IST 5535: Machine Learning Algorithms and Applications

Langtao Chen, Spring 2021

Tree-Based Methods

Reading

- Tree-based methods: book chapter 8

Learning Objectives

- Explain the regression tree and classification algorithms.
- Explain the advantages and disadvantages of decision trees compared with other supervised machine learning methods.
- Explain why pruning a decision may improve performance.
- Explain ensemble learning methods and be able to explain three basic types of ensemble.
- Explain bagging trees.
- Explain random forests and compare random forests with bagging trees.
- Implement tree-based methods in R. Use cross-validation to tune hyperparameters.

AGENDA

- Regression and Classification Trees
- Ensemble Learning and Random Forests

Decision Tree Methods

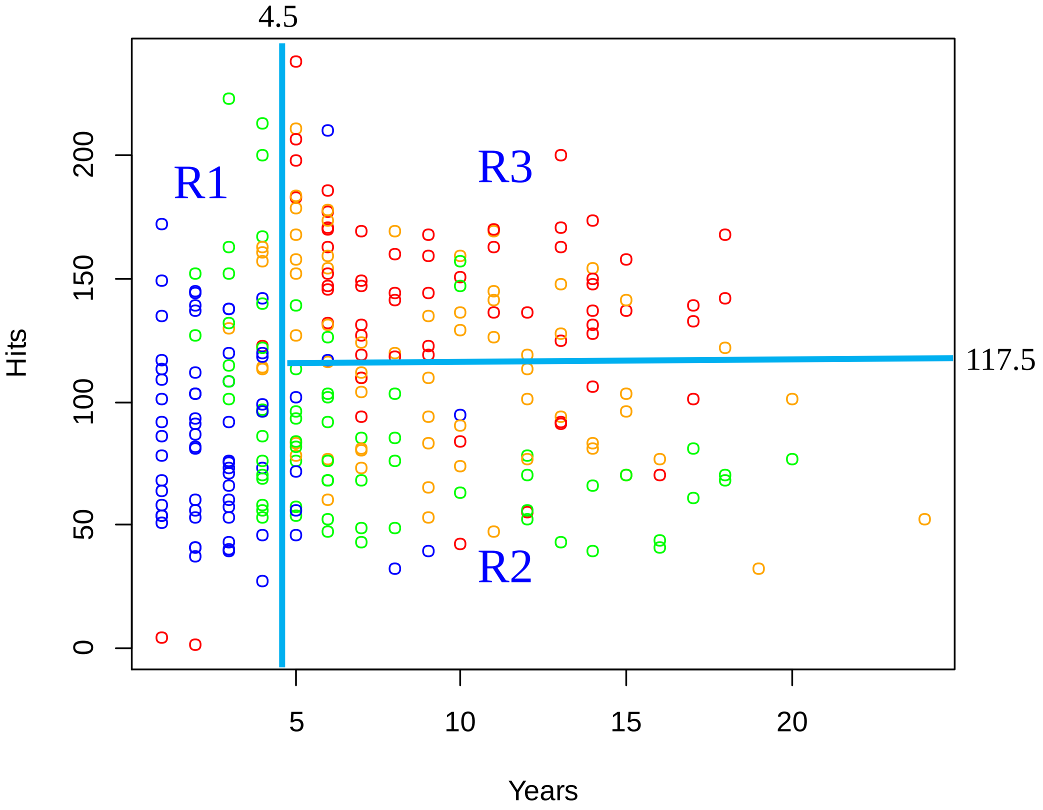
- Decision tree methods can be applied to both regression (regression trees) and prediction (classification trees) problems.

- These methods involve stratifying or segmenting the predictor space into a number of simple regions.

- The splitting rules can be summarized in a tree. That’s why they are called decision tree methods or tree-based methods.

