

A Study of Random Forest Algorithm with implemetation using Weka

¹Dr. N.Venkatesan,
Associate Professor, Dept. of IT.,
Bharathiyar College of Engg. & Technology,
Karikal, India

²Mrs. G.Priya,
Assistant Professor, Dept. of CS & CA
Siga College of Management & Computer Science
Villupuram, India

Abstract— This paper provides the depth exploration of one of the best classification algorithm in data mining. The result.csv dataset used to implement this algorithm is a sample result dataset of a college. This paper tells about the clear vision Random forest algorithm with what it is, where we have to use this algorithm and its benefits then algorithm implementation with clear screen shot with explanations. This paper mainly focus on the development, applications and the significance of Random forest algorithm.

Keywords—Data Mining; Weka tool; random forest algorithm; classification; dataset;

I. INTRODUCTION

Random Forest (RF) is widely used to be a powerful new approach to data exploration, data analysis and predictive modeling. Random Forest are proposed by Breiman (2001) father of CART at University of California, Berkley. The results of random forests constructed from results from individual decision trees out of set of decision trees which are learned independently from a subset of training data. Random Forest has its roots in CART, Learning ensembles, committees of experts and combining models. Random forest offer Data Visualization for High dimensional data (many columns), clustering, outlier and error detection.

A Random Forest is a collection of CART-like trees for growing, combination, testing and Post-processing. one is an ensemble of trees where each tree is growing while training on a sample obtained from the raining set via bagging *without* replacement. This is a known technique from ensemble learning methodology where generalization error is decreased due to combining decisions (or so-called votes) of multiple learners which are usually weak and unstable individually. The second approach is random split selection for a decision tree. This split is chosen randomly from a subset of best splits. Thus, these two ideas led finally to the basis for For algorithm. It generally applies two mechanisms: building an ensemble of trees via bagging *with* replacement (bootstrap) and a random selection of features at each tree node. The first one means that any example selected from the training set can be selected again. Each tree is grown using the obtained bootstrap sample. The second mechanism performs random selecting 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 select) is fixed within the algorithm execution. This article is organized in 5 sections. Basic RF algorithm and its common properties are described in Background section. In addition, this section contains formal techniques used during investigation and outlines its scope revealing the goals. All experimental work is captured in Experiments section where statistical tests are described and main results are presented in graphs. The fourth section analyses experimental results and the fifth section makes conclusions.

II. BACKGROUND

A. RANDOM FOREST CLASSIFIER

- *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 [1] and is implemented in the R package random Forest. The first step in measuring the variable importance in a data set 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 -th feature after training, the values of the -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 -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.

- *Proximity*

These are one of the most useful tools in random forests. The proximities originally formed an $n \times n$ matrix. After a tree is grown, put all of the data, both training and oob, down the tree. If cases k and n are in the same terminal node increase their proximity by one. At the end, normalize the proximities by dividing by the number of trees. Users noted that with large data sets, they could not fit an $n \times n$ matrix into fast memory. A modification reduced the required memory size to $n \times T$ where T is the number of trees in the forest. To speed up the computation-intensive scaling and iterative missing value replacement, the user is given the option of retaining only the n largest proximities to each case. When a test set is present, the proximities of each case in the test set with each case in the training set can also be computed. The amount of additional computing is moderate.

- *OOB (Out of Bag Error)*

In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the run, as follows: Each tree is constructed using a different bootstrap sample from the original data. About one-third of the cases are left out of the bootstrap sample and not used in the construction of the k th tree. Put each case left out in the construction of the k th tree down the k th tree to get a classification. In this way, a test set classification is obtained for each case in about one-third of the trees. At the end of the run, take j to be the class that got most of the votes every time case n was oob. The proportion of times that j is not equal to the true class of n averaged over all cases is the oob error estimate. This has proven to be unbiased in many tests.

- *Features of Random Forests*

- It is unexcelled in accuracy among current algorithms.
- It runs efficiently on large data bases.
- It can handle thousands of input variables without variable deletion.
- It gives estimates of what variables are important in the classification.
- It generates an internal unbiased estimate of the generalization error as the forest building progresses.
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.
- It has methods for balancing error in class population unbalanced data sets.
- Generated forests can be saved for future use on other data.
- Prototypes are computed that give information about the relation between the variables and the classification.
- It computes proximities between pairs of cases that can be used in clustering, locating outliers or (by scaling) give interesting views of the data.
- The capabilities of the above can be extended to unlabeled data, leading to unsupervised clustering, data views and outlier detection.
- It offers an experimental method for detecting variable interactions.

