Software Fault Prediction: A Systematic Mapping Study

Juan Murillo-Morera[1], Christian Quesada-López[2], Marcelo Jenkins[2]

[1] Department of Informatics, National University of Costa Rica
[2] Center for ICT Research, University of Costa Rica
juan.murillo.morera@una.cr, cristian.quesadalopez@ucr.ac.cr,
marcelo.jenkins@ecci.ucr.ac.cr

Abstract. Context: Software fault prediction has been an important research topic in the software engineering field for more than 30 years. Software defect prediction models are commonly used to detect faulty software modules based on software metrics collected during the software development process. **Objective:** Data mining techniques and machine learning studies in the fault prediction software context are mapped and characterized. We investigated the metrics and techniques and their performance according to performance metrics studied. An analysis and synthesis of these studies is conducted. **Method:** A systematic mapping study has been conducted for identifying and aggregating evidence about software fault prediction. **Results:** About 70 studies published from January 2002 to December 2014 were identified. Top 40 studies were selected for analysis, based on the quality criteria results. The main metrics used were: Halstead, McCabe and LOC (67.14%), Halstead, McCabe and LOC + Object-Oriented (15.71%), others (17.14%). The main models were: Machine Learning(ML) (47.14%), ML + Statistical Analysis (31.42%), others (21.41%). The data sets used were: private access (35%) and public access (65%). The most frequent combination of metrics, models and techniques were: Halstead, McCabe and LOC + Random Forest, Naive Bayes, Logistic Regression and Decision Tree representing the (60%) of the analyzed studies. **Conclusions:** This article has identified and classified the performance of the metrics, techniques and their combinations. This will help researchers to select datasets, metrics and models based on experimental results, with the objective to generate learning schemes that allow a better prediction software failures.

Keywords: Fault prediction models, Software metrics, Software quality.

1 Introduction

Software fault prediction has been an important research topic in the software engineering field for more than 30 years [1]. The software measurement data collected during the software development process includes valuable information about a software project status, progress, quality, performance, and evolution.

Software fault prediction models is a significant part of software quality assurance and commonly used to detect faulty software modules based on software measurement data (software metrics) [2]. Fault prediction modeling is an important area of research and the subject of many previous studies that produce fault prediction models, which allows software engineers to focus development activities on fault-prone code, thereby improving software quality and making better use of resource of the system with a high fault probability [3]. The current defect prediction work, focuses on three approaches: estimating the number of defects remaining in software systems, discovering defect associations, and classifying the defect-proneness of software components, typically in two classes, defect-prone and non defect-prone [1]. The first approach, employs statistical methods to estimate a number of defect or defects density [4]. The prediction result can be used as an important measure for the software developer and can be used to control the software process, for example; decide, whether to schedule further inspections or pass the software artifacts to the next development step. The second approach, borrows association rule mining algorithms from the data mining community to reveal software defect associations [5]. The third approach classifies software components as defect-prone and non-defect-prone [6], [7]. Our aim is characterize the models, considering the metrics, techniques, and performance metrics with the objective to find relations, combinations, patterns and learning schemes (metrics, data preprocessing, attributes selector and techniques) with the best performance to predictive software faults. Further we evaluated the experimentation quality following Host checklist [8]. Our intention have been to find strengths and weaknesses in experimental designs.

The remainder of the paper is structured as follows: Section 2 presents the related work, Section 3 presents the research design, Section 4 presents search strategy, Section 5 details the classification scheme used for the map analysis. Section 6 the mapping results are presented. Section 7 discuss the results. Finally, Section 8 presents conclusions and future work.

2 Related Work

In this section, a review of secondary studies that have been conducted in fault prediction software is presented. *Catal et al.* [9], present a systematic review of fault prediction studies listing metrics, techniques, and datasets. The results shows that the percentage of use of public databases and machine learning techniques increased significantly since 2005. *Elberzhager et al.* [10], present a systematic mapping study. The main goal of this study is the identification of existing approaches that are able to reduce testing effort. Therefore, an overview should be presented; both for researchers and practitioners in order to identify on the one hand, future research directions and, on the other hand, potential for improvements in practical environments. *Danijel et al.* [11], identify software metrics and assess their applicability in predicting software faults. The influence of context for the selection and performance of the metrics was investigated.
The aim of this review and that distinguishes it from previous work, is the

characterization of the models, considering the metrics, techniques, and performance metrics with the objective to find relationships, combinations, and learning schemes with the best performance to predictive software faults.

3 Research design

Secondary studies aggregate evidence from primary studies [12]. To increase the validity of the results, it is important to be systematic when evidence is analyzed. Systematic mapping studies (SMS) provides an overview of a research area, and identifies the quantity and type of research[13]. The following sections detail the protocol for the SMS according to the guidelines proposed in [14],[13], and considering the recommendations of [15],[16], due to space limitations, we briefly described the main steps from the protocol.

3.1 Systematic Mapping questions

A systematic mapping study have conducted in order to identify, categorize, analyze and characterize fault prediction models in the context of software quality. Table 1 states the mapping study questions and the motivation for each question as well.

Table 1. Systematic Mapping questions

Question	Description	Motivation
MQ1	¿Which are the main journals and conferences for software fault-proneness research?	Identify the main journals and conferences that provide publications in the area.
MQ2	¿Which are the principal authors for software fault-proneness research?	Determine the principal authors of publications in the area.
MQ3	¿How has the frequency of articles related to metrics changed over time?	The answer indicates research trends over time, such as MCcabe, Halstead, LOC, OO and combined metrics.
MQ4	¿How has the frequency of articles related to data mining and machine learning techniques changed over time?	The answer indicates research trends over time, such as Bayesian Network, Decision Tree and Linear Regression related to models.
MQ5	¿Which combinations of metrics and models have been used?	Analyze the possible combinations of metrics and models and their performance.
MQ6	¿Which has been the quality of the experimentation of the studies?	Analyze the quality of the experimentation based on protocol and the experimental setup.

4 Search strategy

The search strategy aim is to find all relevant primary studies for answering research questions. First, the search string is defined and relevant databases are selected, then, the studies inclusion and exclusion criteria and procedure are defined.

