**RESEARCH ARTICLE**

International Journal of Science, Technology and Society (IJSTS)
Vol. 5, No. 1 & 2 (2020-21): 48-52

# Prediction for Software Quality Open-Source Dataset Based on CART Approach Using Machine Learning Techniques

**Krishan Kumar**

Department of Computer Science, Faculty of Technology, Gurukula Kangri (Deemed to be University), Haridwar, UK

***Publication Info***

***Article History:***
Received: 05-03-2021
Accepted: 27-03-2021
DOI: 10.18091/ijsts.5129

***Keywords:***
Classification tree, Regression tree, Fault prone module, and Quality prediction, J48, Random Forest and Logistic Model Tree.

***Corresponding Author:***
Krishan Kumar
Email:
krishan.kumar@gkv.ac.in

## ABSTRACT

Open-source dataset available for different source of platform.Software quality is an important approach for the user as well as softwaredevelopers also.In this paper the study of various classification and regression tree (CART) method for software quality predictions. In this method have been used for a new algorithm design which is based on partitioning method. Among so many prediction methods over the recently published data mining predictionsfor software quality models such that classification and regression tree (CART), deep neural network (DNN), hierarchical attention network (HAN), multiple linear regression (MLR), stepwise regression (SR), artificial neural network (ANN) and case-based reasoning (CBR).CART algorithm to design an enhanced level of classification accuracy for large (complex) data set when compared with prior to classical CART concepts. CART can perform a balanced tree structure that is misclassify the fault-prone modules to classification-tree models based on Open-source dataset. The Open-source dataset used in Vote to classify the various machine learning algorithms such as J48, Random Forest, Logistic Model Tree. In this paper we also compute the best accuracy for vote dataset. This paper presents details on the cart algorithm to predict the quality of system in different ML algorithms.

## NOTATION

| | |
|---|---|
| CART | Classification and regression tree |
| fp | fault prone |
| nfp | not fault prone |
| ANOVA | Analysis of variance |
| PCA | Principle component analysis |
| ○ | X, Y, Z and Wroot node |
| ▭ | a,b and c rectangle leaf Node |
| □ | Condition value not give |

## INTRODUCTION

In this paper introducedthat a new kind of classification and regression tree (CART) concept to classifying by non-parametric technique that can any variables interactions to determining an outcomes or dependent variables using machine learning classifier algorithms. Outcomes variable is continuous part, CART produces regression trees, if the variable is categorical, and CART produce the classification tree. CART algorithm includes the procedure of classification tree that construction for, regression pruning and optimal classification tree. A software quality model can be used to quality of fault data from, the training data set and sample data to Open-source dataset. The regression tree model of software quality we used to classify software modules by applying our splitting rule to the tree. Here we apply the DNN, HAN, MLR and CBR regression tree algorithm Fig. 1 (Clark et al, 1992).

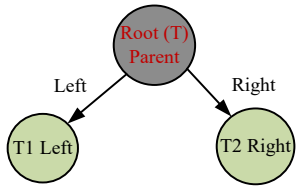


**Fig. 1:** Classify tree rule

CART for machine learning algorithms classification techniques included in decision tree and neural networks. Decision tree is design for classification of discrete variables. CART is a more reliable technique for continuous value. To determine the software quality improvement is a risk-based targeting of software modules depending on their predicted quality (Schneidewind, 2002). A typical, software modules are categorized by a classification model into two risk-based groups, such as fault-prone (fp), and not fault-prone (nfp) problems based on performance (Khoshgoftaar et al, 2001, 2002).

- Modification Expected Cost of Misclassification (MECM) (Khoshgoftaar et al, 2005) rate in fault prone module to CART methodology.
- The number of predicted fault prone (fp) modules, the open-source data set should beequal to number of nodesto allocated by software quality.

Now day`s research the most important and widely used techniques for producing decision tree is the Classification and regression tree (CART) using ML algorithms (Brieman et al, 1984).This techniqueis a classification tree or data splitting tree which important types from the independent variable to predicta split variable differentfrom predict a response variable.

