IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-5, Issue-12 E-ISSN: 2347-2693

An Evidential approach on Feature Subset Selection in Software Defect Prediction

M. Jaikumar[1*] and V. Kathiresan[2]

[1*]Department of Computer Applications, Sri Ramakrishna Mission Vidyalaya College of Arts and Science, Coimbatore, India
[2]Department of Computer Applications, Dr.SNS Rajalakshmi College of Arts and Science, Coimbatore, India

[*]*Corresponding Author: jai2911@gmail.com*

Available online at: www.ijcseonline.org

Received: 09/Nov/2017, Revised: 22/Nov/2017, Accepted: 17/Dec/2017, Published: 31/Dec/2017

Abstract-In software quality research, software defect is a key topic. The characteristic of software attributes influences the performance and effectiveness of the defect prediction model. However this issue is not well explored to the best of our knowledge. So this paper focus on the problem of attribute selection in the context of software defect prediction, we propose a Dempster-Shafer Theory technique with modified combination rule known as Dubois And Prade’s Disjunctive Consensus Rule is adapted for selecting best set of attributes to improve the accuracy of the software defect prediction. Dempster-Shafer Theory (DST) offers an alternative to traditional probabilistic theory for the mathematical representation of uncertainty. The proposed method is evaluated using the data sets from NASA metric data repository.

Keywords- Software Defect, prediction, dempster shafer theory, probability, evidence, reliability

I. Introduction

A software defect is an error, flaw, failure, or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways. Most defects arise from mistakes and errors made by people in either a program's source code or its design, or in frameworks and operating systems used by such programs, and a few are caused by compilers producing incorrect code.

Software defect prediction aims to determine whether a software module is defect-prone by constructing prediction models. The performance of such models is susceptible to the high dimensionality of the datasets that may include irrelevant and redundant features. Feature selection is applied to alleviate this issue. Because many feature selection methods have been proposed, there is an imperative need to analyze and compare these methods in case of uncertainty in selection of attributes. Prior empirical studies may have potential controversies and limitations, such as the contradictory results, usage of private datasets and inappropriate statistical test techniques. This observation leads us to conduct a careful empirical study to reinforce the confidence of the experimental.

II. Related Work

A survey is conducted to help developers identify defects based on existing software metrics using data mining techniques especially Classification and there by improve software quality which leads to reduction in the software development cost in the development and maintenance phase. Some of the existing works are explained in this section.

Yuan Chen, et.al[3] have surveyed the different data mining classification techniques for software defect prediction. They proposed a new model based on Bayesian network and PRM to predict the software defect and manage. Hassan Najadat and IzzatAlsmadi[4]Proposed a new model based on Ridor algorithm to predict fault in modules. They also tested the different classification techniques on the data sets provided by NASA. The result shows that Ridor algorithm is better than the existing technique in terms of accuracy and extraction of number of rules. Ahmet Okutan,OlcayTanerYıldız [5], introduced a new two metrics NOD, for the number of developers and LOCQ for source code quality apart from the metrics which is available in Promise data repository. Using Bayesian network classifier experimental shows that NOC &DIT have very limited and untrustworthy. LOCQ is more effective like CBO & WMC. NOD metric showed that there is a positive correlation between the no of developers and extent of defect prunes. LOC is proved to be one of the best metric for quick defect prediction. LCOM3 & LCOM have less effective compared to LOC,CBO,RFC, and LOCQ&WMC. Thair Nu Phyu [6] reviewed on various classification techniques such as decision tree induction, Bayesian networks, k-nearest neighbor classifier, case-based reasoning, genetic algorithm and fuzzy logic techniques. The results found that there is no proper info that which is the best classifier. Several of the classification methods produce a set of interacting loci that best predict the phenotype. However, a straightforward application of classification methods to large numbers of markers has a potential risk picking up randomly associated markers.

K.Sankar et.al [7], proposed a system which overcomes the problem of insufficiency in accuracy and use of large number of features. This paper proposed Feature selection techniques to predict faults in software code and it also measure the software code and performance of Naive based and SVM classifier. The accuracy is measured by F-mean metric the best defect predictor when compared to the others.

Yan ma et.al [8] proposed a model based on random forests. This is applied on five case studies based on NASA data sets. The results found was better than the result obtained by logistic regression, discriminate analysis and the algorithms in two machine learning software packages . Instead of generating one decision tree, this methodology generates hundreds or even thousands of trees using subsets of the training data. Hence classification accuracy of random forests is more significant over other methods in larger data sets

C.Chung and S.Dhall [9] proposed a various classification methods to predict software defect. Here Three types of classifier such as J48, Random Forest and Naive Bayesian Classifier is applied on various real time data sets of NASA to evaluate the data sets based on different criteria like ROC, Precision, MAE, RAE etc

Sonali Agarwal and DivyaTomar[10] have proposed a feature selection based Linear Twin Support Vector Machine (LSTSVM) model to predict defect prone software modules. F-score technique is used for software defect prediction based on various software metrics. This model is applied on PROMISE data sets and compared with the other existing models. The results say that the performance of the new model is better than the existing machine learning models

