A Deep Analysis of the Precision Formula for Imbalanced Class Distribution

Gabriel Kofi Armah, Guangchun Luo, and Ke Qin

***Abstract*—In this paper we discussed our proposed formula of precision for an imbalanced class distribution which gives a true reflection of a defect predictor in relation to its high classifier performance. Our formula gave values which were closer to the Accuracy computation for both balanced and imbalanced class distribution, thus our formula gave consistent high values for a good predictor irrespective of the size of our target data.**

Approach: We used NASA dataset to come out with well-documented examples as to how to get a higher accuracy, with its corresponding higher precision and subsequently a higher Recall and F- Measure values which are reflection of the higher classifier performance. We used data with the minority class between 5 to 10 percent (5%-10%) data points inclusive. We applied a fixed true positive rate (TPR) of one (1), whiles the false positive rate (FPR) on the other hand ranged from 0.01 to 0.05 inclusive at an interval of 0.01 for our analysis .We used the proposed adjusted formula for precision computation to improve earlier works which were criticized of not being satisfactory. The proposed formula precision (AR) was used to compute the precisions which gave results that were the true reflection of a higher performance predictor. The results in the tables clearly show our assertion for our formula giving good estimated values for precision.

***Index Terms*—True positive rate, false positive rate, precision, recall, F-measure and defect predictor classifier.**

I. Introduction

The performance of data mining classification in the area of machine learning is an onerous work to predict correctly; to be able to make an advancement several factors need to be taken into consideration. Classifying a data wrongly has its own negative side effects (i.e. cost).

In the January, 2007 issue of the IEEE Transactions on Software Engineering Journal, a paper with the title "*Data Mining Static Code Attributes to Learn Defect Predictors"* [1], was published. The authors of [1] carried out many defect prediction experiments. The paper referred to two metrics, namely the probability of detection (pd) or recall and the probability of false alarm (pf). This triggered a comment paper by Zhang and Zhang [2].The two metrics pd and pf attracted the comment from [2]. Zhang and Zhang argument was that the prediction *models by* [3] *were not satisfactory for practical use and needed some form of improvement*.

The limitation they pointed out was with the *precision*, their argument was that the proportion of modules predicted as defective which originally were labelled defective, was low for **7** of the 8 data set used for the experiments (i.e. ranging from 2.02% to 31.55%).The authors thus concluded that by giving the suggestion that, reporting on the performance of software defect prediction models; the true positive rate should be used with precision instead of with the false positive rate.

The comments paper by Zhane and Zhane attracted a reaction by some of the original authors and other two new authors [4]. The main arguments raised were based on the fact that detectors learned in the domain of software engineering rarely yield high precision, and thus low precision predictors can be useful in practice. Although sinking precision to increase the true positive rate may be necessary depending on what objectives are being considered. It is highly inappropriate to absolutely disregard precision in the area of classification.

In April 2011, D. Gray *et al.* presented a paper [5] in which they demonstrated that in a highly imbalanced data class distribution it is rare to rely on true positive rates and false positive rates alone, receiver operating characteristics (ROC) inclusive; in the said paper they said if reliance is based only on these two metrics it may results in excessively optimistic view of classifier performance. They [5] said that after the publication of [2] many defect prediction researchers have continued to report their classification results inappropriately. They therefore made an intuitive contribution which idea they simplified it with examples.

In this paper we have come up with *a transformed (taken the rates into consideration by making the false positive rate (TPR) a constant sort of value)* formula from the original formula for the calculation of precision using the same dataset by [6] .We did so with examples shown in Section III. Our contribution is mainly intrinsic and has been demonstrated with examples.

This paper is divided into the following subsections from this point forward: Section II gives brief description of the overview of Machine Learning; Section III is the description of the imbalance classifier performance and our proposed solution to the precision calculation and Section IV Summary and Conclusion.

II. Brief Overview of Machine Learning

A. Overview

Machine learning can be applied in the area of classification for the prediction of defective modules. When

Manuscript received February 19, 2014; revised May 15, 2014.

G. K. Armah is with the School of Computer Science and Engineering , University of Electronic Science and Technology of China, Chengdu , China (e-mail: gabrielarmah1@yahoo.com).