## RELATED WORKS

Recently working for prediction and ML techniques to define over all related works. A classification tree is an algorithm, represent as a tree graph, which classifies an input value to cross ponds an output value such as a software quality improvement process to any dataset. Alternative classification techniques used in software-quality modeling include:

- Discriminant analysis (Khoshgoftaar, 1996),
- Discriminative power technique (Schneidewind, 1995),
- Logistic regression (Basili, 1996),
- Pattern recognition (Briand, 1992),
- Artificial neural networks (Khoshgoftaar, 1995),
- Fuzzy classification (Ebert, 1996).

The various types of quality-based prediction and ML techniques, Khoshgoftaar et al. (Khoshgoftaar et al., 2004) assess quality of prediction techniques, to evaluate such as CART, DNN, HAN, case base reasoning (CBR) and logistic regression (LR). Guo et al. (Guo et al., 2000) have proposed a model based on statistical plus (S-Plus) techniques of combined model analysis for prediction of a classify module as fault prone (fp) and non-fault prone (nfp) category. In this model build from they used expectation maximum classification tree algorithm along with size of dataset for quality prediction in any software. ML methodology also used for Akaike Information Criterion (Akaike, 1974) in their model and claimed that this technique can be used to develop a new model for predicting software quality to given dataset even when previous knowledge about fault prone (fp) count and modules is unavailable problem size. Gyimthy et al. (Gyimthy et al., 2005) assessed fault prediction for open-source software function. They compared different versions by consulting bug database system to classify the decision-basedsystem.

More techniques used for Object Oriented based systemsuch as methods per Class, Depth of Inheritance Tree, response for a Class Polymorphism. (Khan et al., 2006) compared six Artificial Intelligence based techniques, Bayesian Belief Networks Neural Networks Fuzzy Logic, Support Vector Machines (SVM), Expectation Maximum (EM) Algorithm and CaseBased Reasoning.Somevariety of classification different techniques have been used to tree model to predict software quality, including logistic regression (LR) (Basili et al., 1996), (Khoshgoftaar, 1999), optimal set reduction (OSR) (Briand et al., 1993), artificial neural networks (ANN) (Khoshgoftaar et al., 1995), fuzzy classification (FC) (Ebert, 1996),deep neural network (DNN), hierarchical attention network (HAN), multiple linear regression (MLR), and tree-based model (TBM). Our discover model fault-prone (fp) modules with the CART algorithm (Breiman et al., 1984), (Khoshgoftaar et al., 1999) and the Tree-DISC algorithm (Khoshgoftaar et al., 2000), (SAS, 1995) which process based model refinement consequence of misclassification error rate thetree algorithm (Kass, 1980). S-Plus has a classification tree algorithm (Clark et al., 1992), but do not produce tree with our case-study data set. In this addition points to classification trees, regression trees have also been applied to software quality.We apply to this method, Kitchenham (Kitchenham, 1998) briefly reports using the Classification and Regression Tree (CART) algorithm (Breiman et al., 1984) to model for software productivitysystem. In the future work, Gokhale and Lyu suggest applying a threshold to the predicted quantity to classify modules (Gokhale et al., 1997). Case studies by Gokhale and Lyu (Gokhale et al., 1997) and Troster and Tian (Troster et al., 1995), used the S-Plus and otherclassification algorithm to predict the number of faults and non-faulty module.

We represent a case-studies learning model (Kitchenham, 1998), (Khoshgoftaar et al., 1998) with real response variables and real predictors (Kitchenham, 1998) suggests measurement tree structure a way to classification algorithms (Kitchenham, 1998) to implemented in the WEKA environment system. To minimizes the total specified deviance module of the partitioned subsets system (Naik, 1998).This technique focuses for CART (Gupta et al., 2012) automatically builds a hierarchical tree to decision value from continuous ordinal prediction by first building a maximal tree and then pruning tree method in detail. In this works performs the task of (Naik, 1998) briefly reports using CART method to software project productivity analysis system. CART has been mainly used to model software quality (Gokhale et

al., 1997), (Troster et al., 1995), tree-based methodology system to open-source dataset.

