# Analysis of Software Deformity Prone Datasets with Use of AttributeSelectedClassifier

Maaz Rasheed Malik[1], Liu Yining[2]
School of Information Communication Engineering
Guilin University of Electronic Technology
Guilin China

Salahuddin Shaikh[3]
School of Control & Computer Engineering
North China Electric Power University
Beijing, China

***Abstract*—Software Deformity Prone datasets models are interesting research direction in the era of software world. In this research study, the interest class of software deformity prone is defective model datasets. There are different techniques to predict the deformity prone datasets model. Our proposed solution technique is AttributeSelectedClassifier with selected evaluators and searching method for reducing the dimensionality of training and testing data provided by defected models NASA datasets by attribute selection before being passed on classifiers. We have used three evaluators and search methods. These evaluators are CFSSubsetEval, GainRatio and Principal Component Analysis (PCA). The search methods are BestFirst and Ranker. We have used 12 different classifiers for analyzing the performance of these three evaluators with search methods. The experimental results and analysis are measured with True Positive (TP-Rate), Positive Accuracy, Area under Curve (ROC) and Correctly Classified Instances. The results showed that that CFSSubsetEval and GainRatio performance is better in almost classifiers. Hoeffding tree, Naive Bayes, Multiclass, IBK and Randomizable filtered class increased performance in Positive Accuracy in all techniques. Stacking has worst performance in positive accuracy and True Positive tp-rate in all over technique.**

***Keywords*—GainRatio; CFSSubsetEval; PCA; classification; defect prediction; deformity prone; defect model; classifier; bug model; softwar; attributesubsetclassifier**

## I. Introduction

The presence of software deformity prone influences significantly on software unwavering quality, quality and support cost. Accomplishing without bug software likewise is diligent work, even the software connected carefully on the grounds that most time there is masked bugs. Notwithstanding, creating software deformity prone model which could foresee the flawed modules in the early stage is a genuine test in software engineering. Software deformity prone is a basic action in software advancement. This is on the grounds that anticipating the surrey modules preceding software organization accomplishes the client fulfillment, improves the general software execution. Also, foreseeing the software bug early improves software adjustment to various conditions and expands the asset use. Different methods have been proposed to handle Software deformity prone issue. The most realized methods are Machine Learning (ML) systems [1]. The Machine Learning procedures are utilized broadly in Software deformity prone to foresee the carriage modules dependent on chronicled issue information, fundamental metrics and distinctive software processing systems.

Software deformity models may show different quality as far as the quantity of faulty models. At the point when faulty models are amazingly uncommon in the dataset, this does not mean these deformity prone ought to be ignored, but instead that it is fundamental to catch them. This is known as the class imbalance issue, since there are a lot more faultless models than faulty ones. We use sampling method to address the class imbalance issue. To be specific, when the proportion of faulty models is extremely low, we use oversampling and under sampling strategies.

Many models impressing examine use NASA Repository data models. This task contains models with surely understood facts, yet in addition model quality issues. Software deformity prone created by classifiers ought to be surveyed as far as accuracy. While much experimental analysis used the area under the curve (AUC) which is also known as Receiver operating characteristic (ROC) curve, we incorporate an option, to be specific the F-measure, to address the potential impediments of the AUC. Our elective measure corrects the misusing of figuring misclassification rates in AUC. While observing the numerical differences in forecast results coming about because of the utilization of different metrics on different models, factual tests check whether these differences are measurably significant. We update the test procedure in the writing to analyze fundamentally whether a classifier does undoubtedly generously outperform another classifier.

The choice of metrics being the first and the chief stage in the software deformity prone has an extraordinary impact in the accuracy and the intricacy of the model. More the quantity of metrics in the model, progressively complex is the procedure Inclusion of insignificant metrics can make the accuracy drop extensively. Additionally, the effect of these metrics can be illogical [2]. It is conceivable that the apparently significant metrics can have less value in the software deformity prone, ascribing to some unpretentious, unanticipated variables. Things being what they are, the determination of suitable metrics is conceivable just by experimental confirmation. Remembering the serious results that the poor metrics choice can cause, we dedicate additional endeavors to deliberately choose the determinant metrics. Bug forecast can be considered as a component of metrics and the idea of this capacity remains obscure. Before, analysts have planned software deformity prone models utilizing determinants like past software deformity prone code stir number of engineers, record length, code refactoring, etc. (see Fig. 1) Subsequent to making a far reaching investigation of

the created models and methodologies, we devise a novel way to deal with gauge the software deformity prone of source code at the class level. Individuals who are taking a shot at software deformity prone for the most part apply open informational index and Artificial Intelligence (AI) techniques for developing unrivaled software deformity prone. Software deformity prone is indispensable and vital activity advantageous to give best excellence and increase the reliability of software before the software is introducing.

