Random Forests and Adaptive Nearest Neighbors

Work by Yi Lin, Yongho Jeon (Paper Review)

Jiexin Duan

Department of Statistics
Purdue University

August 16, 2016

Background

- Random forests is an ensemble learning method for classification and regression that constructs a number of randomized decision trees during the training phase and predicts by averaging the results.
- Random forests can be used to deal with big data and high-dimensional models (Sparsity). It is widely used in bioinformatics, survival analysis, quantile regression, ecology, etc.

Decision Trees

- Decision trees can be applied to both regression and classification problems.
- Regression trees are used to predict a quantitative response and classification trees are used to predict a qualitative response.

Regression Trees

- Given a training sample $D_n = (X_1, Y_1), \ldots, (X_n, Y_n)$ in $[0,1]^p \times R$, the objective is to estimate $m_n : [0,1]^p \to R$ of the function $m(x) = E[Y|X = x]$.
- p: dimension of predictors; n: size of the sample (training sets)
- X: input random vector used to estimate $\hat{Y}$, i.e. $m_n(X)$.
- In general trees, $X \in R^p$ rather than $[0,1]^p$ stated in this paper.

Example of Regression Trees

- "Hitters" data set
- Response: log salary of a baseball play
- Predictors: number of years that he has played in the major leagues; number of hits that he made in the previous year.

