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 🇺🇸🇭🇺🇪🇺

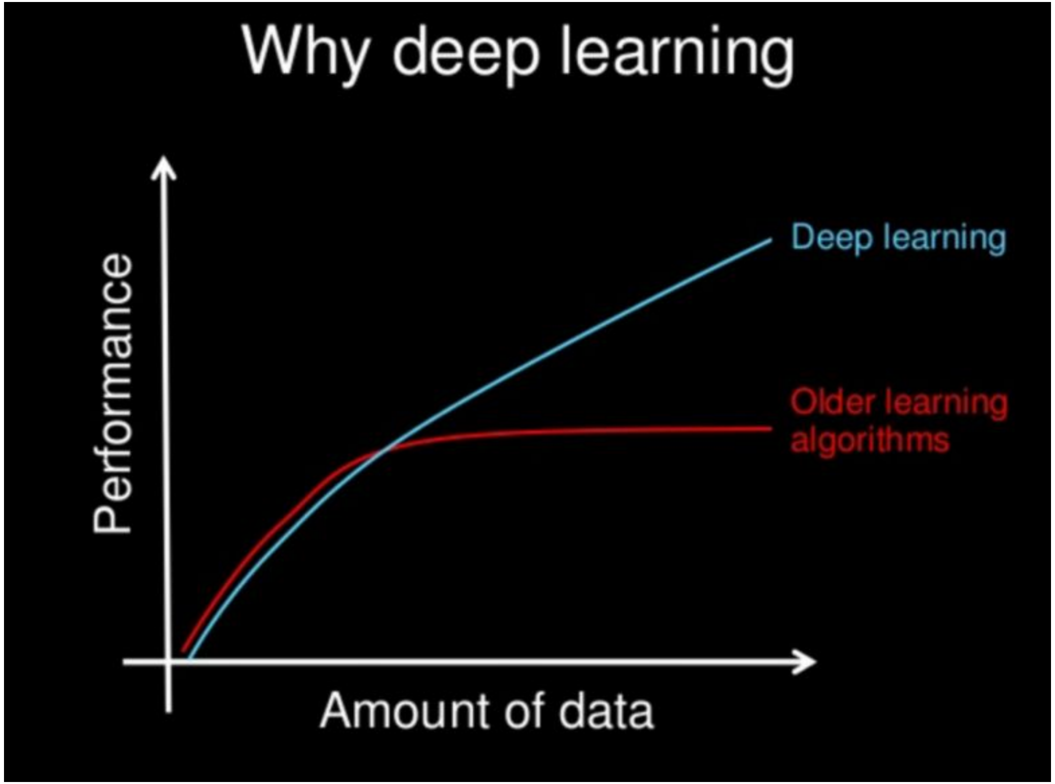
Santa Monica, California   linkedin.com/in/szilard

198 Following   **2,067** Followers

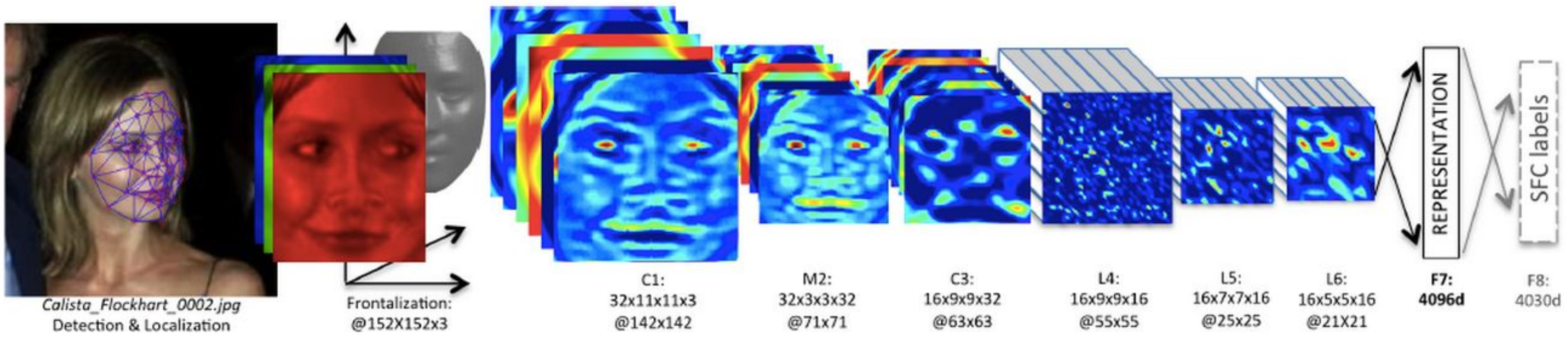
Disclaimer:

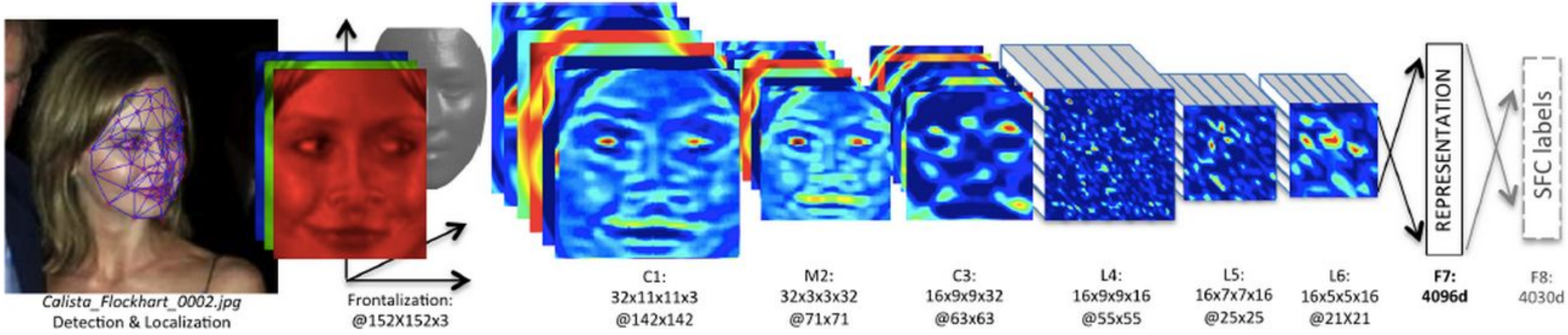
I am not representing my employer (Epoch) in this talk

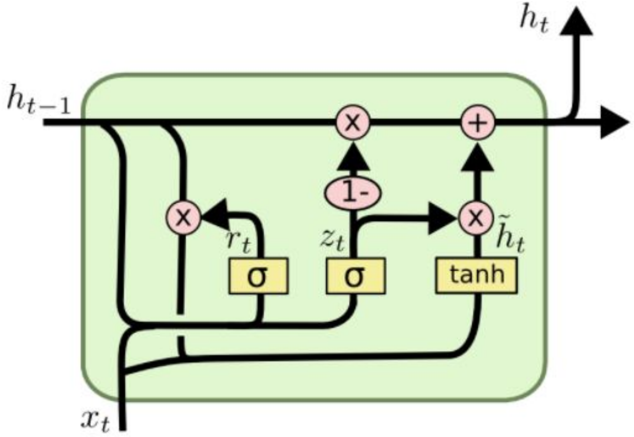
I cannot confirm nor deny if Epoch is using any of the methods, tools, results etc. mentioned in this talk

