ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

HIGH ACCURACY PREDICTION ON SOFTWARE DEFECT DATASETS USING AVERAGE PROBABILITY ENSEMBLE TECHNIQUE

K.Sarath Kumar[*1] and A.K.Puneeth Kumar[2]

[*1]M.Tech Student (CSE), Department of Computer Science Engineering, SEAGI Tirupati.

2HOD Department of Computer Science Engineering, SEAGI Tirupati

ABSTRACT

The present generation software testing plays major role in defect predication. Software defect data includes redundancy, correlation, feature irrelevance and missing value. It is hard to ensure that the software is defective or non-defective. Software applications on day-to-day businesses activities and software attribute prediction such as effort estimation; maintainability, defects and quality classification are growing interest from both academic and industry communities. Software defect predication can be done using several methods, in that random forest and gradient boosting are effective. Even though they are efficient, the defect datasets contain incomplete or irrelevant features. The proposed system Average Probability Ensemble technique used to overcome those problems and gives high classification result to compare another method, because it has integrated with three algorithms to use classification performance. It gives more accurate results in publicly-available software datasets.

Keywords:- *Software defect prediction, Software metrics, and Ensemble learning models. .*

INTRODUCTION

Generally software defect data containing redundancy, correlation, features irrelevance and missing samples. It is hard to ensure balanced distribution between data determine to defective and non-defective software.

Software defect prediction [1] emphasized the success of many algorithms including decision trees, Bayesian methods, and artificial neural networks multilayer. However, these methods are sub-optimal in the case of skewed and redundant defect datasets. The prediction performance of these methods gets worse when the defect datasets contain incomplete or irrelevant features. The data imbalance problems are reduced by using classification methods. The defect classification methods are performing weighted support vector machines and random forest. The support vector machines have given less defect classification result to compare with random forest method.

The random forest gives accurate result in classification problems. To propose a software defect classification method using Average Probability Ensemble (APE) learning module. The proposed APE system incorporates six classifiers Random Forest (RF) Gradient Boosting (GB), Weighted SVMs (W-SVMs), Logistic Regression (LR), Multinomial Naïve Bayes (MNB) and Bernoulli NaiyeBayes (BNB).

In that APE classification are performing random forest and weighted SVMs ensemble learning techniques have been very successful in handling small-size and imbalanced datasets. Ensemble learning approach known as sampling based online bagging. In their empirical study sampling based online bagging achieved balanced performance with positive and negative samples but unstable performance when the class distributions change over time.

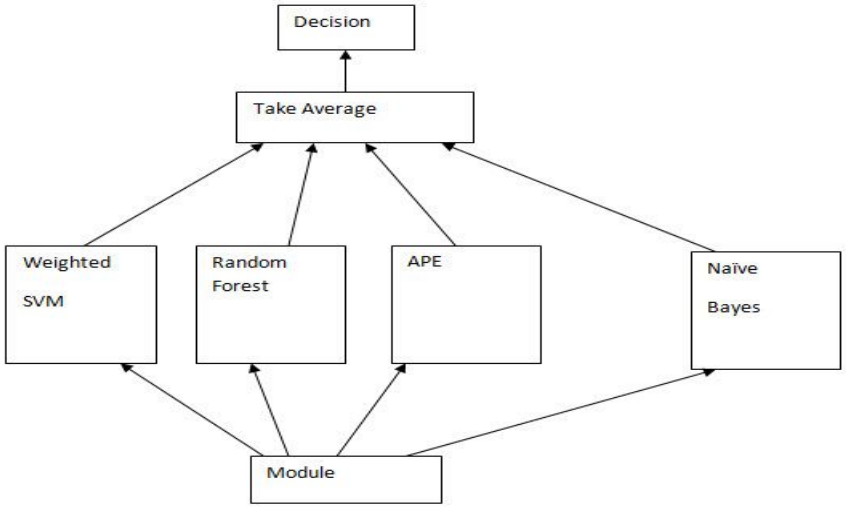
Improve the classification performance of the proposed ensemble classifier; efficient feature selection is combined with the proposed ensemble model yielding an enhanced ensemble classifier. This enhancement resulted in efficient handling of redundant and irrelevant features in software defect datasets. Therefore, the objectives of the project are to demonstrate the positive effect of feature selection on the performance of defect classification and to propose a two-variant ensemble learning algorithm which is robust to both data imbalance and feature redundancy [8].