4.1 Search string

The definition of the search string is based on population, intervention, comparison and outcome (PICO) [17]. Population: Data sets or historial data bases,

software projects and software applications. Intervention: fault prediction or fault-proneness prediction. Comparison: the focus of the study was not limited to comparative studies. In consequence, comparison was not considered in the search string. Outcome: tools, metrics, techniques and models. Characteristics and empirical evidence of performance of fault prediction models were searched. Outcomes about input metrics, techniques, tools, performance metrics (accuracy, area under a curve (AOC), confusion matrix) were included. The search terms used in the systematic mapping study were constructed using the following strategy: (1) major terms were derived from the research questions by identifying the population, intervention, and outcome, (2) alternative spellings and synonyms were identified, based on a reference set of relevant articles. The reference set was created including the articles that are relevant to be included in the systematic mapping study. The reference set consisted of the following references: [18], [19], [20], [21], [22], [23], [24], [25], [26], [15]. These articles were selected according to their content, relevance and relation with the systematic mapping study objective. (3) Besides, alternative terms were included via expert knowledge in the field, (4) the string was constructed with the Boolean OR for alternative spellings and the AND to link PICO categories. Finally, (5) the search string was piloted in several runs in order to reduce the quantity of noise of the results. It was assured that papers in the reference set was returned by the search string. The results were documented for a complete search string reference (Appendix A)[1].

4.2 Selection of databases

The papers were searched based on title, abstract and keywords. The result set of articles were merged and duplicates removed. For the selection of electronic databases, we followed recommendations of several systematic literature and mapping studies in software engineering. (Appendix B) [1]. These electronic databases were considered relevant in software engineering and offers functionality for complex search strings. Further, these databases offers good coverage for papers in the area. For example, SCOPUS indexed ACM and IEEE articles. Finally, these electronic databases are well known because their stability and interoperability with several referential bibliographic databases.

4.3 Study selection criteria and procedure

The inclusion and exclusion of the articles were based on title and abstract. The articles returned by the search were evaluated according to specific inclusion and exclusion criteria. For the inclusion of the articles, proposals of data mining and machine learning in fault prediction area must be mentioned in the abstract. Articles that use statistical analysis, data mining or artificial intelligence techniques to predict defects were considered. Finally a validation should be conducted. The inclusion and exclusion criteria have been defined, evaluated

[1] http://eseg-cr.com/research/2014/Appendix-SLR-JMM-CQL-MJC.pdf

and adjusted by the author and a fellow colleague in several pilot studies in order to reduce bias (Appendix C)[1].

As part of our preliminary selection process, one author applied the search strategy to identify potential primary studies in the electronic databases. The inclusion and exclusion criteria are an input to the study selection procedure. The selection procedure was conducted following these steps: A) Two researchers read the titles and abstracts separately. The papers were categorized as follows: (A.1) Accepted: articles that fulfill the inclusion criteria, (A.2) Rejected: articles that fulfill the exclusion criteria, and (A.3) Not defined: the reviewer is not sure about inclusion and exclusion criteria. The reason for having two researchers is the reduction of bias when including or excluding articles. B) After both authors applied the criteria on the articles independently, the results were compared. The results categorized the articles as follows: (B.1) Included: articles accepted by both authors, (B.2) Excluded: articles rejected by both authors, and (3) Uncertain: the reviewers are not sure about inclusion and exclusion criteria. C) Based on detailed review. Both authors discussed this classification and make the decision about inclusion or exclusion. Finally, accepted articles were read in detail. Flutist of all studies included in the first step were read.

The search contained articles published before 2014. The search was conducted (piloted, refined, and finished) during the first semester of the year 2014. The quality assessment, extraction, mapping and analysis of the data was conducted during the second semester of the year 2014. The preliminary results were written and reviewed in this period.

4.4 Search and selection results

Fig. 1 shows the number of identified and remained articles after each step in the selection process. In total 89 articles were identified in the electronic databases. After inclusion and exclusion criteria, 74 primary studies were selected for a detailed review. Through the detailed reading, 4 articles were discarded, leaving 70 primary studies being selected in the evaluation, extraction, and analysis of this review. Most of the selected papers were indexed by Engineering Village, and SCOPUS.

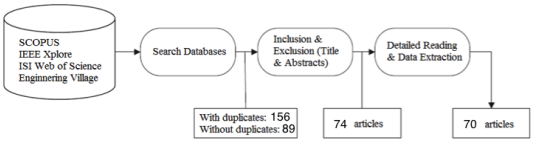

Fig. 1. Exclusion of articles and number of primary studies (Adapted from [27])

[1] http://eseg-cr.com/research/2014/Appendix-SLR-JMM-CQL-MJC.pdf

4.5 Study quality assessment

Quality assessment is concerned with the rigor of the study and the applicability of the fault prediction models in software engineering area. The quality of the articles was evaluated according to the experimentation used and the input metrics, techniques and models, performance metrics and clarity to describe the experiments. The criteria to evaluate quality include high-level questions and additional quality issues. The quality criteria are based on questions listed in Appendix D[1]. The overall quality rank papers from 0.7 (very low quality) to 2.1 (excellent quality). Each criteria is divided in sections. The first is regarding to the data source (data set public or private or project). The second is regarding to the input metrics. The third is regarding to the models and techniques used. Finally, the fourth is regarding to the kind of experimentation, description and evaluation of the results. It is recommended that at least three reviewers are needed to make a valid quality assessment [28]. The quality assessment checklist was conducted for two researchers to eventually adjust them in order to reduce bias.

4.6 Data extraction

Based on the selection procedure, the required data for answering the systematic mapping study questions are extracted. This information is additional to the study quality criteria. A data extraction form was created and filled in for each article by the primary author of this paper. The data extraction form has four different sections to be filled in. The following information was captured in the extraction: The first section captures general information about the article such as: data extractor, title, author, year, journal, conference, and study identifier. The second section extracts data about the characterization of the fault prediction models. The extracted information is related to fault prediction models such as data sets, metrics, techniques, and performance metrics. The third section extract information about the empirical validation, reported results and kind of experimentation. The outcomes of interest related to the fault prediction models. The last section validates experimentation according to [8].

4.7 Analysis

The most advanced form of data synthesis is meta-analysis. However, this approach assumes that the synthesized studies are homogeneous [12]. Meta-analysis is not applied in this review because varieties of model and evaluation approaches have been discussed in the selected papers. Dixon-Woods et al. [29], [27] describes the content analysis and narrative summary as approaches to integrate evidence. Content analysis categorizes data and analyzes frequencies of categories transferring qualitative into quantitative information [27].

