

Gradient Boosting Machines (GBM) in R

Szilárd Pafka, PhD

Chief Scientist, Epoch

R-Ladies Meetup Budapest

January 2018



Edit profile

Szilard

@DataScienceLA

physics PhD, chief (data) scientist, meetup organizer, datascience.la, (visiting) professor, machine learning benchmarks 🇺🇸 🇭🇺 🇪🇺

📍 Santa Monica, California 🔗 [linkedin.com/in/szilard](https://www.linkedin.com/in/szilard)

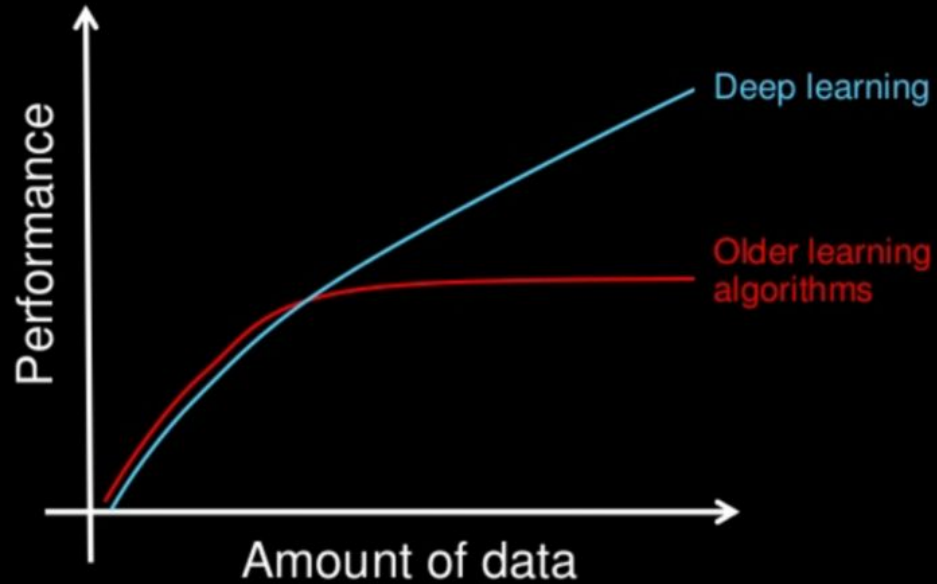
198 Following **2,067** Followers

Disclaimer:

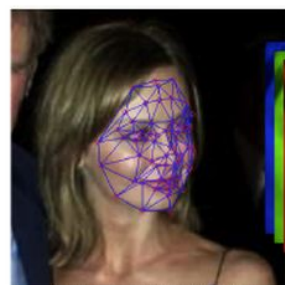
I am not representing my employer (Epoch) in this talk

I cannot confirm nor deny if Epoch is using any of the methods, tools, results etc. mentioned in this talk

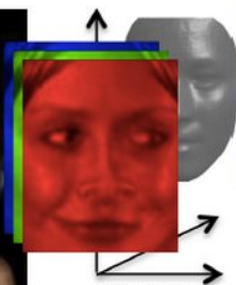
Why deep learning



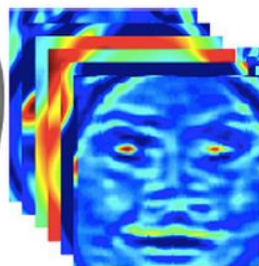
Source: Andrew Ng



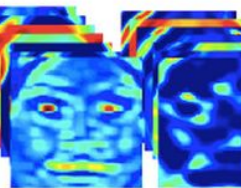
Calista_Flockhart_0002.jpg
Detection & Localization



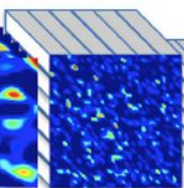
Frontalization:
@152X152x3



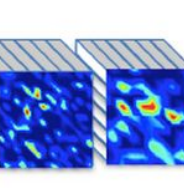
C1:
32x11x11x3
@142x142



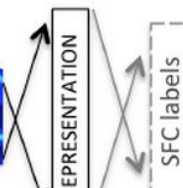
M2:
32x3x3x32
@71x71



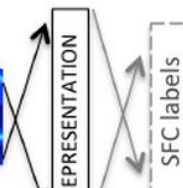
C3:
16x9x9x32
@63x63



L4:
16x9x9x16
@55x55



L5:
16x7x7x16
@25x25



L6:
16x5x5x16
@21x21



F7:
4096d



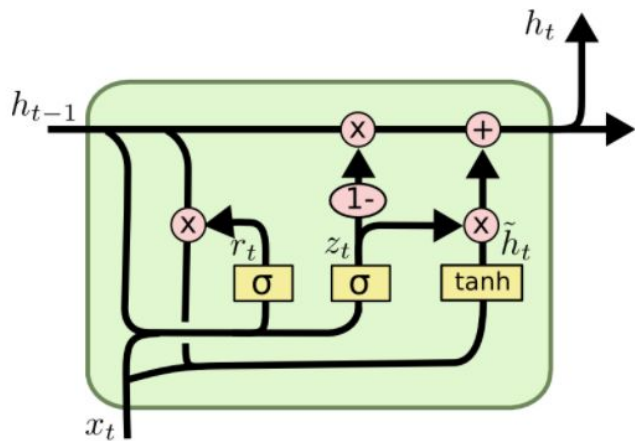
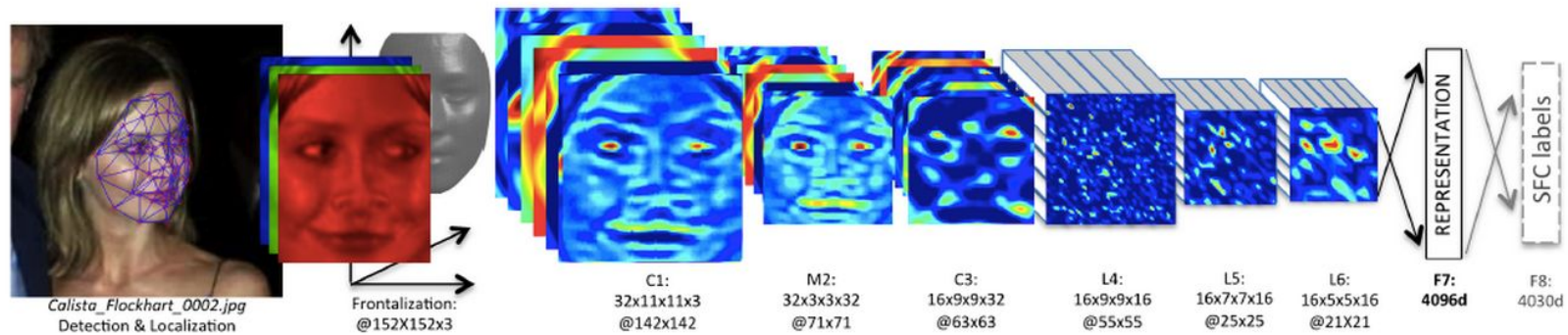
F8:
4030d

