Classifier Evaluation for Software Defect Prediction

Gang Kou[1], Yi Peng[1], Yong Shi[2, 3], Wenshuai Wu[1]

[1] School of Management and Economics,
University of Electronic Science and Technology of China,
No.4, Sec 2, North Jianshe Rd, Chengdu, 610054, China,
kougang@yahoo.com; (Corresponding author)

[2] College of Information Science & Technology,
University of Nebraska at Omaha,
Omaha, NE 68182, USA,
yshi@unomaha.edu

[3] CAS Research Center on Fictitious Economy and Data Sciences,
Beijing 100080, China,
yshi@gucas.ac.cn

Abstract : Feature selection is an essential step in the process of software defect prediction due to the negative effect of irrelevant features on classification algorithms. Hence selecting the most relevant and representative features is critical to the success of software defect detection. Another problem in software defect prediction is the availability of a large number of classification models. This paper applies feature selection and classifier evaluation in the context of software defect prediction. An empirical study is presented to validate the proposed scheme using 9 classifiers over 4 public domain software defect data sets. The results indicate that the proposed scheme can improve the performance of classifiers using the most representative features and recommend classifiers that are accurate and reliable in software defect prediction.

Keywords: software defect detection; classifier evaluation; multi-criteria decision making (MCDM);

1. Introduction

Defects are prevalent in large and complex software systems and cause huge losses to organizations [1-2]. Timely and accurate software defect prediction can help identify faults in an early stage of software development lifecycle, which facilitates efficient test resource allocation, improves software architecture design, and reduces the number of defective modules [3]. Software defects prediction with high accuracy and reliability is a challenge and active research area.

Classification is one of the most important tasks in data mining [4] and is a commonly used approach in software defect prediction. It models software defects prediction as a two-group classification problem through categorizing software modules as either fault-prone (fp) or non-fault-prone (nfp) using historical data. A large number of classification algorithms have been developed over the years for software defect prediction [5-7].

Software defect data sets normally collect large number of attributes to describe the characteristics of software modules at various states of the software development process. Since the attributes collected in software defect data may not be relevant to software defects classification, including all these attributes in the model-building process can deteriorate the performances of classifiers. Thus feature subset selection is an essential step in the process of software defect prediction.

This paper integrates traditional feature selection methods and multi-criteria decision making (MCDM) methods to improve the accuracy and reliability of defect prediction models and evaluate the performances of software defect detection models. An experimental study is designed to validate the propose scheme using 9 classifiers over 4 public domain software defect data sets.

The rest of this paper is organized as follows: section 2 reviews related works. Section 3 describes the research methodologies. Section 4 presents the experimental study and analyzes the results; section 5 summarizes the paper.

2. Related Works in Software Defect Prediction

Compared with other application domains, the use of feature selection in the area of software defect prediction is relatively new. Rodriguez et al. [8] applied attribute selection algorithms in faulty software modules classification and concluded that the classification results using selected feature subsets were better or equal to the results with the original features. They validated the approach using a case study and concluded that the performances of prediction models either improved or unaffected when 85% of the original features were removed.

Researchers from different disciplines have proposed a variety of classification models for software defect prediction. Porter and Selby [9] presented metric-based classification trees to identify high-risk software components. Emam et al. [10] further analyzed the performance of a CBR classifier with different parameters for predicting fault-prone software components. Their conclusion is that a simple CBR is suitable for software defect prediction. Khoshgoftaar and Seliya [11] presented two classification rules in the context of case-based reasoning and showed that the proposed techniques achieved high levels of accuracy and robustness.

Many empirical studies have been conducted to compare different software prediction models and some studies generate contradictory results [12]. For example, Shepperd and Schofield [13] stated that analogies outperformed stepwise regression models on nine industrial datasets. Peng et al. [7] applied a set of MCDM methods to rank classification algorithms for the task of software defect detection. Myrtveit and Stensrud [14], on the other hand, found that regression models surpassed an analogy tool in their experiment. It is hard to tell which prediction models are sufficient for a given defect dataset. Hence algorithm selection is a crucial issue in software defect prediction.