Thair Nu Phyu [6] reviewed on various classification techniques such as decision tree induction, Bayesian networks, k-nearest neighbor classifier, case-based reasoning, genetic algorithm and fuzzy logic techniques. The results found that there is no proper info that which is the best classifier. Several of the classification methods produce a set of interacting loci that best predict the phenotype. However, a straightforward application of classification methods to large numbers of markers has a potential risk picking up randomly associated markers. Ching-PaoChanget al.[11] proposed approach, Action-Based Defect Prediction (ABDP),which uses the classification with decision tree technique to build a prediction model, and performs association rule mining on the records of actions and defects. The association rule mining finds the maximum rule set with specific minimum support and confidence and thus the discovered knowledge can be utilized to interpret the prediction models and software process behaviors. It is used to discover defect patterns, and multi interval discretization to handle the continuous attributes of actions.

Ming Li, et al. proposed [12] a sample based methods for software defect prediction. Three methods such as random sampling with conventional machine learners, random sampling with a semi-supervised learner and active sampling with active semi-supervised learner. They applied a semi-supervised learning method called AcoForest to build a classification model based on a sample and the remaining un-sampled modules they also proposed a novel active semi supervised method called AcoForest which can select unsampled modules and experimented on Promise data sets and found to be the best method. Experimental results show that size does not affect the defect prediction

Dhiman,et al. [13] proposed a model where in it will categorize the software defects using some clustering approach and then the software defects are measured in each clustered separately. This system will analyze the software defect and its integration with software module.

Xiao-YuanFing,et.al [14] have tried to model the effective , efficient and low computational burden using advanced machine learning technique such as collaborative representative classification. The new model proposed by them is CSDP which is used to predict defect in a very efficient manner. KehanGao&Taghi M [15] experimented on promise repository based on criteria 1) Feature selection based on sampled data, and modeling based on original data, 2) feature selection based on sampled data and modeling based on sampled data and 3) feature selection based on sampled data, and modeling based sample data. The experimental results showed that the 1st criteria is the best compared to the others in defect prediction.

III. Components in Feature Selection

Feature selection algorithms to investigate during the splits of features and attempt to locate the finest in the midst of the opposing 2N applicant subsets according to a few assessment function. On the other hand this method is complete and it may be too expensive yet for a intermediate sized feature set(N). Other techniques based on heuristic or accidental search methods effort to decrease computational difficulty by negotiation performance [16]. There are four fundamental steps in a characteristic feature assortment method [16,17] such as subset generation, Feature evaluation function, stopping criterion and a validation procedure as depicted in Figure 1.

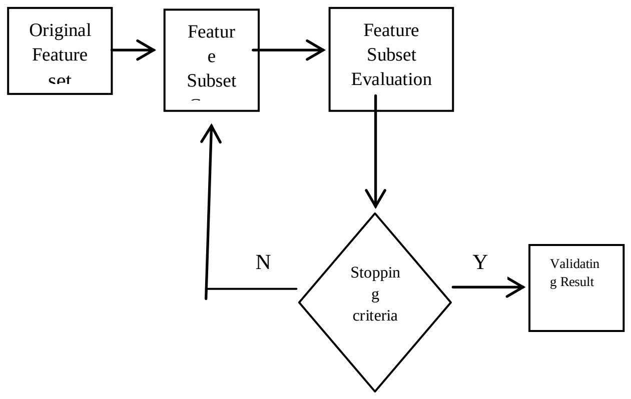

Figure 1: General Feature Selection process

Feature Selection Technique

A. DEMPSTER-SHAFER THEORY

Dempster-Shafer Theory (DST) is a mathematical theory of evidence. The seminal work on the subject is [21], which is an expansion of [20]. In a finite discrete space, Dempster-Shafer theory can be interpreted as a generalization of probability theory where probabilities are assigned to sets as opposed to mutually exclusive singletons. In traditional probability theory, evidence is associated with only one possible event. In DST, evidence can be associated with multiple possible events, e.g., sets of events. As a result, evidence in DST can be meaningful at a higher level of abstraction without having to resort to assumptions about the events within the evidential set. Where the evidence is sufficient enough to permit the assignment of probabilities to single events, the Dempster-Shafer model collapses to the traditional probabilistic formulation. One of the most important features of Dempster-Shafer theory is that the model is designed to cope with varying levels of precision regarding the information and no further assumptions are needed to represent the information. It also allows for the direct representation of uncertainty of system responses where an imprecise input can be characterized by a set or an interval and the resulting output is a set or an interval. There are three important functions in Dempster-Shafer theory: the basic probability assignment function (bpa or m), the Belief function (Bel), and the Plausibility function (Pl). The basic probability assignment (bpa) is a primitive of evidence theory. Generally speaking, the term "basic probability assignment" does not refer to probability in the classical sense. The bpa, represented by m, defines a mapping of the power set to the interval between 0 and 1, where the bpa of the null set is 0 and the summation of the bpa's of all the subsets of the power set is 1. The value of the bpa for a given set A (represented as m(A)), expresses the proportion of all relevant and available evidence that supports the claim that a particular element of X (the universal set) belongs to the set A but to no particular subset of A [22]. The value of m(A) pertains only to the set A and makes no additional claims about any subsets of A. Any further evidence on the subsets of A would be represented by another bpa, i.e. B Ì A, m(B) would the bpa for the subset B. Formally, this description of m can be represented with the following three equations:

B. Proposed Supervised Mutual Information (DEMPSTER SHAFER)with modified combination rule

In supervised MI the features gives a higher classification performance. Individual best features unless they are combined they will give a good classification performance. All the features having high of relevancy with each classes. If the value of the features are inappropriately combined then the redundancy between the features will be inappropriate. Based on the above observations, a new algorithm is proposed to determine the redundancy between the selected features and the candidate features and the relevancy of each attribute is measure using the dempster's belief and plausibility functions.

C. The Frame of Discernment (Θ)

The Hypothesis space is describing all the sets in the complete (exhaustive) space. Θ symbol denotes the frame and the elements are mutually exclusive. If the number of the elements in the set is n, then the power set (set of all subsets of (Θ) will have 2n elements.

D. BPA (Basic Probability Assignment)

In BPA the theory of evidence assigns a Belief mass. Each Belief mass is assigned to a subset. All the subset is called power set. 0and 1 are the positive no given for the sets. In this form the probability value exists.

If Θ is the frame of discernment, then a function

$$m: 2\Theta \rightarrow [0, 1] \text{ is called a bpa, whenever} \quad (1)$$

$$m(\emptyset) = 0 \text{ and} \quad (2)$$

$$\Sigma\, m(A) = 1 \text{ and} \quad (3)$$

$$A \subseteq \Theta \quad (4)$$

Belief (Bel)

Given a frame of discernment Θ and a body of empirical evidence {m(B1), m(B2), m(B3)….}, the belief committed to A ε Θ is

$$\text{Bel}(A) = \Sigma\, m(Bi) \quad (5)$$

$$B \subseteq A \quad (6)$$

$$\text{Also, Bel}(\Theta) = 1 \quad (7)$$

Plausibility Function (Pl)

The plausibility is the sum of all the masses values. The sum of all mass value is Set B.

Set B intersect the set A

$$\text{Pl}(A) = \Sigma\, m(Bi),\ B \mid B \cap A \neq \emptyset \quad (8)$$

Belief Range

The interval [Bel (A), Pl(A)] is called belief range.

Plausibility (Pl) and Belief (Bel) are related as follows

$$\text{Pl}(A) = 1 - \text{Bel}(\bar{A}) \quad (9)$$

Dempster's Combination Rule

The combination called the joint mass (m12) is calculated from the two sets of masses m1 and m2.

m12 (A) =

$$\frac{B \cap C = A, \Sigma\, m1(B)\, m2(C)}{1 - [B \cap C = \phi, \Sigma\, m1(B)\, m2(C)]} \quad (10)$$

Dempster-Shafer Attribute selection algorithm

The generalization Bayesian statistical theory is essential in Dempster–Shafer algorithm. Here all the new feature allows and supports the distributing propositions (Eg. "Determine defect") Dempster-Shafer reasoning system is enumerated by "frame-of-discernment" system. All the mutual exclusive context interpretation are enumerated and denoted by symbol Θ. E.g.: “An instance is read form the dataset” for a reality constrains this instance may belongs to “defect” or “no defect”. The task specifies the instance identity as one of the four possibilities.

Described by:

$$\Theta = \{A,\ B,\ \{A,B\},\ \phi\} \quad (11)$$

Significances are represented as 1.” defect”, 2. ”no defect”, 3.“Either defect or no defect” actually an indication of ignorance or “neither defect nor no defect” is an indication of exceptional situation. For Software defect prediction, the resulting computation complexity is low, as the frame of discernment consists of only two elements (normal and abnormal). There are up to three focal elements of belief functions: {normal},{abnormal}, and {normal or abnormal} (i.e. the uncertainty),resulting in low computation complexity

The Feature subset generation algorithm Si would contribute its observation by assigning its “belief” with the form of discernment symbolΘ. Here, Si denotes Mi. The Probability mass function is called the assignment function. The probability are indicated by “confidence interval” is observation is calculated by Lower bound and Upper bound method. The probability of Si observation is “the detected attribute is the Attribute-A”. According to confidence interval the lower bound is “Belief” and Upper bound is "plausibility"

[Beliefi(A), Plausibilityi(A)]

Beliefi(A) is quantified by all the pieces of evidence Ek that support proposition “Attribute-A”:

$$\text{Beliefi(A)} = \sum_{E_k \subseteq A} m_i(E_k) \quad (12)$$

Plausibilityi(A) is quantified by all the pieces of evidence Ek that do not rule out the proposition "Attribute-A":

$$\text{Plausibilityi(A)} = 1 - \sum_{E_k \cap A = \phi} m_i(E_k) \quad (13)$$

For each proposition in Θ, e.g., "Attribute-A", Dempster-Shafer theory gives a rule of combining Feature Subset Si’s observation mi and Feature SubsetSj’s observation mj:

$$(m_i \oplus m_j)(A) = \frac{\sum_{E_k \cap E_{k'} = A} m_i(E_k)\, m_j(E_{k'})}{1 - \sum_{E_k \cap E_{k'} = \phi} m_i(E_k)\, m_j(E_{k'})} \quad (14)$$

