A Large-Scale Study of the Impact of Feature Selection Techniques on Defect Classification Models

Baljinder Ghotra
Queen's University, Canada
ghotra@cs.queensu.ca

Shane McIntosh
McGill University, Canada
shane.mcintosh@mcgill.ca

Ahmed E. Hassan
Queen's University, Canada
ahmed@cs.queensu.ca

***Abstract*—The performance of a defect classification model depends on the features that are used to train it. Feature redundancy, correlation, and irrelevance can hinder the performance of a classification model. To mitigate this risk, researchers often use feature selection techniques, which transform or select a subset of the features in order to improve the performance of a classification model. Recent studies compare the impact of different feature selection techniques on the performance of defect classification models. However, these studies compare a limited number of classification techniques and have arrived at contradictory conclusions about the impact of feature selection techniques. To address this limitation, we study 30 feature selection techniques (11 filter-based ranking techniques, six filter-based subset techniques, 12 wrapper-based subset techniques, and a no feature selection configuration) and 21 classification techniques when applied to 18 datasets from the NASA and PROMISE corpora. Our results show that a correlation-based filter-subset feature selection technique with a BestFirst search method outperforms other feature selection techniques across the studied datasets (it outperforms in 70%–87% of the PROMISE–NASA data sets) and across the studied classification techniques (it outperforms for 90% of the techniques). Hence, we recommend the application of such a selection technique when building defect classification models.**

I. Introduction

Defect classification models, i.e., statistical regression or machine learning classifiers that are trained to classify software modules to be defective or not help to identify defect-prone modules in a software system before it is released to users. The classifications are used by Software Quality Assurance (SQA) teams to prioritize their limited resources to system areas that are most likely to be defect-prone.

Data preprocessing techniques like feature selection are an important step when building classification models [42, 54, 56]. Prior studies find that defect classification models often exhibit high misclassification rates due to feature redundancy, correlation, and irrelevance [20, 25]. Indeed, prior work suggests that high misclassification rates and low classification accuracy are often attributed to the number of features [59]. Hence, it is advisable to preprocess features before using them to train defect classification models [16].

When selecting features, one must be mindful of irrelevant and redundant features that result in high dimensionality datasets. High dimensionality datasets can lead to a high computational (training) cost and performance degradation of certain classification models [45, 59, 61, 67, 74]. To combat these risks, recent studies apply feature selection techniques prior to training classification models [4, 42]. These feature selection techniques include: (1) filter-based ranking techniques that select features by estimating their capacity for contributing to the fitness of the classification model; (2) filter-based subset techniques that select features that collectively have good predictive capability [21]; and (3) wrapper-based techniques that use a learning algorithm to evaluate the importance of a feature in a predictive model [21].

Recent studies have compared the impact of feature selection techniques on the performance of defect classification models [21, 49, 59, 61, 74]. Some studies conclude that some feature selection techniques are better than others [21, 49, 61, 74], while other studies conclude that there is no significant difference between the performance of feature selection techniques [59]. We suspect that past studies may have arrived at contradictory results because they compared a limited number of classification techniques across a limited number of datasets.

In this study, we conduct a large scale study to analyze the impact of 30 feature selection techniques (including a no feature selection configuration) on the performance of 21 commonly used classification techniques. The 21 studied classification techniques belong to seven families (i.e., Statistical, Nearest Neighbour, Neural Networks, Rule-Based, Support Vector Machines, Decision Tree, and Ensemble Method) and the 29 studied feature selection techniques belong to three families (i.e., filter-based ranking, filter-based subset, and wrapper-based techniques). We train our classification models using datasets from the NASA and PROMISE corpora, which are of different sizes and domains.

A preliminary analysis indicates that features in the datasets of the NASA corpus contains more data redundant features than in the datasets of the PROMISE corpus, suggesting that the corpora should be analyzed separately. Hence, we conduct separate case studies of each corpus.

We train our defect classification models using a combination of feature selection and classification techniques, in order to address the following research question:

RQ: Do feature selection techniques impact the performance of defect classification models?

Our results indicate that a correlation-based filter-subset feature selection technique with a BestFirst search method, which produces feature subsets using greedy hillclimbing with backtracking, outperforms other feature selection techniques

ending up in the top statistical rank for

- 87% of the studied NASA datasets and 90% of the studied classification techniques.
- 70% of the studied PROMISE datasets and all of the 21 studied classification techniques.

Implications: Hence, we recommend that future defect classification studies apply a correlation-based filter-subset feature selection technique with a BestFirst search method when building classification models.

Paper Organization: The rest of the paper is organized as follows. Section II describes the studied feature selection techniques. Section III discusses the design of our case study. Section IV discusses a preliminary comparison of the NASA & PROMISE corpora. Section V discusses the results of our case study. Section VI discloses the threats to the validity of our study. Section VII surveys related work. Finally, Section VIII draw conclusions.

II. Feature Selection Techniques

This section gives a brief of overview of the studied feature selection techniques (*see* Table I). To enable comparison, we select feature selection techniques that have been used in recent studies [21, 49, 59, 61, 74].

A. Filter-Based Feature Ranking Techniques

Filter-based feature ranking-based techniques order features according to importance scores. In our study, we use the Ranker search method with the following importance scores:
Statistics-Based Scores: When using *ChiSquared* [43], feature importance is evaluated by measuring the chi-squared statistics with respect to the class label (i.e., buggy or non-buggy) [55].

Correlation [25] estimates importance using the Pearson Correlation coefficient between the class label and features.

Clustering Variation [17] evaluates the importance of features by measuring their variation coefficients. High variation coefficients for a feature indicate that its values vary across a wide range, which helps in building a more effective classification model [18].
Probability-Based Scores: *Probabilistic Significance* [2] is a conditional probability-based technique where a significance score is assigned to each feature according to its ability to discriminate between different class labels. A high two-way association score between a feature and a class label indicates more a significant feature.

InfoGain [11] is an entropy-based technique. Information gain measures the reduction in uncertainty of a class label after observing a feature. InfoGain is notably biased toward features with more values [74].

GainRatio [58] penalizes multivalued features to mitigate the bias of InfoGain [74].