3. Research Methodology

Results of empirical studies on software defect prediction models do not always converge. Myrtveit et al. [12] analyzed some empirical software engineering studies and identified three factors that may contribute to the divergence: a single sample dataset, choice of accuracy indicators, and cross validation. They concluded that a crucial step in software defect prediction is the design of research procedures.

The inputs are four public-domain software defect datasets provided by the NASA IV&V Facility Metrics Data Program (MDP) repository. Feature selection and classification are conducted in four steps. First, feature selection is conducted using traditional techniques. Features are then ranked using the proposed feature selection method. The third step employs MCDM methods to evaluate feature selection techniques and choose the better performed techniques. In the last step, the selected features are used in the classification to predict software defects. The performances of classifiers are also evaluated using MCDM methods and a recommendation of classifiers for software defect prediction is made based on their accuracy and reliability.

Multiple criteria decision making (MCDM) aims at solving decision problems with multiple objectives and often conflictive constraints [15-18]. Five MCDM methods, i.e., DEA (BCC model), ELECTRE, PROMETHEE, TOPSIS, and VIKOR, are used in the experimental study to evaluate algorithms.

For feature selection algorithms, output components include seven attributes:

– LOC_COMMENTS (The number of lines of comments in a module),
– HALSTEAD_PROG_TIME (The halstead programming time metric of a module),
– MAINTENANCE_SEVERITY (Maintenance Severity),
– NODE_COUNT (Number of nodes found in a given module),
– NUM_OPERATORS (The number of operators contained in a module),
– NUM_UNIQUE_OPERATORS (The number of unique operators contained in a module),
– PERCENT_COMMENTS (Percentage of the code that is comments).

All other attributes are input components. For classification algorithms, input component is false positive rate and output components include the area under receiver operating characteristic (AUC), precision, F-measure, and true positive rate.

4. Experimental study

4.1 Data sources

The data used in this study are modified public-domain software defect datasets provided by the NASA IV&V Facility Metrics Data Program (MDP) repository [19]. The structures of the datasets are summarized in Table l.

Table 1. Dataset structures [20]

Dataset	Number of instances	Normal instances	Bug instances
CM	568	425	143
KC	804	495	309
PC	4472	3718	754
UC	10064	9285	779

CM is from a science instrument written in a C code with approximately 20 kilo-source lines of code (KLOC). KC is about the collection, processing and delivery of satellite metadata and is written in Java with 18 KLOC. PC is flight software from an earth orbiting satellite written in a C code with 26 KLOC. UC is dynamic simulator for attitude control systems. Forty common attributes are selected for each dataset.

4.2 Discussion of results

Table 2 summarizes the feature weights for each dataset. Features that are highly ranked in one or two dataset may have low rankings in other datasets, such as attribute 4, 9, and 27. This indicates that performances of feature selection techniques vary at different datasets. It also shows a need for evaluation of feature selection techniques.

The five MCDM methods are applied to evaluate the eleven feature selection techniques. The rankings of each feature selection techniques are averaged for each dataset. The standard deviations of feature selection techniques generated by the five MCDM rankings are also computed to measure the stability of feature selection techniques. When two feature selection techniques have the same average ranking, the one with smaller standard deviation outperforms the other one in terms of stability. Table 3, 4, 5, and 6 represent the evaluation results of feature selection techniques by MCDM methods for the four datasets. Cfs and Consistency represent CfsSubsetEval and ConsistencySubsetEval, respectively. The other nine techniques are WrapperSubsetEval evaluator with corresponding base learners. For instance, W.NaiveBayes represents WrapperSubsetEval evaluator with naïve Bayes as the base learner.