In Machine Learning, ML can pursue two diverse learning approaches: supervised and unsupervised Learning. The supervised learning algorithm only works with a bunch of datasets models whose names are mentioned. The Experimental results can be supposed with metrics variables of the classification task, or numerical variables of the regression task [3]. On the other hands unsupervised learning, the names of the models in the dataset models are doubtful, and the algorithm often goes for gathering models as indicated by the closeness of their attribute esteems, describing a clustering task. Supervised algorithm technique can be analyzed by classification task where every instance placed with a class, which is observed by the estimation of a unique independent attribute or basically the class attribute. The objective attribute can take on clear cut qualities, every one of them relating to a class. Every model comprises of two sections, in particular a set of predictor attribute esteems and an objective attribute esteem.

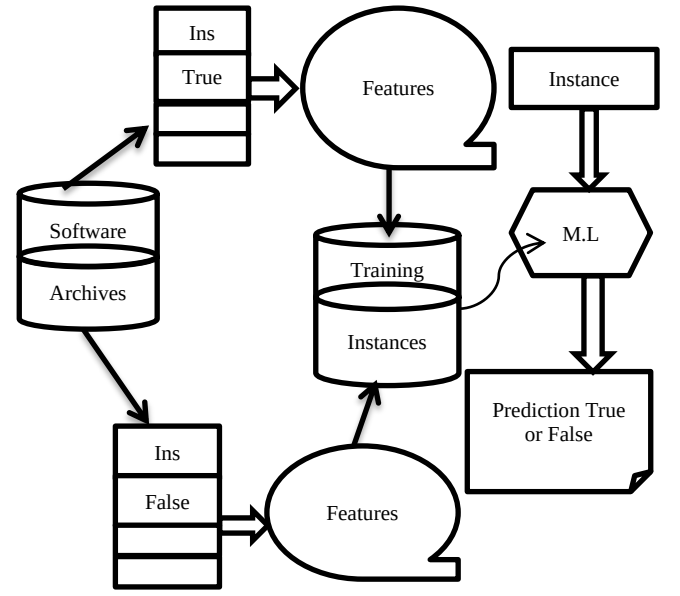


Fig. 1. Flow Chart of Software Deformity Prone Model.

## II. Background

In the era of 2000, the different method metrics were considered. However, in year 2000, there were different impediments for Software deformity Prone. When any item release then it was very difficult to assure the software quality about software metrics. The Software deformity prone datasets couldn't skilled to forecast faults at any point source code which change performs. For resolving this matter, Mockus et al. gave solution datasets model for changes performs. This given solution method was famous as just-in-time (JIT) Software deformity prone datasets model [4]. Mockus proposed solution JIT model been considered advance by various looks that enhanced forecast improvement for modification occurred. Further weakness of Software deformity prone datasets model was to make an imperfection forecast datasets model for different influx task or software with couple of old data. To defeat this issue specialist had done different examinations to construct cross task Software deformity prone model. It raised issue of cross Software deformity prone recognizable proof.

In 1971, Akiyama initially lead the analysis on similar to the quantity of faults and observed source code which caused increasingly quantity of faults. He assured that many line of codes are present in huge Software deformity Prone and these line of codes LOC can measure the complexity of Software deformity Prone. After that he considered these codes as metrics. He proposed and developed primary Software deformity prone model which reliant on the LOC Metrics [5]. For this purpose that he may analyzed where LOC metric is extremely small metric and also measure the complexity of any software. Further overcome this problem, Halsted complexity metric and cyclomatic complexity was given by Halested and McCabe in 1977 and 1976. Which proposed and given individually in era of these years [6]. Once again it needs to be considered in the era of 1970s to 1980, forecast datasets model was not so good to measure the enough quantity of software forecast faults and enhanced the quality of software. This seems to be direct fitting model. This kind of fitting model gives the connection between faults and metrics. All new datasets models are also ignored by fitting model in software deformity Prone. So to overcome this limitation of failure forecast datasets model, a new proposed method given by scientist Shen et al. this proposed model reliant on linear regression. This proposed model also used to test model for new module of software.

Later, few cases were studied by Munson et al. he had been conveyed in these cases that linear regression systems isn't the exact and goof to use for test models or new model datasets. So he proposed and developed another deformity forecast datasets model. This proposed model was reliant on classification approaches. These approaches were order the

datasets model in to two sections usually safe and high hazard. This approaches also proficient and measure accuracy of 92%. These approaches also had faced few arguments and observed that it had no metrics of item located framework and having link of resources for more perfection. One article also considered for arranged framework. This article proposed by Kemerer and Chidamber in 1994. Basically both had proposed many articles set on metrics. One more deformity forecast datasets model proposed by Basili et al. in 1990 [7]. This proposed method reliant on item arranged metrics. This arranged metrics was noticeable to resolve deformity forecast datasets model but however not finally succeeds. So Zimmermann et al. had examined the issue and attempt to make cross task deformity prediction datasets model with classifying fold cross validation forecast and try to make it progressively workable. Pinzer, Taba and Zimmerman et al had contemplated and studied about current pattern of data innovation by social network observation and by network methods. They proposed new idea of Software deformity Prone modified and another was the universal Software deformity prone model.