Symmetrical Uncertainty [75] evaluates the value of a set of features by measuring their symmetrical uncertainty with respect to another set of features.
Instance-Based Techniques: *ReliefF* [35] is an instance-based ranking technique. An instance from the dataset is randomly sampled and its nearest neighbour is located from the same or the opposite class. The relevance score of each feature is updated by comparing the values of the nearest neighbours features to the sampled instance. This process is repeated for a specified number of instances [27].

ReliefFW is a parameter tuning of *ReliefF*. The parameter 'WeightByDistance' (i.e., weigh the nearest neighbours by their distance) is set to *true* in WEKA.
Classifier-based Techniques: *One Rule* [29] generates a one-level decision rule for each individual feature and features are ranked according to their classification error rate.

SVM [26] uses a Support Vector Machine to estimate feature importance. SVM assigns a weight to each feature and uses the square of the assigned weight to rank the features [5].

B. Filter-Based Feature Subset Techniques

Subset-based techniques evaluate the importance of each feature and select a subset of relevant features. A brief description of the subset-based techniques follows:
Correlation-based Feature Subset Selection Technique: Correlation [28] subset-based techniques result in subsets of features instead of evaluating individual features. The best subset consists of features with low inter-correlation to each other, but high correlation with the class label.
Consistency-based Feature Subset Selection Technique: Consistency subset-based technique use consistency as an indicator to measure the importance of a feature subset. This technique results in a minimal feature subset whose consistency is equal to that of all the features [13]. We use three different search methods with Correlation and Consistency subset-based techniques:

1) The *BestFirst* search method finds the feature subsets using greedy hillclimbing with backtracking. The method may (a) start with an empty set of features and search forward, (b) start with the full set of features and search backward, or (c) start at any point and search in both directions by considering all possible single attribute additions and deletions at a given point. We use the forward BestFirst search method.
2) *Genetic Search* [24] finds a subset of features using a genetic algorithm.
3) *Rank Search* [27] uses a feature-subset evaluator to rank all features using a selection search method (e.g., forward selection). Then, subsets of increasing size are evaluated from the ranked list of features i.e., the best attribute, the best attribute plus the next best attribute, and so on. The best attribute set is reported.

C. Wrapper-Based Feature Subset Techniques

Wrapper-based feature subset techniques use predetermined classification techniques and evaluation measures to evaluate the importance of a feature subset. Cross validation is used to estimate the accuracy of the predetermined classification technique for a set of features.

In this study, we employ four classification techniques and three evaluation measures to construct 12 different wrapper-based subset techniques. Naïve Bayes (NB), Logistic Regression (Log), Repeated Incremental Pruning to Produce Error

TABLE I: Overview of the Studied Feature Selection Techniques

Family	Type	Methods	Abbreviation
Filter-based Feature Ranking Techniques	Statistics-based Techniques	Chi-Square	FR1
		Correlation	FR2
		Clustering Variation	FR3
	Probability-based Techniques	Probabilistic Significance	FR4
		InfoGain	FR5
		GainRatio	FR6
		Symmetrical	FR7
	Instance-based Techniques	ReliefF	FR8
		ReliefF-Weight	FR9
	Classifier-based Techniques	OneR	FR10
		SVM	FR11
Filter-based Subset Selection Techniques	Correlation-based Feature Subset Selection	BestFirst	FS1
		Genetic	FS2
		RankSearch	FS3
	Consistency-based Feature Subset Selection	BestFirst	FS4
		Genetic	FS5
		RankSearch	FS6
Wrapper-based Subset Selection Techniques	k-Nearest Neighbor	Area Under the ROC Curve (AUC)	WS1
		Area Under the Precision-Recall Curve (PRAUC)	WS2
		Root Mean Squared Error (RMSE)	WS3
	Logistic Regression	Area Under the ROC Curve (AUC)	WS4
		Area Under the Precision-Recall Curve (PRAUC)	WS5
		Root Mean Squared Error (RMSE)	WS6
	Naïve Bayes	Area Under the ROC Curve (AUC)	WS7
		Area Under the Precision-Recall Curve (PRAUC)	WS8
		Root Mean Squared Error (RMSE)	WS9
	Repeated Incremental Pruning to Produce Error Reduction	Area Under the ROC Curve (AUC)	WS10
		Area Under the Precision-Recall Curve (PRAUC)	WS11
		Root Mean Squared Error (RMSE)	WS12
None	None	No-filter	None

Reduction (Ripper), and K-Nearest Neighbour (KNN) are described in Table II. For KNN, we select $K = 10$ as recommended by a previous study [74].

Evaluation measures:

1) *Area Under the ROC Curve* (AUC) [15] is a trade-off between the true positive rate (TPR) and false positive rate (FPR).
2) *Area Under the Precision-Recall Curve* (AUPRC) [14] is a trade-off between the precision and recall.
3) *Root Mean Squared Error* (RMSE) [8] measures the deviations of the predicted values and the corresponding true values.

III. Study Design

In this section, we discuss the corpora that we use in our study, and our approach to construct and evaluate defect classification models. Figure 1 provides an overview of the steps in our approach.

A. Studied Datasets: We study publicly available datasets from two different corpora (i.e., NASA and PROMISE). The datasets consist of different types of features. We use the cleaned version (D") of the NASA corpus as provided by Shepperd *et al.* [65] and the PROMISE corpus [1]. An overview of the number of features, modules and defective modules (i.e., modules having defects > 0) for each dataset in the studied corpora are shown in Table III. To aid in comparability, we select the studied corpora used in previous defect classification studies [22, 40, 59, 74]. The studied corpora are available online (http://openscience.us/repo/), enabling further replication (and extension) of our results.

B. Fold Generation: In order to train our classification models, we use the 10×10-fold cross-validation approach. Cross-validation divides the dataset into 10 folds of equal size (i.e., 90% training and 10% testing dataset). The divided training dataset is used to train a model and the testing dataset is used to evaluate the performance of the model. Our 10-fold cross-validation approach is repeated 10 times (100 iterations in total) to validate our results.