Table 2. Feature weights for the four datasets (W for Weight, R for Rank)

Attributes	CM Data		KC Data		PC Data		UC Data	
	W	R	W	R	W	R	W	R
att1	0.57	7	0.44	24	0.60	12	0.68	3
att2	0.26	37	0.40	30	0.35	39	0.22	39
att3	0.64	5	0.59	8	0.63	8	0.47	28
att4	0.27	36	0.57	9	0.95	1	0.74	2
att5	0.57	8	0.51	15	0.55	17	0.64	9
att6	0.48	21	0.30	39	0.43	31	0.48	26
att7	0.41	26	0.55	12	0.51	21	0.40	35
att8	0.44	23	0.33	37	0.65	7	0.67	4
att9	0.68	3	0.35	33	0.47	27	0.31	38
att10	0.33	33	0.64	2	0.69	4	0.62	13
att11	0.50	19	0.48	19	0.52	20	0.47	29
att12	0.33	32	0.57	10	0.55	18	0.62	12
att13	0.56	10	0.51	16	0.45	29	0.60	14
att14	0.51	18	0.44	23	0.63	9	0.60	15
att15	0.52	17	0.34	36	0.40	35	0.42	34
att16	0.24	39	0.51	14	0.49	24	0.45	30
att17	0.49	20	0.49	18	0.56	16	0.66	6
att18	0.56	9	0.41	29	0.29	40	0.18	40
att19	0.29	35	0.61	4	0.43	34	0.43	33
att20	0.43	25	0.60	5	0.77	2	0.79	1
att21	0.47	22	0.47	21	0.43	32	0.53	19
att22	0.52	14	0.44	25	0.49	25	0.47	27
att23	0.43	24	0.43	26	0.43	33	0.59	17
att24	0.52	16	0.39	31	0.49	23	0.53	21
att25	0.54	12	0.49	17	0.59	13	0.52	22
att26	0.55	11	0.42	28	0.58	14	0.55	18
att27	0.72	1	0.43	27	0.39	37	0.40	36
att28	0.62	6	0.54	13	0.46	28	0.33	37
att29	0.38	30	0.34	35	0.40	36	0.49	23
att30	0.65	4	0.38	32	0.65	6	0.65	8
att31	0.24	38	0.32	38	0.45	30	0.48	24
att32	0.37	31	0.45	22	0.62	10	0.60	16
att33	0.40	28	0.25	40	0.47	26	0.43	32
att34	0.32	34	0.35	34	0.50	22	0.53	20
att35	0.70	2	0.64	3	0.68	5	0.62	11
att36	0.52	13	0.59	7	0.57	15	0.65	7
att37	0.41	27	0.56	11	0.61	11	0.64	10
att38	0.52	15	0.47	20	0.36	38	0.43	31
att39	0.40	29	0.65	1	0.73	3	0.66	5
att40	0.21	40	0.59	6	0.53	19	0.48	25

Table 3. Rankings of feature selection techniques for CM dataset

Feature Selection	DEA	ELECTRE	PROMETHEE	TOPSIS	VIKOR	Average Ranking	Standard Deviation
Cfs	4	2	4	9	3	4.4	2.702
Consistency	10	11	10	10	11	10.4	0.548
W.NaiveBayes	2	5	2	2	8	3.8	2.683
W.Logistic	11	10	9	8	7	9	1.581
W.RBFNetwork	1	3	5	1	1	2.2	1.789
W.SMO	7	4	8	4	4	5.4	1.949
W.IB1	3	1	1	3	5	2.6	1.673
W.FLR	8	6	3	6	9	6.4	2.302
W.DecisionTable	5	8	7	5	10	7	2.121
W. RIPPER	6	9	11	7	2	7	3.391
W.C4.5	9	7	6	11	6	7.8	2.168