[1] http://eseg-cr.com/research/2014/Appendix-SLR-JMM-CQL-MJC.pdf

4.8 Threats to validity

Search process: It is based on SCOPUS, IEEEXplore, ISI Web of Science, and Engineering Village.

These databases were considered relevant in software engineering and offers functionality for complex search strings. Further, these databases offers good coverage for articles in the area. The search was based on title, abstract, and keywords and we could missed relevant papers. Gray literature and papers in other languages were not included. **Study selection:** It is based on title and abstract. The inclusion and exclusion criteria were defined, evaluated, and adjusted by two researchers in order to reduce bias. A pilot study was conducted showing a high degree of agreement between two researchers. The researchers included articles for detailed review when there was a doubt and no lower score articles were excluded from the review. The review protocol was peer-reviewed to assure good understandability and clarity for inclusion criteria. **Quality assessment**: a single researcher conducted the quality assessment checklist and a random sample was evaluated for second research. The review protocol was peer-reviewed to assure good understandability and clarity for quality criteria. **Data extraction**: a single researcher conducted the data extraction. Detailed form was defined to make the extraction as structured as possible. The review protocol was peer-reviewed to assure good understandability and clarity for extracted information. After extraction, a single reviewer checked the extraction forms to review if something is missing. Test-retest suggested for single research was not conducted [30]. **Generalizability of the results**: the generalizability of the results is limited by the generalizability of the studies included in the review.

5 Classification schema

There are different approaches in the machine learning and data mining context. Chug [31] divided the models and techniques in supervised learning (which is the most popular approach), and unsupervised learning (on which a little less work is done). In our study, we create a classification scheme based on key-wording described in [13]. The first step in the process was to read the abstracts and keywords of 70 studies (Appendix E)[1], evaluating the concepts and categories according to the frequency. The second step of the process was to classify each item within each category. The process is applied to level metrics and models. The main *categories of metrics* were: 1) Halstead, McCabe and LOC. 2) OO. 3) Halstead, McCabe and LOC + OO and 4) others. The main *categories of techniques and models* were: 1) Machine Learning (ML). 2) ML + Classification. 3) ML + Clustering. 4) ML + Statistical Analysis. 5) Clustering and 6) Statistical Analysis.

[1] http://eseg-cr.com/research/2014/Appendix-SLR-JMM-CQL-MJC.pdf

6 Systematic Mapping Study results

In this section, we analyzed the most common journals and conferences, authors, frequency of articles per metric categories, frequency of articles per techniques and models for the 70 studies(MQ1-MQ4). The questions MQ5 and MQ6 were answered with top 40 studies, according to quality criteria (rigor) presented in Table 2, 3 and 4. We analyzed Table 2, 3 in MQ5 and Table 4 in MQ6.

6.1 List of journals and conferences (MQ1)

The main journals were: Automated Software Engineering (ASE) two articles, Expert Systems with Applications (ESWA) two articles, Software Engineering, IEEE Transactions on (IEEE Transactions) one article, International Journal of Software Engineering and Knowledge Engineering (IJSEKE) one article and Software Quality Journal (SQJ) one article. The main conferences were: International Conference on Software Engineering (ICSE) four articles, Association for Computing Machinery (ACM) two articles, International Symposium Software Reliability Engineering (ISSRE) two articles, Predictive Models in Software Engineering (PROMISE) one article and Mining Software Repositories(MSR) one article. (Appendix F)[1].

6.2 List of authors (MQ2)

The most frequent author was Taghi M. Khoshgoftaar with 7 articles. The other major authors were Yue Jiang and Ruchika Malhotra with 5 articles each one, Cagatay Catal and Naeem Seliya with 4 articles. Finally Yunfeng Luo and Yogesh Singh Singh with 3 articles. (Appendix G) [1].

6.3 Frequency of articles per categories of metrics (MQ3)

The metrics investigated with the highest frequency were Halstead, McCabe and LOC (47 studies - 67.14% from total), Halstead, McCabe and LOC + OO (11 studies - 15.71 % from total), OO (5 studies - 7.14% from total), others (7 studies - 10% from total) (see Figure 2).

6.4 Frequency of articles per categories of techniques and models (MQ4)

The techniques and models investigated with the highest frequency were: Machine Learning (ML) (33 studies - 47.14% from total), ML + Classification (4 studies - 5.71% from total), Machine Learning(ML) + Clustering (4 studies - 5.71 % from total), Machine Learning(ML) + Statistical Analysis (22 studies - 31.42% from total), Clustering (2 studies - 2.85% from total), and Statistical Analysis (5 studies - 7.14% from total) (see Figure 3).

