Scalable Automatic Machine Learning in H2O

AI & DL Meetup

Nov 2017

Erin LeDell Ph.D.
H2O.ai

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

Scientific Advisory Council

Dr. Trevor Hastie

- John A. Overdeck Professor of Mathematics, Stanford University
- PhD in Statistics, Stanford University
- Co-author, *The Elements of Statistical Learning: Prediction, Inference and Data Mining*
- Co-author with John Chambers, *Statistical Models in S*
- Co-author, *Generalized Additive Models*

Dr. Robert Tibshirani

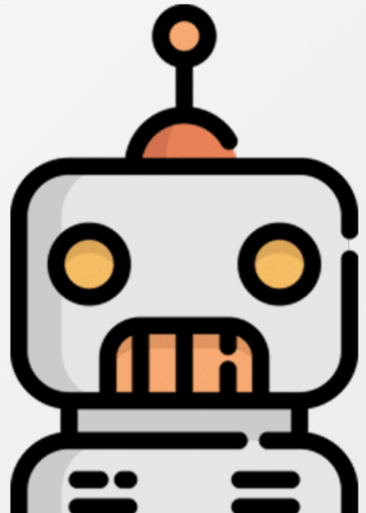
- Professor of Statistics and Health Research and Policy, Stanford University
- PhD in Statistics, Stanford University
- Co-author, *The Elements of Statistical Learning: Prediction, Inference and Data Mining*
- Author, *Regression Shrinkage and Selection via the Lasso*
- Co-author, *An Introduction to the Bootstrap*

Dr. Steven Boyd

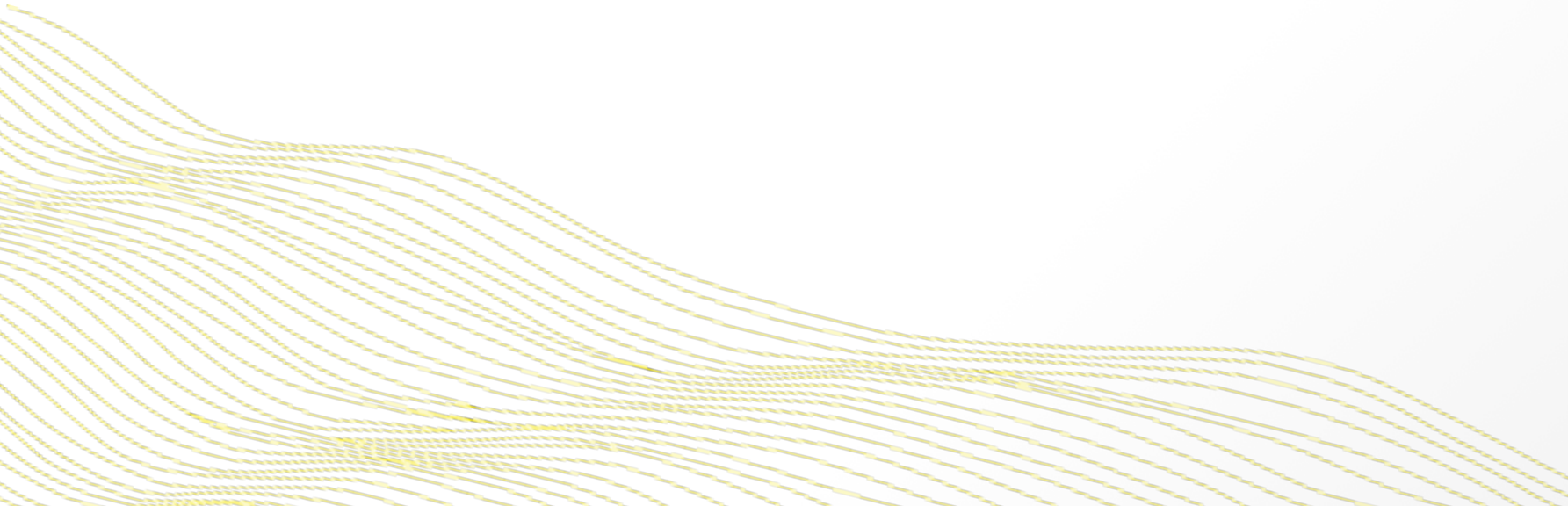
- Professor of Electrical Engineering and Computer Science, Stanford University
- PhD in Electrical Engineering and Computer Science, UC Berkeley
- Co-author, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*
- Co-author, *Linear Matrix Inequalities in System and Control Theory*
- Co-author, *Convex Optimization*

