Advanced Tree-Based Methods (MSBA 7027)
GBM, Stochastic GBM, XGBoost

Zhengli Wang

Faculty of Business and Economics
The University of Hong Kong

2023

Gradient Boosting Machines (GBMs)

Extremely popular ML algorithm
Successful across many domains
Leading methods for winning ML competitions

Builds shallow trees in sequence
Each tree learns and improves on the previous one

How Boosting Works

Main idea: add new models (decision trees) sequentially

Note: the textbook uses "base learner" for weak model, we reserve this term for the Stacking topic to avoid confusion

Start with a weak model,
Sequentially boosts performance by adding new weak models,
Each new weak model fixes mistakes from previous ones

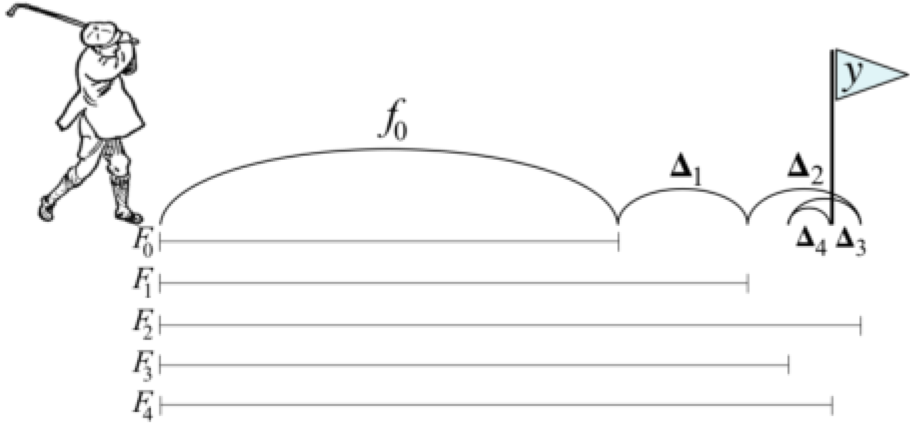
Main steps

1. Specify the weak model (e.g. max depth / min. node size)
2. Sequential training of weak model

How Boosting Works

1. Specify the weak model (e.g. max depth / min. node size)
Usually: small-size decision trees
Most common: Trees with depth ≤ 8

2. Sequential training of weak model
Each weak model tries to fix mistakes of previous ones
In sequential manner



Sequential model ensemble: each iter. gradually nudges predicted values to target

GBM Numerical Example

Assume Learning Rate = 1

X	Y	Pred aft 1st T	Residual aft 1st T	Pred aft 2nd T	Residual aft 2nd T	Pred aft 3rd T	Residual aft 3rd T
2	6	20/3	- 2/3				
5	10	20/3	10/3				
6	4	20/3	- 8/3				

Note: Residual = true value – predict value

GBM Numerical Example

Assume Learning Rate = **0.5**

X	Y	Pred aft 1^{st} T	Residual aft 1^{st} T	Pred aft 2^{nd} T	Residual aft 2^{nd} T	Pred aft 3^{rd} T	Residual aft 3^{rd} T
2	6	20/3	- 2/3				
5	10	20/3	10/3				
6	4	20/3	- 8/3				

Note: Residual = true value – predict value

Ames Housing Example – Basic GBM

Basic Implementation

Hyperparameters

- Boosting Hyperparameters: #Trees, Learning rate

- Tree Hyperparameters: Tree depth, Min Node size

Ames Housing Example – Basic GBM

Basic Implementation

Boosting Hyperparameters:
#Trees: total #trees in the boosting sequence
GBM requires many trees
Can overfit – find best #trees by CV

Learning rate: Controls how quickly GBM algorithm proceeds
Also called "shrinkage"
Value ranges from (0, 1), typical values between (0.001, 0.3)
Smaller values: typically model more accurate, but algorithm takes longer

Ames Housing Example – Basic GBM

Basic Implementation

Tree Hyperparameters:
Tree depth: Controls the complexity (depth) of individual trees
Typical value 3-8
High depth: model more accurate, but algorithm takes longer, more prone to overfitting

Min Node size: min. #observations in terminal nodes
Also controls the complexity of each tree
Typical value 5-15

Ames Housing Example – Basic GBM

Basic Implementation

gbm package in R

Start with: Learning rate 0.1, # Trees = 5000, Tree depth = 3, min node size = 10 (default), 10-fold CV

```
library(gbm)       # for original implementation

# Run a basic GBM model
# RUNTIME: 2 minutes on i7 CPU
set.seed(123)  # for reproducibility
system.time( ames_gbm1 <- gbm(
    formula = Sale_Price ~ .,
    data = ames_train,
    distribution = "gaussian",  # SSE loss function
    n.trees = 5000,             # number of trees
    shrinkage = 0.1,            # learning rate, default is 0.001
    interaction.depth = 3,      # depth of each tree
    n.minobsinnode = 10,
    cv.folds = 10               # 10-fold CV
))
```

For Classification Problem, use distribution = "Bernoulli"

Boosting hyperpara. (n.trees, shrinkage)

Tree hyperpara. (interaction.depth, n.minobsinnode)

Above code returns model_object ames_gbm1, with CV errors for all #trees up to 5000

