STAT3612 Lecture 8

Tree-based Methods

Dr. Aijun Zhang

27 October 2020

Department of 統計及精算學系
Statistics & Actuarial Science

Tree-based Methods

- Original CART (Classification and Regression Trees) by Brieman, Friedman, Olshen, and Stone (1984).
- A single small decision tree is easy to interpret, but lack of prediction performance.
- Ensemble learning can make weak learners strong. Schapire (1990): "The strength of weak learnability".
- **Tree ensembles:** bagging, random forests, boosting, stacking, . . . are among the most powerful machine learning algorithms available today.
- The tree ensembles (typically, random forests and boosting) are black box models, and they can be explained by post-hoc interpretability methods.

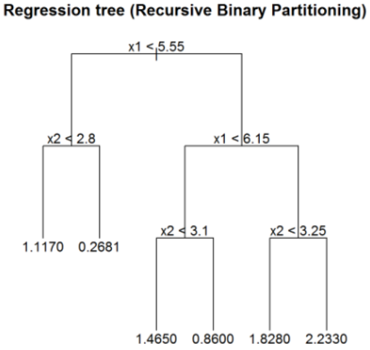
Leo Breiman (1928 - 2005)

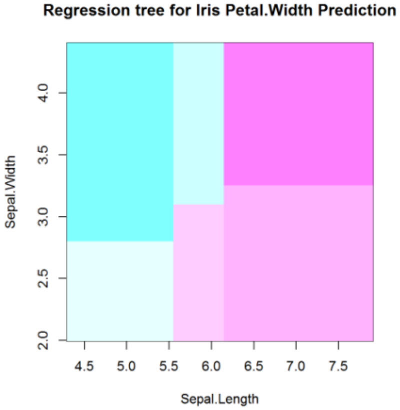
Jerome Friedman

Robert Schapire

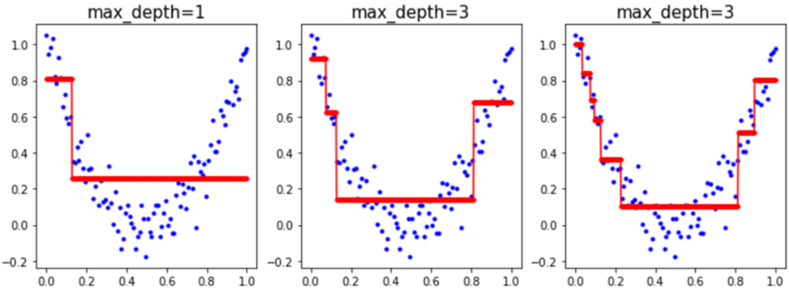
Table of Contents

Decision Trees: 2D case





Decision Trees: 1D case



Regression Trees

- It follows the generalized additive model with piecewise constant features:

$$\hat{f}(\boldsymbol{x}) = \sum_{m=1}^{K} \hat{\mu}_m I(\boldsymbol{x} \in R_m), \quad \hat{\mu}_m = \text{avg}(y_i | \boldsymbol{x}_i \in R_m)$$

- At each recursive step, it finds the best splitting jth variable with the split point s by minimizing the SSE (sum of squared errors):

$$\min_{j,s} \left[\sum_{\boldsymbol{x}_i \in R_1} (y_i - \hat{\mu}_1)^2 + \sum_{\boldsymbol{x}_i \in R_2} (y_i - \hat{\mu}_2)^2 \right]$$

  where the split regions $R_1 = \{\boldsymbol{x}_i | x_{ij} \leq s\}$ and $R_2 = \{\boldsymbol{x}_i | x_{ij} > s\}$.

- This recursive partitioning strategy gives the **tree growing** algorithm, resulting in a large tree T_0.

Cost-complexity Pruning

- For a subtree $T \subseteq T_0$ obtained by pruning T_0, denote by $|T|$ the number of terminal nodes in T. Define the cost complexity criterion

$$\sum_{m=1}^{|T|} \sum_{\boldsymbol{x}_i \in R_m} (y_i - \hat{\mu}_m)^2 + \alpha |T|$$

- The tuning parameter $\alpha \geq 0$ controls the trade-off between tree size and its goodness of fit. It can be estimated by cross-validation.

- Thus we have a tree **first-growing-then-pruning** strategy.

Classification Trees

- Consider the classification problem with multi-class target $(1, 2, \ldots, K)$.
- Let $\hat{p}_{mk}$ be the proportion of class-k observations within terminal node m.
- By majority voting, we classify all the observations in node m to class

$$\hat{k}(m) = \arg\max_{k} \hat{p}_{mk}$$

- Similar to regression trees, we can split nodes and prune the tree upon suitable changes of node impurity measures.