This rule can be bounded directly, if it is viewed mj not as Feature Subset Sj’s observation, instead of the previously combined observations of Feature Subset Sk and Sl. Human perception-Reasoning Process is captured by Dempster – Shafer approach. Where the lower end “belief “and the upper end “plausibility" are the two different key features captured by Dempster – Shafer reasoning process. To compare in order to show the difference, The Bayesian algorithm is a essential subset of Dempster – Shafer approach. Here No mechanism is provided for the quantitative dealings. Quantitative dealings are the range of "belief" and "plausibility". These are humans characteristic attach to the likelihood[20]. In some specific problem the desired features of k varies. If DSI (fi) with DSI (fi-1) is lower than the small value then the process will stop. Here fi and fi-1 are the selected features. The iteration process for the selected features is i-th and i-1th respectively. S is the selected feature and fi fetches small information into S which has already selected. This process leads to a slight change in the classification where the result increases slightly [21].The relationships between Belvalue, Pl value and uncertainty are described in Figure 2.

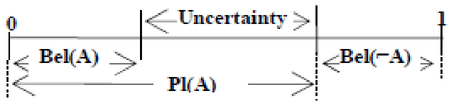

Figure 2: The uncertainty interval for a hypothesis

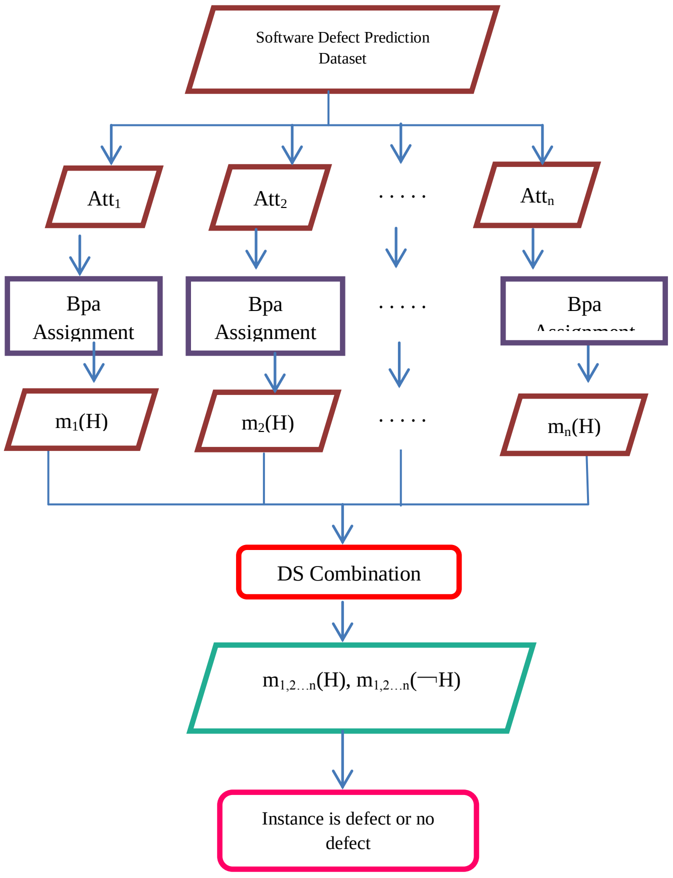

Figure 3: Feature subset Selection in Software defect Prediction

The figure 3 depicts the overall flow of the proposed feature subset selection using modified combined rule of Dempster-shafer theory

DUBOIS AND PRADE'S DISJUNCTIVE CONSENSUS RULE

Dubois and Prade take a set-theoretic view of a body of evidence to form their disjunctive consensus rule in [24, 25]. They define the union of the basic probability assignments m1 È m2 (denoted by $m\cup(C)$) by extending the set-theoretic union:

$$m\cup(C) = \Sigma A \cup B = C\ m1 (A)\ m2 (B)$$

For all A of the power set X. The union does not generate any conflict and does not reject any of the information asserted by the sources. As such, no normalization procedure is required. The drawback of this method is that it may yield a more imprecise result than desirable.
The union can be more easily performed via the belief measure:

Let $Bel1_1 \cup Bel_2$ bethe belief measure associated with $m1\cup m2$. Then for every subset A of the universal setX,

$$Bel_1(A)\text{È}Bel_2(A) = Bel_1(A)\ Bel_2(A)$$

The disjunctive pooling operation is commutative, associative, but not idempotent.