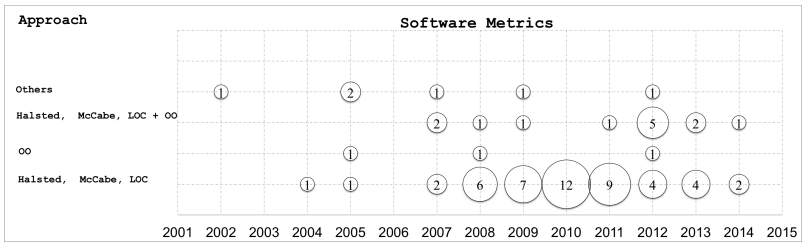

Fig. 2. Frequency of articles per categories of metrics

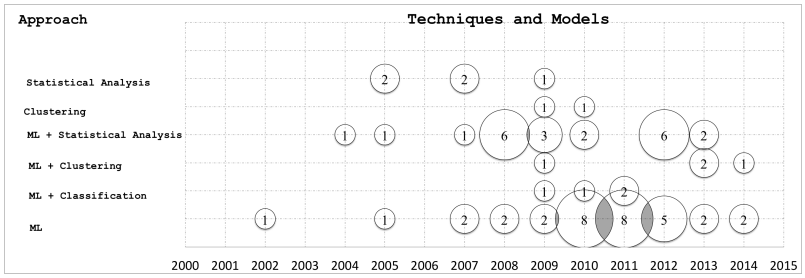

Fig. 3. Frequency of articles per categories of techniques and models

6.5 Combinations of metrics/models and their performance (MQ5)

The most frequent combinations of metrics and models were: Halstead, McCabe and LOC+(NB (and/or) DT (and/or) LR (and/or) RF (and/or) BAG (and/or) BST (and/or) SVM (and/or) AFP (and/or) CC (and/or) K-means (and/or) MLR). We categorize the studies in four groups according to the previous combinations. The G1 with only one technique (see Table 2 and 3, column 2 and 7). This group presented a performance considering metrics such as: PRE(0.39-0.66), REC(0.13-0.37) and AUC(0.53-0.94). The G2 with at least two techniques (see Table 2 and 3, column 2 and 7). This group presented a performance considering metrics such as: ACC(0.67-0.90), FM(0.38-0.39) and AUC(0.55-0.68). The G3 with three or more techniques (see Table 2 and 3, column 2 and 7). This group presented a performance considering metrics such as: BAL(0.29-0.82), REC(0.44-0.99), PREC(0.39-0.99), and AUC(0.40-1.00). The G4 was represented by other combinations between metrics and models (see Table 2 and 3, column 2 and 7). This group presented a performance considering metrics such as: BAL(0.52-0.75), ACC(0.75-0.94), REC(0.94-0.98), PREC(0.95-0.96) and AUC(0.47-0.94). Different performance metrics and the combinations with high performance were: Halstead, McCabe and LOC + NB, Halstead, McCabe and LOC + DT, Halstead, McCabe and LOC + RF and McCabe and

[1] http://eseg-cr.com/research/2014/Appendix-SLR-JMM-CQL-MJC.pdf

LOC + LR and their possible combinations like: Halstead, McCabe and LOC + (NB (and/or) DT (and/or) RF (and/or) LR).

6.6 Quality of the experimentation of the studies (MQ6)

The quality of the experimentation was evaluated using a checklist adapted from [8]. The questions were evaluated with the scale (0 pts, 0.5 pts and 1 pt). With the objective to analyze the experimentation of the studies, we divided all the studies in three groups. The first group with 7 points (see Table 4). This group was characterized by a clear description of the data source used, clear chain of evidence established from observations to conclusions, threats to validity analyses addressed in a systematic way, different views taken on the multiple case collection, analysis methods, multiple authors, and finally conclusions, implications for practice and future research was reported. These studies were characterized for a complete experimental design. An important finding was that the majority of studies classified as experiments were in this group. The second group had a rigor between 5 and 6.5 points (see Table 4) This group was characterized by lack in threats to validity, no clear chain of evidence established from observations to conclusions, no clear conclusions respect to main objective and no clear future work reported. The third group had a rigor between 3 and 4.5 (see Table 4). This group was characterized by lack threats to validity and no clear description of the data source used, clear chain of evidence established from observations to conclusions. In general, few studies had a complete learning scheme configuration, except the studies: SR2, SR29, SR30, SR28, SR26 and SR15.

7 Conclusions and Future work

A total of 70 studies were mapped and classified. The main metrics used in the literature were: Halstead, McCabe and LOC (67.14%), Halstead, McCabe and LOC + OO (15.71%), Combined (10%), OO metrics (7.14%). The main models were: ML (47.14%), ML + Statistical Analysis (31.42%), Statistical Analysis (7.14%), ML + Clustering (5.71%), ML + Classification (5.71%) and Clustering (2.85%). The most frequent combination of metrics, models and techniques were: (Halstead, McCabe and LOC) + (RF, NB, LR and DT) representing the 60% of the analyzed studies. Other important combinations were: OO metrics + (RF, NB, DT, SVM, BST, BAG and MP) with the 10% of the studies. The rest of studies used by Halstead, McCabe and LOC) + (Clustering or PCA) 7.5% or (Halstead, McCabe and LOC) combining others techniques 22.5%. The combination of metrics and most frequent techniques were: Halstead, McCabe and LOC + NB (and/or) DT(J48-C4.5) (and/or) LR (and/or) RF (and/or) BAG (and/or) BST (and/or) SVM, representing 55% of the studies. The other 45% was represented by other type of combinations. The best results were obtained three or more techniques. Related to quality of the experimentation used in the studies, the majority of them classified as experiments, where all the items according to checklist used. The main quality items not completed were: threats to

validity and analysis procedures sufficient for the purpose and evidence established from observations to conclusions. Future work will include finishing the review of all the studies (70 total) and answering systematic literature review questions. The results of this research have allowed to find combinations and some learning schemes.

8 Acknowledgments

This research was supported by the Costa Rican Ministry of Science, Technology and Telecommunications (MICITT). Our thanks to the Empirical Software Engineering Group at University of Costa Rica.

