Measurement of Bug in Documentary Specification of Software Requirement by Data Mining

Dhyan Chandra Yadav, Saurabh Pal

Abstract— Documented software specification is often missing, incomplete or outdated. This causes difficulties in software maintenance efforts especially when a software project involves many people or the project continues over a long period of time. Lack of documented software specification also causes difficulty in testing or verifying the correctness of a software system. In this paper, classification of the data collected from a software project or department in MASC that raises the problem report has been discussed. This data is pre-processed to remove unwanted and less meaningful attributes. These software requirements specifications are then classified into different categories like low, average, high. The processing is done using WEKA data mining tool and compares results of classification with respect to different performance of parameters. This paper has presented the combination of Software Engineering with Data Mining Techniques. In this paper we classified and detect software requirements specification defect in data set by Lad Tree, Random Tree and J48 of data mining. Experimental results show the performance of, Accuracy, Recall, probability of detection, probability of false Alarm, Type-I error, Type-II error and overall misclassification rate parameter takes care of these two error parameters.

Index Terms— Classification Algorithms: Lad Tree, Random Tree, J48 and Weka.

---

1 Introduction

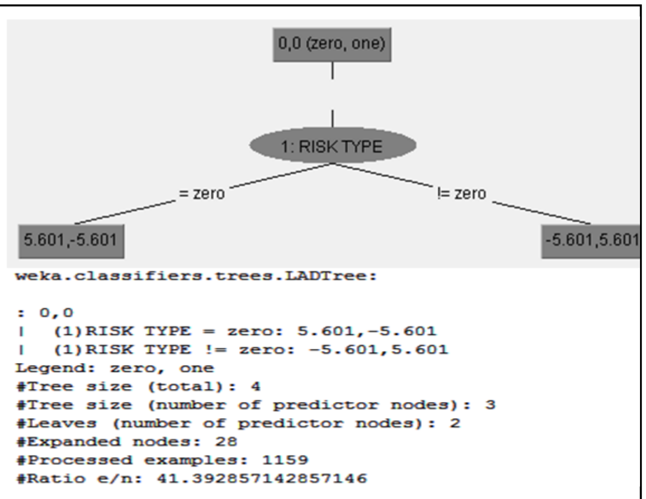
All programs and software developments are established with strong, accurate and documented qualifications. Software products frequently arise with reduced, imperfect and uniform no documented specification. This condition is supplementary intensified by the singularity termed as software development. Software progresses the documented specification is often not updated. This might concentrate the unique specification of slight use after several sequences of sequencer progression. This is particularly accurate for software developments recognized by many developers over a long period of time. Considering the above influences, falling conservation charge by addressing the problem of deficiency, imperfect and invalid specification can possibly save a large amount of unexploited incomes.

1.1 Classification

Tiwari and Chaudhary [1] introduced about decision tree classification algorithm is widely used in statistics, data mining, machine learning. The goal is to create a decision tree model, which uses the given input data to predict the target data classification. For the nodes within the tree, we compare the attribute values. Each branch is a possible classification for the target data. Leaf node is the classification of the target data. In order to classify the unknown target data, target data attribute value judgment on the decision tree. The determination of the process can be represented as the leaf node with the path, which corresponds to a classification rule. In this paper we choose the three algorithms for analyzing documentary requirement specification at different class level and for example comparing following algorithms which are describe below.

1.2 LAD Tree

Boros, Hammer, T. Ibaraki, Kogan, Mayoraz, and Muchnik introduced about [2] software requirements specifications are then classified into different categories like risk or non-risk an each parameter in data set have his specific meaning. The attribute values in each parameter (zero, one and two) have different value. From the figure 1 it is clear that classifier for binary target variable based on learning a logical expression that can distinguish between positive and negative samples in a data set.



```
weka.classifiers.trees.LADTree:

: 0,0
|   (1)RISK TYPE = zero: 5.601,-5.601
|   (1)RISK TYPE != zero: -5.601,5.601
Legend: zero, one
#Tree size (total): 4
#Tree size (number of predictor nodes): 3
#Leaves (number of predictor nodes): 2
#Expanded nodes: 28
#Processed examples: 1159
#Ratio e/n: 41.392857142857146
```