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

Architecture



Input

Fig: 1 The Framework for Software Defect Prediction using Features Subset Selection Process

RELATED WORK

[1]The presents the results of a systematic review conducted to bring together evidence on software maintainability prediction and metrics [2]. The study was targeted at the software quality attribute of maintainability as different to the process of software maintenance. The results showed that maintainability, as understood in the situation of software systems, was in conformance to the definition provided by IEEE. the commonly used maintainability prediction models were based on algorithmic techniques and there was no distinction of which models should be applied to which maintainability sub-characteristic or maintenance type the most commonly used predictors were those based on size, difficulty and combination, and gathered at source code level. The use of prediction techniques and models, precision measures and cross-validation methods was found scarce for validating maintainability prediction models and the most usually used maintainability metric employed an ordinal scale and was based on expert judgment.

[2]Software defect prediction studies usually built models using within-company data, but very few focused on the prediction models trained with cross-company data [3]. It is difficult to employ these models which are built on the within-company data in practice, because of the lack of these local data repositories. Recently, transfer learning has attracted more and more attention for building classifier in target domain using the data from

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS
International Journal of Engineering Researches and Management Studies

related source domain. It is very useful in cases when distributions of training and test instances differ, but is it appropriate for cross-company software defect prediction? the prior works selecting training data which are similar from the test data, we proposed a novel algorithm called Transfer Naive Bayes (TNB), by using the information of all the proper features in training data. The comparative methods, and shows the experiment results on the data sets from different organizations.

[3]Building defect prediction models in large organizations has many challenges due to limited resources and tight schedules in the software development lifecycle. It is not easy to collect data, utilize any type of algorithm and build a permanent model at once. These conducted a study in a large telecommunications company in Turkey to employ a software measurement program and to predict pre-release defects [4]. Based on our prior publication, and shared our experience in terms of the project steps. Introduced new techniques that improve our earlier results. The specific results indicate that about the organization subject to this study, the use of version history information along with code metrics decreased false alarms by 22%, the use of dependencies between modules further reduced false alarms by 8%, and the decision threshold optimization for the Naïve Bayes classifier using code metrics and version history information further improved false alarms by 30% in comparison to a prediction using only code.

[4]A predicting defect-prone [5] software component is an economically important activity and so has received a good deal of attention. However, making sense of the many, and sometimes seemingly inconsistent, results are difficult. The framework is comprised of 1) scheme evaluation and 2) defect prediction components. The scheme evaluation analyzes the prediction performance of competing learning schemes for given historical data sets. The defect predictor builds models according to the evaluated learning scheme and predicts software defects with new data according to the constructed model. In order to demonstrate the performance of the proposed framework, use both simulation and publicly available software defect data sets. The results show that should choose different learning schemes for different data sets, that small details in conducting how evaluations are conducted can completely reverse findings, and last, that our proposed framework is more effective and less prone to bias than previous approaches.

The APE algorithm has used feature subset selection method. Software defect datasets, features represent software metrics extracted from the source code. Forward selection [7] is commonly used technique to select good features. The process of forward selection using a feature set consisting of cyclomatic complexity, weighted methods per class and lines of code software metrics.

ANALYSIS

Software defect predication using several methods such as Bayesian methods, support vector machines. Even though the large defect datasets contain incomplete or irrelevant features. To overcome those problems we are using the random forest technique and solve effectively. Ensemble learning method has similar to random forest, but average probability ensemble (APE) it uses forward method and greedy forward method. In the proposed system some other feature selection techniques can be used in publicly-available software defect datasets.

