Journal of Engineering and Applied Sciences 12 (21): 5603-5608, 2017
ISSN: 1816-949X

Implementation of Random Forest Machine Learning Algorithm

R. Roshen Sarma, Rajath R Joshi, R. Prashanth,
Syed Wajahath and Sharmila Chidaravalli
Department of Information Science and Engineering, Global Academy of Technology,
Ideal Homes Township, RR Nagar, Bengaluru, Karnataka, India

Abstract: This is aimed to implement Random Forest (RF) classification machine learning algorithm performance and investigate its properties. Implementation and all experiments are done in R environment using the Kaggle Dataset-Titanic: machine learning from disaster. Variable importance is estimated for the dataset using this method. Finally, variable selection using importance ranks influence on RF classification rates is analyzed.

Key words: Machine learning, ensemble, random forest, variables, algorithm, prediction

INTRODUCTION

In machine learning, an algorithm Random Forest (RF) that attracts attention of many researchers as it is widely known to be one of the more accurate classifiers. This researche aims to look into RF properties and capture the behaviour on a dataset and evaluate the algorithm performance.

The RF algorithm can also be thought of as a form of nearest neighbor predictor (Bradter *et al*., 2013) is an ensemble approach. The RF algorithm has become an intense research area now a days. It is developed from two successful approaches which finally led to the basis of RF algorithm. The first approach is known method from ensemble learning method where in the generalization error is decreased due to combining decisions of multiple learners which are usually weak and unstable individuals, i.e., the random forest takes the notion of the decision trees to the next level by combining trees with the notion of an ensemble. Thus, in ensemble terms the trees are weak learners). The second approach is random split selection for a decision tree. This split is chosen randomly from a subset of best splits (Breiman, 2001).

A RF is a classifier having a collection of tree structured classifiers h (x, θn), n = 1, ... , where the denote the independent identically distributed random vectors and every tree casts a unit vote for the more popular class at input it generally applies two mechanisms. The first mechanism is building an ensemble of trees via bagging with replacement, i.e., bootstrap aggregating (Michael and Labudde, 2014) which means that by sampling with replacement, some observations may be repeated in each new training data set. Each and every tree is grown using the obtained bootstrap sample. The second mechanism is a random selection of features at each tree node (feature selection) which performs random selection of a small fraction of features and further splitting using the best feature from this set. The size of a fraction, i.e., the number of features to be selected is fixed within the algorithm execution. The algorithm of random forest algorithm is presented in this study. Basic RF algorithm and some of its properties are described in this study. This study, also contains formal techniques used during investigation and outlines its scope revealing the goals. All experimental work is captured in experiments study where the tests are described and main results are presented in graphs. This study also analyses the experimental results and the followed by conclusion.

Each approach has its own draw backs in defiance of its success in improvement of accuracy in classification techniques. The best performance is exhibited by combining multiple learners in an entity when the basic learners are less correlated (i.e., generalization error rises as the correlation between learners in an ensemble increases). Thus, the ensemble should be gathered from weak unstable learners (i.e., trees) as strong learners (i.e., forest) are likely to be more correlated and they tend to over-fit that might outweigh the improvement of individual classification strength. The RF algorithm solves the challenge described earlier by maintaining the strength which minimizes the correlation. It is achieved by introducing randomness into the training process, especially when random selection of features results in diverse learners. These learners are individually strong due to splitting using the best feature from the random fraction. The RF algorithm has another property which

Corresponding Author: R. Roshen Sarma, Department of Information Science and Engineering, Global Academy of Technology, Ideal Homes Townshop, RR Nagar, Bengaluru, Karnataka, India