## MACHINE LEARNING TECHNIQUES

Machine learning are several types of classification tree-decision-based techniques and many splitting methods to represent tree algorithm. We use the CART to decision rule to each external node predicted value, and label of the tree leaf with a classification. Each module showing to data, fault-prone (fp) or non-fault-prone (nfp).A tree algorithm assigns a split partitioning a data set. Each node denotes a left sibling, right sibling and one parent rooted node.

- Each non-terminal node represents a decision based on predict value.
- Each leaf label data with a set of value.
- Compute the input data nodes are independent variable or response variable.
- Updating the data set value fault or non-fault.

## CLASSIFICATION TREE

### Decision Trees

A decision tree is a flow-chart-like tree structure. The internal node denotes a test on an attribute, each branch represents an outcome of the test, and leaf nodes represent classes or class distribution (Gyimthy et al., 2005). The top most node in a tree shown by oval is a root node. Further internal nodes are represented by rectangles, and leaf nodes are denoted by circles which are depicted in Figure 2.

### J48 Algorithm

J48 is a tree-based learning approach. It is developed by Ross Quinlan which is based on iterative dichtomiser (ID3)algorithm. J48 uses divide-and-conquer algorithm to split a root node into a subset of two partitions till leaf node (target node) occur in tree. Given a set T of total instances the following steps are used to construct the tree structure.

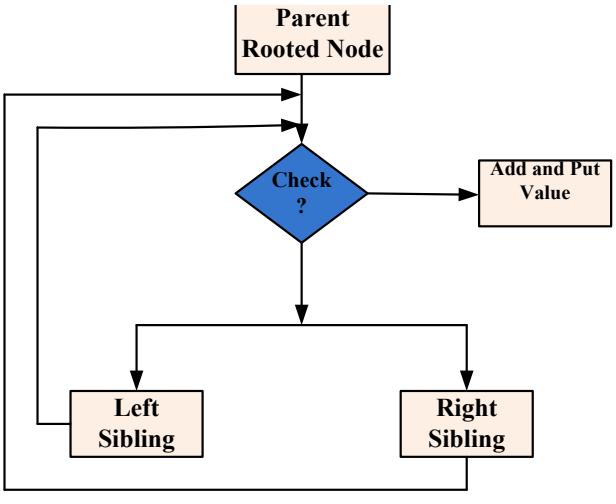


**Fig. 2:** Structure Tree Point

### Random Forest Algorithm

Random Forest algorithm was initially developed by Leo Breiman, a statistician at the University of California (Akaike, 1974) Berkeley. Random Forests is a method by which one cancalculate accuracy rate in better way.

### Logistic Model Tree

Logistic Model Tree (LMT) (Akaike, 1974) algorithm makes a tree with binary and multiclass target variables, numeric and missing values. So, this technique uses logistic regression tree. LMT produces a single outcome in the form of tree containing binary splits on numeric attributes.

### Cross-Validation Test

Cross-validation (CV) method used in order to validate the predicted model. CV test basically divide the training data into a number of partitions or folds. The classifier is evaluated by accuracy on one phase after learned from other one. This process is repeated until all partitions have been used forevaluation (Akaike, 1974). The most common types are 10-fold, n-fold and bootstrap result obtained into a single estimation.

### Process for CART Induction Algorithm

Tree based algorithms are often considered is machine learning techniques, because the structure of a tree is splitting recursive partitioning data set.