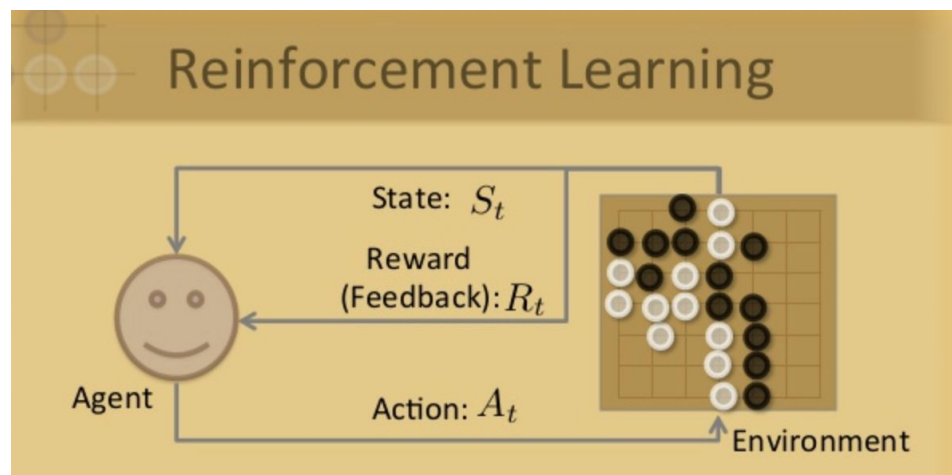
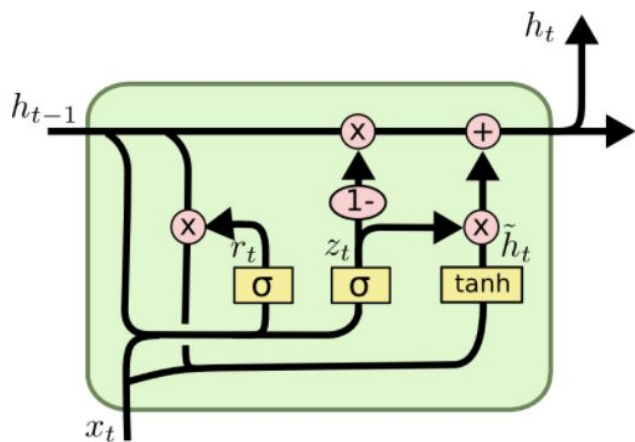
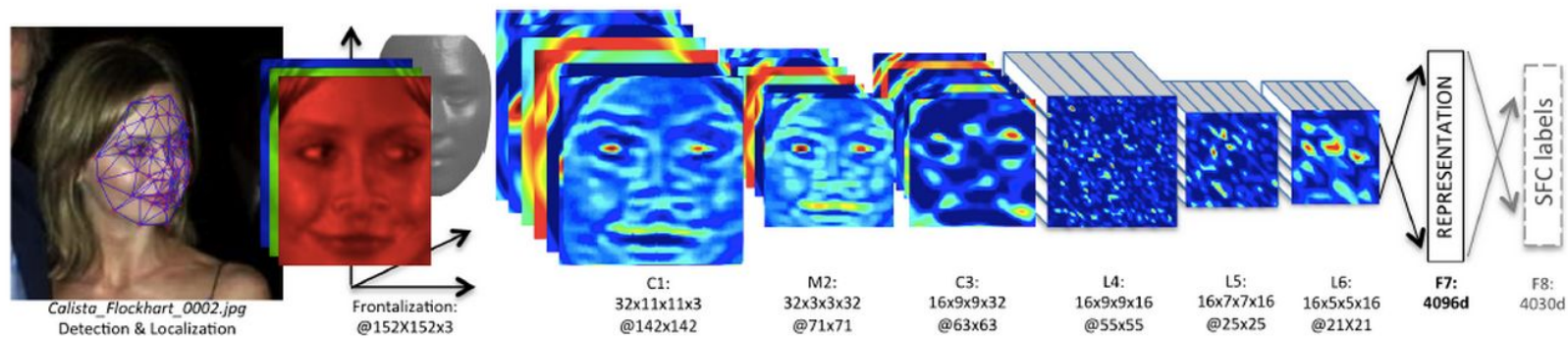


F7:
4096d



F8:
4030d







Params	AUC	Time (s)	Epochs
default: activation = "Rectifier", hidden = c(200,200)	73.1	270	1.8
hidden = c(50,50,50,50), input_dropout_ratio = 0.2	73.2	140	2.7
hidden = c(50,50,50,50)	73.2	110	1.9
hidden = c(20,20)	73.1	100	4.6
hidden = c(20)	73.1	120	6.7

...

RectifierWithDropout, c(200,200,200,200), dropout=c(0.2,0.1,0.1,0)	73.3	440	2.0
ADADELTA rho = 0.95, epsilon = 1e-06	71.1	240	1.7
rho = 0.999, epsilon = 1e-08	73.3	270	1.9
adaptive = FALSE default: rate = 0.005, decay = 1, momentum = 0	73.0	340	1.1
rate = 0.001, momentum = 0.5 / 1e5 / 0.99	73.2	410	0.7
rate = 0.01, momentum = 0.5 / 1e5 / 0.99	73.3	280	0.9
rate = 0.01, rate_annealing = 1e-05, momentum = 0.5 / 1e5 / 0.99	73.5	360	1
rate = 0.01, rate_annealing = 1e-04, momentum = 0.5 / 1e5 / 0.99	72.7	3700	8.7
rate = 0.01, rate_annealing = 1e-05, momentum = 0.5 / 1e5 / 0.9	73.4	350	0.9

