Session-8 Other Ensemble Techniques

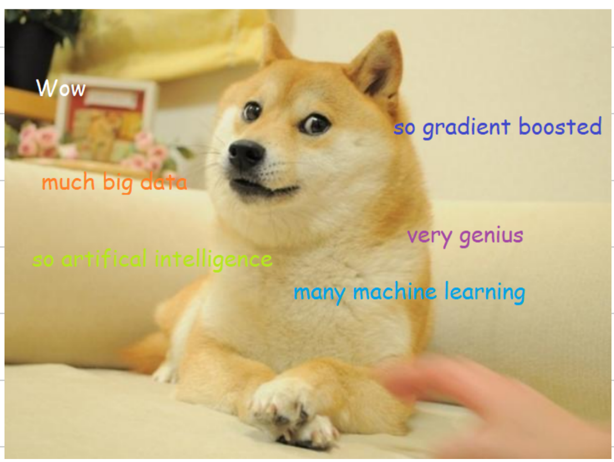
Sep 10, 2025

Agenda

1. XGBoost
2. LightGBM
3. Stacking

Cascading

The client after I expertly explain how the model works



How boosting works for stochastic GBDT.

	f_1	f_2	f_3	f_4	f_5	f_6	y
x_1							
x_2							
x_3							
x_4							
x_5							
x_6							
x_7							
x_8							

$m_2 \rightarrow x_2, x_3 ; F_2, F_5$

$m_3 \rightarrow x_7, x_8 ; F_1, F_6$

This is HW.

m_2.fit(x_2, x_3; F_2, F_5)

$\boxed{m_2} \rightarrow$ m_2.predict(Entire dataset)

$\downarrow$

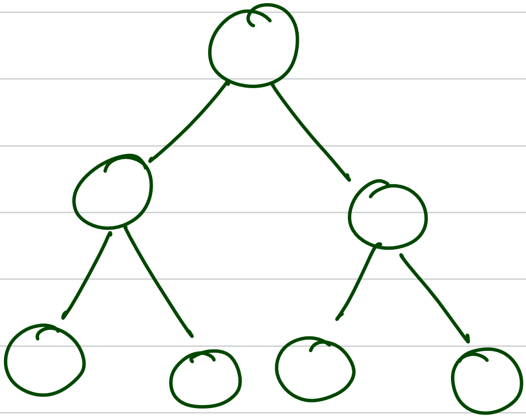
compute residuals

$\downarrow$

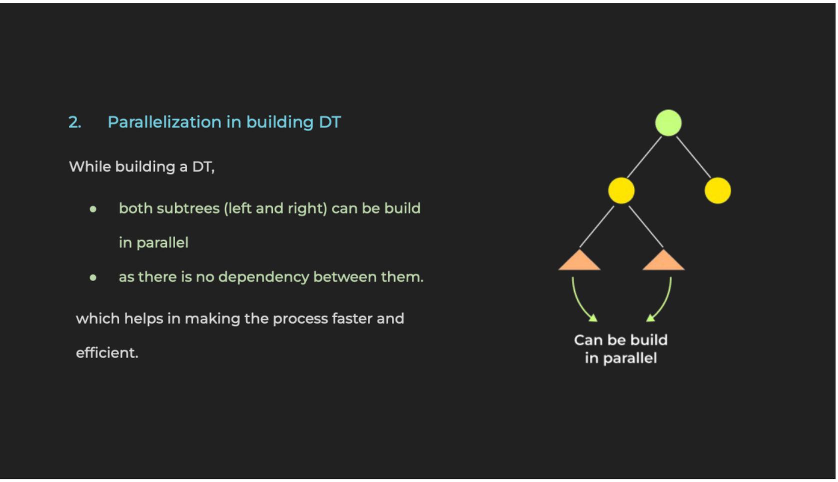
$\boxed{m_3}$ — ϵ_7, ϵ_8

XGBOOST

① Parallelize Feature Selection



② Parallelize building of decision trees



- device [default= cpu]

Added in version 2.0.0.

  - Device for XGBoost to run. User can set it to one of the following values:
    - cpu : Use CPU. — distributed computing ; Apache spark
    - cuda : Use a GPU (CUDA device).
    - cuda:<ordinal> : <ordinal> is an integer that specifies the ordinal of the GPU (which GPU do you want to use if you have more than one devices).
    - gpu : Default GPU device selection from the list of available and supported devices. Only cuda devices are supported currently.
    - gpu:<ordinal> : Default GPU device selection from the list of available and supported devices. Only cuda devices are supported currently.

CPU → $\boxed{\text{CUDA}}$ → GPU

③ Optimized thresholding.