Example: Predicting Baseball Players’ Salaries Using Regression Tree

- Years: the number of years that a player has played in major leagues
- Hits: the number of hits that a player made in the previous year



Red: above 75% quantile

Orange: between 50% and 75% quantiles

Green: between 25% and 50% quantiles

Blue: below 25% quantiles

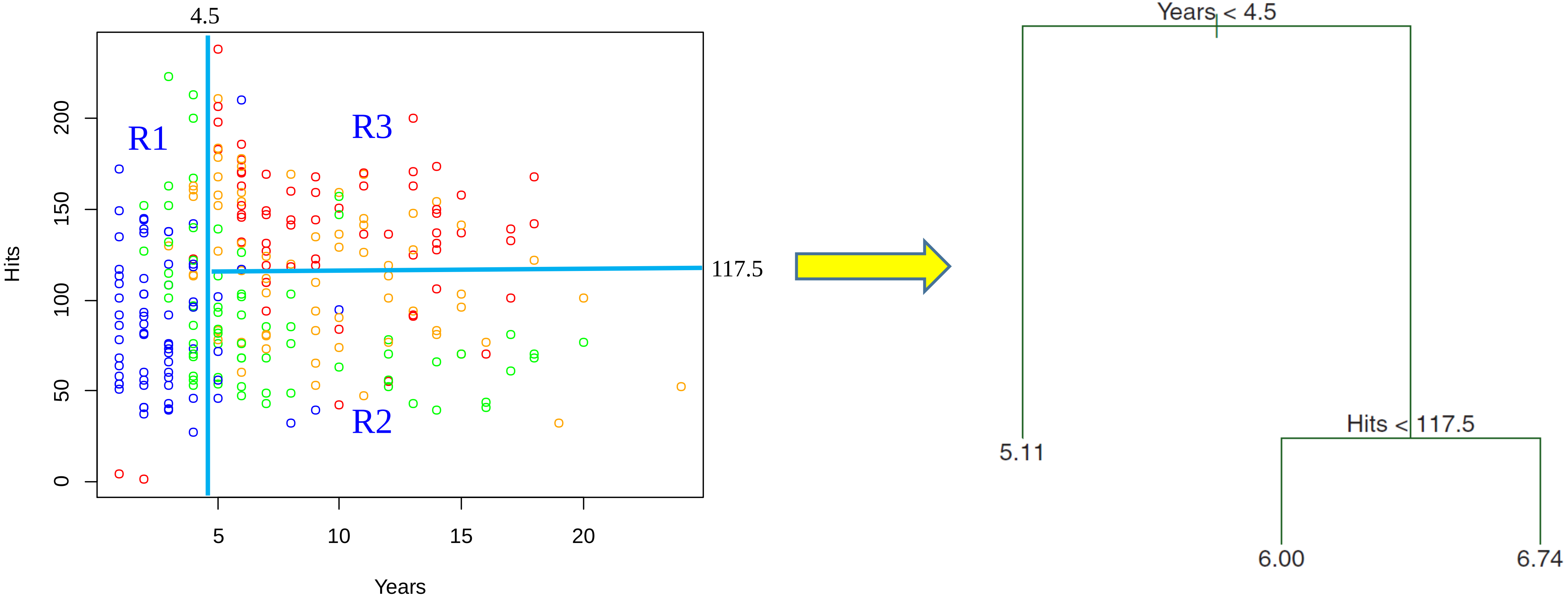
R1 = {X | Years < 4.5}

R2 = {X | Years >= 4.5, Hits < 117.5}

R3 = {X | Years >= 4.5, Hits >= 117.5}

Example: Predicting Baseball Players' Salaries Using Regression Tree

- Construct a regression tree

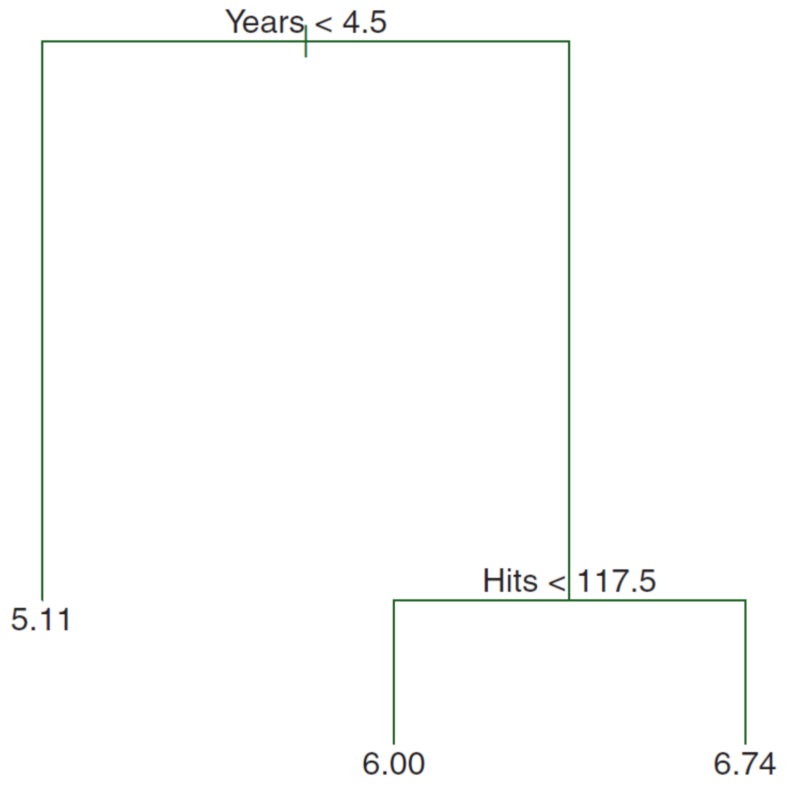


Salary (in $1000) is log transformed; 5.11 is the mean of log(salary) in R1 region.

Predicted salary for a player with more than 5 years and at least 118 hits last year = $1,000*exp(6.74) = $845,561

Interpreting Regression Trees Is Easy

- Years is the most important factor in determining Salary, and players with less experience earn lower salaries than more experienced players.
- Given that a player is less experienced, the number of hits that he made in the previous year seems to play little role in his salary.
- But among players who have been in the major leagues for five or more years, the number of hits made in the previous year does affect salary, and players who made more hits last year tend to have higher salaries.



Building A Regression Tree

- Step 1: We divide the predictor space – that is, the set of possible values for $X_1, X_2, \ldots, X_p$ – into J distinct and non-overlapping regions, $R_1, R_2, \ldots, R_J$.

- Step 2: For every observation that falls into the region R_j, we make the same prediction, which is simply the mean of the response values for the training observations in R_j.

