

Gradient Boosting Machines (GBMs) in the Age of LLMs and ChatGPT

Szilard Pafka, PhD
Chief Scientist, Epoch

R+AI Conference
Nov 2025, Online



Edit profile

Szilard Pafka

@SzilardPafka

physics PhD, chief (data) scientist, meetup organizer, (visiting) professor, machine learning benchmarks

📍 The Woodlands, Texas 🇺🇸 🔗 szilard.github.io/aboutme/

📅 Joined February 2014

224 Following 4,589 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

Machine Learning Algorithms



KDD-2001 San Francisco,
CA August 26-29

[More information](#)

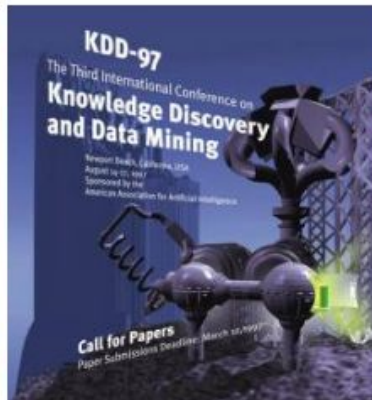


KDD-1996 Portland, OR
August 2-4

[More information](#)

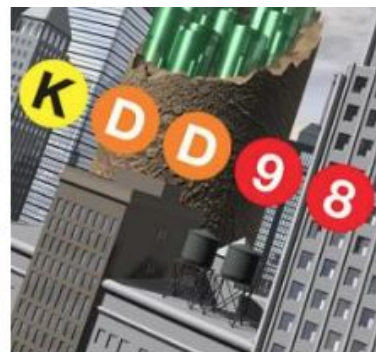
KDD-2002 Edmonton, AB
July 23-26

[More information](#)



KDD-1997 Newport
Beach, CA August 14-17

[More information](#)



KDD-1998 New York, NY
August 27-31

[More information](#)

KDD-1993 workshop
Washington, D.C., July 11-12

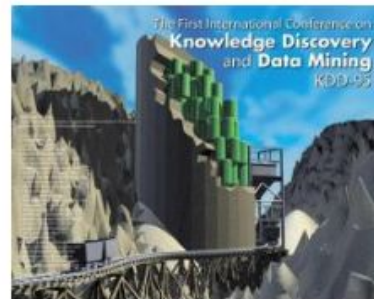
[More information](#)

KDD-1989 workshop
Detroit, MI, August 20



KDD-2000 Boston, MA
August 20-23

[More information](#)

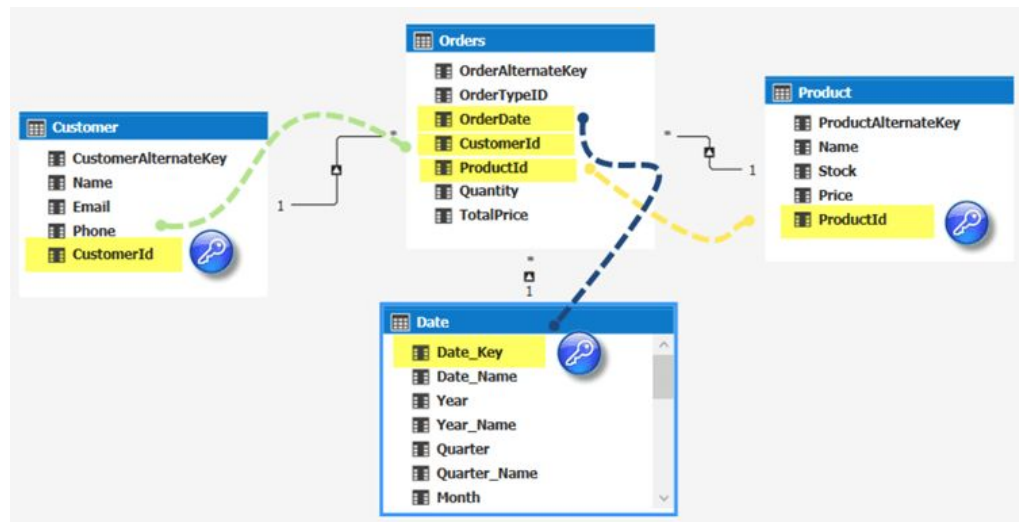


KDD-1995 Montreal, QC
August 20-21

[More information](#)

	A	B	C	D	E	F
1	Country ▼	Salesperson ▼	Order Date ▼	OrderID ▼	Units ▼	Order Amount ▼
2	USA	Fuller	1/01/2011	10392	13	1,440.00
3	UK	Gloucester	2/01/2011	10397	17	716.72
4	UK	Bromley	2/01/2011	10771	18	344.00
5	USA	Finchley	3/01/2011	10393	16	2,556.95
6	USA	Finchley	3/01/2011	10394	10	442.00
7	UK	Gillingham	3/01/2011	10395	9	2,122.92
8	USA	Finchley	6/01/2011	10396	7	1,903.80
9	USA	Callahan	8/01/2011	10399	17	1,765.60
10	USA	Fuller	8/01/2011	10404	7	1,591.25
11	USA	Fuller	9/01/2011	10398	11	2,505.60
12	USA	Coghill	9/01/2011	10403	18	855.01
13	USA	Finchley	10/01/2011	10401	7	3,868.60
14	USA	Callahan	10/01/2011	10402	11	2,713.50
15	UK	Rayleigh	13/01/2011	10406	15	1,830.78
16	USA	Callahan	14/01/2011	10408	10	1,622.40
17	USA	Farnham	14/01/2011	10409	19	319.20
18	USA	Farnham	15/01/2011	10410	16	802.00

	A	B	C	D	E	F
1	Country ▼	Salesperson ▼	Order Date ▼	OrderID ▼	Units ▼	Order Amount ▼
2	USA	Fuller	1/01/2011	10392	13	1,440.00
3	UK	Gloucester	2/01/2011	10397	17	716.72
4	UK	Bromley	2/01/2011	10771	18	344.00
5	USA	Finchley	3/01/2011	10393	16	2,556.95
6	USA	Finchley	3/01/2011	10394	10	442.00
7	UK	Gillingham	3/01/2011	10395	9	2,122.92
8	USA	Finchley	6/01/2011	10396	7	1,903.80
9	USA	Callahan	8/01/2011	10399	17	1,765.60
10	USA	Fuller	8/01/2011	10404	7	1,591.25
11	USA	Fuller	9/01/2011	10398	11	2,505.60
12	USA	Coghill	9/01/2011	10403	18	855.01
13	USA	Finchley	10/01/2011	10401	7	3,868.60
14	USA	Callahan	10/01/2011	10402	11	2,713.50
15	UK	Rayleigh	13/01/2011	10406	15	1,830.78
16	USA	Callahan	14/01/2011	10408	10	1,622.40
17	USA	Farnham	14/01/2011	10409	19	319.20
18	USA	Farnham	15/01/2011	10410	16	802.00



best boosting AUC? #15

✓ Closed



szilard opened on May 31, 2015

@tqchen @hetong007 I'm trying to get a good AUC with boosting

best boosting AUC? #15

✓ Closed



szilard opened on May 31, 2015

@tqchen @hetong007 I'm trying to get a good AUC with boosting

☆ 1.9k stars

👁 145 watching

🔗 332 forks

MODEL	1ST	2ND			
BST-DT	0.580	0.228	AVG	1ST	2ND
RF	0.390	0.525	RF	0.727	0.207
BAG-DT	0.030	0.232	ANN	0.053	0.172
SVM	0.000	0.008	BSTDT	0.059	0.228
ANN	0.000	0.007	SVM	0.043	0.195
KNN	0.000	0.000	LR	0.089	0.132
BST-STMP	0.000	0.000	BAGDT	0.002	0.012
DT	0.000	0.000	KNN	0.023	0.045
LOGREG	0.000	0.000	BSTST	0.004	0.009
NB	0.000	0.000	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

	gbm_1.5-3.tar.gz	2005-10-07 22:49 249K
	gbm_1.5-5.tar.gz	2006-01-21 12:58 249K
	gbm_1.5-7.tar.gz	2006-04-18 11:58 254K
	gbm_1.5.tar.gz	2005-05-09 22:56 250K
	gbm_1.6-1.tar.gz	2007-06-14 08:29 257K

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>

	randomForest_4.5-12.tar.gz	2005-06-21 09:36 80K
	randomForest_4.5-15.tar.gz	2005-09-22 19:35 81K
	randomForest_4.5-16.tar.gz	2006-01-24 10:21 81K
	randomForest_4.5-18.tar.gz	2006-12-10 16:07 67K
	randomForest_4.5-19.tar.gz	2007-10-16 20:38 67K

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; risk score, behavior, fraud, 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

XGBoost A Scalable Tree Boosting System June 02, 2016

26,599 views

212 1 SHARE SAVE ...



DataScience.LA
Published on Jun 3, 2016

SUBSCRIBE 3.4K

3. Parameter tuning and ensembling

```
# train xgboost
xgb <- xgboost(data = data.matrix(tr
              label = train$destino
              eta = 0.001,
              max_depth = 15,
              nround=25,
              subsample = 0.5,
              colsample_bytree = 0.
              seed = 1,
              eval_metric = "merror
              objective = "multi:sc
              num_class = 12,
              nthread = 4
              )
```

2:58 / 4:06

What Kaggle has learned from almost a million data scientists - Anthony Goldbloom

18,153 views



O'Reilly

Published on May 25, 2017



Gilberto Titericz · 1st

Data Scientist at NVIDIA Rapids

4mo ...

In my experience GBMs are, by far, the best for tabular structured data.

Like



35

Reply



Gilberto Titericz • 1st

Data Scientist at NVIDIA Rapids

4mo ...

In my experience GBMs are, by far, the best for tabular structured data.

Like



35

Reply



Szilard Pafka

physics PhD, chief (data) scientist, meetup organizer, (visiting) professor, ...

1mo •



Best algo for tabular data? (most often)

You can see how people vote. [Learn more](#)