According to the experimental examinations, a dominant part of software modules don't cause blames in software frameworks, and flawed modules are up to 20% of all the modules. On the off chance that we partition modules into two unique sorts, flawed and non-broken, most of modules will have a place with the non-defective class and the rest will be individuals from the broken class. In this manner, datasets utilized in Software deformity prone investigations are imbalanced. Accuracy parameter can't be utilized for the exhibition evaluation of imbalanced datasets [8]. For instance, a trivial algorithm, which denotes each module as non-defective, can have 90% accuracy if the level of broken modules is 10%. In this manner, specialists utilize various metrics for the validation of Software deformity prone models.

## III. Computational Solution using Classification via Atttribute Selection Evaluators

Before Attribute selection is well known as feature selection, which has been studied in the era of pattern recognition for a considerable length of time. In reverse eradication, e.g., was announced in the early 1960s (Marill and Green, 1963). Kittler (1978) studies the Attribute selection algorithms that have been developed for pattern recognition. Best-first search and genetic algorithms are standard artificial intelligence systems (Goldberg, 1989; Winston, 1992). The attribute selection is the method by which each attribute in your model set is evaluated in the context of the yield variable (e.g. the class) [9]. The search method is the method by which to attempt or navigate different mixes of attributes in the model set so as to arrive on a short rundown of chosen attribute. Some Attribute Evaluator method requires the use of specific Search Methods. Attribute selection is used for decrease the dimensionality, eliminate inappropriate and redundant data. For example, the GainRatio technique used in the this research must be used with a Ranker Search Method that evaluates each attribute and records the results in a rank order. Another example is CFSSubsetEval technique used in this research must be used with Best-First methods that evaluates each attributes and rundown the results according to best-first attribute. When selecting different Attribute Evaluators, the interface may request that you change the Search Method to something compatible with the chosen technique. Both the Attribute Evaluator and Search Method techniques can be configured.

GainRation (GR) measures the analysis of the class deformity prone model sets in bits; the occurrence of an attribute and the corresponding class is exists if the principal class deformity prone model sets is existing. The predictable reduction in entropy is intensely measures [9]. One case that is filter method, which ranks one particular, attributes allowing attributing significance score for attribute selection. But (CFS) method scores and ranks subsets of attribute together, rather than single attribute.

Most techniques for attribute selection include finding the space of attributes down the subset that is well on the way to anticipate the class best. The quantity of potential attributes subsets increments exponentially with the quantity of attributes, making a comprehensive inquiry unreasonable on everything except the least complex issues. The impact of including each attribute thusly is evaluated by this measure, the best one is picked, and the strategy proceeds. In any case, if no attribute creates an improvement when added to the flow subset, the pursuit closes. This is a standard greedy inquiry methodology and certifications to discover a locally—yet not really comprehensively—ideal arrangement of attributes. Here the distinct valuations of use linked attribute to attribute evaluators such as (ReliefF, Gain Ratio, Entropy and so on.) is one kind of Ranker strategy which is used for giving ranked attributes. The parameter of attributes evaluators make placing as like (true or false) number to choose [10]. There is threshold esteems, which is only set the threshold. Because attributes can be disposed of by this threshold esteems. But attributes cannot be disposed by Default esteem. So we use each this substitute or number to select to shrink the attribute set. The classification and variable positioning is known as filter technique. This is a preprocessing stage, which have ability of the choice to predict. The almost portion of ranker method does the rank that which attributes should to be getting high or low position and allowing to select attribute in the given data indexes. Ranking of the attributes is also given by Rankers, but in other hands ranking order given by their ranking to the evaluator.

## IV. Dimensionality Reduction using Attribute Evaluators PCA

PCA is basically to locate a low-dimension model data set of tomahawks that outline model data set. PCA utilizes the change of each feature to do likewise. Principal Component Analysis is an unsupervised Feature Reduction technique for anticipating high dimensional data into another lower

dimensional portrayal of the data that depicts however much of the change in the model data set as could reasonably be expected with least remaking blunder. So, this progress is achieving through comprehensive method Principal Component Analysis. Other sets of variables or values are developed by this approach which is known as Principal components and these are the conventional combination of first variables. Almost PCA are orthogonal to each other, so there is not any redundant data due to this of PCA and main reason for space of the data is also orthogonal structure in all PCA. In this manner we propose unsupervised feature selection calculations dependent on eigenvectors analysis to distinguish basic unique features for principal component.

Envision that the dimensionality of the Attribute model data set is bigger than only a few. Utilizing a PCA we would now be able to recognize what is the most significant dimensions and simply keeps a couple of them to clarify the vast majority of the difference we find in our data. Thus we can radically diminish the dimensionality of the data. In addition, it will likewise empower us to recognize what the most significant factors in the first feature space are, that contribute most to the most significant PCA. Naturally, one can envision, that a measurement that has very little changeability can't clarify a great part of the happenings and hence isn't as significant as increasingly factor dimensions. Since contains the eigenvalues of the correlation matrix, its entrances relates to the differences of the data in a provided guidance [11]. The important part is then simply the left-particular vector scaled by the standard-deviation of the data in the comparing course. In the event that we just need the primary vital segments it gets the job done to increase the model data set with simply the main lines of the right-solitary vectors.