Algorithm

```
1. Collect a training data set from the specific domain.
2. Shuffle the data set.
3. Break it into P partitions, (say P = 20)
4. For each partition ( i = 0, 1, ..., P-1 )
        a. Let OuterTrainset(i) = all partitions except i.
        b. Let OuterTestset(i) = the i'th partition
        c. Let InnerTrain(i) = randomly chosen 70% of the OuterTrainset(i).
        d. Let InnerTest(i) = the remaining 30% of the OuterTrainset(i).
        e. For j = 0, 1, ..., m
        Search for the best feature set with j components, $fs_{ij}$.using leave-one-out on InnerTrain(i) Let $InnerTestScorei_j$= RMS score of $fs_{ij}$ on InnerTest(i).
        End loop of (j).
        f. Select the $fs_{ij}$ with the best inner test score.
        g. Let $OuterScore_i$ = RMSE score of the selected feature set on OuterTestset(i) End of loop of (i).
5. Return the mean Outer Score
```

Dubois and Prade's Disjunctive Consensus Pooling

The unions of multiple sets are summarized in Table 1.

Table 1: Results of union of multiple sets

Union	*M*è	Linguistic Interpretation
$A\cup A$	0	Failure of Component A
$A\cup B$	0.0196	Failure of Component A or B
$A \cup C$	0.9604	Failure of Component A or C
$B \cup B$	0.0004	Failure of Component B
$B \cup C$	0.0196	Failure of Component B or C
$C \cup C$	0	Failure of Component C
$A \cup B \cup C$	1	Failure of Component A or B or C

Experimental Result

This paper has proposed dempster-shafer theory based attribute selection process for software defect prediction. To validate this process the dataset which is publicly available from NASA MDP repository is used [1, 2]. The simulation of this proposed work is done using matlab. This experiment result uses 10-fold cross validation strategy for evaluating the performance of feature subset selection algorithms namely Greedy Search (GS), Random search (RS), Exhaustive Research (ES) and Proposed Dempster Shafer feature selection algorithm. To validate the each feature selection technique rule induction classifier is used for finding the performance of each in terms of correctly classified instances, incorrectly classified instances, accuracy, precision and recall. The detailed explanation of evaluation metric is shown in the next subsection.

Evaluation Metrics

This work used four different NASA dataset namely CM1, JM1, KC1, PC1. In order to select potential attributes from the whole attributes given by these datasets, four feature selection techniques are implied and their performance is compared using the following metrics.

Table 2: The confusion matrix for the software defect prediction is depicted in the following table 2.

Table 2: software defect prediction

		Module actually has defects	
		No (P)	Yes(N)
Classifier predicts no defects	No(P)	a(TP)	b(FP)
Classifier predicts some defects	Yes(N)	c (FN)	d(TN)

TP = true positives: number of examples predicted positive that are actually positive

FP = false positives: number of examples predicted positive that are actually negative

TN = true negatives: number of examples predicted negative that are actually negative

FN = false negatives: number of examples predicted negative that are actually positive

Accuracy = $\frac{a+d}{a+b+c+d} = \frac{tp+tn}{tp+tn+fp+fn}$ (15)

Probability of Detection =PD = Recall = $\frac{d}{b+d} = \frac{tp}{tp+fn}$ (16)

Probability of False Alarm = PF = $\frac{c}{a+c} = \frac{fp}{tp+fp}$ (17)

Precision = $\frac{d}{c+d} = \frac{tp}{tp+fp}$ (18)

F –Measure= 2 * $\frac{\Pr ecision.recall}{\Pr ecision + recall}$ (19)

Relative Absolute Error = $\frac{|p_1 - a_1| + ... + |p_n - a_n|}{|\bar{a} - a_1| + ... + |\bar{a} - a_n|}$ (20)

Root relative squared error = $\frac{(p_1 - a_1)^2 + ... + (p_n - a_n)^2}{(\bar{a} - a_1)^2 + ... + (\bar{a} - a_n)^2}$ (21)

Where,

Actual target values: a1 a2 … an

Predicted target values: p1 p2 … pn

Mean value of actual target values: $\bar{a}$

Dataset Description

This subsection describes the datasets used in this work. Four public data sets CM1, JM1, PC1 and KC1obtained from NASA MDP [1, 2] Repository made available by PROMISE [3]are used for evaluation of the proposed work. Each of these dataset consists of a set of features characterized by static code metrics, such as LOC counts, Halstead and McCabe complexity metrics. These features are characterizing objectively the software quality. The In McCabe metrics are a collection of four software metrics: Essential complexity, cyclomatic complexity, design complexity and Lines of Code.CM1 dataset consists of 498 instances, JM1 consists of 10885, KC1 dataset consists of 2109 instances and PC1 contains 1109 instances. All the 4 dataset consists of 22 attributes. In the dataset 5 of the attributes are used for representing different lines of code measure, 3 attributes represents the McCabe metrics, 4 attributes refers to Halstead measures, 8 attributes refers derived Halstead measures, a branch-count attribute, and 1 goal field attribute which is called as class which classifies the instance as presence or absence of defect.

Table 3 shows the description of each attributes used in the four dataset of this research work.