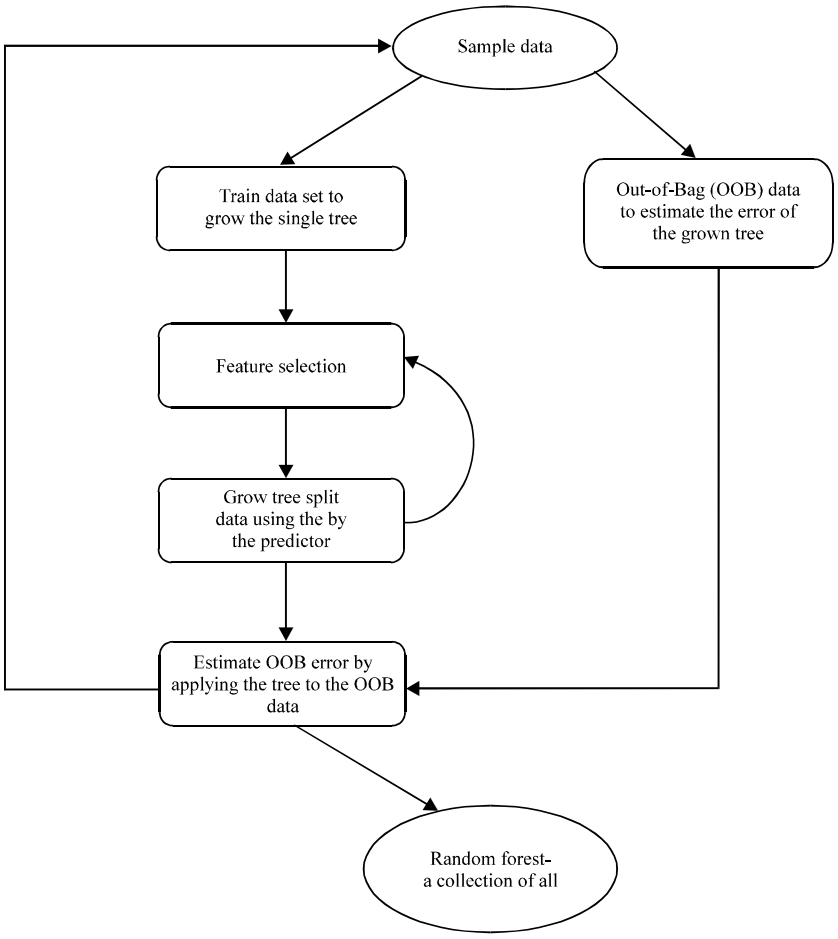

Fig. 1: Proposed model of RF

increases the diversity is that trees are not pruned during the growth. When a leaf size limit is reached, the growth is stopped.

MATERIALS AND METHODS

Proposed algorithm: Random forest algorithm combines a bunch of terrible decision trees into one awesome model. The block diagram of the proposed RF algorithm is shown as in Fig. 1. The performance of the RF algorithm is affected by only two parameters, m which is number of trees in an ensemble to grow and n the number of features to select randomly at each split.

This class of procedures has desirable characteristics. Its accuracy is as good as Ada boost and sometimes better. It's relatively robust to outliers and noise. It's faster than bagging or boosting. It gives useful internal estimates of error, correlation, strength and variable importance. It's simple and easily parallelized.

RF algorithm

For each tree in the forest:

- Take a bootstrap sample of the data
- Randomly select some variables
- For each variable selected, find out the split point which minimizes MSE or gain impurity or gain information if classification
- Split the data using the variable with the lowest MSE (or other statistics)
- Repeat step 2-4

Randomly select new sets of variables at each split until a stopping condition is satisfied or all the data is exhausted. Repeat the process to build several trees. To make a prediction, run an observation down some trees and take an average of the predicted values from all the trees or find the most popular class predicted if classified. 1and 2 is developed using step 1 and 2 as shown.

Step 1: Apply K iterations of bagging to create total K number of trees. Bagging (bootstrap aggregating): for a standard training set D of size n, bagging generates m new training sets D_i, each of size n by sampling from D uniformly and with replacement. By sampling with replacement, some observation maybe repeated in each D_i.

Step 2: Now for each of the k sample training set, we apply the attribute bagging (random subspace creation) and learn the decision tree. Show that the variable for any new code is the best variable (i.e., having least misclassifications error) among extracted random subspaces.

Random subspace (attribute bagging): Let each training object X = (I = 1, 2, 3, … , n) in the training sample set X = (x1, x2, … , xn) be a 'p' dimensional vector. $X = (x_{i1}, x_{i2}, \ldots, x_{ip})$ described by p features. Random subspace sampling one randomly selects r<p features from the dimensional data set X. One thus obtains r dimensional random subspace of the original p dimensional feature space.