Machine Learning Challenge Winning Solutions

- The most frequently used tool by data science competition winners
 - 17 out of 29 winning solutions in kaggle last year used XGBoost
 - Solve wide range of problems: store sales prediction; high energy physics event classification; web text classification; customer behavior prediction; motion detection; ad click through rate prediction; malware classification; product categorization; hazard risk prediction; massive online course dropout rate prediction
- Present and Future of KDDCup. Ron Bekkerman (KDDCup 2015 chair): "Something dramatic happened in Machine Learning over the past couple of years. It is called XGBoost - a package implementing Gradient Boosted Decision Trees that works wonders in data classification. Apparently, every winning team used XGBoost, mostly in ensembles with other classifiers. Most surprisingly, the winning teams report very minor improvements that ensembles bring over a single well-configured XGBoost."
- A lot contributions from the kaggle community

56:01 / 1:16:29



XGBoost A Scalable Tree Boosting System June 02, 2016



DataScience.LA

Subscribe 2.6K

Add to Share ... More

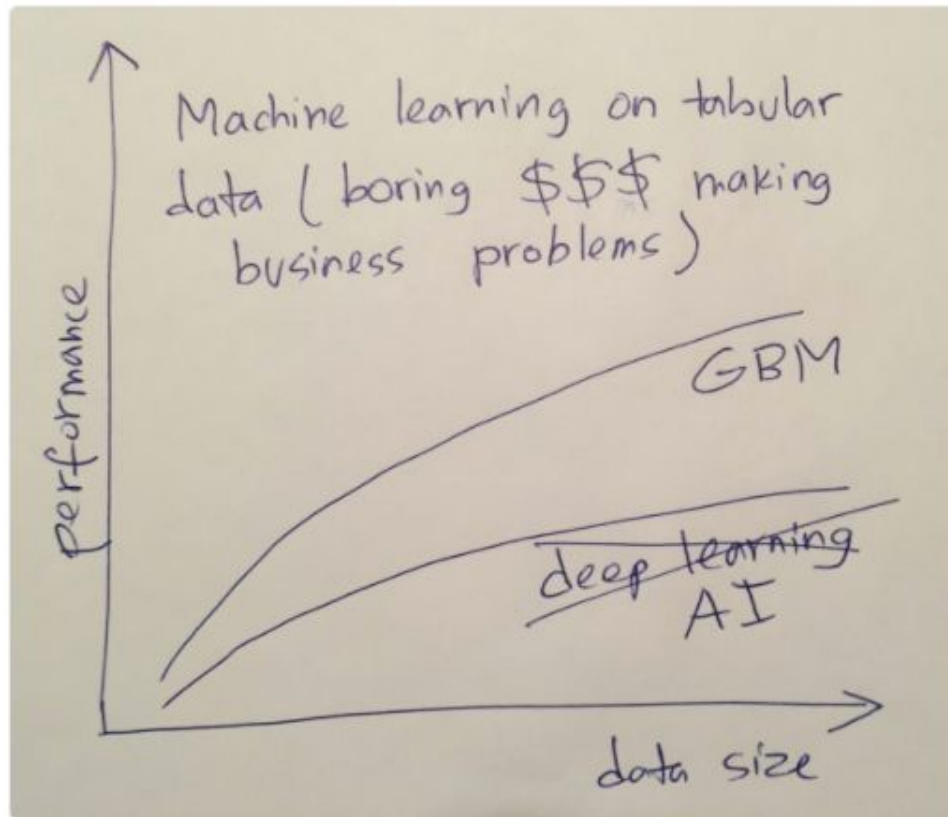
5,632 views

55 0



Szilard @DataScienceLA · 2 Nov 2016

Can anyone beat GBMs with deep learning (ahem, AI) on the airline dataset (or generally tabular/business data)? [github.com/szilard/benchmark...](https://github.com/szilard/benchmark)



2



8



16



MODEL	1ST	2ND
BST-DT	0.580	0.228
RF	0.390	0.525
BAG-DT	0.030	0.232
SVM	0.000	0.008
ANN	0.000	0.007
KNN	0.000	0.000
BST-STMP	0.000	0.000
DT	0.000	0.000
LOGREG	0.000	0.000
NB	0.000	0.000

AVG	1ST	2ND
RF	0.727	0.207
ANN	0.053	0.172
BSTDT	0.059	0.228
SVM	0.043	0.195
LR	0.089	0.132
BAGDT	0.002	0.012
KNN	0.023	0.045
BSTST	0.004	0.009
PRC	0	0
NB	0	0

An Empirical Comparison of Supervised Learning Algorithms

<http://www.cs.cornell.edu/~alexn/papers/empirical.icml06.pdf>

An Empirical Evaluation of Supervised Learning in High Dimensions

<http://lowrank.net/nikos/pubs/empirical.pdf>

MODEL	1ST	2ND
BST-DT	0.580	0.228
RF	0.390	0.525
BAG-DT	0.030	0.232
SVM	0.000	0.008
ANN	0.000	0.007
KNN	0.000	0.000
BST-STMP	0.000	0.000
DT	0.000	0.000
LOGREG	0.000	0.000
NB	0.000	0.000

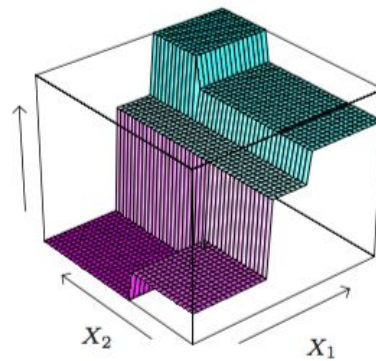
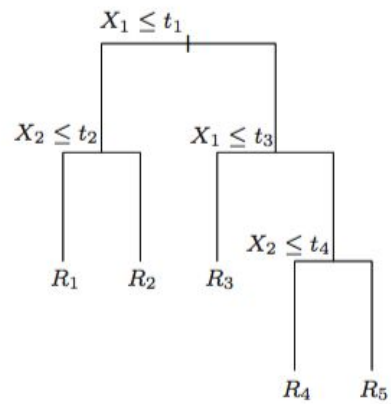
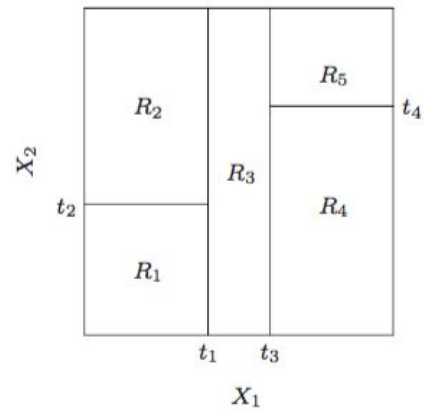
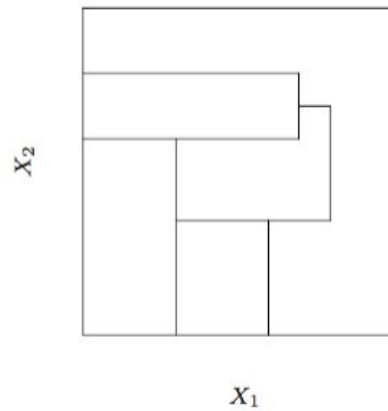
AVG	1ST	2ND
RF	0.727	0.207
ANN	0.053	0.172
BSTDT	0.059	0.228
SVM	0.043	0.195
LR	0.089	0.132
BAGDT	0.002	0.012
KNN	0.023	0.045
BSTST	0.004	0.009
PRC	0	0
NB	0	0