Table 3: Attribute Description of the four Dataset

S.No	Variables	Description
1	Loc	McCabe's line count of code
2	v(g)	McCabe "cyclomatic complexity"
3	ev(g)	McCabe "essential complexity"
4	iv(g)	McCabe "design complexity"
5	N	Halstead total operators + operands
6	V	Halstead "volume"
7	L	Halstead "program length"
8	D	Halstead "difficulty"
9	I	Halstead "intelligence"
10	E	Halstead "effort"
11	B	Halstead
12	T	Halstead's time estimator
13	LOCode	Halstead's line count
14	LOComment	Halstead's count of lines of comments
15	LOBlank	Halstead's count of blank lines
16	lOCodeAndComment	
17	uniq_Op	unique operators
18	uniq_Opnd	unique operands
19	total_Op	total operators
20	total_Opnd	total operands
21	branchCount	% of the flow graph
22	Defects	Yes/No module has/has not one or more

Table 4: Comparison Feature Subset Selection of CM1 Dataset with DST, GS,ES,RS

Dataset: CM1		
Algorithm	**No of features**	**selected attributes**
DST	7	1,4,9,14,15,17,18
GS	6	1,4,14,15,17,18
ES	7	1,4,9,11,14,15,17
RS	6	1,4,9,11,14,17

The above table 4 shows the comparison of 4 different features subset selection for CM1 dataset. In this even though the DST and ES selects 7 attributes out of 22 but they differ in selection of attributes that is DST selects 18th attribute whereas Es selects 11th attribute. While GS and RS selects 6 attributes from whole features they also differ in selection of attributes in this GS it selects 15th and 18th attribute and RS selects 9th and 11th attribute.

Table 5: Comparison Feature Subset Selection of JM1 Dataset with DST, GS,ES,RS

Dataset: JM1		
Algorithm	**No of features**	**selected attributes**
DST	7	3,7,8,9,18,20,21
GS	7	2,3,5,7,8,9,18
ES	7	3,7,8,9,18,20,21
S	6	2,3,7,8,9,18

The above table 5 shows the comparison of 4 different features subset selection for JM1 dataset. In this even though the DST, GS and ES selects 7 SS attributes out of 22 and the DST and ES selects same set of attributes while GS differs by selecting attributes 2,3,5,7,8,9 and 18. While RS selects 6 attributes from whole features.

Table 6: Comparison Feature Subset Selection of KC1 Dataset with DST, GS,ES,RS

Dataset: KC1		
Algorithm	**No of features**	**selected attributes**
DST	8	6,8,9,13,14,15,18,21
GS	9	2,6,8,9,13,14,15,18,21
ES	8	6,8,9,13,14,15,18,21
RS	8	6,8,9,13,14,15,18,20

The above table 6 shows the comparison of 4 different features subset selection for KC1 dataset. In this DST, ES and RS selects 8 attributes out of 22 and the DST and ES selects same set of attributes while RS differs by selecting attributes 21st attribute instead of 20th. GS selects 9 attributes from whole features.

Table 7: Comparison Feature Subset Selection of PC1 Dataset with DST, GS,ES,RS

Dataset: PC1		
Algorithm	**No of features**	**selected attributes**
DST	4	11,14,15,16
GS	4	11,14,15,16
ES	4	11,14,15,16
RS	4	11,14,15,16

The above table 7 shows the comparison of 4 different features subset selection for PC1 dataset. In this DST, GS, ES and RS selects 4 attributes out of 22 and it is observed that all the four selects same set of subsets namely 11,14,15 and 16 which mainly influences the detection of software defect in PC dataset

Validation using Rule Induction Classifier

In this feature subset selection after selecting each feature subset from the four different datasets by four different feature subset selection algorithms and their performance for classifying the given instance as defect or no defect is determined using the rule induction classifier which acts as the validator for each feature subset algorithm. The detailed explanation is given in the following.

Table 8: Performance comparison of subset selection techniques for CM1 dataset using Rule induction classifier

CM1 Performance				
	DST	GS	ES	RS
Correctly classified	90.106	88.96	89.36	89.36
Incorrectly classified	9.8394	11.04	10.64	10.64
Mean absolute error	0.1174	0.1774	0.1781	0.18
Root mean squared error	0.2979	0.3076	0.3026	0.3066
Relative absolute error	99.1988	99.1569	99.5603	100
Root relative squared error	99.9982	103.27	101.6	103
TPR	0.902	0.89	0.991	0.894
FP rate	0.902	0.9	0.902	0.902
Precision	0.813	0.812	0.812	0.812
Recall	0.902	0.89	0.894	0.894
F measure	0.855	0.849	0.851	0.851

In this it is observed from the table 8 that the proposed DST classifies correct instances of classes with the highest percentage of 90.1%, the second ranks ES and RS with value 89.36% and GS holds 3rd rank with 88.96%.

The reason for better performance of DST is even though its selects 7 attributes they are the most promising attributes sub set than the remaining three because the ability to handle uncertainty and confliction in selecting attributes is overcome by the evidence of the rule adopted in DST. So it concludes that the subsets 1,4,9,14,15,17 and 18 are the potential attributes for predicting presence of defect in CM1 dataset. Another interesting observation is that even though ES and RS differs in number of attributes selected while former selects 7 attributes and latter 6 attributes they differ in inclusion of only one extra feature that is 15th attribute which is not presented in RS.