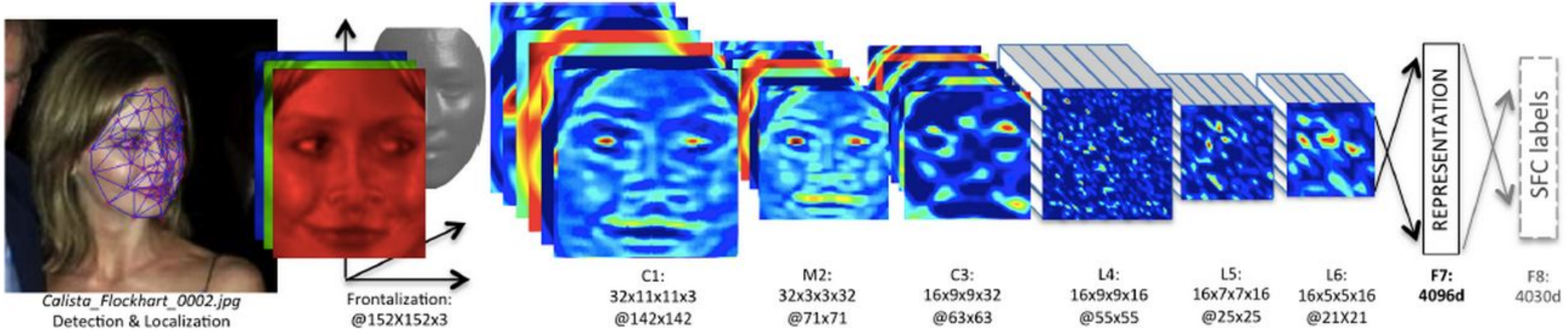


Source: Andrew Ng

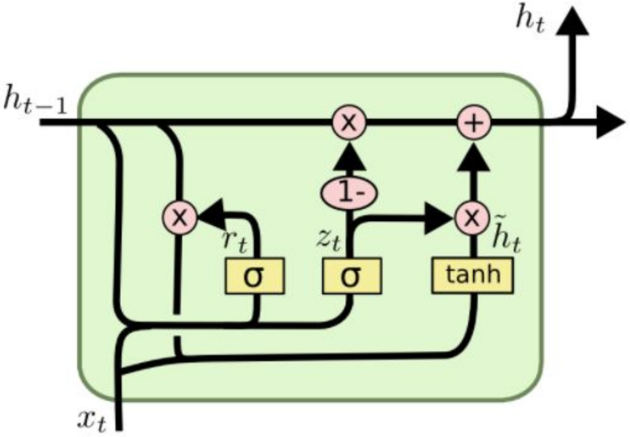

Calista_Flockhart_0002.jpg
Detection & Localization
Frontalization:
@152X152x3
C1:
32x11x11x3
@142x142
M2:
32x3x3x32
@71x71
C3:
16x9x9x32
@63x63
L4:
16x9x9x16
@55x55
L5:
16x7x7x16
@25x25
L6:
16x5x5x16
@21X21
REPRESENTATION
F7:
4096d
SFC labels
F8:
4030d


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
REPRESENTATION
F7:
4096d
SFC labels
F8:
4030d

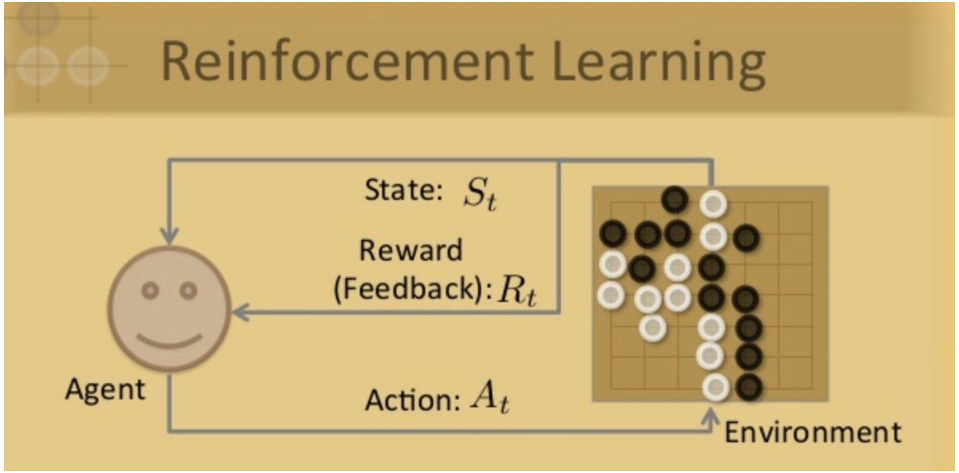

h_t
h_{t-1}
r_t
z_t
$\tilde{h}_t$
1-
σ
σ
tanh
x_t



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
REPRESENTATION
F7:
4096d
SFC labels
F8:
4030d



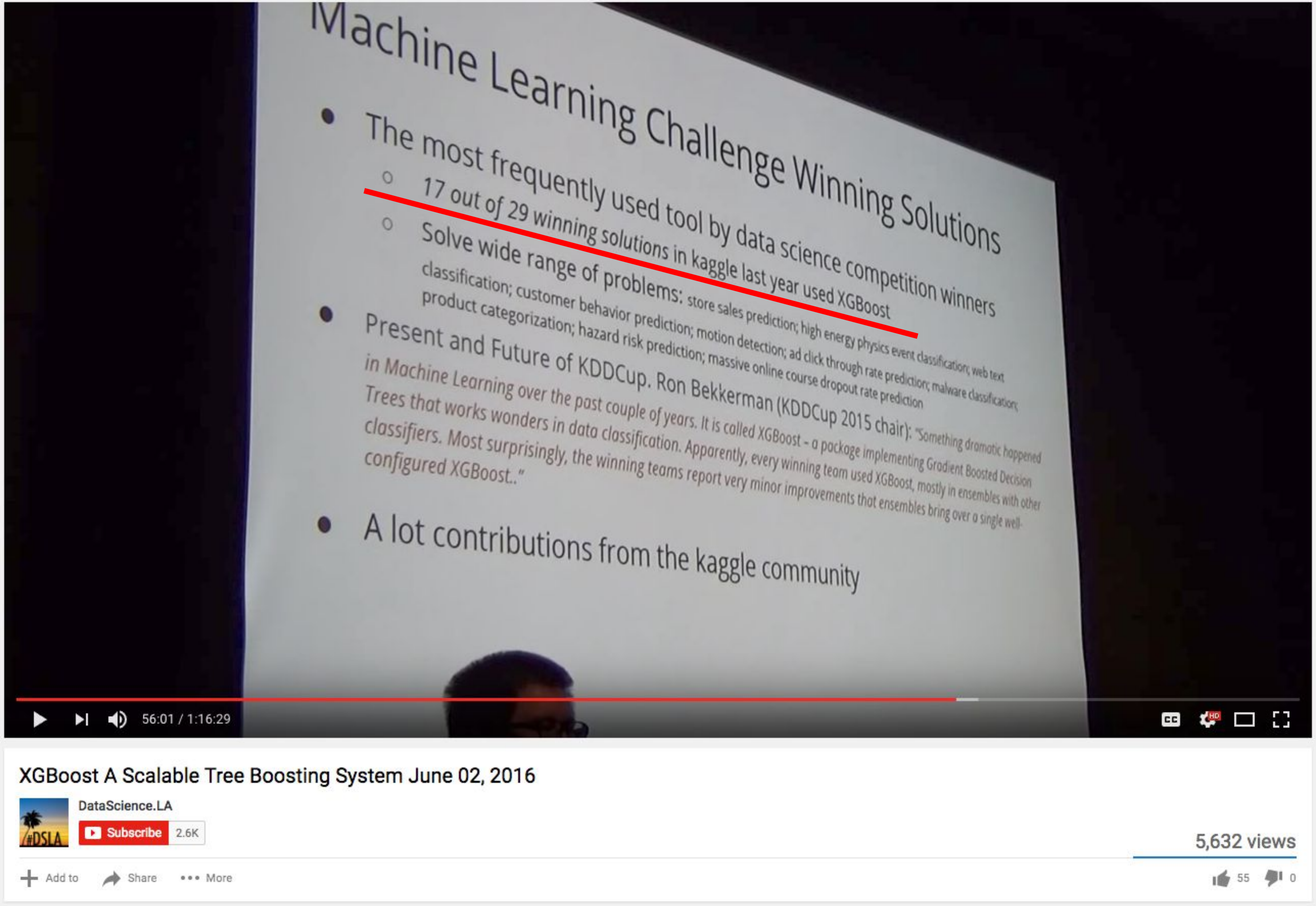
h_t
h_{t-1}
1-
r_t
z_t
$\tilde{h}_t$
σ
σ
tanh
x_t