Gradient Boosted Trees

92%

Neural Nets / Deep Learning

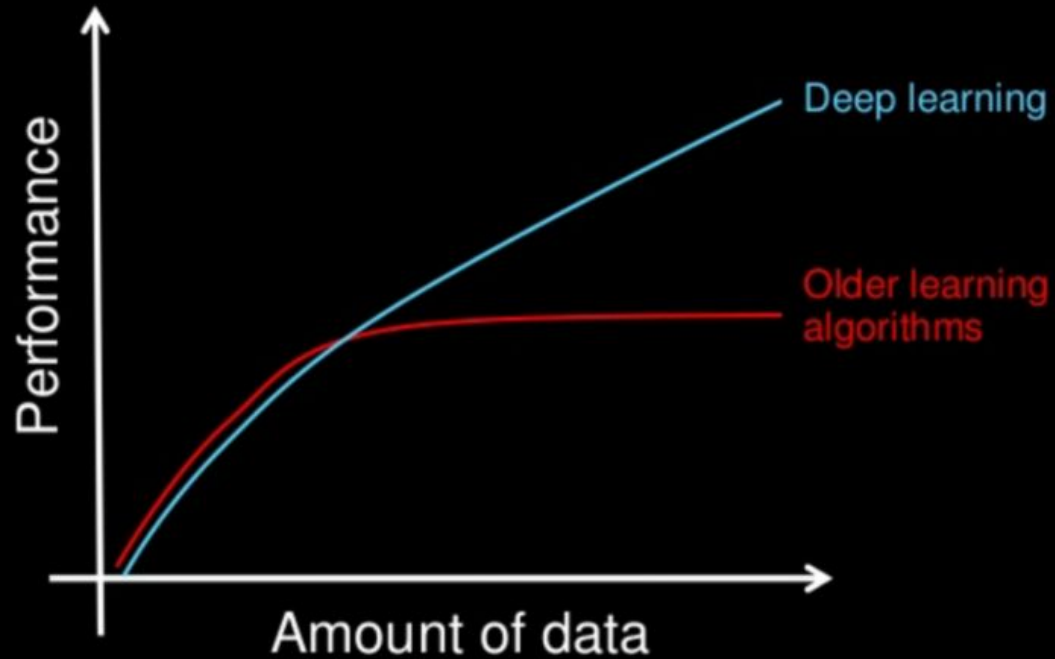
3%

Other

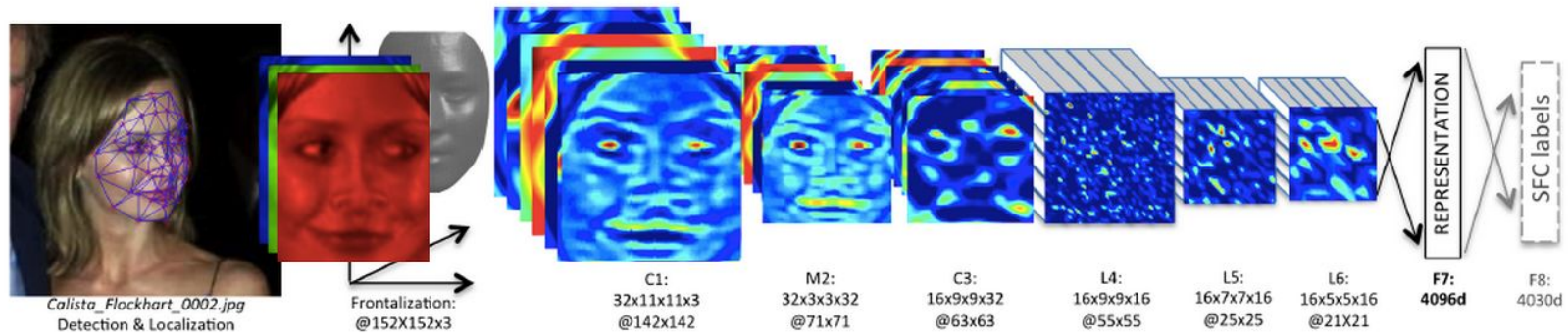
6%

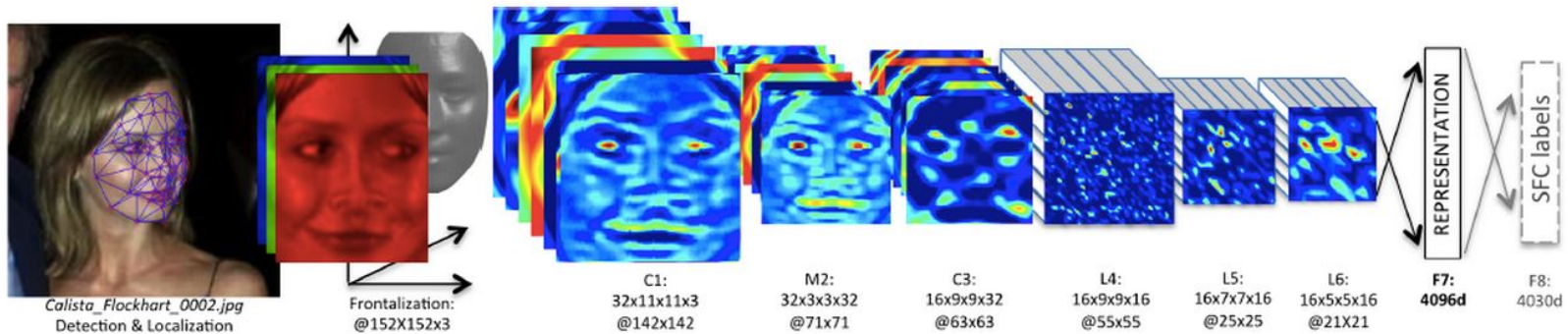
[72 votes](#) • Poll closed

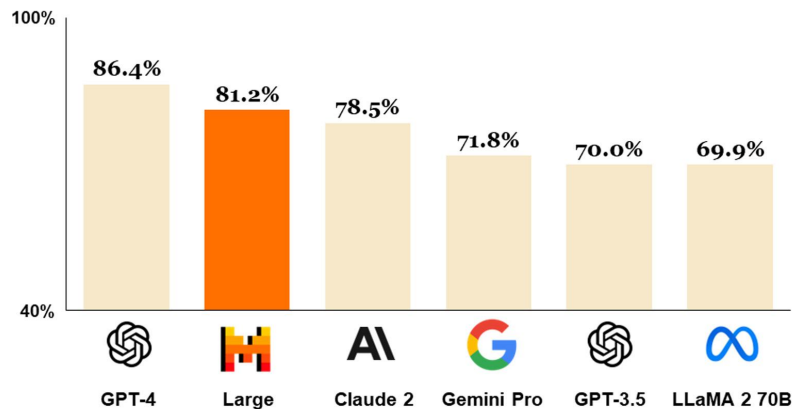
Why deep learning

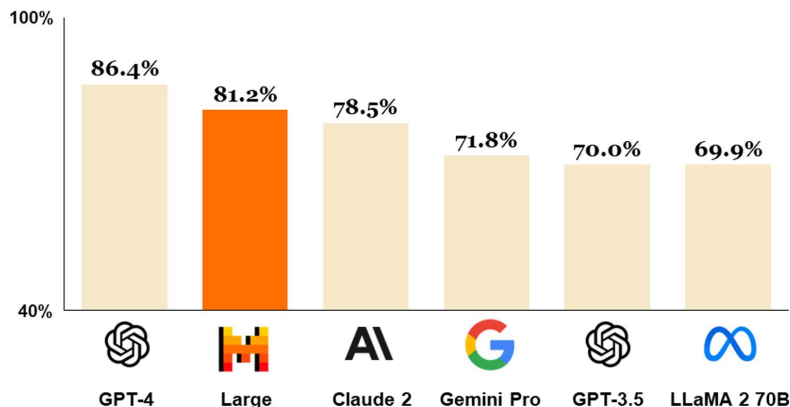


Source: Andrew Ng









ChatGPT 4o



What are the top 10 towns to visit in Tuscany, Italy?



Tuscany, Italy, is renowned for its picturesque landscapes, rich history, and cultural heritage. Here are the top 10 towns to visit in Tuscany:

1. **Florence (Firenze):** The capital of Tuscany, known for its Renaissance art and architecture. Must-see attractions include the Uffizi Gallery, Florence Cathedral, and the Ponte Vecchio.
2. **Siena:** Famous for its medieval cityscape, the Palio horse race, and the stunning Piazza del Campo. The Siena Cathedral is a masterpiece of Gothic architecture.
3. **Pisa:** Home to the iconic Leaning Tower of Pisa, as well as the beautiful Piazza dei Miracoli which includes the Cathedral and the Baptistry.
4. **Lucca:** Known for its well-preserved Renaissance walls that encircle the historic city center. It offers charming streets, beautiful churches, and the Guinigi Tower.
5. **San Gimignano:** Famous for its medieval towers and well-preserved city



WIKIPEDIA
The Free Encyclopedia



WIKIPEDIA
The Free Encyclopedia

Are there any known common issues with the MacBook Air M4 13 inch?

Based on available information, the MacBook Air M4 13-inch (2025) is a highly praised ultraportable laptop, but there are a few reported issues and limitations that could be considered common or notable. These are drawn from reviews, user feedback, and technical analyses. Below is a summary of potential issues, with critical examination of the sources to ensure accuracy:

1. Thermal Throttling Due to Fanless Design

- **Issue:** The MacBook Air M4 lacks an active cooling fan, which can lead to thermal throttling during sustained, intensive workloads (e.g., video encoding, 3D rendering). The 13-inch model may throttle more aggressively than the 15-inch version due to a smaller heatsink. In tests like Handbrake video encoding, performance slows compared to actively cooled M4 devices (e.g., MacBook Pro). 
- **Impact:** For most everyday tasks (web browsing, document editing, light photo editing), this isn't a problem. However, users running heavy CPU/GPU workloads for extended periods may notice reduced performance.
- **Critical Note:** This is a design trade-off for a silent, fanless system. It's not a defect but a limitation inherent to the Air's ultraportable ethos. Users needing sustained high performance might prefer the MacBook Pro.

Write a space shooter game in Python using the pygame framework.



Sure! Let's create a basic space shooter game using the Pygame framework in Python. This game will include a player-controlled spaceship that can move left and right and shoot bullets at incoming enemies.

First, make sure you have Pygame installed. You can install it using pip:

bash

Copy code

```
pip install pygame
```

Here's a complete code for a basic space shooter game:

python

Copy code

```
import pygame
import random

# Initialize Pygame
pygame.init()

# Screen dimensions
WIDTH, HEIGHT = 800, 600
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Space Shooter")

# Colors
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
```

Write a space shooter game in Python using the pygame framework.



Sure! Let's create a basic space shooter game using the Pygame framework in Python. This game will include a player-controlled spaceship that can move left and right and shoot bullets at incoming enemies.

First, make sure you have Pygame installed. You can install it using pip:

bash

Copy code

```
pip install pygame
```

Here's a complete code for a basic space shooter game:

python

Copy code

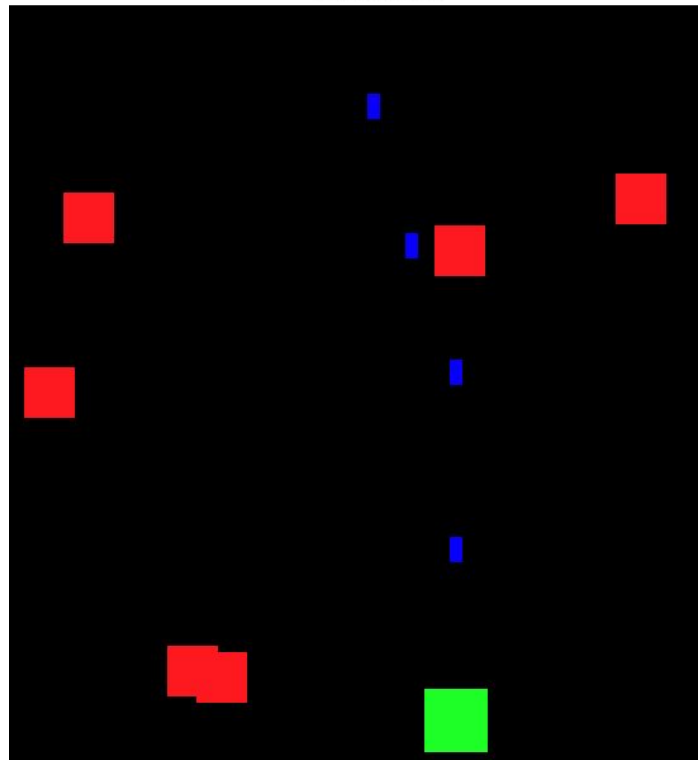
```
import pygame
import random

# Initialize Pygame
pygame.init()

# Screen dimensions
WIDTH, HEIGHT = 800, 600
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Space Shooter")

# Colors
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
```

Space Shooter





Data Analyst

By ChatGPT 

Drop in any files and I can help analyze and visualize your data.



Download Data - INDEX_US_S&P U...
Spreadsheet



Message Data Analyst





Data Analyst

By ChatGPT

Drop in any files and I can help analyze and visualize your data.



Download Data - INDEX_US_S&P U...
Spreadsheet



Message Data Analyst



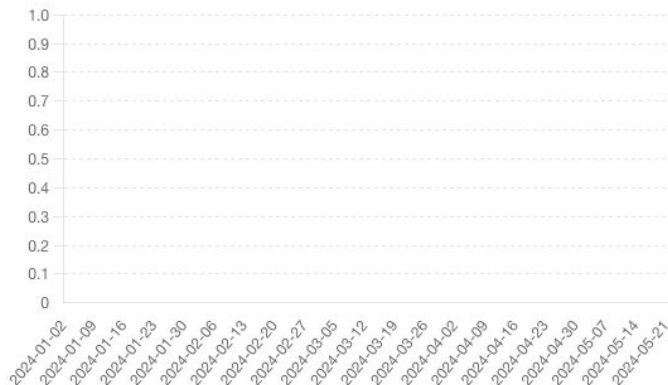
Plot the closing price vs time.



S&P 500 Closing Price Over Time



☒ Closing Price by ☒ Date



Here's the plot of the S&P 500 closing prices over time. If you need any further analysis or specific insights from this data, feel free to let me know! [\[?\]](#)

There is nothing on the plot.



It seems there might be an issue with the data rendering on the plot. I'll recheck the data and regenerate the plot.

The ``Open``, ``High``, ``Low``, and ``Close`` columns are currently of type ``object``. They need to be converted to numeric types for proper plotting. I'll convert these columns and regenerate the plot. [\[↵\]](#)



There is nothing on the plot.

It seems there might be an issue with the data rendering on the plot. I'll recheck the data and regenerate the plot.