Guangchun Luo and Ke Qin are with the School of Computer Science, University of Electronic Science and Technology of China (UESTC), Chengdu, P.O. Box 611731, China (e-mail: gcluor@uestc.edu.cn, qinke@uestc.edu.cn).

DOI: 10.7763/IJMLC.2014.V4.447

we have training sets of modules with labels as defective (*positive*) and non-defective (*negative*) a classification model could be learned from the training data. The model can then be used to classify unknown modules. Prediction models normally have four results: true positives (*TP*), false positives (*FP*), true negatives (*TN*), and false negatives (*FN*) which could be presented by a confusion matrix. In Fig. 1, is a confusion matrix which contains information about actual and predicted classifications carried out by a classification system. The performance of such systems is commonly evaluated using the data in the matrix.

Now, each of the data point can be predicted by a binary classification as belonging to only one of the four possible outcomes in Fig. 1.

- *True positive* (*TP*) is the proportion of positive cases that were correctly identified.
- *False positive* (*FP*) is the proportion of negative cases that were incorrectly classified as positive.
- *True negative* (*TN*) is defined as the proportion of negatives cases that were correctly classified.
- *False negative* (*FN*) is the proportion of positive cases that were incorrectly classified as negative.

	labelled positive	labelled negative
predicted positive	*TP*	*FP*
predicted negative	*FN*	*TN*

Fig. 1. A confusion matrix.

TABLE I: A SUBSET OF STATISTICS DERIVED FROM A CONFUSION MATRIX. MEASURES MARKED WITH '+' HAVE AN OPTIMAL VALUE OF 1; MEASURES MARKED WITH '-' HAVE AN OPTIMAL VALUE OF 0

Alias/Aliases	**Defined As**
Testing Set No. instances	$TP + TN + FP + FN$
No. Instances in Class 1 (Positive Class)	$TP + FN$
No. Instances in Class 2 (Negative Class)	$TN + FP$
Accuracy + Correct Classification Rate 1 – Error Rate	$\frac{TP + TN}{TP + TN + FP + FN}$
Error Rate – Incorrect Classification Rate 1 – Accuracy	$\frac{FP + FN}{TP + TN + FP + FN}$
True Positive Rate + Recall Sensitivity Probability of Detection (pd) 1 – False Negative Rate	$\frac{TP}{TP + FN}$
True Negative Rate + Specificity 1 – False Positive Rate	$\frac{TN}{TN + FP}$
False Positive Rate – Type 1 Error Rate Probability of False Alarm (pf) 1 – True Negative Rate	$\frac{FP}{FP + TN}$
False Negative Rate – Type 2 Error Rate 1 – True Positive Rate	$\frac{FN}{FN + TP}$
Precision +	$\frac{TP}{TP + FP}$
F-Measure + F-Score	$\frac{2 \ Recall \ Precision}{Recall + Precision}$
Balance + Distance from ROC optimal point	$1 - \frac{\sqrt{(0 - pf)^2 + (1 - pd)^2}}{\sqrt{2}}$

The class of interest for this work is the positive class which happens to be the defective models and it also constitute the minority class. We also considered three measures of interest for our proposed formula; the true positive rate (*TPR*), the false positive rate (*FPR*), and precision. The *TPR* describes the proportion of data points labelled as positive which were correctly predicted as positive; the optimal value is 1.The *FPR*; describes the proportion of data points labelled as negative which were incorrectly predicted as positive; its optimal value is 0. Precision on the other hand describes the proportion of modules predicted as defective which were correct; its optimal value is 1.

B. Useful Statistical Data

We can derive a lot of useful statistical data for the classifier performance metrics using the results in the confusion matrix table. A subset of these is defined in Table I [5]. Note that in the table being referenced the last two measures defined (*F- Measure and Balance)* are in their most commonly used form. Weight can also be assigned in order to favor either of their comprising measures [6]. In addition the balance measure which was defined by Menzies et.al as a measure of the distance from a point on the receiver operating characteristic (ROC) curve to the ideal point. Which is typically defined as where the true positive rate is 1 and the false positive rate is 0 [1].