An ensemble of classifiers is proposed to address the problem of robust software defect classification. This model is based on the average probability ensemble approach. Ensemble learning is the process of grouping learning models generated from a set of base classifiers. Such models are expected to exhibit robustness against data imbalance and feature redundancy that usually hinder software defect datasets.

This robustness is guaranteed by averaging the This threshold may be looked at as a quantization step that will definitely introduce errors in the decisions made. Therefore, our selection of average probability ensemble serves two purposes: (1) conformity with the AUC measure and (2) obsoletes the need for threshold selection.

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

In the proposed APE model, each classifier predicts the probability that a software component belongs to the defective class. Then, the output probabilities are averaged as the final probability estimation. An outline of the proposed framework is shown in where seven base classifiers are combined to form the APE model. These classifiers include random forests, gradient boosting, stochastic gradient descent, W-SVMs, logistic regression, multinomial naive Bayes, and Bernoulli naive Bayes.

The selection of the base classifier types is solely motivated by their wide acceptance in the software engineering community specifically and the machine learning experts at large.

Process: 1

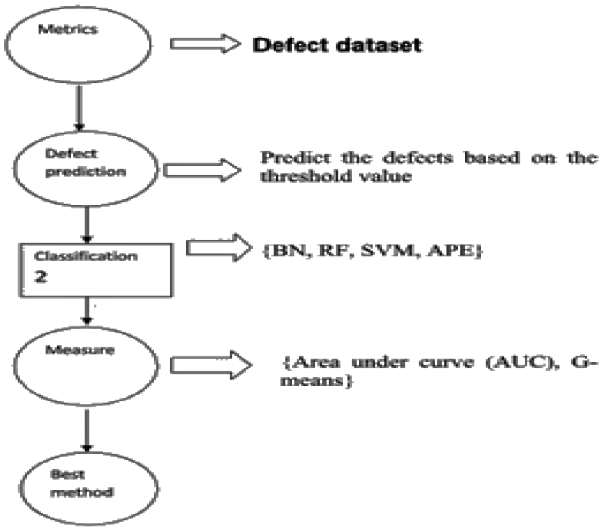


Fig: 2 Process of Project Analysis

Process: 2

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

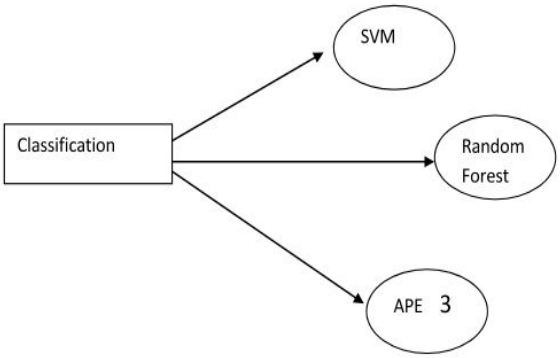


Fig: 3 Process of Classification methods using defect datasets

The classification has several methods such as support vector machines, random forest and average probability ensemble. The APE method find the features selection after go for classification.

IMPLEMENTATION

A. Random Forests

Random forests consist of several classification or regression trees. Using random feature selection, these trees are induced from bootstrap samples of the training data. In classification problems, each data sample is fed down each of the trees in the random forest. Then, the latter outputs as its decision class the class that received most of the votes made by the individual trees. Breiman showed that error rates in random forests depend on the strength of each individual tree and the correlation between any two trees in the forest.

B. Gradient Boosting

Friedman proposed gradient boosting to solve regression problems using a prediction model consisting of an ensemble of weak predictors. These predictors are typically decision trees. Given a set of decision trees, {DT1, DT2, DTi….., DTn}, the gradient boosting algorithm produces a weighted summation of the output decisions of each individual tree as follows:

$$f(x)=w_0+w_1 h_1 (x)+ w_2 h_2 +\ldots\ldots+ w_n h_n(x) \quad (1)$$

C. Logistic Regression

Logistic regression provides a very powerful discriminative model based on the well-known logistic (sigmoid) function:

