APPLYING FAHP TO IMPROVE THE PERFORMANCE EVALUATION RELIABILITY AND VALIDITY OF SOFTWARE DEFECT CLASSIFIERS

Hussam Ghunaim

Under the Supervision of Dr. Julius Dichter

DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

AND ENGINEERING

THE SCHOOL OF ENGINEERING

UNIVERSITY OF BRIDGEPORT

CONNECTICUT

October 2019

APPLYING FAHP TO IMPROVE THE PERFORMANCE EVALUATION RELIABILITY AND VALIDITY OF SOFTWARE DEFECT CLASSIFIERS

APPLYING FAHP TO IMPROVE THE PERFORMANCE EVALUATION RELIABILITY AND VALIDITY OF SOFTWARE DEFECT CLASSIFIERS

Hussam Ghunaim

Under the Supervision of Dr. Julius Dichter

Approvals

Committee Members

Name	Signature	Date
Dr. Julius Dichter		1.22.2020
Dr. Ausif Mahmood	Ausif Mahmood	1-22-2020
Dr. Christian Bach		1-28-2020
Dr. Miad Faezipour	Miad Faezipour	1-28-2020
Dr. Robert Todd	Robt S Todd	1.29.2020
Dr. Amalia Rusu	Amalia Rusu	1/10/20

Ph.D. Program Coordinator

Dr. Khaled M. Elleithy		1/30/2020

Chairman, Computer Science and Engineering Department

Dr. Ausif Mahmood	Ausif Mahmood	1-22-2020

Dean, School of Engineering

Dr. Tarek M. Sobh	T. Sobh	1/20/2020

APPLYING FAHP TO IMPROVE THE PERFORMANCE EVALUATION RELIABILITY AND VALIDITY OF SOFTWARE DEFECT CLASSIFIERS

ABSTRACT

Today's Software complexity makes developing defect-free software almost impossible. On an average, billions of dollars are lost every year because of software defects in the United States alone, while the global loss is much higher. Consequently, developing classifiers to classify software modules into defective and non-defective before software releases, has attracted a great interest in academia and the software industry alike. Although many classifiers have been proposed, none has been proven superior to others. The major reason is that while a research shows that classifier-A is better than classifier-B, we can find other research coming to a diametrically opposite conclusion. These conflicts are usually triggered when researchers report results using their preferred performance quality measures such as recall and precision. Although this approach is valid, it does not examine all possible facets of classifiers' performance characteristics. Thus, performance evaluation might improve or deteriorate if researchers choose other performance measures. As a result, software developers usually struggle to select the most suitable classifier to use in their projects. The goal of this dissertation is to apply the Fuzzy Analytical Hierarchy Process (FAHP) as a popular multi-criteria decision-making technique to overcome these inconsistencies in research outcomes. This evaluation framework incorporates a wider spectrum of performance measures to evaluate classifiers' performance, rather than relying on selected, preferred measures. The results show that this approach will increase software developers' confidence in research outcomes, help them in avoiding false conclusions and indicate reasonable boundaries for them. We utilized 22 popular performance measures and 11 software defect classifiers. The analysis was carried out using KNIME data mining platform and 12 software defect data sets provided by NASA Metrics Data Program (MDP) repository.

ACKNOWLEDGEMENTS

I am extremely grateful to the Almighty Allah, who has helped me all the way to complete this work successfully. I owe a deep debt of gratitude to my wife and four children for their unstinted support, encouragement and patience.

I am honored that my work has been supervised by Dr. Dichter and Dr. Elleithy, who took every occasion to provide their continuous support, comments and guidance for the successful completion of this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Software defects are a serious threat to the success of the software development industry [1]. On an average, billions of dollars are lost every year because of software defects in the United States alone [2], while the global loss is much higher. Although defects can be detected through various quality procedures, finding and fixing defects consume a significant portion of the available resources [3]. Most software defects are normally found within a relatively small number of modules [4] [5]. Therefore, developing software defect classifiers has become a promising methodology to identify defective modules before software release. The expected returns are significant in terms of reducing the overall quality assurance activities’ time and costs [1] [6].

The major aim of software defect classifiers is to classify software modules into defective (dM) and non-defective (ndM). This binary classification can be described as a mapping function from a vector $\boldsymbol{x}$ of M features, where $\boldsymbol{x}_i \in R^M$, to one of the classification classes $y_i \in \{dM,\ ndM\}$ [4].

$$f(x)\colon R^M \mapsto \{dM,\ ndM\} \tag{1.1}$$

This model can be trained by a training data set S that has N instances,

$$S = \{(x_i,\ y_i)\}_{i=1}^{N}. \tag{1.2}$$

Numerous techniques have been proposed to develop classifiers, as for instance regression and logistic regression, neural networks, decision trees, and many other machine- learning algorithms [4] [7] with none of them being superior to the others [3] [8]. This is mainly caused by contradicting benchmarking studies. Various software engineering research papers [1] [3] [9] [10] investigated and challenged the reliability of software defect classifiers' benchmarking studies. The common finding of these studies was that while one-study showed classifier A as better than classifier B, other studies came to the exactly opposite conclusion.

1.1 The Research Problem

Software practitioners face the problem of how they can reliably evaluate the performance of defect classifiers, to select the best performing classifier out of several others [11]. Although there are many performance evaluation measures, they usually provide contradictory results. This contradiction is indeed expected, as each of these measures was developed to capture a specific aspect of classifiers' performance. For example, recall, which is known as True Positive Rate (TPR), represents the proportion of the actually defective modules that are classified defective. Similarly, precision, which is known as Positive Predictive Value (PPV), represents the proportion of classified defective modules that are actually defective [3] [12], and so forth. As a result, the performance quality is highly dependent on the specific measure utilized.

This fact leads to the critical question; which performance evaluation measure(s) should practitioners use? In other words, how can practitioners evaluate classifiers in such a way as to always obtain reliable and valid results? This essential requirement is motivated by two possible scenarios: mistakenly classifying defective modules as non-defective raises the risk of software failure, while classifying non-defective modules as defective increases software quality assurance activities' time and costs.

1.2 Scope, Definitions and Limitations

We collect metrics relating to almost every single detail about software systems. The collected metrics are analyzed to identify any anomalies or unacceptable patterns. In general, software metrics are divided into two types: Product metrics and Process metrics [13]. While product metrics are collected about the software artefact, process metrics are collected about the development environment such as, development methodology, quality assurance activities, etc.

Product metrics can be further divided into static and dynamic metrics. Static metrics are collected about features of the software code, while dynamic metrics are collected during the execution of the code. Table (1.2.1) [13] shows some examples of metrics types. Our research is focused on analyzing static code metrics to predict software defective modules.

This choice can be justified as follows. First, for many software projects, static code can be found published on public repositories. This availably makes it possible for other researchers to replicate and verify our work [14] [15]. Additionally, it is quite easy to share data among researchers utilizing public platforms such as GitHup, GoogleCode, etc. The second reason is, process and dynamic code metrics are highly dependent on the specific software project or company that develops it. This usually makes it hard to find those metrics in the public domain or even to get them from their respective sources.

Table (1.2.1) Examples of software metric types.

Process metrics	**Static metrics**	**Dynamic metrics**
Number of Revisions (NR)	Lines of Code (LOC)	Cyclomatic Complexity
Number of Distinct Committers (NDC)	Branch_Count	Function Point
Number of Modified Lines (NML)	Condition_Count	Halstead Complexity
Number of Defects in Previous Version (NDPV)	Cyclomatic_Density	Bug Counting

There is a great deal of disagreement on the exact definition of defects. Clark and Zubrow [16] have defined software defects as "*any flaw or imperfection in a software work product or software process… A defect is frequently referred to as a fault or a bug*". However, other researchers have provided different definitions for defects occurring at different phases of the software production lifecycle [17], [18], [19]. Below are the most commonly used definitions:

- **Errors/faults/bugs** are mistakes that occur during the design stage or written code errors other than syntax errors

- **Defects** are errors occur at the production phase, before release of the software to customers

- **Failures** are errors occurring on the customer's side, causing operational problems

Although IEEE has published the Standard Glossary of Software Engineering Terminology [20], an international consensus over theses definitions has not yet been established [17], [21].

1.3 Dissertation Questions (Aims)

The dissertation question is: Is it possible to incorporate a wide spectrum of performance evaluation measures into a comprehensive evaluation strategy, rather than relying on one or two performance measures selected by a researcher or a practitioner?

The aim of this dissertation is to apply the Fuzzy Analytical Hierarchy Process (FAHP) as a popular multi-criteria decision-making technique as the proposed comprehensive evaluation strategy.

1.4 Contributions to knowledge

Our contribution is the development of a new evaluation strategy that we believe will improve the reliability of the current implemented evaluation techniques.

CHAPTER 2: RELATED WORK

The reliability of software defect classifiers was scrutinized extensively in many published works [8] [11] [22] [23] [24]. Nonetheless, it seems that there are many opportunities for improvement. For example, performance quality measures such as precision, accuracy, etc. can be improved by applying rigorous reliability and verification techniques [8] [11]. Additionally, many of these measures have been borrowed from other disciplines (e.g. Psychology and social sciences). In many cases when these measures are used ‘as is’, they usually have different implications [12].

It has become a common practice for practitioners and researchers to select their most-preferred statistics to support their point of view. This may lead to vague and misleading conclusions. Forman et al. [25] concluded that comparing different research studies has become complicated, and in many cases, the comparisons are not meaningful.

This dissertation emphasizes the fact that performance evaluation must be seen as a comprehensive strategy, rather than relying on performance measure(s) selected based on one’s preferences. Lanza, et. al stated, “*A metric alone cannot help to answer all the questions about a system and therefore metrics must be used in combination to provide relevant information*” [26].

Shepperd et al. [3] conducted an extensive study to find the reasons for variance in classifiers' performance. Their study included 600 experimental results published in many reputed conferences and journals with low acceptance rates. Surprisingly, researcher bias was among the major and wide-spread influential factors. They found that it is extremely difficult to choose the best performing classification technique, because of this phenomenon.

To solve the problem of researchers' bias, Inse et al. [27] asserted that researchers should improve their research outcomes reporting protocols. Kitchenham [28] also suggested the need to enhance the communication and documentation protocols to include sufficient explicit details about how exactly classifiers were used and evaluated in research.

Fenton [21] extensively discussed the concept of research reliability. In general, he emphasized the empirical validity procedures, where researchers are required to validate their findings by replications of experiments. Empirical validation studies have become an essential part in software defect classification research, because usually we lack the required theoretical validation. This fact has led us to our dissertation contribution, which proposes a comprehensive evaluation scheme that will provide proven better evaluation outcomes, compared to preferred selected performance measure(s).

CHAPTER 3: SOFTWARE DEFECT CLASSIFICATION

The practical purpose of implementing software defect classifiers is to identify the defective modules in large software systems. Although many quality assurance techniques are available and are generally effective in identifying those defects, the cost is prohibitive. Weyuker et al. concluded in their study of large commercial software systems that only 20% of the system components can be effectively checked for defects [29]. This fact is evident from today's software industry. It is almost impossible to find a software that is defect-free. As a result, implementing classifiers in software industry has become an active research area.

To build a classifier, we need to create a data model that can associate a set of independent variables to the dependent variable. In our case, the dependent variable is simply a label to identify defective software modules from non-defective ones. The independent variables are the software metrics designed to capture various features of software systems.

Once we build a classifier, it is necessary to train it on a historical data set and then test it to evaluate its performance. This can be achieved by comparing the classifier predictions to the original dependent variable values in the testing data set. An error function must be defined to measure the correctness of the classifier predictions. Figure (3.1) shows the process of training and testing software defect classifiers. Chapters 4 and 5 describe classifiers' evaluation in more detail.

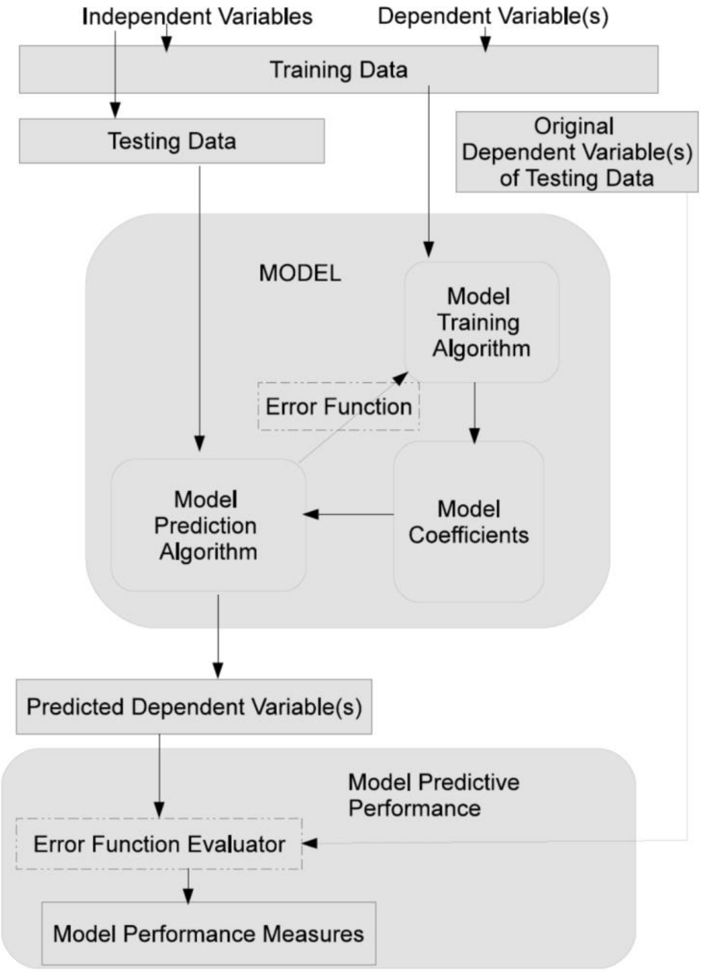

Independent Variables
Dependent Variable(s)
Training Data
Testing Data
Original Dependent Variable(s) of Testing Data
MODEL
Model Training Algorithm
Error Function
Model Prediction Algorithm
Model Coefficients
Predicted Dependent Variable(s)
Model Predictive Performance
Error Function Evaluator
Model Performance Measures

Figure (3.1) The process of training and testing software defect classifiers [30].

Many classifiers exist today in practice. Generally, we can divide classifiers into three major categories: statistical methods, machine learning, and neural networks.

Table (3.1) shows the 11 classifiers used in this research. These classifiers have been chosen based on their popularity in software defect research [4] [31].

Table (3.1) Software defect classifiers.

1	Probabilistic Neural Network (PNN) based on the Dynamic Decay Adjustment (DDA) [32]
2	(SOTA) clustering [33]
3	Fuzzy Rule (FR) [34]
4	Logistic Regression (LR) [35]
5	Naïve Bayes (NB) [36]
6	K Nearest Neighbor (KNN) [37]
7	Multi-Layer Perceptron (MLP-RProp) [38]
8	Support Vector Machine (SVM) [39] [40]
9	Decision Tree C4.5 (DT) [41] [42]
10	SimpleCart (CART) [43]
11	Random Forest (RF) [44]

CHAPTER 4: EVALUATION OF CLASSIFIERS

To evaluate the classifiers' performance, we followed the common practice of using a confusion matrix, table (4.1), where the first column shows the actual (real) positive AP cases (defective modules) and the second column shows the actual (real) negative AN cases (non-defective modules). Similarly, the first row shows the predicted positives (PP) and the second row the predicted negatives (PN). The bottom right cell shows ***T*,** the total number of cases. Figure (4.1) depicts the meanings of the confusion matrix variables. While the optimum desired results would be $fp = fn = 0$, the actual performance of classifiers is still far from achieving this goal. By utilizing these four variables, the classifiers' performance measures can be calculated.

Table (4.1) Confusion Matrix.

tp	fp	**PP**
fn	tn	**PN**
AP	**AN**	***T***

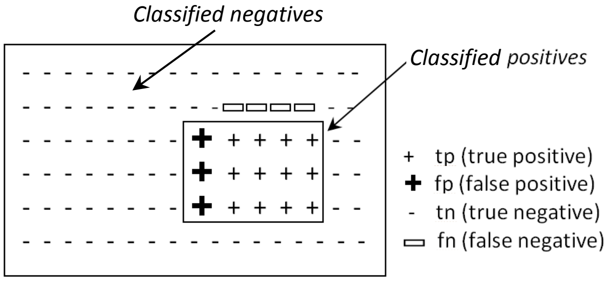


Figure (4.1) Depiction of confusion matrix variables.

Numerous performance measures have been proposed and utilized by researchers and practitioners to evaluate classifiers' performance. Table (4.2) shows the 22 performance measures utilized in our research [3] [45] [46] [47] [48], the selection of which was based on their popularity in software defect classification research [3] [12]. Since Cohen's Kappa is the only measure that needs more clarifications on how to compute its probabilities (i.e. $\Pr(a)$ *and* Pr (e)), we added those clarifications right after the table.

Table (4.2) List of the 22 performance evaluation measures utilized in the study.

1	Recall	$= tp/(tp + fn)$
2	Precision	$= tp/(tp + fp)$
3	Inverse Recall	$= tn/(tn + fp)$
4	Inverse Precision	$= tn/(tn + fn)$
5	Area Under ROC Curve AUC	$= (\mathrm{Recall} + \mathrm{Inverse\ Recall})/2$
6	Accuracy ACC	$= (\mathrm{tp} + \mathrm{tn})/(\mathrm{tp} + \mathrm{fp} + \mathrm{tn} + \mathrm{fn})$
7	F1-Score	$= 2tp/(2tp + fn + fp)$
8	Informedness	$= \ Recall \ + \ Inverse\ Recall - 1$
9	Markedness	$= \ Precision \ + \ Inverse\ Precision - 1$
10	Matthews Correlation Coefficient MCC	$= \frac{tp \times tn - fp \times fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}}$
11	G-Mean1	$= \sqrt{Recall \times Precision}$
12	G-Mean2	$= \sqrt{Recall \times Specificity}$
13	Cohen's Kappa	$= (\mathrm{Pr}\ (\mathrm{a}) - \mathrm{Pr}\ (\mathrm{e}))/(1 - \mathrm{Pr}\ (\mathrm{e})\ )$
14	False Discovery Rate (FDR)	$= fp/(fp + tp)$
15	False Omission Rate (FOR)	$= fn/(fn + tn)$
16	False Positive Rate (FPR)	$= fp/(fp + tn)$
17	False Negative Rate (FNR)	$= fn/(fn + tp)$
18	Predicted Positive Condition Rate	$= (tp + fp)/(tp + tn + fp + fn)$
19	Positive Likelihood Ratio (LR+)	$= Recall/FPR$
20	Negative Likelihood Ratio (LR−)	$= FNR/(Inverse\ Recall)$
21	Diagnostic Odds Ratio (DOR)	$= (LR+)/(LR-)$
22	Prevalence	$= (tp + fn)/(tp + tn + fp + fn)$

Cohen's kappa probabilities are calculated as follows:

$\Pr(a)$: is the observed agreement probability among raters

$$\Pr(a) = \frac{tp + tn}{tp + fp + tn + fn}$$

$\Pr(e)$: is the agreement by chance probability among raters

$$\Pr(e) = R_1(P)R_2(P) + R_1(N)R_2(N)$$

Rater1 percentage of positive responses

$$R_1(P) = \frac{tp + fp}{tp + fp + tn + fn}$$

Rater1 percentage of negative responses

$$R_1(N) = 1 - R_1(P)$$

Rater2 percentage of positive responses

$$R_2(P) = \frac{tp + fn}{tp + fp + tn + fn}$$

Rater2 percentage of negative responses

$$R_2(N) = 1 - R_2(P)$$

CHAPTER 5: EVALUATION RELIABILITY AND VALIDITY

Every aspect of our lives requires measurement, especially in engineering fields. However, no measurement can be useful, unless it possesses a minimum of two characteristics: reliability and validity. Reliability means the ability for a measurement to produce consistent results when repeated in many trials. The more consistent the results are, the more reliable the measurement is. On the other hand, a measurement is considered to be valid if it measures what it is intended to measure [8].