## V. Computational Solution Model Design

We have used 12 different classifiers with NASA PROMISE repository datasets model (Fig. 2). We have used 17 NASA PROMISE repository datasets models. Each datasets model has attributes and class of interest as shown in Table I. The class of interest is defective model and non-defective model. Our experiments performed on defective class model. We have used WEKA tool for performing the experiments results and analysis the observation of each experiments datasets model. Our proposed solution technique is Attribute SelectedClassifier with selected evaluators and searching method for reducing the dimensionality of training and testing data provided by defected models NASA datasets by attribute selection before being passed on these 12 classifiers. We have used four measure units to access the performance of interested class defected models whether the class of interest defected model is still defected or clean. These four measure units are Correctly Classified instances (C.C.I %), TP-Rate, F-Measure (Positive Accuracy) and ROC area under curve. We have used 10-fold cross validation for classification of datasets models. The technique AttributeSelectedClassifier is used with three evaluators and three searching methods where CFSSubsetEval, GainRatio and PCA used as Evaluators. Best first, Random Search and Rankers are used as for searching methods with AttributeSelectedClassifier.

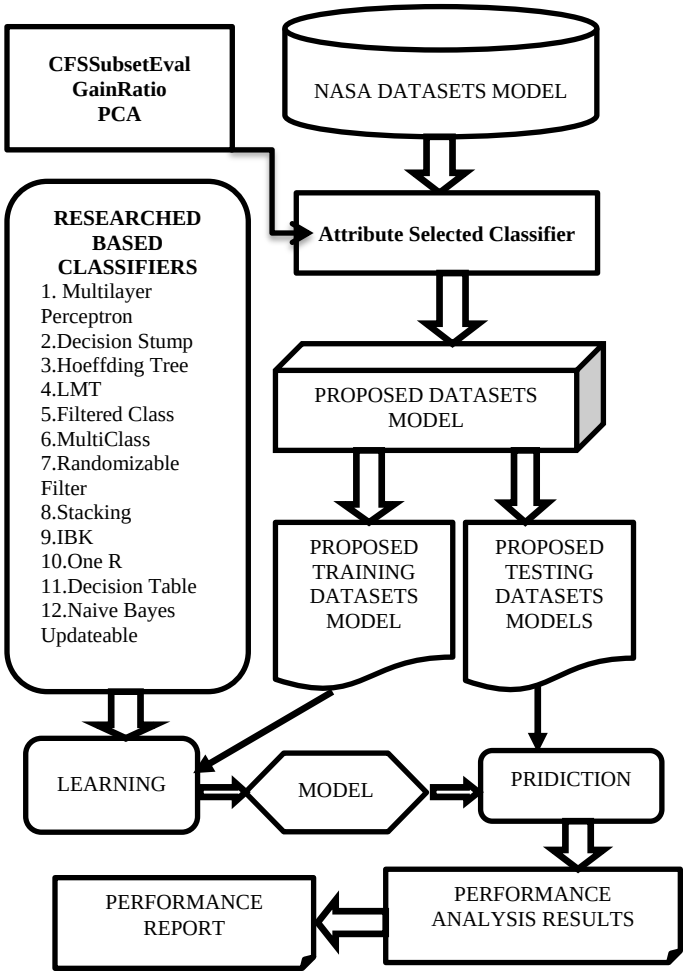


Fig. 2. Proposed Solution Model Design

TABLE. I. NASA PROMISE Repository Datasets Model

| S.NO | Datasets | Attribute | Models | Defective Model | Non-Defective Model |
|---|---|---|---|---|---|
| 1 | AR1 | 30 | 121 | 9 | 112 |
| 2 | AR6 | 30 | 101 | 15 | 86 |
| 3 | AR3 | 30 | 63 | 8 | 55 |
| 4 | AR4 | 30 | 107 | 20 | 87 |
| 5 | AR5 | 30 | 36 | 8 | 28 |
| 6 | CM1 | 38 | 327 | 42 | 285 |
| 7 | JM1 | 22 | 7782 | 1672 | 6110 |
| 8 | KC2 | 22 | 522 | 107 | 415 |
| 9 | KC3 | 40 | 194 | 36 | 158 |
| 10 | MC1 | 39 | 1988 | 46 | 1942 |
| 11 | MC2 | 40 | 125 | 44 | 81 |
| 12 | MW1 | 38 | 253 | 27 | 226 |
| 13 | PC1 | 38 | 705 | 61 | 644 |
| 14 | PC2 | 37 | 745 | 16 | 729 |
| 15 | PC3 | 38 | 1077 | 134 | 942 |
| 16 | PC4 | 38 | 1458 | 158 | 1289 |
| 17 | PC5 | 39 | 17186 | 516 | 16670 |