C. Preprocessing: In our study, we use three types of feature selection techniques (i.e., filter ranking-based techniques, filter subset-based techniques, and wrapper subset-based techniques) to preprocess the datasets. To apply the feature selection techniques to the datasets, we first divide the dataset into training and testing datasets. We preprocess *only the training dataset* using the feature selection technique, since the label of each instance would not be available in the testing dataset [68]. We do note that some prior studies have mistakenly applied feature selection techniques on the whole data set, which others have warned is a modeling violation [47, 74]. Incorrect application of feature selection techniques is one possible reason for the contradictory results across prior studies.

While subset-based techniques produce a subset of the original features, ranking-based techniques rank all features. To select the number of features from the list of ranked features, as recommended by prior work [34], we use $log_2 n$, where n is the total number of features. We train our models with the selected features using different types of classification techniques and evaluate the performance of trained classifica-

TABLE II: An overview of the studied classification techniques.

Family	Classification Technique	Description
Statistical	Naïve Bayes (NB)	*NB* trains probability-based models assuming that all the features are independent of each other.
	Bayesian network (BN)	*BN* produces a directed graph that is composed of V vertexes and E edges, where each feature is represented by a vertex and each edge represents a successor relationship within the network [51]. *BN* produces a conditional probability of a vertex using the values that are assigned to other vertices [9].
	Logistic Regression (Log)	*Log* is a regression model that is used to study the relationship between a categorical outcome variable and one or more features [52].
Neural Network	Radial Basis Function (RBF)	*RBF* [7] consists of three layers: input (the features), output (the dependent variable) and a middle layer, which connects the input and output layers [51].
	MultiLayer Perceptron (MLP)	*MLP* uses the back-propagation algorithm to train the model. *MLP* produces a directed graph consisting of multiple layers of nodes: an input layer, one or more hidden layers, and an output layer. In each subsequent layer, the output of the previous layer is used as input [51].
Decision Tree	Logistic Model Tree (LMT)	*LMT* [38] constructs a decision tree with logistic regression models to make the final decision at the leaves [38].
	Classification and Regression Tree (CART)	*CART* is a combination of both classification and regression trees. To obtain models, *CART* divides the dataset recursively to fit a simple predictive model within each partition [44].
	J48 (J48)	*J48* [57] is based on the C4.5 technique, which builds decision trees using information entropy. To divide the set of training samples into subsets, a rule is chosen by C4.5 at each node of the decision tree [62].
	Alternating Decision Trees (ADT)	*ADT* produces a tree structure with leaf and decision nodes. The outcomes are represented by leaf nodes, while the other nodes contain decision rules [51].
	Decision Stump (DS)	*DS* trains simple binary decision stumps (i.e., 1-level decision trees) for the classification of both nominal and numeric problems [73].
	Naive Bayes/Decision-Tree (NBT)	*NBT* is a combination of both naïve bayes and decision trees. NBTree produces a non-cost sensitive model, which consists of a naïve bayes model for each leaf [64].
	Random Tree (RT)	*RT* is also known as randomized C4.5 and uses a modified version of the C4.5 algorithm. To introduce the split at each internal node of the tree, the decision is randomized [71].
Rule Based	Repeated Incremental Pruning to Produce Error Reduction (Ripper)	*Ripper* [10] uses induction to remove the rules that result in lower classification performance by creating a series of rules with pruning.
	One Rule (OneR)	*OneR* uses one or more values from a single feature to build classification rules [47].
	Decision Table (DT)	*DT* is an extension of one-valued decision trees, which forms a rectangular table T, where features are represented in the columns and a set of decision rules are represented in the rows. To find a good subset of features, *DT*uses an inductive reduction algorithm to reduce the likelihood of over fitting the dataset and to eliminate equivalent rules [51].
	Partial Decision Trees (PART)	*PART* produces a decision list using the divide-and-conquer approach, building partial decision trees in each iteration and making the "best" leaf into a rule.
	Ripple Down Rules (Ridor)	*Ridor* [19] is a rule-based decision tree technique where the decision tree consists of four nodes: a classification, a predicate function, a true branch, and a false branch. Each instance of the testing dataset is pushed down the tree, following the true and false branches at each node using predicate functions. The final outcome is given by the majority class of the leaf nodes [39]
K-NN	K-Nearest Neighbour (KNN)	*KNN* classifies an instance on the basis of the K most similar training examples. To calculate the distance between instances, KNN uses Euclidean distance [40]. We use K = 10 as recommended by a previous study [74].
	Kstar (Kstar)	*Kstar* works similar to KNN, however Kstar uses an entropy-based function to calculate the distance between instances [40].
Support Vector Machine	Voted Perceptron (VP)	*VP* uses the information of all the classification vectors that are stored during the training phase to generate a better classification on the test dataset [37].
Ensemble Method	Random Forest (RandF)	*RandF* [6] trains an ensemble of decision trees using a randomly selected subspaces of the dataset [3].

tion models.

D. Model Construction: In order to measure the impact of feature selection techniques on the performance of defect classification models, we train our models using 21 classification techniques as described in Table II on the preprocessed and original datasets.

TABLE III: An overview of the studied corpora.

Corpus	Dataset	No. of Features	No. of Modules	Defective	Defective (%)
Clean NASA	CM1	37	327	42	12.8
	KC1	21	1,162	294	25.3
	KC3	39	194	36	18.5
	MW1	37	250	25	10
	PC1	37	679	55	8.1
	PC2	37	722	16	2.2
	PC3	37	1,053	130	12.3
	PC4	37	1,270	176	13.8
PROMISE	Ant 1.7	20	745	166	22.3
	Camel 1.6	20	965	188	19.5
	Ivy 1.4	20	241	16	6.6
	Jedit 4.0	20	306	75	24.5
	Log4j 1.0	20	135	34	25.2
	Lucene 2.4	20	340	203	59.7
	Poi 3.0	20	442	281	63.6
	Tomcat 6.0	20	858	77	8.9
	Xalan 2.6	20	885	441	46.4
	Xerces 1.3	20	453	69	15.2

We selected the most commonly used classification techniques in prior defect classification studies [22, 40, 70]. We use the implementations of the studied classification techniques, as provided by the WEKA [72] machine learning toolkit. In this study, we combine 21 classification techniques with 30 feature selection techniques (including a no feature selection configuration) to train our models on 18 datasets ($21 \times 30 \times 18 \times 100$ folds = 1,134,000 folds).