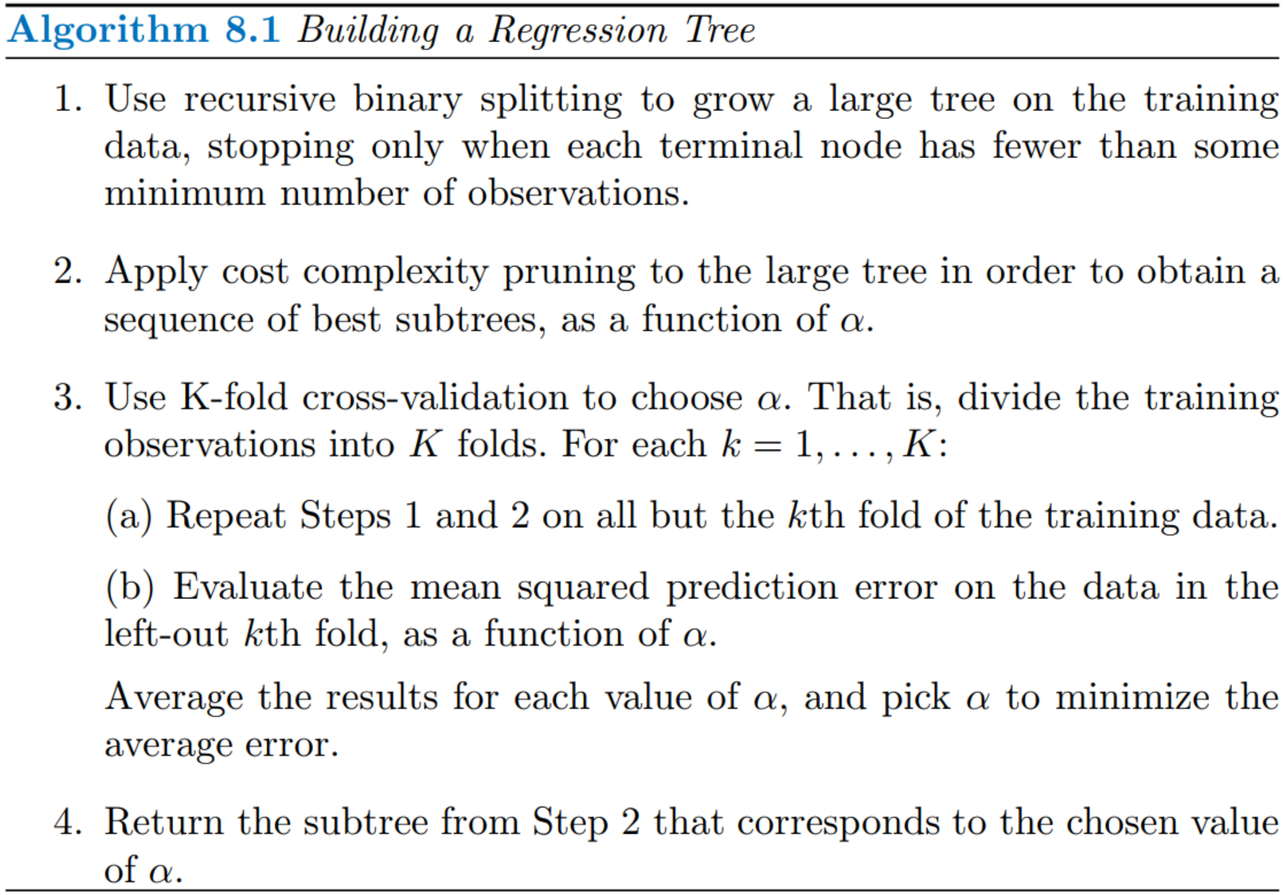
Pruning Tree

- A tree with many terminal nodes tends to over-fit data (low bias, high variance).

- Pruning a tree may improve the prediction performance of the tree by having a better tradeoff between bias and variance.

- Pruned tree has a better interpretation as a smaller set of decisions rules are generated from the pruned tree.

Target: minimize $$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T|$$



Algorithm 8.1 *Building a Regression Tree*

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.
2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α.
3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \dots, K$:
   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.
   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.
   Average the results for each value of α, and pick α to minimize the average error.
4. Return the subtree from Step 2 that corresponds to the chosen value of α.

Decision Tree Algorithms

- There are a couple of decision tree algorithms
  - ID3, C4.5, C5, CART, CHAID, M5 etc.

- These decision tree algorithms mainly differ on
  - Splitting criteria: which variable, what value, etc.
  - Stopping criteria: when to stop building the tree
  - Pruning (generalization method): Pre-pruning versus post-pruning

Growing A Classification Tree

- A *classification tree* is very similar to a regression tree, except that we need to predict a qualitative response rather than a continuous one.
- For each region (or node) we predict the most commonly occurring class among the training data in the region (majority vote).
- Growing a decision tree is similar as growing a regression tree, except that minimizing classification error rate rather than RSS is the objective.
- Alternative criteria to classification error include Gini index and cross-entropy. The two alternative measures are more sensitive to node purity.