Reinforcement Learning
State: S_t
Reward
(Feedback): R_t
Agent
Action: A_t
Environment

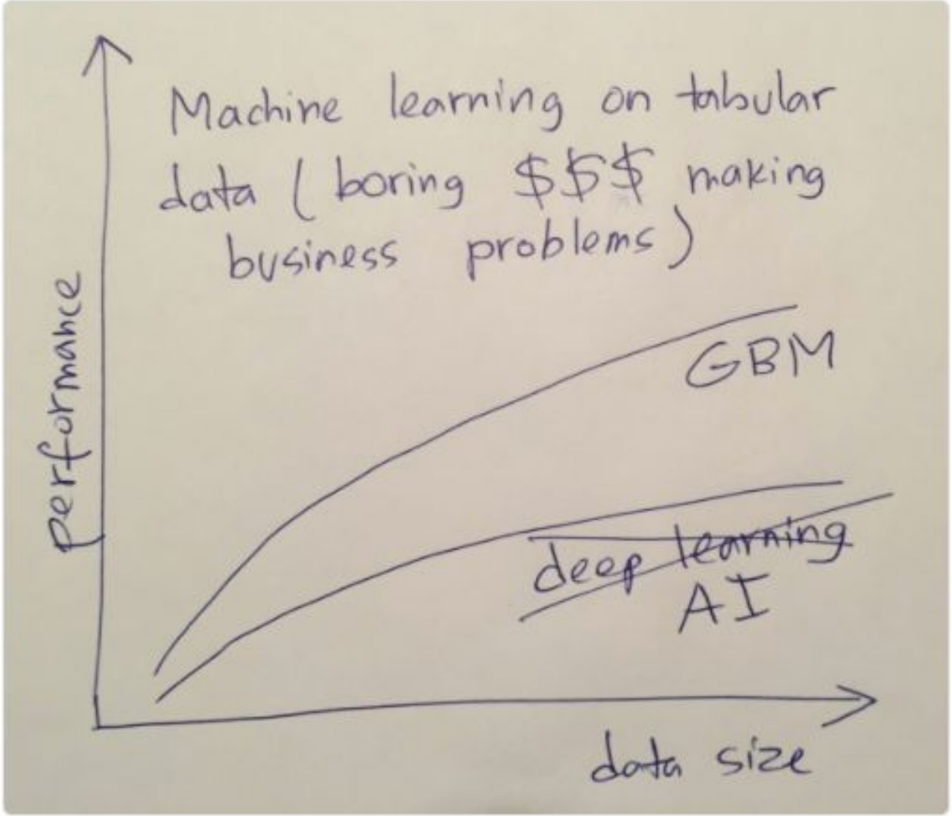

THAT'S NO AI
ITS JUST MORE IF STATEMENTS
imgflip.com

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
5,632 views
Add to
Share
More
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/benchm...



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

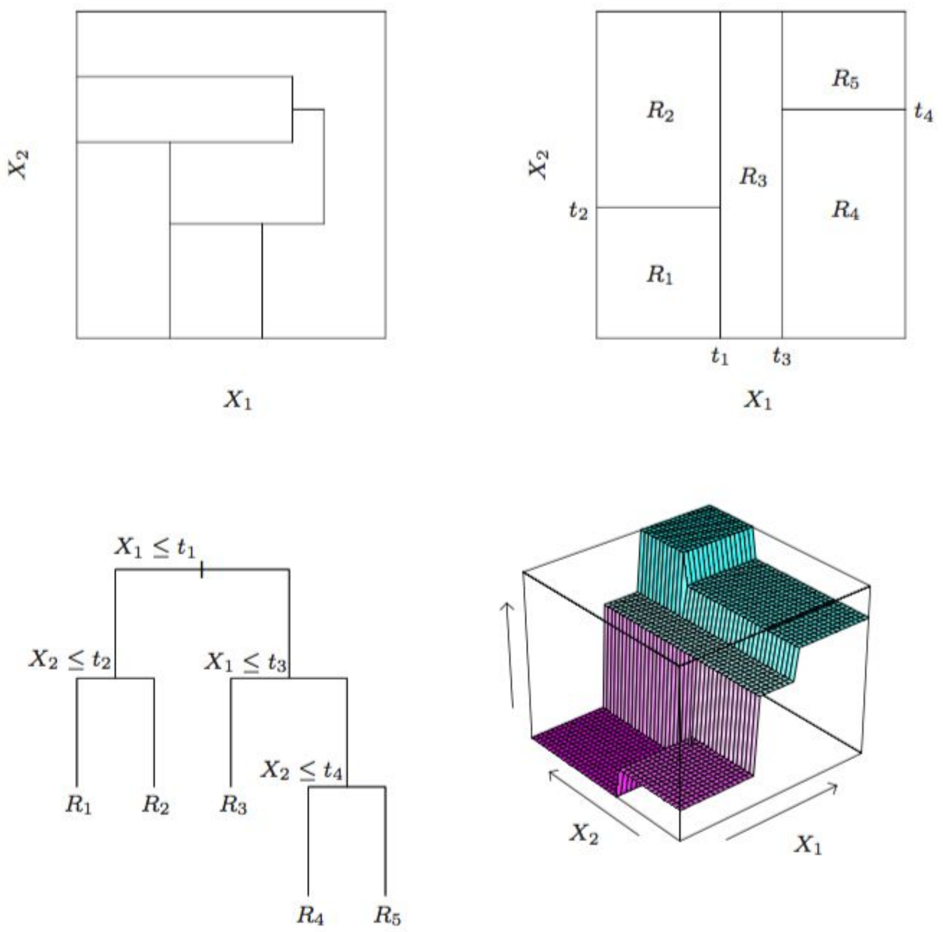
AVG	1ST	2ND
RF	0.727	0.207
ANN	0.053	0.172
BSTDT	0.059	0.228
SVM	0.043	0.195
LR	0.089	0.132
BAGDT	0.002	0.012
KNN	0.023	0.045
BSTST	0.004	0.009
PRC	0	0
NB	0	0