E. Model Evaluation: To compare the performance of defect classification models, we use the Area Under the receiver operating characteristic Curve (AUC) because it is threshold-independent. AUC plots the false positive rate (i.e., $\frac{FP}{FP+TN}$) against the true positive rate (i.e., $\frac{TP}{FN+TP}$). Larger AUC values indicate better performance. AUC values above 0.5 indicate that a model outperforms random guessing.

Statistical Comparison Tests: The Scott-Knott [30] test produces statistically distinct ranks ($\alpha = 0.05$) of classification techniques. The Scott-Knott test uses hierarchical cluster analysis to partition the classification techniques into ranks. The Scott-Knott test starts by dividing the classification techniques into two ranks on the basis of mean AUC values (i.e., mean AUC of the 10×10-fold cross-validation runs for each classification technique per pre-processor). If the divided ranks are statistically significantly different, then Scott-Knott recursively executes again within each rank to further divide the ranks. The test terminates when ranks can no longer be divided into statistically distinct ranks [32, 48].

In this study, instead of using the traditional Scott-Knott test, we use the extended Scott-Knott with effect size awareness (Scott-KnottESD) as suggested by Tantithamthavorn [69]. The extended Scott-Knott test post-processes the statistically

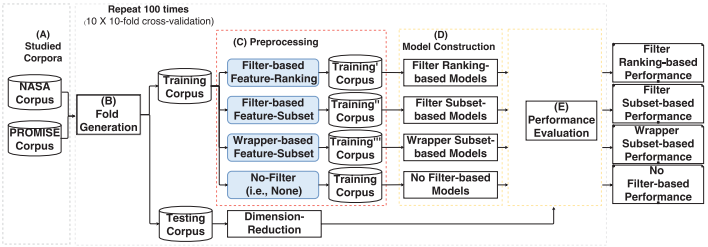

Fig. 1: An overview of our model construction and evaluation approach.

distinct ranks that are produced by the traditional Scott-Knott test, while merging any pair of ranks that have a negligible Cohen's d effect size between them [69].

IV. A Preliminary Comparison of the NASA & PROMISE Datasets

In this section, we present a preliminary comparison of the dataset characteristics of the NASA and PROMISE corpora. An overview of the features in the studied corpora is shown in Table IV.

A. Dataset Analysis

We conduct a dataset redundancy characteristics analysis prior to studying the impact of feature selection techniques on defect classification models since:

1) The datasets within the NASA and PROMISE corpora consist of different types of features. For example, the NASA corpus has measurements of size (LOC) and complexity (i.e., McCabe and Halstead), whereas in addition to size and complexity measures, the PROMISE corpus contains OO metrics like coupling and cohesion.
2) The datasets within the NASA corpus consist of features at the module-level, while the PROMISE corpus datasets consist of features at the class-level.
3) The datasets within NASA (i.e., proprietary) and PROMISE (i.e., open source) corpora developed in different settings.

The datasets of both corpora have data redundancy. We use Principal Component Analysis (PCA) to find how many orthogonal components are needed to account for 95% of the data variance in the datasets of each corpus.

The NASA and PROMISE datasets have different data redundancy characteristics. To account for 95% of the data variance, the NASA datasets need an average of 31% of the components while the PROMISE datasets need an average of 59% of the components to account 95% of the data variance. Indeed, the PCA results are very consistent among datasets within each corpus, but very different when comparing datasets across the corpora. There is consistency in the percentage of components that are needed to account for the different levels of data variance among the datasets of each corpus (i.e., NASA and PROMISE) as shown in Table V.

Our PCA analysis shows that the NASA and PROMISE datasets are quite different from each other in the context of data variance information in their feature sets. Figure 2 shows the PCA analysis of the CM1 NASA-dataset and the Ant PROMISE-dataset. For CM1, the majority of the data variance information is contained in the first principal component while for Ant, the data variance information is more proportionally divided across the different principal components. Hence, future studies should consider the much richer nature of the PROMISE features versus the NASA features.

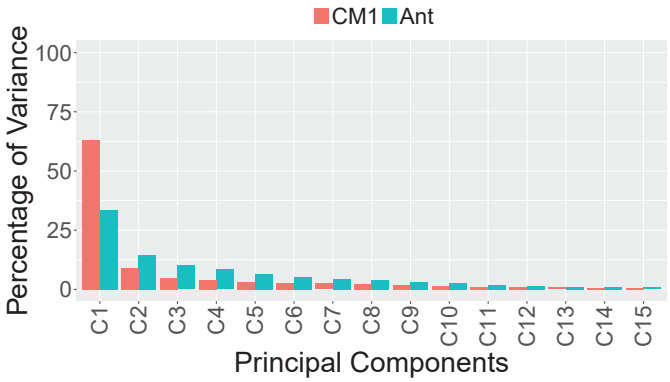

Fig. 2: PCA for the CM1 NASA dataset and Ant PROMISE dataset.

The features in the PROMISE corpus are much richer than the NASA features which are very redundant, highlighting the simplistic nature of the NASA corpus and the importance of examining the NASA corpus separately.

V. Case Study Results

In this section, we present the results of our case study, which addresses our research question:

RQ: Do feature selection techniques impact the performance of defect classification models?

A. Approach

To address our research question, we start with the AUC values of the 21 studied classification techniques that are trained using the preprocessed and non-preprocessed datasets. We use the Scott-Knott test to produce statistically distinct ranks of the 30 feature selection techniques.

1) Dataset level: In order to study the impact of feature selection techniques on the performance of defect classification models in each dataset, we calculate the Scott-Knott rank of

TABLE IV: Features in the studied corpora.