Example of Regression Trees

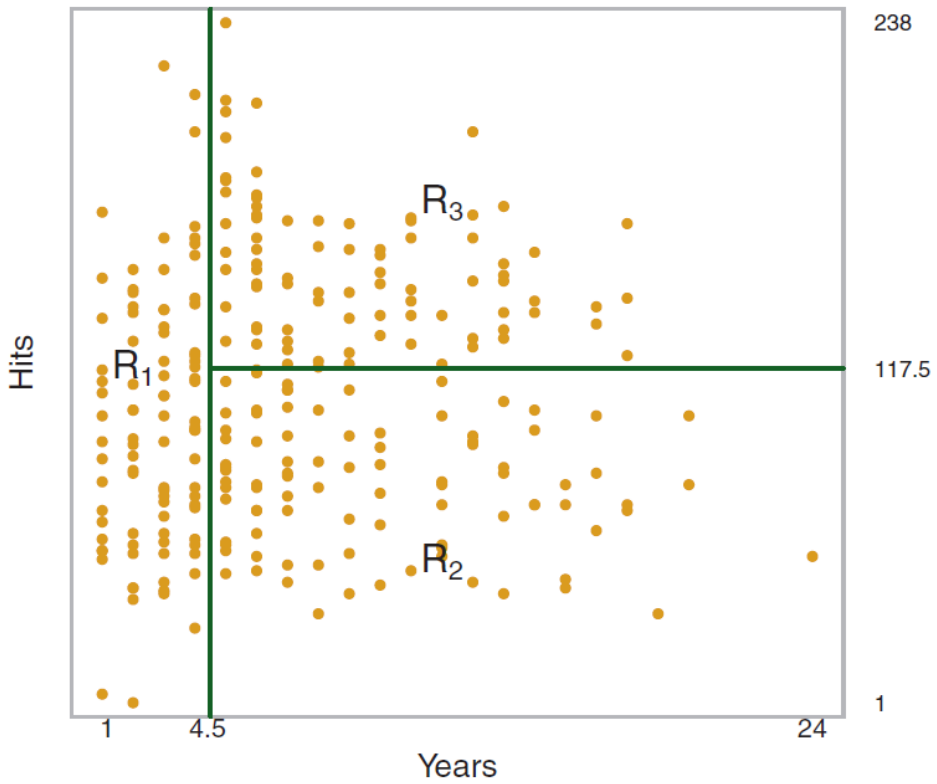

Figure : The three-region partition for "Hitters" data set from regression tree

Example of Regression Trees

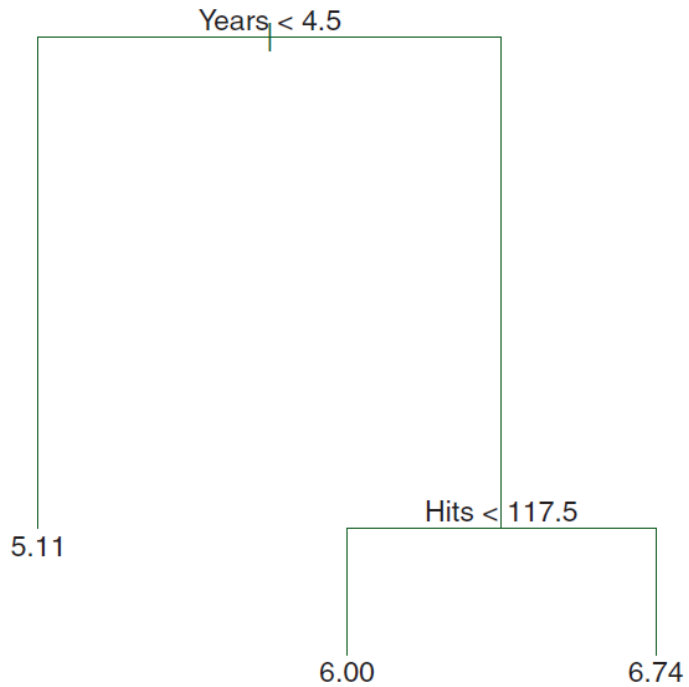

Figure : The regression tree for "Hitters" data set

Notations of Regression Trees

- $m_{try} \in \{1, \ldots, p\}$: number of pre-selected directions for splitting, when $m_{try} < p$, it can used to dealt with high-dimensional data.
- $a_n \in \{1, \ldots, n\}$: number of sampled data points in each tree. If $a_n < n$, it is sub-sampling, and can be used to deal with Big Data.
- $t_n \in \{1, \ldots, a_n\}$: number of leaves(cells) in each tree. If $t_n < a_n$, trees are not fully developed; if $t_n = a_n$, trees are fully developed, i.e. each leave has one number.
- A: a generic cell; $N_n(A)$: number of data points falling in A.
- j: direction of predictor of jth splitting; z: position of cut along the jth coordinate.
- $A_L = \left\{x \in A : x^{(j)} < z\right\}$, $A_R = \left\{x \in A : x^{(j)} \geq z\right\}$

CART-Split Criterion of Regression Trees

- $L_n(j,z) = \frac{1}{N_n(A)} \sum_{i=1}^{n} (Y_i - \overline{Y}_A)^2 1_{X_i \in A} - \frac{1}{N_n(A)} \sum_{i=1}^{n} (Y_i - \overline{Y}_{A_L} 1_{X_i^{(j)} < z} - \overline{Y}_{A_R} 1_{X_i^{(j)} \geq z})^2 1_{X_i \in A}$.
- $(j_n^*, z_n^*) = \underbrace{argmax}_{j \in M_{try}, (j,z) \in C_A} L_n(j,z)$.
- M_{try}: the set of selected predictors to build the tree.
- C_A: the set of all possible cuts in A.
- By CART-Splitting Criterion, we will build a regression tree with m_{try} predictors, a_n data points and t_n leaves.