Every measurement is affected by both random and non-random errors. Random errors occur in every trial, causing a measurement to produce variant results. Non-random errors occur systematically in every trial and cause the measurement results to cluster around specific erroneous values. The extent to which we can control these two sources of errors is variant and dependent on the specific application area. In software engineering, it seems that we have less control over those errors, compared to other engineering areas.

This argument naturally leads us to the question, which measurement should we choose to evaluate classifiers' performance? Numerous publications proposed a vast spectrum of measurements, proposed by people working in the software engineering field having IT or business backgrounds. Consequently, these measurements seemed to be relevant, as they reflected the viewpoint of their creators within their own specific contexts. Nonetheless, many of these measurements have failed to take into its consideration the rigorous requirements of the measurement theory, which is known as the Metrology. Therefore, their reliability and validity are facing serious challenges. Evans [49] described this paradox as "*While software metrics has not yet achieved a degree of scientific maturity, it is still a valid concept and much work has been undertaken in the field.*"

This failure to fully comply with the measurement theory requirements has led to many of these software quality measures being considered invalid. Abran discussed this contradiction in detail in his book titled "*Software metrics and software Metrology* [50]." He suggested a preliminary solution for this contradiction: "*If software engineering is to mature into a recognized engineering discipline, it needs to be supported by measures, measurement methods and well tested descriptive and quantitative models* [51]." Further, Abran asserted that the only way to develop very well- matured measurement knowledge in the discipline of software engineering is to explore, investigate, and apply Metrology concepts and principles.

On the other hand, some software engineers argue that Metrology principles should not be applied to the software engineering discipline, since software is not a physical object [50] [51]. Consequently, they consider that the current software metrics are acceptable, although they failed to comply completely with the Metrology requirements.

In our opinion, this thinking has led to the phenomenal gap between research outcomes and industry practitioners' practices. Moreover, this gap has become very obvious by recognizing the serious lack of validation of any proposed measurements that usually lead to conflicting claims by academia and industry researchers [22]. Finally, since software engineering carries the *'engineering'* title, it necessarily implies its explicit compliance with engineering practices and principles.

In recent years, many scholars have started to pay increasing attention to the deficiencies in measurement reliability and validity in the software engineering field. For example, Abran [51] proposed a framework for validating software measurements as a potential solution to the current uncertainty. The framework contained three major components:

- Validation of the design of a measurement method
- Validation of the application of a measurement method
- Validation of the use of measurement results in a predictive system

Moreover, he asserted that before any measurement is accepted as reliable and valid, it should pass the requirements of this framework. Even though he referenced many other authors' works in this regard, he believed that none of the many proposed verifications of validity is complete or covers the whole variety of measurement methods used. Therefore, a practical and acceptable validation framework still does not exist!

Other authors have listed their own recipes for what a reliable and valid measurement should look like [22] [21] [52] [53]. Below is a summary of the common ingredients that must be clearly defined for any measurement system to be deemed reliable and valid:

- What are the entities measured?
- What are the attributes of the entities we are interested in?
- What are the units applicable to each measured attribute?
- Which scale is the most appropriate for each measured attribute?

Missing any of these elements will result in awkward measurements system outcomes that are difficult to analyze and comprehend. Likewise, other authors have mentioned the importance of following the broader requirements of the measurement theory (the Metrology) [52] [54] [55]. Below is a summary of the most notable questions any measurement system must answer:

- How do we know if we have really measured an attribute?
- When an error margin is acceptable or not?
- Which statements about a measurement are meaningful?
- Which types of attributes can/cannot be measured?
- What kind of scales can these measurements use?
- How to define these scales?

As the result of this vast inconsistency in measurements, it has become a common practice today among researchers in academia and the software development industry alike, to choose personally preferred measures to use in their research. This phenomenon is known as “researcher bias” [3].

CHAPTER 6: FUZZY ANALYTICAL HIERARCHY PROCESS (FAHP)

To avoid the researchers' bias when evaluating the performance of software defect classifiers, this dissertation proposes the application of multi-criteria decision-making (MCDM). MCDM is a set of very effective methodological tools for dealing with complex problems in various domains such as, medicine, business, engineering, etc. Some example tools are AHP, FAHP, TOPSIS, etc. [56] [57] [58].

The Analytical Hierarchy Process (AHP) technique has been implemented widely in the multi-criteria decision-making (MCDM) field. The essence of this technique is based on an expert judgement method to perform pair-wise comparisons between all implemented criteria. However, AHP suffers from a crucial criticism: it is unable to deal with the impression and subjectivity of the expert judgement when performing the pair-wise comparisons method [59] [60] [61].

In recent years, Fuzzy AHP – or for short, FAHP – has gained noticeable attention as a superior substitute to the AHP technique. The essence of the FAHP method is based on the ability to capture the uncertainty when performing the expert judgement method. Zadeh [62] introduced the fuzzy set theory to compromise the human thought vagueness, which was oriented to the rationality of uncertainty due to imprecision or vagueness, i.e., the consideration of the gradual membership of an element to a particular set of elements [59].

Kabir and Peng [45] [63] applied AHP successfully in the field of classifiers' performance evaluation. In this dissertation, the authors apply FAHP in evaluating binary classifiers' performance as a more robust multi-criteria procedure. To our knowledge, this is the first such application.

In 1983, Laarhoven, et al. proposed the use of a triangular fuzzy membership function as the best fit in performing expert judgement: Figure (6.1.a) [64]. Other functions were proposed as well to fit various uses: Figure (6.1.b and 6.1.c). We chose to use the triangular membership method for its suitability to the software defect classifier domain equation (6.1). The reason for this choice is that we need to provide only two boundaries to our judgement, the upper and lower boundaries, when comparing measures pair-wise. Trapezoidal function, for example, provides two middle values in addition to the upper and lower boundaries, which is not necessary in our research. Similar arguments are applicable to other fuzzy membership functions that might require unnecessary complications. Thus, for the sake of simplicity, we made this choice.

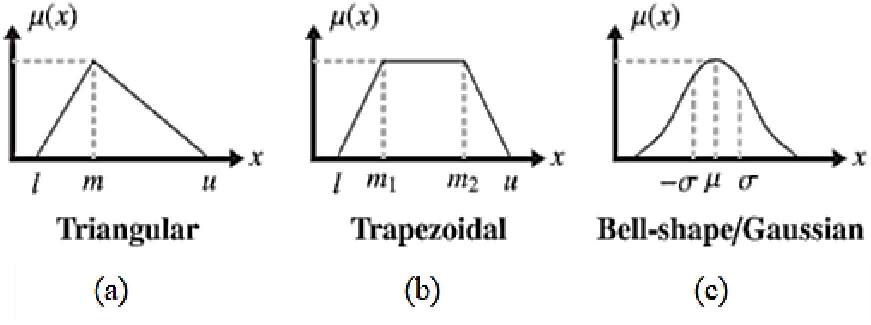


Figure (6.1) Membership functions used in FAHP.

$$\mu(x|\widetilde{M}) = \begin{cases} 0, & x < l, \\ (x-l)/(m-l), & l \le x \le m, \\ (u-x)/(u-m), & m \le x \le u, \\ 0, & x > u. \end{cases} \tag{6.1}$$

Throughout this dissertation, fuzzy quantities are differentiated by a tilde '~' above symbols. A triangular fuzzy number TFN is denoted as (l, m, u), where l denotes the smallest possible value, m the most promising value, and u the largest possible value that describes a fuzzy event. Readers interested in a more detailed introduction to fuzzy numbers and their algebraic operations are recommended to read Harding et al. [65].

CHAPTER 7: EXPERIMENTAL SETUP AND RESULTS

We utilized eleven software defect classifiers table (3.1), chosen based on their popularity in software defect research [4] [31]. The experiments were carried out using KNIME [66] [67], a popular data mining platform and twelve NASA software defect data sets.

KNIME data mining platform was used to run the classifiers on all experimented data sets. The corresponding confusion matrices were constructed and utilized to calculate the classifiers' performance measures, i.e., $E[c \times p]$ matrices, where c is the number of classifiers and p is the number of performance measures. To validate the results, 10-fold cross-validation technique was run on all experiments. Additionally, we normalized all experimented data sets to avoid the dominance of some attributes with large values.

Imbalanced data sets can degrade classifiers' performance and contribute to the unreliability of results [14] [68]. It is quite common for software defect data sets to have non-defective modules as the majority class, with the defective modules as the minority class. Therefore, stratified sampling technique was used to avoid sampling bias. Stratified functionality guaranteed that all created cross-validation folds had class distribution similar to the original data sets distributions, i.e., the ratio of defective to non-defective modules.

For clarity, we start with presenting a summary of the FAHP steps implemented in this study, followed by more detailed calculations in section 7.2 FAHP Application.

Note*: Matrices are denoted by italicized capital letters, and vectors by bold face italicized small letters.*

$Let,$

$c = 11,\ c$ is number of classifiers,

$p = 22,\ p$ is number of performance measures,

d data set

D the set of 12 NASA data sets

```
1) Construct the fuzzy performance measures' pair-wise comparisons $\tilde{A}[p \times p]$ matrix.
2) Compute the $criteria$ fuzzy $weight\ vector\ \tilde{\boldsymbol{w}}$ from $\tilde{A}\ matrix.$
   $\boldsymbol{for\ each\ d} \in \boldsymbol{D\ do}$
         3) Compute the classifiers' evaluation matrix $E[c \times p]$.
         4) Compute the classifiers' scores $S[c \times p]$ matrix.
               a) Compute $p$ number of $B^{(j)}$ matrices (classifiers' pair-wise comparisons) with respect to each criterion $j = 1 \ldots p$
               b) From each $B^{(j)}$, compute $\boldsymbol{s}^{(j)}$ score vectors
               c) Construct the $S[c \times p]$ matrix by combining all $\boldsymbol{s}^{(j)}$ vectors, column wise.
         5) Compute the classifiers' ranking $\boldsymbol{v} = S \cdot \boldsymbol{w}$, where $\boldsymbol{v}_i$ of the vector $\boldsymbol{v}$ represents the global score (i.e. rank) assigned by the FAHP to the $i^{th}$ classifier.
         6) Identify the highest performing classifier compared to the list of experimented classifiers.
   $\boldsymbol{end\ for}$
```

7.1 Data Sets

As the requirement of research replication has become vital for many researchers, we have decided to use the publicly available and widely used NASA software defect data sets [15]. The reasons for this choice are to support the ability to reproduce and verify the published results, and to ease data sharing among researchers [14].

However, NASA data sets suffered from many data quality problems. Shepperd et al. [69] have analyzed in depth these problems that are summarized in table (7.1.1). For clarity, we repeat here the common assumptions about software data sets structure. NASA data sets are organized as matrices of rows and columns. Each row represents one software module (i.e. case), and each column represents one feature (i.e. attribute).

Shepperd et al. [15] performed a comprehensive cleaning strategy to remove all problematic cases and features, table (7.1.2). They published the cleaned-up data sets after removing all cases and features that had one or more of the discussed data quality problems. These data sets are available online at "https://figshare.com/collections/NASA_MDP_Software_Defects_Data_Sets/4054940/1".

Table (7.1.1) Data quality issues with NASA data sets.

Data Quality Problem	Meaning	Consequences
Identical values	Two or more features have the same values for all cases. Similarly, two or more cases have the same values for all features including the prediction label.	Identical features present no additional information. Identical cases confuse learners.
Conflicting values	This problem arises whenever there is a violation of a relational integrity constraint. For example, LOC_TOTAL cannot be less than LOC_EXECUTABLE or LOC_COMMENTS. Fan et al. [70] have discussed integrity constraints in more detail.	Untrustworthy data
Implausible values	The presence of negative or fractional values does not make sense and is not acceptable.	Untrustworthy data
Case inconsistency	Some cases have inconsistent predictions, i.e., two identical cases each result in a different prediction.	Untrustworthy data
Constant values	Features with constant values	They do not present any information.
Missing values	Features with missing values	Confuses the learner

Table (7.1.2) Changes made to NASA data sets after applying the cleaning strategy.
**** df % is the percentage of defective modules.***

NASA Data Sets	Original Data sets			Cleaned-up Data sets		
	#Modules	#Attributes	df % *	#Modules	#Attributes	df % *
CM1	505	41	9.50	327	38	12.84
JM1	10878	22	19.32	7782	22	21.49
KC1	2107	22	15.42	1183	22	26.54
KC3	458	41	9.39	194	22	18.56
MC1	9466	40	0.72	1988	22	2.31
MC2	161	41	32.30	125	22	35.20
MW1	403	41	7.69	253	22	10.67
PC1	1107	41	6.87	705	22	8.65
PC2	5589	41	0.41	745	22	2.15
PC3	1563	41	10.24	1077	22	12.44
PC4	1458	41	12.21	1287	22	13.75
PC5	17186	40	3.00	1711	22	27.53

7.2 FAHP Application

The following are the details of FAHP implementation steps [59] [45]:

Step 1:

Decompose the problem into three hierarchical levels, Figure (7.2.1).

Goal: evaluating the performance of software defect classifiers to select the best-performing classifier

Criteria: twenty-two performance measures

Alternatives: eleven software defect classifiers

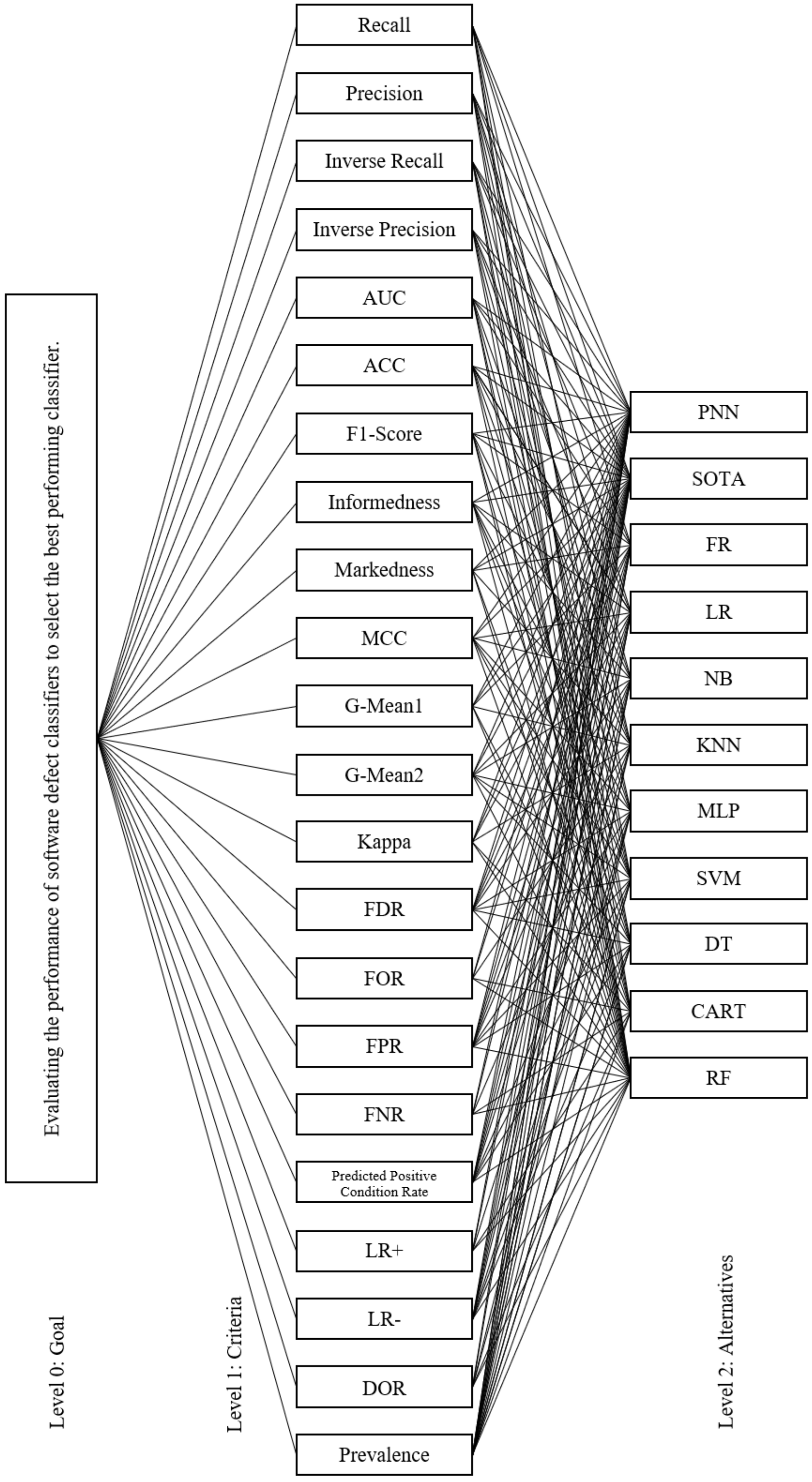


Figure (7.2.1) FAHP hierarchical structure.

Step 2:

Perform fuzzy pair-wise comparisons between all criteria elements using the fundamental scale proposed by Saaty [58], table (7.2.1). At the end of this step, a criteria fuzzy weights vector $\tilde{\boldsymbol{w}}$ is computed. However, this scale is based on crisp evaluation values. As discussed in Chapter 6, crisp evaluation usually leads to unreliable results, due to the expert judgement uncertainty and vagueness. Thus, the scale must be modified to meet FAHP requirements. That is, instead of evaluating the criteria using the crisp scale values, we can use the Triangular Fuzzy Numbers (TFN) to compensate for human uncertainty and increase the reliability of the evaluation. It is to be noted that for any fuzzy number $\tilde{a}$, the reciprocal can be defined as

$$\tilde{a}^{-1} = (l, m, u)^{-1} = \left(\frac{1}{u}, \frac{1}{m}, \frac{1}{l}\right) \tag{7.2.1}$$

Table (7.2.1) AHP and FAHP score interpretations.

AHP Crisp Scale $\boldsymbol{a_{jk}}$	**FAHP TFN** $(\boldsymbol{l}, \boldsymbol{m}, \boldsymbol{u})$ $\tilde{\boldsymbol{a}}_{\boldsymbol{jk}}$	**Interpretation** $\boldsymbol{j}$ and $\boldsymbol{k}$ denote criteria
9	9,9,9	*j* is extremely more important than *k*
7	6,7,8	*j* is strongly more important than *k*
5	4,5,6	*j* is more important than *k*
3	2,3,4	*j* is slightly more important than *k*
1	1,1,1	*j* and *k* are equally important
1/3	1/4,1/3,1/2	*j* is slightly less important than *k*
1/5	1/6,1/5,1/4	*j* is less important than *k*
1/7	1/8,1/7,1/6	*j* is strongly less important than *k*
1/9	1/9,1/9,1/9	*j* is extremely less important than *k*

Table (7.2.1) entries are only suggestive for translating the decision-maker qualitative evaluations of the criteria into quantitative values. It is possible to use other similar scales.

The authors use their extensive experience in the field of binary classifiers evaluation measures to rank their relative importance, following Saaty's fundamental scale of weights. Additionally, the literature provides a large body of research to evaluate the reliability and validity of each of these measures. For brevity, a representative sample is cited in this dissertation [22] [48] [12]. Table (7.2.2) shows the relative fuzzy weights established for these measures.

By assuming that we have p performance evaluation measures (i.e. criteria), we can construct the criteria pair-wise comparison matrix $\tilde{A}$ as follows:

$$\tilde{A}[p \times p] = \begin{bmatrix} \tilde{a}_{11} & \cdots & \tilde{a}_{1k} \\ \vdots & \ddots & \vdots \\ \tilde{a}_{j1} & \cdots & \tilde{a}_{jk} \end{bmatrix} \tag{7.2.2}$$

where $j = 1 \cdots p$ & $k = 1 \cdots p$.