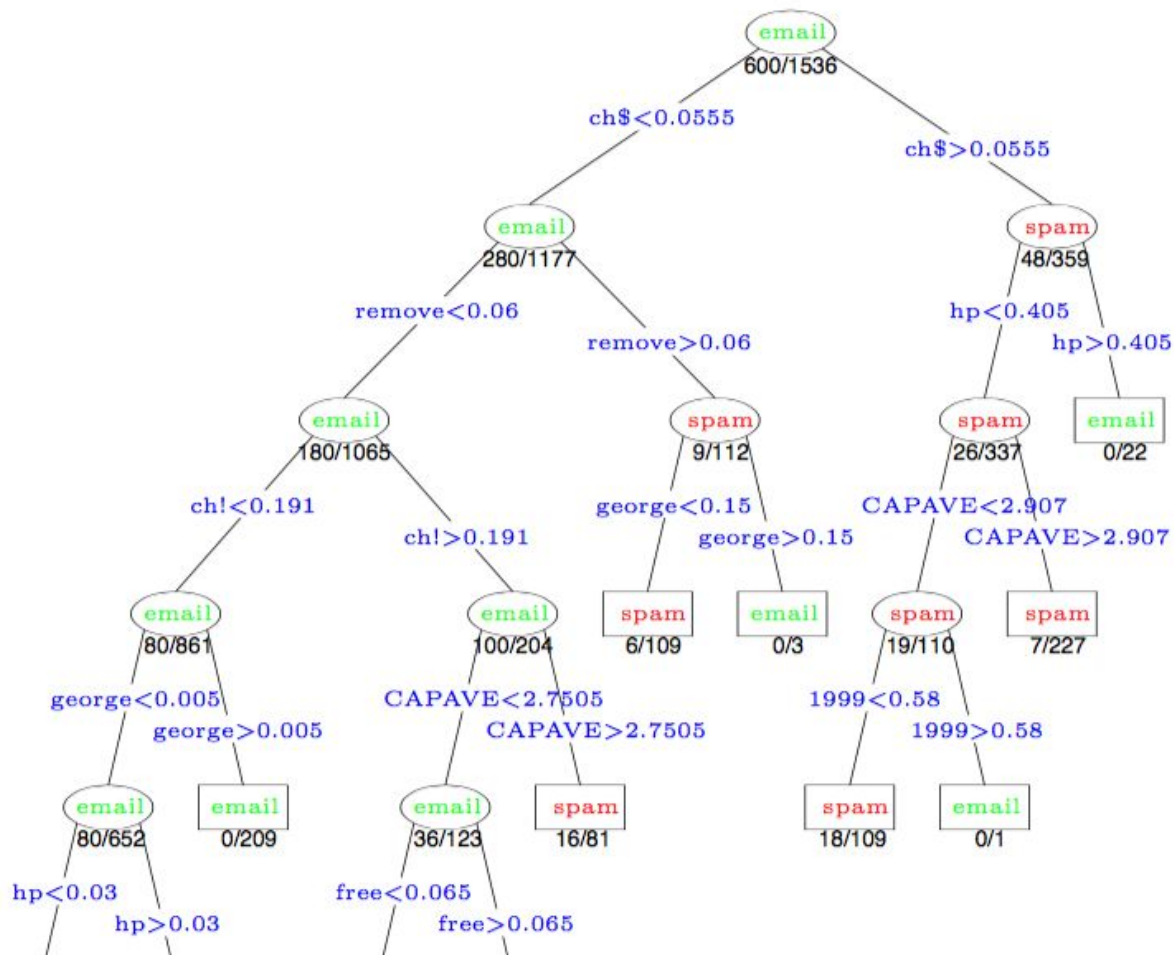
An Empirical Comparison of Supervised Learning Algorithms

<http://www.cs.cornell.edu/~alexn/papers/empirical.icml06.pdf>

An Empirical Evaluation of Supervised Learning in High Dimensions

<http://lowrank.net/nikos/pubs/empirical.pdf>





Algorithm 10.1 *AdaBoost.M1*.

1. Initialize the observation weights $w_i = 1/N$, $i = 1, 2, \dots, N$.
 2. For $m = 1$ to M :
 - (a) Fit a classifier $G_m(x)$ to the training data using weights w_i .
 - (b) Compute
$$\text{err}_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}.$$
 - (c) Compute $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$.
 - (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))]$, $i = 1, 2, \dots, N$.
 3. Output $G(x) = \text{sign} \left[\sum_{m=1}^M \alpha_m G_m(x) \right]$.
-

Algorithm 10.3 *Gradient Tree Boosting Algorithm.*

1. Initialize $f_0(x) = \arg \min_{\gamma} \sum_{i=1}^N L(y_i, \gamma)$.

2. For $m = 1$ to M :

(a) For $i = 1, 2, \dots, N$ compute

$$r_{im} = - \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f=f_{m-1}}.$$

(b) Fit a regression tree to the targets r_{im} giving terminal regions R_{jm} , $j = 1, 2, \dots, J_m$.

(c) For $j = 1, 2, \dots, J_m$ compute

$$\gamma_{jm} = \arg \min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma).$$

(d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

3. Output $\hat{f}(x) = f_M(x)$.

Trevor Hastie
Robert Tibshirani
Jerome Friedman

The Elements of Statistical Learning

Data Mining, Inference, and Prediction

Second Edition



Szilard @DataScienceLA · Jun 17

What's the typical size of datasets you are analyzing?