The `'Open'`, `'High'`, `'Low'`, and `'Close'` columns are currently of type `'object'`. They need to be converted to numeric types for proper plotting. I'll convert these columns and regenerate the plot. [↗]



Analysis

python

Always show details

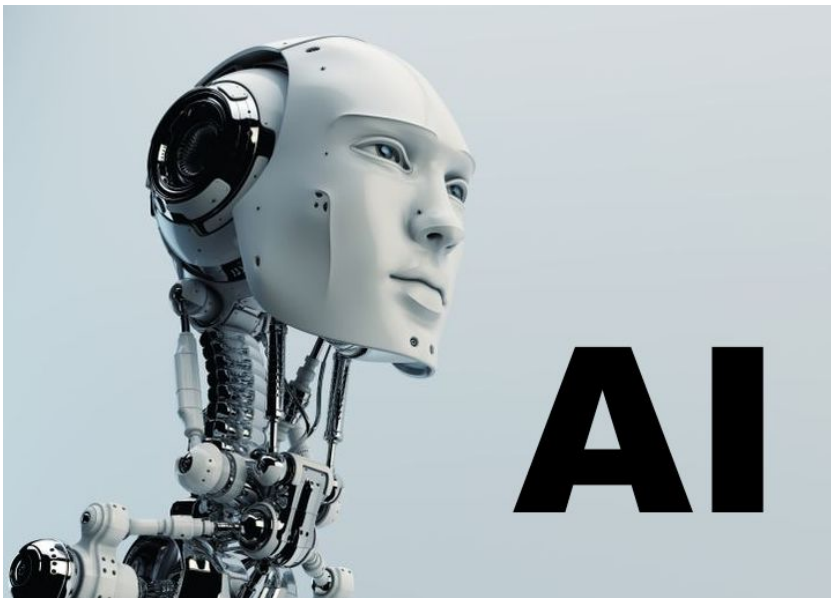
Copy code

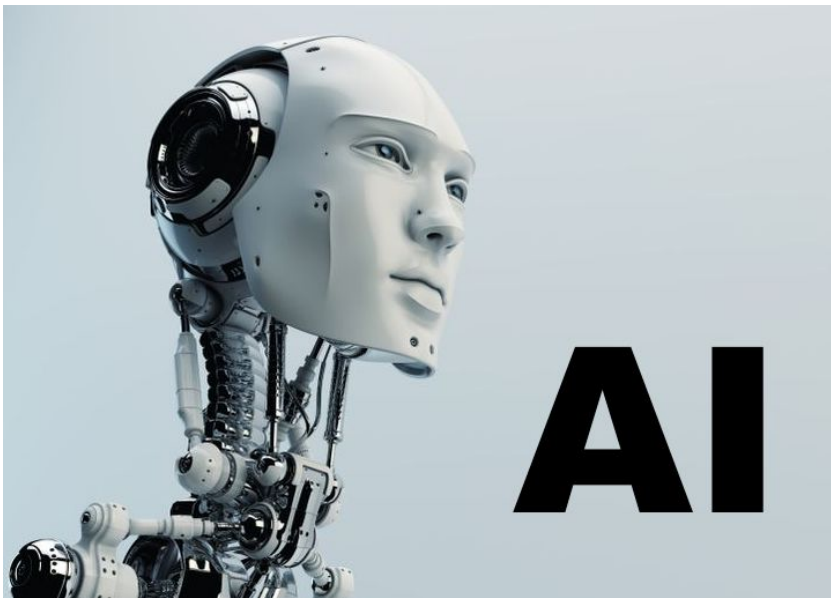
```
# Convert columns to numeric types
data['Open'] = pd.to_numeric(data['Open'].str.replace(',', ''))
data['High'] = pd.to_numeric(data['High'].str.replace(',', ''))
data['Low'] = pd.to_numeric(data['Low'].str.replace(',', ''))
data['Close'] = pd.to_numeric(data['Close'].str.replace(',', ''))

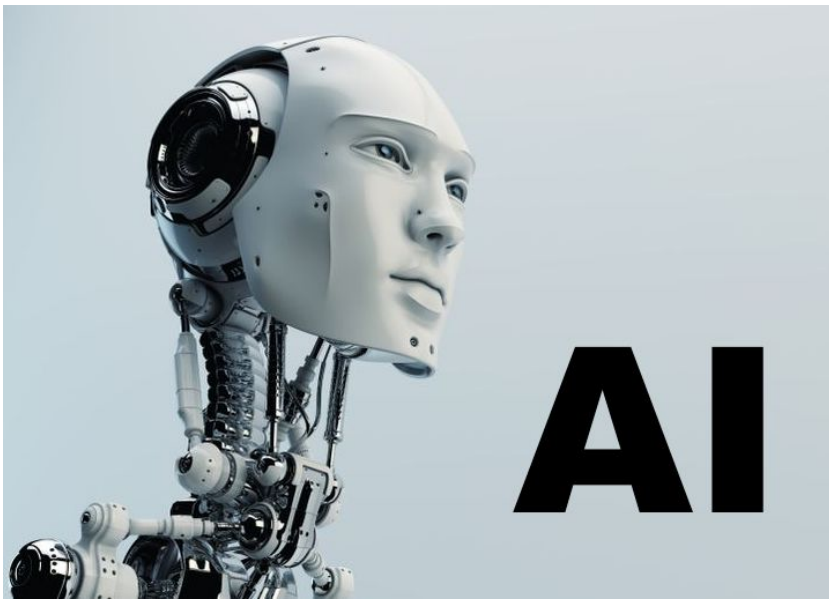
# Plot the closing price vs time again
plt.figure(figsize=(10, 6))
plt.plot(data['Date'], data['Close'], label='Closing Price')
plt.xlabel('Date')
plt.ylabel('Closing Price')
plt.title('S&P 500 Closing Price Over Time')
plt.legend()
plt.grid(True)
plt.show()
```











“a mad race to who knows what.” - Joe Rogan



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)

hidden = c(20)



szilard commented Nov 27, 2015

Trying to see if DL can match RF/GBM in accuracy on the airline dataset (wh years 2005-2006, while validation and test sets sampled disjunctly from 20 kept categorical artificially and are intentionally not encoded as ordinal varia structure of business datasets).

RectifierWithDropout, c(200,200,200)

ADADELTA rho = 0.95, epsilon = 1e-6

rho = 0.999, epsilon = 1e-08

adaptive = FALSE default: rate = 0.005, decay = 1, momentum = 0

rate = 0.001, momentum = 0.5 / 1e5 / 0.99

rate = 0.01, momentum = 0.5 / 1e5 / 0.99

rate = 0.01, rate_annealing = 1e-05, momentum = 0.5 / 1e5 / 0.99

rate = 0.01, rate_annealing = 1e-04, momentum = 0.5 / 1e5 / 0.99

rate = 0.01, rate_annealing = 1e-05, momentum = 0.5 / 1e5 / 0.9

73.3 270 1.9

73.0 340 1.1

73.2 410 0.7

73.3 280 0.9

73.5 360 1

72.7 3700 8.7

73.4 350 0.9

DL with h2o #28



Closed

szilard opened this issue on Nov 27, 2015 · 14 comments



szilard commented on Nov 27, 2015

Owner

+ 😊 ...

Trying to see if DL can match RF/GBM in accuracy on the airline dataset (where train is sampled from years 2005-2006, while validation and test sets sampled disjunctly from 2007). Also, some variables are kept categorical artificially and are intentionally not encoded as ordinal variables (to better match the structure of business datasets).



arnocandel commented on Nov 29, 2015

+ 😊 ...

Yes, after a bit of tinkering, I also cannot get higher than 0.735 test set AUC. On my i7 5820k home PC:

```
system.time({  
md <- h2o.deeplearning(x = Xnames, y = "dep_delayed_15min", training_frame = dx_train,
```

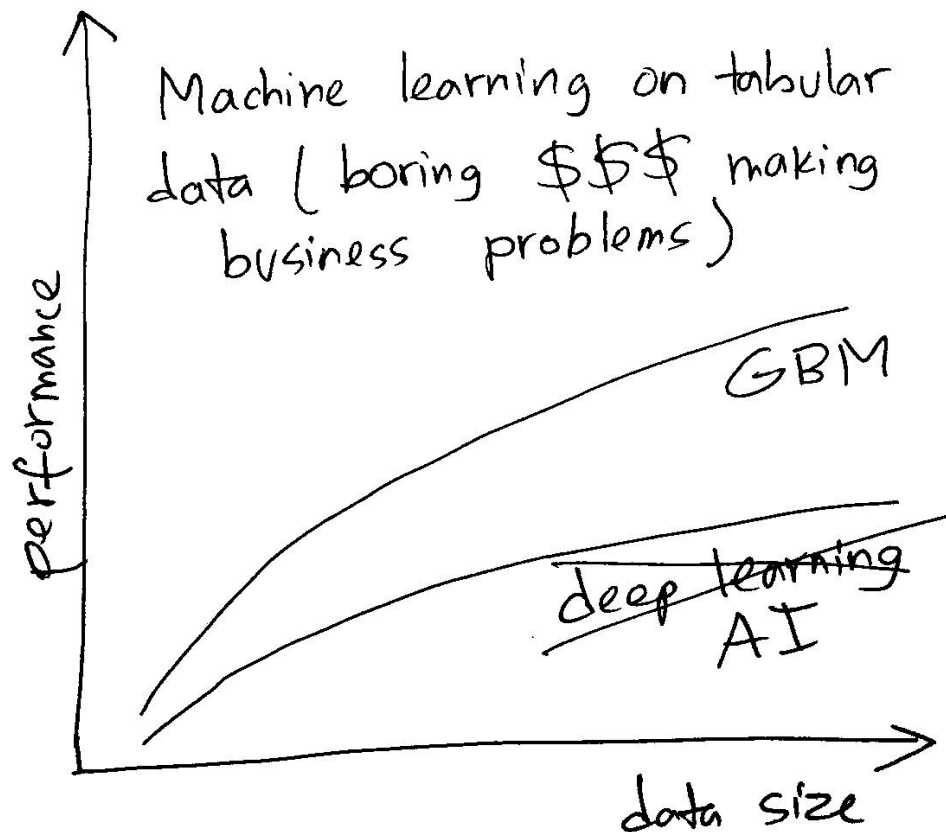
some feature engineering (e.g., cutting the original DepTime into 48 categorical half-hour slots). Out of 675 input neurons, only 2 are always populated with non-zero values (the two numeric features), and 673 values are mostly 0, only 6 categoricals are set to 1. That's where the inefficiency comes from. GBM/DRF are much more efficient at simply cutting up the feature space, which is was seems to be needed here.

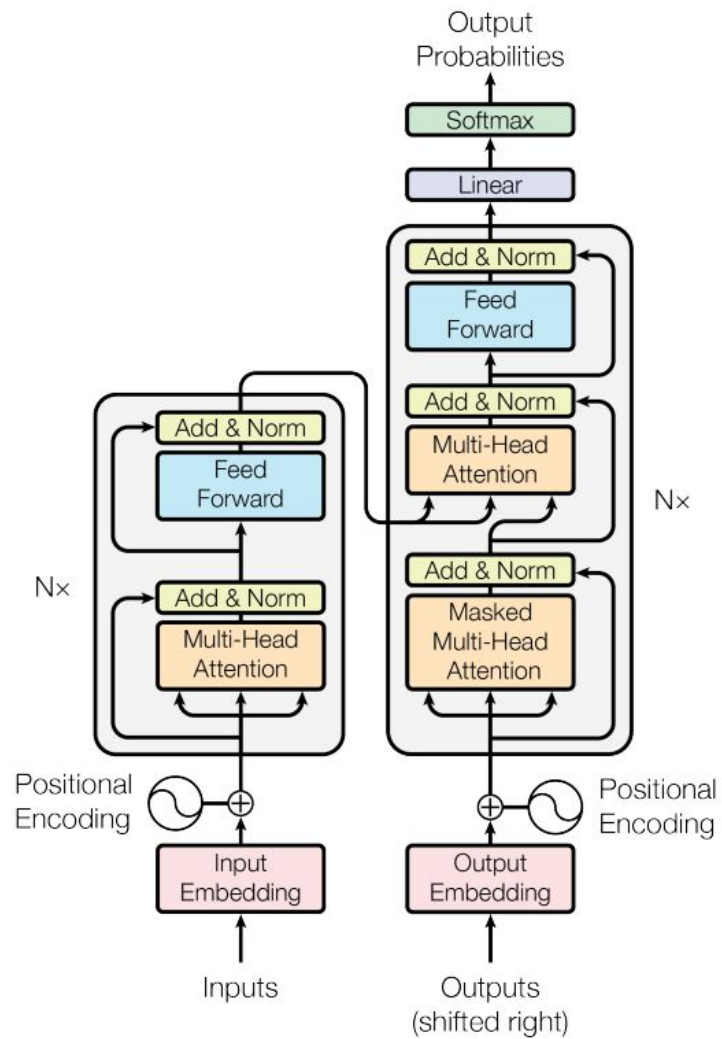
Best,
Arno



Szilard [Deeper than Deep Learning] @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)





MLP [188]

DeepFM [14]

DeepGBM [52]

RLN [54]

TabNet [5]

VIME [67]

TabTransformer [99]

NODE [6]

DNFNet [43]

STG [189]

NAM [190]

SAINT [9]

MLP [188]

DeepFM [14]

DeepGBM [52]

RLN [54]

TabNet [5]

VIME [67]

TabTransformer [99]

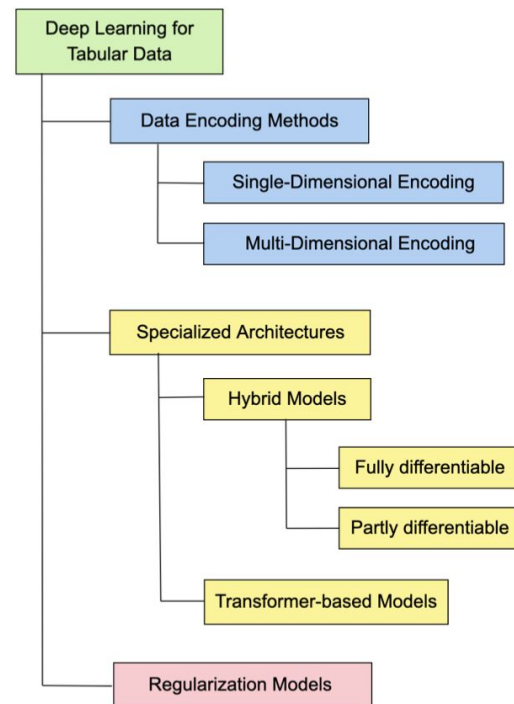
NODE [6]

DNFNet [43]

STG [189]

NAM [190]

SAINT [9]



MLP [188]

DeepFM [14]

DeepGBM [52]

RLN [54]

TabNet [5]

VIME [67]

TabTransformer [99]