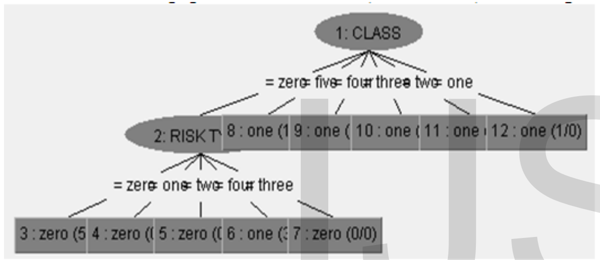
Fig.1. Classification of attributes by LAD Tree

- *Dhyan Chandra Yadav is currently pursuing Ph. D. program in Computer Science, UP, India, E-mail: dc9532105114@gmail.com*
- *Saurabh Pal is currently Working as Assistant Professor, MCA Dept., VBS Purvanchal University, Jaunpur, India, PH-09889807337. E-mail: drsaurabhpal@yahoo.co.in*

The basic assumption of LAD model is that a binary point covered by some positive patterns, but not covered by any negative pattern is positive, and similarly, a binary point covered by some negative patterns, but not covered by positive pattern is negative. The building of LAD model for a given data set classically includes the group of great set designs and the choice of a subgroup of them that pleases the above supposition such that each design in the classical pleases positive necessities in terms of occurrence and similarity. For example we represent Lad Tree classification.

1.3 Random Tree

A random tree [3] is a collection (ensemble) of tree predictors that is called forest. Software requirements specifications are then classified into different categories like low, average, high and each parameter in data set have different value. All the trees are trained with the same parameters but on different training sets. These sets are generated from the original training set using the bootstrap procedure: for each training set, you randomly select the same number of paths as in the creative set. The paths are selected with replacement i.e., some paths will occur more than once and some will be absent.



```
RandomTree
==========

CLASS = zero
|   RISK TYPE = zero : zero (52/0)
|   RISK TYPE = one : zero (0/0)
|   RISK TYPE = two : zero (0/0)
|   RISK TYPE = four : one (3/0)
|   RISK TYPE = three : zero (0/0)
CLASS = five : one (1/0)
CLASS = four : one (2/0)
CLASS = three : one (1/0)
CLASS = two : one (1/0)
CLASS = one : one (1/0)

Size of the tree : 12
```

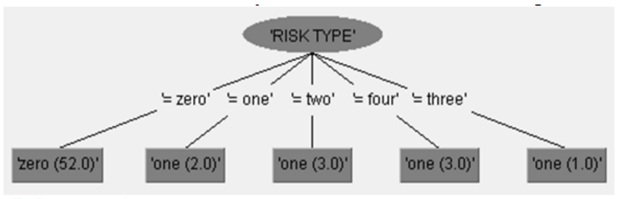
Fig. 2. Classification of attribute by Random Tree

At each node of each trained tree, not all the variables are used to find the best fragmented, but a random subdivision of them. With each node a new subdivision is produced.

1.4 J48 Tree

Hunt and Quinlan presents [4] **C4.5** is a successor of ID3.The **C4.5** is represents by J48 in weka.J48 classification by decision tree leaf nodes represent class level:

1. A flow chart like tree structure internal node denotes a test.
2. On an attribute branch represents an out comes of the test.
3. Decision tree generate consists of two phases.



```
J48 pruned tree
------------------

RISK TYPE = zero: zero (52.0)
RISK TYPE = one: one (2.0)
RISK TYPE = two: one (3.0)
RISK TYPE = four: one (3.0)
RISK TYPE = three: one (1.0)

Number of Leaves  :     5

Size of the tree :      6
```

Fig. 3. Classification of attribute by J48.