References

1. Song, Q., Jia, Z., Shepperd, M., Ying, S., Liu, J.: A general software defect-proneness prediction framework. Software Engineering, IEEE Transactions on **37** (2011) 356–370
2. Wang, H., Khoshgoftaar, T.M., Napolitano, A.: Software measurement data reduction using ensemble techniques. Neurocomputing **92** (2012) 124–132
3. Hall, T., Beecham, S., Bowes, D., Gray, D., Counsell, S.: A systematic literature review on fault prediction performance in software engineering. Software Engineering, IEEE Transactions on **38** (2012) 1276–1304
4. Compton, B.T., Withrow, C.: Prediction and control of ada software defects. Journal of Systems and Software **12** (1990) 199–207
5. Song, Q., Shepperd, M., Cartwright, M., Mair, C.: Software defect association mining and defect correction effort prediction. Software Engineering, IEEE Transactions on **32** (2006) 69–82
6. Khoshgoftaar, T.M., Allen, E.B., Hudepohl, J.P., Aud, S.J.: Application of neural networks to software quality modeling of a very large telecommunications system. Neural Networks, IEEE Transactions on **8** (1997) 902–909
7. Khoshgoftaar, T.M., Allen, E.B., Jones, W.D., Hudepohl, J.P.: Classification tree models of software quality over multiple releases. In: Software Reliability Engineering, 1999. Proceedings. 10th International Symposium on, IEEE (1999) 116–125
8. Host, M., Runeson, P.: Checklists for software engineering case study research. In: Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on. (2007) 479–481
9. Catal, C., Diri, B.: A systematic review of software fault prediction studies. Expert systems with applications **36** (2009) 7346–7354
10. Elberzhager, F., Rosbach, A., MüNch, J., Eschbach, R.: Reducing test effort: A systematic mapping study on existing approaches. Inf. Softw. Technol. **54** (2012) 1092–1106
11. Radjenović, D., Heričko, M., Torkar, R., Živkovič, A.: Software fault prediction metrics: A systematic literature review. Information and Software Technology **55** (2013) 1397–1418
12. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in software engineering. Springer (2012)
13. Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.: Systematic mapping studies in software engineering. In: 12th International Conference on Evaluation and Assessment in Software Engineering. Volume 17. (2008) 1
14. Keele, S.: Guidelines for performing systematic literature reviews in software engineering. Technical report, Technical report, EBSE Technical Report EBSE-2007-01 (2007)
15. Gyimothy, T., Ferenc, R., Siket, I.: Empirical validation of object-oriented metrics on open source software for fault prediction. Software Engineering, IEEE Transactions on **31** (2005) 897–910
16. Brereton, P., Kitchenham, B.A., Budgen, D., Turner, M., Khalil, M.: Lessons from applying the systematic literature review process within the software engineering domain. Journal of systems and software **80** (2007) 571–583
17. Pai, M., McCulloch, M., Gorman, J.D., Pai, N., Enanoria, W., Kennedy, G., Tharyan, P., Colford Jr, J.M.: Systematic reviews and meta-analyses: an illustrated, step-by-step guide. The National medical journal of India **17** (2003) 86–95
18. Malhotra, R.: Comparative analysis of statistical and machine learning methods for predicting faulty modules. Applied Soft Computing **21** (2014) 286–297
19. Jiang, Y., Cukic, B., Menzies, T., Lin, J.: Incremental development of fault prediction models. International Journal of Software Engineering and Knowledge Engineering **23** (2013) 1399–1425
20. Lu, H., Cukic, B.: An adaptive approach with active learning in software fault prediction. In: Proceedings of the 8th International Conference on Predictive Models in Software Engineering. PROMISE '12, New York, NY, USA, ACM (2012) 79–88
21. Singh, P., Verma, S.: Empirical investigation of fault prediction capability of object oriented metrics of open source software. In: Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on, IEEE (2012) 323–327
22. de Carvalho, A.B., Pozo, A., Vergilio, S.R.: A symbolic fault-prediction model based on multiobjective particle swarm optimization. J. Syst. Softw. **83** (2010) 868–882
23. Seliya, N., Khoshgoftaar, T., Van Hulse, J.: Predicting faults in high assurance software. In: High-Assurance Systems Engineering (HASE), 2010 IEEE 12th International Symposium on. (2010) 26–34
24. Gondra, I.: Applying machine learning to software fault-proneness prediction. Journal of Systems and Software **81** (2008) 186–195
25. Jiang, Y., Cuki, B., Menzies, T., Bartlow, N.: Comparing design and code metrics for software quality prediction. In: Proceedings of the 4th international workshop on Predictor models in software engineering, ACM (2008) 11–18
26. Gao, K., Khoshgoftaar, T.: A comprehensive empirical study of count models for software fault prediction. Reliability, IEEE Transactions on **56** (2007) 223–236
27. Petersen, K.: Measuring and predicting software productivity: A systematic map and review. Information and Software Technology **53** (2011) 317–343
28. Kitchenham, B., Sjøberg, D.I.K., Brereton, O.P., Budgen, D., Dybå, T., Höst, M., Pfahl, D., Runeson, P.: Can we evaluate the quality of software engineering experiments? In: Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement. ESEM '10, New York, NY, USA, ACM (2010) 2:1–2:8
29. Dixon-Woods, M., Agarwal, S., Jones, D., Young, B., Sutton, A.: Synthesising qualitative and quantitative evidence: a review of possible methods. Journal of health services research & policy **10** (2005) 45–53B
30. Kitchenham, B.: What?s up with software metrics?–a preliminary mapping study. Journal of systems and software **83** (2010) 37–51
31. Chug, A., Dhall, S.: Software defect prediction using supervised learning algorithm and unsupervised learning algorithm. (2013)

Table 2. Software fault prediction models (articles 1-20)

(Appendix H,I)[a].

