Gradient Boosting Trees: theory and applications

Dmitry Efimov

November 05, 2016

Outline

Decision trees

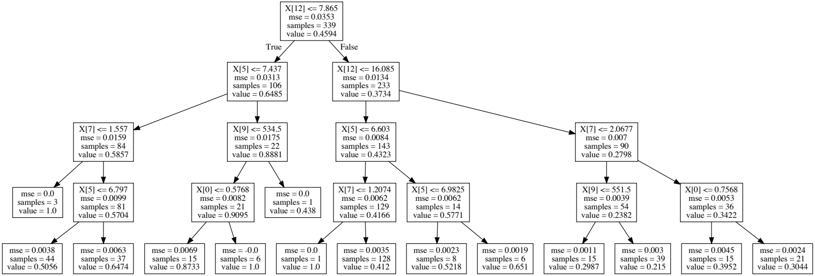
Boosting

Boosting trees

Metaparameters and tuning strategies

How-to-use remarks

Regression tree

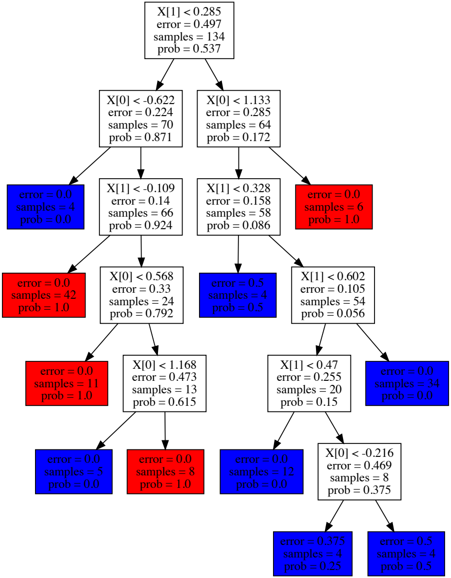


Mean square error for node k: $\frac{1}{m_k} \sum_{i \in R_k} \left(y^{(i)} - \mu_k\right)^2$

m_k - number of samples

μ_k - average

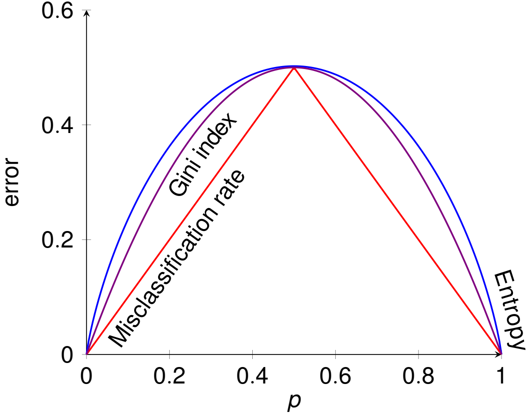
Classification tree



Classification error (two classes example)

p - % of samples from one class in the node

- Misclassification error: $\min(p, 1-p)$
- Gini index: $2p(1-p)$
- Cross-entropy: $-p \ln p - (1-p) \ln(1-p)$



Boosting (backfitting algorithm)

Generalized additive model:

$$\hat{y} = f(x_1, \ldots, x_n) = \alpha + f_1(x_1) + f_2(x_2) + \ldots + f_n(x_n)$$

Algorithm 1 Backfitting algorithm for GAM

set initial values $\alpha = \frac{1}{m}\sum_{i=1}^{m} y^{(i)}$, $f_j = 0$ for all $j = 1, \ldots, n$
repeat
  for $j = 1$ **to** n **do**
    evaluate working targets $z^{(i)} = y^{(i)} - \alpha - \sum_{k=1, k \neq j}^{n} f_k(x_k^{(i)})$
    train model with feature x_j and target z to estimate f_j
until convergence
return α, f_j

Boosting (general idea)