2 Related Work

Shepperd, Schofield and Kitchenham [7] discussed that need of cost estimation for management and software development organizations and give the idea of prediction also give the methods for estimation.

Pal and Pal [8] conducted study on the student performance based by selecting 200 students from BCA course. By means of ID3, c4.5 and Bagging they find that SSG, HSG, Focc, Fqual and FAIn were highly correlated with the student academic performance.

Alsmadi and Magel [9] discussed that how data mining provide facility in new software project its quality, cost and complexity also build a channel between data mining and software engineering.

Yadav and Pal [10] use the ID3 decision tree to generate the important rules that can help to predict student enrollment into an academic programme called the Master of Computer Application. The generated tree yields that Bachelor of Science students in mathematics and computer applications will enroll and will likely to perform better as compared to Bachelor of Science students without any background in mathematics.

Boehm, Clark, Horowitz, Madachy, Shelby and Westland [11] discussed that some software companies suffer from some accuracy problems depend on his data set after prediction software company provide new idea to specify project cost schedule and determine staff time table.

K.Ribu [12] discussed that the need of open source code projects analyzed by prediction and get estimating object oriented software project by case model.

Nagwani and Verma [13] discussed that the prediction of software defect (bug) and duration similar bug and bug average in all software summery, by data mining also discuss about software bug.

Yadav and Pal [14, 15] discussed the use of different classification algorithms using standard quality of software data sets and compared the accuracy level of each method.

Hassan [16] discussed that the complex data source(audio, video, text etc.) need more of buffer for processing it does not support general size and length of buffer.

Li and Reformate [17] discussed that .the software configuration management a system includes documents, software code, status accounting, design model defect tracking and also include revision data.

Elcan [18] discussed that COCOMO model pruned accurate cost estimation and there are many thing about cost estimation because in project development involve more variable so COCOMO measure in term effort and metrics.

Chang and Chu [19] discussed that for discovering pattern of large database and its variables also relation between them by association rule of data mining.

Kotsiantis and Kanellopoulos [20] discussed that high severity defect in software project development and also discussed the pattern provide facility in prediction and associative rule reducing number of pass in database.

Chaurasia and Pal [21, 22] conducted study on the prediction of heart attack risk levels from the heart disease database with data mining technique like Naïve Bayes, J48 decision tree and Bagging approaches and CART, ID3 and Decision Table. The outcome shows that bagging techniques performance is more accurate than Bayesian classification and J48.

Pannurat, N.Kerdprasop and K.Kerdprasop [23] discussed that association rule provide facility the relationship among large dataset as like software project term hug amount , cost record and helpful in process of project development.

Fayyad, Piatesky Shapiro, Smuth and Uthurusamy [24] discussed that classification creates a relationship or map between data item and predefined classes.

Shtern and Vassillios [25] discussed that in clustering analysis the similar object placed in the same cluster also sorting attribute into group so that the variation between clusters is maximized relative to variation within clusters.

Runeson and Nyholm [26] discussed that code duplication is a problem which is language independent. It is appear again and again another problem report in software development and duplication arises using neural language with data mining.

Vishal and Gurpreet [27] discussed that data mining analyzing information and research of hidden information from the text in software project development.

Lovedeep and Arti [28]data mining provide a specific platform for software engineering in which many task run easily with best quality and reduce the cost and high profile problems.

Nayak and Qiu [29] discussed that generally time and cost, related problems arises in software project development these problems mentation in problem report ,data mining provide help in to reduce problems also classify and reduce another software related bugs .

The proposed system will analyze risk of software defects predicts. Predicts categorical class level classifiers based on training set and the values in the class level attribute use the model in classifying new data. We compare between AD Tree, RFP Tree and LAD Tree for probability of detection, probability of false alarm, geometric mean, J static coefficient parameter (for sensitivity and specificity),correctness, completeness, average absolute error and average relative error.