An Empirical Comparison of Supervised Learning Algorithms

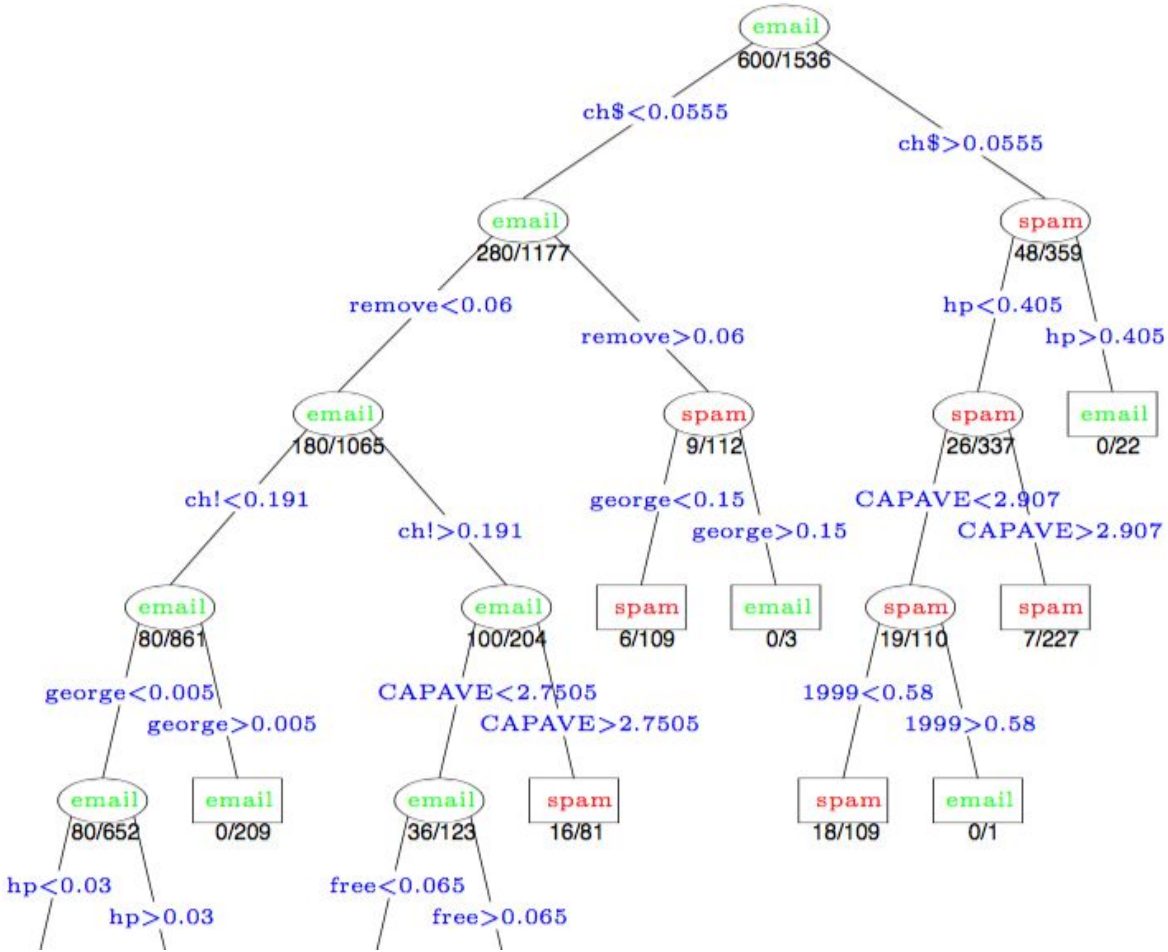
http://www.cs.cornell.edu/~alexn/papers/empirical.icml06.pdf

An Empirical Evaluation of Supervised Learning in High Dimensions

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



X_2
X_1
R_2
R_5
t_4
X_2
R_3
t_2
R_4
R_1
t_1
t_3
X_1
$X_1 \leq t_1$
$X_2 \leq t_2$
$X_1 \leq t_3$
$X_2 \leq t_4$
R_1
R_2
R_3
R_4
R_5
X_2
X_1


email
600/1536
ch$<0.0555
ch$>0.0555
email
280/1177
spam
48/359
remove<0.06
remove>0.06
hp<0.405
hp>0.405
email
180/1065
spam
9/112
spam
26/337
email
0/22
ch!<0.191
ch!>0.191
george<0.15
george>0.15
CAPAVE<2.907
CAPAVE>2.907
email
80/861
email
100/204
spam
6/109
email
0/3
spam
19/110
spam
7/227
george<0.005
george>0.005
CAPAVE<2.7505
CAPAVE>2.7505
1999<0.58
1999>0.58
email
80/652
email
0/209
email
36/123
spam
16/81
spam
18/109
email
0/1
hp<0.03
hp>0.03
free<0.065
free>0.065

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
   $$\mathrm{err}_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}.$$
   (c) Compute $\alpha_m = \log((1 - \mathrm{err}_m)/\mathrm{err}_m)$.
   (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 $G(x) = \mathrm{sign}\left[\sum_{m=1}^M \alpha_m G_m(x)\right]$.

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 $i = 1, 2, \ldots, N$ compute

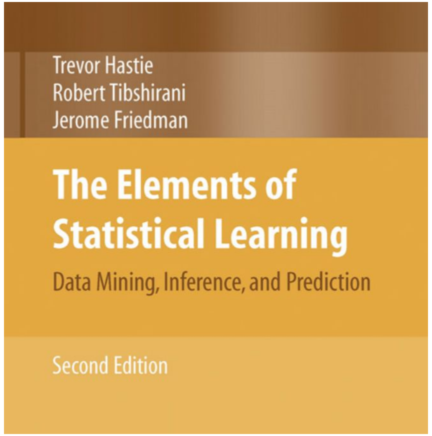
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$