F1, F2, F3

fure → [0, 100]

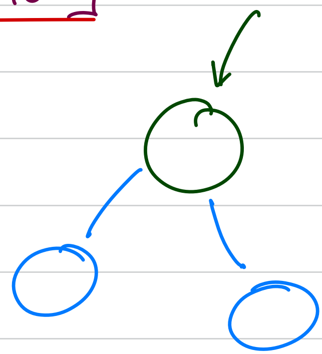
\> 10✓

20✓

30✓

⋮

99✓



④ Diff types of base Learner

→ Gb Linear — Reg (Linear Reg)

→ Gb tree — DT (stump n)

→ Dart — DT with dropout

NN model

class **xgboost.XGBClassifier**(**, objective='binary:logistic', use_label_encoder=None, **kwargs*)

Bases: `XGBModel`, `ClassifierMixin`

Implementation of the scikit-learn API for XGBoost classification.

Parameters:

- **n_estimators** (*int*) – Number of boosting rounds.
- **max_depth** (*Optional[int]*) – Maximum tree depth for base learners.
- **max_leaves** – Maximum number of leaves; 0 indicates no limit.
- **max_bin** – If using histogram-based algorithm, maximum number of bins per feature
- **grow_policy** – Tree growing policy. 0: favor splitting at nodes closest to the node, i.e. grow depth-wise. 1: favor splitting at nodes with highest loss change.
- **learning_rate** (*Optional[float]*) – Boosting learning rate (xgb's "eta")
- **verbosity** (*Optional[int]*) – The degree of verbosity. Valid values are 0 (silent) - 3 (debug).
- **objective** (*Union[str, Callable[[numpy.ndarray, numpy.ndarray], Tuple[numpy.ndarray, numpy.ndarray]], NoneType]*) – Specify the learning task and the corresponding learning objective or a custom objective function to be used (see note below).
- **booster** (*Optional[str]*) – Specify which booster to use: gbtree, gblinear or dart.
- **tree_method** (*Optional[str]*) – Specify which tree method to use. Default to auto. If this parameter is set to default, XGBoost will choose the most conservative option available. It's recommended to study this option from the parameters document tree method
- **n_jobs** (*Optional[int]*) – Number of parallel threads used to run xgboost

What does parameter n_estimator represent in Tree based model?

17 users have participated

A	Number of feature	12%
B	Number of parameter to be tuned	24%
C	Number of ensembled trees	65%
D	None of these	0%

End Quiz Now

Rank	Name	Score	Points
2	Samyucktha Ra...	1/1	95.93
1	Karthik	1/1	97.11
3	Shoreya gupta	1/1	95.93
4	SHASHANK JHA	1/1	95.13
5	Sumanth Andhavarapu	1/1	94.60
6	Souvik Adhikary	1/1	94.47
7	Snehal Adhikary	1/1	92.87
8	Perisetla Pavan Kalyan	1/1	92.50
9	Purushottam Kumar	1/1	92.47
10	Deependu Ghosh	1/1	90.96

```
from xgboost import XGBClassifier
```

LightGBM

① GOSS

Gradient based one sided sampling.

x_i	y_i	M_0	E_0	M_1	E_1

LightGBM to other machine learning algorithms.

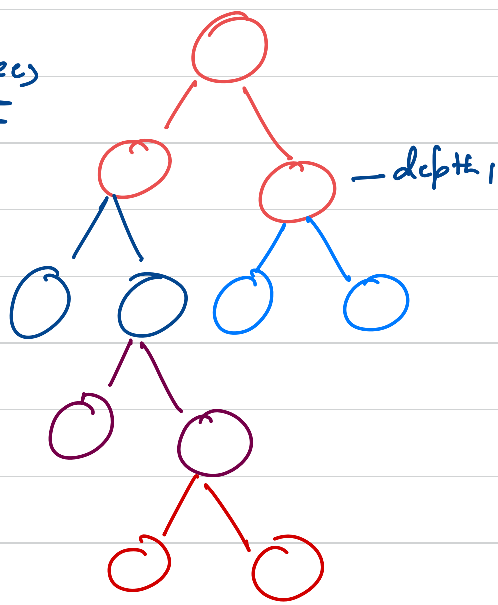
E_1	$\to$ sort(E_1)	M_2	E_2
10	21		
9	10		
3	9		
8	8		
6	6		
21	3		

$((r_1, r_4), \varepsilon_0)$ $(r_1, r_2, \varepsilon_1)$

	Age (x_i)	Price (y_i)	M_0	ε_0	M_1	ε_1	M_2	ε_2	M_3
r_1	5	17	10.6	6.4	5.5	0.9	0.8	0.1	
r_2	2	20	10.6	9.4	8.0	1.4	1.1	0.3	
r_3	10	8	10.6	−2.6	−1.5	−1.1	−0.8	−0.3	
r_4	15	5	10.6	−5.6	−1.1	−1.1	−0.9	−0.2	
r_5	20	3	10.6	−7.6	−1.6	−1.6	−1.1	−0.5	

2 Decision trees — model training

Depth wise decision trees



3 Exclusive Feature bundling.

f_1	f_2	f_3	Composite Column
1	0	0	1
0	1	0	2
0	0	1	3
1	0	0	1
0	1	0	2

When to use GOSS and EFB ? Select the one which applies

0 users have participated

✓	A	GOSS = Sampling, EFB = Dimensionality Reduction	0%
	B	EFB = Sampling, GOSS = Dimensionality Reduction	0%
	C	GOSS = Sequencing, EFB = Dimensionality Reduction	0%
	D	GOSS = Sampling, EFB = Slows GBDT	0%