Agenda

- Intro to Automatic Machine Learning (AutoML)
- Bayesian Hyperparameter Optimization
- Random Grid Search & Stacked Ensembles
- H2O Machine Learning Platform Overview
- H2O's AutoML (R, Python, GUI)



AutoML Overview



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

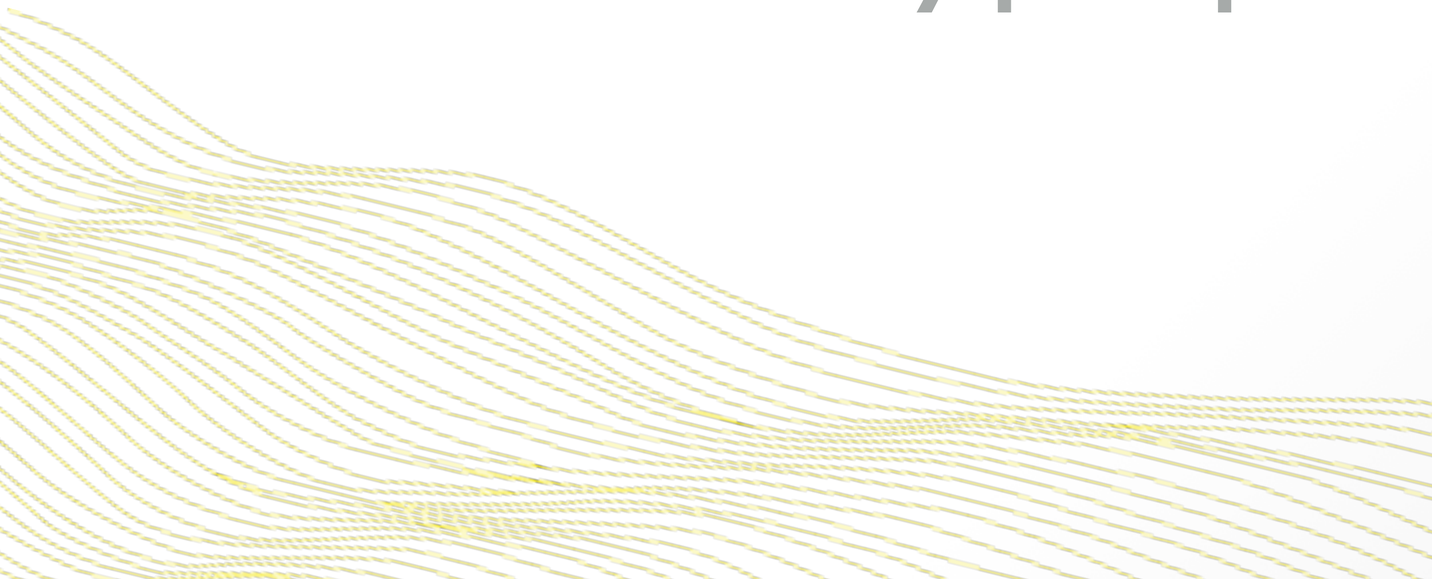
Model Generation

- 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)

