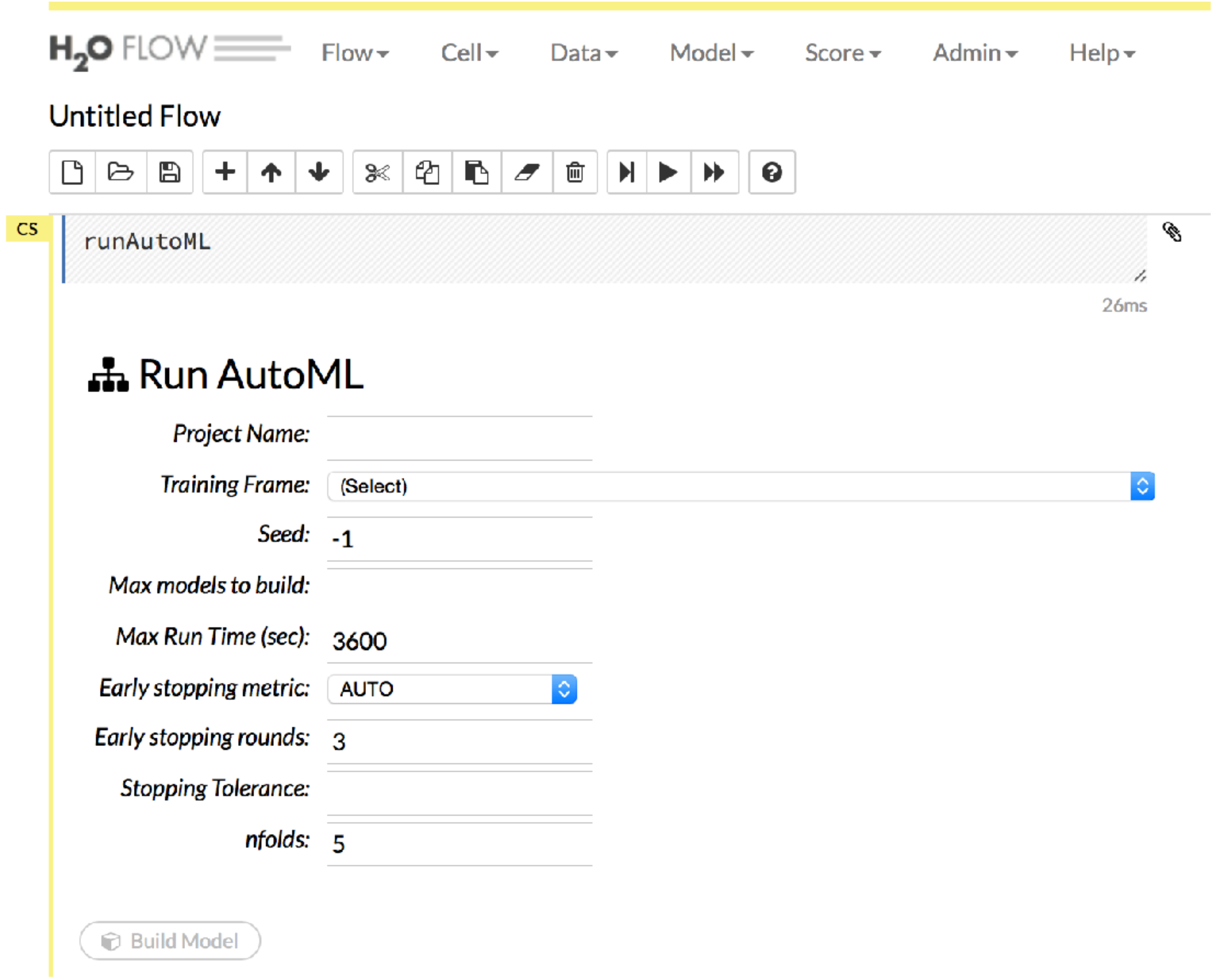
- Collect the predicted values from k-fold CV that was performed on each of the L base learners
- Column-bind these prediction vectors together to form a new design matrix, Z
- Train the metalearner using Z, y

H2O AutoML

- Basic data pre-processing (as in all H2O algos).
- Trains a random grid of GBMs, DNNs, GLMs, etc. using a carefully chosen hyper-parameter space
- Individual models are tuned using a validation set.
- Two Stacked Ensembles are trained ("All Models" ensemble & a lightweight "Best of Family" ensemble).
- Returns a sorted "Leaderboard" of all models.