3 Methodology

Our research approach is to use J48, Random Tree and LAD Tree; to model the relationships between the measurable properties of a software product and its quality. The research methodology is divided into 6 steps to achieve the desired results:

Step 1: In this step, prepare the data and specify the source of data.

Step 2: In this step select the specific data and transform it into different format by weka.

Step 3: In this step, implement data mining algorithms and checking of all the relevant bugs and errors is perform.

Step 4: **T**he decision is taken on the presence of bugs in source code. If Bug is present then proceed further, otherwise it will stop.

Step 5**:** We classify the relevant bugs using J48, Lad tree and Random Tree algorithm at particular time.

Step 6**:** At the end, the results are display and evaluated. data.

3.1 Data Preparation

In this step only those fields were selected which were required for data mining. A few derived variables were selected. Where some of the information for the variables was extracted from the dataset. All the response variables which were derived from dataset parameter are given in table 1 for reference. Table.2. represents explanatory variables severity, state, time to fixed, priority and risk type. All variables have his specific meaning and corresponding values.

Class (0) =Fully Specify , Class (1) = Not Fully Specify

The domain values for some of the variables were defined in the Table 1 for the present investigation. In our comparison we contain details documented requirement specification the

TABLE 1
REPRESENTS SRS DOCUMENTARY PARAMETERS USED IN THE COMPUTATIONAL TECHNIQUE

PROPERTY	DESCRIPTION
SOURCE	Name of a project or department in MASC that raises the PR.
MEASUREMENT TYPE	(Duplicate-Bug)Srs With Metrics Count
SAMPLE SIZE	61 TOTAL:9 SRS not specify means BUG arises and 52 SRS specify means NONBUG arises in software bug-tracking system,
DEPENDABLE VARIABLE	Description
SRS(0)	0= Software Requirement Specify.
SRS (1)	1=Software Requirement not fully Specify.

error arises in problem report. Documented Specification provide a specific environment to develop software quality but in the absent of fully documented specification required project cannot successfully performance. Bug is tracked by GANTS which is a bug tracking system in GNU. It is set on MASC intranet to collect and maintain all problem reports from every department of MASC. The documented requirement bug creates in software document categories by class field.

Table 2 represents explanatory variables Reusable, Correct, Complete, Verifiable, Observable, Not Redundant, Project Self Governing and Unambiguous. All variables have his specific meaning and corresponding values.

