Today

- Basic boost algorithm for ants
- Innovations so far that address bias-variance tradeoff
- Putting all the innovations together
  - stochastic gradient descent
  - extreme gradient boosting
  - xgboost

Boosting – model or training?

- Boosting = training algorithm
- Model ensemble = model algorithm
  - e.g. for a tree-based model
  - walk each tree to get predictions
  - average the trees, weighted by the learning rate
- In this sense, the model algorithm is the same as bagging or random forests but models are weighted

Innovations so far

- What innovations do we have so far that address the bias-variance tradeoff to give accurate out-of-sample predictions?

Stochastic gradient descent

Training algorithm (linear model)

set λ (learning rate)
make initial guess for β_0, β_1
for many iterations
    randomly sample rows (y, x) ← new step
    find gradient at β_0, β_1
    step down: $\beta = \beta - \lambda$ gradient(β)
print β_0, β_1

random sample options: bootstrap, subset, single points, mini-batch

Stochastic gradient descent

- stoch_gradient_descent.R

Stochastic gradient boosting

Training algorithm

load y, x, x_{new}

set $\hat{f}(x_{\text{new}})$ = 0

set $r \leftarrow y$ (residuals equal to the data)

for m in 1 to n_iterations

randomly sample rows ← new step

train model on r and x

predict residuals, $\hat{r}_m(x)$, from trained model

update residuals: $r \leftarrow r - \lambda \hat{r}_m(x)$

predict y increment, $\hat{f}_m(x_{\text{new}})$, from trained model

update prediction: $\hat{f}(x_{\text{new}}) \leftarrow \hat{f}(x_{\text{new}}) + \lambda \hat{f}_m(x_{\text{new}})$

return $\hat{f}(x_{\text{new}})$

Extreme gradient boosting

Training algorithm

load y, x, x_{new}

set $\hat{f}(x_{\text{new}})$ = 0

set $r \leftarrow y$ (residuals equal to the data)

for m in 1 to n_iterations

randomly sample rows and columns

train model on r and x

predict residuals, $\hat{r}_m(x)$, from trained model

update residuals: $r \leftarrow r - \lambda \hat{r}_m(x)$

predict y increment, $\hat{f}_m(x_{\text{new}})$, from trained model

update prediction: $\hat{f}(x_{\text{new}}) \leftarrow \hat{f}(x_{\text{new}}) + \lambda \hat{f}_m(x_{\text{new}})$

return $\hat{f}(x_{\text{new}})$

Extreme gradient boosting

- ants_extreme_gradient_boost.R

Boosting packages in R

- gbm: gradient boosting machines
  - boosted decision trees
  - C++
  - regression, classification, + extensions (e.g. survival, count data, quantile regression)
  - fast, good, stable, maintained
  - retired (no new features)
- See ants_boosted_tree.R

Boosting packages in R

- gbm3: successor to gbm
  - apparently mostly abandoned ca 2017 (perhaps due to xgboost?)
  - active development restarted Jan 2024
  - github only for now
  - watch this space?

Boosting packages in R

- xgboost: extreme gradient boosting
  - R interface to very fast C++ library
  - many additional algorithm innovations to speed training
  - large data innovations
  - inbuilt parallel and GPU support
  - current industry standard (interfaces to all major data science languages)

xgboost

- slow learning
  - gradient descent, boosting
- random sampling data
  - like bagging, but no replacement
- random sampling variables
  - like random forest but options (per tree, per node)
- regularized cost function
  - like smoothing spline, tree complexity penalty
- best split algorithm optimizes the whole tree
  - not per split

xgboost

- ants_xgboost.R
- We'll look at:
  - one hot encoding
  - hyperparameters

Categorical variables

- “One hot” encoding

habitat
“forest”
“forest”
“bog”
“grassland”
“bog”
“grassland”
“forest”

bog	forest	grassland
0	1	0
0	1	0
1	0	0
0	0	1
1	0	0
0	0	1
0	1	0

- binary categories: only needs one column

xgboost hyperparameters

- There are a lot of them, reflecting all the different algorithmic components that can be included
- To see what's available, in R type

```
?xgboost
```

Tuning strategies

- Tuning is hyperparameter optimization
- Grid search
- Random search
- Evolutionary algorithms
- Bayesian optimization

Bischl et al (2023). Hyperparameter optimization: foundations, algorithms, best practices, and open challenges. https://doi.org/10.1002/widm.1484.

ML “pipeline” packages

- R: tidy models
- R: caret (predecessor to tidy models)
- Python: scikit-learn
- Handle
  - data pre-processing (“feature engineering”)
  - test-train splits
  - hyperparameter tuning
- Can use for independent project (optional)