Bayesian Optimization of Hyperparameters



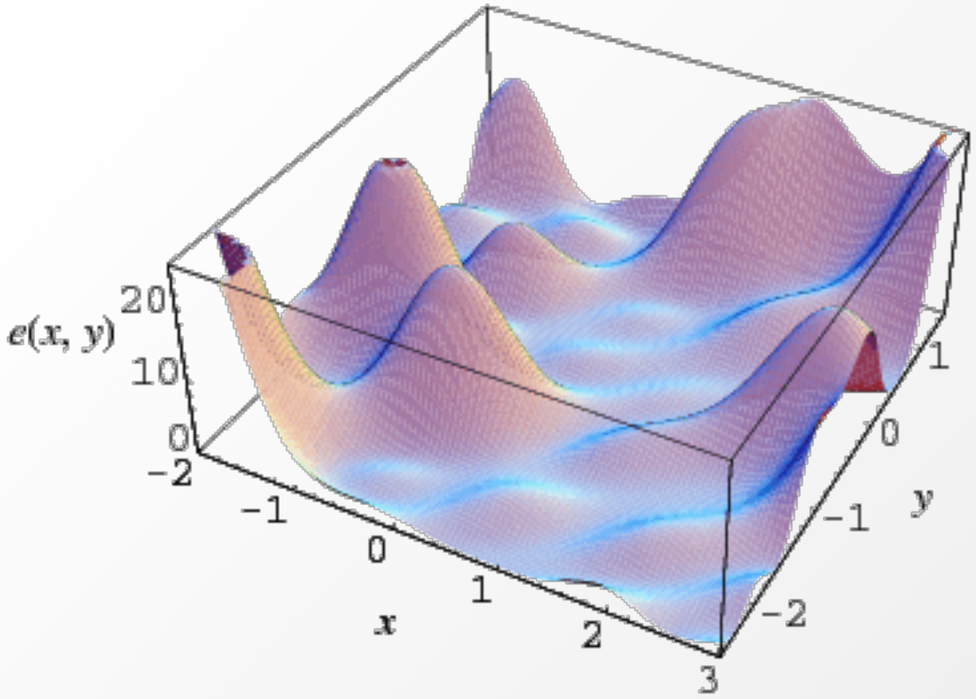
Bayesian Optimization

- Bayesian Hyperparameter Optimization consists of developing a statistical model of the function mapping hyperparameter values to the objective (e.g. AUC, MSE), evaluated on a validation set.
- Different approaches based on: Gaussian Processes, Tree Structured Parzen Estimator, Random Forest

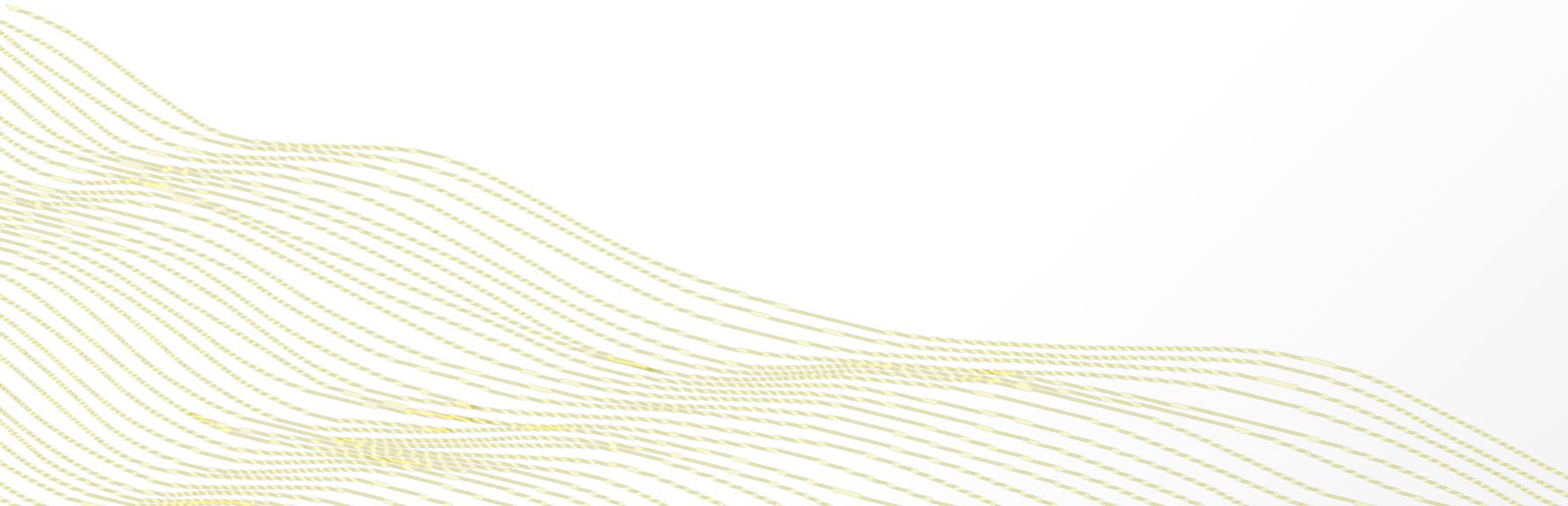
AKA “Sequential Model-based Optimization (SMBO)”