Classification Trees

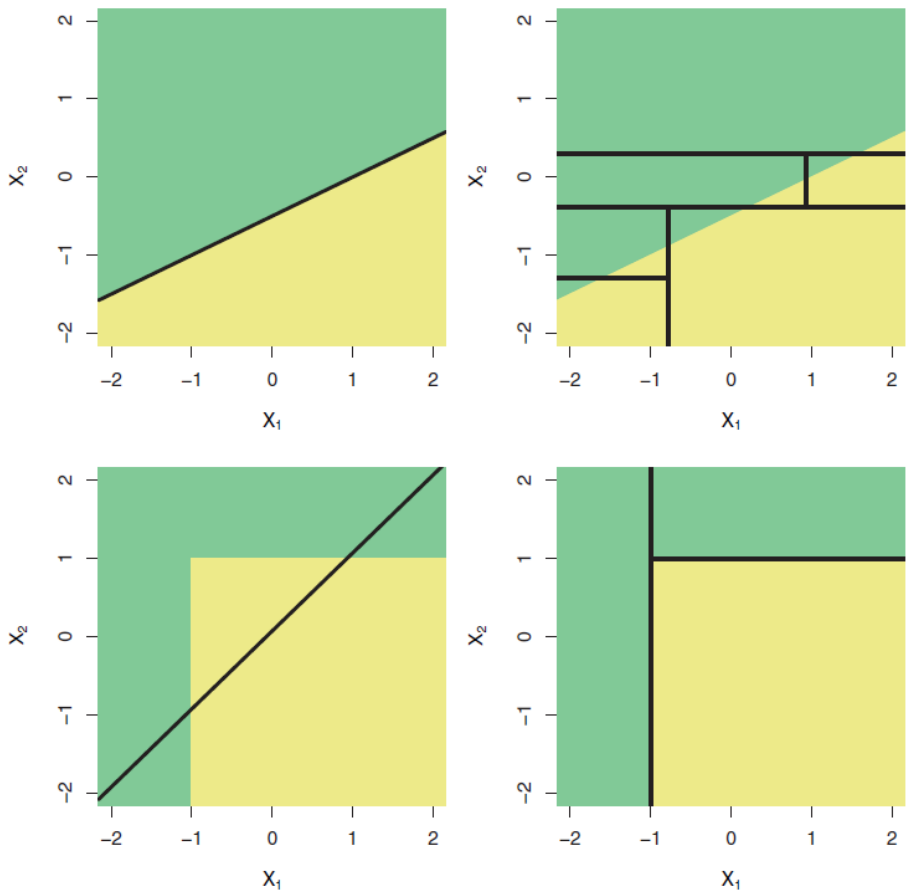

Figure : Decision Trees vs Linear Regression

Advantages and Disadvantages of Trees

- **Advantages:**
  1. Trees are very easy to explain to people.
  2. Decision trees are more close to human decision-making mode.
  3. Trees can be displayed graphically.
  4. Trees can easily handle qualitative predictors without the need to create dummy variables.
- **Disadvantages:**
  1. Trees generally don't have the same level of predictive accuracy as other approaches.
  2. Trees can be very non-robust. A small change in the data may cause a large change in the final estimated tree.
- So, we need random forests method to overcome these disadvantages.

Random Forests

- Random forests contains many trees by bootstrap M trees with replacement.
- $m_{M,n}(x; \Theta_1, \ldots, \Theta_M, D_n) = \frac{1}{M} \sum_{j=1}^{M} m_n(x; \Theta_j, D_n)$ (1)
- **Notations:**
- x: query point used to predict value of y.
- D_n: training sample.
- $\Theta_1, \ldots, \Theta_M$: independent random variables distributed as a generic random variable Θ.

Algorithm of Random Forests

Algorithm 1: Breiman's random forest predicted value at $\mathbf{x}$

Input: Training set $\mathcal{D}_n$, number of trees $M > 0$, $m_{\text{try}} \in \{1, \ldots, p\}$, $a_n \in \{1, \ldots, n\}$, $t_n \in \{1, \ldots, a_n\}$, and $\mathbf{x} \in [0,1]^p$.