Table 4. Rankings of feature selection techniques for KC dataset

Feature Selection	DEA	ELECTRE	PROMETHEE	TOPSIS	VIKOR	Average Ranking	Standard Deviation
Cfs	5	4	2	2	1	2.8	1.643
Consistency	11	10	5	4	11	8.2	3.421
W.NaiveBayes	6	2	7	10	7	6.4	2.881
W.Logistic	4	11	11	11	8	9	3.082
W.RBFNetwork	8	3	6	7	6	6	1.871
W.SMO	9	6	8	8	3	6.8	2.387
W.IB1	1	1	1	1	2	1.2	0.447
W.FLR	3	5	3	3	5	3.8	1.095
W.DecisionTable	2	8	10	9	4	6.6	3.435
W. RIPPER	10	7	9	5	9	8	2.000
W.C4.5	7	9	4	6	10	7.2	2.387

Table 5. Rankings of feature selection techniques for PC dataset

Feature Selection	DEA	ELECTRE	PROMETHEE	TOPSIS	VIKOR	Average Ranking	Standard Deviation
Cfs	1	7	1	2	1	2.4	2.608
Consistency	7	10	10	9	7	8.6	1.517
W.NaiveBayes	9	8	6	11	5	7.8	2.387
W.Logistic	8	11	11	10	9	9.8	1.304
W.RBFNetwork	11	4	5	6	3	5.8	3.114
W.SMO	10	9	9	8	11	9.4	1.140
W.IB1	3	1	2	1	2	1.8	0.837
W.FLR	6	2	3	3	4	3.6	1.517
W.DecisionTable	5	3	4	4	10	5.2	2.775
W. RIPPER	2	6	8	5	8	5.8	2.490
W.C4.5	4	5	7	7	6	5.8	1.304

Table 6. Rankings of feature selection techniques for UC dataset

Feature Selection	DEA	ELECTRE	PROMETHEE	TOPSIS	VIKOR	Average Ranking	Standard Deviation
Cfs	5	6	4	1	1	3.4	2.160
Consistency	3	5	8	9	9	6.8	2.754
W.NaiveBayes	1	1	1	3	3	1.8	1.000
W.Logistic	4	3	9	4	7	5.4	2.708
W.RBFNetwork	9	7	7	11	8	8.4	1.915
W.SMO	7	10	5	6	6	6.8	2.160
W.IB1	8	11	3	8	5	7	3.317
W.FLR	6	2	2	2	2	2.8	2.000
W.DecisionTable	2	4	6	5	11	5.6	1.708
W. RIPPER	11	9	11	10	10	10.2	0.957
W.C4.5	10	8	10	7	4	7.8	1.500

Based on the ranking results, five highly-ranked feature selection techniques: Cfs, W.RBFNetwork, W.IB1, W.FLR, and W. NaiveBayes, are chosen to recalculate feature weights and the results are summarized in Table 7. Thirteen features that are ranked above the twentieth position and have average weights larger than 0.5 are selected for the classification task.

The selected features are then used in classification algorithms to predict software defects. The results are compared with classification results using features selected by traditional feature selected technique. Table 8, 9, 10, and 11 represent the classification results and comparisons between the proposed feature selection scheme and traditional feature selection techniques of the four datasets. The results suggest that the proposed feature selection scheme helps classifiers achieve better classification outcomes in terms of the five performance measures in general.

Table 7. Recalculated feature weights (W for Weight, R for Rank)