Rigor	Article-Group	Data set	Metrics	Techniques	Performance Metrics	Performance Intervals
2.1	SR1-G3	LLTS, Eclipse project, and KC1.	Halstead, McCabe and LOC.	RANQ,NB,MLP,K-NN and LR	ROC(AUC).	min AUC = 0.83 and max AUC = 0.85.
2.1	SR2-G3	KC3.	Halstead, McCabe, LOC and Miscellaneous	CA,LR,J48 and NB.	Precision, Recall, Confusion Matrix and ROC(AUC).	min pd=0.03 and max pd=0.97 / min notpf=0.54 and max pf=0.98 / min bal=0.29 and max bal=0.82.
2.1	SR3-G2	KC1, KC2, PC1, CM1 and JM1.	Halstead, McCabe and LOC.	NB and RF	Confusion matrix, ROC, AUC, FPR and FNR.	min FPR=1.1 and max FPR=5.1 / min FNR=29.0 and max FNR=45.8.
2.1	SR4-G3	CM1, KC1, KC3, KC4, PC1, PC3, PC4, MW1, MC2, JM1, MC1, PC2, PC5	Halstead, LOC (code). McCabe(design) and LOC	RF, BAG, LR, BST and NB	Recall, Precision, Confusion, Matrix, AUC and ROC.	min AUC = 0.78 and max AUC = 0.84.
2.1	SR5-G3	CM1, KC1, KC3, KC4, PC1, PC3, PC4, MW1, MC2, JM1, MC1, PC2 and PC5.	Halstead, McCabe and LOC.	RF, BAG, LR, BST and NB.	Confusion matrix, ROC and ANOVA.	min AUC = 0.95 and max AUC = 0.97.
2.1	SR6-G2	CM1, JM1 and PC1 and JPL.	Halstead, McCabe and LOC	AR, DT, k-NN, NBC, SVM, BAG and BST	accuracy rate (ACR).	min ACC = 67.1 and max ACC = 78.7.
2.1	SR7-G2	SP1, SP2, SP3, and SP4.	Halstead, McCabe, LOC	NB and DT(C4.5)	Recall, Precision, Confusion Matrix, F-Measure and Geometric Mean.	min ACC = 0.85 and max ACC = 0.87 / min FM = 0.38 max FM = 0.39.
2.1	SR8-G3	Ant (vers: 1.3, 1.4, 1.5, 1.6, 1.7). Camel(vers: 1.0, 1.2, 1.4, 1.6). Ivy(vers: 1.1, 1.4, 2.0). Jedit(vers: 3.2, 4.0). Lucene(vers:2.0, 2.2, 2.4).	Halstead, McCabe and LOC.	NB, DT(C4.5-J48), SVM and LR.	Confusion matrix, precision, recall, AUC and F-measure.	min Rec = 0.44, max Rec = 0.83 / min Prec = 0.39, max Prec = 0.57 / min FM = 0.31, max FM = 0.62.
2.0	SR9-G1	CM1, JM1, KC1, KC3, KC4, MC1, MW1, PC1, PC2, PC3, PC4.	Halstead, McCabe and LOC	J48 Tree, DWT and PCA.	Confusion Matrix and ROC(AUC).	min ACC = 0.92 and max ACC = 0.94 / min Pre = 0.39 and max Pre = 0.66 / min Rec = 0.13 and max Rec = 0.37.
2.0	SR10-G3	CM1, JM1, KC1, PC1.	Halstead, McCabe and LOC.	LR, PR, SVR, NN, SVLR, NB and J48 Tree.	MAE and Sensitivity.	min MAE = 0.06 and max MAE = 0.28.
2.0	SR11-G4	CM1, JM1, KC1, KC2 and PC1.	Halstead, McCabe, LOC and Branch	RF, LDF and LR.	Confusion Matrix and ROC.	The min ACC = 0.75 and max ACC = 0.94 / FPR > 0.87.
2.0	SR12-G1	AR3, AR4, and AR5.	Halstead, McCabe and LOC.	DT and K-means.	Confusion Matrix, Overall Error Rate, FPR, FNR.	min FNR = 0.00 , max FNR = 0.25 / min FPR = 0.00 , max FPR = 0.10.
2.0	SR13-G3	CM1, KC1, KC3, MC1, MC2, PC1, PC2, PC3, PC4 and PC5.	Halstead, McCabe and LOC.	Clustering(SK-means, H-Clustering, MDBC, RF, NB and J48 Tree.	Recall, Precision, ROC, RAE, F-Measure, MSE, RMSE, MAE, RMAE and Accuracy.	min Rec = 0.73, max Rec = 0.99 / min Prec = 0.72, max Prec = 0.99.
2.0	SR14-G4	CM1, KC3, MC2, MW1, PC1, PC2, PC3 and PC4.	6 Halstead, McCabe, LOC and Branch	CCA, NB, BAG, LR and BO	Confusion Matrix, AUC and ROC.	min AUC = 0.63 and max AUC = 0.94.
2.0	SR15-G4	Train1, Test1, Train2 and Test2	BRS, EBS, EBC, BP, BS and BR	RF, SVM and MP	Statistic Test(Test1-Test2).	min FP = 0.00 and max FP = 0.03 / min FN = 0.00 and max FN = 0.65.
2.0	SR16-G4	Eclipse, Lucene and Xalan.	OO, McCabe	RF, LR, NB and DT.	Statistic Test(Whitney U-test).	min F-measure = 0.44 and max F-measure = 0.75 / min BAL = 0.52 and max BAL = 0.75.
2.0	SR17-G1	KC1, PC3, PC4, PC1, CM1 and KC3	Halstead, McCabe and LOC.	AFP and ALS	Confusion matrix, ROC and AUC.	min AUC = 0.60 and max AUC = 0.85.
2.0	SR18-G4	Apache POI.	CK(OO) and QMOOD metrics	LR,UR, MR, NN, DT, SVM, BST, RF, BAG and MP.	Precision, recall, confusion matrix, ROC, sp, ac and sens.	min Sen = 74.7 and max Sen = 89.3 / min Spec = 51 and max Spec = 80 / min AUC = 0.70 and max AUC 0.87.
2.0	SR19-G4	iText, a JAVA-PDF library.	CK(OO) metrics: CBO, NOC, WMC, RFC, DIT and LCOM.	J48, NB.	Precision, recall, Confusion Matrix, AUC and ROC, Accuracy, F-measure.	min AUC = 0.47, max AUC = 0.82.
2.0	SR20-G4	Eclipse releases 2.0, 2.1, and 3.0.	31 code metrics and 18 change metrics	LR, NB, DT and SCC.	Precision, recall, confusion matrix,F-measure and Balance-measure.	Rec > 0.80, TPR > 0.75 and FPR < 0.30.

[a] http://eseg-cr.com/research/2014/Appendix-SLR-JMM-CQL-MJC.pdf

Table 3. Software fault prediction models (articles 21-40)

(Appendix H,I)[a].