Trees Vs. Linear Models

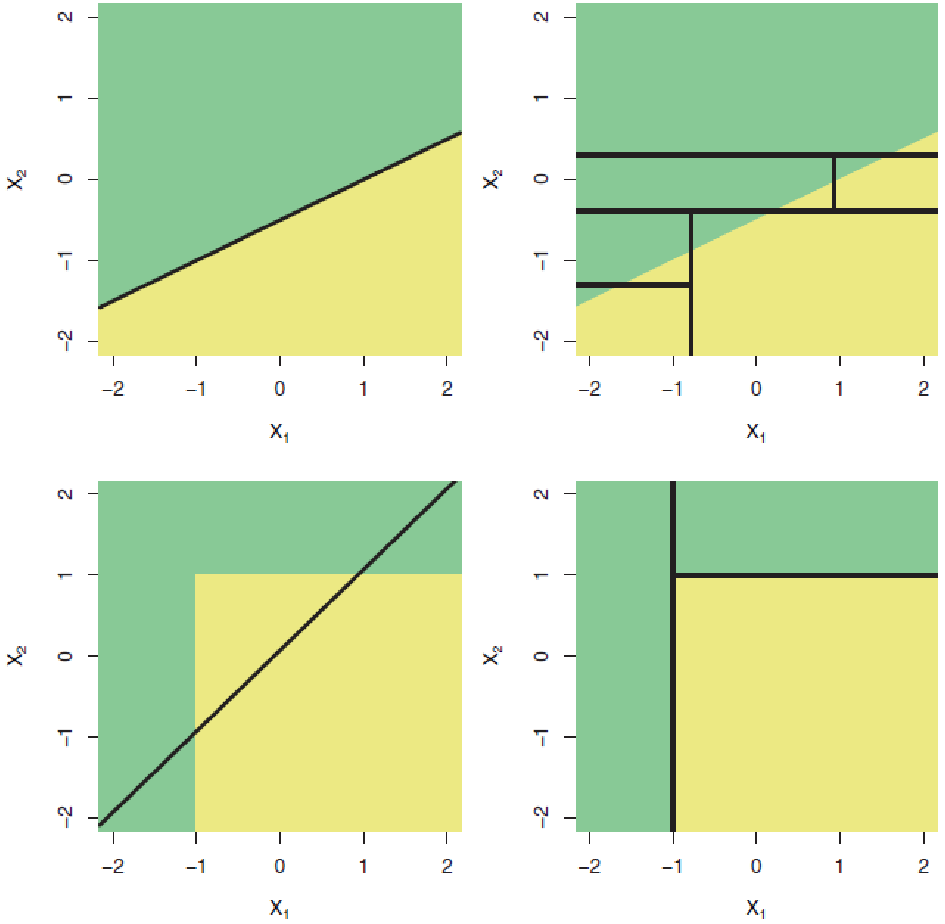
- Linear regression model

$$f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j$$

- Regression tree model

$$f(X) = \sum_{m=1}^{M} c_m \cdot 1_{(X \in R_m)}$$

Top Row: True linear boundary, linear regression works well



Bottom Row: True nonlinear boundary, decision trees are preferred

Summary of Decision Trees

- Advantages
  - Trees are very easy to explain to people. In fact, they are even easier to explain than linear regression!
  - Some people believe that decision trees more closely mirror human decision-making than do the regression and classification approaches.
  - Trees can be displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).
  - Trees can easily handle qualitative predictors without the need to create dummy variables.

- Disadvantages
  - Trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches.
  - Additionally, trees can be very non-robust. In other words, a small change in the data can cause a large change in the final estimated tree.