Attributes	CM Data		KC Data		PC Data		UC Data	
	W	R	W	R	W	R	W	R
att1	0.51	12	0.39	24	0.48	14	0.52	17
att2	0.2	38	0.29	35	0.27	35	0.24	38
att3	0.63	5	0.68	1	0.65	6	0.4	25
att4	0.28	32	0.47	15	0.96	1	0.68	3
att5	0.66	3	0.51	10	0.47	16	0.49	20
att6	0.54	9	0.17	40	0.26	37	0.27	35
att7	0.33	28	0.47	14	0.4	23	0.3	34
att8	0.25	34	0.26	37	0.68	4	0.67	5
att9	0.7	2	0.3	34	0.3	32	0.24	37
att10	0.3	31	0.55	4	0.53	13	0.58	12
att11	0.27	33	0.45	17	0.26	36	0.4	26
att12	0.24	35	0.44	18	0.41	21	0.59	11
att13	0.52	11	0.51	11	0.23	38	0.55	15
att14	0.44	17	0.39	24	0.65	5	0.51	18
att15	0.4	20	0.31	31	0.17	40	0.41	24
att16	0.19	39	0.46	16	0.37	27	0.45	23
att17	0.5	13	0.49	13	0.45	17	0.57	13
att18	0.53	10	0.36	27	0.3	31	0.21	39
att19	0.22	36	0.54	6	0.28	34	0.33	33
att20	0.38	24	0.55	5	0.77	2	0.85	1
att21	0.36	26	0.41	23	0.35	28	0.5	19
att22	0.38	23	0.39	26	0.4	22	0.4	27
att23	0.33	29	0.33	28	0.41	20	0.62	9
att24	0.42	18	0.26	36	0.28	33	0.39	28
att25	0.55	8	0.51	9	0.62	10	0.48	22
att26	0.49	14	0.44	19	0.62	9	0.65	8
att27	0.76	1	0.26	38	0.34	29	0.26	36
att28	0.56	7	0.53	7	0.2	39	0.19	40
att29	0.4	21	0.31	29	0.39	26	0.53	16
att30	0.64	4	0.31	33	0.65	7	0.56	14
att31	0.2	37	0.31	32	0.4	24	0.4	32
att32	0.37	25	0.43	20	0.64	8	0.69	2
att33	0.39	22	0.23	39	0.39	25	0.36	29
att34	0.35	27	0.31	30	0.44	18	0.49	21
att35	0.63	6	0.63	3	0.56	12	0.61	10
att36	0.47	15	0.51	8	0.41	19	0.66	7
att37	0.41	19	0.5	12	0.58	11	0.66	6
att38	0.46	16	0.42	22	0.31	30	0.35	30
att39	0.31	30	0.64	2	0.74	3	0.67	4
att40	0.14	40	0.43	21	0.47	15	0.35	31

Table 8. Classification results comparison for CM dataset

Classifiers	Without feature weight & MCDM					With feature weight & MCDM				
	TP Rate	FP Rate	Precision	F-Measure	ROC	TP Rate	FP Rate	Precision	F-Measure	ROC
Naïve Bayes	0.788	0.498	0.767	0.771	0.77	0.793	0.481	0.774	0.798	0.762
Logistic	0.824	0.394	0.813	0.816	0.773	0.808	0.445	0.793	0.796	0.793
RBFNetwork	0.788	0.59	0.76	0.751	0.756	0.813	0.521	0.8	0.787	0.777
SMO	0.793	0.682	0.837	0.722	0.556	0.782	0.716	0.83	0.701	0.553
IB1	0.829	0.284	0.833	0.831	0.773	0.85	0.34	0.842	0.844	0.755
FLR	0.497	0.23	0.775	0.516	0.634	0.394	0.384	0.832	0.567	0.605
DecisionTable	0.824	0.378	0.815	0.818	0.814	0.839	0.389	0.83	0.829	0.813
RIPPER	0.824	0.347	0.819	0.821	0.743	0.839	0.389	0.83	0.829	0.731
C4.5	0.762	0.304	0.794	0.773	0.759	0.829	0.299	0.83	0.83	0.763