Loss function for nonparametric model:

$$L(f) = \frac{1}{2m}\sum_{i=1}^{m}(y^{(i)} - f(x^{(i)}))^2$$

- From backfitting algorithm: $f^{new} = f^{old} + g$, where g is a building block algorithm
- Gradient Descent with respect to f: $f^{new} = f^{old} - \alpha \frac{dL}{df}\Big|_{f=f^{old}}$

General idea: we train the building block algorithm with the outputs

$$g = -\frac{dL}{df}\Big|_{f=f^{old}}$$

Boosting trees

Algorithm 2 Gradient Tree Boosting

Initialize $f_0(x) = \arg\min_{\mu} \sum_{i=1}^{m} L(y^{(i)}, \mu)$

for $k = 1$ **to** K **do**

Compute working target $r_k^{(i)} = -\left(\frac{dL}{df}\right)\Big|_{f=f_{k-1}(x^{(i)})}$

Fit a regression tree to the targets $r_k^{(i)}$ with terminal nodes $R_{kj}, j = 1, \ldots, J_k$ and compute

$$\gamma_{kj} = \arg\min_{\gamma} \sum_{x^{(i)} \in R_{kj}} L(y^{(i)}, f_{k-1}(x^{(i)}) + \gamma)$$

Update $f_k(x) = f_{k-1}(x) + \sum_{j=1}^{J_k} \gamma_{kj} \mathbb{1}\{x \in R_{kj}\}$

return $f_K(x)$

Metaparameters

- **General:** booster, seed, subsample, colsample_bytree, colsample_bylevel, eval_metric
- **Optimization related:** objective, eta, gamma, lambda, alpha, num_round, scale_pos_weight
- **Tree related:** max_depth, min_child_weight

General metaparameters

- **booster**: gbtree, gblinear, dart
- **seed**
- **subsample**: number of training examples for each tree
- **colsample_bytree**: number of features for each tree
- **colsample_bylevel**: number of features for each tree node
- **eval_metric**: rmse, mae, logloss, auc, map

Optimization and tree related metaparameters

Optimization:

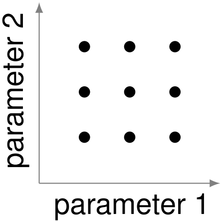
- **objective**: reg:linear, binary:logistic, multi:softprob, rank:pairwise
- **eta**: learning rate
- **gamma**: minimum loss reduction required
- **lambda**: L2 regularization
- **alpha**: L1 regularization
- **scale_pos_weight**: weights for classes
- **num_round**: number of iterations

Tree:

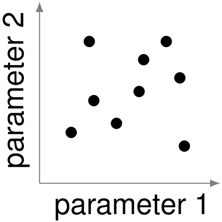
- **max_depth**: maximum depth of tree
- **min_child_weight**: minimum size of tree node

Tuning strategies

- Grid search:



- Randomized search:



- Manual tuning

When to apply xgboost? (just my observations)

- features of different origins: categorical, numerical, ordinal
- features are not correlated a lot
- the number of features is comparatively small
- the problem is not of some specific type (for example, not image recognition or time series)
- the parametric approach cannot be used

General strategy

1. Use xgboost with basic parameters without tuning
2. Read literature about other approaches
3. Compare the results

Usecases

- relational datasets (Genentech, RiskyBusiness, Deloitte):
  Ex.: github.com/diefimov/genentech_2016
- datasets with features of different origins (Otto):
  Ex.: github.com/diefimov/otto_2015
- works for time series, but they should be converted to the traditional format (West Nile, Western Australia):
  Ex.: github.com/diefimov/west_nile_virus_2015

References

- T.Chen and C.Guestrin. ”XGBoost: A Scalable Tree Boosting System.” *In 22nd SIGKDD Conference on Knowledge Discovery and Data Mining*, 2016
- https://xgboost.readthedocs.io/en/latest/
- T.Hastie, R.Tibshirani and J.Friedman ”The elements of statistical learning.” *Springer*, 2009
- https://github.com/diefimov/MTH594_MachineLearning

Thank you! Questions?

Dmitry Efimov

diefimov@gmail.com

kaggle.com/efimov

github.com/diefimov