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

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

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

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



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

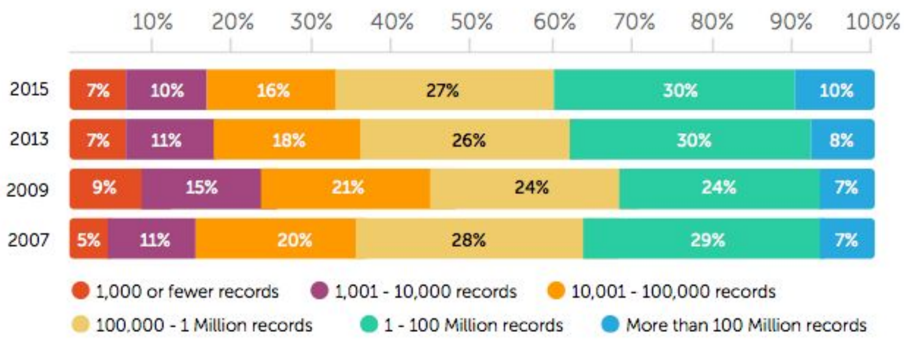
48% 100MB-10GB

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

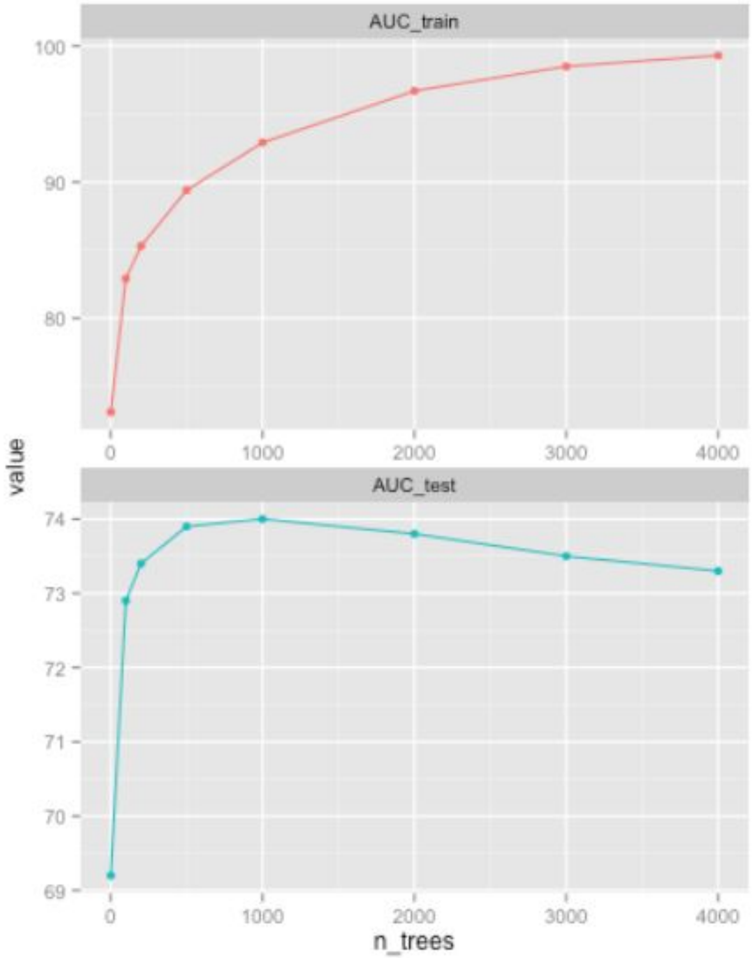
Be the first to review this item

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"))
```



AUC_train
100
90
80
0
1000
2000
3000
4000
value
AUC_test
74
73
72
71
70
69
0
1000
2000
3000
4000
n_trees

blog.h2o.ai/2016/06/h2o-gbm-tuning-tutorial-for-r/

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 ca dreds of GBs large and H2O clusters with dozens of compute nodes.

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

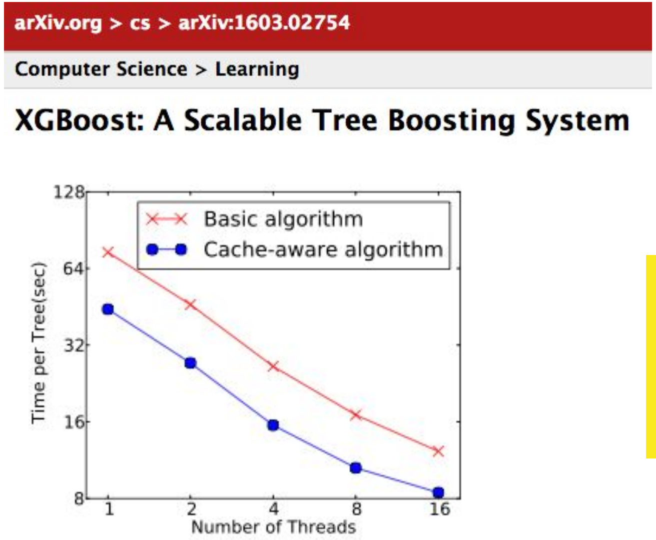
MACHINE LEARNING MASTERY

Start Here

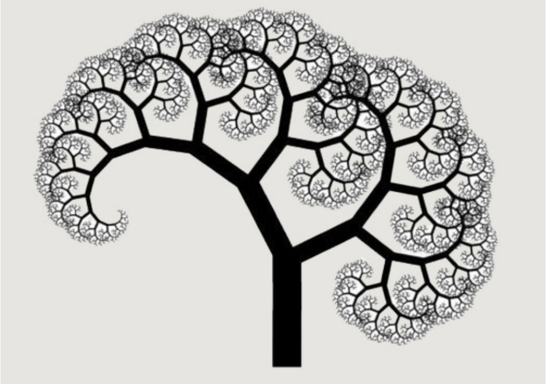
Search...

How to Configure the Gradient Boosting Algorithm