Every entry $\tilde{a}_{jk}$ represents the importance of criterion j relative to criterion k, where $\tilde{a}_{jk} = (1,1,1) \ \forall \, j = k$.

Once matrix $\tilde{A}$ is constructed, we can calculate the criteria fuzzy weights vector $\tilde{\boldsymbol{w}}$ by applying the Geometric Mean method proposed by Buckley [71]. The method can be applied in three steps:

Table (7.2.2) The relative fuzzy weights established for the evaluation measures, Ã matrix.

	F1 score	AUC ROC	G-Means1	G-Means2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	FDR	FOR	FPR	FNR	Predicted Positive Condition Rate	ACC	Informedness	Markedness	LR+	LR−	DOR	Prevalence
F1 score	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8	9,9,9	9,9,9	9,9,9	9,9,9
AUC ROC	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8	9,9,9	9,9,9	9,9,9	9,9,9
G-Means1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8	9,9,9	9,9,9	9,9,9	9,9,9
G-Means2	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8	9,9,9	9,9,9	9,9,9	9,9,9
Matthews correlation coefficient MCC	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8	9,9,9	9,9,9	9,9,9	9,9,9
Cohen's Kappa	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8	9,9,9	9,9,9	9,9,9	9,9,9
Recall	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8
Precision	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8
Inverse Recall	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8
Inverse Precision	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6	6,7,8	6,7,8	6,7,8	6,7,8
False Discovery Rate FDR	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/6,1/5,1/7	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6
False Omission Rate FOR	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/6,1/5,1/7	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6
False Positive Rate FPR	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/6,1/5,1/7	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6
False Negative Rate FNR	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/6,1/5,1/7	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4	4,5,6	4,5,6	4,5,6	4,5,6
Predicted Positive Condition Rate	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4
Accuracy ACC	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4
Informedness	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4
Markedness	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1	2,3,4	2,3,4	2,3,4	2,3,4
Positive likelihood ratio LR+	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1
Negative likelihood ratio LR−	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1
Diagnostic odds ratio DOR	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1
Prevalence	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/9,1/9,1/9	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/8,1/7,1/6	1/6,1/5,1/4	1/6,1/5,1/5	1/6,1/5,1/6	1/6,1/5,1/7	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1/4,1/3,1/2	1,1,1	1,1,1	1,1,1	1,1,1

Firstly, we calculate the fuzzy geometric mean value $\tilde{r}_j$ for each row j in $\tilde{A}_j$

$$\tilde{r}_j = \left(\left(\prod_{k=1}^{p} l_k \right)^{1/p},\ \left(\prod_{k=1}^{p} m_k \right)^{1/p},\ \left(\prod_{k=1}^{p} u_k \right)^{1/p} \right) \qquad (7.2.3)$$

Secondly, sum all fuzzy geometric mean values column wise and find their reciprocal, $\left(\tilde{r}_1 \oplus \tilde{r}_2 \oplus \cdots \oplus \tilde{r}_j \right)^{-1}$. The multiplication and addition of two fuzzy numbers operations are defined as,

$$\tilde{a}_1 \otimes \tilde{a}_2 = (l_1, m_1, u_1) \otimes (l_2, m_2, u_2) = (l_1 \times l_2, m_1 \times m_2, u_1 \times u_2) \qquad (7.2.4)$$

$$\tilde{a}_1 \oplus \tilde{a}_2 = (l_1, m_1, u_1) \oplus (l_2, m_2, u_2) = (l_1 + l_2, m_1 + m_2, u_1 + u_2) \qquad (7.2.5)$$

Lastly, calculate the criteria fuzzy weights vector $\tilde{\boldsymbol{w}}$,

$$\tilde{\boldsymbol{w}}_j = \tilde{r}_j \otimes \left(\tilde{r}_1 \oplus \tilde{r}_2 \oplus \cdots \oplus \tilde{r}_j \right)^{-1} \qquad (7.2.6)$$

To ease the comparisons of classifiers' rankings, we can defuzzify $\tilde{\boldsymbol{w}}$ using the center of area COA concept [72],

$$\boldsymbol{w}_j = \left(\frac{l,m,u}{3} \right), j = 1 \cdots p \qquad (7.2.7)$$

At the end of this step, table (7.2.3) is computed.

Step 3:

Perform pair-wise comparisons between all classifiers with respect to every criterion. At the end of this step, the classifiers scores matrix S is constructed.

Table (7.2.3) Computing the criteria of fuzzy weights vector $\tilde{\boldsymbol{w}}$.

	Fuzzy Geometric Mean Value $\tilde{r}$ (l, u, m)			**Fuzzy Weights $\tilde{w}$ (l, u, m)**			**Defuzzified Weights w**	**Normalized Defuzzified Weights w**
	lower (l)	***middle*** (m)	***upper*** (u)	***lower*** (l)	***middle*** (m)	***upper*** (u)		
F1 score	3.014	3.475	3.878	0.076	0.102	0.135	0.105	0.101
AUC ROC	3.014	3.475	3.878	0.076	0.102	0.135	0.105	0.101
G-Mean1	3.014	3.475	3.878	0.076	0.102	0.135	0.105	0.101
G-Mean2	3.014	3.475	3.878	0.076	0.102	0.135	0.105	0.101
MCC	3.014	3.475	3.878	0.076	0.102	0.135	0.105	0.101
Kappa	3.014	3.475	3.878	0.076	0.102	0.135	0.105	0.101
Recall	1.385	1.727	2.153	0.035	0.051	0.075	0.054	0.052
Precision	1.385	1.727	2.153	0.035	0.051	0.075	0.054	0.052
Inverse Recall	1.385	1.727	2.153	0.035	0.051	0.075	0.054	0.052
Inverse Precision	1.385	1.727	2.153	0.035	0.051	0.075	0.054	0.052
(FDR)	0.696	0.864	1.077	0.018	0.025	0.038	0.027	0.026
(FOR)	0.696	0.864	1.077	0.018	0.025	0.038	0.027	0.026
(FPR)	0.696	0.864	1.077	0.018	0.025	0.038	0.027	0.026
(FNR)	0.696	0.864	1.077	0.018	0.025	0.038	0.027	0.026
Predicted Positive Condition Rate	0.361	0.439	0.541	0.009	0.013	0.019	0.014	0.013
(BACC)	0.361	0.439	0.541	0.009	0.013	0.019	0.014	0.013
Informedness	0.361	0.439	0.541	0.009	0.013	0.019	0.014	0.013
Markedness	0.361	0.439	0.541	0.009	0.013	0.019	0.014	0.013
(LR+)	0.211	0.236	0.272	0.005	0.007	0.009	0.007	0.007
(LR−)	0.211	0.236	0.272	0.005	0.007	0.009	0.007	0.007
(DOR)	0.211	0.236	0.272	0.005	0.007	0.009	0.007	0.007
Prevalence	0.211	0.236	0.272	0.005	0.007	0.009	0.007	0.007
Sum	**28.698**	**33.915**	**39.438**					
Sum^{-1}	**0.025**	**0.029**	**0.035**					

$$[c \times p] = \begin{bmatrix} s_{11} & \cdots & s_{1j} \\ \vdots & \ddots & \vdots \\ s_{i1} & \cdots & s_{ij} \end{bmatrix} \tag{7.2.8}$$

where $i = 1 \cdots c$ & $j = 1 \cdots p$.

Every entry s_{ij} of matrix S represents the score of the i^{th} classifier with respect to the j^{th} criterion. To construct the matrix S, we have first to compute classifiers' pair-wise comparison $B^{(j)}$ matrices with respect to every criterion j.

$$B^{(j)}[c \times c] = \begin{bmatrix} b_{11} & \cdots & b_{1h} \\ \vdots & \ddots & \vdots \\ b_{i1} & \cdots & b_{ih} \end{bmatrix} \tag{7.2.9}$$

where $i = 1 \cdots c$ & $h = 1 \cdots c$.

Each entry $b_{ih}^{(j)}$ of the matrix $B^{(j)}$ represents the evaluation of the i^{th} classifier compared to the h^{th} classifier with respect to the j^{th} criterion. We can compute $b_{ih}^{(j)}$ by dividing the performance evaluation of classifier i over the performance evaluation of classifier h with respect to the measure j. If $b_{ih}^{(j)} > 1$, then the i^{th}classifier is better than the h^{th}classifier, and if $b_{ih}^{(j)} < 1$, then the i^{th}classifier is worse than the h^{th}classifier. When two classifiers' performances are equal, then $b_{ih}^{(j)} = 1$. Matrix B entries satisfy the following properties:

$b_{ih}^{(j)} \cdot b_{hi}^{(j)} = 1$ and $b_{ih}^{(j)} = 1, \forall\ i = h$.

The matrix $E[c \times p]$ entries are utilized in computing $B^{(j)}$ matrices. The matrix E contains the performance evaluation of each classifier presented by the 22-performance measures. In total, we have 12 E matrices for the 12 data sets experimented. The process of computing E matrices is as follows:

```
1) Start KNIME
    $\boldsymbol{for\ each\ d \in D\ do}$    $\rhd$ $D$ is the set of 12 NASA data sets and $d$ is a data set
        2) Load data set $d$
        3) Run every classifier to generate its confusion matrix
        4) Use the generated confusion matrix to compute the 22-performance measures
        5) Construct the corresponding $E$ matrix
    $\boldsymbol{end\ for}$
```

Once $B^{(j)}$ matrices are computed, they need to be normalized column wise. That is, divide each entry b_{ih} in a particular column h over the sum of all entries of this column, equation (7.2.10). This operation is repeated for all columns in matrix $B^{(j)}$.

$$\overline{b}_{ih} = \frac{b_{ih}}{\sum_{i=1}^{c} b_{ih}} \tag{7.2.10}$$

We use equation (7.2.11) to find the scores vector $\boldsymbol{s}^{(j)}$ that contains the classifiers' pair-wise comparisons scores with respect to every criterion j. The c-dimension column vector $\boldsymbol{s}^{(j)}$ is computed by taking the averages row-wise for every row i in $B^{(j)}$.

$$\boldsymbol{s}^{(j)} = \frac{\sum_{h=1}^{c} \overline{b}_{ih}}{c} \tag{7.2.11}$$

Now, we can construct matrix S by combining all computed $\boldsymbol{s}^{(j)}$ scores vectors,

$$S = \left[\boldsymbol{s}^{(1)} \cdots \boldsymbol{s}^{(j)}\right] \text{, where } j = 1 \cdots p \tag{7.2.12}$$

Each column in the matrix S corresponds to one of the $\boldsymbol{s}^{(j)}$ column vectors.

Step 4:

Calculate the vector $\boldsymbol{v}$ of the classifiers' priorities by multiplying the classifiers' pair-wise comparison scores matrix S by the defuzzified criteria weights vector $\boldsymbol{w}$, equation (7.2.13).

$$\boldsymbol{v} \;=\; S \cdot \boldsymbol{w} \tag{7.2.13}$$

Each $\boldsymbol{v}_i$ entry represents the score (i.e. rank) assigned by the FAHP process to the i^{th} classifier in comparison to all other $(c - 1)$ classifiers.

7.3 Results

The experiments resulted in 12 E matrices, 12 S matrices, and 264 B matrices. For brevity, we will report the summary of the results. The appendices A and C contain matrices E and S respectively.

We can notice from table (7.3.1) that every data set reveals a unique order of the experimented classifiers' performance ranks. These results conform to much-published research that every data set (i.e. software project) is a unique product and possesses unique characteristics. Kastro et al. [73] concluded that it is almost impossible to have two identical software products in terms of developing process, programming languages used, programmers' experience, algorithm complexity, or even the development methodology. Myrtveit et al. [8] reported similar findings.

Table (7.3.1) Classifiers' ranks per every data set.

	Data sets											
Ranks	CM1	JM1	KC1	KC3	MC1	MC2	MW1	PC1	PC2	PC3	PC4	PC5
1	MLP	***RF***	***RF***	CART	***RF***	MLP	***RF***	FR	FR	***RF***	***RF***	***RF***
2	CART	FR	SOTA	DT	FR	NB	FR	***RF***	DT	FR	FR	FR
3	FR	CART	CART	FR	DT	***RF***	DT	SOTA	***RF***	CART	MLP	DT
4	DT	DT	DT	PNN	KNN	FR	CART	CART	MLP	DT	CART	CART
5	***RF***	KNN	FR	MLP	CART	KNN	MLP	DT	KNN	SOTA	KNN	MLP
6	SOTA	SOTA	KNN	***RF***	SOTA	CART	LR	KNN	SOTA	KNN	PNN	PNN
7	LR	MLP	PNN	KNN	MLP	LR	SOTA	MLP	CART	MLP	DT	KNN
8	KNN	LR	MLP	SOTA	PNN	PNN	PNN	PNN	LR	PNN	SOTA	SOTA
9	PNN	PNN	LR	NB	LR	SOTA	KNN	LR	PNN	LR	LR	LR
10	NB	NB	NB	LR	NB	DT	NB	NB	NB	NB	NB	NB
11	SVM	SVM	SVM	SVM	SVM	SVM	SVM	SVM	SVM	SVM	SVM	SVM

However, some interesting trends can be inferred. Random Forest ***RF*** has won the first rank 7 times and the second rank once. Fuzzy Rule FR has won the first rank twice and the second rank 6 times. This shows that these particular classifiers perform very well. On the contrary, SVM has won the last rank (i.e. the 11th rank) 12 times, which implies that this classifier consistently performs poorly in these experiments. Close to this performance is NB that won the 10th rank 10 times, the 9th rank once and surprisingly won the 2nd rank once too.

To make clear the final comparisons among all the competing classifiers, table (7.3.2) shows the average rank for each classifier over all experimented data sets. The procedure we follow is to count the number of times each classifier achieves a particular rank, then multiply this number by the rank itself. The sum of these numbers is divided by the total number of available ranks. Small average rank values indicate better performing classifiers, in comparison to classifiers having larger averages. Table (7.3.2) confirms our earlier observations in this section.

Table (7.3.2) Averaged Data Sets Ranks.

	Classifiers										
Ranks	RF	FR	CART	DT	MLP	KNN	SOTA	PNN	LR	NB	SVM
1	7	2	1	0	2	0	0	0	0	0	0
2	2	12	2	4	0	0	2	0	0	2	0
3	6	6	9	9	3	0	3	0	0	0	0
4	0	4	16	16	4	4	0	4	0	0	0
5	5	5	5	5	15	20	5	0	0	0	0
6	6	0	6	0	0	18	24	12	6	0	0
7	0	0	7	7	28	14	7	7	14	0	0
8	0	0	0	0	8	8	24	40	16	0	0
9	0	0	0	0	0	9	9	27	54	9	0
10	0	0	0	10	0	0	0	0	10	100	0
11	0	0	0	0	0	0	0	0	0	0	132
Sum	26	29	46	51	60	73	74	90	100	111	132
Average	2.4	2.6	4.2	4.6	5.5	6.6	6.7	8.2	9.1	10.1	12.0

On the other hand, we averaged the matrix E for the 12 data sets and applied FAHP to this one averaged matrix. As expected, the final rankings perfectly match the previous ones.

7.4 Threats to validity

The first threat to validity comes from the fact that this dissertation results and conclusions are biased in favor of the data sets and classifiers we used [74]. However, we believe that by choosing the publicly available NASA data sets, replication should be possible and would be encouraged by other researchers. The same argument applies for choosing the most common classifiers in the field of software defect prediction [4] [31] [73]. Moreover, NASA data sets meet all the requirements that would increase the external validity of our research, as stated by Khoshgoftaar et al. [75], that is, increasing the generalization of the results outside our experimental settings:

- Be large enough to be comparable to real industry projects
- Developed in an industrial environment, rather than an artificial setting
- Developed by a group of developers rather than an individual
- Developed by professionals, rather than students

On the other hand, and in order to decrease the presence of internal validity threats, we decided to use the cleaned-up NASA data sets instead of the original ones, as discussed earlier in section (7.1). This allows us to avoid the noise sources existing in the original NASA data sets.

Moreover, some data sets contain a relatively small number of modules, such as, MC2 and KC3, especially when the 10-fold cross-validation technique is employed. Some classifiers that are sensitive to the size of data sets might lose some of their performance quality [76]. This effect might be increased after performing the cleaning procedures on NASA data sets. As table (7.1.2) shows, this resulted in a smaller number of observations for each experimented data set.

CHAPTER 8: CONCLUSIONS

There is a substantial need to design and develop reliable software defect classifiers that classify software components into defective and non-defective. The benefit of achieving this objective is the ability to focus software defect- detection efforts and project resources on part of a system, rather than testing the whole system.

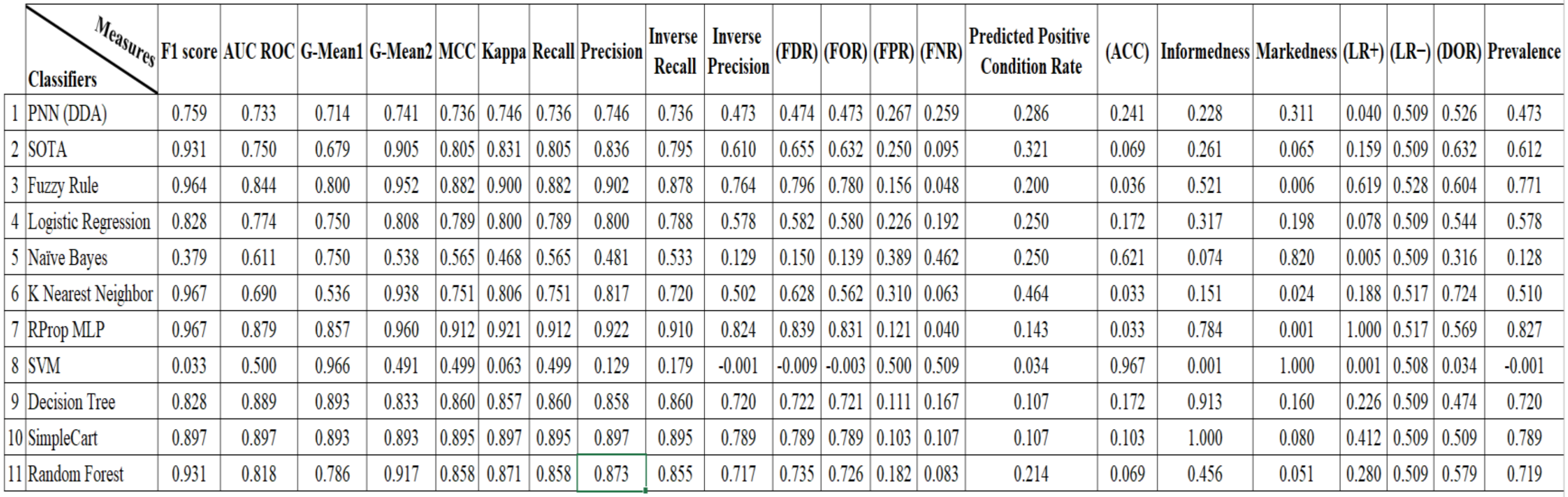
However, the major problem that software practitioners face is how to reliably evaluate classifiers and how to select the best fit for their software development projects. Since the evaluation of software defect classifiers' performance is highly dependent on the specific measures employed, the performance evaluation might improve or deteriorate, if practitioners choose different performance measures.

As we believe that performance evaluation must be seen as a comprehensive strategy rather than relying on preferred selection of performance measures, Fuzzy Analytical Hierarchy Process FAHP is used in this research to satisfy this requirement. FAHP allowed us to combine a wider spectrum of evaluation measures, in contrast to relying on preferred selection of one or two evaluation measures. Another strength comes from the fact that FAHP employs fuzzy membership function to account for human nature of uncertainty and vagueness when evaluating and comparing performance measures with one another. The results show that this approach will increase software developers' confidence in research outcomes, help them in avoiding false conclusions and providing them with reasonable boundaries.