Hyperparameter Optimization Software

- mlrHyperopt, mlrMBO (R)
- Scikit-Optimize (Python)
- Hyperopt (Python)
- Spearmint (Python)
- Auto-WEKA, SMAC (Java)
- SigOpt (SaaS)
- etc.



Random Grids + Stacked Ensembles



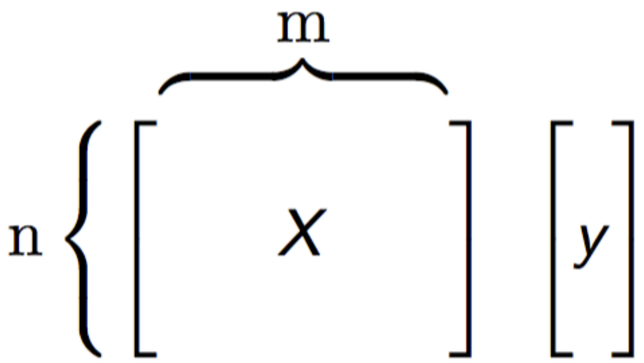
Random Grid Search & Stacking

- 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.

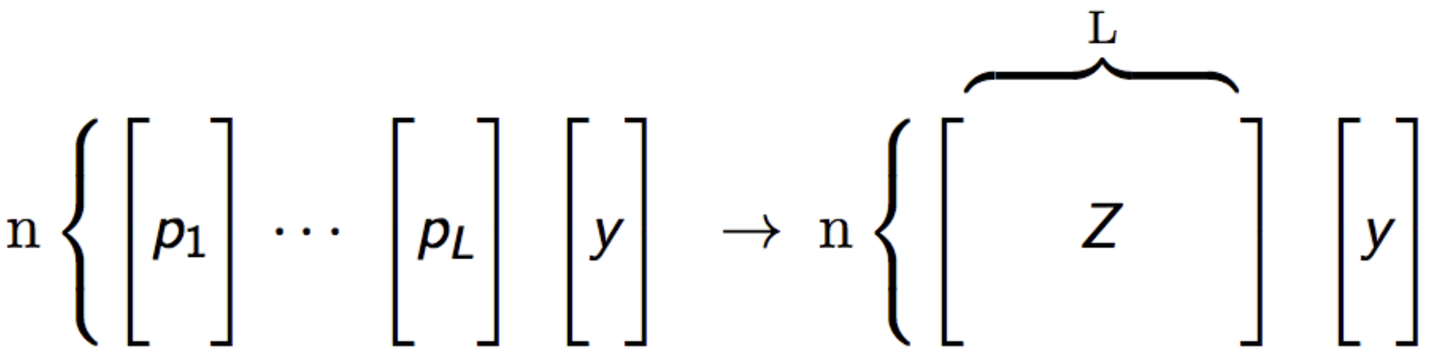
Stacking (aka Super Learner Algorithm)



"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)