18% <100MB

48% 100MB-10GB

18% 10GB-1TB

16% >1TB

151 votes • Final results



Szilard @DataScienceLA · Jun 17

What's the typical size of datasets you are analyzing?

18% <100MB

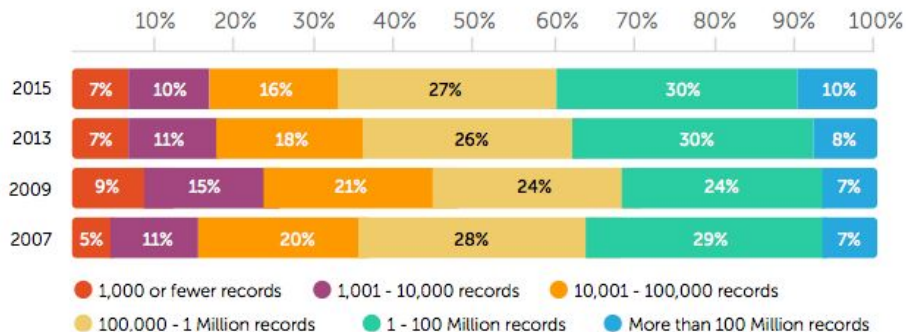
48% 100MB-10GB

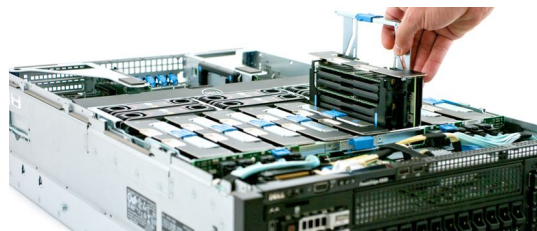
18% 10GB-1TB

16% >1TB

151 votes • Final results

DATASETS





Kingston Technology Value RAM 128GB Kit (4x32GB) 2133MHz DDR4 ECC Reg CL15 (KVR21R15D4K4/128)

by [Kingston Technology](#)

[Be the first to review this item](#)

Was: \$743.99

Price: **\$743.96** & **FREE Shipping**. [Details](#)



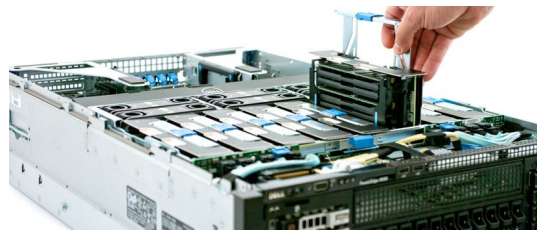
Kingston Technology Value RAM 128GB Kit (4x32GB) 2133MHz DDR4 ECC Reg CL15 (KVR21R15D4K4/128)

by [Kingston Technology](#)

[Be the first to review this item](#)

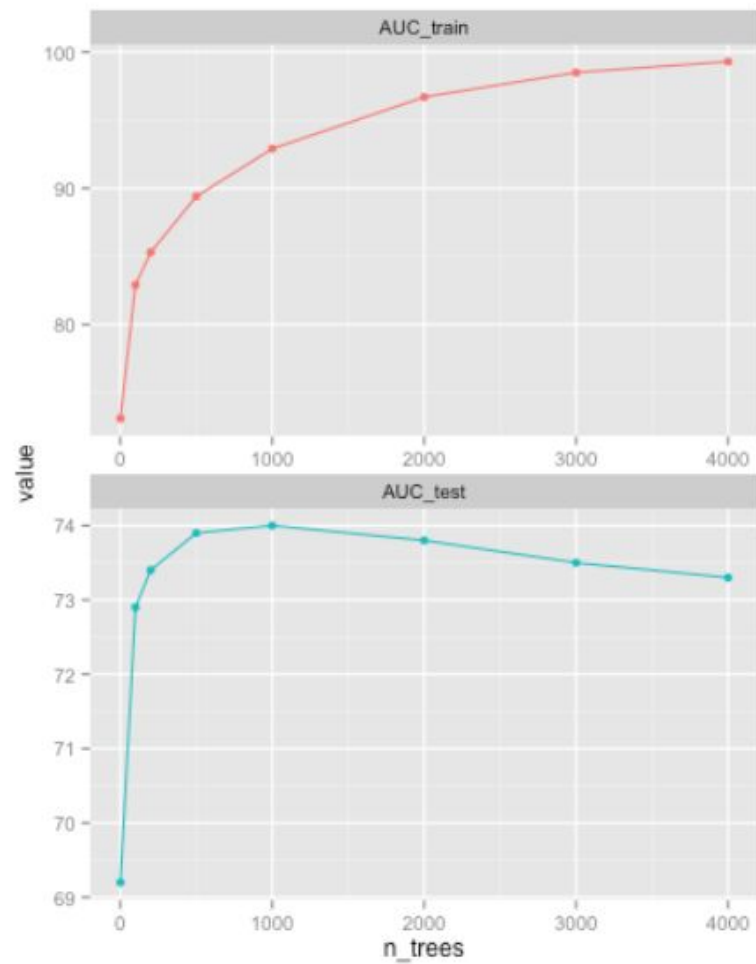
Was: \$743.99

Price: **\$743.96** & **FREE Shipping**. [Details](#)