APPENDIX A: $\boldsymbol{E}$ MATRICES

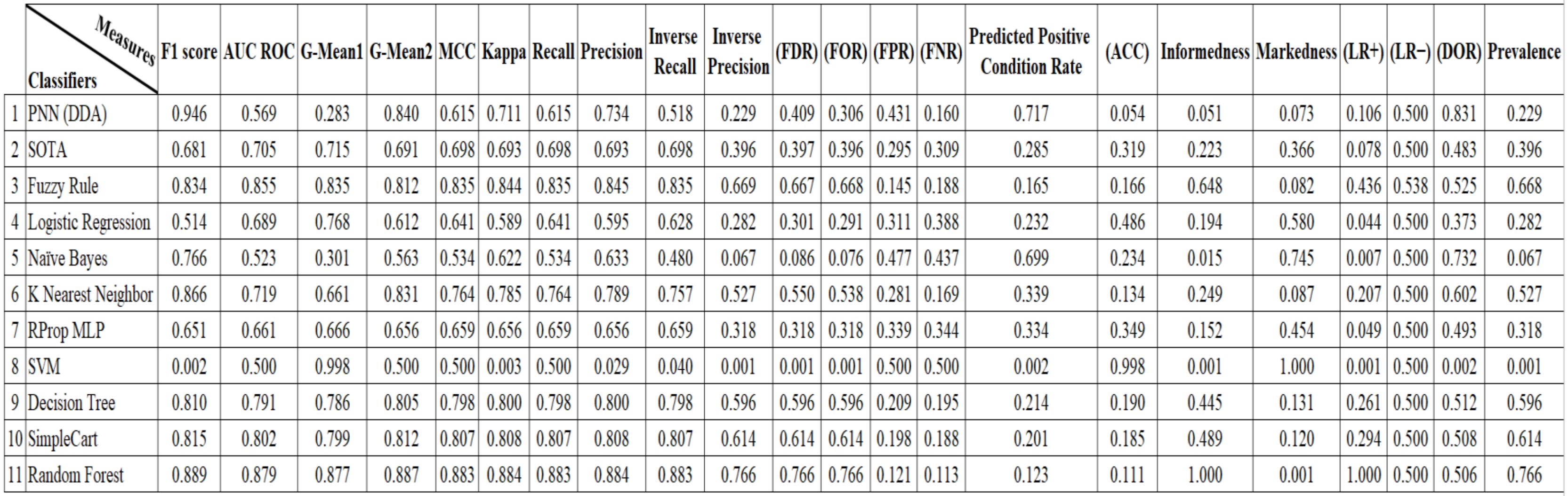
$E[c \times p]$ is the classifiers' performance evaluation matrix computed for 12 NASA software defect data sets.

Data Set: CM1



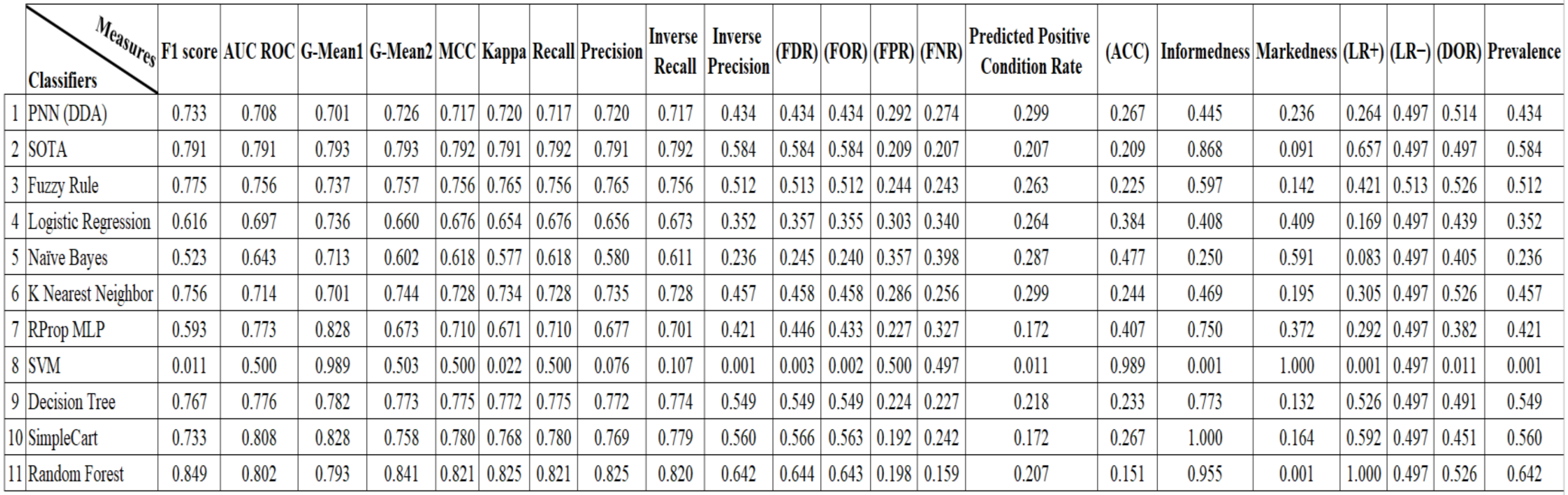
	Measures / Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.759	0.733	0.714	0.741	0.736	0.746	0.736	0.746	0.736	0.473	0.474	0.473	0.267	0.259	0.286	0.241	0.228	0.311	0.040	0.509	0.526	0.473
2	SOTA	0.931	0.750	0.679	0.905	0.805	0.831	0.805	0.836	0.795	0.610	0.655	0.632	0.250	0.095	0.321	0.069	0.261	0.065	0.159	0.509	0.632	0.612
3	Fuzzy Rule	0.964	0.844	0.800	0.952	0.882	0.900	0.882	0.902	0.878	0.764	0.796	0.780	0.156	0.048	0.200	0.036	0.521	0.006	0.619	0.528	0.604	0.771
4	Logistic Regression	0.828	0.774	0.750	0.808	0.789	0.800	0.789	0.800	0.788	0.578	0.582	0.580	0.226	0.192	0.250	0.172	0.317	0.198	0.078	0.509	0.544	0.578
5	Naïve Bayes	0.379	0.611	0.750	0.538	0.565	0.468	0.565	0.481	0.533	0.129	0.150	0.139	0.389	0.462	0.250	0.621	0.074	0.820	0.005	0.509	0.316	0.128
6	K Nearest Neighbor	0.967	0.690	0.536	0.938	0.751	0.806	0.751	0.817	0.720	0.502	0.628	0.562	0.310	0.063	0.464	0.033	0.151	0.024	0.188	0.517	0.724	0.510
7	RProp MLP	0.967	0.879	0.857	0.960	0.912	0.921	0.912	0.922	0.910	0.824	0.839	0.831	0.121	0.040	0.143	0.033	0.784	0.001	1.000	0.517	0.569	0.827
8	SVM	0.033	0.500	0.966	0.491	0.499	0.063	0.499	0.129	0.179	-0.001	-0.009	-0.003	0.500	0.509	0.034	0.967	0.001	1.000	0.001	0.508	0.034	-0.001
9	Decision Tree	0.828	0.889	0.893	0.833	0.860	0.857	0.860	0.858	0.860	0.720	0.722	0.721	0.111	0.167	0.107	0.172	0.913	0.160	0.226	0.509	0.474	0.720
10	SimpleCart	0.897	0.897	0.893	0.893	0.895	0.897	0.895	0.897	0.895	0.789	0.789	0.789	0.103	0.107	0.107	0.103	1.000	0.080	0.412	0.509	0.509	0.789
11	Random Forest	0.931	0.818	0.786	0.917	0.858	0.871	0.858	0.873	0.855	0.717	0.735	0.726	0.182	0.083	0.214	0.069	0.456	0.051	0.280	0.509	0.579	0.719

Data Set: JM1



	Measures Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.946	0.569	0.283	0.840	0.615	0.711	0.615	0.734	0.518	0.229	0.409	0.306	0.431	0.160	0.717	0.054	0.051	0.073	0.106	0.500	0.831	0.229
2	SOTA	0.681	0.705	0.715	0.691	0.698	0.693	0.698	0.693	0.698	0.396	0.397	0.396	0.295	0.309	0.285	0.319	0.223	0.366	0.078	0.500	0.483	0.396
3	Fuzzy Rule	0.834	0.855	0.835	0.812	0.835	0.844	0.835	0.845	0.835	0.669	0.667	0.668	0.145	0.188	0.165	0.166	0.648	0.082	0.436	0.538	0.525	0.668
4	Logistic Regression	0.514	0.689	0.768	0.612	0.641	0.589	0.641	0.595	0.628	0.282	0.301	0.291	0.311	0.388	0.232	0.486	0.194	0.580	0.044	0.500	0.373	0.282
5	Naïve Bayes	0.766	0.523	0.301	0.563	0.534	0.622	0.534	0.633	0.480	0.067	0.086	0.076	0.477	0.437	0.699	0.234	0.015	0.745	0.007	0.500	0.732	0.067
6	K Nearest Neighbor	0.866	0.719	0.661	0.831	0.764	0.785	0.764	0.789	0.757	0.527	0.550	0.538	0.281	0.169	0.339	0.134	0.249	0.087	0.207	0.500	0.602	0.527
7	RProp MLP	0.651	0.661	0.666	0.656	0.659	0.656	0.659	0.656	0.659	0.318	0.318	0.318	0.339	0.344	0.334	0.349	0.152	0.454	0.049	0.500	0.493	0.318
8	SVM	0.002	0.500	0.998	0.500	0.500	0.003	0.500	0.029	0.040	0.001	0.001	0.001	0.500	0.500	0.002	0.998	0.001	1.000	0.001	0.500	0.002	0.001
9	Decision Tree	0.810	0.791	0.786	0.805	0.798	0.800	0.798	0.800	0.798	0.596	0.596	0.596	0.209	0.195	0.214	0.190	0.445	0.131	0.261	0.500	0.512	0.596
10	SimpleCart	0.815	0.802	0.799	0.812	0.807	0.808	0.807	0.808	0.807	0.614	0.614	0.614	0.198	0.188	0.201	0.185	0.489	0.120	0.294	0.500	0.508	0.614
11	Random Forest	0.889	0.879	0.877	0.887	0.883	0.884	0.883	0.884	0.883	0.766	0.766	0.766	0.121	0.113	0.123	0.111	1.000	0.001	1.000	0.500	0.506	0.766

Data Set: KC1

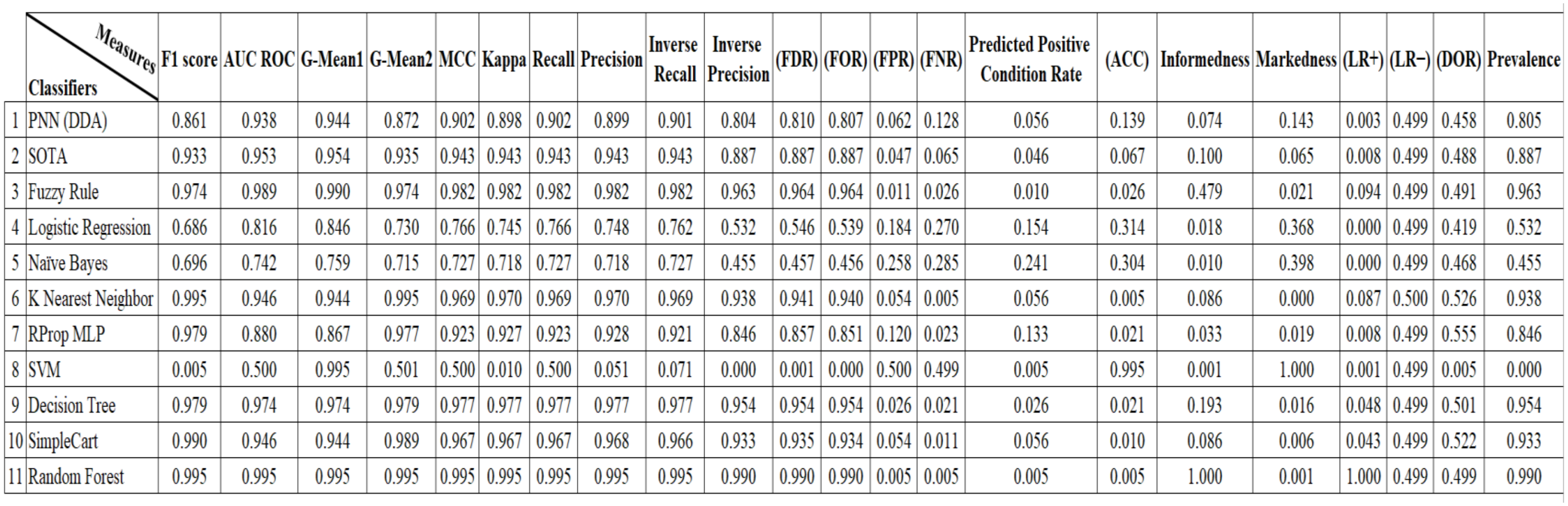


	Measures / Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.733	0.708	0.701	0.726	0.717	0.720	0.717	0.720	0.717	0.434	0.434	0.434	0.292	0.274	0.299	0.267	0.445	0.236	0.264	0.497	0.514	0.434
2	SOTA	0.791	0.791	0.793	0.793	0.792	0.791	0.792	0.791	0.792	0.584	0.584	0.584	0.209	0.207	0.207	0.209	0.868	0.091	0.657	0.497	0.497	0.584
3	Fuzzy Rule	0.775	0.756	0.737	0.757	0.756	0.765	0.756	0.765	0.756	0.512	0.513	0.512	0.244	0.243	0.263	0.225	0.597	0.142	0.421	0.513	0.526	0.512
4	Logistic Regression	0.616	0.697	0.736	0.660	0.676	0.654	0.676	0.656	0.673	0.352	0.357	0.355	0.303	0.340	0.264	0.384	0.408	0.409	0.169	0.497	0.439	0.352
5	Naïve Bayes	0.523	0.643	0.713	0.602	0.618	0.577	0.618	0.580	0.611	0.236	0.245	0.240	0.357	0.398	0.287	0.477	0.250	0.591	0.083	0.497	0.405	0.236
6	K Nearest Neighbor	0.756	0.714	0.701	0.744	0.728	0.734	0.728	0.735	0.728	0.457	0.458	0.458	0.286	0.256	0.299	0.244	0.469	0.195	0.305	0.497	0.526	0.457
7	RProp MLP	0.593	0.773	0.828	0.673	0.710	0.671	0.710	0.677	0.701	0.421	0.446	0.433	0.227	0.327	0.172	0.407	0.750	0.372	0.292	0.497	0.382	0.421
8	SVM	0.011	0.500	0.989	0.503	0.500	0.022	0.500	0.076	0.107	0.001	0.003	0.002	0.500	0.497	0.011	0.989	0.001	1.000	0.001	0.497	0.011	0.001
9	Decision Tree	0.767	0.776	0.782	0.773	0.775	0.772	0.775	0.772	0.774	0.549	0.549	0.549	0.224	0.227	0.218	0.233	0.773	0.132	0.526	0.497	0.491	0.549
10	SimpleCart	0.733	0.808	0.828	0.758	0.780	0.768	0.780	0.769	0.779	0.560	0.566	0.563	0.192	0.242	0.172	0.267	1.000	0.164	0.592	0.497	0.451	0.560
11	Random Forest	0.849	0.802	0.793	0.841	0.821	0.825	0.821	0.825	0.820	0.642	0.644	0.643	0.198	0.159	0.207	0.151	0.955	0.001	1.000	0.497	0.526	0.642

Data Set: KC3

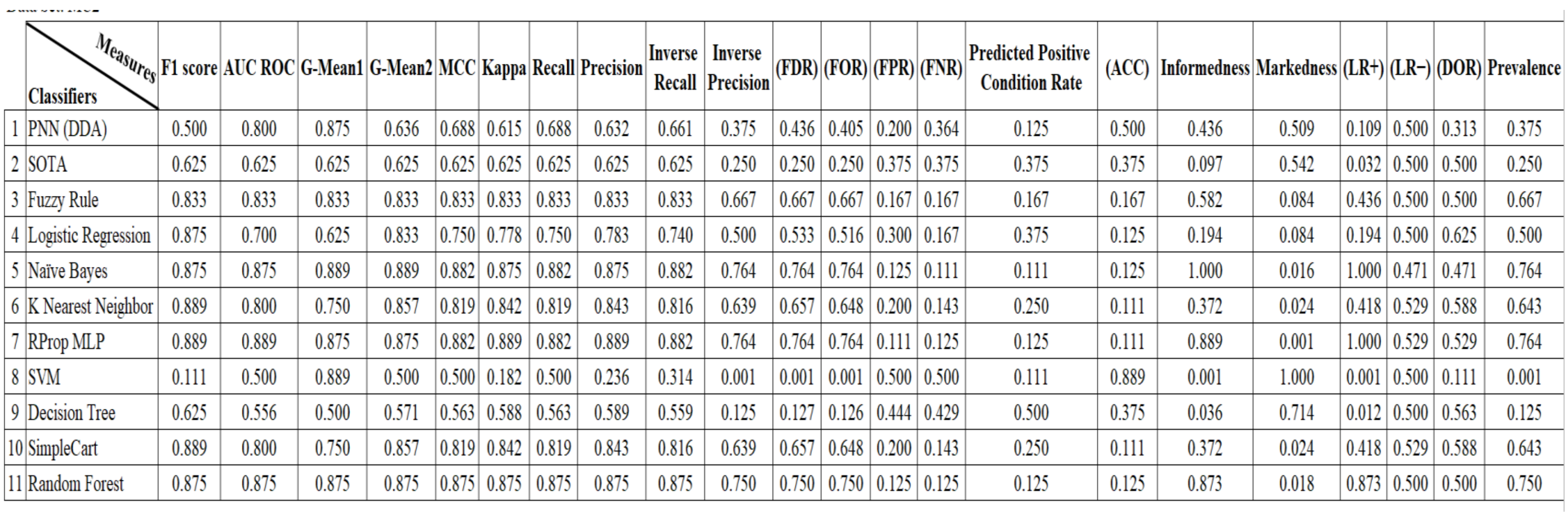
	Measures Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.867	0.867	0.875	0.875	0.871	0.867	0.871	0.867	0.871	0.742	0.742	0.742	0.133	0.125	0.125	0.133	0.421	0.092	0.198	0.484	0.484	0.742
2	SOTA	0.800	0.706	0.688	0.786	0.744	0.750	0.744	0.751	0.742	0.488	0.492	0.490	0.294	0.214	0.313	0.200	0.107	0.241	0.035	0.484	0.548	0.485
3	Fuzzy Rule	0.867	0.929	0.933	0.875	0.900	0.897	0.900	0.897	0.899	0.800	0.804	0.802	0.071	0.125	0.067	0.133	0.857	0.082	0.402	0.500	0.467	0.800
4	Logistic Regression	0.600	0.750	0.813	0.684	0.706	0.667	0.706	0.671	0.698	0.413	0.434	0.423	0.250	0.316	0.188	0.400	0.153	0.458	0.024	0.484	0.387	0.415
5	Naïve Bayes	0.600	0.818	0.875	0.700	0.738	0.692	0.738	0.701	0.725	0.475	0.518	0.496	0.182	0.300	0.125	0.400	0.268	0.420	0.042	0.484	0.355	0.479
6	K Nearest Neighbor	0.938	0.714	0.625	0.909	0.781	0.811	0.781	0.818	0.765	0.563	0.623	0.592	0.286	0.091	0.375	0.063	0.103	0.036	0.107	0.500	0.656	0.563
7	RProp MLP	0.867	0.813	0.813	0.867	0.840	0.839	0.840	0.839	0.839	0.679	0.679	0.679	0.188	0.133	0.188	0.133	0.255	0.105	0.121	0.484	0.516	0.678
8	SVM	0.063	0.500	0.941	0.516	0.502	0.111	0.502	0.177	0.243	0.004	0.016	0.008	0.500	0.484	0.059	0.938	0.001	1.000	0.001	0.485	0.061	0.004
9	Decision Tree	0.867	0.929	0.938	0.882	0.902	0.897	0.902	0.897	0.901	0.804	0.811	0.808	0.071	0.118	0.063	0.133	0.919	0.081	0.431	0.484	0.452	0.806
10	SimpleCart	0.938	0.938	0.938	0.938	0.938	0.938	0.938	0.938	0.938	0.875	0.875	0.875	0.063	0.063	0.063	0.063	1.000	0.001	1.000	0.500	0.500	0.875
11	Random Forest	0.867	0.813	0.813	0.867	0.840	0.839	0.840	0.839	0.839	0.679	0.679	0.679	0.188	0.133	0.188	0.133	0.255	0.105	0.121	0.484	0.516	0.678