Corpus	Category	Feature	Definition	Rationale
NASA	McCabe	cyclomatic_complexity, cyclomatic_density, design_complexity essential complexity and pathological_complexity	Measures of the branching complexity of the software module.	Complex software modules may be more prone to defects [46].
	Halstead	content, difficulty, effort, error_est, length, level, prog_time, volume, num_operands, num_operators, num_unique_operands, num_unique_operators	Estimates of the complexity of reading a software module based on the used vocabulary (e.g., number of operators and operands).	Software modules that are complex to understand may increase the likelihood of incorrect maintenance, and thus, increase defect proneness [31].
	Size	LOC_total, LOC_blank, LOC_comment, LOC_code_and_comment, LOC_executable and Number_of_lines	Measures of the size of a software module.	Larger software modules may be difficult to understand entirely, and thus, may be more prone to defects [36].
	Miscellaneous	branch_count, call_pairs, condition_count, decision_count, decision_density, design_density, edge_count, essential_density, parameter_count, maintenance_severity, modified_condition_count, multiple_condition_count, global_data_density, global_data_complexity, percent_comments, normalized_cyclomatic_complexity and node_count	Metrics that are not well defined metrics in the MDP database [47].	N/A
PROMISE	CK	amc(average method complexity), cbm(coupling between methods), cbo(coupling between objects), dit(depth of inheritance tree), ic(inheritance coupling), lcom(lack of cohesion in methods), lcom3(another lack of cohesion measure), noc(number of children), rfc(response for a class), and wmc(weighted methods per class)	Measures of the complexity of a class within an object-oriented system design.	More complex classes may be more prone to defects [23].
	McCabe	avg_cc(average mccabe) and max_cc(maximum mccabe)	Measures of the branching complexity of the software module	Complex software modules may be more prone to defects [46].
	QMOOD	cam(cohesion among classes), dam(data access), mfa(functional abstraction), moa(aggregation), and npm(number of public methods)	Measures of the behavioural and structural design properties of classes, objects, and the relations between them [63].	Higher QMOOD metrics imply higher class quality.On the other hand, lower QMOOD metrics imply lower class quality, which may lead to defects [33].
	Size	loc(lines of code)	Measures of the size of a software module	Larger software modules may be difficult to understand entirely, and thus, may be more prone to defects [36].
	Martin	ca(afferent couplings) and ce(efferent couplings)	Measures of the relationship between software components, including calls as well as number of instances [50].	Higher values indicate low encapsulation,reusability, and maintainability, which may lead to defects [50].

TABLE V: Number of components that are needed to account for 95% of the data variance.

Corpus	Dataset	Total Components	95% Variance (No. of Components)	95% Variance (No. of Components %)
NASA	CM1	37	11	30
	KC1	21	7	33
	KC3	39	10	26
	MW1	37	11	30
	PC1	37	12	32
	PC2	36	10	28
	PC3	37	12	32
	PC4	37	14	38
PROMISE	Ant	20	12	60
	Camel	20	12	60
	Ivy	20	10	50
	Jedit	20	12	60
	Log4j	20	12	60
	Lucene	20	12	60
	Poi	20	12	60
	Tomcat	20	12	60
	Xalan	20	12	60
	Xerces	20	12	60

each feature selection technique across all of the classification techniques. For each dataset, we feed the AUC values of the 10×10 fold cross-validation iterations of 21 classification techniques after applying and without applying the feature selection technique to the Scott-Knott test. The Scott-Knott test results in statistically distinct ranks of 30 feature selection techniques for each dataset i.e., each feature selection technique ends up with eight (i.e., eight datasets studied) and 10 (i.e., 10 datasets studied) Scott-Knott ranks in the NASA and PROMISE corpora, respectively. We then calculate the number of datasets in which each feature selection technique appears in the top-rank.

2) Classification-Technique level: In order to find the impact of feature selection techniques, we feed the AUC values of the 10×10 fold cross-validation iterations of each classification technique after applying and without applying the feature selection technique across all of the datasets to the Scott-Knott test. After the Scott-Knott test run, each feature selection technique has 21 ranks (i.e., one for each of the studied classification technique). We calculate the number of classification techniques in which each feature selection technique appears in the top-rank.

B. Results

NASA: Classification-Technique level

FS1 outperforms other feature selection techniques, appearing in top Scott-Knott rank for 90% of the studied classification techniques. Figure 3 shows the mean AUC values of the 30 feature selection techniques for each classification technique across the NASA corpus. Figure 4 shows the Scott-Knott rank of each feature selection technique across the 21 studied classification techniques, FS1 appears in top Scott-Knott rank for 19 of the 21 classification techniques. In addition to FS1, ranking-based techniques (i.e., FR2, FR11), filter subset-based techniques (i.e., FS2 and FS3), and wrapper-based technique WS8 outperform other feature selection techniques, appearing in the top Scott-Knott rank for >80% of the classification techniques. FR3 is the worst performing feature selection technique, appearing in the top Scott-Knott rank for only the CART classification technique.

Our results show that the impact of feature selection techniques also varies across different classification techniques. For CART, all of the 30 feature selection techniques (including a no feature selection configuration) appear in the same statistical rank i.e., CART is not sensitive to feature selection techniques. On the other hand, there is only one feature selection technique (i.e., FR11) that appears in the top Scott-Knott rank for VP.

Furthermore, we find that for RandF, filter-based and wrapper-based subset feature selection techniques outperform filter-based ranking ones. This agrees with the findings of Xu *et al.* [74].