## VI. Experimental Results

The Research Experimental results and analysis are illustrated in Tables II to V and with respective Fig. 3 to 6. The above tables and following figures explains the performance of software deformity prone datasets models with proposed solution. The above tables and graphs are explained by measure units which are TP-Rate, Area under Curve, Positive Accuracy and Correctly Classified Instances. These all are analyzed by proposed solution which is AttributeSelectedClassifier. This proposed solution contains three techniques. So, all experiments results and analyses in above tables and following graphs are described by these techniques. The table shows results as No-Attribute Selection, CFSSubsetEval, GainRatio and PCA. No-Attribute Selection means, the experiments results are performed by without using any attributeselectedclassifier techniques. CFSSubsetEval, GainRatio and PCA which means that experiments results are performed by attributeselectedclassifier. We have used 12 classifiers for compare the analysis results of these techniques.

In Table II, we have analyzed Correctly Classified Instances Results performance where we have seen that overall performance of GainRatio is better than other two techniques. Hoefdding tree, LMT, Multi Class, Stacking and Decision table, these have better performance in all three techniques.

In Tables III to V, we have analyses the Area under Curve, Positive Accuracy and TP-Rate, where the ROC performance of IBK and Decision stump are decreased in all three techniques. The TP-Rate performance of Multilayer Perceptron, Filtered and One R is also decreased in all three techniques. The Positive accuracy of these three techniques is also increased specially in Hoeffding tree, Naive Bayes, Multi Class, IBK and Randomizable filtered class. Stacking has worst performance in positive accuracy and tp-rate in all over technique. In overall experimental analysis, from Fig. 3 to 6, we illustrated that CFSSubsetEval and GainRatio performance is better in almost classifiers. In case of Correctly Classified Instances performance, where Fig. 3 shows that CFS SubsetEval performance is better in all over the classifiers.

TABLE. II. Correctly Classified Instances Performance

| S.NO | Name: CLASSIFIERS | NO ATTRIBUTE SELECTED C.C.1 % | CFSSubsetEval C.C.1 % | GainRatio C.C.1 % | PCA C.C.1 % |
|---|---|---|---|---|---|
| 1 | Multilayer Perceptro | 74.61% | 86.87% | 89.43% | 81.27% |
| 2 | Decision Stump | 70.63% | 86.76% | 89.86% | 80.45% |
| 3 | Hoeffding Tree | 67.21% | 87.27% | 87.64% | 86.87% |
| 4 | LMT | 72.28% | 86.72% | 87.32% | 86.9% |
| 5 | FILTERE | 76.29% | 87.2% | 87.24% | 80.33% |
| 6 | MULTICLASS | 71.66% | 87.36% | 87.47% | 86.37% |
| 7 | RANDOMIZABL | 71.34% | 83.32% | 89.87% | 83.33% |
| 8 | STACKING | 56.12% | 86.12% | 86.12% | 86.12% |
| 9 | IBK | 61.72% | 83.2% | 89.02% | 83.93% |
| 10 | ONE R | 67.79% | 86.06% | 86.52% | 85.64% |
| 11 | DECISION TABLE | 71.05% | 87.46% | 87.71% | 86.05% |
| 12 | Navie-Bayes-up | 74.56% | 85.97% | 87.49% | 86.25% |

TABLE. III. Area under Curve Performance (ROC)

| S.NO | Name: CLASSIFIERS | NO ATTRIBUTE SELECTED (ROC) | CFSSubsetEval (ROC) | GainRatio (ROC) | PCA (ROC) |
|---|---|---|---|---|---|
| 1 | Multilayer Perceptron | 0.644 | 0.731 | 0.75 | 0.551 |
| 2 | Decision Stump | 0.672 | 0.665 | 0.673 | 0.651 |
| 3 | Hoeffding Tree | 0.574 | 0.594 | 0.599 | 0.564 |
| 4 | LMT | 0.451 | 0.725 | 0.737 | 0.614 |
| 5 | FILTERE | 0.584 | 0.615 | 0.617 | 0.561 |
| 6 | MULTICLASS | 0.625 | 0.76 | 0.765 | 0.758 |
| 7 | RANDOMIZABL | 0.625 | 0.626 | 0.724 | 0.675 |
| 8 | STACKIN | 0.458 | 0.658 | 0.558 | 0.458 |
| 9 | IBK | 0.656 | 0.638 | 0.625 | 0.653 |
| 10 | ONE R | 0.479 | 0.596 | 0.595 | 0.566 |
| 11 | DECISION TABLE | 0.654 | 0.642 | 0.636 | 0.566 |
| 12 | Navie-Bay | 0.552 | 0.732 | 0.739 | 0.725 |