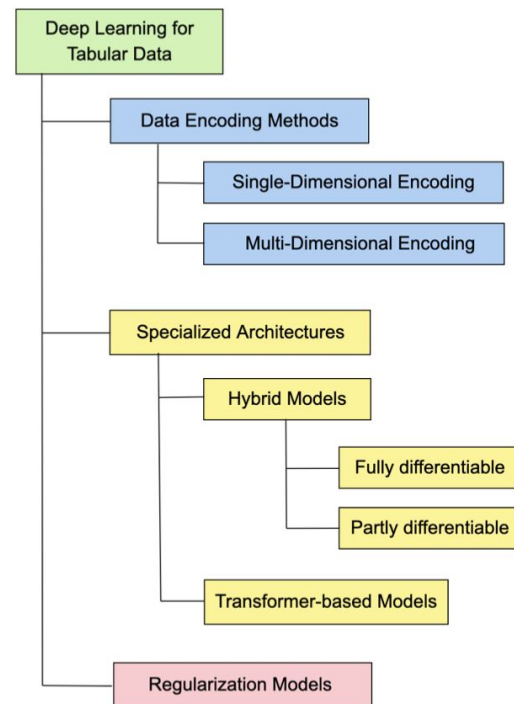
NODE [6]

DNFNet [43]

STG [189]

NAM [190]

SAINT [9]

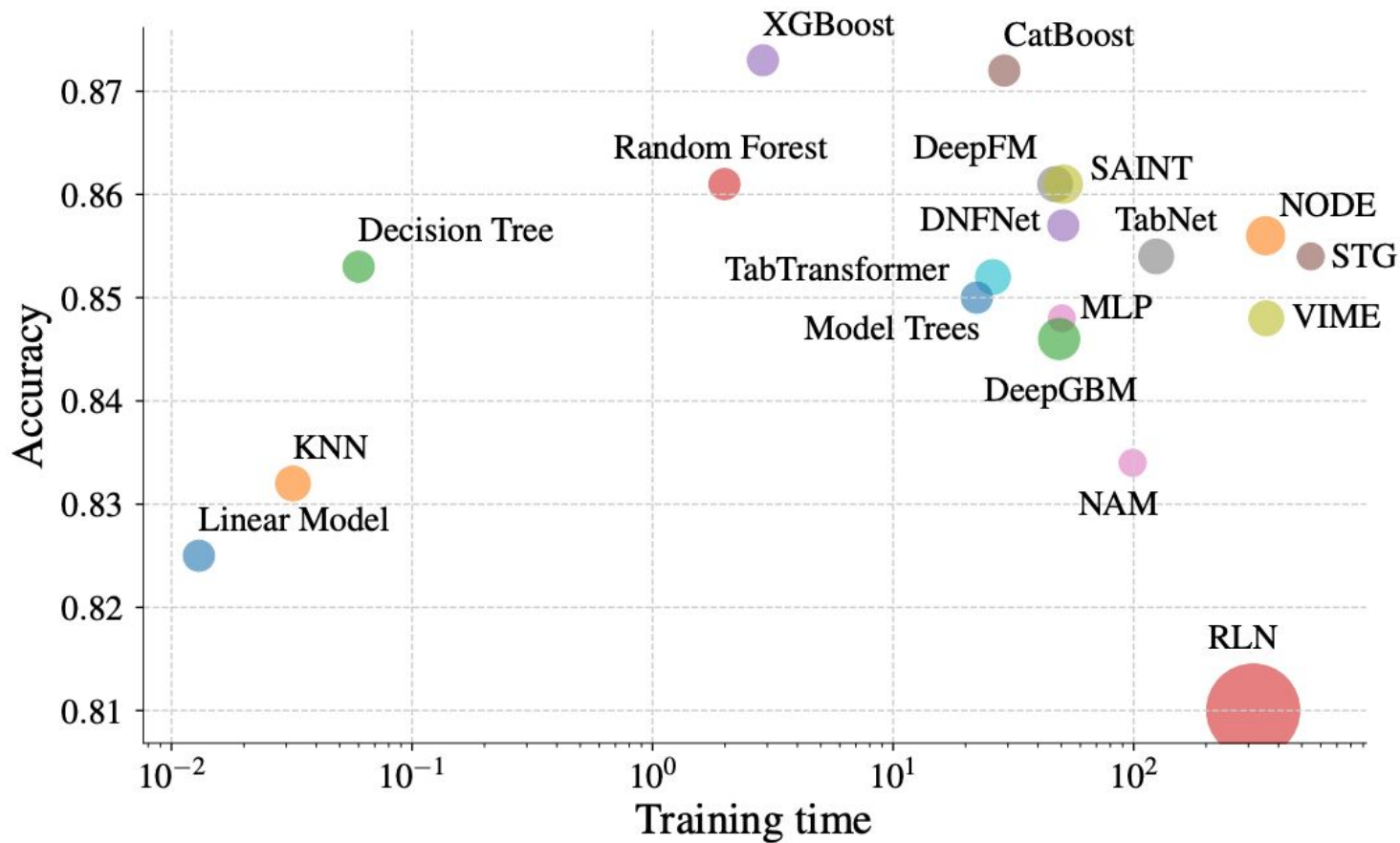


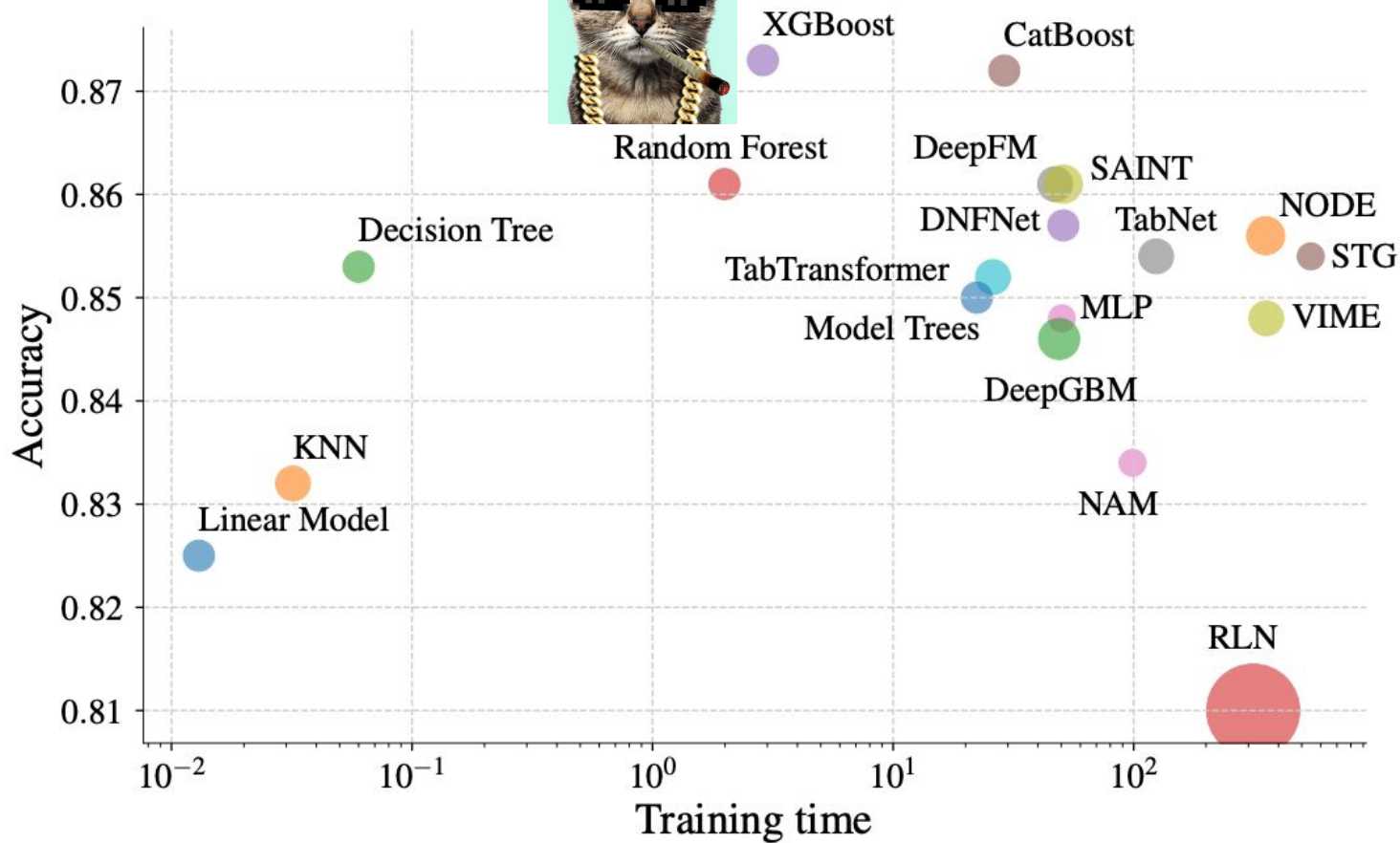
SUBMITTED TO THE IEEE, FEBRUARY 2022

1

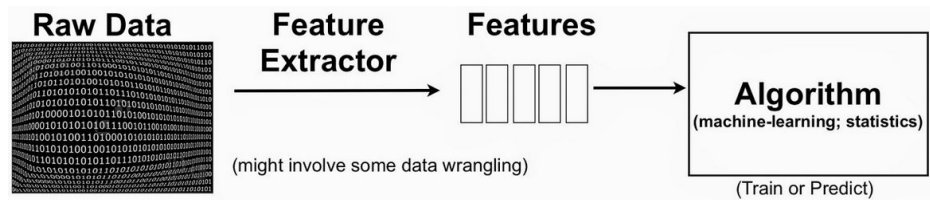
Deep Neural Networks and Tabular Data: A Survey

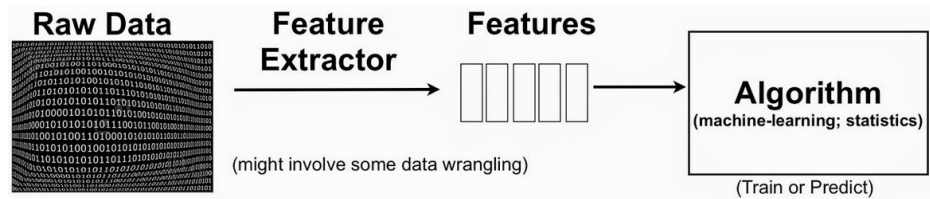
Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug,
Martin Pawelczyk and Gjergji Kasneci





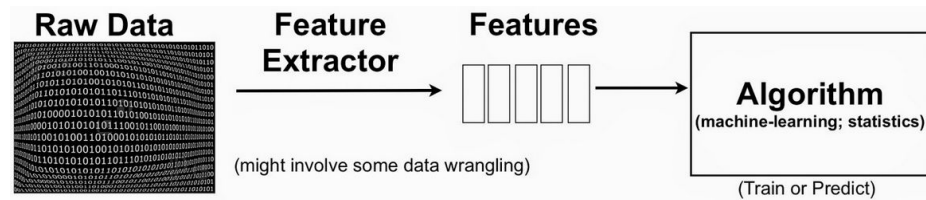






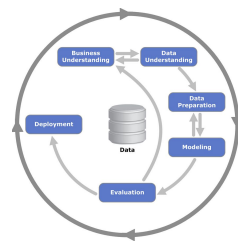
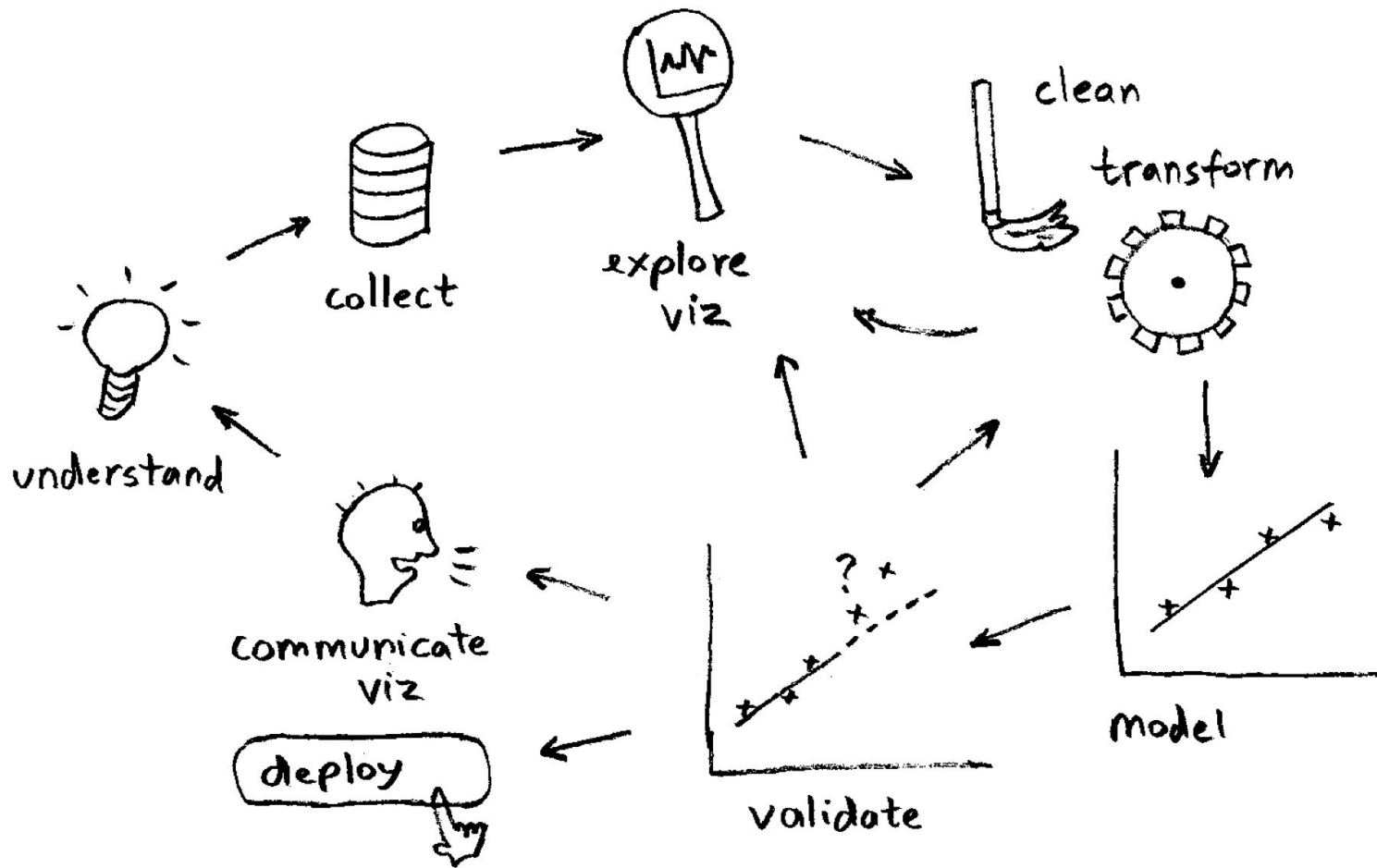
Hyperparameter
tuning