Model	vCPU	Mem (GiB)
r3.8xlarge	32	244
x1.32xlarge	128	1,952

```
h2o.gbm(x, y, training_frame, model_id, checkpoint, ignore_const_cols = TRUE,
  distribution = c("AUTO", "gaussian", "bernoulli", "multinomial", "poisson",
    "gamma", "tweedie", "laplace", "quantile", "huber"), quantile_alpha = 0.5,
  tweedie_power = 1.5, huber_alpha = 0.9, ntrees = 50, max_depth = 5,
  min_rows = 10, learn_rate = 0.1, learn_rate_annealing = 1,
  sample_rate = 1, sample_rate_per_class, col_sample_rate = 1,
  col_sample_rate_change_per_level = 1, col_sample_rate_per_tree = 1,
  nbins = 20, nbins_top_level = 1024, nbins_cats = 1024,
  validation_frame = NULL, balance_classes = FALSE, class_sampling_factors,
  max_after_balance_size = 5, seed, build_tree_one_node = FALSE,
  nfolds = 0, fold_column = NULL, fold_assignment = c("AUTO", "Random",
"Modulo", "Stratified"), keep_cross_validation_predictions = FALSE,
  keep_cross_validation_fold_assignment = FALSE,
  score_each_iteration = FALSE, score_tree_interval = 0,
  stopping_rounds = 0, stopping_metric = c("AUTO", "deviance", "logloss",
"MSE", "AUC", "misclassification", "mean_per_class_error"),
  stopping_tolerance = 0.001, max_runtime_secs = 0, offset_column = NULL,
  weights_column = NULL, min_split_improvement = 1e-05,
  histogram_type = c("AUTO", "UniformAdaptive", "Random", "QuantilesGlobal",
    "RoundRobin"), max_abs_leafnode_pred, pred_noise_bandwidth = 0,
  categorical_encoding = c("AUTO", "Enum", "OneHotInternal", "OneHotExplicit",
    "Binary", "Eigen"))
```



Arno Candel in GBM, R, Technical, Tutorials | June 16, 2016

H2O GBM Tuning Tutorial for R

In this tutorial, we show how to build a well-tuned H2O GBM model for a supervised classification task, and use a small dataset to allow you to reproduce these results in a few minutes on a laptop. This script can run on hundreds of GBs large and H2O clusters with dozens of compute nodes.

machinelearningmastery.com/configure-gradient-boosting-algorithm/



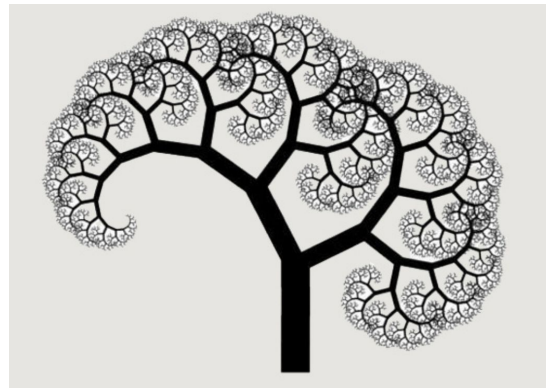
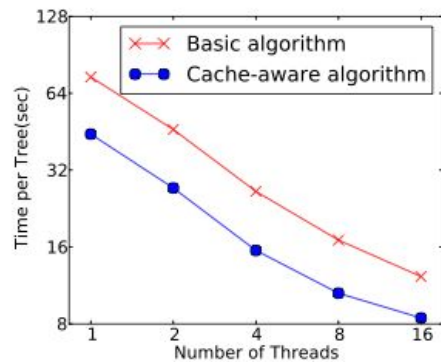
[Start Here](#)

Search...

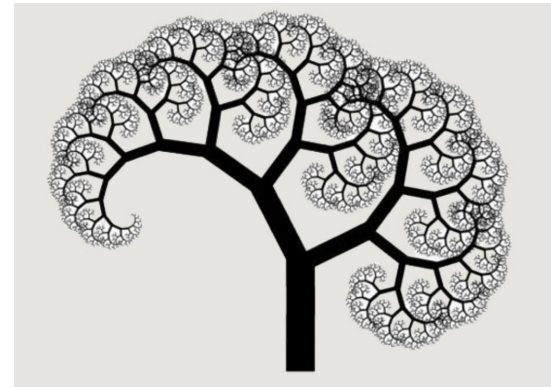
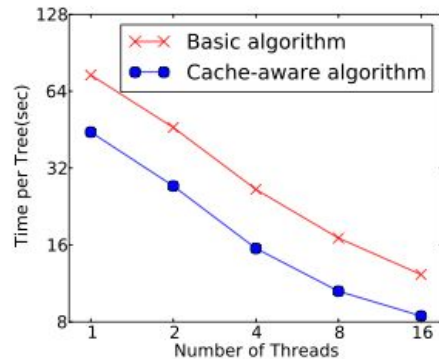
How to Configure the Gradient Boosting Algorithm

by Jason Brownlee on September 12, 2016 in XGBoost

XGBoost: A Scalable Tree Boosting System



XGBoost: A Scalable Tree Boosting System



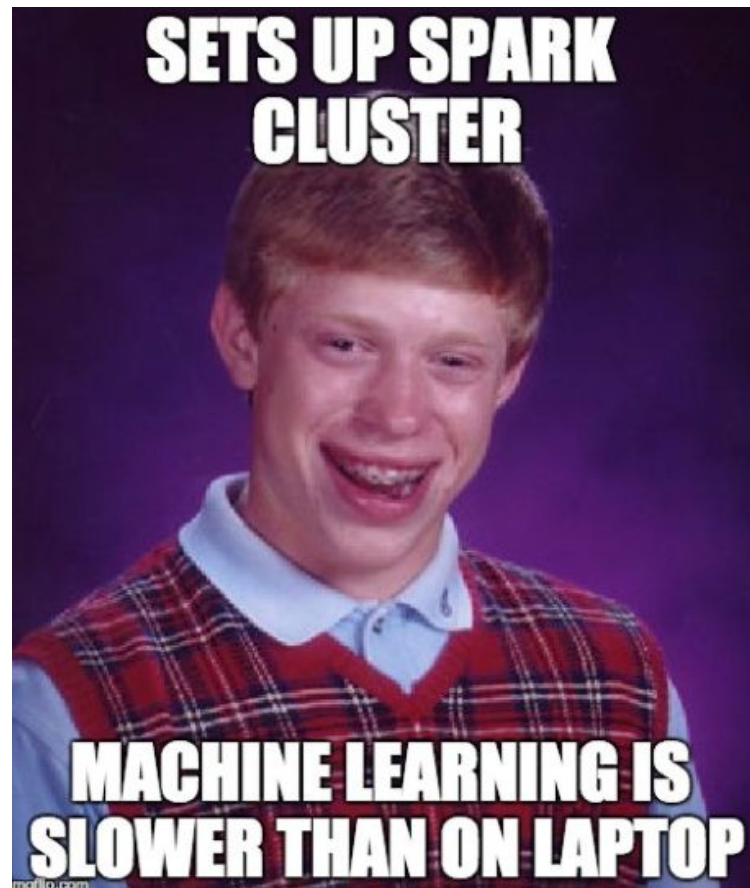
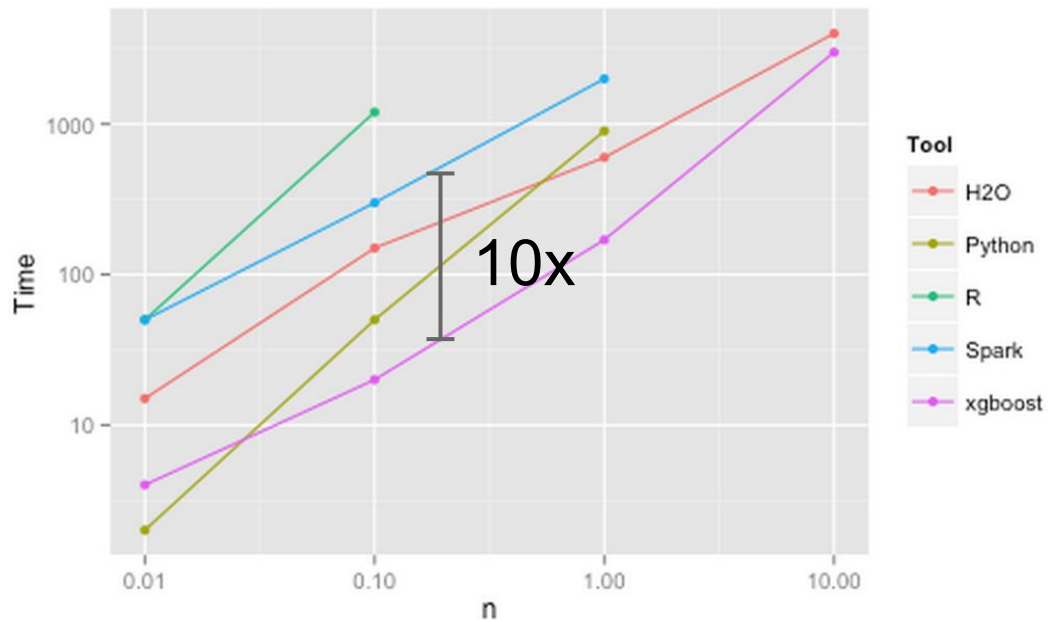
← → ↻ <https://cran.r-project.org/web/packages/xgboost/index.html>