Data Set: MC1



	Measures / Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.861	0.938	0.944	0.872	0.902	0.898	0.902	0.899	0.901	0.804	0.810	0.807	0.062	0.128	0.056	0.139	0.074	0.143	0.003	0.499	0.458	0.805
2	SOTA	0.933	0.953	0.954	0.935	0.943	0.943	0.943	0.943	0.943	0.887	0.887	0.887	0.047	0.065	0.046	0.067	0.100	0.065	0.008	0.499	0.488	0.887
3	Fuzzy Rule	0.974	0.989	0.990	0.974	0.982	0.982	0.982	0.982	0.982	0.963	0.964	0.964	0.011	0.026	0.010	0.026	0.479	0.021	0.094	0.499	0.491	0.963
4	Logistic Regression	0.686	0.816	0.846	0.730	0.766	0.745	0.766	0.748	0.762	0.532	0.546	0.539	0.184	0.270	0.154	0.314	0.018	0.368	0.000	0.499	0.419	0.532
5	Naïve Bayes	0.696	0.742	0.759	0.715	0.727	0.718	0.727	0.718	0.727	0.455	0.457	0.456	0.258	0.285	0.241	0.304	0.010	0.398	0.000	0.499	0.468	0.455
6	K Nearest Neighbor	0.995	0.946	0.944	0.995	0.969	0.970	0.969	0.970	0.969	0.938	0.941	0.940	0.054	0.005	0.056	0.005	0.086	0.000	0.087	0.500	0.526	0.938
7	RProp MLP	0.979	0.880	0.867	0.977	0.923	0.927	0.923	0.928	0.921	0.846	0.857	0.851	0.120	0.023	0.133	0.021	0.033	0.019	0.008	0.499	0.555	0.846
8	SVM	0.005	0.500	0.995	0.501	0.500	0.010	0.500	0.051	0.071	0.000	0.001	0.000	0.500	0.499	0.005	0.995	0.001	1.000	0.001	0.499	0.005	0.000
9	Decision Tree	0.979	0.974	0.974	0.979	0.977	0.977	0.977	0.977	0.977	0.954	0.954	0.954	0.026	0.021	0.026	0.021	0.193	0.016	0.048	0.499	0.501	0.954
10	SimpleCart	0.990	0.946	0.944	0.989	0.967	0.967	0.967	0.968	0.966	0.933	0.935	0.934	0.054	0.011	0.056	0.010	0.086	0.006	0.043	0.499	0.522	0.933
11	Random Forest	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.990	0.990	0.990	0.005	0.005	0.005	0.005	1.000	0.001	1.000	0.499	0.499	0.990

Data Set: MC2

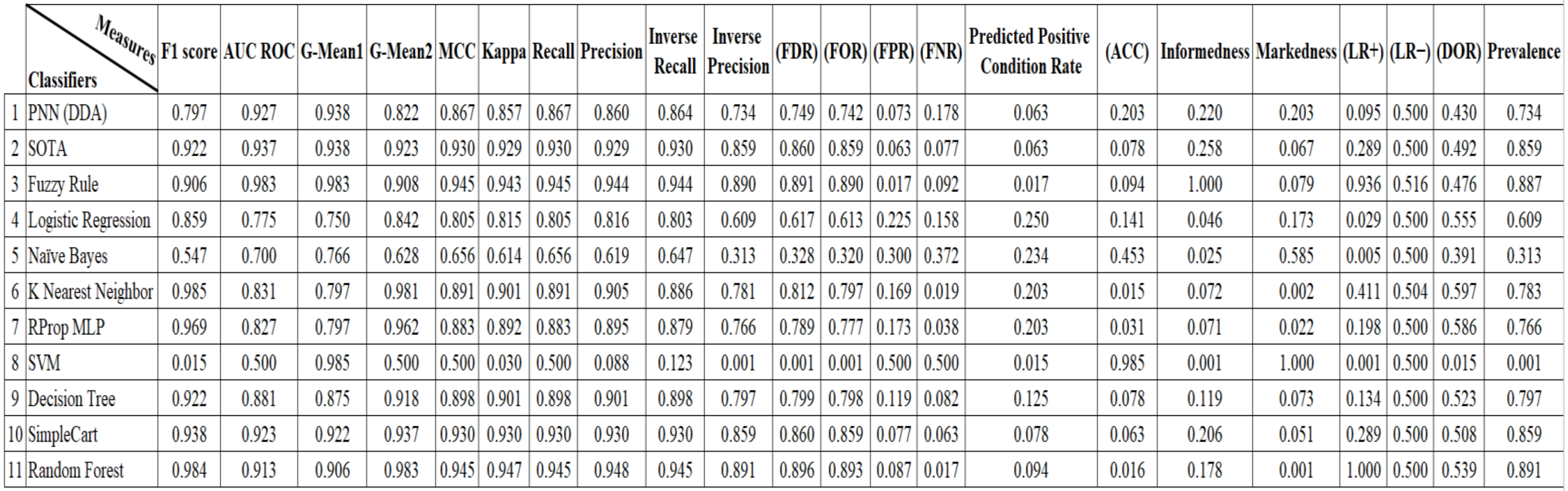


	Measures / Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.500	0.800	0.875	0.636	0.688	0.615	0.688	0.632	0.661	0.375	0.436	0.405	0.200	0.364	0.125	0.500	0.436	0.509	0.109	0.500	0.313	0.375
2	SOTA	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.250	0.250	0.250	0.375	0.375	0.375	0.375	0.097	0.542	0.032	0.500	0.500	0.250
3	Fuzzy Rule	0.833	0.833	0.833	0.833	0.833	0.833	0.833	0.833	0.833	0.667	0.667	0.667	0.167	0.167	0.167	0.167	0.582	0.084	0.436	0.500	0.500	0.667
4	Logistic Regression	0.875	0.700	0.625	0.833	0.750	0.778	0.750	0.783	0.740	0.500	0.533	0.516	0.300	0.167	0.375	0.125	0.194	0.084	0.194	0.500	0.625	0.500
5	Naïve Bayes	0.875	0.875	0.889	0.889	0.882	0.875	0.882	0.875	0.882	0.764	0.764	0.764	0.125	0.111	0.111	0.125	1.000	0.016	1.000	0.471	0.471	0.764
6	K Nearest Neighbor	0.889	0.800	0.750	0.857	0.819	0.842	0.819	0.843	0.816	0.639	0.657	0.648	0.200	0.143	0.250	0.111	0.372	0.024	0.418	0.529	0.588	0.643
7	RProp MLP	0.889	0.889	0.875	0.875	0.882	0.889	0.882	0.889	0.882	0.764	0.764	0.764	0.111	0.125	0.125	0.111	0.889	0.001	1.000	0.529	0.529	0.764
8	SVM	0.111	0.500	0.889	0.500	0.500	0.182	0.500	0.236	0.314	0.001	0.001	0.001	0.500	0.500	0.111	0.889	0.001	1.000	0.001	0.500	0.111	0.001
9	Decision Tree	0.625	0.556	0.500	0.571	0.563	0.588	0.563	0.589	0.559	0.125	0.127	0.126	0.444	0.429	0.500	0.375	0.036	0.714	0.012	0.500	0.563	0.125
10	SimpleCart	0.889	0.800	0.750	0.857	0.819	0.842	0.819	0.843	0.816	0.639	0.657	0.648	0.200	0.143	0.250	0.111	0.372	0.024	0.418	0.529	0.588	0.643
11	Random Forest	0.875	0.875	0.875	0.875	0.875	0.875	0.875	0.875	0.875	0.750	0.750	0.750	0.125	0.125	0.125	0.125	0.873	0.018	0.873	0.500	0.500	0.750

Data Set: MW1

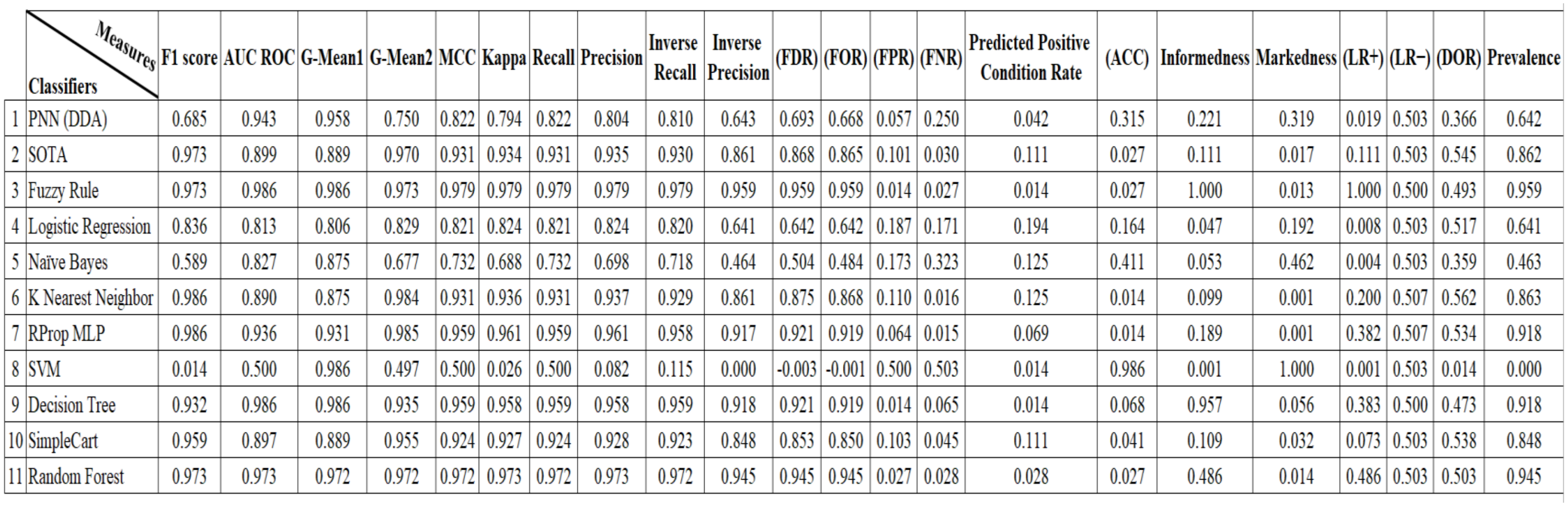
	Measures Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.783	0.750	0.727	0.762	0.755	0.766	0.755	0.766	0.754	0.510	0.512	0.511	0.250	0.238	0.273	0.217	0.316	0.261	0.062	0.511	0.533	0.510
2	SOTA	0.957	0.710	0.591	0.929	0.774	0.815	0.774	0.824	0.752	0.547	0.638	0.591	0.290	0.071	0.409	0.043	0.228	0.024	0.223	0.511	0.689	0.552
3	Fuzzy Rule	0.958	0.821	0.762	0.941	0.860	0.885	0.860	0.887	0.854	0.720	0.763	0.741	0.179	0.059	0.238	0.042	0.506	0.005	0.525	0.533	0.622	0.729
4	Logistic Regression	0.696	0.842	0.864	0.731	0.780	0.762	0.780	0.765	0.775	0.559	0.573	0.566	0.158	0.269	0.136	0.304	0.684	0.317	0.098	0.511	0.422	0.557
5	Naïve Bayes	0.565	0.722	0.773	0.630	0.669	0.634	0.669	0.639	0.661	0.338	0.352	0.345	0.278	0.370	0.227	0.435	0.252	0.538	0.025	0.511	0.400	0.336
6	K Nearest Neighbor	0.958	0.697	0.545	0.923	0.752	0.807	0.752	0.817	0.723	0.504	0.620	0.559	0.303	0.077	0.455	0.042	0.190	0.027	0.193	0.522	0.717	0.513
7	RProp MLP	0.958	0.742	0.636	0.933	0.797	0.836	0.797	0.843	0.781	0.595	0.675	0.634	0.258	0.067	0.364	0.042	0.277	0.016	0.284	0.522	0.674	0.603
8	SVM	0.042	0.500	0.957	0.489	0.499	0.077	0.499	0.144	0.200	-0.002	-0.011	-0.004	0.500	0.511	0.043	0.958	0.001	1.000	0.001	0.511	0.043	-0.002
9	Decision Tree	0.826	0.864	0.864	0.826	0.845	0.844	0.845	0.845	0.845	0.690	0.690	0.690	0.136	0.174	0.136	0.174	0.842	0.159	0.210	0.511	0.489	0.689
10	SimpleCart	0.870	0.800	0.773	0.850	0.821	0.833	0.821	0.834	0.820	0.642	0.650	0.646	0.200	0.150	0.227	0.130	0.474	0.124	0.157	0.511	0.556	0.644
11	Random Forest	0.957	0.880	0.864	0.950	0.910	0.917	0.910	0.917	0.909	0.820	0.830	0.825	0.120	0.050	0.136	0.043	1.000	0.001	1.000	0.511	0.556	0.822

Data Set: PC1



	Measures Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.797	0.927	0.938	0.822	0.867	0.857	0.867	0.860	0.864	0.734	0.749	0.742	0.073	0.178	0.063	0.203	0.220	0.203	0.095	0.500	0.430	0.734
2	SOTA	0.922	0.937	0.938	0.923	0.930	0.929	0.930	0.929	0.930	0.859	0.860	0.859	0.063	0.077	0.063	0.078	0.258	0.067	0.289	0.500	0.492	0.859
3	Fuzzy Rule	0.906	0.983	0.983	0.908	0.945	0.943	0.945	0.944	0.944	0.890	0.891	0.890	0.017	0.092	0.017	0.094	1.000	0.079	0.936	0.516	0.476	0.887
4	Logistic Regression	0.859	0.775	0.750	0.842	0.805	0.815	0.805	0.816	0.803	0.609	0.617	0.613	0.225	0.158	0.250	0.141	0.046	0.173	0.029	0.500	0.555	0.609
5	Naïve Bayes	0.547	0.700	0.766	0.628	0.656	0.614	0.656	0.619	0.647	0.313	0.328	0.320	0.300	0.372	0.234	0.453	0.025	0.585	0.005	0.500	0.391	0.313
6	K Nearest Neighbor	0.985	0.831	0.797	0.981	0.891	0.901	0.891	0.905	0.886	0.781	0.812	0.797	0.169	0.019	0.203	0.015	0.072	0.002	0.411	0.504	0.597	0.783
7	RProp MLP	0.969	0.827	0.797	0.962	0.883	0.892	0.883	0.895	0.879	0.766	0.789	0.777	0.173	0.038	0.203	0.031	0.071	0.022	0.198	0.500	0.586	0.766
8	SVM	0.015	0.500	0.985	0.500	0.500	0.030	0.500	0.088	0.123	0.001	0.001	0.001	0.500	0.500	0.015	0.985	0.001	1.000	0.001	0.500	0.015	0.001
9	Decision Tree	0.922	0.881	0.875	0.918	0.898	0.901	0.898	0.901	0.898	0.797	0.799	0.798	0.119	0.082	0.125	0.078	0.119	0.073	0.134	0.500	0.523	0.797
10	SimpleCart	0.938	0.923	0.922	0.937	0.930	0.930	0.930	0.930	0.930	0.859	0.860	0.859	0.077	0.063	0.078	0.063	0.206	0.051	0.289	0.500	0.508	0.859
11	Random Forest	0.984	0.913	0.906	0.983	0.945	0.947	0.945	0.948	0.945	0.891	0.896	0.893	0.087	0.017	0.094	0.016	0.178	0.001	1.000	0.500	0.539	0.891

Data Set: PC2



	Measures Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.685	0.943	0.958	0.750	0.822	0.794	0.822	0.804	0.810	0.643	0.693	0.668	0.057	0.250	0.042	0.315	0.221	0.319	0.019	0.503	0.366	0.642
2	SOTA	0.973	0.899	0.889	0.970	0.931	0.934	0.931	0.935	0.930	0.861	0.868	0.865	0.101	0.030	0.111	0.027	0.111	0.017	0.111	0.503	0.545	0.862
3	Fuzzy Rule	0.973	0.986	0.986	0.973	0.979	0.979	0.979	0.979	0.979	0.959	0.959	0.959	0.014	0.027	0.014	0.027	1.000	0.013	1.000	0.500	0.493	0.959
4	Logistic Regression	0.836	0.813	0.806	0.829	0.821	0.824	0.821	0.824	0.820	0.641	0.642	0.642	0.187	0.171	0.194	0.164	0.047	0.192	0.008	0.503	0.517	0.641
5	Naïve Bayes	0.589	0.827	0.875	0.677	0.732	0.688	0.732	0.698	0.718	0.464	0.504	0.484	0.173	0.323	0.125	0.411	0.053	0.462	0.004	0.503	0.359	0.463
6	K Nearest Neighbor	0.986	0.890	0.875	0.984	0.931	0.936	0.931	0.937	0.929	0.861	0.875	0.868	0.110	0.016	0.125	0.014	0.099	0.001	0.200	0.507	0.562	0.863
7	RProp MLP	0.986	0.936	0.931	0.985	0.959	0.961	0.959	0.961	0.958	0.917	0.921	0.919	0.064	0.015	0.069	0.014	0.189	0.001	0.382	0.507	0.534	0.918
8	SVM	0.014	0.500	0.986	0.497	0.500	0.026	0.500	0.082	0.115	0.000	-0.003	-0.001	0.500	0.503	0.014	0.986	0.001	1.000	0.001	0.503	0.014	0.000
9	Decision Tree	0.932	0.986	0.986	0.935	0.959	0.958	0.959	0.958	0.959	0.918	0.921	0.919	0.014	0.065	0.014	0.068	0.957	0.056	0.383	0.500	0.473	0.918
10	SimpleCart	0.959	0.897	0.889	0.955	0.924	0.927	0.924	0.928	0.923	0.848	0.853	0.850	0.103	0.045	0.111	0.041	0.109	0.032	0.073	0.503	0.538	0.848
11	Random Forest	0.973	0.973	0.972	0.972	0.972	0.973	0.972	0.973	0.972	0.945	0.945	0.945	0.027	0.028	0.028	0.027	0.486	0.014	0.486	0.503	0.503	0.945