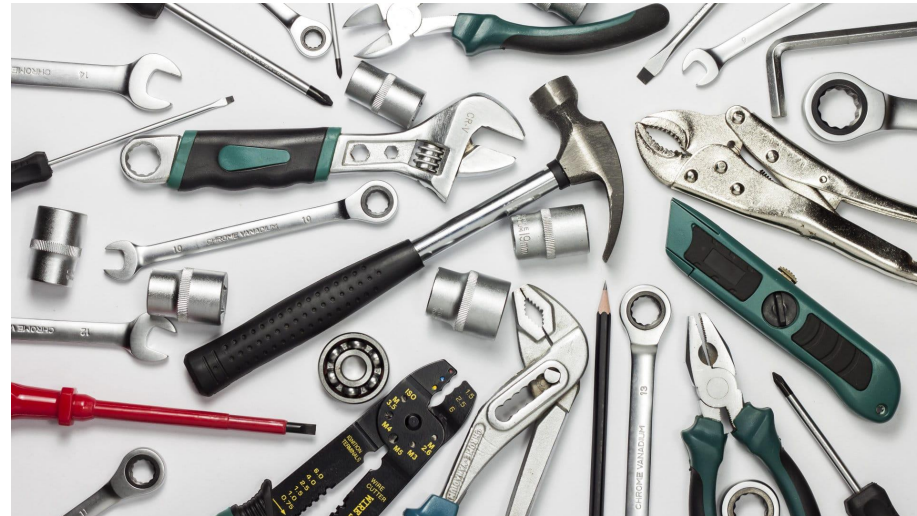
Hyperparameter
tuning

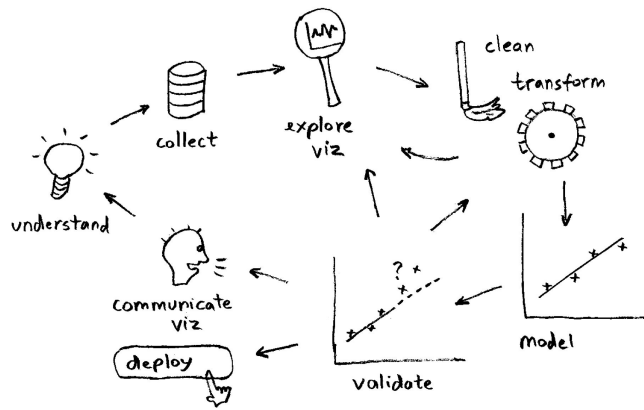










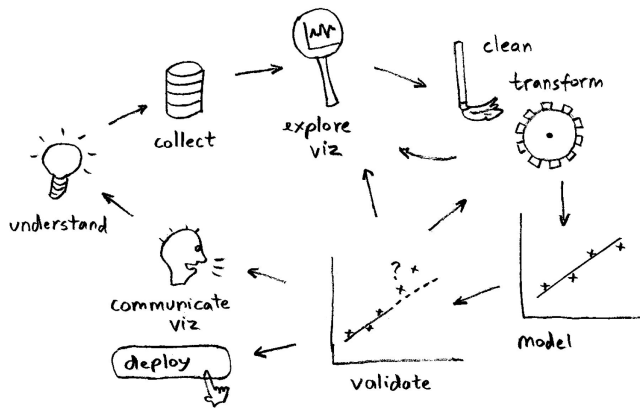


Data Analyst

By ChatGPT



Drop in any files and I can help analyze and visualize your data.



Data Analyst

By ChatGPT

Drop in any files and I can help analyze and visualize your data.

What are the top 3 algorithms for predictive analytics/machine learning with tabular data?



The top three algorithms for predictive analytics and machine learning with tabular data are:

1. Gradient Boosting Machines (GBM)

- **Description:** GBM is an ensemble learning technique that builds models sequentially, where each new model attempts to correct the errors of the previous ones. Popular implementations include XGBoost, LightGBM, and CatBoost.
- **Use Cases:** GBM is highly effective for regression and classification tasks. It's widely used in competitions and practical applications due to its high accuracy and ability to handle a variety of data types and structures.

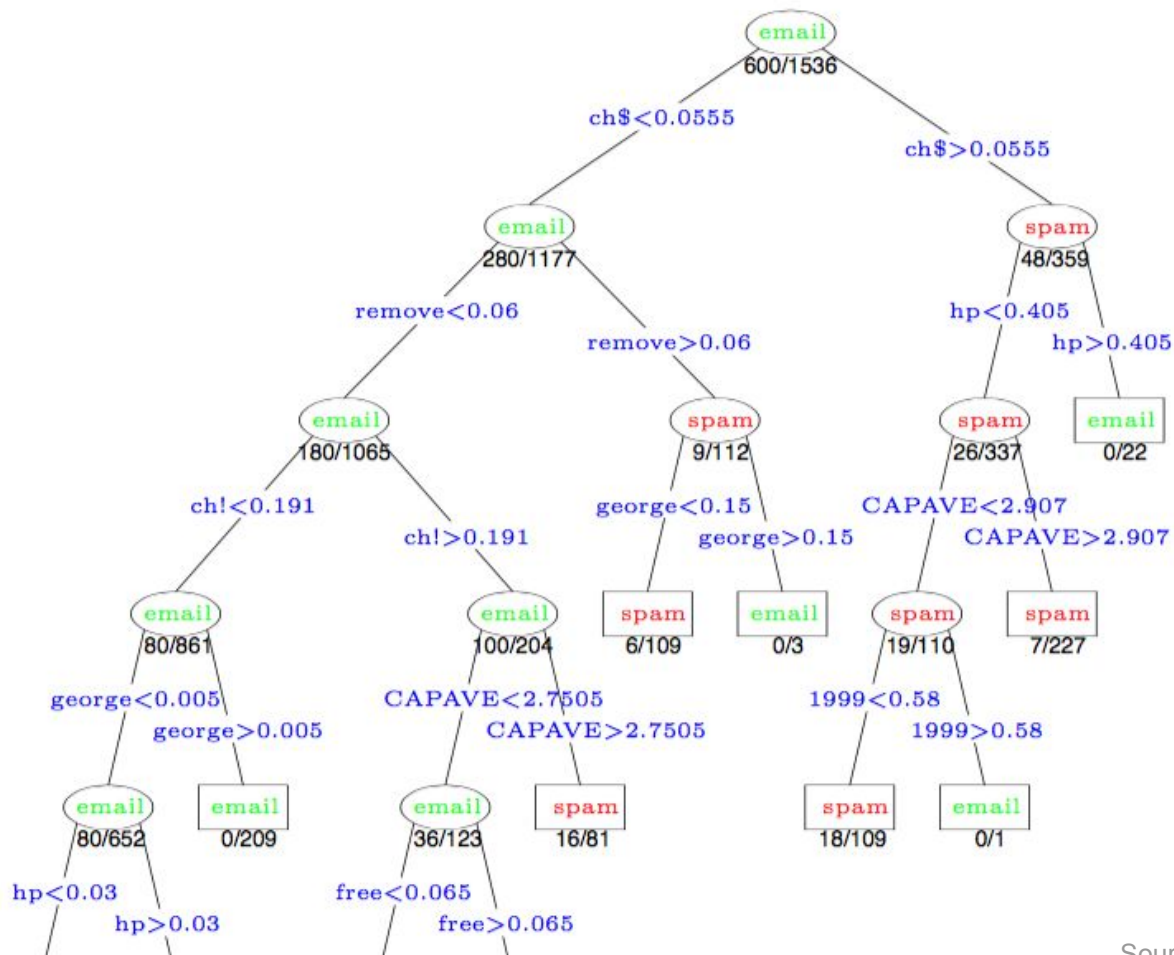
2. Random Forest

- **Description:** Random Forest is an ensemble method that builds multiple decision trees and merges their results to improve accuracy and control over-fitting. Each tree is trained on a random subset of the data.
- **Use Cases:** Random Forest is robust for both classification and regression tasks. It's preferred for its simplicity and effectiveness, especially when dealing with large datasets and features.

3. Logistic Regression

GBM?

TELL ME MORE



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



open source

- R packages
- Python scikit-learn
- Vowpal Wabbit
- H2O
- xgboost
- Spark MLlib
- a few others



open source

- R packages
- Python scikit-learn
- Vowpal Wabbit
- H2O
- xgboost
- Spark MLlib
- a few others

100% FREE



open source

- R packages
- Python scikit-learn
- Vowpal Wabbit
- H2O
- xgboost
- Spark MLlib
- a few others



100% FREE





open source

- R packages
- Python scikit-learn
- Vowpal Wabbit
- H2O
- xgboost
- Spark MLlib
- a few others



100% FREE



stackoverflow





open source

- R packages
- Python scikit-learn
- Vowpal Wabbit
- H2O
- xgboost
- Spark MLlib
- a few others



100% FREE





open source

- R packages
- Python scikit-learn
- Vowpal Wabbit
- H2O
- xgboost
- Spark MLlib
- a few others



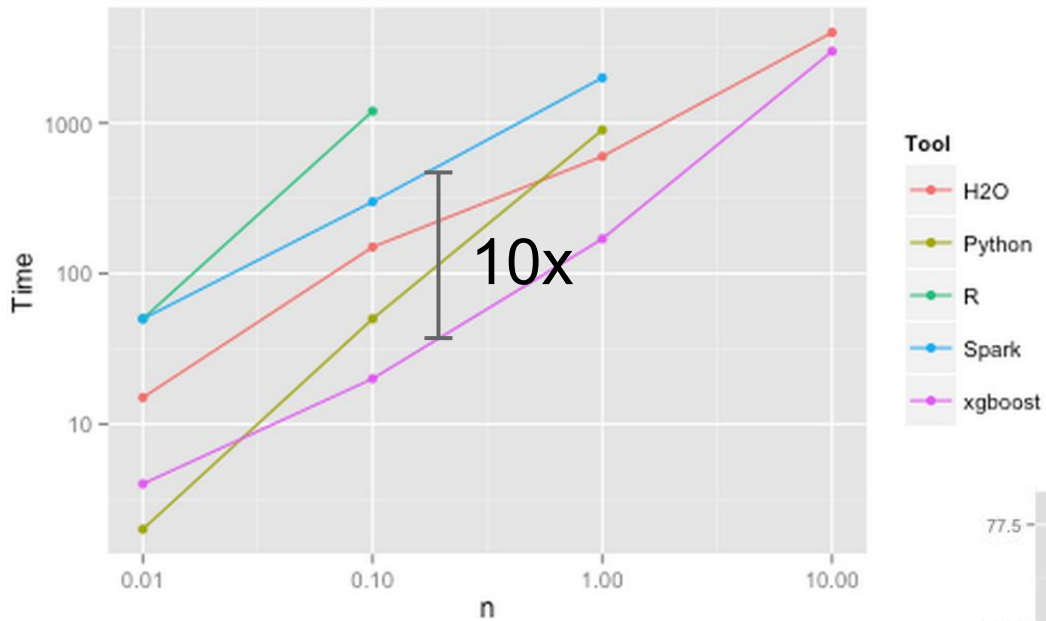
szilard / **benchm-ml**

★ Star

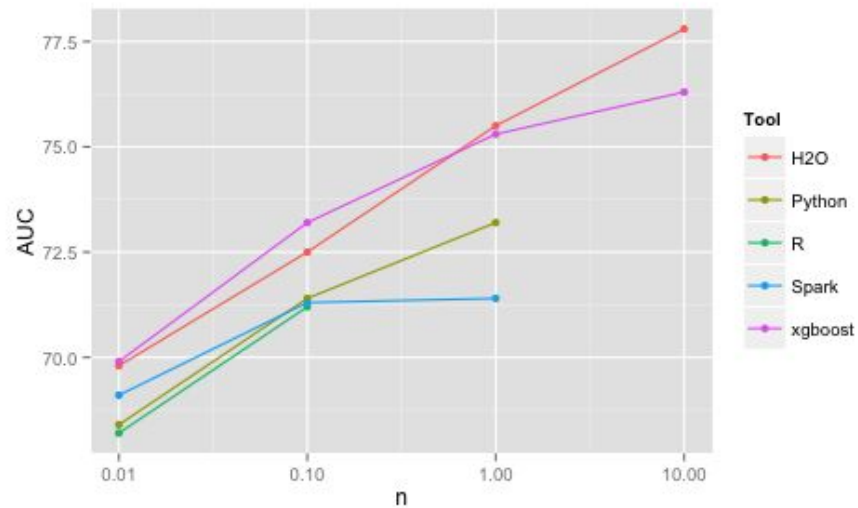
1,203

Simple/limited/incomplete benchmark

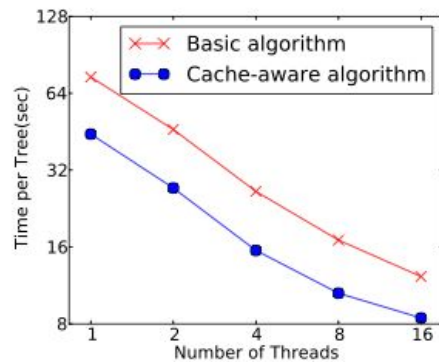
(2015-)