$$g(z)=1/1+e^{-z} \quad (2)$$

The logistic function has very attractive properties including continuous differentiability and linear relation between the function and its derivatives (of any order).The logistic regression has been successfully applied in classification problems. Given two classes, labeled Y=0
and Y = 1, and N n-dimensional features {x1, x2, . . . , xi, . . . , xN} where each feature sample is treated as a random vector consisting of discrete randomvariable, the logistic regression yields a generative model that learns p(Y|x) using a direct application of Bayes rule as follows:

$$P(Y/x) = \quad (3)$$

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

D. ***Weighted Support Vector Machines***

The Support Vector Machines (SVMs). SVMs have found successful application in many fields including bioinformatics, text mining, image recognition, system identification and leak detection. SVMs represent a discriminative classifier, defined by a separating hyper plane, finds an optimal hyper plane that separates samples pertaining to two different labels (binary classification). Data samples featured on the hyper planes represent the support vectors.

$w x + b>0$ for non defect cases (4)

$w x + b=0$ for marginal cases (5)

$wx+b<0$ for defect cases (6)

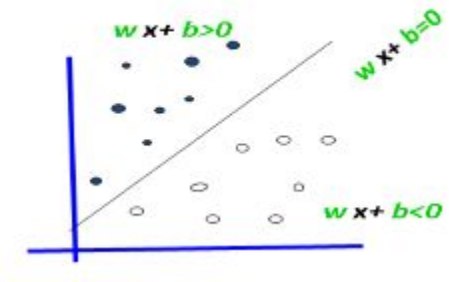


Fig: 4 Linear Classifications in SVMs

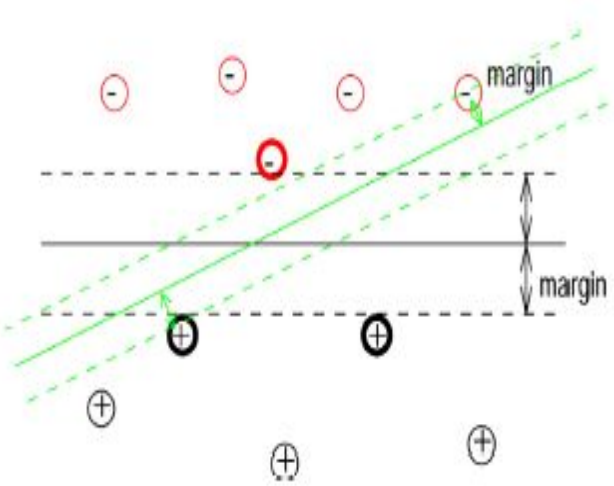


Fig: 5 Hyperplanes in SVMs

Given Nm-dimensional feature vectors x ={x , x , ... , x , ... , x }(i =1,2, ... , N) and their associated class label y 1, +1}, in the performance evaluation, includes Pearson's correlation. [fig.5]Where $z^{(i)} = \phi(x)$ defines the non-linear mapping, ϕ(.), applied on the ith feature vector $x^{(i)}$. The SVM hyper plane is expressed by w and b. The slack variables, $c^{(i)}$, allow the training samples to be mis-classified or located inside the margin. Penalizes the solutions impacted with many training error using the term ϕ (.).

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

EXPERIMENTAL RESULT

The proposed Average probability Ensemble evaluated algorithms were implemented using java language. Classification performance is measured using the AUC measure. ROC curves are popular metrics to evaluate classification algorithms next to imbalanced datasets [9]. The AUC measure determines the level of the area below a ROC curve and is computed as follows:

$$AUC = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} 1_{pi>pj} \quad (7)$$

Where the index i loop over the correctly predicted positive class samples and j loops over the correctly predicted negative class samples. p_i, p_j are the predicted probabilities to the data sample i and j respectively. Finally, $1_{pi>pj}$ returns 1 if and only if $p_i > p_j$, and 0 otherwise.