Table 9. Classification results comparison for KC dataset

Classifiers	Without feature weight & MCDM					With feature weight & MCDM				
	TP Rate	FP Rate	Precision	F-Measure	ROC	TP Rate	FP Rate	Precision	F-Measure	ROC
Naïve Bayes	0.78	0.347	0.813	0.758	0.932	0.806	0.301	0.828	0.791	0.928
Logistic	0.813	0.428	0.8	0.803	0.808	0.901	0.121	0.901	0.901	0.938
RBFNetwork	0.864	0.205	0.874	0.859	0.905	0.853	0.204	0.856	0.85	0.936
SMO	0.89	0.151	0.892	0.888	0.87	0.875	0.148	0.875	0.875	0.864
IB1	0.886	0.126	0.887	0.887	0.88	0.85	0.137	0.86	0.852	0.856
FLR	0.456	0.212	0.788	0.462	0.622	0.409	0.257	0.742	0.408	0.576
DecisionTable	0.901	0.133	0.902	0.9	0.935	0.908	0.124	0.909	0.907	0.924
RIPPER	0.912	0.092	0.913	0.912	0.922	0.908	0.117	0.908	0.908	0.899
C4.5	0.905	0.107	0.905	0.905	0.926	0.912	0.095	0.913	0.912	0.909

Table 10. Classification results comparison for PC dataset

Classifiers	Without feature weight & MCDM					With feature weight & MCDM				
	TP Rate	FP Rate	Precision	F-Measure	ROC	TP Rate	FP Rate	Precision	F-Measure	ROC
Naïve Bayes	0.831	0.594	0.806	0.814	0.818	0.78	0.347	0.823	0.768	0.932
Logistic	0.877	0.506	0.866	0.861	0.887	0.873	0.428	0.8	0.863	0.868
RBFNetwork	0.834	0.804	0.791	0.768	0.8	0.864	0.205	0.874	0.859	0.905
SMO	0.864	0.625	0.857	0.834	0.62	0.89	0.151	0.892	0.888	0.87
IB1	0.884	0.263	0.888	0.886	0.811	0.886	0.126	0.887	0.887	0.88
FLR	0.836	0.791	0.802	0.774	0.522	0.456	0.212	0.788	0.462	0.622
DecisionTable	0.864	0.54	0.848	0.847	0.859	0.901	0.133	0.902	0.9	0.935
RIPPER	0.864	0.528	0.849	0.848	0.668	0.912	0.092	0.913	0.912	0.922
C4.5	0.889	0.362	0.883	0.885	0.851	0.905	0.107	0.905	0.905	0.926

Table 11. Classification results comparison for UC dataset

Classifiers	Without feature weight & MCDM					With feature weight & MCDM				
	TP Rate	FP Rate	Precision	F-Measure	ROC	TP Rate	FP Rate	Precision	F-Measure	ROC
Naïve Bayes	0.908	0.542	0.908	0.908	0.907	0.917	0.626	0.906	0.91	0.876
Logistic	0.932	0.597	0.921	0.923	0.93	0.932	0.618	0.92	0.922	0.926
RBFNetwork	0.926	0.843	0.917	0.898	0.868	0.924	0.84	0.902	0.897	0.914
SMO	0.928	0.849	0.933	0.898	0.539	0.927	0.86	0.932	0.896	0.533
IB1	0.884	0.263	0.888	0.886	0.811	0.951	0.286	0.951	0.951	0.868
FLR	0.925	0.88	0.931	0.892	0.522	0.925	0.88	0.931	0.892	0.522
DecisionTable	0.936	0.597	0.925	0.926	0.915	0.941	0.532	0.933	0.934	0.928
RIPPER	0.941	0.372	0.939	0.94	0.789	0.941	0.41	0.938	0.939	0.787
C4.5	0.951	0.3	0.951	0.951	0.851	0.953	0.266	0.954	0.953	0.9

The next step is to use MCDM methods to evaluate classification algorithms based on their performances on software defect datasets. Table 12, 13, 14, and 15 summarize the evaluation results of the nine classifiers on the four datasets. The rankings of classifiers vary with different datasets. Even within a dataset, different MCDM methods may produce divergent rankings for the same classifier. For example, RIPPER was ranked the second best classifier by ELECTRE and the worst classifier by DEA for CM dataset. In general, FLR outperforms other classifiers. It was ranked the best classifier by at least two MCDM methods for every dataset. SMO achieves good performances on PC and UC, which are larger than CM and KC. The performances of other classifiers on the four software defect datasets are rather mixed.