Output: Prediction of the random forest at $\mathbf{x}$.

for $j = 1, \ldots, M$ **do**
  Select a_n points, without replacement, uniformly in $\mathcal{D}_n$.
  Set $\mathcal{P}_0 = \{[0,1]^p\}$ the partition associated with the root of the tree.
  For all $1 \leq \ell \leq a_n$, set $\mathcal{P}_\ell = \varnothing$.
  Set $n_{\text{nodes}} = 1$ and level $= 0$.
  while $n_{\text{nodes}} < t_n$ **do**
    if $\mathcal{P}_{\text{level}} = \varnothing$ **then**
      level $=$ level $+ 1$
    else
      Let A be the first element in $\mathcal{P}_{\text{level}}$.
      if A *contains exactly one point* **then**
        $\mathcal{P}_{\text{level}} \leftarrow \mathcal{P}_{\text{level}} \setminus \{A\}$
        $\mathcal{P}_{\text{level}+1} \leftarrow \mathcal{P}_{\text{level}+1} \cup \{A\}$

Algorithm of Random Forests

else
  Select uniformly, without replacement, a subset $\mathcal{M}_{\text{try}} \subset \{1, \ldots, p\}$ of cardinality m_{try}.
  Select the best split in A by optimizing the CART-split criterion along the coordinates in $\mathcal{M}_{\text{try}}$ (*see details below*).
  Cut the cell A according to the best split. Call A_L and A_R the two resulting cell.
  $\mathcal{P}_{\text{level}} \leftarrow \mathcal{P}_{\text{level}} \setminus \{A\}$
  $\mathcal{P}_{\text{level+1}} \leftarrow \mathcal{P}_{\text{level+1}} \cup \{A_L\} \cup \{A_R\}$
  $n_{\text{nodes}} = n_{\text{nodes}} + 1$
end
end
end
Compute the predicted value $m_n(\mathbf{x}; \Theta_j, \mathcal{D}_n)$ at $\mathbf{x}$ equal to the average of the Y_i's falling in the cell of $\mathbf{x}$ in partition $\mathcal{P}_{\text{level}} \cup \mathcal{P}_{\text{level+1}}$.
end
Compute the random forest estimate $m_{M,n}(\mathbf{x}; \Theta_1, \ldots, \Theta_M, \mathcal{D}_n)$ at the query point $\mathbf{x}$ according to (1).

Figure : Breiman's random forest predicted value at x

Several Types of Random Forests

- Different types of randomness of trees:
  1. Bootstrap.
  2. Selection without replacement.
  3. Sub-sampling.
  4. Selection from original sample points.

Several Types of Random Forests

- Different types of splitting schemes:
  1. Non-adaptive splitting schemes (i.e. splitting independent of response Ys), eg, **Purely random splitting**: for each internal node, randomly select a variable and a cut-point on that variable for all splitting steps.
  2. Adaptive splitting schemes (i.e. splitting dependent on response Ys), eg, **Random input selection**: at each node, a small group of F input variables are randomly selected, the best split is searched for these F input variables. If F=1, we also call it **Random side selection**.

Several Types of Random Forests

- Comments:
  There are many different types of random forests, but they have similar essence under the view of k-PNNs.

Problem Setup

Consider independent and identically distributed observations $\{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ of a random pair $(\mathbf{X}, Y)$. Here $\mathbf{X} = (X^{(1)}, \ldots, X^{(d)}) \in \mathbb{R}^d$ is the input vector and $Y \in \mathbb{R}$ is the response variable. We wish to estimate the regression function, $g(\mathbf{x}) = E(Y|\mathbf{X} = \mathbf{x})$. For any point $\mathbf{x}_0 \in \mathbb{R}^d$, the mean squared error (MSE) at $\mathbf{x}_0$ of an estimator $\hat{g}(\mathbf{x}_0)$ of $g(\mathbf{x}_0)$ is

$$\begin{aligned} MSE[\hat{g}(\mathbf{x}_0)] &= E[\hat{g}(\mathbf{x}_0) - g(\mathbf{x}_0)]^2 \\ &= \left[E\big(\hat{g}(\mathbf{x}_0) - g(\mathbf{x}_0)\big)\right]^2 + \text{var}(\hat{g}(\mathbf{x}_0)) \\ &= \text{bias}^2 + \text{variance}. \end{aligned}$$

The integrated MSE is $IMSE(\hat{g}) = E_{\mathbf{X}} MSE[\hat{g}(\mathbf{X})]$.