The proposed APE ensemble learning model is benchmarked against the basic classifiers W-SVMs and random forests. More-over, some feature selection techniques are assessed to verify. More specifically, the selection criterion, considered in the performance evaluation, includes Pearson's correlation, Fisher's criterion, and the GFS approach. Finally, the classification performance of the proposed APE model is evaluated using publicly available software defect datasets [10]:

1. Ant-1.7.
2. Camel-1.6.
3. KC3 datasets.
4. MC1.
5. PC2.
6. PC4

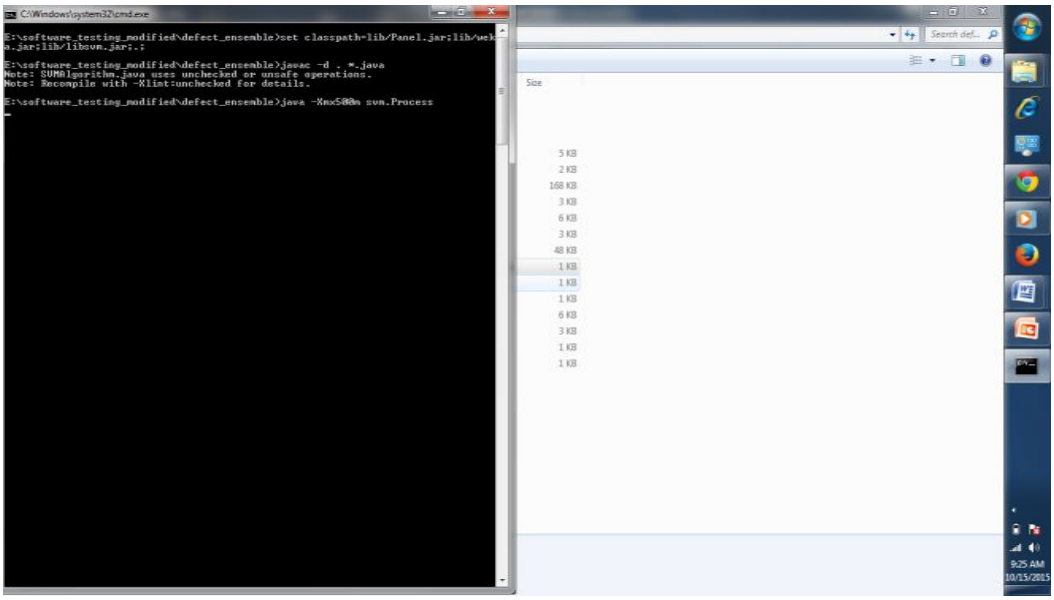


Fig 6 Run the project

[Fig.6]We create the batch dot file to run the code. Batchfile nothing but integrated file when the run the file open the command prompt to connect the home page

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

Fig 7 Home page of software defect prediction using Average Probability Ensemble

[Fig.7]This is the home page, here showing the four labels the first label random forest it algorithm load the dataset it has classify the data and find defects. The second label also same load the dataset, classify the data and find the defects. The third label view the result in accuracy of both algorithms, and final label is exit. To click the exit quit the project.