Table 12. MCDM evaluation of classifiers for CM dataset

	DEA	ELECTRE	PROMETHEE	TOPSIS	VIKOR
Naïve Bayes	2	3	2	2	2
Logistic	8	7	6	6	1
RBFNetwork	7	5	7	5	6
SMO	6	9	4	8	5
IB1	5	8	8	9	9
FLR	1	1	1	1	3
DecisionTable	3	6	9	3	4
RIPPER	9	2	3	7	7
C4.5	4	4	5	4	8

Table 13. MCDM evaluation of classifiers for KC dataset

	DEA	ELECTRE	PROMETHEE	TOPSIS	VIKOR
Naïve Bayes	5	5	3	2	7
Logistic	1	2	2	1	1
RBFNetwork	7	4	4	4	9
SMO	6	6	6	5	8
IB1	9	9	5	7	6
FLR	4	1	1	3	3
DecisionTable	3	3	7	6	2
RIPPER	2	8	9	9	5
C4.5	8	7	8	8	4

Table 14. MCDM evaluation of classifiers for PC dataset

	DEA	ELECTRE	PROMETHEE	TOPSIS	VIKOR
Naïve Bayes	9	9	3	4	7
Logistic	8	6	7	7	5
RBFNetwork	2	3	4	3	3
SMO	1	1	2	2	1
IB1	5	8	9	9	9
FLR	4	2	1	1	2
DecisionTable	3	4	6	6	6
RIPPER	7	5	5	5	8
C4.5	6	7	8	8	4

Table 15. MCDM evaluation of classifiers for UC dataset

	DEA	ELECTRE	PROMETHEE	TOPSIS	VIKOR
Naïve Bayes	5	8	3	4	6
Logistic	3	4	5	5	3
RBFNetwork	2	5	4	3	2
SMO	1	2	2	2	1
IB1	8	7	8	8	7
FLR	4	1	1	1	5
DecisionTable	7	3	7	6	4
RIPPER	6	9	6	7	8
C4.5	9	6	9	9	9

4.3 Conclusion remarks

Feature selection and classification are two important tasks in software defect prediction. An experimental study was designed to validate the propose scheme using 9 classifiers over 4 public domain software defect data sets. Five MCDM methods (i.e., DEA, ELECTRE, PROMETHEE, TOPSIS, and VIKOR) were applied in the study to evaluate feature selection techniques and classifiers for software defect prediction. Thirteen features were selected using the weights and MCDM methods. The experimental results indicate that fuzzy lattice reasoning (FLR) outperforms other classifiers for the selected datasets. In addition, SMO achieves good performances on two larger datasets. The performances of other classifiers on the four software defect datasets are rather mixed.

The experimental study suggests that MCDM methods may generate different rankings for classifiers on the same dataset. How to find a compromised solution when MCDM methods generate conflicting rankings of classifiers is a future research direction. In addition, the experimental study chose only four relatively small datasets. Applying the proposed scheme to larger sizes datasets is another direction of future work.

Acknowledgements

This research has been supported by grants from the National Natural Science Foundation of China (#70901015 for Gang Kou, #70901011 and #71173028 for Yi Peng, #70921061 for Yong Shi).

REFERENCES

1. FILIP, F. G., **Decision Support and Control for Large-scale Complex Systems**, Annual Reviews in Control, Elsevier, vol. 32, no. 1, 2008, pp. 61-70.

2. FILIP, F. G., K. LEIVIISKA, **Large-scale Complex Systems, Springer Handbook of Automation**, Springer, Berlin, 2009, pp. 619-638.

3. LESSMANN, S., 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, 2008, pp. 485-496.

4. PENG, Y., G. KOU, Y. SHI, Z. CHEN, **A Descriptive Framework for The Field of Data Mining and Knowledge Discovery**, International Journal of Information Technology and Decision Making, vol. 7, no. 4, 2008, pp. 639-682.