End Quiz Now

Based on all quizzes from the session

Rank	Name	Score	Points
2	Deependu Ghosh	2/2	182.82
1	SA Souvik Adhikary	2/2	186.82
3	SR Samyucktha Ra...	2/2	176.64
4	Sri Harsha Nanduri	2/2	172.97
5	Shoreya gupta	2/2	172.42
6	TS Tanvi Singh	2/2	129.86
7	Paramhans N. Chetiwal	2/2	124.46
8	K Karthik	1/2	97.11
9	SJ SHASHANK JHA	1/2	95.13
10	S Sumanth Andhavarapu	1/2	94.60

```
import lightgbm as lgb
```

Stacking

x_i | y_i

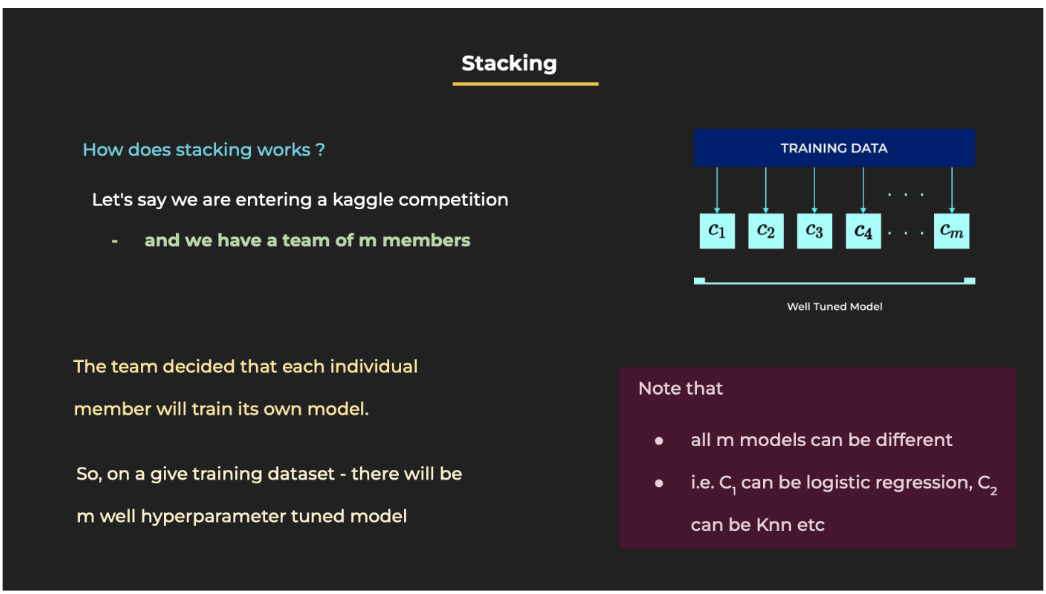
Linear Reg.

Logistic Reg. → 0.1 | 0.11

D.T. → 0.2 | 0.11

R.F — 0.01 | 0.11

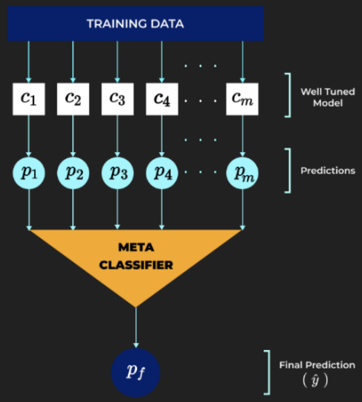
Neural Netwo — 0.02 | 0.11



How do we combine them?

Each model will give us a prediction (P_1, P_2, ... P_n)

- We train another model (**Meta Classifier**)
  - using predictions as input data
  - and original target variable (y) as target variable



Note : Instead of predictions, we can use class probabilities as input feature for meta classifier

The prediction given by Meta Classifier is treated as final prediction

What is stacking?

0 users have participated

A	A meta-algorithm that combines the predictions of multiple ensemble models	0%
B	A method for reducing the dimensionality of the feature space	0%
C	A method for adjusting the weights of different models in an ensemble	0%
D	A method for combining the predictions of different models in an ensemble	0%

End Quiz Now

2	Samyucktha Ra...	3/3	269.11
1	Souvik Adhikary	3/3	272.69
3	Deependu Ghosh	3/3	263.95
4	Sri Harsha Nanduri	3/3	233.23
5	Karthik	2/3	182.59
6	SHASHANK JHA	2/3	180.99
7	Purushottam Kumar	2/3	180.53
8	Shoreya gupta	2/3	172.42
9	Perisetla Pavan Kalyan	2/3	155.36
10	Snehal Adhikary	2/3	155.07

Cascading

Baselearners - DT
unconstrain depth