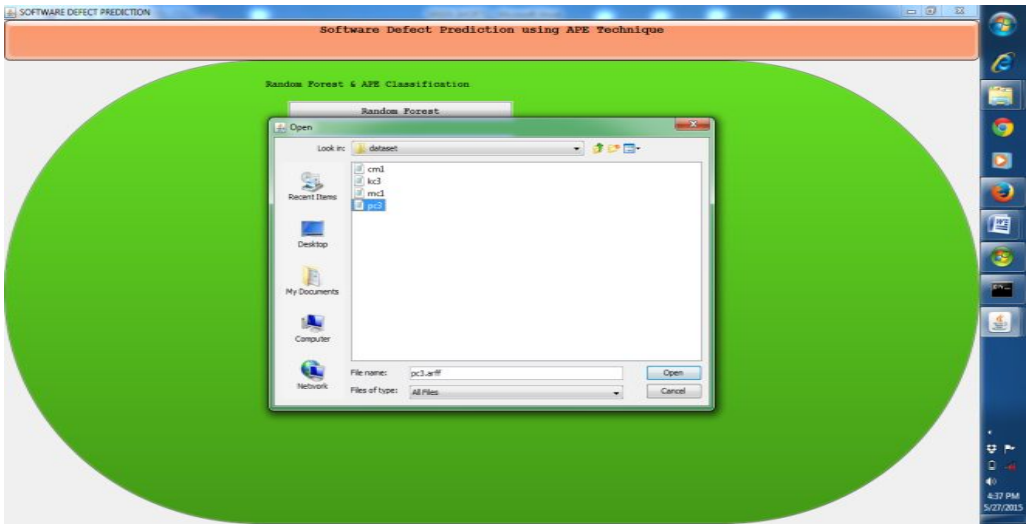


Fig 8 Load the dataset into Random Forest

[Fig.8]Here we load the dataset in to random forest, click on the random forest button open the dataset folder click the which dataset we are find the defects then click the open.

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

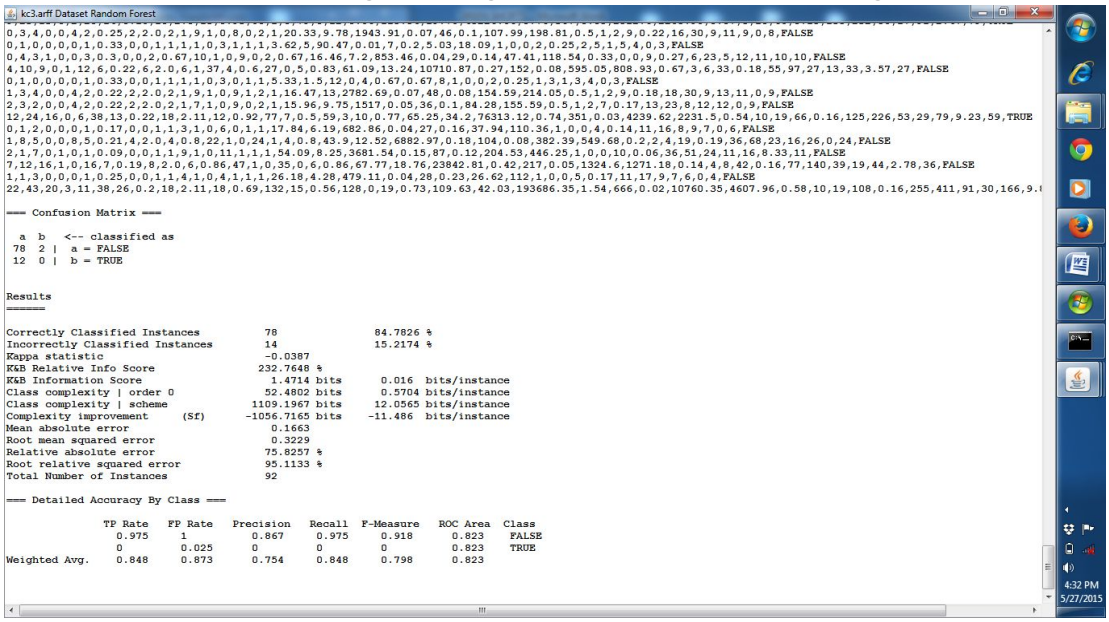


Fig 9 Result of the dataset

[Fig.9]Here it has show the data what have in the dataset classify the data following the confusion matrix it contains information about actual and predicted classification done by a classification system.