Definition of k-NNs

- Given a distance metric, a k-NN (Nearest Neighbors) method estimates $g(x_0)$ by looking at the k sample points that are closest to x_0:

$$\hat{g}(x_0) = \sum_{i=1}^{n} w_i y_i,$$

$$where\ w_i = \frac{1}{k}\ for\ k - NNs\ of\ x_0; w_i = 0\ otherwise$$

Definition of Voting Points

We start by looking at random forests locally at a target point $\mathbf{x}_0$. Given the training data $\{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$, at a target point $\mathbf{x}_0$, the prediction from the mth randomized tree is $\sum_{i=1}^{n} W_{im} y_i$, with the weight W_{im} as $1/k_m$ if $\mathbf{x}_i$ is among the k_m sample points in the terminal node containing the target point $\mathbf{x}_0$ and 0 otherwise. Averaging over M trees, the random forest prediction at $\mathbf{x}_0$ is $\sum_{i=1}^{n} \bar{W}_i y_i$, with $\bar{W}_i = (1/M) \sum_{m=1}^{M} W_{im}$. Therefore, the random forest can be viewed as a weighted average of the y_i's. Because $\sum_{i=1}^{n} W_{im} = 1$, we have

$$\sum_{i=1}^{n} \bar{W}_i = 1. \tag{1}$$

- **Voting points** are the sample points with positive weights.

Definition of Hyperrectangle

- A hyperrectangle defined by two points a and b is a hyperrectangle with the two points as opposite vertices.
- Eg: If d=2, this is a square defined by two opposite vertices.

Definition of Monotone Distance Measure

- For any two points a and b, any point c in the hyperrectangle defined by a and b is closer to a than b is. Any distance measure in Euclidean space with this property is called **Monotone distance measure**.
- Interpretations: Under this metric, all points in a hyperrectangle are nearer to one vertex.

Definition of k-PNNs

Definition 1. A sample point $\mathbf{x}_i$ is called a *k-potential nearest neighbor* (*k*-PNN) to a target point $\mathbf{x}_0$ if there exists a monotone distance metric under which $\mathbf{x}_i$ is among the k closest to $\mathbf{x}_0$ among all of the sample points.

Properties of k-PNNs

- Proposition 0:

  Therefore, any k-PNN is a k-NN under a suitably chosen monotone metric. The number of k-PNNs is typically much larger than k and depends on the number and configuration of the sample points.

- Interpretations:
  1. k-PNNs is a special type of k-NNs.
  2. The number and distribution of sample points will affect number of k-PNNs.

Properties of k-PNNs

Proposition 1. A sample point $\mathbf{x}_i$ is a k-PNN to $\mathbf{x}_0$ if and only if there are fewer than k sample points other than $\mathbf{x}_i$ in the hyperrectangle defined by $\mathbf{x}_0$ and $\mathbf{x}_i$.

- Interpretations:
  All points in the same terminal node as x_0 should be k-PNNs.

Relationship of Random Forest and k-PNNs

- Proposition 2. For random forests with terminal node size k, the voting points for a target point x_0 belong to the set of k-PNNs of x_0 regardless of the splitting scheme used.
- Interpretations:
  1. Only k-PNNs can become a voting points.
  2. We can view random forests as a weighted k-PNNs method.
  3. The difference for different types of random forests is how to assign weights to all k-PNNs.

Properties of Expected Number of k-PNNs

Consider a random sample of n points $\{\mathbf{x}_i, i = 1, \ldots, n\}$ from a density function $f(\mathbf{x})$ supported on $[0, 1]^d$. Let $A_k(n, d, \mathbf{x}_0, f)$ denote the expected number of k-PNNs of a fixed point $\mathbf{x}_0 \in [0, 1]^d$. For example, $A_k(n, d, \mathbf{0}, 1)$ is the expected number of k-PNNs of the origin $\mathbf{0} = (0, \ldots, 0)$ among n uniform random points in $[0, 1]^d$.