Despite the success of previous approaches in improvement of classification accuracy, each one has its own drawbacks. The technique of combining multiple learners in an ensemble shows the best performance (i.e. the smallest generalization error) when the basic learners are least correlated. In other words, generalization error rises as the correlation between learners in the ensemble increases. Therefore, the ensemble should be gathered from weak unstable learners since strong learners are likely to be more correlated and tend to over fit which may outweigh improvement of individual classification strength.

RF algorithm solves described challenge by minimizing correlation while maintaining strength. It is achieved by injecting the randomness into the training process. Particularly, random selection of features results in diverse learners which are still individually strong due to splitting using the best feature from the random fraction. Another property increasing diversity is that trees are not pruned during growing. Instead, growing is stopped when a leaf size limit is reached.

In fact, RF algorithm has only two main parameters affecting its performance considerably. The number of trees m in an ensemble to grow and the number of features k to select randomly at each split. Moreover, it is naturally suitable for multiclass classification challenges because it consists of tree classifiers. Unlike many classifiers cause growing number of parameters to



tune while combining them as multiple learners in an ensemble in order to deal with multiclass classification, RF algorithm retains only two mentioned parameters mainly to consider. The RF algorithm is presented below.

Algorithm 1 Random Forest

Input: dataset $T = (x, y)$, number of trees m , number of random features k

Output: RF , a set of grown trees

Initialize RF

for $i = 1$ **to** m **do**

$T' \leftarrow bootstrap(T)$

$Tree \leftarrow trainDT(T', k)$

add $Tree$ to RF

end for

Algorithm 1 shows the technique of building an ensemble of decision trees using bagging. The function $trainDT(T', k)$ performs training of a decision tree on a bootstrap sample T' selecting k features randomly at each split. The process of training a decision tree needs clarification and described in Algorithm 2.

Algorithm 2 Decision Tree

Input: a sample $T = (x, y)$, number of random features k , a leaf size limit $lsize$

Output: $Tree$, a trained decision tree Initialize $Tree$, $fnum$ as a total number of features, $tnum$ as a predefined number of thresholds

function $trainDT(T, k)$

if $sizeof(T) \leq lsize$ **then**

$label \leftarrow indexof(max \text{ in } histogram(T))$

else

$fraction \leftarrow random(k \text{ from } fnum)$

$thresholds \leftarrow random(tnum)$

$(f, t) \leftarrow max \text{ of split function in } fraction \text{ and } thresholds$

$(leftT, rightT) \leftarrow split(T, f, t)$

add left child $\leftarrow trainDT(leftT, k)$

add right child $\leftarrow trainDT(rightT, k)$

end if

end function

III. IMPLEMENTATION

This algorithm is implemented through weka with the following screen shot and its output. Weka was developed at the University of Waikato in New Zealand; the name stands for *Waikato Environment for Knowledge Analysis*. It provides a uniform interface to many different learning algorithms, along with methods for pre- and post processing and for evaluating the result of learning schemes on any given dataset.

A. Dataset Result.csv

Markid	Regno	code	Int Marks	Ext Marks	Total	Result
M1	10UCCA006	UCCM601	24	51	75	PASS
M2	10UCCA006	UCCM602	25	39	64	PASS
M3	10UCCA006	UCCM603	25	43	68	PASS
M4	10UCCA006	UCCM604	25	53	78	PASS
M5	10UCCA006	UCCM606	21	49	70	PASS
M6	10UCCA006	UCCM607	-	88	88	PASS
M7	10UCCA006	UCCO601	24	46	70	PASS
M8	10UCCA006	UCCR605	60	39	99	PASS

M9	10UCCA006	USKS601	84	-	84	PASS
M10	10UCCA021	UCCM601	20	43	63	PASS
M11	10UCCA021	UCCM602	19	37	56	PASS
M12	10UCCA021	UCCM603	19	34	53	PASS
M13	10UCCA021	UCCM604	20	36	56	PASS
M14	10UCCA021	UCCM606	17	39	56	PASS
M15	10UCCA021	UCCM607	-	76	76	PASS
M16	10UCCA021	UCCO601	18	37	55	PASS
M17	10UCCA021	UCCR605	60	38	98	PASS
M18	10UCCA021	USKS601	77	-	77	PASS
M19	12UMAT046	UMAM402	15	13	28	FAIL
M20	12UMAT058	UMAM402	15	14	29	FAIL
M21	12UMAT071	UPHA402	18	23	41	FAIL
M22	12UPHY014	UCHA401	19	8	27	FAIL
M23	12UPHY015	UCHA401	19	6	25	FAIL
M24	12UPHY021	UCHA401	20	14	34	FAIL
M25	12UTAM001	UBCE403	15	24	39	FAIL
M26	12UTAM003	usks401	29	-	29	FAIL
M27	12UTAM012	UBCE403	13	18	31	FAIL
M28	12UTAM013	UBCE403	12	11	23	FAIL
M29	12UTAM013	UTAM401	22	17	39	FAIL