"Level-one" data

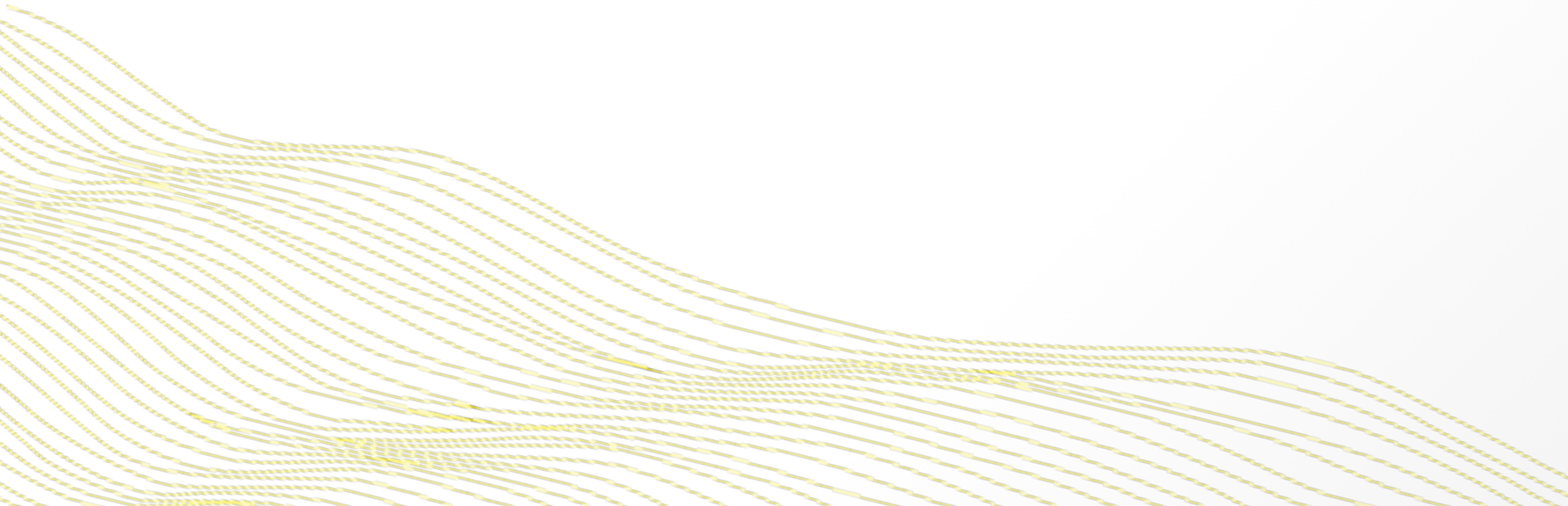
- Collect the predicted values from k-fold CV that was performed on each of the L base learners
- Column-bind these prediction vectors together to form a new design matrix, Z
- Train the metalearner using Z, y

Stacking vs Ensemble Selection

- Stacking uses all the given models (good and bad) and uses a second-level metalearning algorithm to find the optimal combination of base learners.

- With Ensemble Selection, rather than combine good and bad models in an ensemble, forward stepwise selection is used to find a subset of models that, when averaged together, yield the best performance.