AGENDA

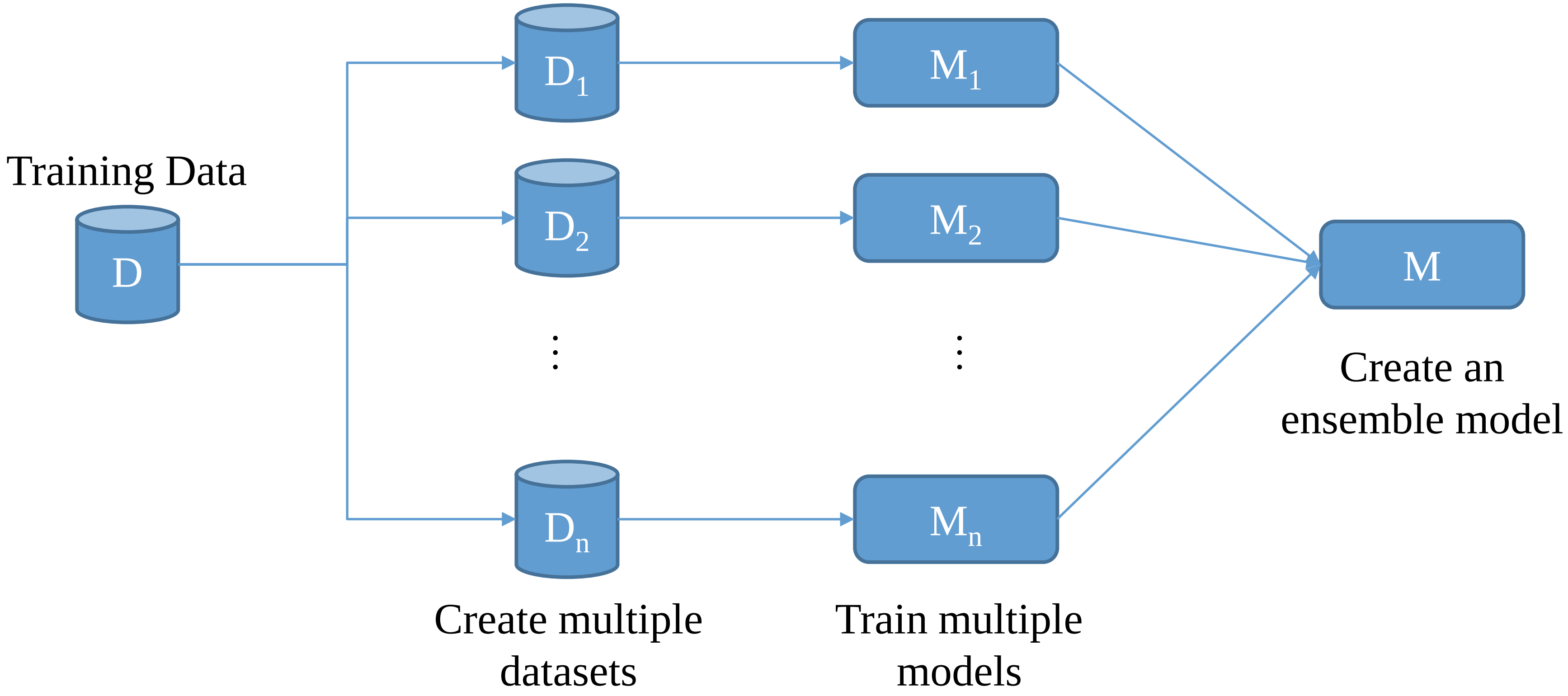
- Regression and Classification Trees
- Ensemble Learning and Random Forests

Improving Decision Trees

- Decision trees usually do not have the same level of predictive accuracy as other supervised machine learning approaches.
- Trees can be very non-robust. A small change in the data can cause a large change in the final estimated tree (large variance).
- To reduce the variance of decision trees, we can aggregate many decision trees to form a single classifier.
- By having a better trade-off between bias and variance, predictive performance can be substantially improved.

Ensemble Learning Methods

- Like the “wisdom of crowd”, ensemble learning methods:
  - Train a collection of simple or weak learning methods;
  - Then combine their results to get a single and better algorithm.



Basic Types of Ensemble

- Bagging (bootstrap aggregating): train learners in parallel on different samples of the data, then combine by voting (classification) or by averaging (regression).

- Stacking: combine model outputs using a second-stage learner like linear regression.

- Boosting: train learners on the filtered output of other learners.

Bagging Trees

- Bagging trees is an ensemble of n_{tree} decision trees by bagging method:
  - Step 1: Draw n_{tree} bootstrap samples from the original data with replacement;
  - Step 2: For each of the bootstrap samples, grow an unpruned decision tree with each node split among all predictors;
  - Step 3: Then combine the n_{tree} decision trees by aggregation:
    - For classification, use majority votes;
    - For regression, use average.

Bagging Trees Vs. Decision Tree

- Bagging generates a large number of decision trees;
- Bagging improves prediction accuracy at the expense of interpretability.