by **Jason Brownlee** on September 12, 2016 in **XGBoost**


arXiv.org > cs > arXiv:1603.02754
Computer Science > Learning
XGBoost: A Scalable Tree Boosting System
Basic algorithm
Cache-aware algorithm
Time per Tree(sec)
128
64
32
16
8
1
2
4
8
16
Number of Threads

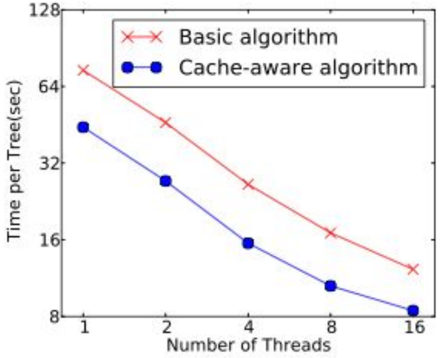
H_2O.ai



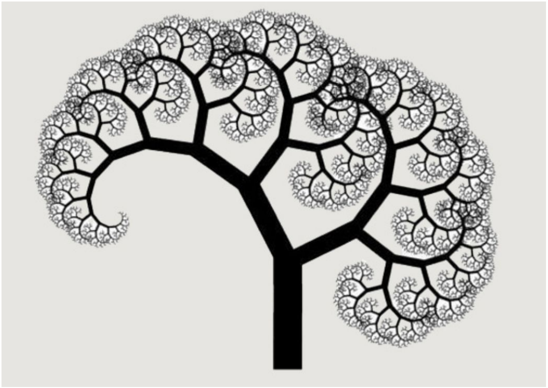
arXiv.org > cs > arXiv:1603.02754

Computer Science > Learning

XGBoost: A Scalable Tree Boosting System



H2O.ai



R

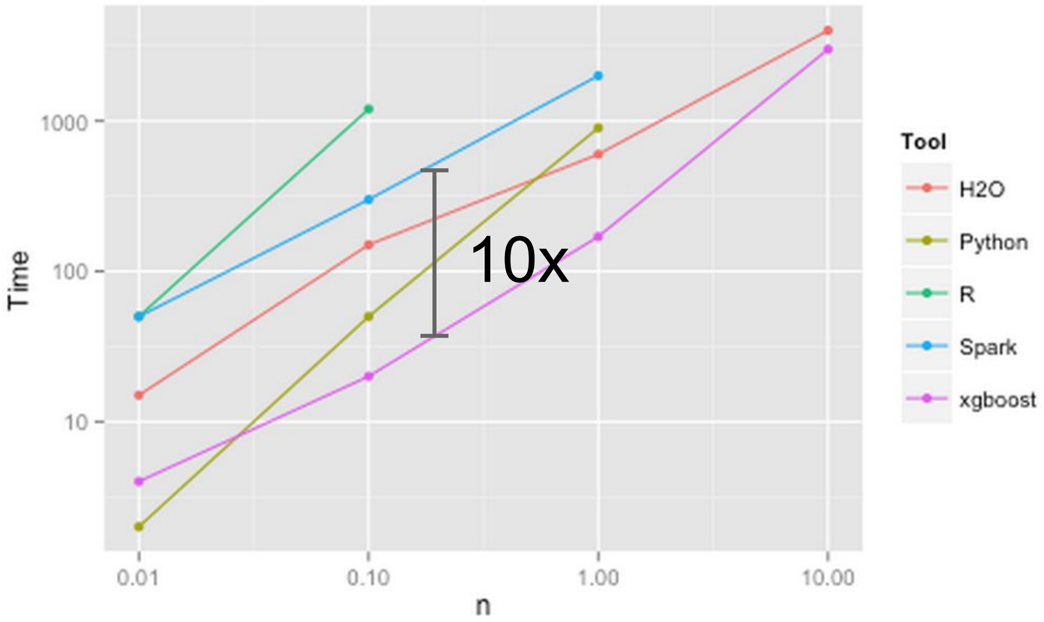
https://cran.r-project.org/web/pa

xgboost: Extreme Gradient Boosting

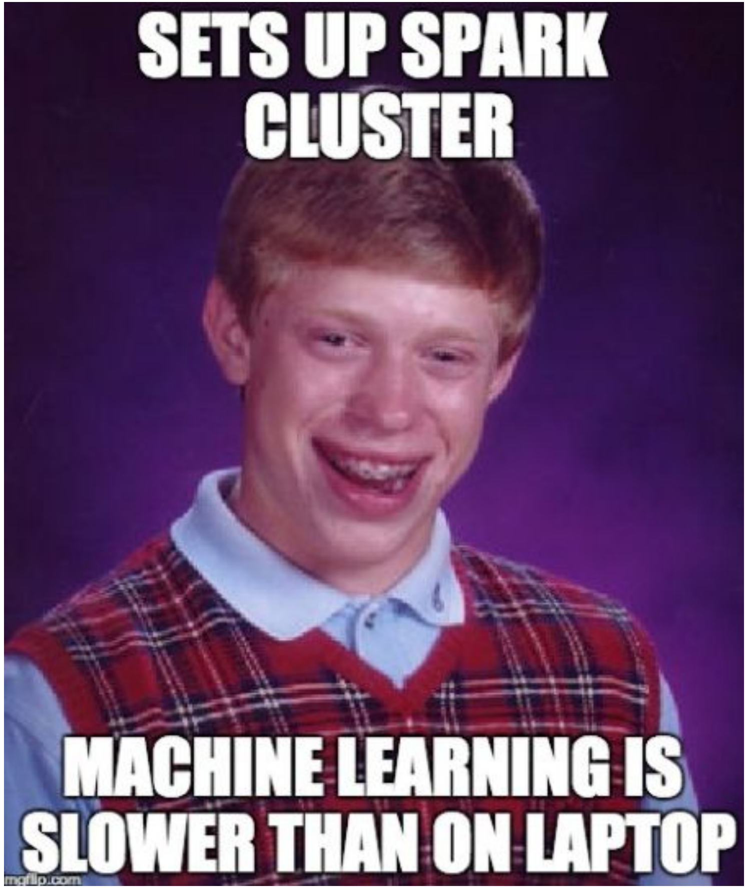
https://cran.r-project.org/

h2o: R Interface for H2O

PyData



Tool
H2O
Python
R
Spark
xgboost
10x
Time
1000
100
10
n
0.01
0.10
1.00
10.00



SETS UP SPARK CLUSTER
MACHINE LEARNING IS SLOWER THAN ON LAPTOP

szilard / benchm-ml

Star 1,203

Simple/limited/incomplete benchmark

szilard / benchm-ml

Star 1,203

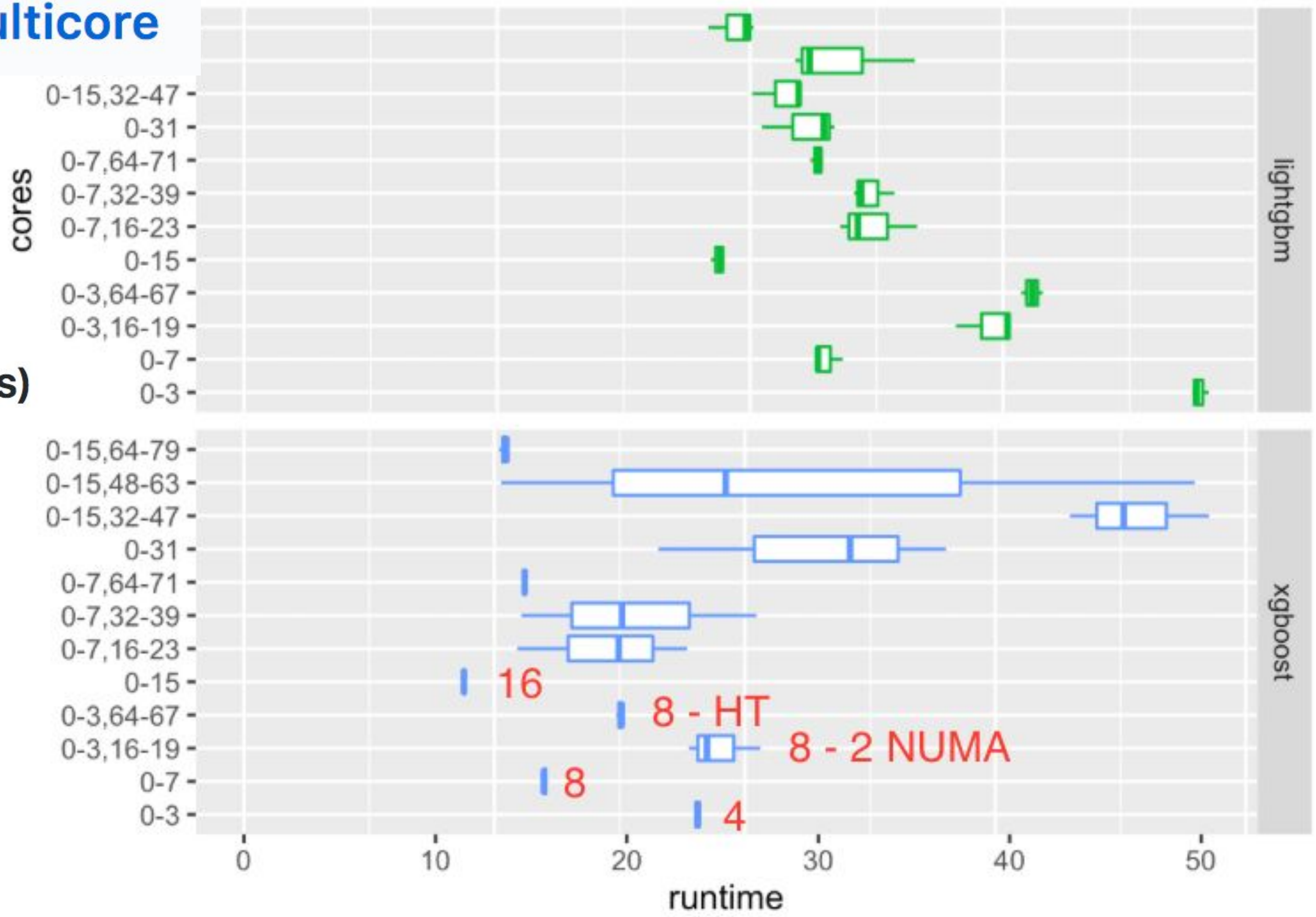
Simple/limited/incomplete benchmark

szilard / GBM-perf

