Random forest

Random forests or **random decision forests** are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.[1][2] Random decision forests correct for decision trees' habit of overfitting to their training set.[3]:587–588

The first algorithm for random decision forests was created by Tin Kam Ho[1] using the random subspace method,[2] which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.[4][5][6]

An extension of the algorithm was developed by Leo Breiman[7] and Adele Cutler,[8] and "Random Forests" is their trademark.[9] The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho[1] and later independently by Amit and Geman[10] in order to construct a collection of decision trees with controlled variance.

Contents

History

The general method of random decision forests was first proposed by Ho in 1995,[1] who established that forests of trees splitting with oblique hyperplanes, if randomly restricted to be sensitive to only selected feature dimensions, can gain accuracy as they grow without suffering from overtraining. A subsequent work along the same lines[2] concluded that other splitting methods, as long as they are randomly forced to be insensitive to some feature dimensions, behave similarly. Note that this observation of a more complex classifier (a larger forest) getting more accurate nearly monotonically is in sharp contrast to the common belief that the complexity of a classifier can only grow to a certain level of accuracy before being hurt by overfitting. The explanation of the forest method's resistance to overtraining can be found in Kleinberg's theory of stochastic discrimination.[4][5][6]

The early development of Breiman's notion of random forests was influenced by the work of Amit and Geman[10] who introduced the idea of searching over a random subset of the available decisions when splitting a node, in the context of growing a single tree. The idea of random subspace selection from Ho[2] was also influential in the design of random forests. In this method a forest of trees is grown, and variation among the trees is introduced by projecting the training data into a randomly chosen subspace before fitting each tree or each node. Finally, the idea of randomized node optimization, where the decision at each node is selected by a randomized procedure, rather than a deterministic optimization was first introduced by Dietterich.[11]

The introduction of random forests proper was first made in a paper by Leo Breiman.[7] This paper describes a method of building a forest of uncorrelated trees using a CART like procedure, combined with randomized node optimization and bagging. In addition, this paper combines several ingredients, some previously known and some novel, which form the basis of the modern practice of random forests, in particular:

1. Using out-of-bag error as an estimate of the generalization error.
2. Measuring variable importance through permutation.

The report also offers the first theoretical result for random forests in the form of a bound on the generalization error which depends on the strength of the trees in the forest and their correlation.

Algorithm

Preliminaries: decision tree learning

Decision trees are a popular method for various machine learning tasks. Tree learning "come[s] closest to meeting the requirements for serving as an off-the-shelf procedure for data mining", say Hastie *et al.*, "because it is invariant under scaling and various other transformations of feature values, is robust to inclusion of irrelevant features, and produces inspectable models. However, they are seldom accurate".[3]:352

In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance.[3]:587–588 This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

Tree bagging

The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging, to tree learners. Given a training set $X = x_1, ..., x_n$ with responses $Y = y_1, ..., y_n$, bagging repeatedly (B times) selects a random sample with replacement of the training set and fits trees to these samples:

For b = 1, ..., B:

1. Sample, with replacement, n training examples from X, Y; call these X_b, Y_b.
2. Train a classification or regression tree f_b on X_b, Y_b.

After training, predictions for unseen samples x' can be made by averaging the predictions from all the individual regression trees on x':

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} f_b(x')$$

or by taking the majority vote in the case of classification trees.

This bootstrapping procedure leads to better model performance because it decreases the variance of the model, without increasing the bias. This means that while the predictions of a single tree are highly sensitive to noise in its training set, the average of many trees is not, as long as the trees are not correlated. Simply training many trees on a single training set would give strongly correlated trees (or even the same tree many times, if the training algorithm is deterministic); bootstrap sampling is a way of de-correlating the trees by showing them different training sets.

Additionally, an estimate of the uncertainty of the prediction can be made as the standard deviation of the predictions from all the individual regression trees on x':

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B}(f_b(x') - \hat{f})^2}{B-1}}.$$

The number of samples/trees, B, is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set. An optimal number of trees B can be found using cross-validation, or by observing the *out-of-bag error*: the mean prediction error on each training sample x_i, using only the trees that did not have x_i in their bootstrap sample.[12] The training and test error tend to level off after some number of trees have been fit.