5. KOU, G., Y. PENG, Z. CHEN, Y. SHI, **Multiple Criteria Mathematical Programming for Multi-class Classification and Application in Network Intrusion Detection**, Information Sciences, vol. 179, no. 4, 2009, pp. 371-381.

6. PENG, Y., G. KOU, G. WANG, H. WANG, F. KO, **Empirical Evaluation of Classifiers For Software Risk Management**, International Journal of Information Technology and Decision Making, vol. 8, no. 4, 2009, pp. 749 -768.

7. PENG, Y., G. WANG, H. WANG, **User Preferences based Software Defect Detection Algorithms Selection using MCDM**, Information Sciences, doi:10.1016/j.ins.2010.04.019, 2010.

8. RODRIGUEZ, D., R. RUIZ, J. CUADRADO-GALLEGO, J. AGUILAR-RUIZ, M. GARRE, **Attribute Selection in Software Engineering Datasets for Detecting Fault Modules**, EUROMICRO, 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO 2007), 2007, pp. 418-423

9. PORTER, A. A., R. W. SELBY, **Evaluating Techniques for Generating Metric-Based Classification Trees**, Journal of Systems and Software, vol. 12, no. 3, 1990, pp. 209-218.

10. EMAM, K. E., S. BENLARBI, N. GOEL, S. N. RAI, **Comparing Case**-based **Reasoning Classifiers** for **Predicting High Risk Software Components**, Journal of Systems and Software, vol. 55, no. 3, 2001, pp. 301-310.

11. KHOSHGOFTAAR, T. M., N. SELIYA, **Analogy-Based Practical Classification Rules for Software Quality Estimation**, Empirical Software Eng., vol. 8, no. 4, 2003, pp. 325-350.

12. MYRTVEIT, I., E. STENSRUD, M. SHEPPERD, **Reliability and Validity in Comparative Studies of Software Prediction Models**, IEEE Transcriptions Software Engineering, vol. 31, no. 5, 2005, pp. 380-391.

13. SHEPPERD, M. J., C. SCHOFIELD, **Estimating Software Project Effort Using Analogies**, IEEE Transcriptions Software Engineering, vol. 23, no. 12, 1997, pp. 736-743.

14. MYRTVEIT, I., E. STENSRUD, **A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models**, IEEE Transcriptions Software Engineering, vol. 25, no. 4, 1999, pp. 510-525.

15. ERGU, D., G. KOU, Y. PENG, Y. SHI, **A Simple Method to Improve the Consistency Ratio of the Pair-wise Comparison Matrix in ANP**, European Journal of Operational Research, vol. 213, no. 1, 2011, pp. 246-259.

16. ERGU, D., G. KOU, Y. SHI, Y. SHI, **Analytic Network Process in Risk Assessment and Decision Analysis**, Computers & Operations Research, doi:10.1016/j.cor.2011.03.005, 2011.

17. KOU, G., Y. SHI, S. Y. WANG, **Multiple Criteria Decision Making and Decision Support Systems -** Guest editor's introduction, DOI:10.1016/j.dss.2010.11.027, Decision

Support Systems, vol. 51, no. 2, 2011, pp. 247-249.

18. PENG, Y., G. KOU, G. WANG, Y. SHI, **FAMCDM: A Fusion Approach of MCDM Methods to Rank Multiclass Classification Algorithms**, Omega, vol. 39, no. 6, 2011, pp. 677-689.

19. CHAPMAN, M., P. CALLIS, W. JACKSON, **Metrics Data Program**, NASA IV and V Facility, http://mdp.ivv.nasa.gov/, 2004.

20. PENG, Y., G. KOU, G. WANG, W. WU, Y. SHI, **Ensemble of Software Defect Predictors: An AHP-based Evaluation Method**, International Journal of Information Technology & Decision Making, vol. 10, no. 1, 2011, pp. 187-206.