xgboost: Extreme Gradient Boosting

← → ↻ <https://cran.r-project.org/web/packages/h2o/index.html>

h2o: R Interface for H2O





 szilard / **benchm-ml**

★ Star

1,203

Simple/limited/incomplete benchmark

 szilard / **benchm-ml**

★ Star

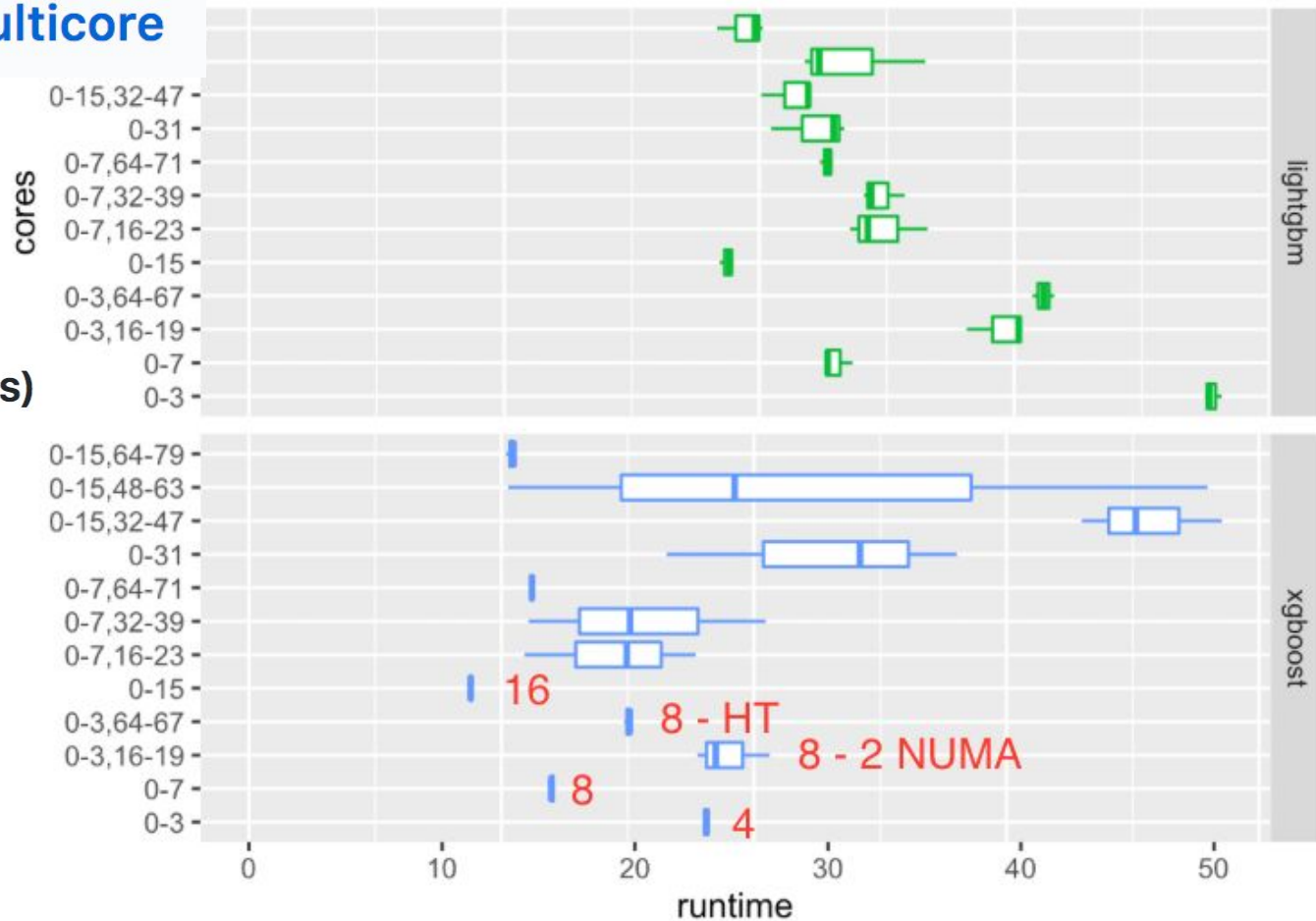
1,203

Simple/limited/incomplete benchmark

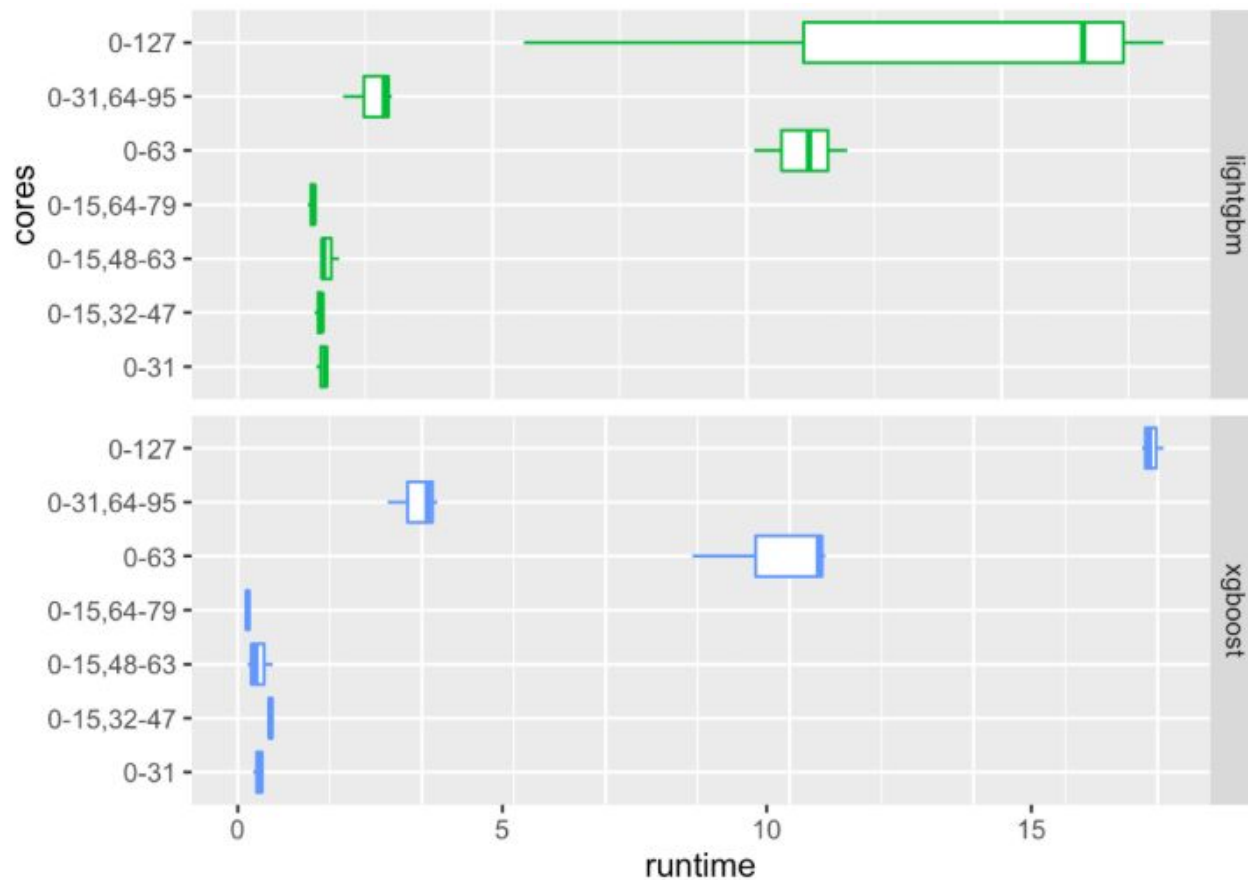
 szilard / **GBM-perf**

Tool	Version	Time[s] 1M	Time[s] 10M	AUC 1M	AUC 10M
h2o	cran 3.10.4.6	25	140	0.762	0.776
xgboost	cran 0.6-4	20	290	0.750	0.751
lightgbm	github 97ca38d	6	50	0.764	0.775

x1.32xlarge (128 cores)

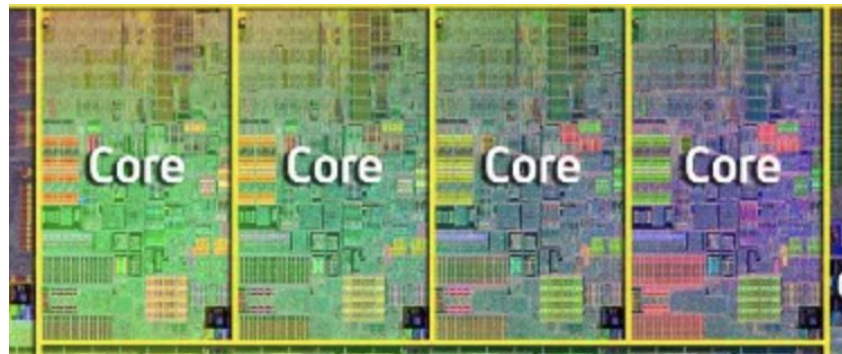


x1.32xlarge (128 cores)



16 cores vs 1:

Tool	1M data	10M/100M
lightgbm	4x	6x
xgboost	6x	4x
h2o	6x	12x



16 cores:

Tool	Time [s]
xgboost	4
lightgbm	0.5
h2o	10

GBM: 100 trees, depth 10 , learning rate 0.1

On r4.8xlarge (32 cores, 250GB RAM)

Tool	Version	Time[s] 1M	Time[s] 10M	AUC 1M	AUC 10M
h2o	cran 3.10.4.6	25	140	0.762	0.776
xgboost	cran 0.6-4	20	290	0.750	0.751
xgboost hist	github 6776292	20	170	0.766	0.772
lightgbm	github 97ca38d	6	50	0.764	0.775

With GPU support on p2.xlarge (Tesla K80, 12GB)



Tool	Version	Time[s] 1M	Time[s] 10M	AUC 1M	AUC 10M
h2o xgboost	deep water 3.11.0.266	20	180	0.715	0.708
xgboost hist	github 64c8f6f	6	50	0.750	0.740
lightgbm	github 1d5867b	30	120	0.771	0.789

Next: R code (hands-on)



✉ spafka@gmail.com

🐦 [@DataScienceLA](https://twitter.com/DataScienceLA)

in linkedin.com/in/szilard

🐙 github.com/szilard