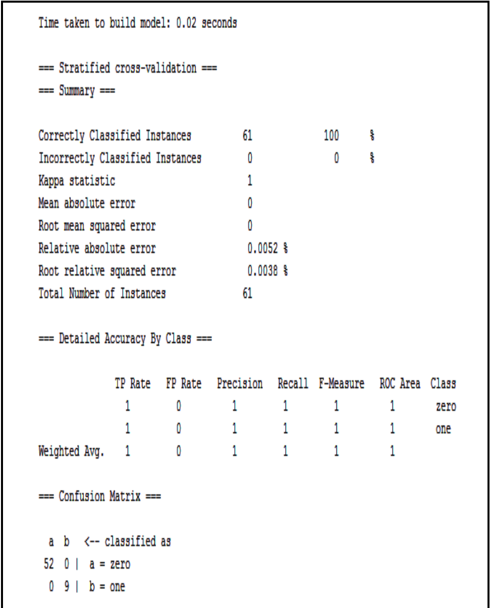
3.2 Data Selection and Transformation

Once Predictive model is created, it is necessary to check

TABLE 2
THE SOFTWARE BUG EXPLANATORY VARIABLES USED IN THE COMPUTATIONAL TECHNIQUE

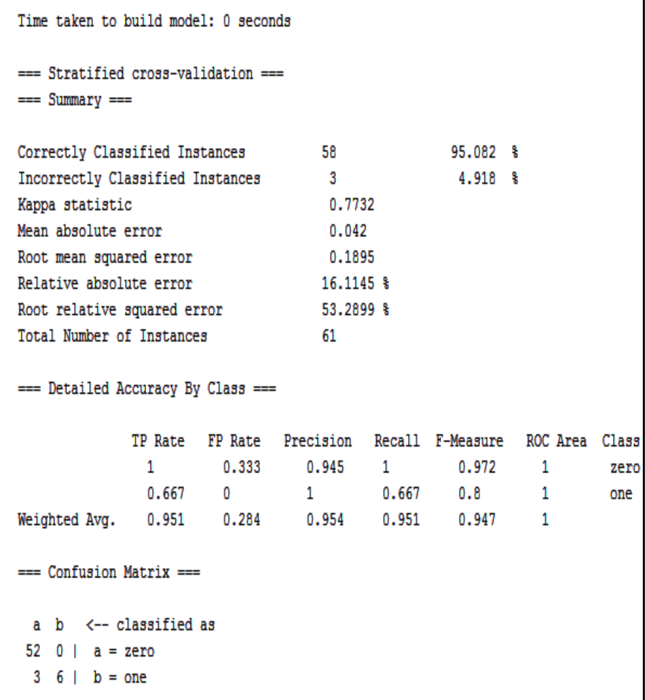
Explanatory Variable	Description
Severity	{1=normal,0=serious}describe the severity of problem report
Class	{0=sw-bug, 1=doc bug,2=change request,3=support,4=mistaken,5=duplicate}category of bug class
State	{0=closed,1=open,2=active,3=analysed,4=suspended,5=resolved,6=feedback}status of problem report analysis/non analysis
Time To Fix	{0=withintwodays, 1=within one week,2=within two week,3=within three week,4=within four week,5=within five week}take time duration in of problem report
Priority	{0=not,1=high,2=medium,3=low}describe schedule permit duration
Risk Type	{0=not,1=high,2=midium,3=low,4=cosmetic}risks can be defined as uncertainty and loss in project process.

how accurate it is, The Accuracy of the predictive model is calculated based on the Recall, probability of detection, probability of false Alarm. The calculation is based on true positives, false positives, true negative, false negative values and measurement of these values depends on confusion matrix. Accuracy is the overall correctness of the model and is calculated as the sum of correct classifications divided by the total number of classifications.



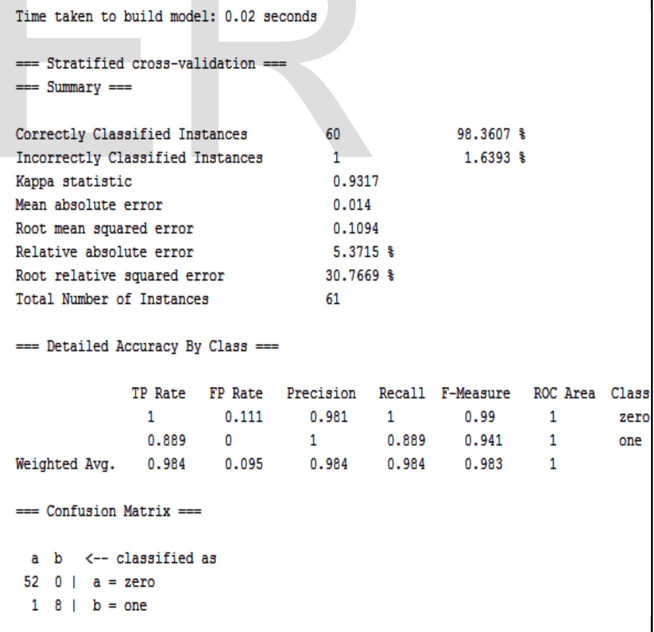
```
Time taken to build model: 0.02 seconds

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

Correctly Classified Instances          61              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0.0052 %
Root relative squared error              0.0038 %
Total Number of Instances               61

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          1        zero
                 1         0          1         1         1          1        one
Weighted Avg.    1         0          1         1         1          1

=== Confusion Matrix ===

  a  b   <-- classified as
 52  0 |  a = zero
  0  9 |  b = one
```