Properties of Expected Number of k-PNNs

Theorem 1. $A_k(n, d, \mathbf{0}, 1) = \{k(\log n)^{d-1}/(d-1)!\}[1+o(1)]$, as $n \to \infty$.

Lemma 1. $A_k(n, d, \mathbf{0}, 1) \le A_k(n, d, \mathbf{x}_0, 1) \le 2^d A_k(n, d, \mathbf{0}, 1)$, for any $\mathbf{x}_0 \in [0, 1]^d$.

Theorem 2. Assume that the density $f(\mathbf{x})$ is bounded away from 0 and infinity in $[0, 1]^d$. Then there exists $0 < \Lambda_1 \le \Lambda_2$, such that

$$\Lambda_1 k(\log n)^{d-1} \le A_k(n, d, \mathbf{x}_0, f) \le \Lambda_2 k(\log n)^{d-1}$$

for any $\mathbf{x}_0 \in [0, 1]^d$ and n; that is, the expected number of k-PNNs is of order $k(\log n)^{d-1}$.

Properties of k-PNNs

- Interpretations:
  1. We can get asymptotic properties for expected number of k-PNNs.
  2. We can get the upper bound and lower bound of expected number of k-PNNs.
  3. We can get the order of expected number of k-PNNs.

Effect of Maximum Terminal Node Size k

Theorem 3. Consider the regression problem $Y = g(\mathbf{X}) + \epsilon$ with $E(\epsilon) = 0$ and $\text{var}(\epsilon) = \sigma^2$. Assume that $\mathbf{X}$ is distributed in $[0, 1]^d$ and that its density is bounded away from 0 and infinity in $[0, 1]^d$. Let $\{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ be a random sample. Consider an estimator $\hat{g}$ resulting from a nonadaptive random forest with terminal node size k. There exists $\Lambda_3 > 0$ such that for any n,

$$MSE[\hat{g}(\mathbf{x}_0)] \geq \Lambda_3 k^{-1} (\log n)^{-(d-1)}, \qquad \forall \mathbf{x}_0 \in [0, 1]^d,$$

and

$$IMSE(\hat{g}) \geq \Lambda_3 k^{-1} (\log n)^{-(d-1)}.$$

Interpretations of Theorem 3

- 1. Theorem 3 states that a lower bound to the rate of convergence of the MSE of random forests with nonadaptive splitting schemes is $k^{-1}(logn)^{-(d-1)}$.
- 2. The lower bound of MSE decreases as n,k,d increases.
- 3. Lower bound may not be achieved for some splitting schemes (stated in the paper).
- 4. Theorem 3 also applies to adaptive random forests.

Example of Regression Random Forests

- 1. n=1000 (both for training and testing data).
- 2. d=4 or 10.
- 3. $0 \leq X^{(1)} \leq 100$, $40\pi \leq X^{(2)} \leq 560\pi$, $0 \leq X^{(3)} \leq 1$, $1 \leq X^{(4)} \leq 11$, $X^{(5)}$ *to* $X^{(10)}$ are six uniform noises.

$$y = \left[(X^{(1)})^2 + (X^{(2)}X^{(3)} - \frac{1}{X^{(2)}X^{(4)}})^2\right]^{0.5} + \epsilon,\ \epsilon \sim N(0, 125^2)$$

- 4. M=100 trees.
- 5. Splitting schemes: random input selection with F=3 (i.e. each time 3 Xs are selected).