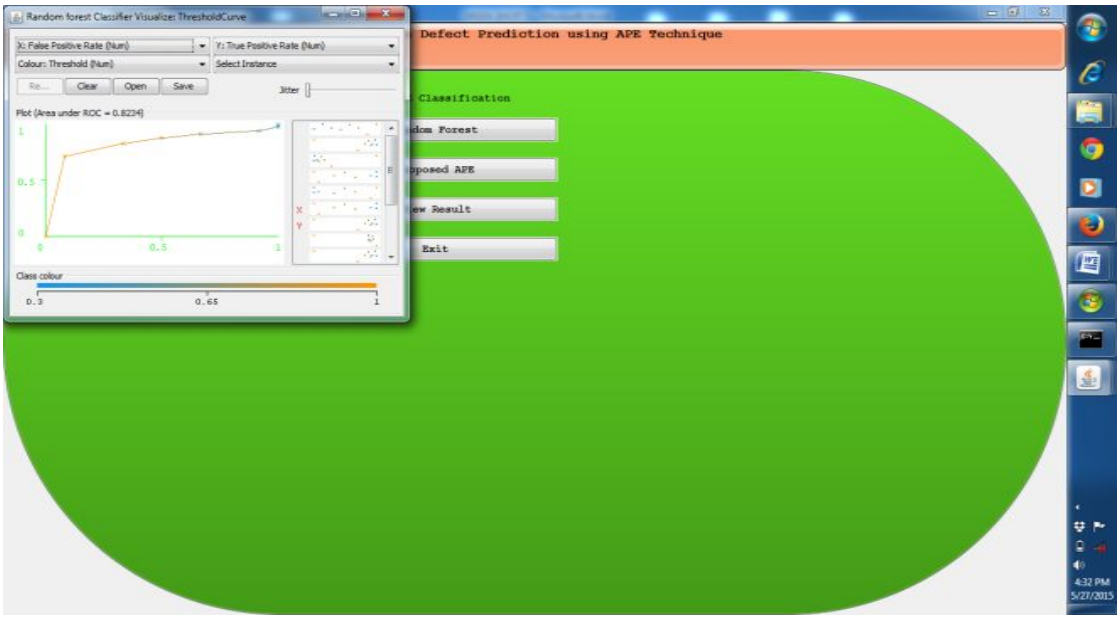


Fig 10 Graph result of kc3 dataset

The random forest generates the graph results and the same process for the average probability ensemble technique. Here given input as a dataset and classify the data while generate the graph. After we have click the result button show the accuracy of result of both methods [i.e. Fig.11]