From bagging to random forests

The above procedure describes the original bagging algorithm for trees. Random forests differ in only one way from this general scheme: they use a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. This process is sometimes called "feature bagging". The reason for doing this is the correlation of the trees in an ordinary bootstrap sample: if one or a few features are very strong predictors for the response variable (target output), these features will be selected in many of the B trees, causing them to become correlated. An analysis of how bagging and random subspace projection contribute to accuracy gains under different conditions is given by Ho.[13]

Typically, for a classification problem with p features, $\sqrt{p}$ (rounded down) features are used in each split.[3]:592 For regression problems the inventors recommend $p/3$ (rounded down) with a minimum node size of 5 as the default.[3]:592

ExtraTrees

Adding one further step of randomization yields *extremely randomized trees*, or ExtraTrees. These are trained using bagging and the random subspace method, like in an ordinary random forest, but additionally the top-down splitting in the tree learner is randomized. Instead of computing the locally *optimal* feature/split combination (based on, e.g., information gain or the Gini impurity), for each feature under consideration, a random value is selected for the split. This value is selected from the feature's empirical range (in the tree's training set, i.e., the bootstrap sample).[14]

Properties

Variable importance

Random forests can be used to rank the importance of variables in a regression or classification problem in a natural way. The following technique was described in Breiman's original paper[7] and is implemented in the R package randomForest.[8]

The first step in measuring the variable importance in a data set $\mathcal{D}_n = \{(X_i, Y_i)\}_{i=1}^{n}$ is to fit a random forest to the data. During the fitting process the out-of-bag error for each data point is recorded and averaged over the forest (errors on an independent test set can be substituted if bagging is not used during training).

To measure the importance of the j-th feature after training, the values of the j-th feature are permuted among the training data and the out-of-bag error is again computed on this perturbed data set. The importance score for the j-th feature is computed by averaging the difference in out-of-bag error before and after the permutation over all trees. The score is normalized by the standard deviation of these differences.

Features which produce large values for this score are ranked as more important than features which produce small values. The statistical definition of the variable importance measure was given and analyzed by Zhu *et al.*[15]

This method of determining variable importance has some drawbacks. For data including categorical variables with different number of levels, random forests are biased in favor of those attributes with more levels. Methods such as partial permutations[16][17] and growing unbiased trees[18][19]can be used to solve the problem. If the data contain groups of correlated features of similar relevance for the output, then smaller groups are favored over larger groups.[20]

Relationship to nearest neighbors

A relationship between random forests and the k-nearest neighbor algorithm (k-NN) was pointed out by Lin and Jeon in 2002.[21] It turns out that both can be viewed as so-called *weighted neighborhoods schemes*. These are models built from a training set $\{(x_i, y_i)\}_{i=1}^{n}$ that make predictions $\hat{y}$ for new points x' by looking at the "neighborhood" of the point, formalized by a weight function W:

$$\hat{y} = \sum_{i=1}^{n} W(x_i, x')\, y_i.$$

Here, $W(x_i, x')$ is the non-negative weight of the i'th training point relative to the new point x' in the same tree. For any particular x', the weights for points x_i must sum to one. Weight functions are given as follows:

- In k-NN, the weights are $W(x_i, x') = \frac{1}{k}$ if x_i is one of the k points closest to x', and zero otherwise.
- In a tree, $W(x_i, x') = \frac{1}{k'}$ if x_i is one of the k' points in the same leaf as x', and zero otherwise.

Since a forest averages the predictions of a set of m trees with individual weight functions W_j, its predictions are

$$\hat{y} = \frac{1}{m} \sum_{j=1}^{m} \sum_{i=1}^{n} W_j(x_i, x')\, y_i = \sum_{i=1}^{n} \left(\frac{1}{m} \sum_{j=1}^{m} W_j(x_i, x') \right) y_i.$$

This shows that the whole forest is again a weighted neighborhood scheme, with weights that average those of the individual trees. The neighbors of x' in this interpretation are the points x_i sharing the same leaf in any tree j. In this way, the neighborhood of x' depends in a complex way on the structure of the trees, and thus on the structure of the training set. Lin and Jeon show that the shape of the neighborhood used by a random forest adapts to the local importance of each feature.[21]

Unsupervised learning with random forests

As part of their construction, random forest predictors naturally lead to a dissimilarity measure among the observations. One can also define a random forest dissimilarity measure between unlabeled data: the idea is to construct a random forest predictor that distinguishes the “observed” data from suitably generated synthetic data.[7][22] The observed data are the original unlabeled data and the synthetic data are drawn from a reference distribution. A random forest dissimilarity can be attractive because it handles mixed variable types very well, is invariant to monotonic transformations of the input variables, and is robust to outlying observations. The random forest dissimilarity easily deals with a large number of semi-continuous variables due to its intrinsic variable selection; for example, the "Addcl 1" random forest dissimilarity weighs the contribution of each variable according to how dependent it is on other variables. The random forest dissimilarity has been used in a variety of applications, e.g. to find clusters of patients based on tissue marker data.[23]