**Input:** parent root node T, split child node ($T_x$, $T_y$), data partitioning set S1, S2, ....., Sn.

**Output:** classification and regression tree, rooted parent node at T.

**Beginning** (parent root node T, split child node ($T_x$,$T_y$), data partitioning set S)

1. input given data item value
2. put the item value of each node
3. let n be the number of sibling node of T
4. if (T splits) condition check
5. value is numeric or not
6. if (no numeric value)
   then
   compute value for without splitting Gini method no found every node value
   else
   yes (value is numeric)
   then
7. partition S into $S_1$,.....,$S_n$ and label node T
   endelseif
8. find split value

**Table 1:** Represent Dataset

| *Software Defect Open-Source Data Set* | |
|---|---|
| Attributes | 17 |
| Instances | 435 |
| Sum of Weight | 435 |

**Table 2:** Confusion Matrix

| | | *Predicted* | | | | | |
|---|---|---|---|---|---|---|---|
| | | *J48* | | *RF* | | *LMT* | |
| | *WEKA* | *NO* | *YES* | *NO* | *YES* | *NO* | *YES* |
| | NO | 259 | 8 | 259 | 8 | 259 | 8 |
| *Actual* | YES | 8 | 160 | 9 | 159 | 6 | 162 |

**Table 3:** Accuracy of ML Classifier Algorithms

| *Classifier Algorithms* | *Instances Correctly Predicted* | *Instances Incorrectly Predicted* | *Accuracy* | *Total Taken to Build Model (S)* |
|---|---|---|---|---|
| J48 | 419 | 16 | 96.32 | 0.06s |
| RF | 418 | 17 | 96.02 | 0.10s |
| LMT | 421 | 14 | 96.78 | 0.37s |

**Table 4:** Misclassification Error Rate

| *Error Rate* | *J48* | *RF* | *LMT* |
|---|---|---|---|
| | 3.67 | 3.90 | 3.21 |

9. create leaf node
10. value right build tree
11. condition must be true
12. then return step 9
13. find each node value (tree T)
14. end if

## ANALYSIS OF EXPERIMENT WORK

The open-source data repository that contains 17 attributes and 435 in stances respectively. WEKA (Schneidewind, 2002) tool have been applied on"VOTE" data set taking standard 10-fold cross validation for performance evaluation of the different algorithms (Table 1). Table 2 reveals confusion matrix for mentioned three algorithms, which maps the actual and predicted values for the respective algorithms. In Table 3 and 4 authors have calculated different parameters values and accuracy value and they have found that the accuracy of the LMT is 96.78% which is best among J48, and RF methods of classification.

## CONCLUSIONS

In this paper presents on detailsreview of CART approach to basic new idea a CART of an approach and introduces the binary recursive partitioning method. This idea to represent a novel algorithm which produces better approach as compare to other prediction techniques. We observed that LMT is having maximum accuracy and minimum error rate. On the basis of different parameter measures.

## REFERENCES


1. Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6), 716-723.
2. Basili, V.R., Briand, L.C. and Melo, W. (1996). A validation of object-oriented design metrics as quality indicators. *IEEE Trans. Software Engineering*, 22, 751–761.
3. Briand, L. C., Basili, V. R. and Hetmanski, C. J. (1993). Developing interpretable models with optimized set reduction for identifying high-risk software components. IEEE Transactions on Software Engineering, 19, 1028-1044, https://doi.org/10.1109/32.256851
4. Briand, L.C., Basili, V.R. and Thomas, W.M. (1992). A pattern recognition approach for software engineering data analysis. *IEEE Trans. Software Engineering*, 18, 931–942.
5. Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984). Classification and Regression Trees. *Chapman & Hall, London*.
6. Clark, L. A. and Pregibon, D. (1992). Tree-based models: Statistical Models in S. *Wadsworth, Pacific Grove, California,* 377-419.
7. Ebert, C. (1996). Classification techniques for metric-based software development. Software Quality J., 5, 255–272.
8. Gokhale, S. S. and Lyu, M. R. (1997). Regression tree modeling for the prediction of software quality. *Proceedings of the Third ISSAT International Conference on Reliability and Quality in Design, Anaheim, CA,* 31-36.
9. Guo, P. and Lyu, M. (2000). Software quality prediction using mixture models with EM algorithm. Proceedings First Asia-Pacific Conference on Quality Software, Hong Kong, China, 69-78, DOI: 10.18091/ijsts.51200 10.1109/APAQ.2000.883780.
10. Gupta, D. L., Malviya, A. K. and Singh, S. (2012). Performance analysis of classification tree learning algorithms. *International Journal of Computer Applications,* 55(6).
11. Gyimothy, T. and Ferenc, R. (2005). Empirical validation of object oriented metrics on open source software for fault prediction. *IEEE Transactions on Software Engineering*, 31 (10), 897-910.
12. Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied Statistics,* 29**,** 119-127**.**
13. Khan, M. J., Shamail, S., Awais, M. M., and Hussain, T. (2006). Comparative study of various artificial intelligence techniques to predict software quality. *IEEE International Multitopic Conference, Islamabad, Pakistan*, 173-177, DOI: 10.18091/ijsts.51200 10.1109/INMIC.2006.358157.
14. Khoshgoftaar T. M., Allen E.B. and Deng, J. (2001). Controlling overfitting in software quality models: Experiments with regression trees and classification. *IEEE Computer Society*, 190-198.
15. Khoshgoftaar, T. M., Allen, E.B. and Deng, J. (2002). Using