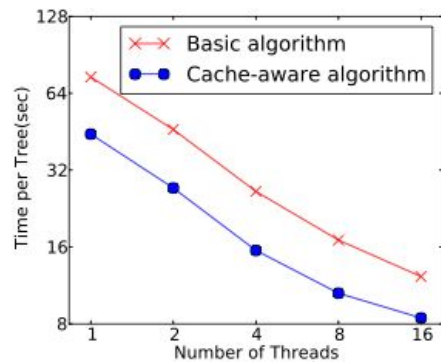
 szilard / benchm-ml



XGBoost: A Scalable Tree Boosting System

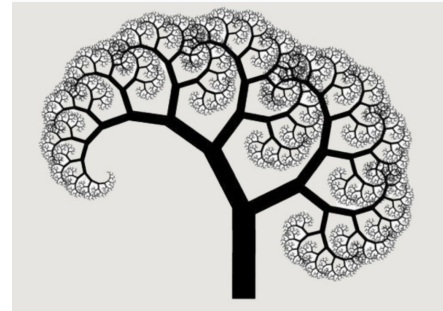
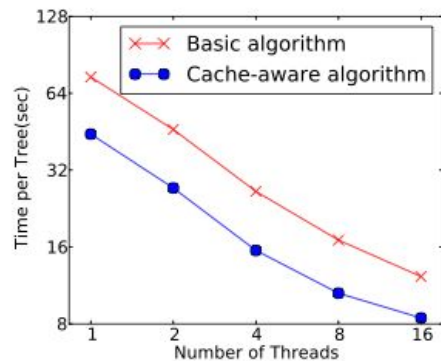


XGBoost: A Scalable Tree Boosting System



2015

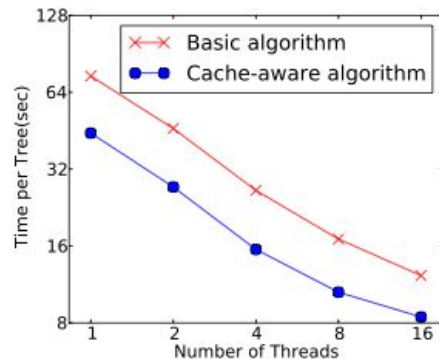
XGBoost: A Scalable Tree Boosting System



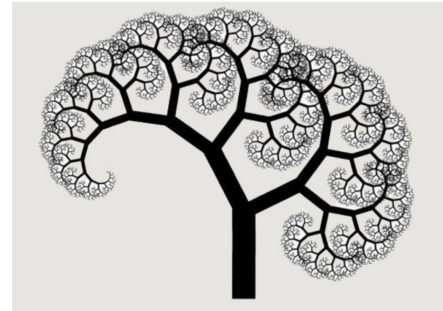
CatBoost

2017

XGBoost: A Scalable Tree Boosting System



Microsoft / LightGBM



CatBoost



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

xgboost: Extreme Gradient Boosting

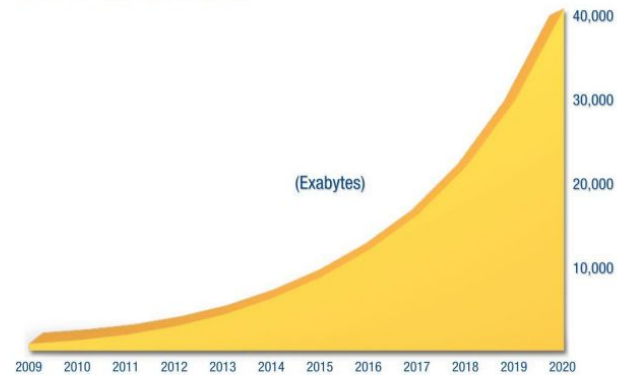
<https://cran.r-project.org/web/packages/h2o/index.html>

h2o: R Interface for H2O



Figure 1

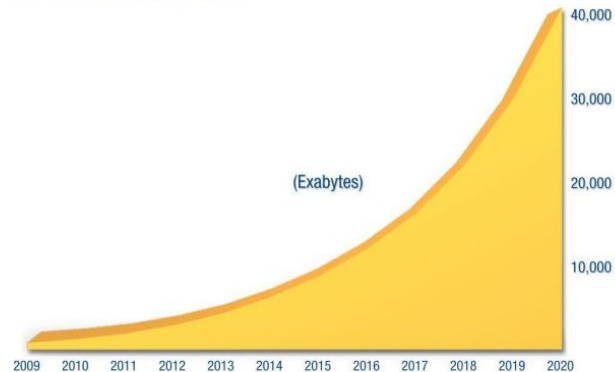
The Digital Universe: 50-fold Growth from the Beginning of 2010 to the End of 2020



Source: IDC's Digital Universe Study, sponsored by EMC, December 2012

Figure 1

The Digital Universe: 50-fold Growth from the Beginning of 2010 to the End of 2020



Source: IDC's Digital Universe Study, sponsored by EMC, December 2012



Hadley Wickham

@hadleywickham

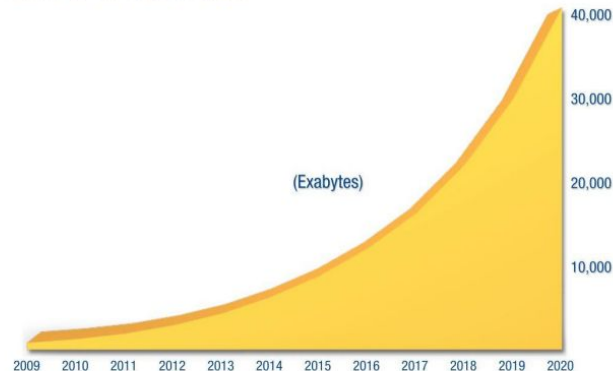


Following

"It takes a big man to admit his data is small" —
[@jcheng](#)

Figure 1

The Digital Universe: 50-fold Growth from the Beginning of 2010 to the End of 2020



Source: IDC's Digital Universe Study, sponsored by EMC, December 2012



Hadley Wickham

@hadleywickham



Following

"It takes a big man to admit his data is small" —
[@jcheng](#)

TYPICAL SIZE OF DATASETS

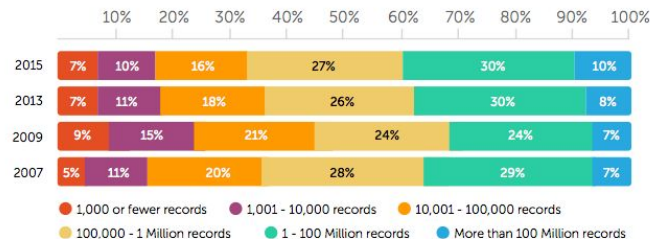
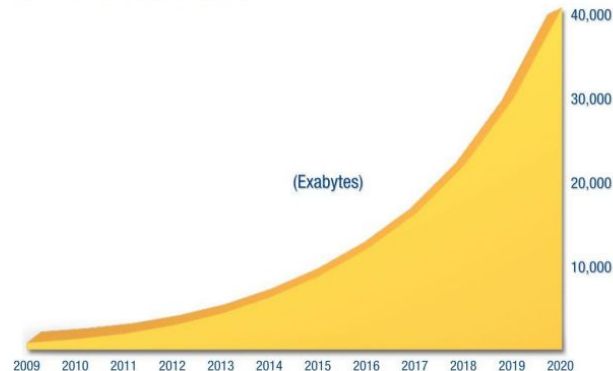


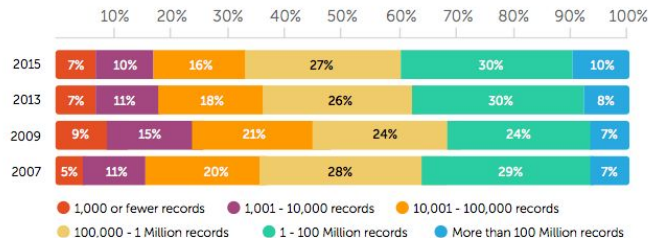
Figure 1

The Digital Universe: 50-fold Growth from the Beginning of 2010 to the End of 2020



Source: IDC's Digital Universe Study, sponsored by EMC, December 2012

TYPICAL SIZE OF DATASETS



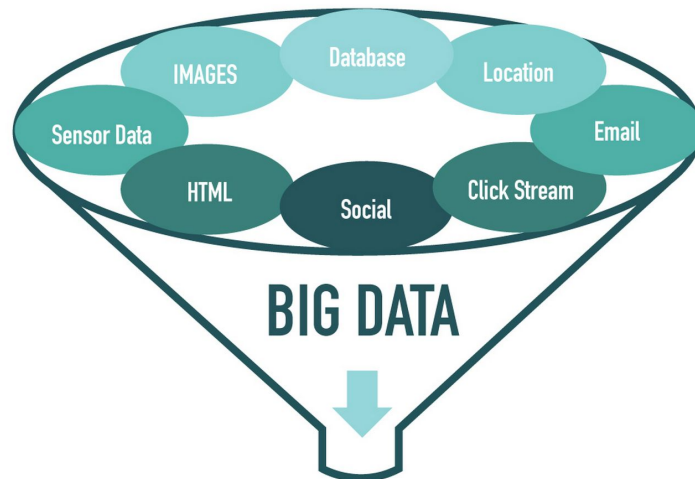
Hadley Wickham

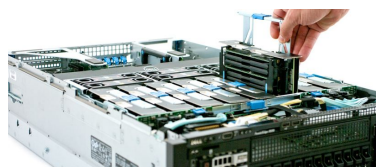
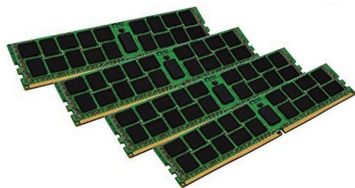
@hadleywickham



Following

"It takes a big man to admit his data is small" —
@jcheng





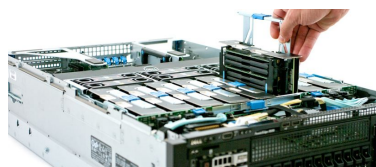
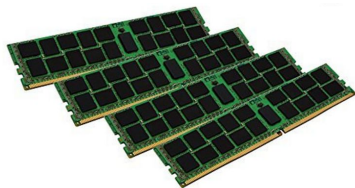
Kingston Technology Value RAM 128GB Kit (4x32GB) 2133MHz DDR4 ECC Reg CL15 (KVR21R15D4K4/128)

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](#)

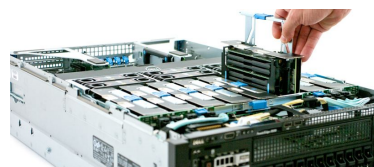
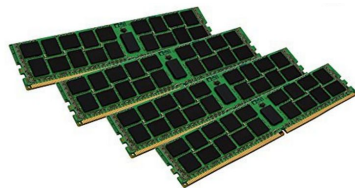
[Be the first to review this item](#)

Was: ~~\$743.99~~

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



Model	vCPU	Mem (GiB)	
r3.8xlarge	32	244	(2015)
x1e.32xlarge	128	3,904	
u-12tb1.metal	448	12	(TiB)



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](#)

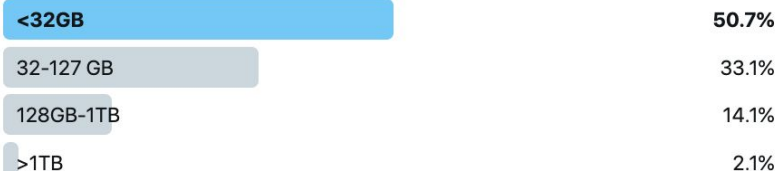


Model	vCPU	Mem (GiB)	
r3.8xlarge	32	244	(2015)
x1e.32xlarge	128	3,904	
u-12tb1.metal	448	12	(TiB)



Szilard [Deeper than Deep Learning]
[@DataScienceLA](#)

How much RAM do you have on the server/desktop/laptop you are most commonly using to train machine learning models?