Fig. 4. Instances Classified by Lad Tree



```
Time taken to build model: 0 seconds

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

Correctly Classified Instances          58               95.082  %
Incorrectly Classified Instances         3                4.918  %
Kappa statistic                          0.7732
Mean absolute error                      0.042
Root mean squared error                  0.1895
Relative absolute error                 16.1145 %
Root relative squared error             53.2899 %
Total Number of Instances               61

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0.333      0.945     1         0.972      1        zero
                 0.667     0          1         0.667     0.8        1        one
Weighted Avg.    0.951     0.284      0.954     0.951     0.947      1

=== Confusion Matrix ===

  a  b   <-- classified as
 52  0 |  a = zero
  3  6 |  b = one
```

Fig. 5. Instances Classified by Random Tree



```
Time taken to build model: 0.02 seconds

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

Correctly Classified Instances          60               98.3607 %
Incorrectly Classified Instances         1                1.6393 %
Kappa statistic                          0.9317
Mean absolute error                      0.014
Root mean squared error                  0.1094
Relative absolute error                  5.3715 %
Root relative squared error             30.7669 %
Total Number of Instances               61

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0.111      0.981     1         0.99       1        zero
                 0.889     0          1         0.889     0.941      1        one
Weighted Avg.    0.984     0.095      0.984     0.984     0.983      1

=== Confusion Matrix ===

  a  b   <-- classified as
 52  0 |  a = zero
  1  8 |  b = one
```

Fig. 6. Instances Classified by J48.

3.3 Data Mining Implementation

The paper presents an approach to classifying the Documentary Specification Category in order to predict .They design, implement and evaluate a series of classifier. The classifiers were used to declare surety of bug. They used the Lad Tree, Random Tree and J48 algorithms to improve the prediction accuracy. This method is of considerable useful in identifying bug in very large data set. Weka is open source software that implements a large collection of machine learning algorithms and is widely used in data mining applications. From the above data bug.arff file was created. This file was loaded into weka explorer and analyzes SRS category for required software prediction. Predicts categorical class level classifiers based on training set and the values in the class level attribute use the model in classifying new data. The problem in particular is a comparative study of performance of different classifier such as LAD tree, Random Tree and J48 by using various parameters of documentary specification category, data set containing 9 attributes, 61 instances. To investigate further the classifier performance in classification accuracy, Recall, PF, PD, Type-I Error, Type-II Error, and Overall Classifications rate. Visualized these parameters into tables and graphs for their performance.

3.4 Result and Discussion

The Performance of the given work is determined on the basis of different parameters- Accuracy, Recall, probability of detection, probability of false Alarm, Type-I error, Type-II error and overall misclassification rate parameter takes care of these two error parameters to evaluate the Performance of fault prediction models . Proposed work is compared with the different class of specification of documented software requirement.

PD=recall={D/(C+D)}=TP/(TP+FN) -(1)
PF={B/(A+B)}=FP/(FP+TN) -(2)
Accuracy=(A=D)/(A+B+C+D) -(3)

Some researchers used Type-I error and Type-II error parameters to evaluate the performance of fault prediction mod-

TABLE 3

CLASSIFICATION ACCURACY BY LAD TREE, RANDOM TREE AND J48 ALGORITHMS

Algorithms	PD	PF	Accuracy
Random tree	0.66	0	0.95
Lad Tree	1.00	0	1.00
J48	0.88	0	0.98

els .The overall misclassification rate parameter takes care of these two error parameters. Formulas 4, 5, and 6 are used to calculate the Type-I error, Type-II error, and overall misclassification rate respectively. If a non-faulty module is predicted as a faulty module, a Type-I error occurs, and if a faulty module is predicted as a non-faulty module, a Type-II error occurs. A Type-II error is more significant than a Type-I error because faulty modules cannot be detected in that case.