III. Imbalanced Class Distribution Effects on Metrics Measures

A. Analysis of Precision Measure

Technically speaking any dataset that exhibits an unequal distribution between its classes can be described as an imbalanced dataset. In Machine learning or data mining field imbalanced data corresponds to data sets exhibiting significant, and in some cases extreme imbalanced data, specifically this form of imbalanced is normally between classes as stated above, where in each case one class severely out represents another [7]-[9].

We begin our analysis with a perfectly balanced data set of 2000 points, 1000 in each class. The assumption here is that, we have achieved a classification performance of *true positive rate (TPR) of 1* and a *false positive rate (FPR) of 0.01*. Here we may say that our classifier is performing perfectly well. The reason being that all the data points in the positive class have been correctly classified (*TPR*=1), and only 1 percent of data points in the negative class (1000 ×0.01=10). Thus 10 data points have been incorrectly classified. The precision of such classifier comes up to: "Precision= $\frac{TP}{TP+FP}$ = $\frac{1000}{1000+10}$ = 0.99 (2 significant figures)," here we have correctly predicted 99% to be in the positive class rightly. A confusion matrix for this example is depicted in Fig. 2.

	labelled positive	labelled negative
predicted positive	*TP*=1000	*FP*=10
predicted negative	*FN*=0	*TN*=990

Fig. 2. *TPR*=1, *FPR*=0.01, Precision=0.99.

In this example using the original precision formula works perfectly to give the expected high precision value. Now let us consider a highly imbalanced class data set distribution; which is what the reality in software defect prediction is what is normally captured. We still consider our 2000 data points but this time with only 15 data points in the positive class being 0.75 percent. Now let us say we once more have achieved a *TPR* of 1 and a *FPR* of 0.01 as used in our previous example. Thus we still expect that the classifier performance would be at the same level as previous, since we are using the same parameters for *TPR* and *FPR*.

However, computing the precision gives us something less: "Precision= $\frac{15}{15+20}$ =0.43" thus the precision is less than 0.99 this is as a result of the imbalanced class; this is the problem this work seeks to address. So here only 43% of all positive predictions were correctly predicted; even though all the 15 *TP* data were correctly predicted; this is what informed us to come up with our intuitive new precision formulae. In our work, we took into account the *TP*R and *FPR* and also stabilize the value of *FP* into a constant. A confusion matrix for the second example is shown in Fig. 3.

	labelled positive	labelled negative
predicted positive	*TP*=15	*FP*=20
predicted negative	*FN*=0	*TN*=1965

Fig. 3. *TPR*=1, *FPR*=0.01, Precision=0.43.

In the domain of software defect prediction, often the data sets under study represents less than 1 percent of the data point in total [1], [10] and [11]. Referencing [5] presented an example of using the most imbalanced of the NASA Metric Data Program; PC2. The data set contained 5589 data points, 23 of which data points were labelled as defective, this actually constituted 0.4 percent of the total data points. Using a *TPR* of 1 all 23 data points labelled 'defective' were correctly classified. Using a *FPR* of 0.01 which represented 1 percent of' (5589-23=5566)' data points were also incorrectly classified as 'non- defective' which is approximately (5566 *0.01 ≈ 56) false positives, resulting in a precision 0.29, despite the other metrics which indicates a near optimal performance. As a result of the high imbalanced class distribution, small changes in the *FPR* from 0.01 to 0.05 had a large effect on actual number of the false positive data points from 56 to 280 though maintaining the *TPR* of 1; this also resulted in changes in the classifier performance. The precision; at this stage reduced from, 0.29 to 0.08. This is the reality if we use the original precision formula. However using our proposed formula gives a higher precision of 0.95(AR) as shown in Table II.

Tables II to III; present statistics for each of the 13 NASA Metrics data program data sets. The tables show class distribution for each data set with the precision when *TPR*=1 and *FPR* from 0.01 and 0.05 in details. We ranked the data sets in ascending order of percentage of modules in positive (minority) class. In their work [3] they said precision is required to give a more accurate representation of the true performance of a classifier.