NASA: Dataset level

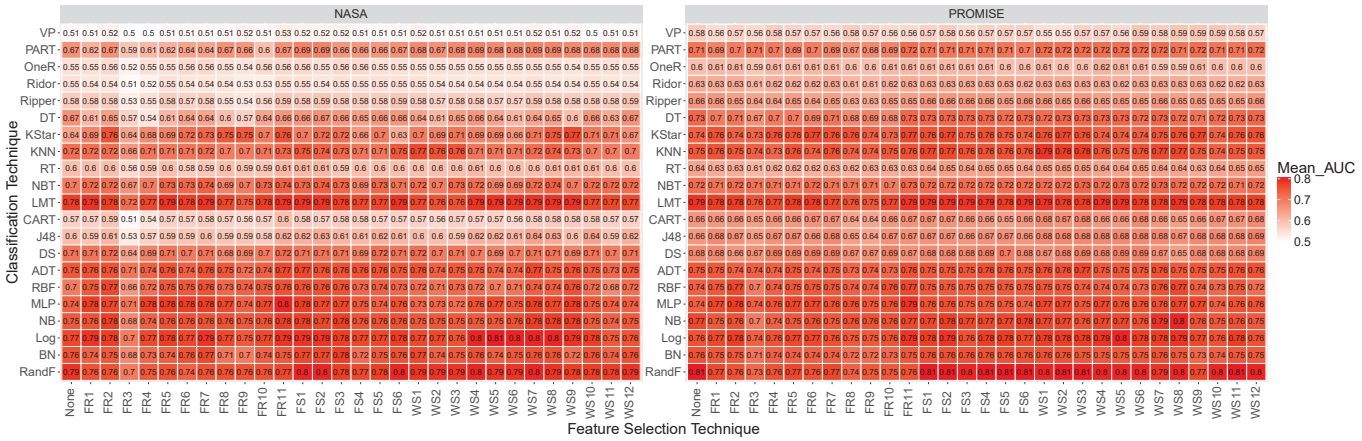

Fig. 3: Mean AUC value of each feature selection technique for each classification technique across all datasets in the NASA and PROMISE corpora.

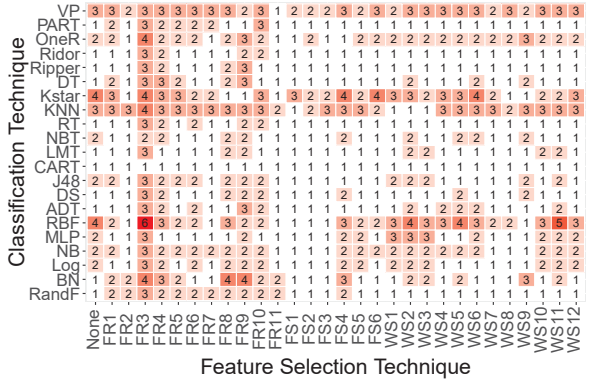

Fig. 4: Scott-Knott rank of each feature selection technique for each classification technique across all the eight NASA datasets. Darker colors indicate lower Scott-Knott rank.

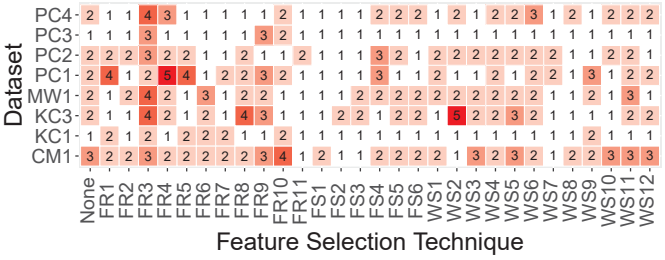

Fig. 5: Scott-Knott rank of each feature selection technique for each NASA dataset across all the 21 classification techniques. Darker colors indicate lower Scott-Knott rank.

FR11, FS1, and FS2 appear in the top Scott-Knott rank for 87% of the studied datasets. Figure 5 shows the Scott-Knott rank of each feature selection technique for each dataset. We find that the filter-based feature subset techniques (i.e., FS1 and FS2) and filter-based feature ranking technique FR11 appears in the top Scott-Knott rank in seven of the eight studied datasets. On the other hand, None, FR4, WS4, WS6, and WS11 are performing worse appearing in the top Scott-Knott rank for only two datasets.

Furthermore, the impact of feature selection techniques varies across the datasets. For dataset CM1, there are only five feature selection techniques that appear in the top Scott-Knott rank, while for dataset PC3, 27 feature selection techniques appear in the top Scott-Knott rank. Figure 6 highlights the AUC values distribution of the 10×10 fold cross-validation iterations of each classification technique after applying feature selection techniques for each dataset.

The filter-based feature subset technique FS1 outperforms other feature selection techniques, appearing in the top Scott-Knott rank for 87% of the studied datasets and 90% of the studied classification techniques.

PROMISE: Classification-Technique level

FS1 and FS2 techniques outperform other feature selection techniques, appearing in the top Scott-Knott rank for 100% of the studied classification techniques. The mean AUC values of the 30 feature selection techniques for each classification technique across the PROMISE corpus is shown in Figure 3. Furthermore, Figure 7 shows the Scott-Knott rank of each feature selection techniques across all the classification techniques. For all of the 21 studied classification techniques, FS1 and FS2 appear in the top Scott-Knott rank. Similar to the NASA corpus, FR3 appears in the top Scott-Knott rank for least number of classification techniques (i.e., eight only). Furthermore, similar to the NASA corpus, our results support the findings of recent work by Xu *et al.* [74], i.e., for RandF, filter-based and wrapper-based subset feature selection techniques outperform ranking-based ones. However, Xu *et al.* [74] study is only limited to the impact of feature selection techniques on the performance of *Random Forest*.

Our results show a similar pattern to the NASA corpus, the impact of feature selection techniques varies across different classification techniques. First, we find that all of the feature selection techniques appear in the same Scott-Knott rank for CART, confirming that CART is not sensitive to the feature

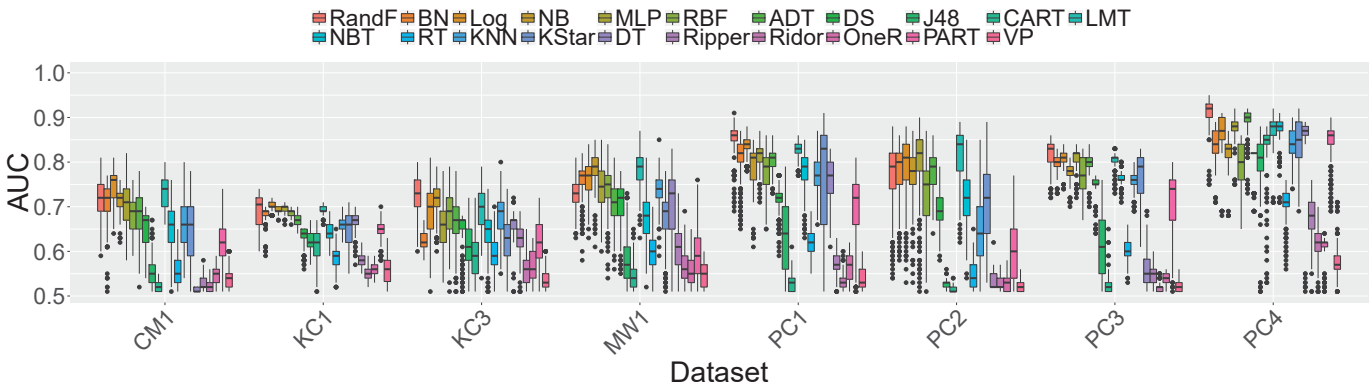

Fig. 6: The AUC values of each dataset for each classification technique across all the feature selection techniques in the NASA corpus.