Type-I error ={B/(A+B+C+D)}=FP/(TN+FP+FN+TP) -(4)
Type-II error ={C/A+B+C+D}=FN/(TN+FP+FN+TP) -(5)
Overall misclassification rate =(C+B)/(A+B+C+D) -(6)

TABLE 4
ERROR CLASSIFICATION BY LAD TREE, RANDOM TREE AND J48 ALGORITHMS

Algorithms	Type-I Error	Type-II Error	Overall Misclassification rate
Random Tree	0	0.04	0.04
Lad Tree	0	0.00	0.00
J48	0	0.01	0.01

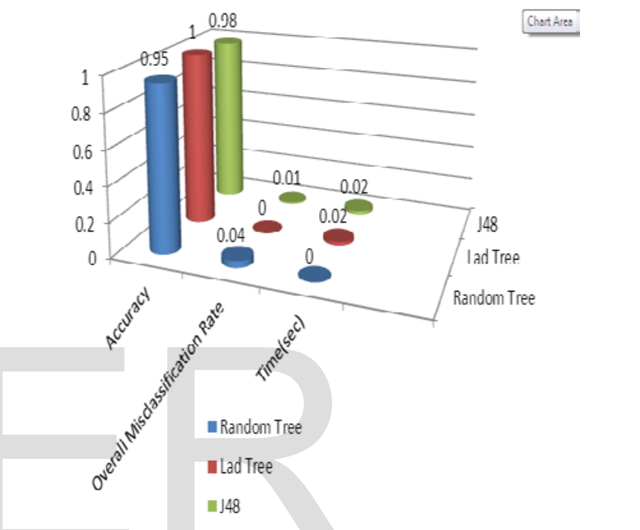


Fig. 7. Graph Represents the Performance of J48, Lad Tree and Random Tree .

Given Table 3, 4 and graph shows the comparison between the classified attribute by weka tool. From table and graph it is clear-

- Probability of detection of Lad Tree is high compare to Random Tree and J48.
- Probability of false Alarm is zero for all tree algorithms.
- Accuracy level Lad tree is high compare to Random and J48.
- All three algorithms have zero level for Type-I error.
- Lad Tree has less error level for Type-II error compare to other two algorithms.
- Overall misclassification rate level is low compare to other two algorithms.
- Lad Tree algorithm has time taken to build a model is less high compare to other two algorithms.

4 CONCLUSION

In this paper were done experiments with Weka Machine Learning Tool in order to choose the best Data Mining algorithms to be applied over selected datasets. In this paper three different classifiers are applied on data set for detect the best results. We measure Accuracy, Recall, Overall Misclassifications and Time taken to build model parameters are used for performance evaluation and graphs are plotted. The results confirm that for LAD Tree algorithms is a best classifier in comparison of Random Tree and J48 algorithms.

In future classification algorithms can be applied. It will include other types of errors like logical and find more accurate value.

REFERENCES

[1] Sunita Tiwari and Neha Chaudhary "Data mining And Warehousing" Dhanpati Rai and Co.(P) Ltd. First edition: 2010. Kaufmann Publishers Third Edition. 2012.

[2] E. Boros, P. L. Hammer, T. Ibaraki, A. Kogan, E. Mayoraz, and I. Muchnik, "An implementation of logical analysis of data," *IEEE Trans. On Knowl. Data Eng*, vol. 12, no. 2, pp. 292–306, 2000.

[3] Wikipedia contributors, "Random_tree," Wikimedia Foundation, 13-Jul-2014.

[4] J. R. Quinlan, C4.5: programs for machine learning, Morgan Kaufmann,San Francisco,1993.

[5] M. Shepperd, C. Schofield, and B. Kitchenham, "Effort estimation using analogy," in of the 18th International Conference On Software Engineering, pp.170-178. Berlin, Germany, 1996.

[6] Alsmadi and Magel, "Open source evolution Analysis," in proceeding of the 22nd IEEE International Conference on Software Maintenance (ICMS'06), phladelphia, pa.USA, 2006.