H2O Platform

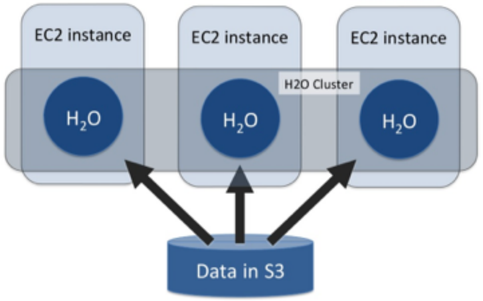


H2O Machine Learning Platform

- 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

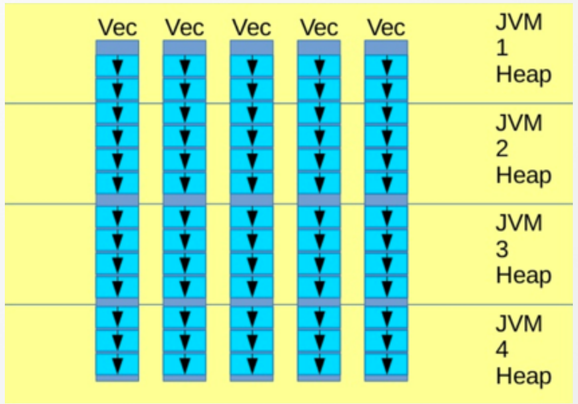
H2O Cluster



- 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

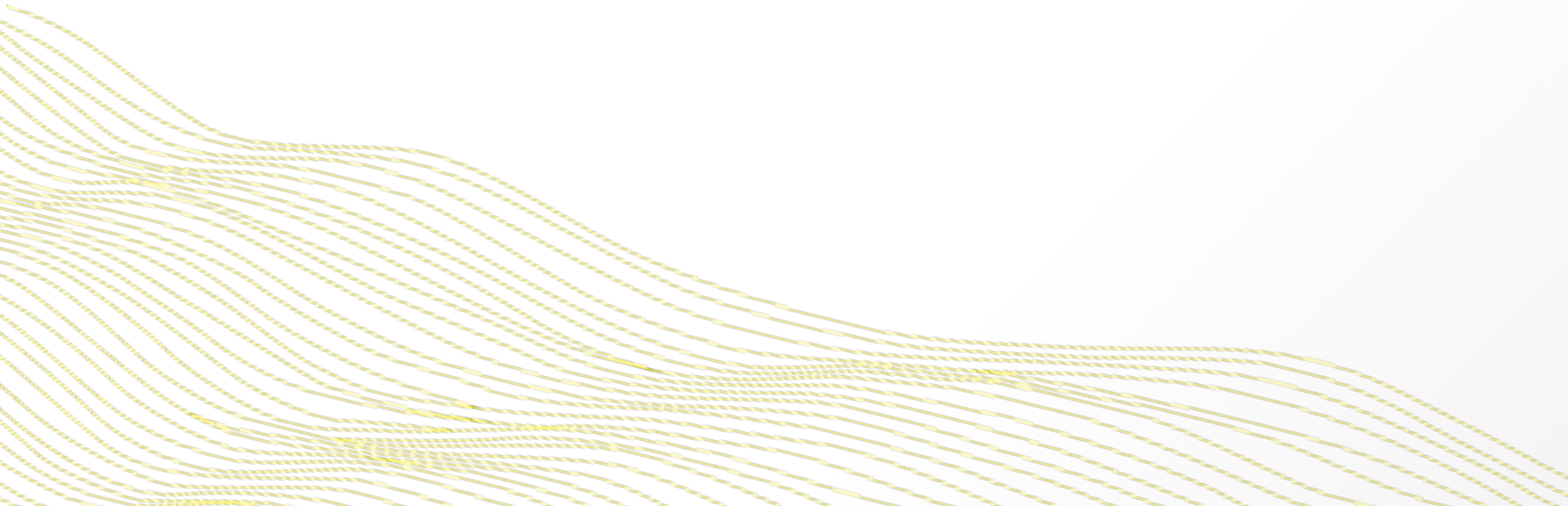
- Distributed data frames (collection of vectors).
- Columns are distributed (across nodes) arrays.
- Works just like R's data.frame or Python Pandas DataFrame



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

H2O AutoML



H2O AutoML (first 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)

H2O AutoML

- Basic data pre-processing (as in all H2O algos).
- Trains a random grid of GBMs, DNNs, GLMs, etc. using a carefully chosen parameter space; individual models are tuned using a validation set.
- A Stacked Ensemble is trained using all models.
- Returns a sorted "Leaderboard" of all models.