Example of Regression Random Forests

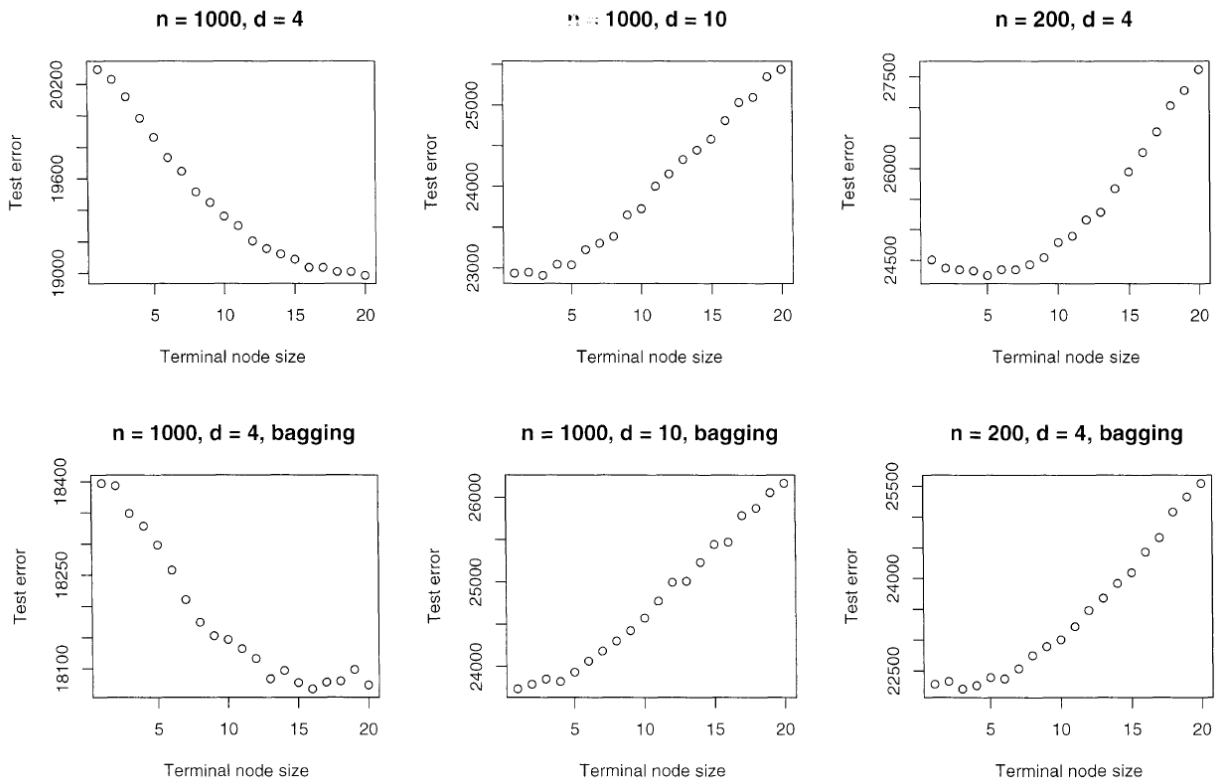

Figure : The Prediction Error of Regression Random Forests Varies with Terminal Node Size k in the Above Example.

Example of Classification Random Forests

- 1. n=1000 (500 for both classes, both for training and testing data);
- 2. d=2.
- 3. Two classes have distributions of

$$N\left[\begin{array}{c} 0 \\ 0 \end{array}, \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right] \text{ and } N\left[\begin{array}{c} 1 \\ 1 \end{array}, \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right]$$

- 4. M=100 trees.
- 5. Splitting scheme: random input selection with F=1 (i.e. random side selection).