Random Forests

- Random forests (RF) is an extension of bagging trees. RF usually has a better performance than bagging trees.

- Random forests is an ensemble of n_{tree} decision trees by bagging method:
  - Step 1: Draw n_{tree} bootstrap samples from the original data with replacement;
  - Step 2: For each of the bootstrap samples, grow an unpruned decision tree with the following modification:
    - At each node, instead of choosing the best split among all predictors, randomly sample m_{try} predictors;
    - Choose the best split from the m_{try} predictors;
  - Step 3: Then combine the n_{tree} decision trees by aggregation:
    - For classification, use majority votes;
    - For regression, use average.

Random Forests

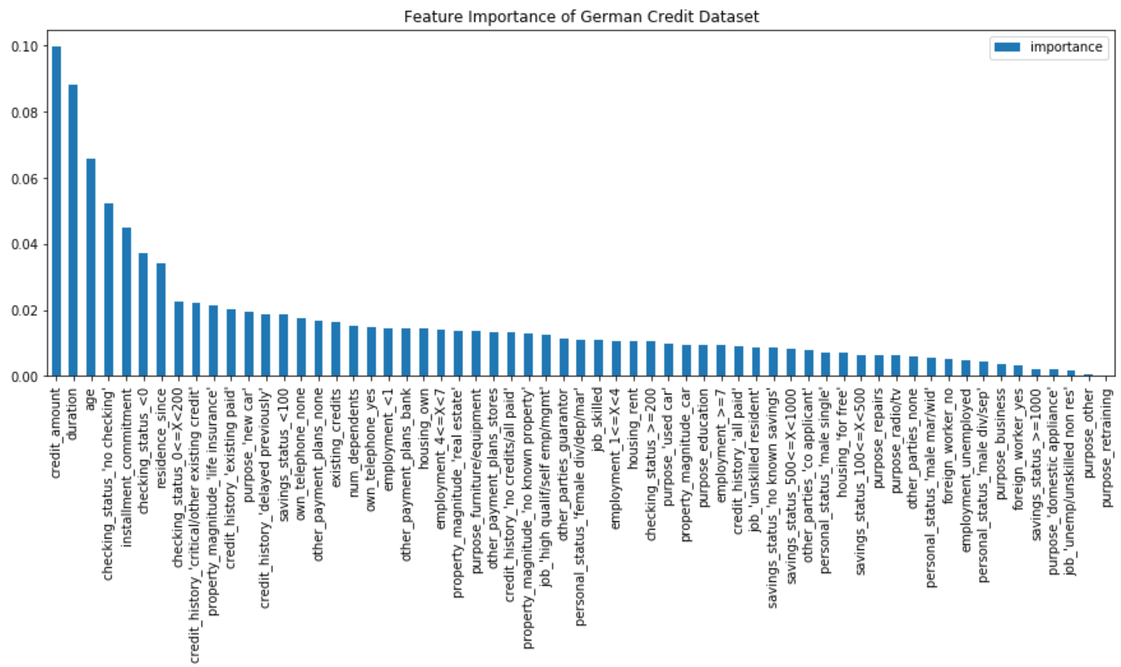
- The principle of diversity: Need to grow different rather than similar decision trees, in order to get the "wisdom of crowd".

- The diversity principle is implemented by:
  - Draw n_{tree} bootstrap samples from the original data with replacement;
  - When growing a decision tree, randomly choose m_{try} predictors to find the best split for each node.

n_{tree} and m_{try} are two hyperparameters of Random Forests.

Variable/Feature Importance

- Random forests provide the importance score for all predictors.
- The importance of a feature measures how much the feature can help reduce impurity of the data.
- Thus, random forests can be used to select features for other algorithms.
  - In practice, many competitions use random forests for feature selection.