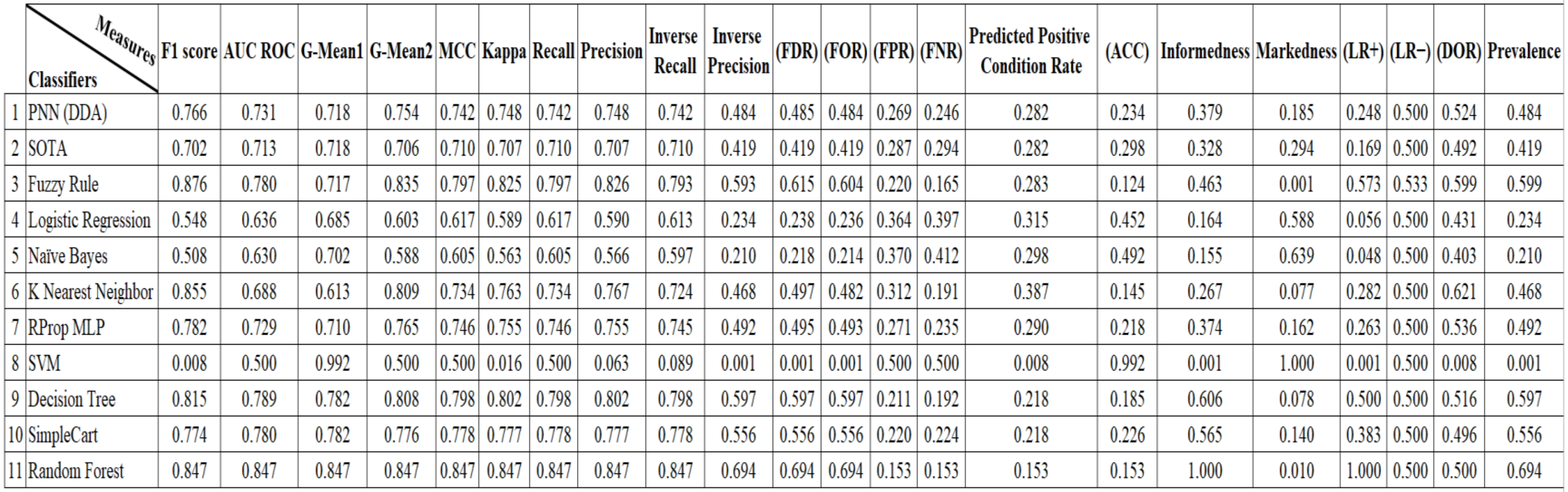
Data Set: PC3

	Measures Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.723	0.829	0.851	0.755	0.787	0.773	0.787	0.775	0.785	0.574	0.584	0.579	0.171	0.245	0.149	0.277	0.455	0.308	0.045	0.500	0.436	0.574
2	SOTA	0.894	0.792	0.766	0.878	0.830	0.840	0.830	0.842	0.827	0.660	0.671	0.665	0.208	0.122	0.234	0.106	0.333	0.117	0.085	0.500	0.564	0.660
3	Fuzzy Rule	0.902	0.912	0.905	0.894	0.903	0.907	0.903	0.907	0.903	0.807	0.806	0.807	0.088	0.106	0.095	0.098	1.000	0.086	0.276	0.523	0.517	0.807
4	Logistic Regression	0.681	0.744	0.766	0.706	0.723	0.711	0.723	0.712	0.722	0.447	0.450	0.448	0.256	0.294	0.234	0.319	0.225	0.402	0.019	0.500	0.457	0.447
5	Naïve Bayes	0.234	0.815	0.947	0.553	0.590	0.364	0.590	0.437	0.471	0.181	0.368	0.258	0.185	0.447	0.053	0.766	0.401	0.804	0.014	0.500	0.144	0.181
6	K Nearest Neighbor	0.968	0.746	0.670	0.955	0.819	0.843	0.819	0.850	0.805	0.638	0.700	0.669	0.254	0.045	0.330	0.032	0.228	0.024	0.194	0.500	0.649	0.638
7	RProp MLP	0.872	0.796	0.777	0.859	0.824	0.832	0.824	0.833	0.823	0.649	0.655	0.652	0.204	0.141	0.223	0.128	0.343	0.143	0.073	0.500	0.548	0.649
8	SVM	0.011	0.500	0.989	0.500	0.500	0.021	0.500	0.073	0.102	0.001	0.001	0.001	0.500	0.500	0.011	0.989	0.001	1.000	0.001	0.500	0.011	0.001
9	Decision Tree	0.787	0.860	0.872	0.804	0.830	0.822	0.830	0.823	0.829	0.660	0.664	0.662	0.140	0.196	0.128	0.213	0.610	0.225	0.078	0.500	0.457	0.660
10	SimpleCart	0.862	0.871	0.872	0.863	0.867	0.866	0.867	0.866	0.867	0.734	0.734	0.734	0.129	0.137	0.128	0.138	0.679	0.138	0.133	0.500	0.495	0.734
11	Random Forest	0.979	0.885	0.872	0.976	0.926	0.929	0.926	0.930	0.924	0.851	0.861	0.856	0.115	0.024	0.128	0.021	0.787	0.001	1.000	0.500	0.553	0.851

Data Set: PC4

	Measures Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.901	0.885	0.883	0.899	0.892	0.893	0.892	0.893	0.892	0.784	0.784	0.784	0.115	0.101	0.117	0.099	0.394	0.094	0.107	0.500	0.509	0.784
2	SOTA	0.910	0.849	0.838	0.903	0.874	0.878	0.874	0.879	0.873	0.748	0.752	0.750	0.151	0.097	0.162	0.090	0.271	0.089	0.081	0.500	0.536	0.748
3	Fuzzy Rule	0.962	0.943	0.941	0.960	0.951	0.952	0.951	0.952	0.951	0.902	0.903	0.903	0.057	0.040	0.059	0.038	0.893	0.021	0.628	0.507	0.517	0.902
4	Logistic Regression	0.766	0.842	0.856	0.785	0.811	0.802	0.811	0.803	0.810	0.622	0.627	0.624	0.158	0.215	0.144	0.234	0.254	0.259	0.029	0.500	0.455	0.622
5	Naïve Bayes	0.477	0.803	0.883	0.628	0.680	0.599	0.680	0.619	0.649	0.360	0.431	0.394	0.197	0.372	0.117	0.523	0.181	0.584	0.009	0.500	0.297	0.360
6	K Nearest Neighbor	0.982	0.838	0.811	0.978	0.896	0.905	0.896	0.907	0.892	0.793	0.817	0.805	0.162	0.022	0.189	0.018	0.246	0.002	0.370	0.500	0.586	0.793
7	RProp MLP	0.982	0.908	0.901	0.980	0.941	0.944	0.941	0.944	0.941	0.883	0.889	0.886	0.092	0.020	0.099	0.018	0.524	0.001	0.786	0.500	0.541	0.883
8	SVM	0.009	0.500	0.991	0.500	0.500	0.018	0.500	0.067	0.094	0.001	0.001	0.001	0.500	0.500	0.009	0.991	0.001	1.000	0.001	0.500	0.009	0.001
9	Decision Tree	0.883	0.875	0.874	0.882	0.878	0.879	0.878	0.879	0.878	0.757	0.757	0.757	0.125	0.118	0.126	0.117	0.353	0.116	0.081	0.500	0.505	0.757
10	SimpleCart	0.946	0.905	0.901	0.943	0.923	0.925	0.923	0.925	0.923	0.847	0.849	0.848	0.095	0.057	0.099	0.054	0.503	0.041	0.251	0.500	0.523	0.847
11	Random Forest	0.973	0.947	0.946	0.972	0.959	0.960	0.959	0.960	0.959	0.919	0.920	0.919	0.053	0.028	0.054	0.027	1.000	0.009	1.000	0.500	0.514	0.919

Data Set: PC5



	Measures / Classifiers	F1 score	AUC ROC	G-Mean1	G-Mean2	MCC	Kappa	Recall	Precision	Inverse Recall	Inverse Precision	(FDR)	(FOR)	(FPR)	(FNR)	Predicted Positive Condition Rate	(ACC)	Informedness	Markedness	(LR+)	(LR−)	(DOR)	Prevalence
1	PNN (DDA)	0.766	0.731	0.718	0.754	0.742	0.748	0.742	0.748	0.742	0.484	0.485	0.484	0.269	0.246	0.282	0.234	0.379	0.185	0.248	0.500	0.524	0.484
2	SOTA	0.702	0.713	0.718	0.706	0.710	0.707	0.710	0.707	0.710	0.419	0.419	0.419	0.287	0.294	0.282	0.298	0.328	0.294	0.169	0.500	0.492	0.419
3	Fuzzy Rule	0.876	0.780	0.717	0.835	0.797	0.825	0.797	0.826	0.793	0.593	0.615	0.604	0.220	0.165	0.283	0.124	0.463	0.001	0.573	0.533	0.599	0.599
4	Logistic Regression	0.548	0.636	0.685	0.603	0.617	0.589	0.617	0.590	0.613	0.234	0.238	0.236	0.364	0.397	0.315	0.452	0.164	0.588	0.056	0.500	0.431	0.234
5	Naïve Bayes	0.508	0.630	0.702	0.588	0.605	0.563	0.605	0.566	0.597	0.210	0.218	0.214	0.370	0.412	0.298	0.492	0.155	0.639	0.048	0.500	0.403	0.210
6	K Nearest Neighbor	0.855	0.688	0.613	0.809	0.734	0.763	0.734	0.767	0.724	0.468	0.497	0.482	0.312	0.191	0.387	0.145	0.267	0.077	0.282	0.500	0.621	0.468
7	RProp MLP	0.782	0.729	0.710	0.765	0.746	0.755	0.746	0.755	0.745	0.492	0.495	0.493	0.271	0.235	0.290	0.218	0.374	0.162	0.263	0.500	0.536	0.492
8	SVM	0.008	0.500	0.992	0.500	0.500	0.016	0.500	0.063	0.089	0.001	0.001	0.001	0.500	0.500	0.008	0.992	0.001	1.000	0.001	0.500	0.008	0.001
9	Decision Tree	0.815	0.789	0.782	0.808	0.798	0.802	0.798	0.802	0.798	0.597	0.597	0.597	0.211	0.192	0.218	0.185	0.606	0.078	0.500	0.500	0.516	0.597
10	SimpleCart	0.774	0.780	0.782	0.776	0.778	0.777	0.778	0.777	0.778	0.556	0.556	0.556	0.220	0.224	0.218	0.226	0.565	0.140	0.383	0.500	0.496	0.556
11	Random Forest	0.847	0.847	0.847	0.847	0.847	0.847	0.847	0.847	0.847	0.694	0.694	0.694	0.153	0.153	0.153	0.153	1.000	0.010	1.000	0.500	0.500	0.694

APPENDIX B: *B* MATRICES

Since *B* matrices must be computed for each of the 22 evaluation measures and repeated for each of the 12 experimented data sets, our research resulted in computing 264 *B* matrices. For obvious reasons, we cannot provide all of them in this dissertation. However, in future, we will provide a permeant cloud-based repository location, where interested researchers can access our work for further scrutiny and replication.

APPENDIX C: $\boldsymbol{S}$ MATRICES

$S[c \times p]$ score matrix represents the classifiers' pair-wise comparisons with respect to every evaluation measure j for the 12 NASA software defect data sets. The measures are numbered from 1 ... 22 to manage the limited space in the tables. Every column in matrix S is a $\boldsymbol{s}^{(j)}$ vector that represents the classifiers' pair-wise comparisons with respect to a specific evaluation measure *(j)*.

Data Set: CM1

	s Vectors **Classifiers**	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.089	0.087	0.083	0.083	0.086	0.091	0.086	0.090	0.090	0.077	0.075	0.076	0.102	0.128	0.120	0.096	0.048	0.114	0.013	0.090	0.096	0.077
2	SOTA	0.110	0.089	0.079	0.101	0.094	0.102	0.094	0.101	0.098	0.100	0.103	0.101	0.096	0.047	0.135	0.027	0.055	0.024	0.053	0.090	0.115	0.100
3	Fuzzy Rule	0.114	0.101	0.093	0.106	0.103	0.110	0.103	0.109	0.108	0.125	0.125	0.125	0.060	0.024	0.084	0.014	0.111	0.002	0.206	0.094	0.110	0.126
4	Logistic Regression	0.098	0.092	0.087	0.090	0.092	0.098	0.092	0.097	0.097	0.095	0.091	0.093	0.086	0.095	0.105	0.068	0.067	0.073	0.026	0.090	0.099	0.094
5	Naïve Bayes	0.045	0.073	0.087	0.060	0.066	0.057	0.066	0.058	0.065	0.021	0.024	0.022	0.149	0.228	0.105	0.247	0.016	0.302	0.002	0.090	0.057	0.021
6	K Nearest Neighbor	0.114	0.082	0.062	0.104	0.088	0.099	0.088	0.099	0.088	0.082	0.099	0.090	0.118	0.031	0.195	0.013	0.032	0.009	0.062	0.092	0.131	0.083
7	RProp MLP	0.114	0.105	0.099	0.107	0.107	0.113	0.107	0.112	0.112	0.135	0.132	0.133	0.046	0.020	0.060	0.013	0.167	0.000	0.333	0.092	0.103	0.135
8	SVM	0.004	0.060	0.112	0.055	0.058	0.008	0.058	0.016	0.022	0.000	-0.001	-0.001	0.191	0.251	0.015	0.384	0.000	0.368	0.000	0.090	0.006	0.000
9	Decision Tree	0.098	0.106	0.104	0.093	0.101	0.105	0.101	0.104	0.105	0.118	0.114	0.116	0.042	0.082	0.045	0.068	0.194	0.059	0.075	0.090	0.086	0.117
10	SimpleCart	0.106	0.107	0.104	0.099	0.105	0.110	0.105	0.109	0.110	0.129	0.124	0.127	0.040	0.053	0.045	0.041	0.213	0.029	0.137	0.090	0.092	0.129
11	Random Forest	0.110	0.098	0.091	0.102	0.100	0.107	0.100	0.106	0.105	0.117	0.116	0.116	0.070	0.041	0.090	0.027	0.097	0.019	0.093	0.090	0.105	0.117

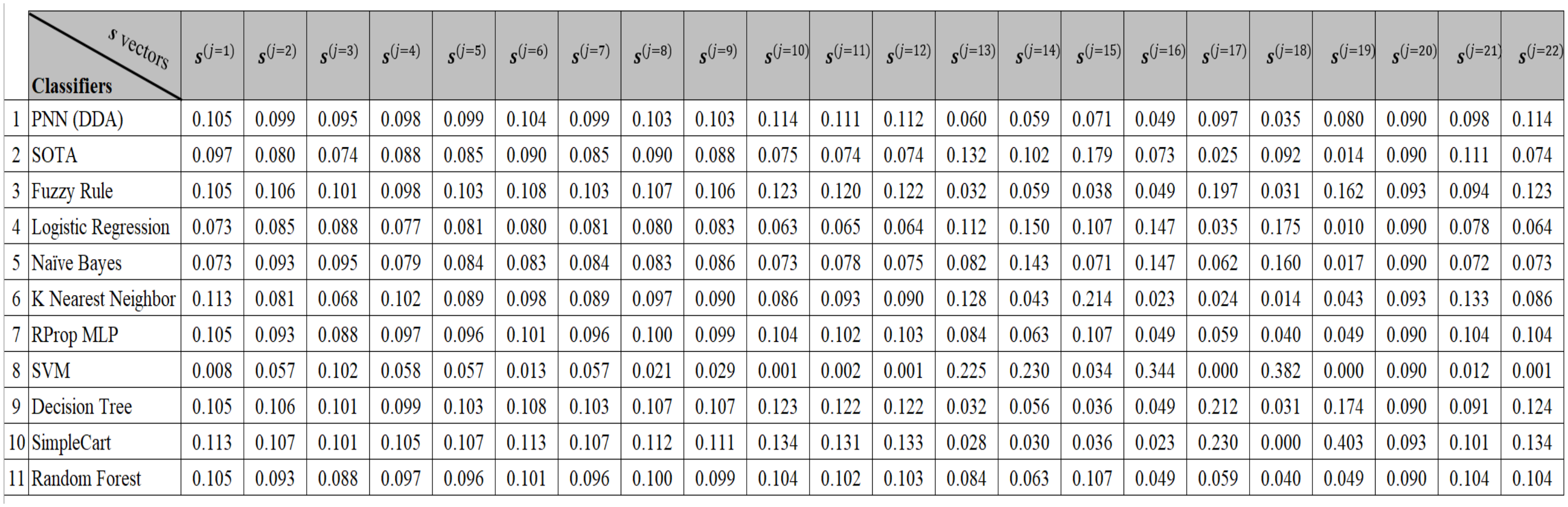
Data Set: JM1

	s vectors / Classifiers	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.122	0.074	0.037	0.105	0.079	0.096	0.079	0.098	0.073	0.051	0.087	0.067	0.130	0.054	0.217	0.017	0.015	0.020	0.043	0.090	0.149	0.051
2	SOTA	0.088	0.092	0.093	0.086	0.090	0.094	0.090	0.093	0.098	0.089	0.084	0.087	0.089	0.103	0.086	0.099	0.064	0.101	0.031	0.090	0.087	0.089
3	Fuzzy Rule	0.107	0.111	0.109	0.101	0.108	0.114	0.108	0.113	0.118	0.150	0.142	0.146	0.044	0.063	0.050	0.051	0.187	0.022	0.176	0.097	0.094	0.150
4	Logistic Regression	0.066	0.090	0.100	0.076	0.083	0.080	0.083	0.080	0.088	0.063	0.064	0.064	0.094	0.130	0.070	0.151	0.056	0.159	0.018	0.090	0.067	0.063
5	Naïve Bayes	0.099	0.068	0.039	0.070	0.069	0.084	0.069	0.085	0.068	0.015	0.018	0.017	0.144	0.146	0.211	0.073	0.004	0.205	0.003	0.090	0.132	0.015
6	K Nearest Neighbor	0.111	0.093	0.086	0.104	0.099	0.106	0.099	0.106	0.107	0.118	0.117	0.118	0.085	0.056	0.102	0.042	0.072	0.024	0.083	0.090	0.108	0.118
7	RProp MLP	0.084	0.086	0.087	0.082	0.085	0.089	0.085	0.088	0.093	0.071	0.068	0.070	0.102	0.115	0.101	0.108	0.044	0.125	0.020	0.090	0.088	0.071
8	SVM	0.000	0.065	0.130	0.062	0.065	0.000	0.065	0.004	0.006	0.000	0.000	0.000	0.151	0.167	0.000	0.309	0.000	0.275	0.000	0.090	0.000	0.000
9	Decision Tree	0.104	0.103	0.102	0.101	0.103	0.108	0.103	0.107	0.112	0.133	0.127	0.130	0.063	0.065	0.065	0.059	0.128	0.036	0.105	0.090	0.092	0.134
10	SimpleCart	0.105	0.104	0.104	0.101	0.104	0.109	0.104	0.108	0.114	0.138	0.131	0.134	0.060	0.063	0.061	0.057	0.141	0.033	0.118	0.090	0.091	0.138
11	Random Forest	0.114	0.114	0.114	0.111	0.114	0.119	0.114	0.118	0.124	0.172	0.163	0.168	0.037	0.038	0.037	0.034	0.288	0.000	0.403	0.090	0.091	0.172