Example of Classification Random Forests

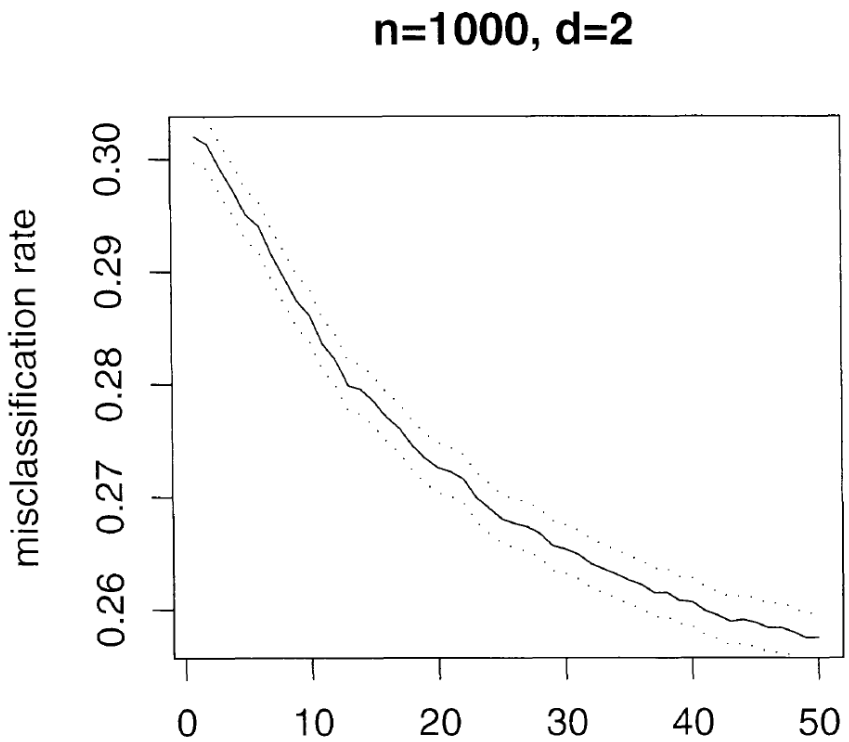

Figure : The Misclassification Rate of Classification Random Forests Varies with Terminal Node Size k in the Above Example. The dotted lines represent the standard error estimate.

Conclusions for Terminal Node Size K

- 1. In many empirical situations, the datasets are high-dimensional, with the sample size n small relative to the dimension d. In such high-dimension situations, the largest trees are usually optimal (i.e. k is small).
- 2. In other situations, it is better to tune the tree size for best performance (i.e. k is large).

Definition of Random Point Selection

- In **random point selection**, at current node, on each input variable, we randomly select a split point, and then choose the best split among these d splits.
- Interpretations:
  1. As all splitting schemes have same k-PNNs, the author designs a new splitting scheme that can compute much faster than random side selections.
  2. In random side selection, more important variables are more likely to be selected to split.
  3. Random side selection can be used both in linear and non-linear regressions.

Discussion

- **Advantages of k-PNNs**
  1. Can be applied to random forests with all kinds of splitting schemes and randomization of trees.
  2. Avoid complicated probability calculation of each terminal node.
  3. Can be the foundation of many kinds of random forests such as quantile random forests and survival random forests.
- **Disadvantages k-PNNs**
  1. Stopping Criterion can only be terminal size not larger than k.
  2. Monotone Distance Metric is not easy to deal with as Euclidean Metric.
  3. Each terminal node need to be hyperrectangle.

Future Work

- Prove some properties of random forests based on the idea of k-PNNs.
- Apply the idea of k-PNNs to explore other kinds of random forests such as quantile regression forests and random survival forests.
- Design the randomness of trees and splitting schemes which can be faster.

Selected Reference

- Breiman, L., *Random Forests*. Machine Learning, 45:5-32, 2001.
- Short, R., Fukunaga, K. *The optimal Distance Measure for Nearest-Neighbors Classification*. IEEE Transactions on Information Theory, 27:655-665, 1981.
- Myles, J., Hand, D. *The Multiclass Metric Problem in Nearest-Neighbor Classification*. Pattern Recognition, 23:1291-1297, 1990.
- Gyorfi, M., Krzyzak, A., and Walk, H. *A Distribution-Free Theory of Nonparametric Regression*. Springer, New York, 2002.

THANK YOU!