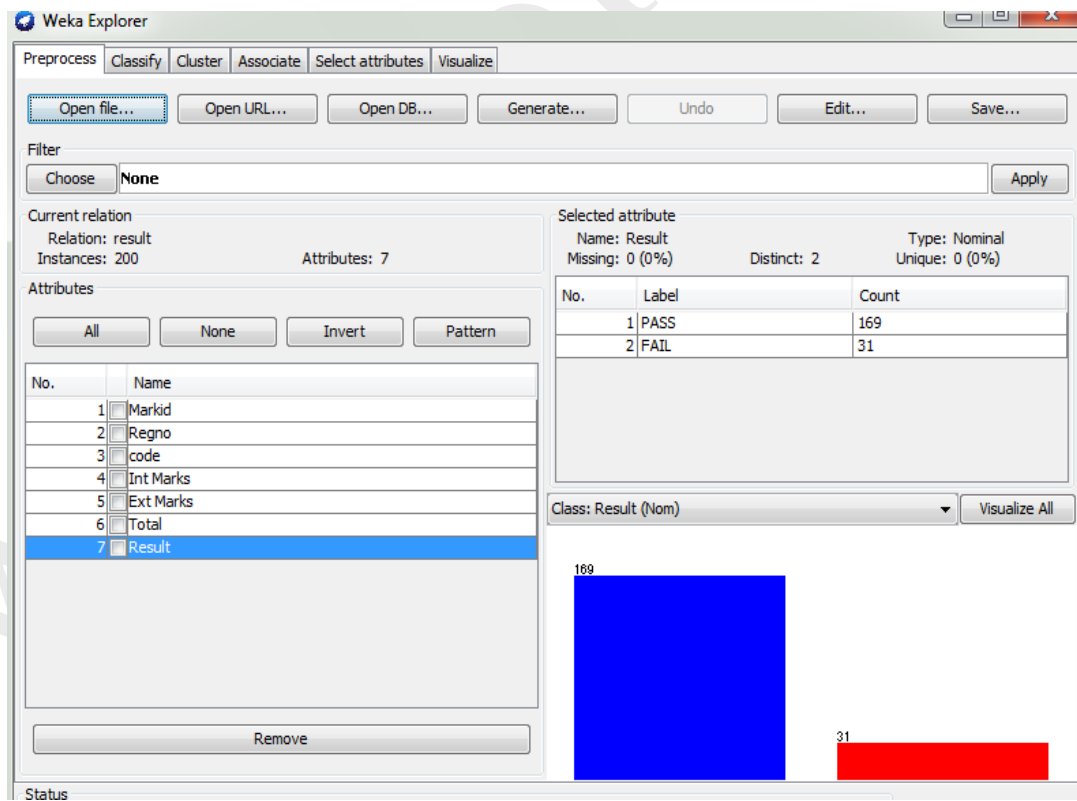
Total No of Instances taken: 200

Pass: 169

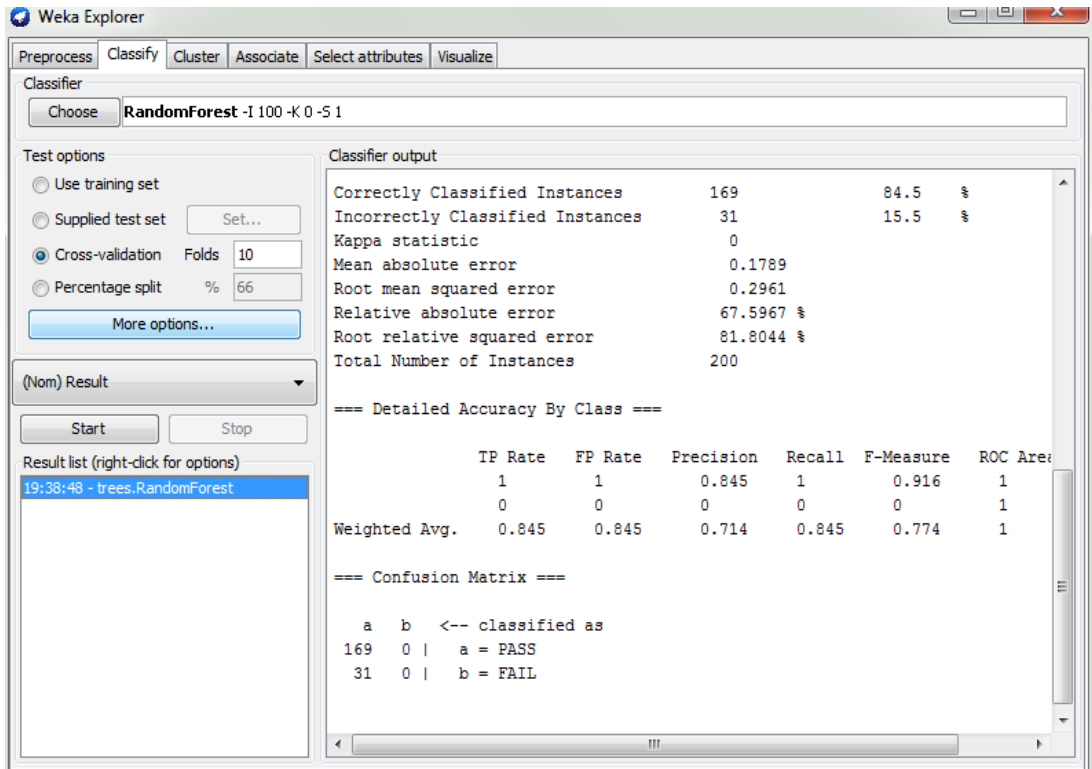
Fail: 31

Pass or Fail is determined by Int Marks and Ext Marks attributes.

B. Screen of Data Preprocessing in weka



C. Screen of Random Forest implementation in weka



Classifier Output

=== Run information ===

Scheme:weka.classifiers.trees.RandomForest -I 100 -K 0 -S 1

Relation: result

Instances: 200

Attributes: 7

- Markid
- Regno
- code
- Int Marks
- Ext Marks
- Total
- Result

Test mode:10-fold cross-validation



==== Classifier model (full training set) ====

Random forest of 100 trees, each constructed while considering 3 random features.

Out of bag error: 0.15

Time taken to build model: 0.13 seconds

==== Stratified cross-validation ====

==== Summary ====

Correctly Classified Instances	169	84.5 %
Incorrectly Classified Instances	31	15.5 %
Kappa statistic	0	
Mean absolute error	0.1789	
Root mean squared error	0.2961	
Relative absolute error	67.5967 %	
Root relative squared error	81.8044 %	
Total Number of Instances	200	

==== Detailed Accuracy By Class ====

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	1	1	0.845	1	0.916	1	PASS
	0	0	0	0	0	1	FAIL
Weighted Avg.	0.845	0.845	0.714	0.845	0.774	1	

==== Confusion Matrix ====

```

a b <-- classified as
169 0 | a = PASS
31 0 | b = FAIL

```

IV. CONCLUSION AND FUTURE WORK

This paper was implemented with an ordinary small sample dataset with 200 instances. This would be a very small work of a research. This work is extended to world wide web dataset for the implementation of web mining. Web spam detection is a very big problem of the today's internet world. This paper analysis work in that extends to the application of content spam features.