Available in H2O >=3.14

H2O AutoML in Flow GUI



H2O AutoML in R

Example

```r
library(h2o)
h2o.init()

train <- h2o.importFile("train.csv")


aml <- h2o.automl(y = "response_colname",
                  training_frame = train,
                  max_runtime_secs = 600)

lb <- aml@leaderboard
```

H2O AutoML in Python

Example

```python
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
```

H2O AutoML Leaderboard

model_id	auc	logloss
StackedEnsemble_AllModels_0_AutoML_20171121_012135	0.788321	0.554019
StackedEnsemble_BestOfFamily_0_AutoML_20171121_012135	0.783099	0.559286
GBM_grid_0_AutoML_20171121_012135_model_1	0.780554	0.560248
GBM_grid_0_AutoML_20171121_012135_model_0	0.779713	0.562142
GBM_grid_0_AutoML_20171121_012135_model_2	0.776206	0.564970
GBM_grid_0_AutoML_20171121_012135_model_3	0.771026	0.570270
DRF_0_AutoML_20171121_012135	0.734653	0.601520
XRT_0_AutoML_20171121_012135	0.730457	0.611706
GBM_grid_0_AutoML_20171121_012135_model_4	0.727098	0.666513
GLM_grid_0_AutoML_20171121_012135_model_0	0.685211	0.635138

Example Leaderboard for binary classification

AutoML Pro Tips!

Before you press the "red button"

AutoML Pro Tips: Input Frames

- Don't use `leaderboard_frame` unless you really need to; use cross-validation metrics to generate the leaderboard instead (default).
- If you only provide `training_frame`, it will chop off 20% of your data for a validation set to be used in early stopping. To control this proportion, you can split the data yourself and pass a `validation_frame` manually.

AutoML Pro Tips: Exclude Algos

- If you have sparse, wide data (e.g. text), use the `exclude_algos` argument to turn off the tree-based models (GBM, RF).
- If you want tree-based algos only, turn off GLM and DNNs via `exclude_algos`.

AutoML Pro Tips: Time & Model Limits

- AutoML will stop after 1 hour unless you change `max_runtime_secs`.
- Running with `max_runtime_secs` is not reproducible since available resources on a machine may change from run to run. Set `max_runtime_secs` to a big number (e.g. 999999999) and use `max_models` instead.

AutoML Pro Tips: Cluster memory

- Reminder: All H2O models are stored in H2O Cluster memory.
- Make sure to give the H2O Cluster a lot of memory if you're going to create hundreds or thousands of models.
- e.g. `h2o.init(max_mem_size = "80G")`

After you press the "red button"

AutoML Pro Tips: Early Stopping

- If you're expecting more models than are listed in the leaderboard, or the run is stopping earlier than `max_runtime_secs`, this is a result of the default "early stopping" settings.
- To allow more time, increase the number of `stopping_rounds` and/or decrease value of `stopping_tolerance`.

AutoML Pro Tips: Add More Models

- If you want to add (train) more models to an existing AutoML project, just make sure to use the same training set and `project_name`.
- If you set the same seed twice it will give you identical models as the first run (not useful), so change the seed or leave it unset.

AutoML Pro Tips: Saving Models

- You can save any of the individual models created by the AutoML run. The model ids are listed in the leaderboard.

- If you're taking your leader model (probably a Stacked Ensemble) to production, we'd recommend using "Best of Family" since it only contains 5 models and gets most of the performance of the "All Models" ensemble.

H2O AutoML Tutorial

H2O AutoML Tutorial

https://tinyurl.com/automl-h2oworld17

Code available here

H2O Resources

- Documentation: http://docs.h2o.ai
- Tutorials: https://github.com/h2oai/h2o-tutorials
- Slidedecks: https://github.com/h2oai/h2o-meetups
- Videos: https://www.youtube.com/user/0xdata
- Stack Overflow: https://stackoverflow.com/tags/h2o
- Google Group: https://tinyurl.com/h2ostream
- Gitter: http://gitter.im/h2oai/h2o-3
- Events & Meetups: http://h2o.ai/events

Contribute to H2O!

open source

Get in touch over email, Gitter or JIRA.

https://github.com/h2oai/h2o-3/blob/master/CONTRIBUTING.md

Thank you!

@ledell on Github, Twitter

erin@h2o.ai

http://www.stat.berkeley.edu/~ledell