Variants

Instead of decision trees, linear models have been proposed and evaluated as base estimators in random forests, in particular multinomial logistic regression and naive Bayes classifiers.[24][25]

Kernel random forest

In machine learning, kernel random forests establish the connection between random forests and kernel methods. By slightly modifying their definition, random forests can be rewritten as kernel methods, which are more interpretable and easier to analyze.[26]

History

Leo Breiman[27] was the first person to notice the link between random forest and kernel methods. He pointed out that random forests which are grown using i.i.d. random vectors in the tree construction are equivalent to a kernel acting on the true margin. Lin and Jeon[28] established the connection between random forests and adaptive nearest neighbor, implying that random forests can be seen as adaptive kernel estimates. Davies and Ghahramani[29] proposed Random Forest Kernel and show that it can empirically outperform state-of-art kernel methods. Scornet[26] first defined KeRF estimates and gave the explicit link between KeRF estimates and random forest. He also gave explicit expressions for kernels based on centered random forest[30] and uniform random forest,[31] two simplified models of random forest. He named these two KeRFs Centered KeRF and Uniform KeRF, and proved upper bounds on their rates of consistency.

Notations and definitions

Preliminaries: Centered forests

Centered forest[30] is a simplified model for Breiman's original random forest, which uniformly selects an attribute among all attributes and performs splits at the center of the cell along the pre-chosen attribute. The algorithm stops when a fully binary tree of level k is built, where $k \in \mathbb{N}$ is a parameter of the algorithm.

Uniform forest

Uniform forest[31] is another simplified model for Breiman's original random forest, which uniformly selects a feature among all features and performs splits at a point uniformly drawn on the side of the cell, along the preselected feature.

From random forest to KeRF

Given a training sample $\mathcal{D}_n = \{(\mathbf{X}_i, Y_i)\}_{i=1}^n$ of $[0,1]^p \times \mathbb{R}$-valued independent random variables distributed as the independent prototype pair $(\mathbf{X}, Y)$, where $\mathbf{E}[Y^2] < \infty$. We aim at predicting the response Y, associated with the random variable $\mathbf{X}$, by estimating the regression function $m(\mathbf{x}) = \mathbf{E}[Y \mid \mathbf{X} = \mathbf{x}]$. A random regression forest is an ensemble of M randomized regression trees. Denote $m_n(\mathbf{x}, \Theta_j)$ the predicted value at point $\mathbf{x}$ by the j-th tree, where $\Theta_1, \ldots, \Theta_M$ are independent random variables, distributed as a generic random variable Θ, independent of the sample $\mathcal{D}_n$. This random variable can be used to describe the randomness induced by node splitting and the sampling procedure for tree construction. The trees are combined to form the finite forest estimate $m_{M,n}(\mathbf{x}, \Theta_1, \ldots, \Theta_M) = \frac{1}{M}\sum_{j=1}^{M} m_n(\mathbf{x}, \Theta_j)$. For regression trees, we have $m_n = \sum_{i=1}^{n} \frac{Y_i \mathbf{1}_{\mathbf{X}_i \in A_n(\mathbf{x}, \Theta_j)}}{N_n(\mathbf{x}, \Theta_j)}$, where $A_n(\mathbf{x}, \Theta_j)$ is the cell containing $\mathbf{x}$, designed with randomness Θ_j and dataset $\mathcal{D}_n$, and $N_n(\mathbf{x}, \Theta_j) = \sum_{i=1}^{n} \mathbf{1}_{\mathbf{X}_i \in A_n(\mathbf{x}, \Theta_j)}$.

Thus random forest estimates satisfy, for all $\mathbf{x} \in [0,1]^d$, $m_{M,n}(\mathbf{x}, \Theta_1, \ldots, \Theta_M) = \frac{1}{M}\sum_{j=1}^{M}\left(\sum_{i=1}^{n} \frac{Y_i \mathbf{1}_{\mathbf{X}_i \in A_n(\mathbf{x}, \Theta_j)}}{N_n(\mathbf{x}, \Theta_j)}\right)$. Random regression forest has two level of averaging, first over the samples in the target cell of a tree, then over all trees. Thus the contributions of observations that are in cells with a high density of data points are smaller than that of observations which belong to less populated cells. In order to improve the random forest methods and compensate the misestimation, Scornet[26] defined KeRF by

$$\tilde{m}_{M,n}(\mathbf{x}, \Theta_1, \ldots, \Theta_M) = \frac{1}{\sum_{j=1}^{M} N_n(\mathbf{x}, \Theta_j)} \sum_{j=1}^{M}\sum_{i=1}^{n} Y_i \mathbf{1}_{\mathbf{X}_i \in A_n(\mathbf{x}, \Theta_j)},$$