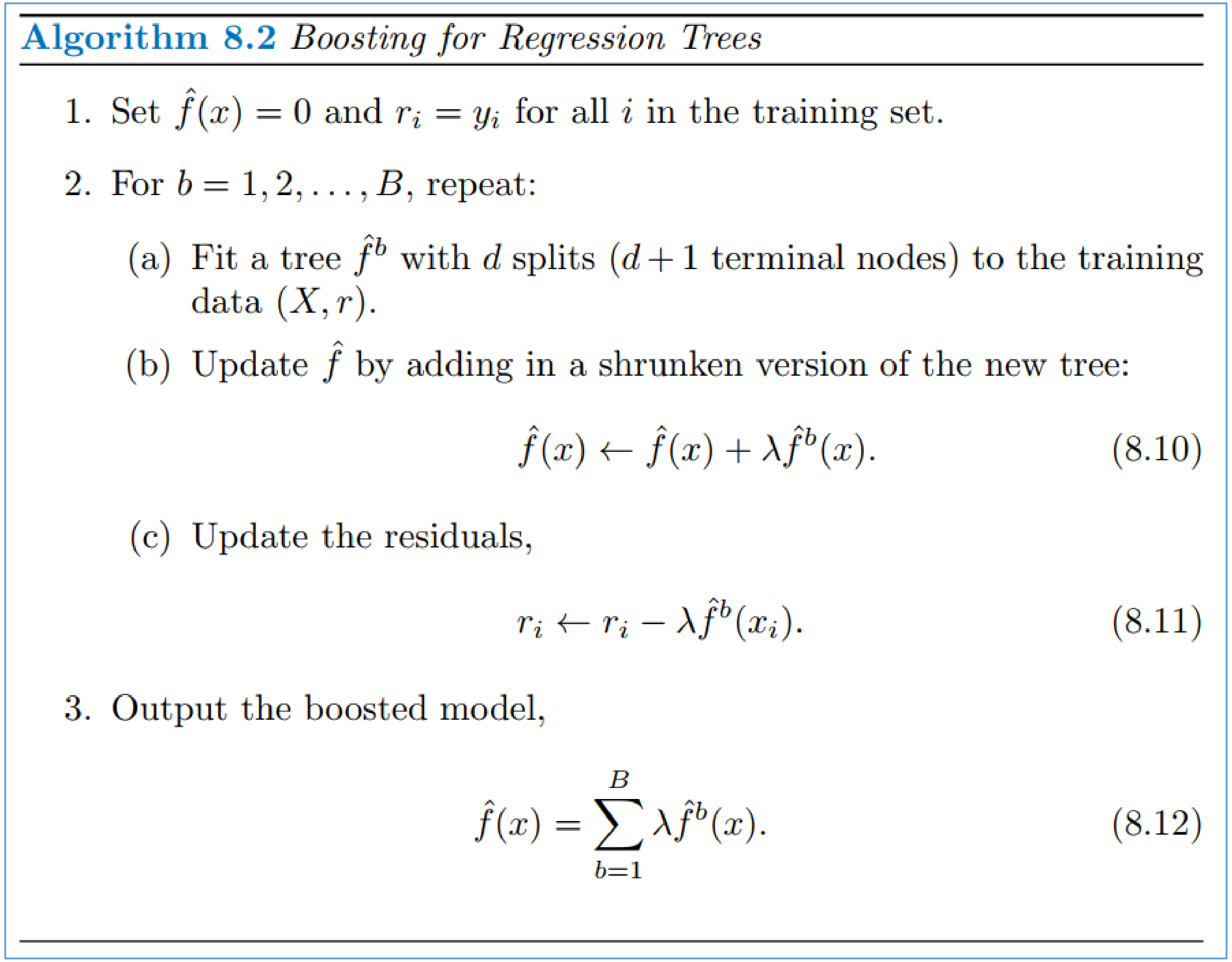


Boosting

- Fit a decision tree using outcome Y;
- Fit a small decision tree to the current residual;
- Repeat the small decision tree building to slowly boost/improve the current model.

No bootstrap sampling involved.

A famous algorithm is called GBM (Gradient Boosting Machine).



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:
            $\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x).$    (8.10)
    (c) Update the residuals,
            $r_i \leftarrow r_i - \lambda \hat{f}^b(x_i).$    (8.11)
3. Output the boosted model,
        $\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x).$    (8.12)
```

Q & A