ISSN: 2394-7659
IMPACT FACTOR- 2.187

IJERMS

International Journal of Engineering Researches and Management Studies

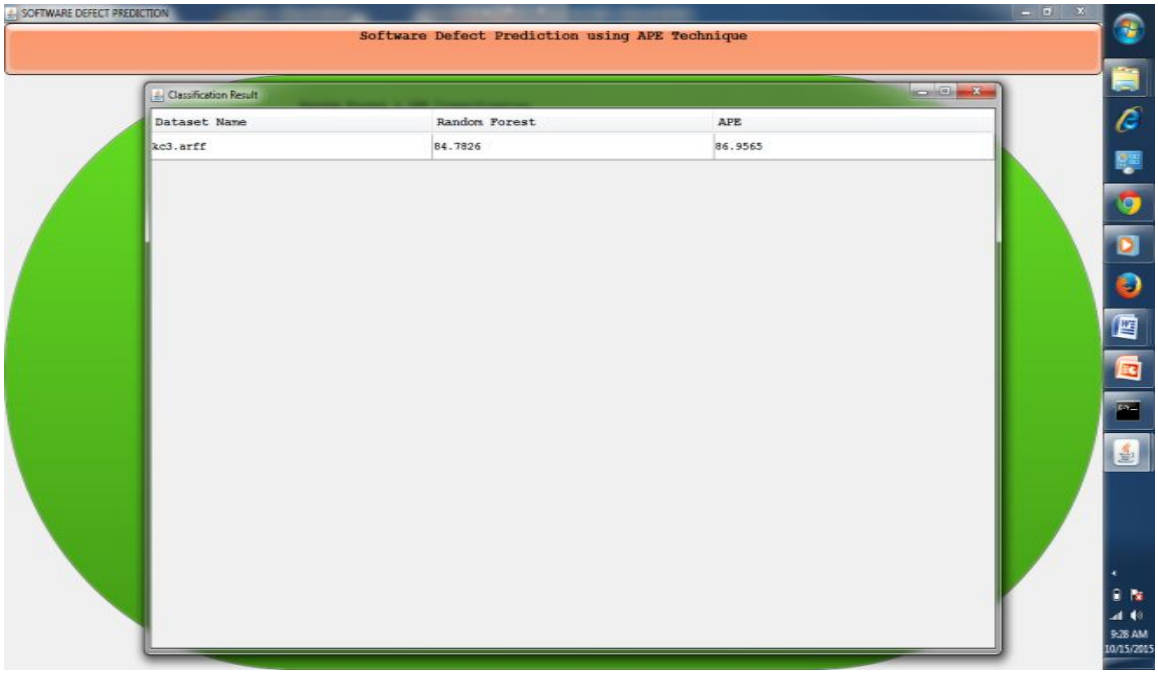


Fig 11 Result of Both Methods

This is the case for the W-SVMs classifier with the KC3, PC2 and ant-1.7 datasets. Also, the classifier based on random forests yielded a similar behavior with the PC2. The non-enhanced version of the proposed model, simple APE, yielded a poor performance also against the PC2 dataset. However, the enhanced version did not only outperform the other classifiers but also attained the highest classification performance reported in the literature using the software defect datasets considered and the G-mean measure. The drastic performance improvement attributed to the feature selection process where only meaningful software metrics are retained reveals an interesting finding that is worth the analysis carried out in Section.

CONCLUSION AND FUTURE WORK

In this project investigated several feature selection techniques for software defect prediction and observed that selecting few quality features makes for much higher.

AUC then presented the efficacy of ensemble learning against imbalanced datasets. We demonstrated the effectiveness of using average probability ensemble (APE) which attained improved results over conventional methods such as weighted SVMs and random forests . Finally, the enhanced version of the proposed model, APE combination of algorithms attained even higher AUC measures for each datasets such as PC2, PC4 and MC1 datasets. For future work, we intend to apply Average Probability Ensemble techniques to further verify that many features, present in publicly-available software defect datasets.

ISSN: 2394-7659
IMPACT FACTOR- 2.187

International Journal of Engineering Researches and Management Studies

REFERENCES

1. *Issam H. Laradji, Mohammad Alshayeb,LahouariGhouti.Software defect prediction using ensemble learning on selectedfeaturesInformation and Computer Science Department, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia.*
2. *M. Riaz, E. Mendes, E. Tempero, A systematic review of software maintainability prediction and metrics, in: Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement, 2009, pp. 367–377.*
3. *Y. Ma, G. Luo, X. Zeng, A. Chen, Transfer learning for cross-company software defect prediction, Inform. Softw. Technol. 54 (2012) 248–256.*
4. *Tosun, A. Bener, B. Turhan, T. Menzies, Practical considerations in deploying statistical methods for defect prediction: a case study within the Turkish telecommunications industry, Inform. Softw. Technol. 52 (2010) 1242–1257.*
5. *Q. Song, Z. Jia, M. Shepperd, S. Ying, J. Liu, A general software defect-proneness prediction framework,IEEE Trans. Softw. Eng. 37 (2011) 356–370.*
6. *N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, 2011. arXiv:1106.1813.*
7. *T.M. Khoshgoftaar, E. Geleyn, L. Nguyen, L. Bullard, Cost-sensitive boosting in software quality modeling, in: Proceedings, 7th IEEE International Symposium on High Assurance Systems Engineering, 2002, pp. 51–60.J.R.*
8. *Miyamoto, J. Miyakoshi, K. Matsuzaki, T. Irie, False-positive reduction of liver tumor detection using ensemble learning method, in: SPIE Medical Imaging, 2013, pp. 86693B–86693B.*
9. *G. Wu, E. Chang, Adaptive feature-space conformal transformation for imbalanced-data learning, in: International Conference on Machine Learning (ICML 2003), 2003.*
10. *D. Gray, D. Bowes, N. Davey, Y. Sun, B. Christianson, The misuse of the NASA metrics data program data sets for automated software defect prediction, in: Evaluation & Assessment in Software Engineering EASE 25, 2011, pp. 12–25.*