Rigor	Article-Group	Data set	Metrics	Techniques	Performance Metrics	Performance Intervals
2.0	SR21-G4	KC1	CBO, DIT, LCOM, NOC, RFC, WMC, CK and SLOC	RA, UCA, MCA, NF and ANFIS.	Confusion matrix and Spearman Correlation.	Results related to correlation study.
2.0	SR22-G4	Release Software Rhino, 1.4R3, 1.5R1, 1.5R2, 1.5R3, 1.5R4, 1.5R5.	CK OO metrics: DIT, NOC, WMC, CBO, LCOM	NB and J48	Recall, Precision, Confusion matrix, ROC and F-measure.	min Rec = 0.94 and max Rec = 0.98 / min Prec = 0.95 and max Prec = 0.96 / min AUC = 0.47 and max AUC = 0.49.
2.0	SR23-G4	NASA IV - V (MDP) data repository.	Halstead, McCabe and LOC.	SA, ACP, SVM and ANN.	MSE.	min MSE = 0.78 and max MSE = 0.85.
2.0	SR24-G3	CM1, KC1, KC3, KC4, PC1, PC3, PC4, MW1 and MC2.	Halstead, McCabe and LOC.	RF, BAG, LR, BST, NB, Jrip, Ibk, J48, Decorate and AODE	Confusion Matrix, Precision, AOC and AUC.	min AUC = 0.4 and max AUC = 1.0.
2.0	SR25-G4	AR5, AR4 and AR3.	Metrics industrial thresholds	Clustering and NB.	Confusion matrix, FPR, FNR and Error.	min FPR = 32.14 and max FPR = 44.09 / min FNR = 12.5 and max FNR = 25.
2.0	SR26-G1	JM1, PC1, PC3, PC4 and KC1	Halstead, McCabe and LOC.	KNN and SVM.	Confusion Matrix, AUC and ANOVA	min AUC = 0.83 and max AUC = 0.94.
2.0	SR27-G1	JM1, PC1, PC3, PC4 and KC1.	Halstead, McCabe and LOC.	RF	Confusion Matrix, ROC, AUC, pf and pd.	min FPR = 0.03 and max FPR = 0.3 / TPR = 0.31 and max TPR = 0.73.
2.0	SR28-G4	Eclipse and KC1.	Eclipse: file, class and method. KC1: CBO, DEPTH, LCOM and WMC.	BN.	Confusion Matrix, F-Measure and True-False Positive Rate.	min TPR = 0.83 and max TPR = 0.84 / min FPR = 0.49 and max FPR = 0.52 / min Prec = 0.22 and max Prec = 0.46 / min F-Measure = 0.60 and max F-Measure = 0.60.
2.0	SR29-G2	SP1, SP2, SP3, SP4, CCCS-2, CCCS-4, CCCS-12, CM1, JM1, KC1, KC2, KC3, MW1 and PC1.	Halstead, McCabe and LOC.	DT(C4.5), RF, AdaCost, Adc2, Csb2, MetaCost, Weighting, and RUS.	Confusion matrix, TPR, TNR, FPR, FNR, ACR and F-measure.	FPR = 0.03 and max FPR = 0.3 / TPR = 0.31 and max TPR = 0.73 / min ACC = 0.70 and max ACC = 0.90
2.0	SR30-G2	CM1, JM1, KC1, KC3, KC4, MC1, MC2, MW1, PC1, PC2, PC3, PC4, PC5, ar1,ar3,ar4 and ar6.	Halstead, McCabe and LOC.	NB, J48 and OneR	Confusion Matrix, ROC, Eval and Pred.	min Bal = 28.9 and max Bal = 82.8 / min AUC = 0.48 and max AUC = 0.96.
1.9	SR31-G1	NASA: PC1,KC1, KC2, KC3, CM1, MW1, MC2, ar3, ar4 and ar5.	Halstead, McCabe and LOC.	CC, NN-filter and TNB.	Confusion matrix, AOC, AUC, pd, pf, TP, FP, FN and TN	min AUC = 0.53 and max AUC = 0.77.
1.9	SR32-G1	LLTS, NASA: KC1 and the Eclipse.	Halstead, McCabe and LOC.	RF.	Confusion Matrix, ROC, AUC.	min AUC = 0.83 and max AUC = 0.87.
1.9	SR33-G4	AR1 and AR6.	Halstead, McCabe, LOC and Micellaneous.	ANN, SVM and DT and CCN.	Confusion Matrix, ROC, AUC.	min AUC = 0.8 and max AUC = 0.9.
1.9	SR34-G4	Ant, Tomcat, Jedit, Velocity, Synapse, Poi, Lucene, Xalan and Ivy.	LOC, CBO, LOCQ, WMC, RFC, LCOM, LCOM3, DIT, and NOC	BN	Confusion Matrix, ROC and AUC.	min AUC = 0.62 and max AUC = 0.84.
1.9	SR35-G4	CXF, Camel, Derby, Felix, Hbase, HadoopC, Hive, Lucene, OpenEJB, OpenJPA, Qpid and Wicket.	File, Class and Method. reported	LR, J48, SVM, and NB.	Confusion Matrix and AUC.	min AUC = 0.6 and max AUC=0.8.
1.9	SR36-G2	KC2,KC1,CM1 and PC1.	Halstead, McCabe and LOC	NB, SVM and NN.	Confusion matrix, Accuracy, Precision, Recall and F-measure.	min ACC = 0.69 and max ACC = 0.78 / min Rec = 0.72 and max Rec = 0.80 / min Prec = 0.68 and max Prec = 0.82.
1.9	SR37-G4	PC1, CM1, JM1, KC1, KC2 and KC1.	Halstead, McCabe, LOC, Miscellaneous and CK-Suite	MOPSO-N, NN, BN, NB, SVM and DT(C45).	Confusion matrix, Precision, Recall Accuracy, Sensitivity, Specificity and F-measure.	min AUC = 0.78 and max AUC = 0.84 / min ACC = 0.84 and max ACC = 0.93.
1.9	SR38-G1	AR3, AR4, and AR5.	Halstead, McCabe and LOC.	K-Means and X Means.	Confusion Matrix, FPR and FNR.	The ranges were min FPR = 0.14 and max FPR = 0.44 / min FNR = 0,05 and max FNR = 0.25.
1.9	SR39-G2	CM1, JM1, KC1, KC3, KC4, MC1, MW1, PC1, PC2, PC3 and PC4.	Halstead, McCabe and LOC.	NB and DT(J48).	Confusion Matrix, Recall, Precision, AUC, F-Measure and True Positive Rate.	The ranges were min AUC = 0.55 and max AUC = 0.68 / min Rec = 0.90 and max Rec = 0.98 / min Prec = 0.83 and mac Prec = 0.94.
1.9	SR40-G1	CM1, JM1, KC1, KC3, MC1, MC2, MW1, PC1, PC2, PC3, PC4, PC5.	Halstead, McCabe and LOC	MLR.	Confusion matrix, Average and AUC.	The range was min AUC = 0.70 and max AUC = 0.74.

[a] http://eseg-cr.com/research/2014/Appendix-SLR-JMM-CQL-MJC.pdf

Table 4. Case studies (Checklist) (1-40) based on [8]

(Appendix H,J)[a].