The GS holds last position because it fails to pick the 9th attribute of the feature set. The attributes Halstead intelligence and unique operands influences in prediction of software defects.

The error rates (MSE, RMSE,RAE,RRSE) produced by DST is considerably less while considering the remaining three algorithms. And the accuracy measures of better performance based on (TPR,FPR, Precision, Recall and F-measure) are increased from other remaining feature subset selection techniques.

Table 9: Performance comparison of subset selection techniques for JM1 dataset using Rule induction classifier

JM1 Performance			
	DST, ES	GS	RS
Correctly classified	80.86	80.7	80.72
Incorrectly classified	19.14	19.3	19.28
Mean absolute error	0.2887	0.3001	0.2996
Root mean squared error	0.3815	0.3887	0.3884
Relative absolute error	92.48	96.14	95.99
Root relative squared error	96.57	98.4	98.31
TPR	0.81	0.81	0.81
FP rate	0.7	0.77	0.76
Precision	0.77	0.76	0.76
Recall	0.81	0.81	0.81
F measure	0.76	0.74	0.74

In this it is observed that the proposed DST and RS feature set based classification classifies correct instances of classes with the highest percentage of 80.86% with 19.14% as incorrectly classified, the second ranks RS with value 80.72% correctly classified that is 8786 instances are correctly predicted with 19.28%is incorrectly classified that is 2099 and GS holds 3rd rank with very slight difference with the performance o RS by wrongly predicting two more instances are incorrectly predicted and produces 80.70% as correctly classified and 19.30% as incorrectly classified of in 88.96%.

Still DST performs because its error rates (MSE, RMSE,RAE,RRSE) are lesser than GS, ES and RS. It is observed that DTS, GS and ES selected same number of attributes 7 but vary in attribute selection. DTS and ES takes into a account total operands and percentage of flow graph while GS selects Mc Cabes cyclomatic complexity and Halstead total operators and operands which leads to worst performance compared to other methods. The RS produces 6 attributes as subsets for predicting software defect by not concentrating on total operands and percentage of flow graph. So it is concluded that in JM1 dataset the software defect prediction is mainly influenced by 7 potential attributed namely McCabe "essential complexity", Halstead "program length", Halstead "difficulty", Halstead "intelligence", unique operands, total operands and % of the flow graph.

Table 10: Performance comparison of subset selection techniques for KC1 dataset using Rule induction Classifier

KC1 Performance		
	DST ,ES,RS	GS
Correctly classified	84.26	84.16
Incorrectly classified	15.74	15.84
Mean absolute error	0.219	0.2214
Root mean squared error	0.3495	0.3512
Relative absolute error	83.71	84.64
Root relative squared error	96.68	97.14
TPR	0.84	0.84
FP rate	0.69	0.70
Precision	0.81	0.80
Recall	0.84	0.84
F measure	0.81	0.81

From the table 10 an interesting pattern is observed that DST,ES and RS produces same result in classification of instances as presence or absence of defect with correctly classified instances as 1777 with 84.26% and 332 are wrongly classified with 15.74%. These three feature selection selects same number of attributes namely Halstead "volume", Halstead "difficulty", Halstead "intelligence", Halstead's line count, Halstead's count of lines of comments, Halstead's count of blank lines, unique operands and % of the flow graph. The GS feature subset selection produces very slight variation in its performance with wrongly predicting two more instances compared to other three methods and yields 84.16% as correctly classified and 15.84% as incorrectly classified. The error percentage of GS is more similar to DTS, ES and RST with precision, recall and F-measure with values 0.80, 0.84 and 0.81 respectively.

Table 11:Performance comparison of subset selection techniques for PC1 dataset using Rule induction Classifier

PC1 Performance	
	DST, ES, GS,RS
Correctly classified	92.97
Incorrectly classified	7.03
Mean absolute error	0.1248
Root mean squared error	0.2548
Relative absolute error	96
Root relative squared error	99.86
TPR	0.93
FP rate	0.90
Precision	0.90
Recall	0.93
F measure	0.90

The result of the table 11 shows that the PC1 dataset is influenced by only four attributes namely, 11,14,15 and 16. It is a more interesting result produced by all the four feature subset algorithms produces same result of correctly classified instances as1031 with 92.97% and 78 instances as incorrectly classified is represented with 7.03%. The mean absolute error is 0.1248 and root mean square error is 0.2358. The percentage of relative absolute error and relative route square error is 96% and 99. 86% respectively. The values of true positive rate and recall 0.93. the value of false positive rate, precision and f measure is 0.90. Among 22 attributes with 1109 instances of PC1 dataset only 4 attributes are more promising they are Halstead, Halstead's count of lines of comments, Halstead's count of blank lines, IOCode and Comment and all the 4 features subset methods selects same set of features contributes higher classification of accuracy.