which is equal to the mean of the Y_i's falling in the cells containing $\mathbf{x}$ in the forest. If we define the connection function of the M finite forest as $K_{M,n}(\mathbf{x}, \mathbf{z}) = \frac{1}{M}\sum_{j=1}^{M} \mathbf{1}_{\mathbf{z} \in A_n(\mathbf{x}, \Theta_j)}$, i.e. the proportion of cells shared between $\mathbf{x}$ and $\mathbf{z}$, then almost surely we have $\tilde{m}_{M,n}(\mathbf{x}, \Theta_1, \ldots, \Theta_M) = \frac{\sum_{i=1}^{n} Y_i K_{M,n}(\mathbf{x}, \mathbf{x}_i)}{\sum_{\ell=1}^{n} K_{M,n}(\mathbf{x}, \mathbf{x}_\ell)}$, which defines the KeRF.

Centered KeRF

The construction of Centered KeRF of level k is the same as for centered forest, except that predictions are made by $\tilde{m}_{M,n}(\mathbf{x}, \Theta_1, \ldots, \Theta_M)$, the corresponding kernel function, or connection function is

$$K_k^{cc}(\mathbf{x}, \mathbf{z}) = \sum_{k_1, \ldots, k_d, \sum_{j=1}^d k_j = k} \frac{k!}{k_1! \cdots k_d!} \left(\frac{1}{d}\right)^k \prod_{j=1}^d \mathbf{1}_{\lceil 2^{k_j} x_j \rceil = \lceil 2^{k_j} z_j \rceil},$$
$$\text{for all } \mathbf{x}, \mathbf{z} \in [0,1]^d.$$

Uniform KeRF

Uniform KeRF is built in the same way as uniform forest, except that predictions are made by $\tilde{m}_{M,n}(\mathbf{x}, \Theta_1, \ldots, \Theta_M)$, the corresponding kernel function, or connection function is

$$K_k^{uf}(\mathbf{0}, \mathbf{x}) = \sum_{k_1, \ldots, k_d, \sum_{j=1}^d k_j = k} \frac{k!}{k_1! \ldots k_d!} \left(\frac{1}{d}\right)^k \prod_{m=1}^d \left(1 - |x_m| \sum_{j=0}^{k_m - 1} \frac{(-\ln |x_m|)^j}{j!}\right) \text{ for all } \mathbf{x} \in [0,1]^d.$$

Properties

Relation between KeRF and random forest

Predictions given by KeRF and random forests are close if the number of points in each cell is controlled:

> Assume that there exist sequences $(a_n), (b_n)$ such that, almost surely,
>
> $$a_n \leq N_n(\mathbf{x}, \Theta) \leq b_n \text{ and } a_n \leq \frac{1}{M} \sum_{m=1}^M N_n \mathbf{x}, \Theta_m \leq b_n.$$
>
> Then almost surely,
>
> $$|m_{M,n}(\mathbf{x}) - \tilde{m}_{M,n}(\mathbf{x})| \leq \frac{b_n - a_n}{a_n} \tilde{m}_{M,n}(\mathbf{x}).$$

Relation between infinite KeRF and infinite random forest

When the number of trees M goes to infinity, then we have infinite random forest and infinite KeRF. Their estimates are close if the number of observations in each cell is bounded:

> Assume that there exist sequences $(\varepsilon_n), (a_n), (b_n)$ such that, almost surely
>
> - $\mathbf{E}[N_n(\mathbf{x}, \Theta)] \geq 1$,
> - $\mathbf{P}[a_n \leq N_n(\mathbf{x}, \Theta) \leq b_n \mid \mathcal{D}_n] \geq 1 - \varepsilon_n/2$,
> - $\mathbf{P}[a_n \leq \mathbf{E}_\Theta[N_n(\mathbf{x}, \Theta)] \leq b_n \mid \mathcal{D}_n] \geq 1 - \varepsilon_n/2$,
>
> Then almost surely,
>
> $$|m_{\infty,n}(\mathbf{x} - \tilde{m}_{\infty,n}(\mathbf{x})| \leq \frac{b_n - a_n}{a_n} \tilde{m}_{\infty,n}(\mathbf{x}) + n\varepsilon_n \left(\max_{1 \leq i \leq n} Y_i\right).$$

Consistency results

Assume that $Y = m(\mathbf{X}) + \varepsilon$, where ε is a centered Gaussian noise, independent of $\mathbf{X}$, with finite variance $\sigma^2 < \infty$. Moreover, $\mathbf{X}$ is uniformly distributed on $[0,1]^d$ and m is Lipschitz. Scornet[26] proved upper bounds on the rates of consistency for centered KeRF and uniform KeRF.