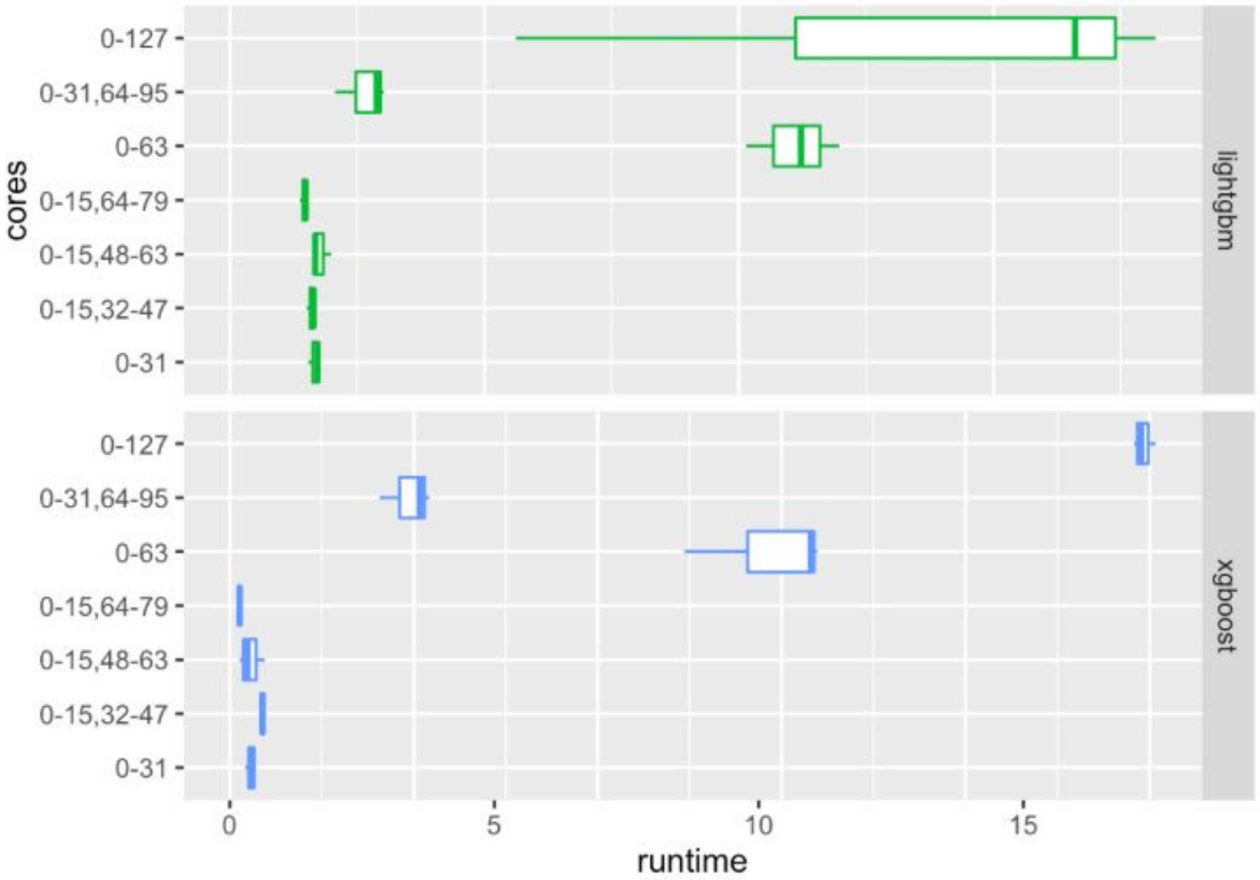
Tool	Version	Time[s] 1M	Time[s] 10M	AUC 1M	AUC 10M
h2o	cran 3.10.4.6	25	140	0.762	0.776
xgboost	cran 0.6-4	20	290	0.750	0.751
lightgbm	github 97ca38d	6	50	0.764	0.775

szilard / **GBM-multicore**

x1.32xlarge (128 cores)

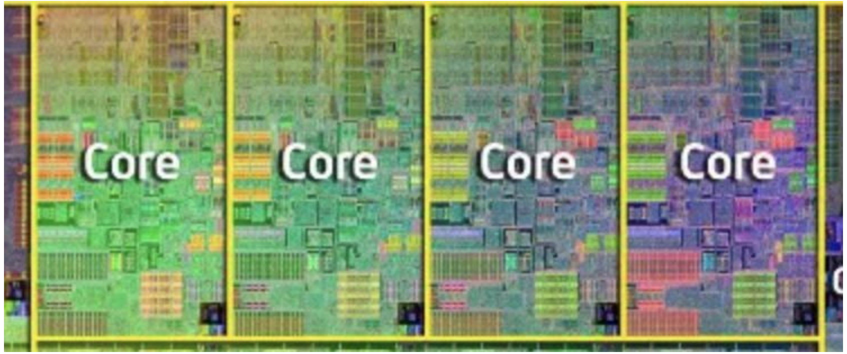


x1.32xlarge (128 cores)



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

szilard / GBM-perf

GBM: `100` trees, depth `10`, learning rate `0.1`

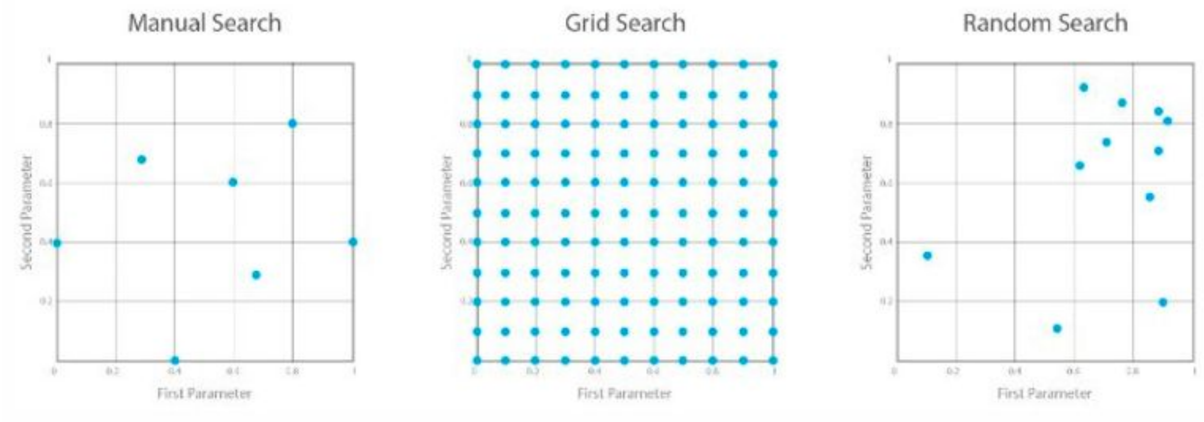
On r4.8xlarge (32 cores, 250GB RAM)

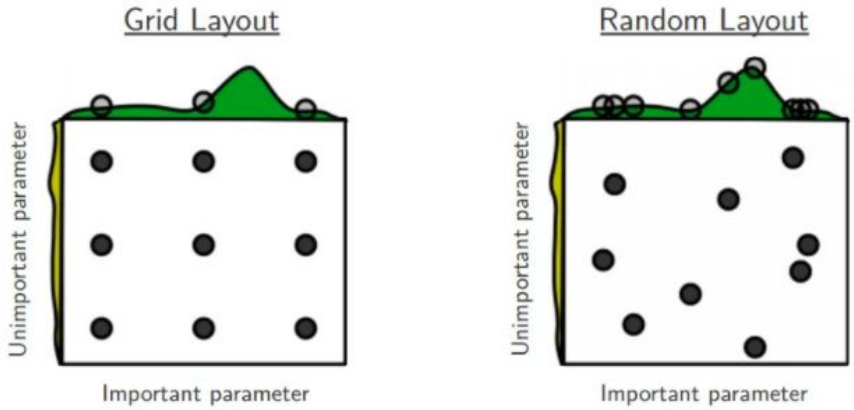
Tool	Version	Time[s] 1M	Time[s] 10M	AUC 1M	AUC 10M
h2o	cran 3.10.4.6	25	140	0.762	0.776
xgboost	cran 0.6-4	20	290	0.750	0.751
xgboost hist	github 6776292	20	170	0.766	0.772
lightgbm	github 97ca38d	6	50	0.764	0.775

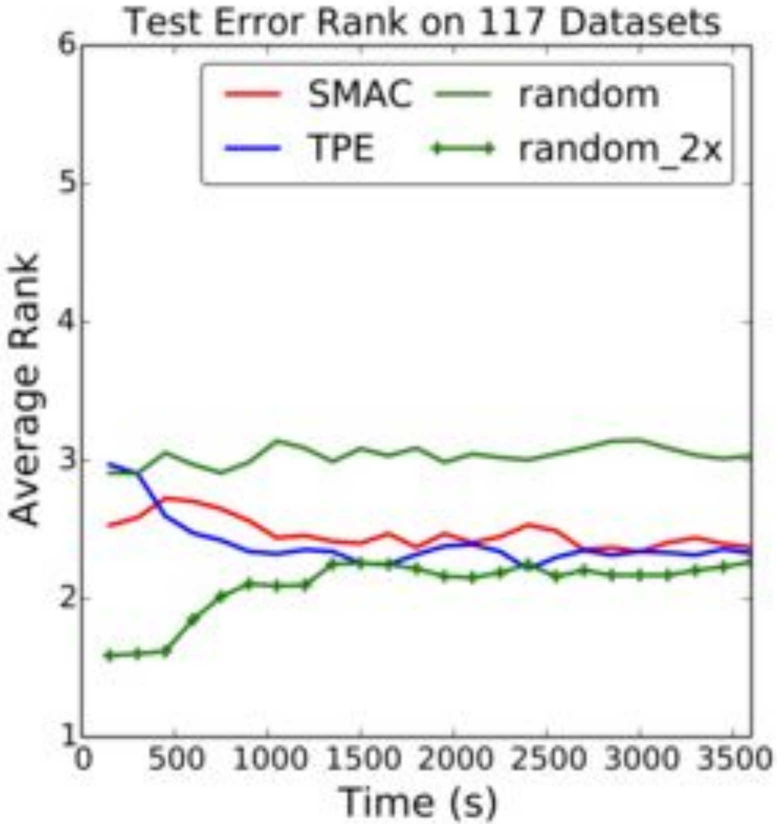
With GPU support on p2.xlarge (Tesla K80, 12GB)

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

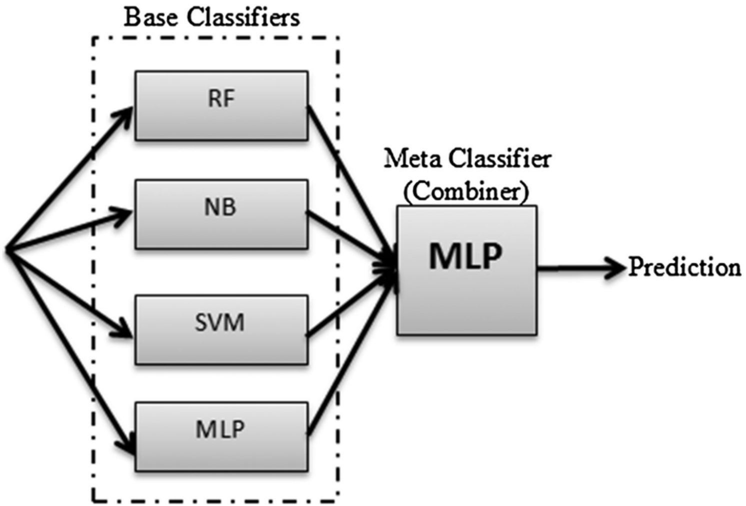
R code (hands-on)


Manual Search
Grid Search
Random Search
Second Parameter