142 votes · Final results



Szilard @DataScienceLA · Aug 3

I wish my [#machinelearning](#) worked... ("both" is not a choice 😊) [#bigdata](#)
[#datascience](#) [#rstats](#) [#pydata](#) cc @h2o @databricks @cloudera @kaggle

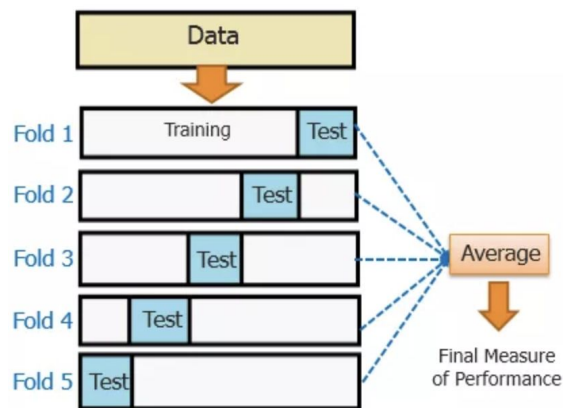
10% on 10x bigger data

70% 10x faster

20% I don't care about either

104 votes • Final results





Hyperparameter
tuning

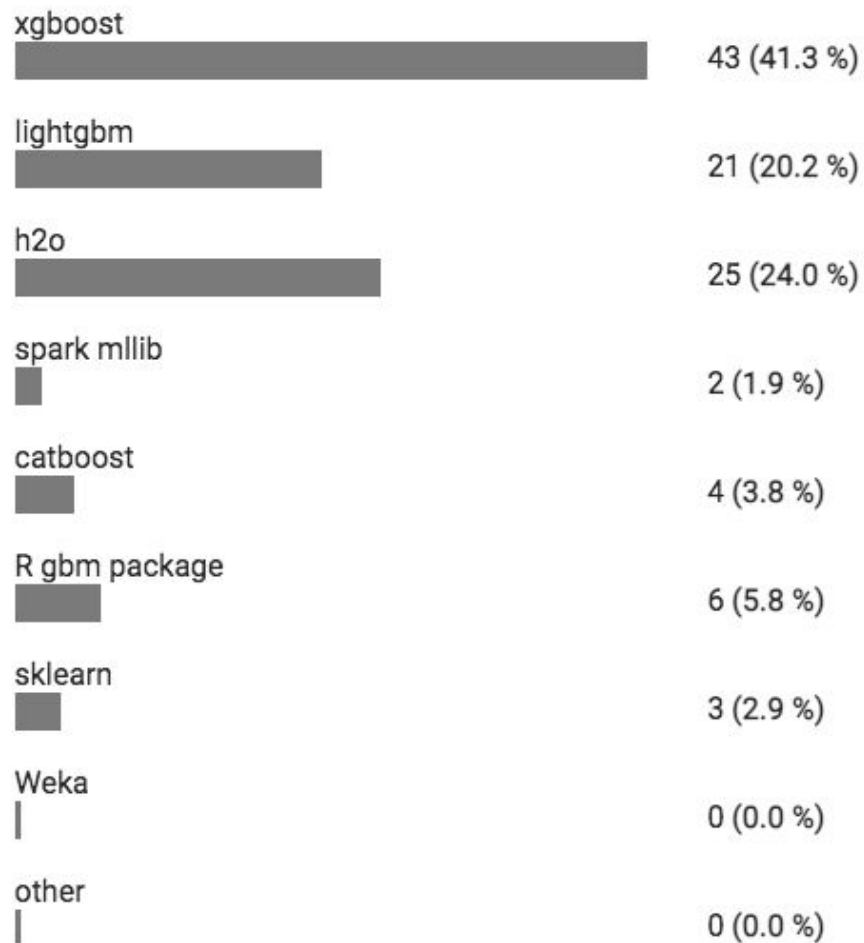




szilard / GBM-perf

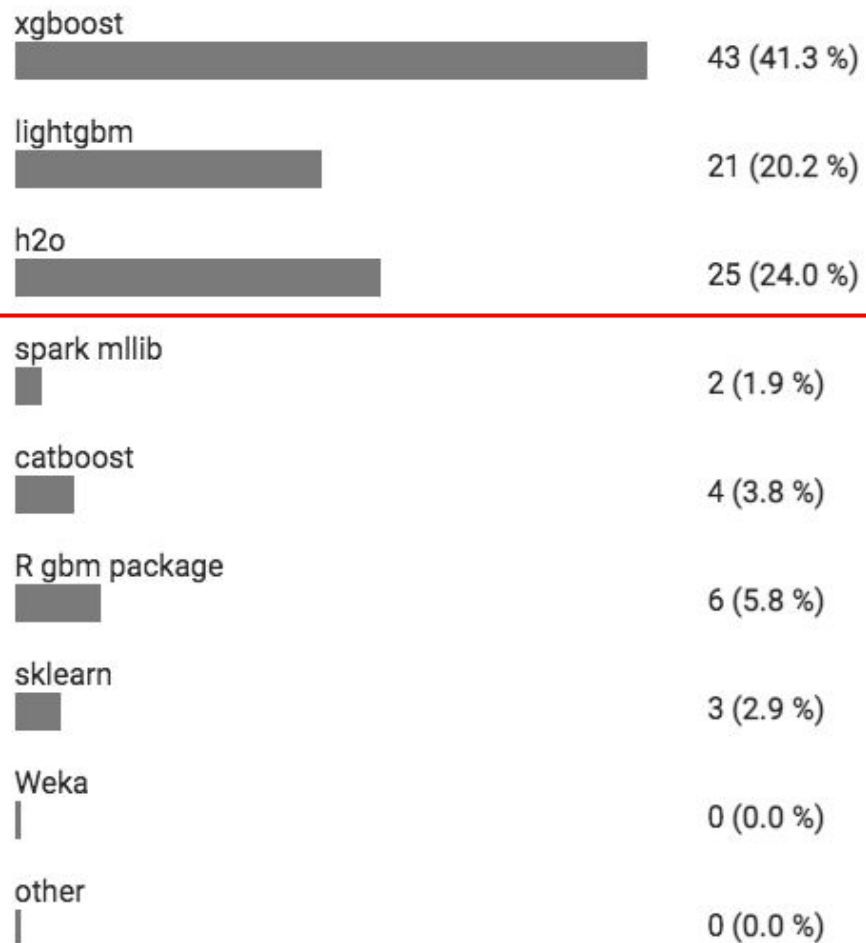
(2017-)

```
git clone https://github.com/szilard/GBM-perf.git
cd GBM-perf/cpu
sudo docker build -t gbmp perf_cpu .
sudo docker run --rm gbmp perf_cpu
```



104 people have already voted
You have already answered this poll

[i show the configuration of this Ferendum](#)



104 people have already voted
You have already answered this poll

[show the configuration of this Ferendum](#)



Szilard [Deeper than Deep Learning]

@DataScienceLA



What gradient boosting machine(GBM) library have you been using the most in 2020? (4 options, for others please reply to tweet)

xgboost

53.5%

lightgbm

26.7%

h2o

10.9%

catboost

8.9%

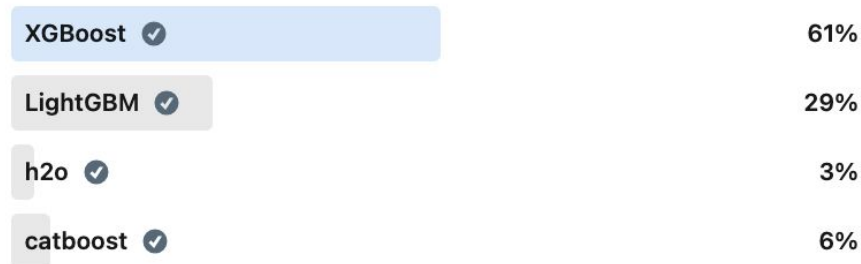
570 votes · Final results

10:59 AM · Sep 9, 2020 · Twitter Web App

 **Szilard Pafka** • 1st
physics PhD, chief (data) scientist, meetup organizer, (visiting) professor, m...
1w • 

2024 update: What gradient boosting machine (GBM) library have you been using the most this year?

The author can see how you vote. [Learn more](#)

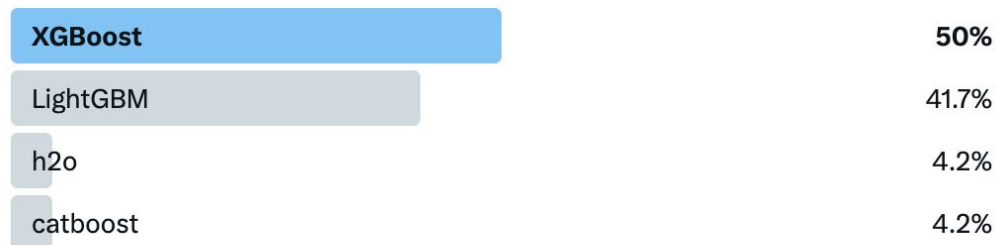


31 votes • Poll closed

 **Szilard Pafka**
@SzilardPafka

...

2024 update: What gradient boosting machine (GBM) library have you been using the most this year?



24 votes • Final results

r4.8xlarge (32 cores, but run on physical cores only/no hyperthreading) with software as of 2021-01-14:

Tool	Time[s] 100K	Time[s] 1M	Time[s] 10M	AUC 1M	AUC 10M
h2o	12	15	90	0.762	0.776
xgboost	0.6	3.5	40	0.748	0.754
lightgbm	2.6	4.2	20	0.765	0.792
catboost	3.8	10	80	0.734	0.735

r4.8xlarge (32 cores, but run on physical cores only/no hyperthreading) with software as of 2021-01-14:

Tool	Time[s] 100K	Time[s] 1M	Time[s] 10M	AUC 1M	AUC 10M
h2o	12	15	90	0.762	0.776
xgboost	0.6	3.5	40	0.748	0.754
lightgbm	2.6	4.2	20	0.765	0.792
catboost	3.8	10	80	0.734	0.735

p3.2xlarge (1 GPU, Tesla V100) with software as of 2021-01-15 (and CUDA 11.0):



Tool	Time[s] 100K	Time[s] 1M	Time[s] 10M	AUC 1M	AUC 10M
h2o xgboost	6.4	14	45	0.749	0.756
xgboost	3.6	6.5	11	0.748	0.756
lightgbm	7	10	42	0.767	0.792
catboost	1.8	4.6	37	0.732 ?!	0.736 ?!

100M records and RAM usage

CPU (m5.12xlarge):

Tool	time [s]	AUC	RAM train [GB]
h2o	520	0.775	8
xgboost	510	0.751	15
lightgbm	310	0.774	5
catboost	3360	0.723 ?!	140

UPDATE 2020-09-08:

Tool	time [s]	AUC	RAM train [GB]
catboost	930	0.736	50

100M records and RAM usage

GPU (Tesla V100):

Tool	time [s]	AUC	GPU mem [GB]	extra RAM [GB]
h2o xgboost	270	0.755	4	30
xgboost	80	0.756	6	0
lightgbm	400	0.774	3	6
catboost	crash (OOM)		>16	14

UPDATE 2020-09-08:

catboost still crashes out-of-memory

CPU (m5.12xlarge):

Tool	time [s]	AUC	RAM train [GB]
h2o	520	0.775	8
xgboost	510	0.751	15
lightgbm	310	0.774	5
catboost	3360	0.723 ?!	140

UPDATE 2020-09-08:

Tool	time [s]	AUC	RAM train [GB]
catboost	930	0.736	50

```
## exporting model for scoring
```

```
h2o.download_mojo(md_rf, path = "./h2o")
```

```
## building prediction service
```

```
# (need jetty-runner.jar R00T.war from Steam)
```

```
java -jar jetty-runner.jar R00T.war
```

```
curl -X POST --form mojo=@h2o_RF.zip --form jar=@h2o-genmodel.jar \  
localhost:8080/makewar > h2o_RF_M0J0.war
```

GitHub Gist



szilard / [h2o_scoring.R](#)

H₂O.ai

```
## exporting model for scoring
```

```
h2o.download_mojo(md_rf, path = "./h2o")
```

```
## building prediction service
```

```
# (need jetty-runner.jar R00T.war from Steam)
```

```
java -jar jetty-runner.jar R00T.war
```

```
curl -X POST --form mojo=@h2o_RF.zip --form jar=@h2o-genmodel.jar \  
localhost:8080/makewar > h2o_RF_M0J0.war
```

```
## run prediction service
```

```
java -jar jetty-runner.jar --port 20000 h2o_RF_M0J0.war
```

```
## score via REST API
```

```
time curl "http://localhost:20000/predict?Month=c-8&DayofMonth=c-21&Day  
# (fast scoring needs JVM to warm up with a few requests)
```