TABLE. IV. F-MEASURE POSITIVE ACCURACY PERFORMANCE

| S.NO | Name: CLASSIFIERS | NO ATTRIBUTE SELECTED F-MEASU | CFSSubsetEval F-MEASURE | GainRatio F-MEASURE | PCA F-MEASURE |
|---|---|---|---|---|---|
| 1 | Multilayer Perceptron | 0.141 | 0.268 | 0.336 | 0.309 |
| 2 | Decision Stump | 0.103 | 0.215 | 0.221 | 0.146 |
| 3 | Hoeffding | 0.156 | 0.216 | 0.223 | 0.16 |
| 4 | LMT | 0.177 | 0.264 | 0.257 | 0.218 |
| 5 | FILTER | 0.166 | 0.253 | 0.263 | 0.168 |
| 6 | MULTICLASS | 0.113 | 0.261 | 0.273 | 0.256 |
| 7 | RANDOMIZABL | 0.217 | 0.312 | 0.322 | 0.341 |
| 8 | STACKIN | 0 | 0 | 0 | 0 |
| 9 | IBK | 0.158 | 0.333 | 0.325 | 0.356 |
| 10 | ONE R | 0.161 | 0.259 | 0.275 | 0.209 |
| 11 | DECISION TABLE | 0.168 | 0.259 | 0.276 | 0.174 |
| 12 | Navie-Bayes-upd | 0.18 | 0.38 | 0.397 | 0.304 |

TABLE. V. TP-RATE PERFORMANCE

| S.NO | Name: CLASSIFIERS | NO ATTRIBUTE SELECTED TP-RATE | CFSSubsetEval TP-RATE | GainRatio TP-RATE | PCA TP-RATE |
|---|---|---|---|---|---|
| 1 | Multilayer Perceptron | 0.292 | 0.211 | 0.278 | 0.264 |
| 2 | Decision Stump | 0.192 | 0.503 | 0.413 | 0.427 |
| 3 | Hoeffding Tree | 0.123 | 0.571 | 0.687 | 0.416 |
| 4 | LMT | 0.214 | 0.606 | 0.592 | 0.361 |
| 5 | FILTERED | 0.213 | 0.205 | 0.218 | 0.141 |
| 6 | MULTICLASS | 0.286 | 0.499 | 0.201 | 0.397 |
| 7 | RANDOMIZABLE FILTER | 0.301 | 0.488 | 0.509 | 0.333 |
| 8 | STACKIN | 0 | 0 | 0 | 0 |
| 9 | IBK | 0.349 | 0.315 | 0.307 | 0.341 |
| 10 | ONE R | 0.208 | 0.203 | 0.324 | 0.173 |
| 11 | DECISION TABLE | 0.208 | 0.197 | 0.216 | 0.117 |
| 12 | Navie-Bayes-upd | 0.44 | 0.366 | 0.416 | 0.254 |

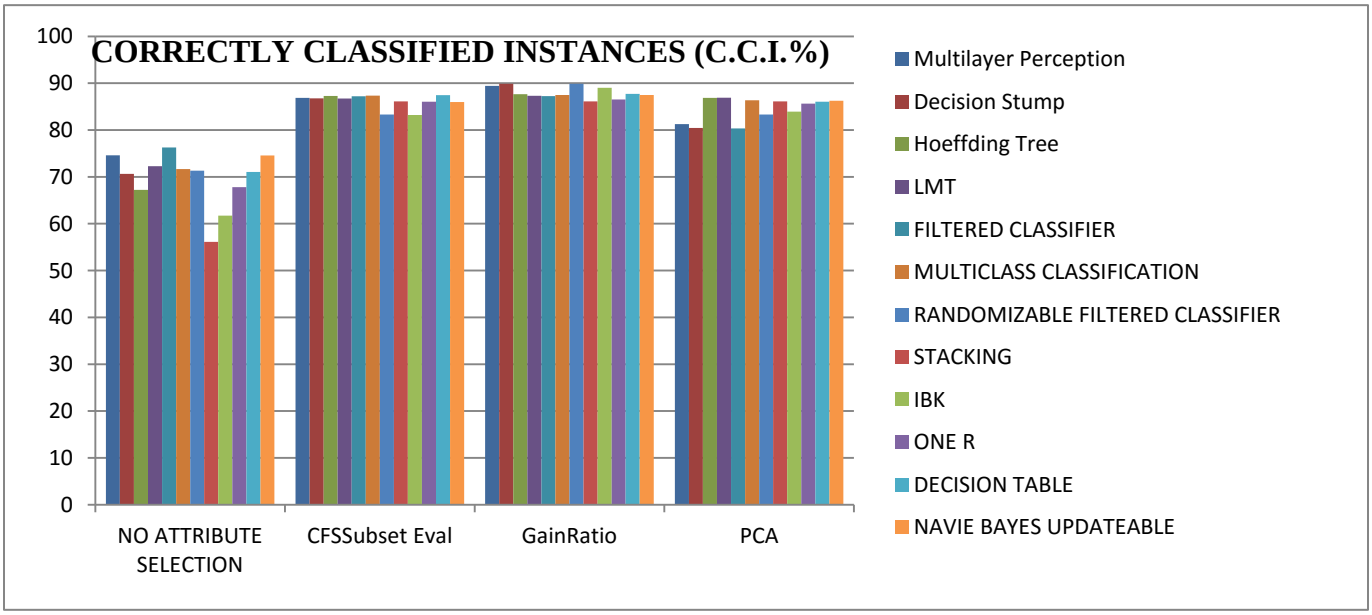


Fig. 3. Correctly Classified Instances Performance (C.C.I %).