First Parameter


Grid Layout
Random Layout
Unimportant parameter
Important parameter



- Gaussian Processes (GP)
- Tree of Parzen Estimators (TPE)
- Sequential Model-based Algorithm Configuration (SMAC)



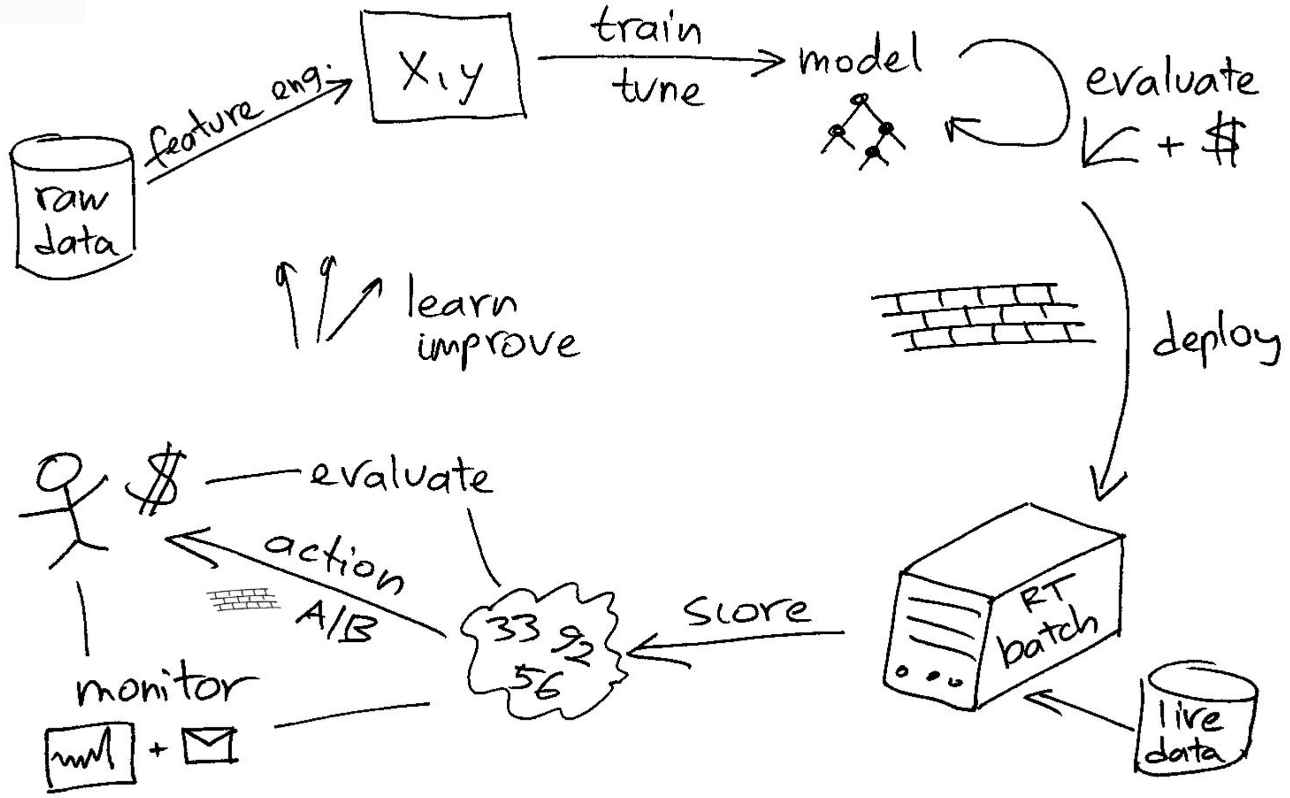
$$n\Bigg\{\overbrace{\begin{bmatrix}\quad X \quad\end{bmatrix}}^{m}\quad\begin{bmatrix} y \end{bmatrix}$$

"Level-zero" data

$$n\Bigg\{\begin{bmatrix}p_1\end{bmatrix}\cdots\begin{bmatrix}p_L\end{bmatrix}\quad\begin{bmatrix}y\end{bmatrix}\rightarrow n\Bigg\{\overbrace{\begin{bmatrix}\quad Z\quad\end{bmatrix}}^{L}\quad\begin{bmatrix}y\end{bmatrix}$$

"Level-one" data

szilard / ml-prod



More:

szilard / benchm-ml

Unwatch 149 | Unstar 1,381 | Fork 248

szilard / teach-data-science-UCLA-master-appl-stats

szilard / teach-ML-CEU-master-bizanalytics

szilard / talks

szilard / ML-scoring

szilard / GBM-tune

szilard / GBM-perf

szilard / GBM-multicore

GitHubGist Search...

szilard / h2o_scoring.R

szilard / ML_with_H2O.R

spafka@gmail.com

@DataScienceLA

linkedin.com/in/szilard

github.com/szilard