B. Proposed Precision Formula

Steps:

$$\text{Let } \Pr ecision = \left[\frac{\left\langle \operatorname{Re} levantData \right\rangle \cap \left\langle RetrievalData \right\rangle}{\left\langle \operatorname{Re} trievalData \right\rangle} \right]$$

In our intuitive formula for Precision (imbalanced data, but with a high performance), we applied the original precision formula:

$$\Pr ecision = \frac{TP}{TP + FP} \times 100\% \tag{1}$$

The next thing we did was to suppress *FP* to approximately one (1) using the *FPR*, by rewriting *FP* as the power of its rate $FP^{(FPR)}$

Thus,

$$FP = FP^{(FPR)} \tag{2}$$

We used *FPR* value in the range of 0.01 to 0.05 both extreme values inclusive. We also maintain the *TP* values by raising it to a power of 1, thus *TPR*=1(fixed throughout, we used 1, because the classifier is assume to be of a high performance).

$$\text{Also, } TP = TP^{(TPR)} \tag{3}$$

Now substituting (2) and (3) into (1) gives us (4), we obtain our proposed precision (*AR*) formula:

Thus is given by:

$$\Pr ecision(AR) = \left(\frac{TP^{(TPR)}}{TP^{(TPR)} + FP^{(FPR)}} \right) \times 100\% \tag{4}$$

This (4) is our proposed formula for computing our Precision (*AR*) for an imbalanced dataset given that we have a high performance classifier (predictor).

In Table II, [5]; in Column *four* we computed the percentage positive (minority) class data point; column five we computed percentage precision (original precision formula) and column six consists of precision(*AR*) our proposed formula. This table presents statistic for each of the 13 NASA metrics data program data sets. These data sets were chosen because they have been heavily used in software defect prediction research.

Applying the same principle as in equation (4) we can formulate the Recall and F-Measure formulae as follows:

$$\operatorname{Re} call(AR) = \left(\frac{TP^{(TPR)}}{FN = 0 + TP^{(TPR)}} \right) \times 100\% \tag{5}$$

Assumption; *TPR*=1 and *FN*=0, since the predictor is able to detect all the true positives. This gives a Recall (*AR*) of *100%.or 1.*Similarly; F-Measure can be computed using the original formula in conjunction with the *TPR* and *FPR*, thus:

$$F - Measure(AR) = 2 \times \left(\frac{\Pr ecision(AR) \times \operatorname{Re} call(AR)}{\Pr ecision(AR) + \operatorname{Re} call(AR)} \right) \tag{6}$$

TABLE II: METRIC DATA PROGRAM DATA SETS RANKED IN ASCENDING % DATA POINTS IN MINORITY CLASS. *TPR*=1 AND *FPR*=0.01.

NASA DATA SET ALIAS	Num. of Data points	Num. of positive Class Data point minority	% Age of Positive class Data pts.	% Age Precision original *TPR*=1 *FPR*=0.01	% Age Precision **(AR)** *TPR*=1 *FPR*=0.01
PC2	5589	23	0.4	29.114	**95.6696**
MC1	9466	68	0.7	41.975	**98.4844**
PC5	17186	516	3	75.549	**99.7964**
PC4	1107	76	6.9	88.372	**98.6714**
MW1	403	31	7.7	88.571	**96.8328**
MC3	458	43	9.4	91.489	**97.6963**
CM1	505	48	9.5	90.566	**97.9268**
PC3	1563	160	10.2	91.954	**99.3624**
PC4	1458	178	12.2	93.194	**99.4269**
KC1	2107	325	15.4	94.752	**99.6843**
JM1	10878	2102	19.3	95.982	**99.9503**
MC2	161	52	32.3	98.113	**98.1132**
KC4	125	61	48.8	98.387	**98.3871**