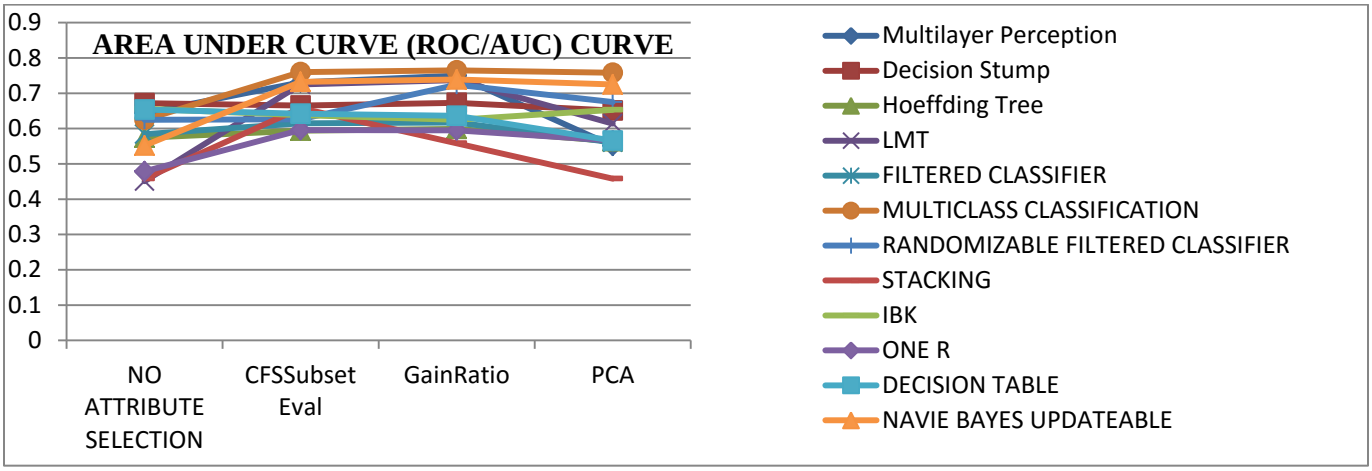


Fig. 4. Area under Curve Performance (ROC).

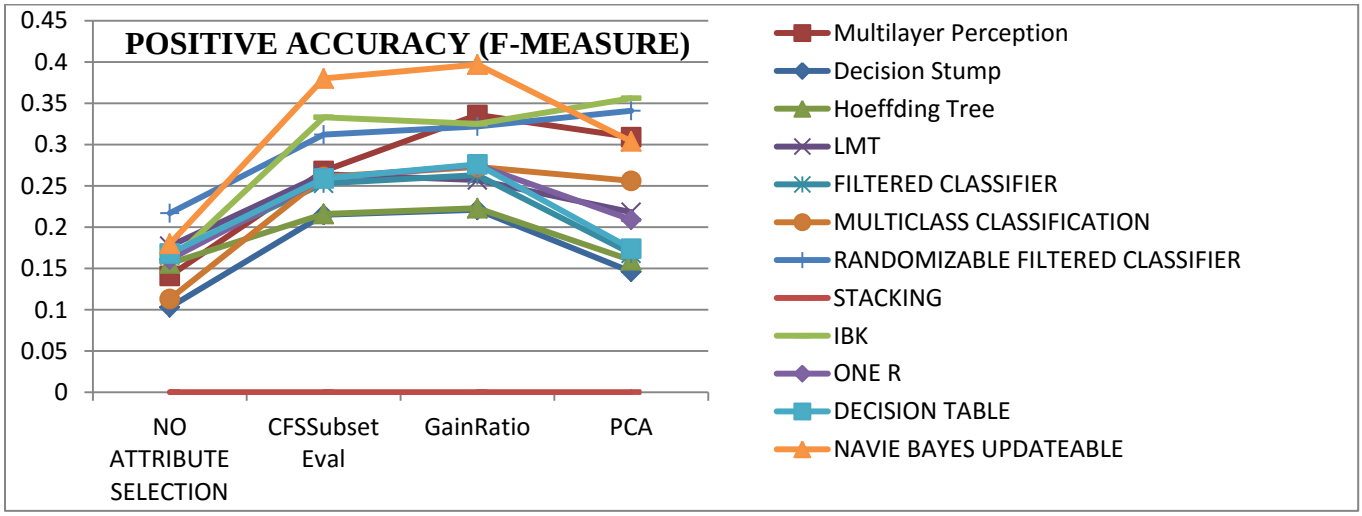


Fig. 5. Positive Accuracy F-Measure Performance.