[7] M. Shepperd, C. Schofield, and B. Kitchenham, "Effort estimation using analogy," in of the 18th International Conference on Software Engineering,pp.170- 178. Berlin, Germany, 1996.

[8] Pal, A. K., & Pal, S., "Classification model of prediction for placement of students", International Journal of Modern Education and Computer Science (IJMECS), 5(11),2013, 49.

[9] Alsmadi and Magel, "Open source evolution Analysis," in proceeding of the 22nd IEEE International Conference on Software Maintenance (ICMS'06), phladelphia, pa.USA, 2006.

[10] Yadav, Surjeet Kumar, and Saurabh Pal. "Data mining application in enrollment management: A case study." International Journal of Computer Applications (IJCA) 41.5, 2012, 1-6.

[11] Boehm, Clark, Horowitz, Madachy, Shelbyand Westland, "Cost models for future software lifecy cle Process COCOMO2.0."in Annals of software Engineering special volume on software process and prodocuct measurement, J.D.Arther and S.M.Henry, Eds, vol.1, pp.45-60, j.c. Baltzer AG,science publishers, Amsterdam,The Netherlnds, 1995.

[12] Ribu,Estimating object oriented software projects With use cases,M.S.thesis, University of Oslo Department of informatics, 2001.

[13] N. Nagwani and S. Verma,"prediction data mining Model for software bug estimation using average Weighted similiarity,"In proceeding of advance Computing conference (IACC), 2010.

[14] Yadav, Dhyan Chandra, and Saurabh Pal. "Analysis Receiver Operating Characteristics of Software Quality Requirement by Classification Algorithms Analysis" IJCA 116.8, 2015.

[15] Yadav, Dhyan Chandra, and Saurabh Pal. "Software Bug Detection using Data Mining." IJCA, 115.15, 2015

[16] A.E.Hassan ,"The road ahead for mining software Repositories", in processing of the future of software Maintenance at the 24th IEEE international Conference on software maintenance, 2008.

[17] Z.Li and Reformat,"A practical method for the Software fault prediction",in proceeding of IEEE Nation conference information reuse and Integration (IRI), 2007.

[18] C.Elcan,"The foundations of cost sensitive learning In processing of the 17 International conference on Machine learning, 2001.

[19] C.Chang and C.Chu,"software defect prediction Using international association rule mining", 2009.

[20] S.Kotsiantis and D.Kanellopoulos,"Associan rule mining:Arecent overview",GESTS international transactiona on computer science science and Engineering, 2006.

[21] Chauraisa V. and Pal S., "Data Mining Approach to Detect Heart Diseases", International Journal of Advanced Computer Science and Information Technology (IJACSIT),Vol. 2, No. 4,2013, pp 56-66.

[22] Chauraisa V. and Pal S., "Early Prediction of Heart Diseases Using Data Mining Techniques", Carib.j.SciTech,,Vol.1, pp. 208-217, 2013.

[23] N.Pannurat,N.Kerdprasop and K.Kerdprasop"Database reverses engineering based On Association rule mining",IJCSI international Journal Of computer science issues 2010.

[24] U.M.Fayyad,G Piatesky Shapiro,P.Smuth and R.Uthurusamy,"Advances in knowledge discovery And data mining",AAAI Press,1996.

[25] M.Shtern and Vassilios,"Review article advances in Software engineering clustering methodologies for Software engineering",Tzerpos volume,2012.

[26] P.Runeson and O.Nyholm,"Detection of duplicate Defect report using neural network processing",in Proceeding of the 29th international conference on Software engineering 2007.

[27] G.Vishal and S.L. Gurpreet,"A servey of text mining Techniques and applications",journal of engineering Technologies in web intelligence, 2009.

[28] Lovedeep and VarinderKaurArti" Application of Data mining techniques in software engineering"International journal of electrical,electronics and computer system(IJEECS) Volume-2 issue-5, 6. 2014.

[29] Richi Nayak andTianQiu"Adata mining application"international journal of software engineeringandKnowledgeengineering volume.15, issue-04,2005.