Data Set: KC1

	Classifiers / s vectors	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.102	0.089	0.082	0.093	0.091	0.099	0.091	0.098	0.096	0.091	0.090	0.091	0.096	0.086	0.124	0.069	0.068	0.071	0.061	0.091	0.108	0.091
2	SOTA	0.111	0.099	0.092	0.101	0.101	0.108	0.101	0.107	0.106	0.123	0.122	0.122	0.069	0.065	0.086	0.054	0.133	0.027	0.152	0.091	0.104	0.123
3	Fuzzy Rule	0.108	0.095	0.086	0.097	0.096	0.105	0.096	0.104	0.101	0.108	0.107	0.107	0.080	0.077	0.110	0.058	0.092	0.043	0.098	0.094	0.110	0.108
4	Logistic Regression	0.086	0.088	0.086	0.084	0.086	0.090	0.086	0.089	0.090	0.074	0.074	0.074	0.100	0.107	0.110	0.100	0.063	0.123	0.039	0.091	0.092	0.074
5	Naïve Bayes	0.073	0.081	0.083	0.077	0.078	0.079	0.078	0.079	0.082	0.050	0.051	0.050	0.118	0.126	0.120	0.124	0.038	0.177	0.019	0.091	0.085	0.050
6	K Nearest Neighbor	0.106	0.090	0.082	0.095	0.093	0.101	0.093	0.100	0.098	0.096	0.095	0.096	0.094	0.081	0.124	0.063	0.072	0.058	0.071	0.091	0.110	0.096
7	RProp MLP	0.083	0.097	0.096	0.086	0.090	0.092	0.090	0.092	0.094	0.089	0.093	0.091	0.075	0.103	0.072	0.106	0.115	0.112	0.068	0.091	0.080	0.089
8	SVM	0.002	0.063	0.115	0.064	0.064	0.003	0.064	0.010	0.014	0.000	0.001	0.000	0.165	0.157	0.005	0.257	0.000	0.300	0.000	0.091	0.002	0.000
9	Decision Tree	0.107	0.097	0.091	0.099	0.098	0.106	0.098	0.105	0.104	0.116	0.114	0.115	0.074	0.072	0.091	0.060	0.119	0.040	0.122	0.091	0.103	0.116
10	SimpleCart	0.102	0.101	0.096	0.097	0.099	0.105	0.099	0.104	0.104	0.118	0.118	0.118	0.063	0.076	0.072	0.069	0.154	0.049	0.137	0.091	0.095	0.118
11	Random Forest	0.119	0.101	0.092	0.107	0.104	0.113	0.104	0.112	0.110	0.135	0.134	0.135	0.065	0.050	0.086	0.039	0.147	0.000	0.232	0.091	0.110	0.135

Data Set: KC3



	Classifiers \ s vectors	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.105	0.099	0.095	0.098	0.099	0.104	0.099	0.103	0.103	0.114	0.111	0.112	0.060	0.059	0.071	0.049	0.097	0.035	0.080	0.090	0.098	0.114
2	SOTA	0.097	0.080	0.074	0.088	0.085	0.090	0.085	0.090	0.088	0.075	0.074	0.074	0.132	0.102	0.179	0.073	0.025	0.092	0.014	0.090	0.111	0.074
3	Fuzzy Rule	0.105	0.106	0.101	0.098	0.103	0.108	0.103	0.107	0.106	0.123	0.120	0.122	0.032	0.059	0.038	0.049	0.197	0.031	0.162	0.093	0.094	0.123
4	Logistic Regression	0.073	0.085	0.088	0.077	0.081	0.080	0.081	0.080	0.083	0.063	0.065	0.064	0.112	0.150	0.107	0.147	0.035	0.175	0.010	0.090	0.078	0.064
5	Naïve Bayes	0.073	0.093	0.095	0.079	0.084	0.083	0.084	0.083	0.086	0.073	0.078	0.075	0.082	0.143	0.071	0.147	0.062	0.160	0.017	0.090	0.072	0.073
6	K Nearest Neighbor	0.113	0.081	0.068	0.102	0.089	0.098	0.089	0.097	0.090	0.086	0.093	0.090	0.128	0.043	0.214	0.023	0.024	0.014	0.043	0.093	0.133	0.086
7	RProp MLP	0.105	0.093	0.088	0.097	0.096	0.101	0.096	0.100	0.099	0.104	0.102	0.103	0.084	0.063	0.107	0.049	0.059	0.040	0.049	0.090	0.104	0.104
8	SVM	0.008	0.057	0.102	0.058	0.057	0.013	0.057	0.021	0.029	0.001	0.002	0.001	0.225	0.230	0.034	0.344	0.000	0.382	0.000	0.090	0.012	0.001
9	Decision Tree	0.105	0.106	0.101	0.099	0.103	0.108	0.103	0.107	0.107	0.123	0.122	0.122	0.032	0.056	0.036	0.049	0.212	0.031	0.174	0.090	0.091	0.124
10	SimpleCart	0.113	0.107	0.101	0.105	0.107	0.113	0.107	0.112	0.111	0.134	0.131	0.133	0.028	0.030	0.036	0.023	0.230	0.000	0.403	0.093	0.101	0.134
11	Random Forest	0.105	0.093	0.088	0.097	0.096	0.101	0.096	0.100	0.099	0.104	0.102	0.103	0.084	0.063	0.107	0.049	0.059	0.040	0.049	0.090	0.104	0.104

Data Set: MC1

	s Vectors / Classifiers	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.095	0.097	0.092	0.090	0.093	0.098	0.093	0.098	0.098	0.097	0.097	0.097	0.047	0.096	0.071	0.073	0.036	0.070	0.002	0.091	0.093	0.097
2	SOTA	0.103	0.098	0.093	0.097	0.098	0.103	0.098	0.103	0.102	0.107	0.106	0.107	0.036	0.049	0.058	0.035	0.048	0.032	0.006	0.091	0.099	0.107
3	Fuzzy Rule	0.107	0.102	0.097	0.101	0.102	0.107	0.102	0.107	0.107	0.116	0.116	0.116	0.008	0.019	0.013	0.014	0.231	0.010	0.073	0.091	0.100	0.116
4	Logistic Regression	0.075	0.084	0.083	0.076	0.079	0.082	0.079	0.081	0.083	0.064	0.065	0.065	0.139	0.202	0.195	0.165	0.009	0.181	0.000	0.091	0.085	0.064
5	Naïve Bayes	0.077	0.077	0.074	0.074	0.075	0.079	0.075	0.078	0.079	0.055	0.055	0.055	0.195	0.213	0.305	0.159	0.005	0.195	0.000	0.091	0.095	0.055
6	K Nearest Neighbor	0.109	0.098	0.092	0.103	0.100	0.106	0.100	0.106	0.105	0.113	0.113	0.113	0.041	0.004	0.071	0.003	0.041	0.000	0.067	0.091	0.107	0.113
7	RProp MLP	0.108	0.091	0.085	0.101	0.096	0.102	0.096	0.101	0.100	0.102	0.103	0.102	0.091	0.017	0.169	0.011	0.016	0.009	0.006	0.091	0.113	0.102
8	SVM	0.001	0.052	0.097	0.052	0.052	0.001	0.052	0.006	0.008	0.000	0.000	0.000	0.378	0.373	0.006	0.522	0.000	0.491	0.000	0.091	0.001	0.000
9	Decision Tree	0.108	0.101	0.095	0.101	0.101	0.107	0.101	0.106	0.106	0.115	0.114	0.115	0.019	0.015	0.032	0.011	0.093	0.008	0.037	0.091	0.102	0.115
10	SimpleCart	0.109	0.098	0.092	0.102	0.100	0.106	0.100	0.105	0.105	0.112	0.112	0.112	0.041	0.008	0.071	0.005	0.041	0.003	0.033	0.091	0.106	0.112
11	Random Forest	0.109	0.103	0.097	0.103	0.103	0.109	0.103	0.108	0.108	0.119	0.119	0.119	0.004	0.004	0.006	0.003	0.481	0.000	0.775	0.091	0.101	0.119

Data Set: MC2

	s vectors Classifiers	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.063	0.097	0.103	0.076	0.083	0.077	0.083	0.079	0.083	0.069	0.078	0.073	0.073	0.137	0.050	0.166	0.090	0.169	0.024	0.090	0.059	0.068
2	SOTA	0.078	0.076	0.074	0.075	0.076	0.079	0.076	0.078	0.078	0.046	0.045	0.045	0.137	0.142	0.149	0.124	0.020	0.180	0.007	0.090	0.095	0.046
3	Fuzzy Rule	0.104	0.101	0.098	0.100	0.101	0.105	0.101	0.104	0.104	0.122	0.119	0.120	0.061	0.063	0.066	0.055	0.120	0.028	0.097	0.090	0.095	0.122
4	Logistic Regression	0.110	0.085	0.074	0.100	0.091	0.098	0.091	0.098	0.092	0.091	0.095	0.093	0.109	0.063	0.149	0.041	0.040	0.028	0.043	0.090	0.118	0.091
5	Naïve Bayes	0.110	0.106	0.105	0.106	0.107	0.110	0.107	0.109	0.110	0.140	0.136	0.138	0.046	0.042	0.044	0.041	0.206	0.005	0.223	0.085	0.089	0.139
6	K Nearest Neighbor	0.111	0.097	0.088	0.103	0.099	0.106	0.099	0.105	0.102	0.117	0.117	0.117	0.073	0.054	0.099	0.037	0.077	0.008	0.093	0.095	0.111	0.117
7	RProp MLP	0.111	0.108	0.103	0.105	0.107	0.112	0.107	0.111	0.110	0.140	0.136	0.138	0.040	0.047	0.050	0.037	0.183	0.000	0.223	0.095	0.100	0.139
8	SVM	0.014	0.061	0.105	0.060	0.061	0.023	0.061	0.029	0.039	0.000	0.000	0.000	0.182	0.189	0.044	0.295	0.000	0.332	0.000	0.090	0.021	0.000
9	Decision Tree	0.078	0.067	0.059	0.068	0.068	0.074	0.068	0.073	0.070	0.023	0.023	0.023	0.162	0.162	0.199	0.124	0.007	0.237	0.003	0.090	0.106	0.023
10	SimpleCart	0.111	0.097	0.088	0.103	0.099	0.106	0.099	0.105	0.102	0.117	0.117	0.117	0.073	0.054	0.099	0.037	0.077	0.008	0.093	0.095	0.111	0.117
11	Random Forest	0.110	0.106	0.103	0.105	0.106	0.110	0.106	0.109	0.109	0.137	0.134	0.135	0.046	0.047	0.050	0.041	0.180	0.006	0.194	0.090	0.095	0.137

Data Set: MW1

	Classifiers \ s vectors	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.091	0.090	0.087	0.085	0.089	0.094	0.089	0.092	0.093	0.086	0.081	0.084	0.094	0.117	0.103	0.089	0.066	0.106	0.022	0.090	0.094	0.086
2	SOTA	0.112	0.085	0.071	0.104	0.091	0.100	0.091	0.099	0.093	0.092	0.101	0.097	0.109	0.035	0.155	0.018	0.048	0.010	0.080	0.090	0.121	0.093
3	Fuzzy Rule	0.112	0.099	0.091	0.105	0.102	0.108	0.102	0.107	0.106	0.122	0.121	0.121	0.067	0.029	0.090	0.017	0.106	0.002	0.189	0.094	0.109	0.122
4	Logistic Regression	0.081	0.101	0.103	0.082	0.092	0.093	0.092	0.092	0.096	0.094	0.091	0.093	0.059	0.132	0.052	0.125	0.143	0.128	0.035	0.090	0.074	0.094
5	Naïve Bayes	0.066	0.087	0.092	0.070	0.079	0.078	0.079	0.077	0.082	0.057	0.056	0.057	0.104	0.182	0.086	0.179	0.053	0.218	0.009	0.090	0.070	0.056
6	K Nearest Neighbor	0.112	0.084	0.065	0.103	0.089	0.099	0.089	0.099	0.090	0.085	0.099	0.092	0.113	0.038	0.172	0.017	0.040	0.011	0.069	0.092	0.126	0.086
7	RProp MLP	0.112	0.089	0.076	0.104	0.094	0.102	0.094	0.102	0.097	0.100	0.107	0.104	0.097	0.033	0.137	0.017	0.058	0.006	0.102	0.092	0.118	0.101
8	SVM	0.005	0.060	0.114	0.055	0.059	0.009	0.059	0.017	0.025	0.000	-0.002	-0.001	0.187	0.251	0.016	0.394	0.000	0.404	0.000	0.090	0.007	0.000
9	Decision Tree	0.096	0.104	0.103	0.092	0.100	0.103	0.100	0.102	0.105	0.116	0.110	0.113	0.051	0.085	0.052	0.072	0.177	0.064	0.076	0.090	0.086	0.116
10	SimpleCart	0.101	0.096	0.092	0.095	0.097	0.102	0.097	0.101	0.102	0.108	0.103	0.106	0.075	0.074	0.086	0.054	0.099	0.050	0.057	0.090	0.097	0.108
11	Random Forest	0.112	0.106	0.103	0.106	0.108	0.112	0.108	0.111	0.113	0.138	0.132	0.135	0.045	0.025	0.052	0.018	0.210	0.000	0.360	0.090	0.097	0.138

Data Set: PC1

	s Vectors / Classifiers	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.090	0.101	0.097	0.087	0.094	0.098	0.094	0.097	0.098	0.098	0.099	0.098	0.040	0.112	0.046	0.094	0.100	0.090	0.028	0.091	0.084	0.098
2	SOTA	0.104	0.102	0.097	0.098	0.101	0.106	0.101	0.105	0.105	0.115	0.113	0.114	0.035	0.048	0.046	0.036	0.117	0.030	0.085	0.091	0.096	0.115
3	Fuzzy Rule	0.102	0.107	0.102	0.097	0.102	0.108	0.102	0.107	0.107	0.119	0.117	0.118	0.009	0.058	0.012	0.043	0.456	0.035	0.276	0.094	0.093	0.118
4	Logistic Regression	0.097	0.084	0.078	0.090	0.087	0.093	0.087	0.092	0.091	0.081	0.081	0.081	0.125	0.099	0.186	0.065	0.021	0.077	0.008	0.091	0.109	0.081
5	Naïve Bayes	0.062	0.076	0.079	0.067	0.071	0.070	0.071	0.070	0.073	0.042	0.043	0.042	0.166	0.233	0.174	0.210	0.011	0.259	0.001	0.091	0.076	0.042
6	K Nearest Neighbor	0.111	0.090	0.083	0.104	0.096	0.103	0.096	0.102	0.100	0.104	0.107	0.106	0.094	0.012	0.151	0.007	0.033	0.001	0.121	0.091	0.117	0.104
7	RProp MLP	0.110	0.090	0.083	0.102	0.095	0.102	0.095	0.101	0.099	0.102	0.104	0.103	0.096	0.024	0.151	0.014	0.032	0.010	0.059	0.091	0.115	0.102
8	SVM	0.002	0.054	0.102	0.053	0.054	0.003	0.054	0.010	0.014	0.000	0.000	0.000	0.277	0.313	0.011	0.457	0.000	0.443	0.000	0.091	0.003	0.000
9	Decision Tree	0.104	0.096	0.091	0.098	0.097	0.103	0.097	0.102	0.102	0.106	0.105	0.106	0.066	0.051	0.093	0.036	0.054	0.032	0.040	0.091	0.102	0.106
10	SimpleCart	0.106	0.100	0.095	0.100	0.101	0.106	0.101	0.105	0.105	0.115	0.113	0.114	0.043	0.040	0.058	0.029	0.094	0.023	0.085	0.091	0.099	0.115
11	Random Forest	0.111	0.099	0.094	0.105	0.102	0.108	0.102	0.107	0.107	0.119	0.118	0.118	0.048	0.011	0.070	0.007	0.081	0.000	0.295	0.091	0.105	0.119

Data Set: PC2

	Classifiers \ s vectors	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.077	0.098	0.094	0.079	0.086	0.088	0.086	0.089	0.089	0.080	0.085	0.082	0.042	0.170	0.049	0.150	0.067	0.151	0.007	0.091	0.075	0.080
2	SOTA	0.109	0.093	0.088	0.102	0.098	0.104	0.098	0.103	0.102	0.107	0.106	0.107	0.075	0.021	0.131	0.013	0.034	0.008	0.042	0.091	0.111	0.107
3	Fuzzy Rule	0.109	0.102	0.097	0.102	0.103	0.109	0.103	0.108	0.107	0.119	0.117	0.118	0.010	0.018	0.016	0.013	0.306	0.006	0.375	0.090	0.101	0.119
4	Logistic Regression	0.094	0.084	0.079	0.087	0.086	0.092	0.086	0.091	0.090	0.080	0.078	0.079	0.138	0.116	0.230	0.078	0.014	0.091	0.003	0.091	0.105	0.080
5	Naïve Bayes	0.066	0.086	0.086	0.071	0.077	0.076	0.077	0.077	0.079	0.058	0.062	0.060	0.128	0.219	0.148	0.196	0.016	0.219	0.001	0.091	0.073	0.057
6	K Nearest Neighbor	0.111	0.092	0.086	0.103	0.098	0.104	0.098	0.103	0.102	0.107	0.107	0.107	0.081	0.011	0.148	0.006	0.030	0.000	0.075	0.092	0.115	0.107
7	RProp MLP	0.111	0.097	0.092	0.103	0.101	0.107	0.101	0.106	0.105	0.114	0.113	0.113	0.047	0.010	0.082	0.006	0.058	0.000	0.144	0.092	0.109	0.114
8	SVM	0.002	0.052	0.097	0.052	0.052	0.003	0.052	0.009	0.013	0.000	0.000	0.000	0.370	0.342	0.016	0.471	0.000	0.475	0.000	0.091	0.003	0.000
9	Decision Tree	0.105	0.102	0.097	0.098	0.101	0.106	0.101	0.106	0.105	0.114	0.113	0.113	0.011	0.044	0.016	0.033	0.293	0.026	0.144	0.090	0.096	0.114
10	SimpleCart	0.108	0.093	0.088	0.100	0.097	0.103	0.097	0.102	0.101	0.105	0.104	0.105	0.076	0.030	0.131	0.020	0.033	0.015	0.027	0.091	0.110	0.105
11	Random Forest	0.109	0.101	0.096	0.102	0.102	0.108	0.102	0.107	0.107	0.117	0.116	0.116	0.020	0.019	0.033	0.013	0.149	0.007	0.182	0.091	0.103	0.117

Data Set: PC3

	s vectors / Classifiers	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.091	0.095	0.092	0.086	0.092	0.098	0.092	0.096	0.097	0.093	0.090	0.092	0.076	0.109	0.087	0.090	0.090	0.095	0.023	0.091	0.090	0.093
2	SOTA	0.113	0.091	0.082	0.100	0.096	0.106	0.096	0.105	0.103	0.106	0.103	0.105	0.092	0.054	0.137	0.034	0.066	0.036	0.044	0.091	0.117	0.106
3	Fuzzy Rule	0.114	0.104	0.097	0.102	0.105	0.115	0.105	0.113	0.112	0.130	0.124	0.127	0.039	0.047	0.056	0.032	0.198	0.026	0.144	0.095	0.107	0.130
4	Logistic Regression	0.086	0.085	0.082	0.081	0.084	0.090	0.084	0.088	0.090	0.072	0.069	0.071	0.114	0.130	0.137	0.103	0.045	0.124	0.010	0.091	0.095	0.072
5	Naïve Bayes	0.030	0.093	0.102	0.063	0.069	0.046	0.069	0.054	0.058	0.029	0.057	0.041	0.082	0.198	0.031	0.248	0.079	0.248	0.007	0.091	0.030	0.029
6	K Nearest Neighbor	0.122	0.085	0.072	0.109	0.095	0.107	0.095	0.106	0.100	0.103	0.108	0.106	0.113	0.020	0.193	0.010	0.045	0.007	0.101	0.091	0.134	0.103
7	RProp MLP	0.110	0.091	0.084	0.098	0.096	0.105	0.096	0.104	0.102	0.105	0.101	0.103	0.091	0.063	0.130	0.041	0.068	0.044	0.038	0.091	0.113	0.105
8	SVM	0.001	0.057	0.107	0.057	0.058	0.003	0.058	0.009	0.013	0.000	0.000	0.000	0.222	0.221	0.006	0.320	0.000	0.308	0.000	0.091	0.002	0.000
9	Decision Tree	0.099	0.098	0.094	0.092	0.096	0.104	0.096	0.102	0.103	0.106	0.102	0.105	0.062	0.087	0.075	0.069	0.120	0.069	0.040	0.091	0.095	0.106
10	SimpleCart	0.109	0.100	0.094	0.099	0.101	0.110	0.101	0.108	0.108	0.118	0.113	0.116	0.057	0.061	0.075	0.045	0.134	0.042	0.069	0.091	0.102	0.118
11	Random Forest	0.124	0.101	0.094	0.112	0.108	0.118	0.108	0.116	0.115	0.137	0.133	0.135	0.051	0.011	0.075	0.007	0.155	0.000	0.522	0.091	0.115	0.137