Available in H2O >=3.14

H2O AutoML in 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
```

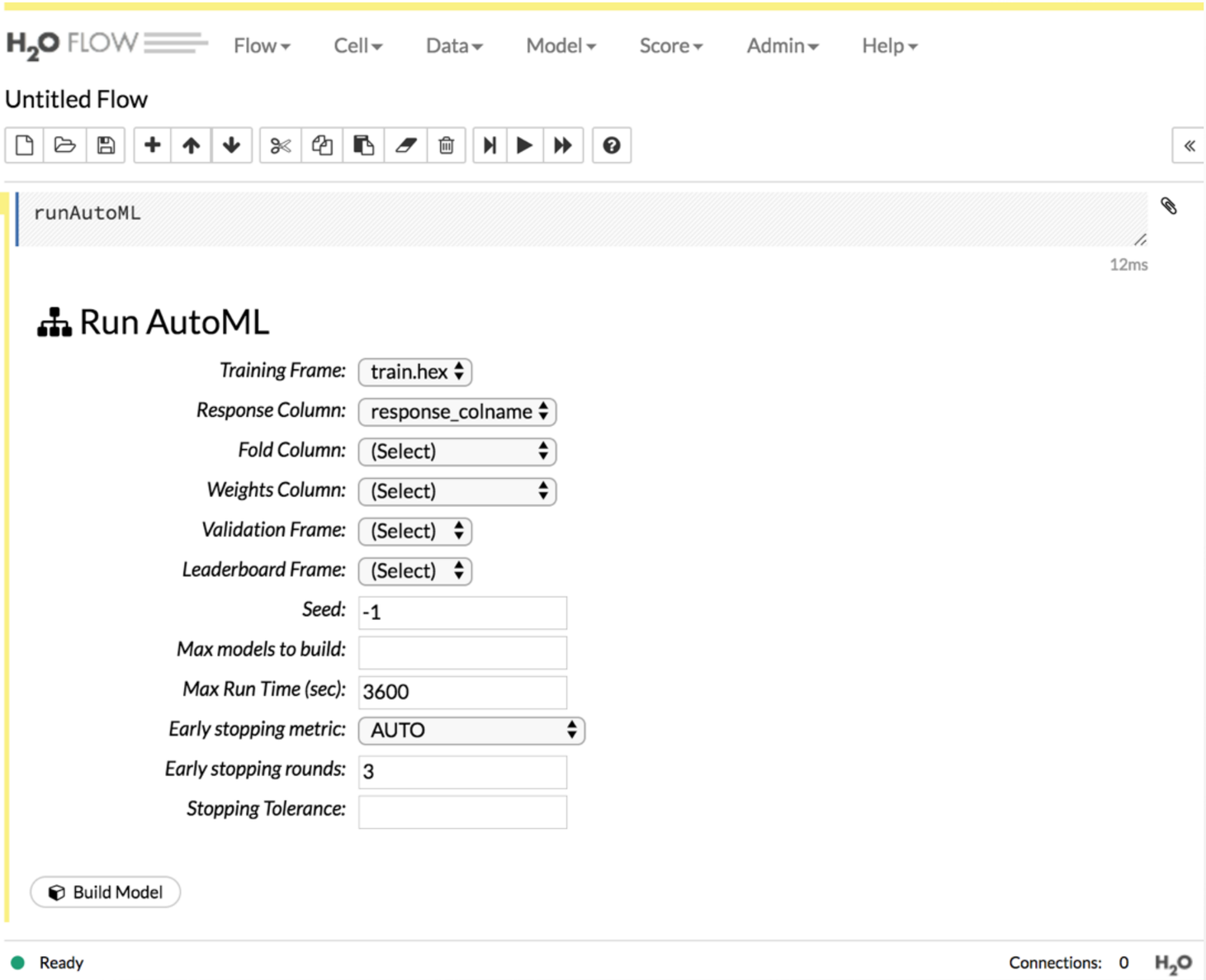
H2O AutoML in Python

```
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 in Flow GUI



H2O AutoML 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

Example Leaderboard for binary classification

H2O Resources

- Documentation: http://docs.h2o.ai
- Tutorials: https://github.com/h2oai/h2o-tutorials
- Slidedecks: https://github.com/h2oai/h2o-meetups
- Video Presentations: https://www.youtube.com/user/0xdata
- Events & Meetups: http://h2o.ai/events

Thank you!

@ledell on Github, Twitter

erin@h2o.ai

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