TABLE III: PRECISION STATISTICS FOR EACH OF THE 13 NASA METRICS DATA SETS WITH *TPR*=1, *FPR*=0.05

NASA DATA SET ALIAS	Num. of Data points	Num. of positive Class Data point minority	% Age of Positive class Data pts.	% Age Precision original *TPR*=1 *FPR*=0.05	% Age Precision **(AR)** *TPR*=1 *FPR*=0.05
PC2	5589	23	0.4	7.6412	**94.5531**
MC1	9466	68	0.7	12.6394	**98.0389**
PC5	17186	516	3	38.2222	**99.7295**
PC4	1107	76	6.9	59.375	**98.0417**
MW1	403	31	7.7	62	**96.3972**
MC3	458	43	9.4	67.1875	**97.3634**
CM1	505	48	9.5	67.6056	**97.621**
PC3	1563	160	10.2	69.5652	**99.233**
PC4	1458	178	12.2	73.5537	**99.3131**
KC1	2107	325	15.4	78.5024	**99.66164**
JM1	10878	2102	19.3	82.7233	**99.9356**
MC2	161	52	32.3	91.2281	**97.9583**
KC4	125	61	48.8	95.3125	**98.2976**

Substituting (4) and (5) into (6) yields the formula below:

Since we are dealing with a predictor that has a recall of 100% it means that the false negative is zero (0); thus *FN* is assigned the value ,0.

$$F-measure(AR)=2\times\left\{\frac{\left(\dfrac{TP^{(TPR)}}{TP^{(TPR)}+FP^{(FPR)}}\times\dfrac{TP^{(TPR)}}{FN=0+TP^{(TPR)}}\right)}{\left(\dfrac{TP^{(TPR)}}{TP^{(TPR)}+FP^{(FPR)}}+\dfrac{TP^{(TPR)}}{FN=0+TP^{(TPR)}}\right)}\right\}\times 100\%$$

$$F-measure(AR)=2\times\left\{\frac{\left(\dfrac{TP^{(TPR)}}{TP^{(TPR)}+FP^{(FPR)}}\right)\times 1}{\left(\dfrac{TP^{(TPR)}}{TP^{(TPR)}+FP^{(FPR)}}\right)+1}\right\}\times 100\% \qquad (7)$$

Precision, Recall or probability of detection, F-Measure and others are measures used as classifier performance evaluators. F-Measure is a method of measuring test accuracy; it combines the precision and recall measurements [12].

C. Discussion

Using our F-measure formula with dataset PC2 it gives a precision (*AR*) value of 95.6696 from Table II and a recall of 100%, so using (7) gives us an F-measure (*AR*) value of 97.79; which justifies the high performance classifier (predictor) we are considering. Table III compares our precision (*AR*) values with the precision using the original formula, for the 13 NASA data set, but this time the *FPR* is pegged at 0.05 with the same true positive rate of 1, since the assertion is that this predictor has a high performance one it makes our high valued precision a better measure.

In information retrieval, a perfect precision score of 1.0 means that every result retrieved by a search was relevant, whereas a perfect recall score of 1.0 means that all the relevant document(data) were retrieved by the search(but says nothing about how many irrelevant data were also retrieved).

From Table III, the data sets PC2, MC1 and PC5 have the minimum percentage positive class data points of 0.4%, 0.7% and 3% respectively; this also gave lower precision values of 7.64%, 12.64% and 38.22% in the same order when the original precision formula was used. Interestingly our proposed precision (*AR*) formula gave the expected higher values in correspondence to the higher classifier performance; the precision (*AR*) values are; PC2 (94.55%), MC1 (98.04%) and PC5 (99.73%).

Our proposed precision values gives a positive correlation with the almost 100% performance of the high classifier(predictor), thus making our proposed formula a better precision formula for computing precision for a higher predictor for an imbalanced data set irrespective of how high the ratio of the non-defective points has over the defective points. However, precision (*AR*) formula from Table II & III for MC2 and KC4 gave close values with the original precision formula. The precision values for the original formula and ours, precision (*AR*) for *FPR*= 0.02 to 0.04 are displayed under appendix.

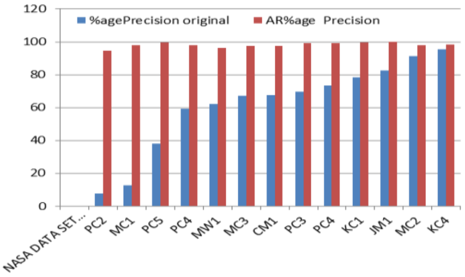

Fig. 4. Comparison of precision values at TPR=1, FPR=0.05.