regression trees to classify fault-prone software modules. *IEEE Trans. Reliability*, 51(4), 455-462.

16. Khoshgoftaar, T. M., Seliya, N. and Herzberg, A. (2005). Resource-oriented software quality classification models. *The Journal of Systems and Software*, 76(2), 111-126.
17. Khoshgoftaar, T. M., Allen, E.B., Kalaichelvan, K. S. and Goel, N. (1996). Early quality prediction: A case study in telecommunications. *IEEE Software*, 13, 65–71.
18. Khoshgoftaar, T.M. and Lanning, D.L. (1995). A neural network approach for early detection of program modules having high risk in the maintenance phase. *J. Systems and Software,* 29, 85–91.
19. Khoshgoftaar, T.M. and Seliya, N. (2004). Comparative assessment of software quality classification techniques: An empirical case study. *Empirical Software. Engg*, 9(3), 229-257.
20. Khoshgoftaar, T.M. and Allen, E.B. (1999). Logistic regression modeling of software quality. *International Journal of Reliability, Quality and Safety Engineering*, 6(4), 303-317.
21. Khoshgoftaar, T. M., Allen, E. B., Naik, A., Jones, W. D. and Hudepohl, J. P. (1999). Using classification trees for software quality models: Lessons learned. *International Journal of Software Engineering and Knowledge Engineering*, 9(2), 17-23.
22. Khoshgoftaar, T. M., Yuan, X. and Allen, E. B. (2000). Balancing misclassification rates in classification-tree models of software quality. *Empirical Software Engineering: An International Journal,* 5(4), 313-330.
23. Khoshgoftaar, T. M., Allen, E. B., and Naik, A. (1998). Using classification trees for software quality models: Lessons learned. *Proceeding Third IEEE Conference. High-Assurance Systems Engineering, Bethesda, USA,* 82–89.
24. Kitchenham, B. A. (1998). A procedure for analyzing unbalanced datasets. *IEEE Trans. Software Engineering*, 24, 278–301.
25. Naik, A. (1998). Prediction of software quality using classification tree modeling. *Master's Thesis, Florida Atlantic University, Boca Raton*.
26. SAS Institute staff. TREEDIS Cacro (beta version). (1995). Technical report, SAS Institute, Inc., Cary, NC, Documentation with macros.
27. Schneidewind, N.F. (2002). Body of knowledge for software quality measurement. *IEEE Computer*, 35(2), 77-83.
28. Schneidewind, N.F. (1995). Software metrics validation: Space Shuttle flight software example. *Annals of Software Engineering*, 1, 287–309.
29. Troster, J. and Tian, J. (1995). Measurement and defect modeling for a legacy software system. *Annals of Software Engineering*, 1, 95-118.