GitHub Gist

Search...



szilard / [h2o_scoring.R](#)

The logo for H2O.ai, featuring the text "H2O.ai" in a bold, black, sans-serif font. The "2" is stylized with a subscript. The background is a solid yellow rectangle.

```
## read CSV (e.g. data.table::fread) or get data from database (SQL connector)
## do one-hot-encoding of categorical variables (e.g. Matrix::sparse.model.matrix)

## special optimized data structure
dxbg_train <- xgb.DMatrix(data = X_train, label = y_train)

## TRAIN
md <- xgb.train(data = dxbg_train,
                objective = "binary:logistic",
                nround = 100, max_depth = 10, eta = 0.1,
                tree_method = "hist")

## SCORE
yhat <- predict(md, newdata = X_test)

## evaluation (score distribution, ROC curve, AUC etc.)
```



```
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 ca dreds 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	lightgbm	h2o	catboost
easy R install	CRAN	CRAN	java+CRAN	devtools+pre-binary
maintained	yes	yes	yes	yes
preprocessing	1-hot	1-hot/categ int	not needed	categ internal
new cats scoring	no	no	yes	no
early stopping	yes	yes	yes	yes
speed (CPU)	fastest	fastest	slow (small data)	slow
GPU supported	yes	yes	via xgboost	yes, but mem usage
speed GPU	fastest	slow	indirectly/slow	slow on larger data
REST scoring	no	no	yes	no
other algos	RF	RF	RF/GLM/NN	none
best for	Kaggle	Kaggle	prod/real-time	Kaggle

Winning Solution of KaggleDays 2019 Competition in San Francisco

Winning Solution of KaggleDays 2019 Competition in San Francisco

During the first two to three hours of the competition, we focused on exploratory data analysis (EDA). We first analyzed the distributions of raw features between train and test

Winning Solution of KaggleDays 2019 Competition in San Francisco

During the first two to three hours of the competition, we focused on exploratory data analysis (EDA). We first analyzed the distributions of raw features between train and test

For model training, LightGBM is chosen as the baseline model for doing quick experiments on raw and engineered features. It turned out that our engineered features improved AUC in

Winning Solution of KaggleDays 2019 Competition in San Francisco

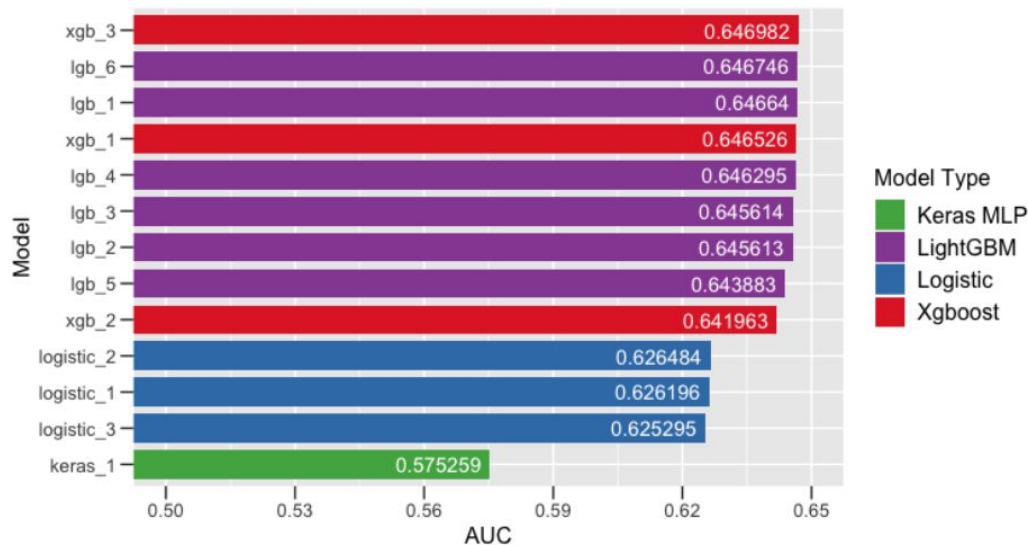
During the first two to three hours of the competition, we focused on exploratory data analysis (EDA). We first analyzed the distributions of raw features between train and test

For model training, LightGBM is chosen as the baseline model for doing quick experiments on raw and engineered features. It turned out that our engineered features improved AUC in

Based on the LightGBM baseline notebook, I created other notebooks for Xgboost, Logistic Regression, Random Forests and Extra Trees models. To save some time, I ran Random

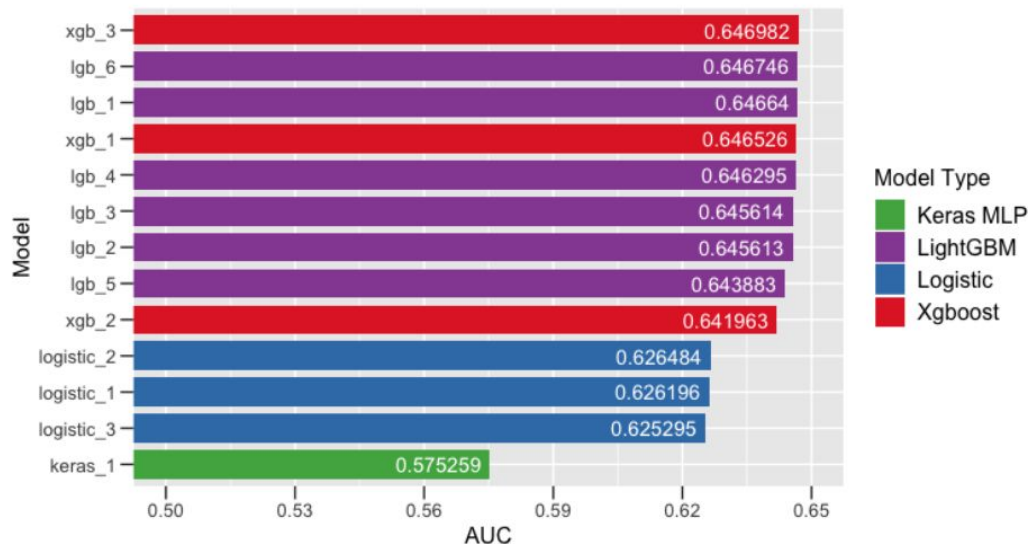
Winning Solution of KaggleDays 2019 Competition in San Francisco

During the first two to three hours of the competition, we focused on exploratory data analysis (EDA). We first analyzed the distributions of raw features between train and test



Winning Solution of KaggleDays 2019 Competition in San Francisco

During the first two to three hours of the competition, we focused on exploratory data analysis (EDA). We first analyzed the distributions of raw features between train and test



For blending and stacking, we looked for diverse models that perform relatively well, but are not highly correlated with each other. Xgboost and LightGBM had very similar

I have a CSV file with data on millions of credit card transactions that has as columns various input parameters (e.g. country, payment type, past spending on various timeframes etc.) and a binary output whether the transaction turned out to be fraudulent or not. I'd like to build a system that can predict the likelihood of fraud for new transactions. How would you go about it?

Thought for 10s

Awesome—this is a classic (and very high-leverage) supervised ML problem. Here's a practical, end-to-end plan I'd follow to get from raw transactions to a production fraud-risk scorer that outputs a calibrated probability for each new transaction.

1) Frame the objective

- **Target:** `is_fraud` (0/1). Predict $P(\text{fraud} \mid \text{transaction})$.
- **Decision:** convert probabilities to actions (approve / step-up auth / decline / send to manual review).
- **Optimize for business value, not just AUC.** Define costs:
 - Cost(FN) = loss on successful fraud.
 - Cost(FP) = friction/revenue loss from falsely blocking/stepping up.
 - Add **capacity constraints** (e.g., manual review can handle N tx/day).

2) Data splitting & leakage control (critical)

- **Time-based split** (train on older data → validate/test on newer). Fraud patterns drift.
- **Group awareness:** keep all transactions from the same **card/account/device** in the same fold to

3) Data quality & preprocessing

- Handle **missing values** (explicit “missing” category for categoricals; simple imputation for numerics).
- **Categoricals**: high-cardinality (merchant_id, device_id, IP, BIN) → target/impact encoding with nested CV, or use CatBoost.
- **Rare category bucketing** for very infrequent values.
- **Outliers**: cap extreme numerics (winsorize) when helpful.
- **Normalization** not required for tree models; useful for linear/NNs.
- Remove/guard against **data leakage features** (post-transaction signals, fraud investigation flags, etc.).

4) Feature engineering (where most lift comes from)

- **Velocity features** (per card/account/device/IP/merchant):
 - Count/amount in last 5/15/60 minutes, 24 hours, 7/30 days.
 - Number of distinct merchants/countries/devices in short windows.
- **Behavioral patterns**: time-of-day, day-of-week, weekend/holiday, recency since last purchase.

6) Modeling baseline → strong learners

- **Baselines:** regularized logistic regression (gives fast, transparent benchmark).
- **Workhorses:** **Gradient-boosted trees** (LightGBM/XGBoost) or **CatBoost** (great with categoricals).
- Consider a **hybrid**: simple rules for known patterns (e.g., impossible geodistance) + model score.

```
dtrain <- xgb.DMatrix(X_train, label = y_train)
dval   <- xgb.DMatrix(X_val,   label = y_val)

param <- list(
  objective      = "binary:logistic",
  eval_metric    = "aucpr",           # directly optimize PR-AUC proxy
  eta            = 0.03,
  max_depth      = 8,
  subsample      = 0.8,
  colsample_bytree = 0.8,
  lambda         = 2.0,
  alpha          = 1.0,
  scale_pos_weight = scale_pos_weight,
  nthread        = max(1, parallel::detectCores() - 1)
)

watch <- list(train = dtrain, eval = dval)
bst <- xgb.train(
  params = param, data = dtrain, nrounds = 1200,
  watchlist = watch, verbose = 0,
  early_stopping_rounds = 100
)
```

Why are you using "traditional" machine learning? Can't you use LLMs instead for better results?

You *can* try to use an LLM for fraud detection, but for this kind of data—millions of structured credit card transactions with engineered numeric/categorical features—the evidence still points toward “traditional” supervised learning (GBDTs like LightGBM, XGBoost, CatBoost) being the better core choice.

What are the top 5 machine learning algorithms / implementations that would likely provide the best accuracy on tabular data such as credit card fraud? Give me a short answer.

1. **LightGBM** – fast, memory-efficient GBDT, great for large/tabular data.
2. **CatBoost** – GBDT with excellent categorical handling.
3. **XGBoost** – mature, robust GBDT baseline.
4. **Random Forest** – strong, stable ensemble baseline.
5. **TabNet** – deep learning model for tabular data (good if mixing with embeddings).

TABULAR DATA YOU HAVE



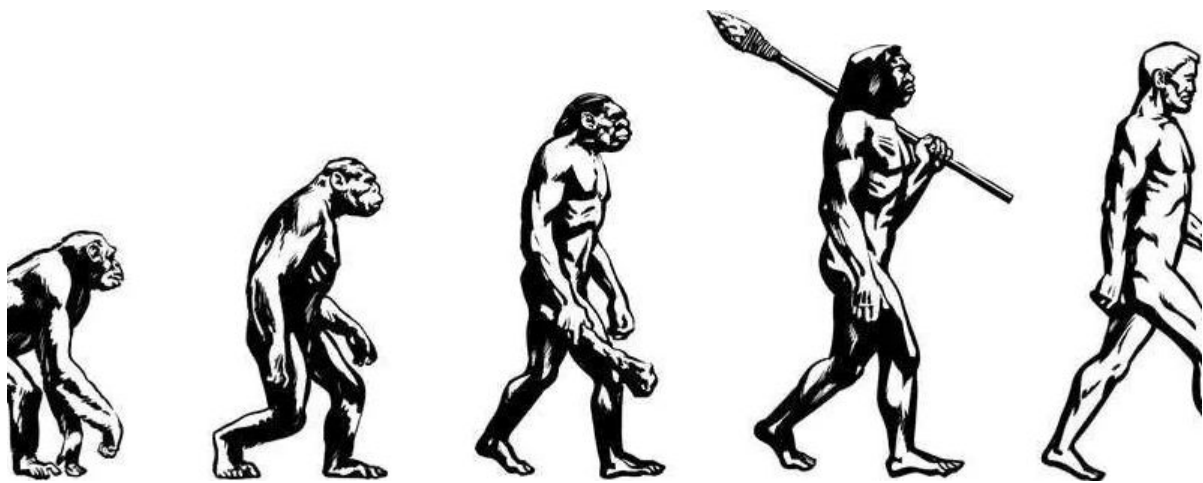
GBM USE YOU MUST











More:



szilard / **benchm-ml**

★ Star

1,203



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



szilard / **teach-ML-CEU-master-bizanalytics**



szilard / **GBM-intro**



szilard / **GBM-workshop**



szilard / **ML-scoring**



szilard / **GBM-perf**



szilard / **GBM-tune**



szilard / **GBM-multicore**

GitHubGist

Search...



szilard / **h2o_scoring.R**



szilard / **ML_with_H2O.R**



 spafka@gmail.com

 [@SzilardPafka](https://twitter.com/SzilardPafka)

 linkedin.com/in/szilard

 github.com/szilard