Ames Housing Example – Basic GBM

Basic Implementation

gbm package in R

Start with: Learning rate 0.1, # Trees = 5000, Tree depth = 3, min Node size = 10 (default), 10-fold CV

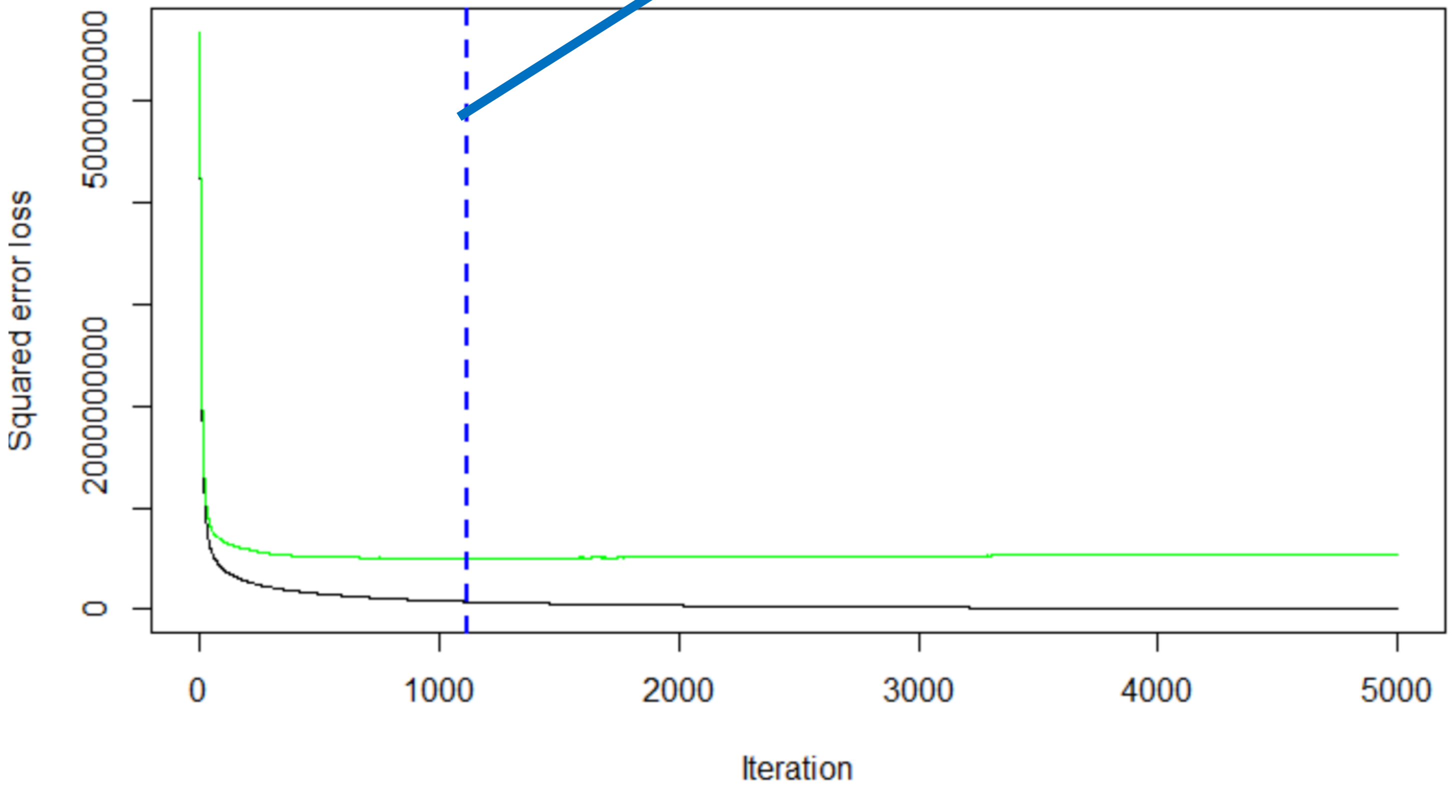
```
# find index for number trees with minimum CV error
best <- which.min(ames_gbm1$cv.error)

# get MSE and compute RMSE
sqrt(ames_gbm1$cv.error[best])

# Plot error curve - Figure 12.6
# Training and cross-validated MSE as n trees are added to the GBM algorithm.
gbm.perf(ames_gbm1, method = "cv")
# NOTE: Our results show a cross-validated SSE of 22402 which was achieved with 1119 trees.
#       Gray and green line: Training and cross-validated MSE
#       Blue line  - Achieved with 1119 trees
```

22402

Ames Housing Example – Basic GBM

Basic Implementation

Achieve min CV MSE: About 1000 trees



Green curve: cross-validated MSE
Grey curve: training MSE

Ames Housing Example – Basic GBM

Tuning Hyperparameters

- Simple way: full grid-search (more details refer to RF)
- More advanced (from book): alternating optimization between learning rate & tree hyperpara.

Ames Housing Example – Basic GBM

Tuning Hyperparameters: Alternating Optimization

1. Set learning rate = 0.1 (always a good start value) and determine best #trees
2. Fix tree hyperparameters & tune learning rate → assess speed vs. performance
3. Fix learning rate & tune tree-specific hyperparameters