Data Set: PC4

	s Vectors Classifiers	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.102	0.095	0.090	0.095	0.096	0.102	0.096	0.101	0.101	0.103	0.101	0.102	0.068	0.064	0.100	0.045	0.085	0.042	0.032	0.091	0.102	0.103
2	SOTA	0.104	0.091	0.085	0.096	0.094	0.100	0.094	0.100	0.099	0.098	0.097	0.098	0.089	0.062	0.138	0.041	0.059	0.040	0.024	0.091	0.107	0.098
3	Fuzzy Rule	0.109	0.101	0.096	0.102	0.102	0.109	0.102	0.108	0.107	0.118	0.117	0.118	0.033	0.026	0.050	0.017	0.193	0.010	0.188	0.092	0.104	0.119
4	Logistic Regression	0.087	0.091	0.087	0.083	0.087	0.092	0.087	0.091	0.091	0.082	0.081	0.081	0.093	0.137	0.123	0.106	0.055	0.117	0.009	0.091	0.091	0.082
5	Naïve Bayes	0.054	0.086	0.090	0.067	0.073	0.068	0.073	0.070	0.073	0.047	0.056	0.051	0.116	0.237	0.100	0.236	0.039	0.263	0.003	0.091	0.060	0.047
6	K Nearest Neighbor	0.112	0.090	0.083	0.104	0.096	0.103	0.096	0.103	0.101	0.104	0.106	0.105	0.095	0.014	0.161	0.008	0.053	0.001	0.111	0.091	0.117	0.104
7	RProp MLP	0.112	0.098	0.092	0.104	0.101	0.108	0.101	0.107	0.106	0.116	0.115	0.116	0.054	0.012	0.084	0.008	0.113	0.000	0.235	0.091	0.108	0.116
8	SVM	0.001	0.054	0.101	0.053	0.054	0.002	0.054	0.008	0.011	0.000	0.000	0.000	0.293	0.319	0.008	0.449	0.000	0.451	0.000	0.091	0.002	0.000
9	Decision Tree	0.100	0.094	0.089	0.094	0.094	0.100	0.094	0.100	0.099	0.099	0.098	0.099	0.073	0.075	0.107	0.053	0.076	0.053	0.024	0.091	0.101	0.099
10	SimpleCart	0.108	0.097	0.092	0.100	0.099	0.106	0.099	0.105	0.104	0.111	0.110	0.111	0.056	0.036	0.084	0.024	0.109	0.018	0.075	0.091	0.105	0.111
11	Random Forest	0.111	0.102	0.096	0.103	0.103	0.110	0.103	0.109	0.108	0.121	0.119	0.120	0.031	0.018	0.046	0.012	0.216	0.004	0.299	0.091	0.103	0.121

Data Set: PC5

	s vectors Classifiers	$s^{(j=1)}$	$s^{(j=2)}$	$s^{(j=3)}$	$s^{(j=4)}$	$s^{(j=5)}$	$s^{(j=6)}$	$s^{(j=7)}$	$s^{(j=8)}$	$s^{(j=9)}$	$s^{(j=10)}$	$s^{(j=11)}$	$s^{(j=12)}$	$s^{(j=13)}$	$s^{(j=14)}$	$s^{(j=15)}$	$s^{(j=16)}$	$s^{(j=17)}$	$s^{(j=18)}$	$s^{(j=19)}$	$s^{(j=20)}$	$s^{(j=21)}$	$s^{(j=22)}$
1	PNN (DDA)	0.102	0.093	0.087	0.094	0.094	0.101	0.094	0.100	0.100	0.102	0.101	0.101	0.085	0.082	0.103	0.066	0.088	0.058	0.070	0.090	0.102	0.102
2	SOTA	0.094	0.091	0.087	0.088	0.090	0.096	0.090	0.095	0.095	0.088	0.087	0.088	0.090	0.098	0.103	0.085	0.076	0.093	0.048	0.090	0.096	0.088
3	Fuzzy Rule	0.117	0.100	0.087	0.105	0.101	0.112	0.101	0.111	0.107	0.125	0.128	0.126	0.069	0.055	0.103	0.035	0.108	0.000	0.163	0.096	0.117	0.126
4	Logistic Regression	0.073	0.081	0.083	0.075	0.078	0.080	0.078	0.079	0.082	0.049	0.050	0.049	0.115	0.132	0.115	0.128	0.038	0.185	0.016	0.090	0.084	0.049
5	Naïve Bayes	0.068	0.081	0.085	0.074	0.077	0.076	0.077	0.076	0.080	0.044	0.045	0.045	0.116	0.137	0.109	0.140	0.036	0.201	0.014	0.090	0.079	0.044
6	K Nearest Neighbor	0.114	0.088	0.074	0.101	0.093	0.103	0.093	0.103	0.097	0.099	0.103	0.101	0.098	0.064	0.142	0.041	0.062	0.024	0.080	0.090	0.121	0.098
7	RProp MLP	0.105	0.093	0.086	0.096	0.095	0.102	0.095	0.101	0.100	0.104	0.103	0.103	0.085	0.078	0.106	0.062	0.087	0.051	0.075	0.090	0.105	0.104
8	SVM	0.001	0.064	0.120	0.063	0.064	0.002	0.064	0.008	0.012	0.000	0.000	0.000	0.157	0.166	0.003	0.282	0.000	0.315	0.000	0.090	0.002	0.000
9	Decision Tree	0.109	0.101	0.095	0.101	0.101	0.108	0.101	0.108	0.107	0.126	0.124	0.125	0.066	0.064	0.080	0.053	0.141	0.025	0.142	0.090	0.101	0.126
10	SimpleCart	0.103	0.100	0.095	0.097	0.099	0.105	0.099	0.104	0.105	0.117	0.116	0.116	0.069	0.074	0.080	0.064	0.131	0.044	0.109	0.090	0.097	0.117
11	Random Forest	0.113	0.108	0.102	0.106	0.108	0.115	0.108	0.114	0.114	0.146	0.144	0.145	0.048	0.051	0.056	0.044	0.232	0.003	0.284	0.090	0.098	0.146

REFERENCES

[1] B. Turhan, A. Tosun, and A. Bener, "Empirical evaluation of mixed-project defect prediction models," in *37th EUROMICRO Conference on Software Engineering and Advanced Application*, Oulu, Finland, 2011, pp. 396-403.

[2] D. Lo, SC. Khoo, J. Han, and C. Liu, *Mining Software specifications: methodologies and applications*, Boca Raton, FL, USA: Chapman and Hall/CRC Press, 2011, pp 1-15.

[3] M. Shepperd, D. Bowes, and T. Hall, "Researcher bias: The use of machine learning in Software defect prediction," *IEEE Transactions on Software Engineering*, vol. 40, no. 6, pp. 603-616, 2014.

[4] S. Lessmann, S. Member, B. Baesens, C. Mues, 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.

[5] L. Madeyski, and M. Jureczko, "Which process metrics can significantly improve defect prediction models? An empirical study," *Software Quality Journal*, vol. 23, no. 3, pp. 393-422, 2014.

[6] M. D’Ambros, M. Lanza, and R. Robbes, "Evaluating defect prediction approaches: a benchmark and an extensive comparison," *Empirical Software Engineering*, vol. 17, no. 4-5, pp. 531-577, 2011.

[7] R. S. Wahono, N. S. Herman, and S. Ahmad, "A comparison framework of classification models for software defect prediction," *Advanced Scientific Letters*, vol. 20, no. 10-11, pp. 1945-1950, 2014.

[8] I. Myrtveit, E. Stensrud, and M. Shepperd, "Reliability and validity in comparative studies of software prediction models," *IEEE Transactions on Software Engineering*, vol. 31, no. 5, pp. 380-391, 2005.

[9] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell, "A systematic literature review on fault prediction performance in software engineering," *IEEE Transactions on Software Engineering*, vol. 38, no. 6, pp. 1276-1304, 2012.

[10] H. Wang, T. M. Khoshgoftaar, and Q. Liang, "A study of software metric selection techniques: stability analysis and defect prediction model performance," *International Journal on Artificial Intelligence Tools*, vol. 22, no. 05, pp. 1360010, 2013.

[11] I. Myrtveit and E. Stensrud, "Validity and reliability of evaluation procedures in comparative studies of effort prediction models," *Empirical Software Engineering*, vol. 17, no. 1-2, pp. 23-33, 2012.

[12] D. M. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37-63, 2011.

[13] A. Vesra, " A study of various static and dynamic metrics for open source software," International Journal of Computer Applications, vol. 122, no. 10, 2015.

[14] S. Wang and X. Yao, "Using class imbalance learning for software defect prediction," *IEEE Transactions on Reliability*, vol. 62, no. 2, pp. 434-443, 2013.

[15] M. Shepperd, Q. Song, Z. Sun, and C. Mair. *NASA MDP Software Defects Data Sets*. [Online]. Available: figshare.com.

[16] B. Clark, and D. Zubrow, "How good is the software: a review of defect prediction techniques," *Sponsored by the US department of Defense*, Carnegie Mellon University, 2001.

[17] N. Fenton, and M. Neil, "A critique of software defect prediction models," *IEEE Transactions on Software Engineering*, vol. 25, no. 5, pp. 675-689, 1999.

[18] N. Fenton and N. Ohlsson, "Quantitative analysis of faults and failures in a complex software system," *IEEE Transactions on Software Engineering*, vol. 26, no. 8, pp. 797-814, 2000.

[19] C. R. Pandian, *Software metrics: A guide to planning, analysis, and application*. 2003, Boca Raton, FL, USA: Chapman and Hall/CRC Press, 2003.

[20] J. Radatz, A. Geraci, and F. Katki, *IEEE standard glossary of software engineering terminology*, IEEE Std, 1990, 610.12-1990.

[21] N. Fenton and J. Bieman, *Software metrics: a rigorous and practical approach*, 3rd ed., Boca Raton, FL, USA: Chapman and Hall/CRC Press, 2014.

[22] N. Fenton and B. Kitchenham, "Validating software measures," *Software Testing, Verification and Reliability*, vol. 1, no. 2, pp. 27-42, 1991.

[23] C. Andersson, "A replicated empirical study of a selection method for software reliability growth models," *Empirical Software Engineering*, vol. 12, no. 2, pp. 161, 2007.

[24] L. J. White, "The importance of empirical work for software engineering papers," *Software Testing, Verification and Reliability*, vol. 12, no. 4, pp. 195-196, 2002.

[25] G. Forman and M. Scholz, "Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement," *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 1, pp. 49-57, 2010.

[26] M. Lanza, and R. Marinescu, *Object-oriented metrics in practice: using software metrics to characterize, evaluate, and improve the design of object-oriented systems*, 2007: Springer Science & Business Media.

[27] D. C. Ince, L. Hatton, and J. Graham-Cumming, "The case for open computer programs," *Nature*, vol. 482, no. 7386, pp. 485, 2012.

[28] B. Kitchenham, "What’s up with software metrics?–A preliminary mapping study," *Journal of systems and software*, vol. 83, no. 1, pp. 37-51, 2010.

[29] A. Oram, and G. Wilson, *Making software: What really works, and why we believe it*, Sebastopol, CA: O'Reilly Media, Inc., 2010.

[30] D. H. Bowes, "Factors Affecting the Performance of Trainable Models for Software Defect Prediction", in *School of Computer Sciences*, University of Hertfordshire, 2013.

[31] T. M. Khoshgoftaar, K. Gao, A. Napolitano, and R. Wald, "A comparative study of iterative and non-iterative feature selection techniques for software defect prediction," *Information Systems Frontiers*, vol. 16, no. 5, pp. 801-822, 2014.

[32] M. R. Berthold, "A probabilistic extension for the DDA algorithm," in *IEEE International Conference on Neural Networks*, Washington, DC, USA, 1996, vol. 1, pp. 341-346.

[33] J. Herrero, A. Valencia, and J. Dopazo, "A hierarchical unsupervised growing neural network for clustering gene expression patterns," *Bioinformatics*, vol. 17, no. 2, pp. 126-136, 2001.

[34] H. Enderton and H. B. Enderton, *A mathematical introduction to logic*, San Diego, CA : Academic Press, 2001.

[35] D. R. Cox, "The regression analysis of binary sequences," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 20, no. 2, pp. 215-232, 1958.

[36] R. Stuart and N. Peter, "Artificial intelligence: a modern approach," Upper Saddle River, NJ, USA: Prentice Hall, 2003.

[37] B. V. Dasarathy, "Nearest neighbor ({NN}) norms:{NN} pattern classification techniques," *IEEE computer society press*, 1991.

[38] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *IEEE International Conference on Neural Networks,* San Francisco, CA, USA, 1993, pp. 586-591.

[39] J. C. Platt, "12 fast training of support vector machines using sequential minimal optimization," *Advances in kernel methods*, pp. 185-208, 1999.

[40] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, "Improvements to Platt's SMO algorithm for SVM classifier design," *Neural computation*, vol. 13, no. 3, pp. 637-649, 2001.

[41] J. R. Quinlan, *C4.5: programs for machine learning*, San Mateo, CA: Morgan Kaufmann Publishers, 2014.

[42] J. Shafer, R. Agrawal, and M. Mehta, "SPRINT: A scalable parallel classifer for data mining," in *Proceedings International Conference of Very Large Data Bases*, Bombay, India, 1996, pp. 544-555.

[43] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "*Classification and Regression Trees*," The Wadsworth Statistics and Probability Series, Belmont, CA: Wadsworth, pp. 356, 1984.

[44] W. Y. Loh, "Classification and regression trees," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 14-23, 2011.

[45] Y. Peng, G. Kou, G. Wang, W. Wu, and Y. Shi, "Ensemble of software defect predictors: an AHP-based evaluation method," *International Journal of Information Technology & Decision Making*, vol. 10, no. 01, pp. 187-206, 2011.

[46] Y. Jiang, B. Cukic, and Y. Ma, "Techniques for evaluating fault prediction models," *Empirical Software* Engineering, vol. 13, no. 5, pp. 561-595, 2008.

[47] M. Vihinen, "How to evaluate performance of prediction methods? Measures and their interpretation in variation effect analysis," *BMC genomics*, vol. 13, no. 4, pp. S2, 2012. DOI. 10.1186/1471-2164-13-S4-S2.

[48] C. Ferri, J. Hernández-Orallo, and R. Modroiu, "An experimental comparison of performance measures for classification," *Pattern Recognition Letters*, vol. 30, no. 1, pp. 27-38, 2009.

[49] M. W. Evans and J. J. Marciniak, *Software quality assurance and management*, New York, NY: Wiley, 1987.

[50] A. Abran, *Software metrics and software metrology*, John Wiley & Sons, 2010.

[51] A. Abran, *Software metrics need to mature into software metrology (recommendations)* in *NIST Workshop on Advancing Measurements and Testing for Information Technology (IT), Maryland, USA*, Oct 26-27, 1998.

[52] N. Fenton, "Software measurement: A necessary scientific basis," I*EEE Transactions on software engineering*, vol. 20, no. 3, pp. 199-206, 1994.

[53] H. Zuse, *Software complexity*. NY, USA: Walter de Cruyter, 1991.

[54] F. S. Roberts, *Measurement theory*. Cambridge University Press, 1985.

[55] L. Finkelstein, and M. Leaning, "A review of the fundamental concepts of measurement," *Measurement*, vol. 2, no. 1, pp. 25-34, 1984.

[56] B. Daneshvar Rouyendegh, "The DEA and intuitionistic fuzzy TOPSIS approach to departments' performances: a pilot study," *Journal of Applied Mathematics*, vol. 2011, 2011. DOI:10.1155/2011/712194.

[57] T. L. Saaty, "Decision making with the analytic hierarchy process," *International journal of services sciences*, vol. 1, no. 1, pp. 83-98, 2008.

[58] T. L. Saaty, *The analytical hierarchy process*. New York, USA: McGraw-Hill, 1980.

[59] S. Kubler, J. Robert, W. Derigent, A. Voisin, and Y. Le Traon, "A state-of the-art survey & testbed of fuzzy AHP (FAHP) applications," *Expert Systems with Applications,* vol. 65, pp. 398-422, 2016.

[60] B. D. Rouyendegh and T. Erkart, "Selection Of Academic Staff Using The Fuzzy Analytic Hierarchy Process(FAHP): A Pilot Study," *Tehnicki vjesnik,* vol. 19, no. 4, pp. 923-929, 2012.

[61] M. Z. Naghadehi, R. Mikaeil, and M. Ataei, "The application of fuzzy analytic hierarchy process (FAHP) approach to selection of optimum underground mining method for Jajarm Bauxite Mine, Iran," *Expert Systems with Applications,* vol. 36, no. 4, pp. 8218-8226, 2009.

[62] L. A. Zadeh, "Fuzzy sets," *Information and control,* vol. 8, no. 3, pp. 338-353, 1965.

[63] G. Kabir and M. A. A. Hasin, "Comparative analysis of AHP and fuzzy AHP models for multicriteria inventory classification," *International Journal of Fuzzy Logic Systems,* vol. 1, no. 1, pp. 1-16, 2011.

[64] P. J. Van Laarhoven and W. Pedrycz, "A fuzzy extension of Saaty's priority theory," *Fuzzy sets and Systems,* vol. 11, no. 1-3, pp. 229-241, 1983.

[65] J. Harding, E. A. Walker, and C. L. Walker, *The Truth Value Algebra of Type-2 Fuzzy Sets: Order Convolutions of Functions on the Unit Interval*. Boca Raton, FL, USA: Chapman and Hall/CRC Press, 2016.

[66] N. Nenkov and I. Ibryam, "A survey of the open source platforms Rapidminer and Konstanz Information Miner (KNIME) for data processing, analysis and

mining," *Proceedings of Pedagogical College, Dobrich,* Bulgaria, 2013, pp. 124-129.

[67] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, K. Thiel, B. Wiswedel, "KNIME-the Konstanz information miner: version 2.0 and beyond," *ACM SIGKDD explorations Newsletter,* vol. 11, no. 1, pp. 26-31, 2009.

[68] D. Rodriguez, I. Herraiz, R. Harrison, J. Dolado, and J. C. Riquelme, "Preliminary comparison of techniques for dealing with imbalance in software defect prediction," in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, London, United Kingdom, 2014, pp. 43.

[69] M. Shepperd, C. Qinbao Song, C. Zhongbin Sun, and C. Mair, "Data Quality: Some Comments on the NASA Software Defect Datasets," *IEEE Transactions on Software Engineering,* vol. 39, no. 9, pp. 1208-1215, 2013. DOI: 10.1109/TSE.2013.11.

[70] W. Fan, F. Geerts, X. Jia, "A Revival of Integrity Constraints for Data Cleaning", *Proceedings VLDB Endowment*, Auckland, New Zealand, 2008, vol. 1, no. 2, pp. 1522-1523.

[71] J. J. Buckley, "Fuzzy hierarchical analysis," *Fuzzy sets and systems,* vol. 17, no. 3, pp. 233-247, 1985.

[72] B. Schott and T. Whalen, "Nonmonotonicity and discretization error in fuzzy rule-based control using COA and MOM defuzzification," in *Proceedings of IEEE 5th International Fuzzy Systems,* New Orleans, LA, USA, 1996, vol. 1, pp. 450-456.

[73] Y. Kastro and A. B. Bener, "A defect prediction method for software versioning," *Software Quality Journal,* vol. 16, no. 4, pp. 543-562, 2008.

[74] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *IEEE Transactions on Software Engineering*, no. 1, pp. 2-13, 2007.

[75] T. M. Khoshgoftaar, N. Seliya, and N. Sundaresh, "An empirical study of predicting software faults with case-based reasoning," *Software Quality Journal*, vol. 14, no. 2, pp. 85-111, 2006.

[76] C. Catal and B. Diri, "Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem," *Information Sciences*, vol. 179, no. 8, pp. 1040-1058, 2009.