Fig. 4 shows the precision values graphically for the original precision values and the precision values for our proposed formula and precision (*AR*) outperformed precision(original) in all the 13 NASA datasets used for the analysis.

IV. Summary and Conclusion

Precision snags, especially when class distributions are highly askew. This work was carried out to propose a new precision formula which took into consideration the true positive and false positive rates. This approach was used to correct the problem of precision values which turn to deviate from the expected norm as a result of the use of imbalanced datasets. The proposed formula was derived with a fixed true positive rate of 1 and used a range (0.01 -0.05) for the false positive rate. We used the same assumptions for precision (*AR*) to compute Recall (*AR*) and F-Measure (*AR*).

From the analysis carried out with the 13 NASA datasets our proposed formula for precision, provided higher values for all 13 datasets compared with the original formula for calculating precision. Precision (*AR*) higher values highly supported the assertion that the predictor has a high performance.

Appendix

Computation of Precision (original) and Precision (*AR*), *FPR*=0.02, 0.03 and 0.04 and *TPR*=1 in Tables IV to VI.

The values for precision (*AR*) perform better than the values obtained using the original precision formula and this further affirms the superiority of our proposed precision formula.

TABLE IV: Metric Data Program Data Sets Ranked in Ascending % Data Points in Minority Class. *TPR*=1 and *FPR*=0.02

NASA DATA SET ALIAS	Num. of Data points	Num. of positive Class Data point minority	% Age of Positive class Data pts.	% Age Precision original *TPR*=1 *FPR*=0.02	% Age Precision **(AR)** *TPR*=1 *FPR*=0.02
PC2	5589	23	0.4	17.037	95.4398
MC1	9466	68	0.7	26.5627	98.3933
PC5	17186	516	3	60.7774	99.7828
PC4	1107	76	6.9	78.3505	98.6209
MW1	403	31	7.7	81.5789	96.755
MC3	458	43	9.4	84.3137	97.633
CM1	505	48	9.5	82.7586	97.8651
PC3	1563	160	10.2	85.1064	99.3364
PC4	1458	178	12.2	87.2549	99.4039
KC1	2107	325	15.4	90.0277	99.6705
JM1	10878	2102	19.3	92.2739	99.9473
MC2	161	52	32.3	96.2963	98.0874
KC4	125	61	48.8	98.3871	98.3871

TABLE V: Metric Data Program Data Sets Ranked in Ascending % Data Points in Minority Class. *TPR*=1 and *FPR*=0.03

NASA DATA SET ALIAS	Num. of Data points	Num. of positive Class Data point minority	% Age of Positive class Data pts.	% Age Precision original *TPR*=1 *FPR*=0.03	% Age Precision **(AR)** *TPR*=1 *FPR*=0.03
PC2	5589	23	0.4	12.1053	95.1752
MC1	9466	68	0.7	19.4286	98.288
PC5	17186	516	3	50.7874	99.767
PC4	1107	76	6.9	71.028	98.5624
MW1	403	31	7.7	73.8095	96.6497
MC3	458	43	9.4	76.7857	97.5499
CM1	505	48	9.5	77.4194	97.7947
PC3	1563	160	10.2	79.2079	99.3057

PC4	1458	178	12.2	82.4074	99.3773
KC1	2107	325	15.4	85.9788	99.6546
JM1	10878	2102	19.3	88.8795	99.9438
MC2	161	52	32.3	94.5455	98.0512
KC4	125	61	48.8	96.8254	98.3538

TABLE VI: METRIC DATA PROGRAM DATA SETS RANKED IN ASCENDING % DATA POINTS IN MINORITY CLASS. *TPR*=1 AND *FPR*=0.04

NASA DATA SET ALIAS	Num. of Data points	Num. of positive Class Data point minority	% Age of Positive class Data pts.	% Age Precision original *TPR*=1 *FPR*=0.04	% Age Precision **(AR)** *TPR*=1 *FPR*=0.04
PC2	5589	23	0.4	9.3496	94.8788
MC1	9466	68	0.7	15.3153	98.1699
PC5	17186	516	3	43.6173	99.7493
PC4	1107	76	6.9	64.9573	98.4965
MW1	403	31	7.7	67.3913	96.5299
MC3	458	43	9.4	71.6667	97.4615
CM1	505	48	9.5	71.6418	97.7099
PC3	1563	160	10.2	74.0741	99.2712
PC4	1458	178	12.2	77.7293	99.3468
KC1	2107	325	15.4	82.0707	99.6364
JM1	10878	2102	19.3	85.691	99.9399
MC2	161	52	32.3	92.8571	98.0078
KC4	125	61	48.8	95.3125	98.3159