Classification Trees

The impurity measures for each terminal node of a classification tree:

- Misclassification error:

$$\frac{1}{N_m} \sum_{\boldsymbol{x}_i \in R_m} I(y_i \neq \hat{k}(m)) = 1 - \hat{p}_{m\hat{k}(m)}$$

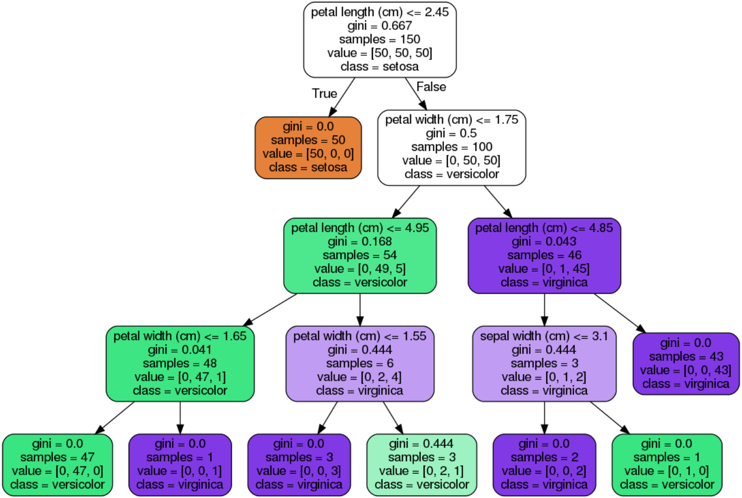
- Gini-index:

$$\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

- Cross-entropy or deviance:

$$\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

Classification Trees



Decision Trees: Summary

- automatically select variables that are used to define the splits;
- are easy to interpret for small-size trees (not so easy for large trees);
- recursively partition the input space as a divide-and-conquer operation;
- may handle both numeric/categorical features seamlessly;
- may deal with missing data effectively;
- but, often suffer from high-variance and therefore usually have poor generalization performances.

Table of Contents

Tree Ensembles

- Tree-based ensembe learners use trees as building blocks to construct powerful prediction models.
- The key is to get rid of the variance by averaging, thus improve the prediction performance.
- **Bagging:** "bootstrap aggregation" of multiple trained trees.
- **Random forests:** improves bagging by split-variable randomization.
- **Boosting:** sequential ensembles (AdaBoost, GBM, XGBoost, ...)

Bagging

- **Bagging** means "bootstrap aggregation" and it takes the form of

$$\hat{f}_{\text{bag}}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(\mathbf{x})$$

  where $f^b(x)$ is trained on the bth bootstrap sample (B in total).

- **Out-Of-Bag (OOB)**: For bootstrap resampling (with replacement), the probability of not being covered by the bth bootstrap sample is

$$\Pr(\mathbf{x}_i \notin \mathbb{X}_b) = \left(1 - \frac{1}{n}\right)^n \approx e^{-1} = 0.368.$$

  Therefore about one-third of observations are not used to fit the bagged trees, which are called Out-Of-Bag (OOB) observations.

- Model evaluation based on the OOB observations yields the OOB error estimate, which is similar to the cross-validation estimate of test error.

Random Forests

- **Random forests** improve the bagging algorithm by *decorrelating* the trees via split-variable randomization.
- Each time only m out of p predictors are chosen at random as split variables. Typical values of m are $\sqrt{p}$ (classification case) and $p/3$ (regression case).
- Therefore, random forests use both horizontal (sample-wise) and vertical (feature-wise) randomization techniques.
- The trees trained in such way have less correlated performance and make the averaging predictor less variable and more reliable.

Boosting

- **Boosting** fits trees sequentially by using information from previous fitted trees.
- (In contrast, bagging or random forests that fit trees in parallel on each re-sampled observations).
- There are multiple boosting algorithms, including AdaBoost, Gradient Boosting, XGBoost (extreme gradient boosting), LightGBM, . . .
- Let us take a look at the AdaBoost algorithm for regression problem (from Chapter 8 of ISLR2013).

Boosting

Algorithm 8.2 *Boosting for Regression Trees*

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.
2. For $b = 1, 2, \ldots, B$, repeat:
   (a) Fit a tree $\hat{f}^b$ with d splits ($d+1$ terminal nodes) to the training data (X, r).
   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:
   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x). \tag{8.10}$$
   (c) Update the residuals,
   $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i). \tag{8.11}$$
3. Output the boosted model,
$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x). \tag{8.12}$$

Thank You!

Q&A or Email ajzhang@umich.edu