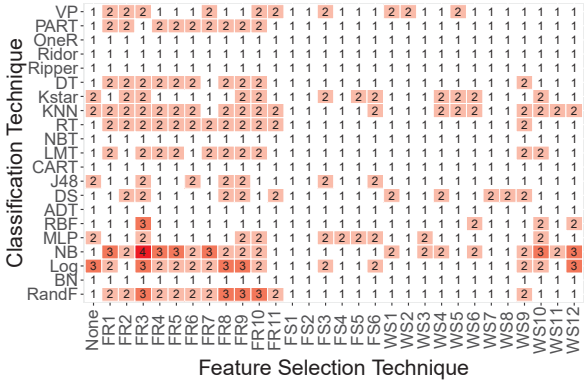

Fig. 7: Scott-Knott rank of each feature selection technique for each classification technique across all the 10 PROMISE datasets. Darker colors indicate lower Scott-Knott rank.

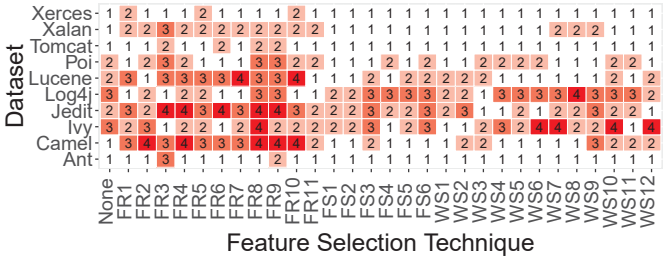

Fig. 8: Scott-Knott rank of each feature selection technique for each PROMISE dataset across all the 21 classification techniques. Darker colors indicate lower Scott-Knott rank.

selection configuration across both NASA and PROMISE corpora. In addition to CART, BN, ADT, NBT, Ripper, Ridor, and OneR are not sensitive to the feature selection techniques.

In summary, the FS1 feature selection technique consistently outperforms other feature selection techniques across different classification techniques and across different corpora.

PROMISE: Dataset level

FS1, FS2, FS4, WS1, WS4, and WS6 feature selection techniques appear in the top Scott-Knott rank for 70% of the studied datasets. Figure 8 shows the Scott-Knott rank of each feature selection technique across the datasets. Filter-based subset techniques (FS1, FS2, and FS4) and Wrapper-based subset techniques (WS1, WS4, and WS6) outperform other feature selection techniques, appearing in the top Scott-Knott rank for seven of the 10 studied datasets. Filter-based ranking technique FR9 appears in the top Scott-Knott rank for only one dataset.

Similar to the NASA corpus, our results show that the impact of feature selection techniques varies across the datasets. For Ant and Xerces datasets, 28 feature selection techniques appear in the same statistical rank (i.e., top Scott-Knott rank) while for Jedit there are only four feature selection techniques that appear in the top Scott-Knott rank. The AUC values distribution of the 10×10 fold cross-validation iterations of each classification technique after applying feature selection techniques for each dataset are shown in Figure 9.

Similar to our results of the NASA corpus, FS1 appears in the top Scott-rank for 70% of the studied datasets and 100% of the studied classification techniques in the PROMISE corpus.

VI. Threats to Validity

We now discuss the threats to the validity of our case study.

Construct Validity: We used datasets from two different corpora (NASA and PROMISE) to train our classification models. The corpora that are used consist of different sets of features, which may explain the differences that we observe when drawing conclusions across the corpora. Nevertheless, our main findings still hold on both corpora, i.e., filter-based subset feature selection technique FS1 outperforming other feature selection techniques ending in the top Scott-Knott rank for the majority of the datasets (70% and 87% of the studied datasets for PROMISE and NASA corpora, respectively) and classification techniques (90% and 100% of the studied classification techniques for NASA and PROMISE corpora, respectively) used.

Internal Validity: We apply the feature selection techniques to the studied classification techniques with default parameter

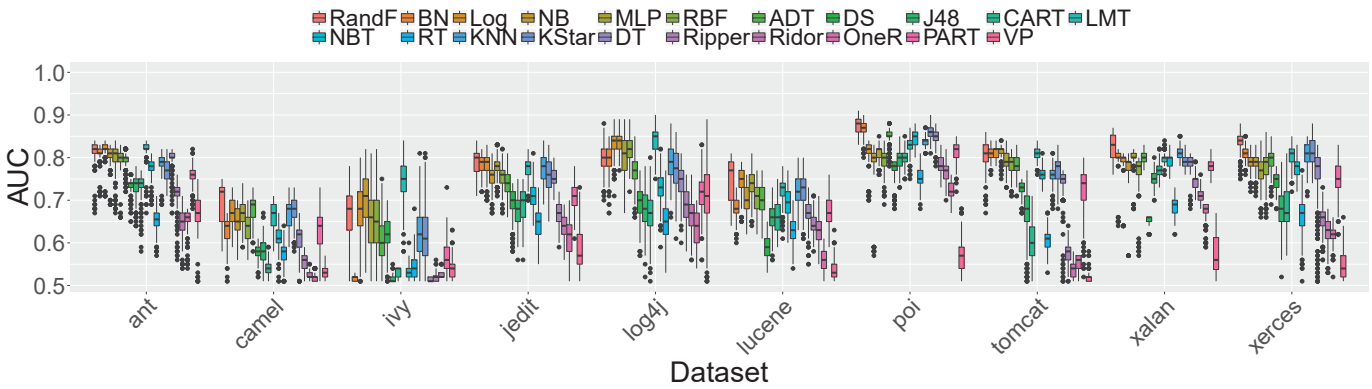

Fig. 9: The AUC values of each dataset for each classification technique across all the feature selection techniques in the PROMISE corpus.

settings except for KNN. We use $K = 10$ for KNN for parameter tuning as recommended by a previous study [74]. However, only two (i.e., CART and MLP) out of the 21 used classification techniques have a large impact of parameter tuning as shown in a previous study [70], i.e., the majority of the classification techniques used in our study are not sensitive to parameter tuning. Furthermore, we use the AUC to compare the performance of our models. Other performance measures may yield different results. Future studies should examine classification techniques after applying parameter tuning.