Acknowledgment

This work was supported by my Ph.D. guide and colleagues friends who gave me best support to fulfill this paper successfully.

References

- [1] L. Breiman, L. Random Forests, Machine Learning, vol. 45 pp. 5–32, 2001.
- [2] Robnik-Sikonja, M. Improving to Machine Learning, MIT Press, Cambridge, Massachusetts, 2nd Edition
- [3] Witten, Ian and Frank, Eibe. Data Mining, Practical Machine Learning Tools and Techniques
- [4] Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., Witten I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations, Volume 11 Issue 1 (2009).
- [5] R. Nicole, "Title of paper with only first word capitalized," J. Name For. Abbrev., in press.
- [6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740-741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
- [7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

BIOGRAPHIES

Dr. N. Venkatesan has completed his Ph.D. in Madurai kamaraj Univeristy. He is at present working as Associate Professor, Dept. of IT., Bharathiyar College of Engg. & Technology, Karikal. He is an approved Ph.D. guide under bharathiar University. His area of interest is Data Mining and Image Processing.

Mrs.G. Priya, Completed her M.C.A., M.Phil., degree in Madurai Kamaraj University, Madurai. She has 7 years of teaching experience. She is at present working as Assistant Professor, Dept. of CS & CA in Siga College of Management and Computer Science, Villupuram. Her research area is Web mining, Data Structures and Algorithms.