Rigor	Article	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Experimental Setup - Learning scheme configuration
7	SR3	1	1	1	1	1	1	1	Is a Experiment. **Test option:** 10 fold CrossValidation. **Data preprocessing:** remove outliers. **AttributeSelection:** not applied.
7	SR2	1	1	1	1	1	1	1	Is an Experiment with ANOVA. **Test option:** CrossValidation **Data preprocessing:** logarithmic filter.**AttributeSelection:** CfsSub+BestFirst, CfsSub+GeneticSearch, InfoGainAttEval, DWT and PCA.
7	SR27	1	1	1	1	1	1	1	Is a Case study. **Test option:** not applied. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
7	SR29	1	1	1	1	1	1	1	Is a Case study. **Test option:** 10 fold CrossValidation. **Data preprocessing:** not apply. **AttributeSelection:** not applied.
7	SR8	1	1	1	1	1	1	1	Is a Experiment. **Test option:** 5 fold CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
7	SR19	1	1	1	1	1	1	1	Is a Case study. **Test option:** 10 fold CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
7	SR20	1	1	1	1	1	1	1	Is a Case study. **Test option:** 10 fold CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
7	SR22	1	1	1	1	1	1	1	Is a Case study. **Test option:** CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
7	SR32	1	1	1	1	1	1	1	Is a Case study. **Test option:** 5-fold CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** Feature selection technique.
7	SR31	1	1	1	1	1	1	1	Is a Case study. **Test option:** not applied. **Data preprocessing:** Log filter. **AttributeSelection:** not applied.
7	SR33	1	1	1	1	1	1	1	Is a Case study. **Test option:** K-CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** forward selection and backward.
7	SR37	1	1	1	1	1	1	1	Is a Case study. **Test option:** not applied. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
6.5	SR35	1	1	1	1	1	1	0.5	Is a Case study. **Test option:** not applied. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
6	SR1	1	1	1	1	0	1	1	Is an Experiment with ANOVA. **Test option:** 5 fold crossValidation.**Data preprocessing:** remove non-numeric attributes.**AttributeSelection:** not applied.
6	SR4	1	1	1	1	0	1	1	Is a Case study. **Test option:** CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
6	SR5	1	1	1	1	0	1	1	Is a Experiment. **Test option:** 10x10 CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** filter of code metrics or design metrics.
6	SR6	0.5	0.5	1	1	1	1	1	Is a Experiment. **Test option:** Split randomly. **Data preprocessing:** not applied. **AttributeSelection:** Feature selection and randomize.
6	SR11	1	1	1	0	1	1	1	Is a Case study.**Test option:** not applied.**Data preprocessing:** not applied.**AttributeSelection:** selected five attributes from the literature per each data set.
6	SR12	1	1	1	0	1	1	1	Is a Case study.**Test option:** not applied.**Data preprocessing:** not applied. **AttributeSelection: not applied**.
6	SR16	1	1	1	1	0	1	1	Is a Case study. **Test option:** not applied. **Data preprocessing:** not applied. **AttributeSelection:** random sampling.
6	SR17	1	1	1	1	0	1	1	Is a Case study. **Test option:** not applied. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
6	SR18	1	1	1	1	0	1	1	Is a Case study. **Test option:** 10 fold CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
6	SR21	1	1	1	1	0	1	1	Is a Case study. **Test option:** not applied. **Data preprocessing:** Standardization mean-0, standard desviation-1. **AttributeSelection:** not applied.
6	SR23	1	1	1	1	0	1	1	Is a Case study. **Test option:** CrossValidation. **Data preprocessing:** remove. **AttributeSelection:** not applied.
6	SR29	1	1	1	1	0	1	1	Is a Experimental Case study. **Test option:** 10 fold CrossValidation. **Data preprocessing:** logarithmic filters. **AttributeSelection:** random undersampling.
6	SR30	1	1	1	1	0	1	1	Is a Case study. **Test option:** MxN CrossValidation. **Data preprocessing:** Logarithmic values and none. **AttributeSelection:** (Backward elimination) BE and (Forward selection) FS.
6	SR34	1	1	1	1	0	1	1	Is a Case study. **Test option:** CrossValidation. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
5.5	SR24	1	1	1	1	0	1	0.5	Is a Case study. **Test option:** 10x10 CrossValidation. **Data preprocessing:** not applied, log, nom. **AttributeSelection:** not applied.
5.5	SR28	1	1	1	1	0	0.5	1	Is a Case study. **Test option:** 10 fold CrossValidation. **Data preprocessing:** logarithmic filters. **AttributeSelection:** Remove Attributes. Is divided by categories.
5	SR39	1	1	1	1	0	0.5	0.5	Is a Case study. It is a theoretical study. **Test option:** Split. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
5	SR40	1	1	1	1	0	0.5	0.5	Is a Case study. **Test option:** CrossValidation **Data preprocessing:** not applied. **AttributeSelection:** not applied.
5	SR25	1	1	0.5	0.5	1	0	1	Is a Case study. **Test option:** not applied. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
5	SR26	1	1	0.5	0.5	1	0	1	Is a Case study and Experiment. **Test option:** 5-fold CrossValidation. **Data preprocessing:** Remove all no numeric attributes. **AttributeSelection:** Feature Ranking and data sampling.
5	SR13	1	1	1	0	1	0.5	0.5	Is a Case study.**Test option:** 10 fold crossValidation. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
5	SR14	1	1	1	1	0	0.5	0.5	Is a Case study.**Test option:** not applied.**Data preprocessing:** remove 11 replaceable code metrics.**AttributeSelection:** not applied.
4.5	SR10	1	1	1	1	0.5	0	0	Is a Case study.**Test option:** splitValidation 30(train)-70(test). **Data preprocessing:** PCA, CFS and CBS.**AttributeSelection:** no applied.
4	SR36	0.5	0.5	1	1	0	0.5	0.5	Is a Case study. It is a theoretical study. **Test option:** not applied. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
3.5	SR38	0.5	0.5	0.5	1	0	0	1	Is a Case study. **Test option:** Split. **Data preprocessing:** not applied. **AttributeSelection:** not applied.
3.5	SR15	0.5	0.5	0.5	1	0	0.5	0.5	Is a Case study. **Test option:** splitValidation.**Data preprocessing:** normalized and raw(not-normalize).**AttributeSelection:** CfsSubsetEval, ConsistencySubsetEval and PCA.
3	SR9	1	0	0.5	0.5	0	0	1	Is a Case study with correlation analysis.**Data preprocessing:** not applied. **AttributeSelection:** random method for short and proposed method.

[a] http://eseg-cr.com/research/2014/Appendix-SLR-JMM-CQL-MJC.pdf