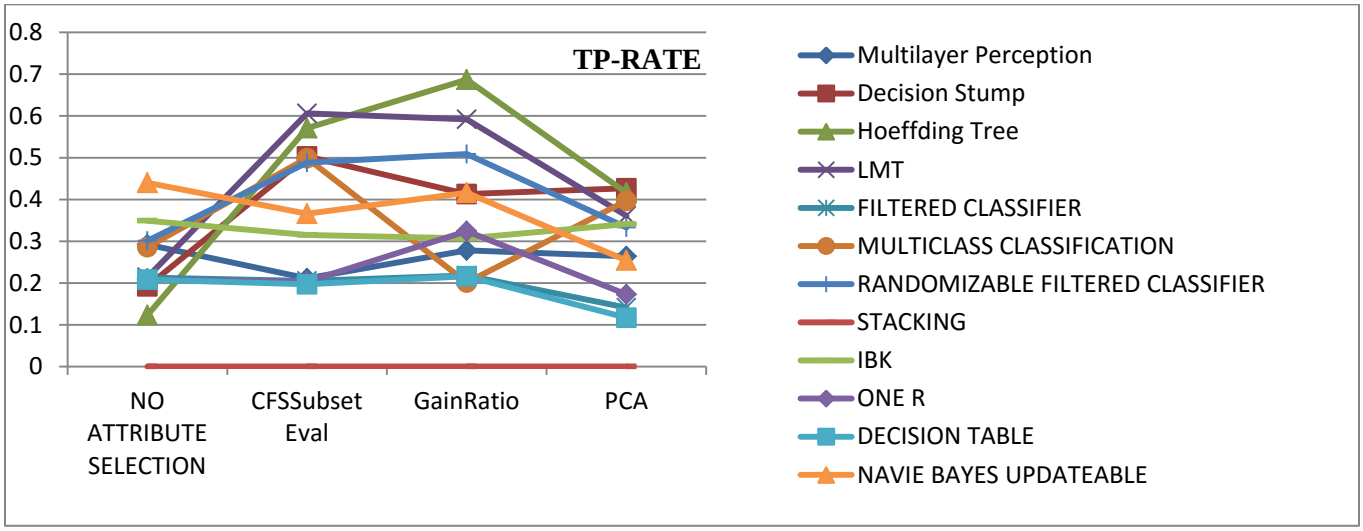


Fig. 6. TP-Rate Performance.

## VII. CONCLUSION

Software Deformity Prone datasets models are widely used with different evaluation parameters which easily comparing the experimental results and analysis with old research studies work. We investigate 17 NASA PROMISE repository datasets models with proposed solution used Attribute Selected Classifier. We have used 12 different classifiers for analysis the performance of TP-Rate, Positive Accuracy, ROC Curve and Correctly Classified Instances. Where we have analyzed that CFSSubsetEval and GainRatio performance is better in almost classifiers. Hoefdding tree, Navie Bayes, MultiClass, IBK and Randomizable filtered class increased performance in Positive Accuracy in all techniques. Stacking has worst performance in positive accuracy and tp-rate in all over technique.

Our future work will be analysis of class imbalance problem with the help of linear regression and rule class attribute method, where we will overcome the problem of over-generalization datasets models.

## REFERENCES

[1] Aloraini A. Different machine learning algorithms for breast cancer diagnosis. International Journal of Artificial Intelligence and Applications, 3(6):21–30, 2012.

[2] Kaur G, Chhabra A. Improved J48 classification algorithm for the prediction of diabetes. International Journal of Computer Applications, 98(22):13–7, 2014.

[3] Wang KJ, Adrian AM. Breast cancer classification using hybrid synthetic minority over-sampling technique and artificial immune recognition system algorithm. International Journal of Computer Science and Electronics Engineering, pp. 408–12, 2013.

[4] Ho, Tin Kam (1995). Random Decision Forests (PDF). Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, pp. 278–282, 1995.

[5] Jiawei han and micheline kamber, Data mining concepts and techniques, second edition,285-291.

[6] H.S. OH and W.S. SEO, , "Development of a Decision Tree Analysis model that predicts recovery from acute brain injury", Japan Journal of Nursing Science, pp. 1742-7924, 2012.

[7] Karim O. Elish, Mahmoud O. Elish, "Predicting defect-prone software modules using supportvector machines", The Journal of Systems and Software, 81(50, pp.649-660, 2008.

[8] Kalai Magal. R, Shomona Gracia Jacob, "Improved Random Forest Algorithm for Software Defect Prediction through Data Mining Techniques", International Journal of Computer Applications (0975-8887), volume 117-No.23, pp.18-22, 2015.

[9] S.Kramer, N.Lavrac, P.Flach, Propositionalization Approaches to Relational Data Mining. In S. Dzeroski & N. Lavrac (Eds.), Relational data mining, 2001:262-291.

[10] M.Kubat, S.Matwin, Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. International Conference on Machine Learning, 1997:179-186.

[11] S.Lessmann, B.Baesens, C Mues, S.Pietsch, Benchmarking Classification Models for Software Defect Prediction: A Proposed Framework and Novel Findings. IEEE Transactions on Software Engineering, 2008, 34(4):485–496.