Therefore, the modified training set $X_b = (X_{1b}, X_{2b}, \ldots, X_{nb})$ consists of r dimensional training objects $X_{ib} = (x_{i1b}, x_{i2b}, \ldots, x_{irb})$ for i = (1, 2, … , n) where r components. X_{ijb} (j = 1, 2, … , x) are randomly selected for p components x_{ij} (j = 1, 2, … , p) of the training vector X. Here b>1. The most important parameters that will be considered by RF are:

- N-estimators: number of trees in the forest. Choose a number as high as your computer can handle
- Maximum features: number of features considered while looking for a split
- Minimum sampler leaf: minimum number of samples in a newly created leaf

The random forest algorithm model can be trained efficiently by considering parameters like n jobs, random state, OOB score. Among these OOB prediction does the work correctly. About one third of observations don't show up in a bootstrap sample because an individual tree in the forest is made from a bootstrap sample. Meaning that about a third of the data was not used to build that tree, we can track which observations were used to build which tree. After the forest is built, we can take each observation in the dataset and identify which tree used the observation and which tree did not use the observation (based on bootstrap sample). We use the trees, the observations was not used to build to predict the true values of the observation. About a third of the trees in the forest will not use any specific observation from the data set. OOB predictions are similar to the following awesome:

- Train a model with n-estimator trees but exclude one observation from the dataset
- Use the trained model to predict the excluded observation. Record the prediction
- Repeat the process for every single observation in the dataset
- Collect all your final predictions, these will be similar to your OOB prediction errors

Analyzing the performance of RF: A pair of values (feature/threshold) which is used for splitting, maximizes the split function is fixed. The general form of impurity based split functions (Siknoja, 2004) is:

$$I(y_j) = \text{imp}(y) - \sum_j \frac{|Dy_j|}{|D|}\text{imp}\left(\frac{y}{y_j}\right) \quad (1)$$

Where:

y_j = The set of split indexes
imp(y) = The impurity of class labels before the split
imp(y, y_j) = The impurity of class labels after the split on y_j

If I values are high then it causes better splits. Gini impurity is the most commonly used split function. It is measured by:

$$IG = \sum_i p_i(1-p_i) - \frac{|D_l|}{|D|}\sum_i p_i^l(1-p_i^l) - \frac{|D_r|}{|D|}\sum_i p_i^r(1-p_i^r) \quad (2)$$

And the information gain entropy is given by:

$$IE = H(D) - \frac{|D_l|}{|D|}H(|D||D_l) - \frac{|D_r|}{|D|}H(|D||D_l|) \quad (3)$$

Here, $H(T) = -\Sigma_i p_i \log(p_i)$ the entropy function, T_l, T_r -left and right splits correspondingly, p_i, p_i^l, p_i^r-probabilities of class i in samples T, T_l, T_r. Therefore, randomness is introduced into the learning process of each tree and completed ensemble consists of diverse learners (Khabsa *et al*., 2016). Even if more trees are added the RF does not over fit and this is one of its valuable properties. The generalization error declines instead. The upper bound for this error is given by:

$$GE \leq \bar{\rho}\left(\frac{1}{s^2} - 1\right) \quad (4)$$

Where:
ρ = The mean value of correlation
s = The strength of the ensemble

Thus, correlation should be decreased while strength increased in order to lower the error bound considerably. Once RF is trained it is treated as a completed ensemble of base learners (i.e., trees). Each learner produces a decision individually in order to perform classification of arbitrary example x and then the generation of decision of the ensemble as a whole is generated by combining the decisions of all learners in the ensemble. Simple voting where all learners are equal and the final decision is derived as a majority vote is the easiest way to perform that. Assigning weights to each learner making some of them more infiuential than others A more general approach. The class prediction becomes a weighted sum of individual ones (Alpaydin, 2010):

$$\tilde{y}_i \sum_j w_j d_{ij} \quad (5)$$

Here, $W_j \geq 0$, $\Sigma_j W_j = 1$-defines the weights which may depend on the training error of a corresponding learner, v_i defines the prediction for class j. The final decision is given as:

$$\tilde{v} = \arg\max \tilde{v}_i \quad (6)$$

Bayesian combination scheme can be derived from Eq. 5 if weights are considered as approximations of prior probabilities for learners and decisions as conditional likelihoods:

$$P(C_i|x) = \sum_{j=1}^{m} P(M_j) P(C_i|x, M_j) \quad (7)$$

where, M_j, j = 1, … , m defines the base learner models, all mentioned combination schemes are potentially applicable to RF as it is an ensemble of trees. Since, RF is an ensemble of trees, all the mentioned combination schemes are potentially applicable to it. There is another property which allows applying RF not only for classification but for feature interpretation and selection as well. The fact that, trees actually select the best feature from a random fraction at every split helps to derive the property mentioned earlier consequently. Thus, tree rules are built on the most important features while training. Therefore, the corruption of less important features does not lead to significant variance of tree error rate while affection of the most important ones does. This idea can be used to formulate a technique for assessment of variable (or feature) importance. For each tree in RF, an Out-of-Bag sample (OOB) is considered and is associated with its corresponding tree. OOB_j contains examples which are not used to train tree d (Genuer *et al*., 2010) err (OOB_j) denotes the test error on this sample for tree d. By permuting the values of feature t randomly a new sample OOB_j^t is obtained from OOB_j. Then evaluation of tree d on sample OOB_j^t is performed and the test error is captured. Finally, importance of feature t is estimated as average growth of error rate across all m trees in RF:

$$VI(t) = \frac{1}{m} \sum_{j=1}^{m} \left[err(OOB_j^t) - err(OOB_j^t) \right] \quad (8)$$

In experiments section, evaluation and visualization of classifiers performance is illustrated. This makes it necessary to describe briefily this technique here. For many practical applications, especially where unequal error costs take place assessing and comparing classifiers only by its average error rates and deviations is not enough.

Calculation of a confusion matrix for every dataset is possible to which consists of four rates: true positive, false positive, false negative and true negative. The true positive rate against the false positive rate is shown by a basic ROC graph. A whole curve showing classification performance in ROC space is produced by several approaches. Threshold averaging technique (Fawcett, 2006) is performed to get final curves with vertical and horizontal deviations.

RESULTS AND DISCUSSION

All experimental tests look into two datasets. The first one is to train which consists 891 observations with 12 variables. The second one is test consisting of 418 observations with 11 variables. The observations in each dataset are marked with labels. Both datasets include observations of three classes only. Here, the number of features in a fraction n = √ (ftnum), leaf size limit lsize = 5, the number of random thresholds tpnum = 5 are the recommended parameter values used throughout all experiments unless explicitly stated. Information gain measure Eq. 3) is applied as splitting criteria. While comparing to Gini impurity in both datasets there is no significant difference in performance as shown in an experimental test. It is shown that on the given datasets RF with weighted and Bayesian combination scheme performs just slightly better than RF with simple voting.

Hence, the classification performance results are not affected by choice of the combination scheme considerably. The two fold cross validation is applied in all experiments when a small size of explored datasets is considered. From the original dataset, examples for each fold are selected randomly. The first RF classifier is made

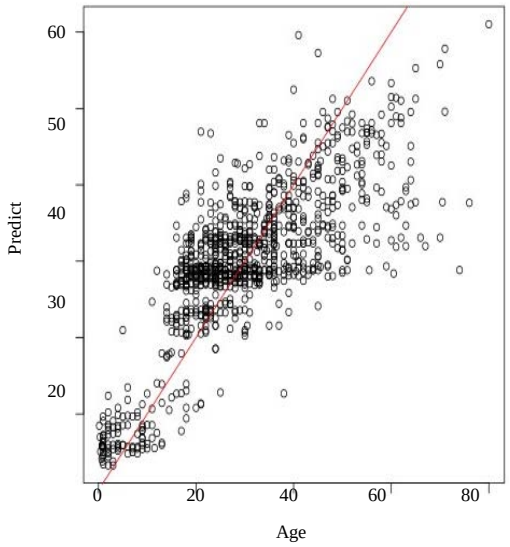

Fig. 2: Decisions plot based on age

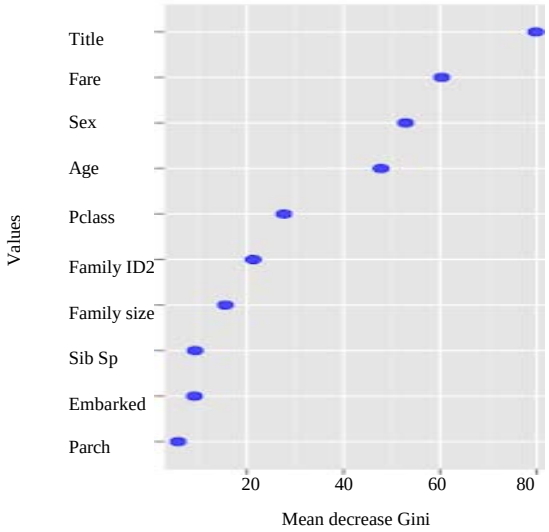

Fig. 3: Gini important variable

to train on the first fold and test on the second one. In opposite, the second RF classifier is learned on the second fold and tested on the first one. This methodology ensures that each of the two classifiers does not experience examples which another one does. On the samples of equal size and its results are captured both the classifiers are trained and tested. A mean test error is calculated from the test errors of two classifiers and the whole procedure is repeated for several times and folds are formed on each step via random selection of examples. Thus, the described method is used in all experiments.