IV. CONCLUSION

An attribute selection process for software defect prediction is illustrated in this research work using the proposed Dempster Shafer theory with modified combination rule known as Dubois and Prade's Disjunctive Consensus Rule. The results indicated that the proposed approach produce significantly better results for finding important attributes which can be used for defect prediction. It is observed that the DST based Feature subset selection produces consistent performance of holding first rank in all the four selected dataset while the others are fail to do so. This is because the DST holds itself the justification of feature selection based on the rule of evidence, handling the presence of uncertainty in selection of attributes based on mass value of each selected attribute and their combination rule, while other are intended only on observation of searching the attributes based on heuristic approach. The PC1 dataset has more accuracy result in prediction of software defects while comparing the other datasets. And the number of attributes chosen for this process is also comparatively very less. All the four feature subset methods are producing their utmost performance on this dataset more effectively.

REFERENCES

[1]. http://promise.site.uottawa.ca/SERepository

[2]. http://mdp.ivv.nasa.gov.

[3]. http://promise.site.uottawa.ca/SERepository/datasets-page.html

[4]. Y. Chen, P. Du,Xi , X.-H. Shen, "Research on Software Defect Prediction Based on Data Mining", Computer and Automation Engineering(ICCAE), 2nd International Conference, (2010), vol. 1, pp. 563-567.

[5]. H.Najadat and I.Alsmadi, "Enhance Rule Based Detection for Software Fault Prone Modules", International Journal of Software Engineering and Its Applications, vol. 6, no. 1, (2012).

[6]. A.Okutan, O. T.Yildiz, "Software defect prediction using Bayesian networks", Empirical Software Engineering, (2014), vol. 19, no. 1, pp. 154-181.

[7]. T. Nu Phyu, "Survey of Classification Techniques in DataMining", International MultiConference of Engineers and Computer Scientists, (2009); Hong Kong.

[8]. K. Sankar, S. Kannan and P.Jennifer, "Prediction of Code Fault Using Naive Bayes and SVM Classifiers Middle-East Journal of Scientific Research", vol. 20, no. 1, (2014), pp.108-113.

[9]. Y. Ma,C. Bojan,"Singh:Robust prediction of fault-proneness by random forests ,Software Reliability Engineering", ISSRE 2004. 15th International Symposium,(2004),pp. 417-428.

[10]. A. Chug1 and S. Dhall1, "Software Defect Prediction Using Supervised Learning Algorithm and Unsupervised Learning Algorithm", The Next Generation Information Technology Summit (4th International Conference),(2013),pp.1-6.

[11]. S. Agarwal and D.Tomar, "A Feature Selection Based Model for Software Defect Prediction", International Journal of Advanced Science and Technology, vol.65,(2014), pp. 39-58.

[12]. C.-P.Chang a,*, C.-P.Chu a, Y.-F.Yehb, "Integrating in-process software defect prediction with association mining to discover defect pattern", Information and Software Technology ,vol. 51, no. 2, (2009), pp. 375-384.

[13]. M. L., H. Zhang, R. Wu, Z.-H. Zhou, "Sample-based software defect prediction with active and semi-supervised learning", Automated Software Engineering , (2012), vol. 19, no. 2, pp. 201-230

[14]. P.Dhiman, M.C. Manish,"A Clustered Approach to Analyze the Software Quality Using Software Defects, Advanced Computing & Communication Technologies (ACCT)", 2012 Second International Conference,(2012).

[15]. X. Yuan, H.W. Zhang, S. Ying,F. Wang, "Software defect prediction based on collaborative representation classification", Proceedings in ICSE Companion 2014, 36th International Conference on Software Engineering, pp. 632-633.

[16]. K. Gao, T..M.Khoshgoftarr, "Software Defect Prediction for high-dimensional and class-imbalanced data", 23rd International Conference on Software Engineering & Knowledge Engineering (SEKE'2011), Eden Roc Renaissance, (2011)Miami Beach, USA.

[17]. The Global Conference for Wikimedia,(2014); London.

[18]. M. Surendra Naidu, "Classification of Defects in Software Using Decision Tree Algorithm", International Journal of Engineering Science and Technology (IJEST), (2013).

[19]. Black, Paul E. (2 February 2005). "greedy algorithm". Dictionary of Algorithms and Data Structures. U.S. National Institute of Standards and Technology (NIST). Retrieved 17 August 2012.

[20]. Dempster, A. P. (1967). "Upper and Lower Probabilities Induced by a MultivaluedMapping." The Annals of Statistics 28: 325-339.

[21]. Shafer, G. (1976). A Mathematical Theory of Evidence. Princeton, NJ, PrincetonUniversity Press

[22]. Klir, G. J. and M. J. Wierman (1998). Uncertainty-Based Information: Elements ofGeneralized Information Theory. Heidelberg, Physica-Verlag.

[23]. Zadeh, L. A. (1986). A Simple View of the Dempster-Shafer Theory of Evidence and itsImplication for the Rule of Combination. The AI Magazine. 7: 85-90.

[24]. Dubois, D. and H. Prade (1986). "A Set-Theoretic View on Belief Functions: LogicalOperations and Approximations by Fuzzy Sets." International Journal of GeneralSystems 12: 193-226.

[25]. Dubois, D. and H. Prade (1992). "On the combination of evidence in variousmathematical frameworks." Reliability Data Collection and Analysis. J. Flammand T. Luisi. Brussels, ECSC, EEC, EAFC: 213-241.