REFERENCES

[1] T. Menzies, J. Greenwald, and A. Frank, "Data Mining Static Code Attributes to Learn Defect Predictors," *IEEE Trans. Software Eng*., vol. 33, no.1, pp. 2-13 , 2007.

[2] H. Zhang and X. Zhang, "Comments on data mining static code attributes to learn defect predictors," *IEEE Trans. Software Eng*., vol. 33, no. 9, pp. 635-636 , 2007.

[3] J. Davis and M. Goadrich "The relationship between precision-recall and roc curves," in *Proc. the 23rd International Conference on Machine Learning, ICML '06, ACM, New York, NY, USA*,pp. 233–240 ,2006.

[4] T. Menzies, A. Dekhtyar, J. Distefano, and J. Greenwald, "Problems with precision: A response to comments on 'data mining static attributes to learn defect predictors," *IEEE Trans. Software Eng*., vol. 33, no. 9, pp. 637-640, 2007.

[5] D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, "Further Thoughts of Precession," in *Proc. 15^{th} Int'l Conf. on Evaluation and Assessment in Software Eng. Durham, United Kingdom*, pp. 129-133, 2011.

[6] Y. Jiang, B. Cukic, and Y. Ma, "Techniques for evaluating fault prediction models," *Empirical Softw. Eng*., vol. 13, no. 5, pp. 561–595, 2008.

[7] H. He and X. Shen, "A Ranked Subspace Learning Method for Gene Expression Data Classification," in *Proc. Int. Conf. Artificial Intelligence*, pp. 358-364, 2007.

[8] M. Kubat, R. C. Holte, and S. Motwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine Learning*, vol. 30, no. 2/3, pp. 195-215, 1998.

[9] R. Pearson, G. Gorey, and J. Shwaber, "Imbalanced clustering for microarray time-series," in *Proc. Int'l Conf. Machine Learning workshop, learning from imbalanced Data setsii*, 2003.

[10] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking classification models for software defect prediction: A proposed framework and novel findings," *IEEE Transactions on Software Engineering*, vol. 34, no. 4, pp. 485–496, 2008.

[11] Y. Jiang and B. Cukic, "Misclassification cost-sensitive fault prediction models," in *Proc. the 5th International Conference on Predictor Models in Software Engineering*, *PROMISE'09*, ACM, New York, NY, USA, 2009, pp. 20:1–20:10.

[12] B. Mishbra and K. Shukla, "Impact of attribute selection on defect proneness prediction in OO software," in *Proc. 2011 Int'l Conf. Computer and Communication Technology (ICCT)*, pp. 367-372l, 2011.

Gabriel Kofi Armah became a member of IACSIT in Feb. 2014. He obtained his BSc. in computer science in 1997 from KNUST in Ghana and master degree in MIS from University of Ghana in 2003. He is currently doing his PhD in software engineering at UESTC in China.

He has done a two year compulsory national service in Ghana. He is a computer science lecturer at the University for Development Studies, Ghana. He has some publication to his credit. His research interest includes software engineering, algorithms, data mining using machine learning and Weka.

Guangchun Luo received his Ph.D. degree in computer science from the UESTC in 2004.

He is currently a full professor of School of Graduate at the UESTC in China. He has over seventy publications to his name. His research interests include software engineering, mobile networks and network security.

Ke Qin received his ME and Ph.D. degrees from the UESTC in 2006 and 2010 respectively. He was also a visiting scholar at Carleton University, Ottawa, Canada in 2008. He is currently an associate professor at the UESTC, China. He has over forty publications to his credit His research interests include chaos, chaotic neural networks and nonlinear systems.