Prediction 1: The first experiment is aimed to investigate the performance of RF against the number of trees m from 1 to m trees trained and added to the ensemble. The first trick is to use bagging for bootstrap aggregating. A randomized sample of all the rows in the train data set is taken by bagging with replacement. This is easy to simulate in R using the sample function. Each decision tree grown with bagging would evolve slightly differently. The RFs considers only a subset of pool of available variables, usually the square root of the number available. We have 10 variables in our case, so using a subset of three variables would be rational. For every node in the decision trees, the selection of available variables is changed. The simple voting scheme combines decisions of individual trees. We then create our first decision tree prediction based on the gender values to differentiate between male and female and age values to differentiate between children and adults as shown in Fig. 2.

Prediction 2: We first try to clean up the NA (Not Available) values and missing values. In our dataset, there are a lot of age values missing. The tree would search for another variable if any of our decision trees split in a similar way to age and use them instead. We notice that 20% of the values are missing and we use "ANOVA" method of our decision tree as we are not trying to predict a category but a continuous variable. Embarked and fare both are lacking values in two different ways. The RF algorithm can digest up to 32 levels. The family ID variable has more levels than the RF algorithm, so we could either convert these levels to underlying integers and having the tree treat them as continuous variables or to keep it under the threshold we could manually reduce the number of levels. The syntax is like decision trees but with a few extra options and the model is now ready to run.The important variable is as shown in Fig. 3.

The OOB observations is not wasted by the RF algorithm, it uses them to see how well each tree performs on unseen data. The Gini graph checks to see if every variable is taken out and the variable is important if it has a high score. The title variable was at the top for both the measures. There are more than one ensemble model. A forest of conditional inference trees is one of the ensemble models and their decisions are made using a statistical test rather than a purity measure but the basic construction of each tree is similar. Conditional inference trees can handle more levels of factors than random forests. The prediction requires a few extra nudges for conditional inference forests like the OOB values. Each of the predictions focus on points of interest conditions. Every experiment provides the results of RF algorithm on train and test datasets.

CONCLUSION

Research aims to evaluate RF classification performance and investigate its properties on the datasets. Predictions focus on the two main parameters of the RF algorithm: the number of trees m and the number of randomly selected features n on each split. The mathematics behind decision trees is considered by the Gini technique but fundamentally measures purity of the nodes are at the end of the tree. The lower values of n have more unstable error rates which are shown by wider conditional intervals. All the properties outlined make RF an efficient and powerful classifier with accuracy comparable or more favorable than other state-of-the-art-classifiers. Additionally, RF is an easily parallelized and essentially multi-core friendly algorithm. Since, trees can be trained simultaneously on separately generated bootstrap samples. The latter additionally increases its popularity in practical machine learning applications.

REFERENCES

Alpaydin, E., 2010. Introduction to Machine Learning. 2nd Edn., MIT Press, Cambridge, Massachusetts, ISBN:9780262012430, Pages: 537.

Bradter, U., W.E. Kunin, J.D. Altringham, T.J. Thom and T.G. Benton, 2013. Identifying appropriate spatial scales of predictors in species distribution models with the random forest algorithm. Methods Ecol. Evol., 4: 167-174.

Breiman, L., 2001. Random forests. Mach. Learn., 45: 5-32.

Fawcett, T., 2006. An introduction to ROC analysis. Pattern Recognit. Lett., 27: 861-874.

Genuer, R., J.M. Poggi and C.T. Malot, 2010. Variable selection using random forests. Pattern Recognit. Lett., 31: 2225-2236.

Khabsa, M., A. Elmagarmid, I. Ilyas, H. Hammady and M. Ouzzani, 2016. Learning to identify relevant studies for systematic reviews using random forest and external information. Mach. Learn., 102: 465-482.

Michael, R.C. and A.R. LaBudde, 2014. An introduction to bootstrap methods with applications to R. John Wiley & Sons, Hoboken, New Jersey, USA.,.

Sikonja, M.R., 2004. Improving Random Forests. In: Machine Learning ECML, Boulicaut, J.F. (Ed.). Springer, Berlin, Germany, pp: 359-370.