External Validity: We performed our experiments on 18 datasets, which may threaten the generalizability of our study results. On the other hand, these datasets are part of the two most commonly studied corpora (NASA and PROMISE) and have been analyzed by several defect classification studies in the past [40, 51, 53]. We use 30 feature selection techniques (11 filter-based ranking techniques, six filter-based subset techniques, 12 wrapper-based subset techniques, and a no feature selection configuration) and applied them to 21 different classification techniques to train our classification models. Other feature selection techniques may yield different results. However, we include the feature selection techniques that are commonly used in prior studies [21, 59, 61, 74] and applied them to a variety of classification techniques that have also frequently used in past defect classification studies [21, 22, 27, 40, 59, 61, 70, 74]. Furthermore, we use different types of feature selection techniques within the selected three categories — filter ranking-based (i.e., statistics-based, probability-based, instance-based, and classifier-based), filter subset-based (i.e., Correlation-based and Consistency-based), and wrapper-based (i.e., use different four classifiers). Nonetheless, additional replication studies may prove fruitful.

VII. Related Work

Plenty of recent research has examined the high dimensionality problem of datasets using feature selection techniques.

Hall *et al.* [27] find that wrapper-based techniques outperform other techniques, but with a high computational cost. Rodrìguez *et al.* [60] apply three filter and two wrapper-base feature selection techniques. They conclude that the smaller datasets maintain the prediction capability with a lower number of features than the original datasets. In addition, wrapper-based feature selection techniques outperform others. Menzies *et al.* [47] apply the InfoGain feature ranking technique in order to rank features and compare the performance of classification models that are trained using three classification techniques (i.e., NB, J48, and OneR). They conclude that Naïve Bayes yields the best classification model.

Muthukumaran *et al.* [49] studied ten feature selection techniques (seven ranking-based techniques and three subset-based techniques) and find that feature selection techniques tend to improve classification accuracy. The results of their study show that subset-based feature selection techniques outperform other techniques [49]. Rathore *et al.* [59] conducted a study with fifteen different feature selection techniques that consist of ranking-based and subset-based techniques. They find that InfoGain and PCA ranking-based techniques tend to outperform other ranking-based techniques, whereas ClassifierSubsetEval and Logistic Regression outperform other subset-based techniques.

Xu *et al.* [74] conducted a study to compare the impact of 32 feature selection techniques on the performance of defect classification models that are trained using the Random Forest classifier on the three corpora (i.e., noisy NASA, clean NASA, and open source AEEEM). They found a significant difference in the impact of feature selection techniques. Furthermore, their results show that filter and wrapper subset-based techniques outperform other techniques. We select 24 of 32 feature selection techniques used in Xu *et al.* [74] study. However, the scope of Xu *et al.* [74] study limits to a single classification technique (i.e., Random Forest). However, in order to generalize the impact of feature selection techniques, we select the 21 most commonly used classification techniques.

Table VI provides an overview of the prior work. The limitations of the studies discussed above are:

1) The impact of feature selection techniques is studied using a limited number of classification techniques. Hall *et al.* [27], Rodrìguez *et al.* [60], and Rathore *et al.* [59] use only two classification techniques. On the other hand,

TABLE VI: Comparison of our study to the previous studies

Study	Corpus used	Classification Techniques	Feature Selection Techniques	Feature Selection type	Main Findings
Hall *et al.* [27]	Corpus: UCI repository [41] No. of datasets: 19	No. of techniques: 2 Techniques: C4.5 and NB	No. of techniques: 6	Filter feature ranking: 3 Filter feature subset: 2 Wrapper Subset: 1	Yes, there is an impact. Wrapper-based subset techniques outperform others.
Rodr\`{i}guez *et al.* [60]	Corpus: PROMISE [66] No. of datasets: 5	No. of techniques: 2 Techniques: C4.5 and NB	No. of techniques: 5	Filter feature ranking: 0 Filter feature subset: 3 Wrapper Subset: 2	Yes, there is an impact. Wrapper-based subset techniques outperform others but it is more computationally expensive.
Muthukumaran *et al.* [49]	Corpus: NASA [65] No. of datasets: 11 Corpus: AEEEM [12] No. of datasets: 5	No. of techniques: 3 Techniques: NB, RandF, and Log	No. of techniques: 10	Filter feature ranking: 7 Filter feature subset: 2 Wrapper Subset: 1	Yes, there is an impact. Filter feature subset outperforms other techniques.
Rathore *et al.* [59]	Corpus: PROMISE [1] No. of datasets: 4	No. of Techniques: 2 Techniques: NB and RandF	No. of techniques:15	Filter feature ranking: 7 Filter feature subset: 6 Wrapper Subset: 1	Yes, there is an impact. Feature ranking techniques outperform other techniques.
Xu *et al.* [74]	Corpus: NASA [65] No. of datasets: 11 Corpus: AEEEM [12] No. of datasets: 5	No. of Techniques: 1 Techniques: RandF	No. of techniques:32	Filter feature ranking: 14 Filter feature subset: 2 Wrapper Subset: 12 Other (Clustering-based and PCA): 4	Yes, there is an impact. Filter-based and wrapper-based feature subset evaluation methods can achieve the best performance.
Menzies *et al.* [47]	Corpus: NASA [65] No. of datasets: 8	No. of Techniques: 1 Techniques: OneR, J48, and NB	No. of techniques: 1	Filter feature ranking: 1 Filter feature subset: 0 Wrapper Subset: 0	Yes, there is an impact. Naive Bayes produces the best classification model.
Our study	Corpus: NASA No. of datasets: 8 Corpus: PROMISE No. of datasets: 10	No. of techniques: 21 Techniques: RandF, BN, Log, NB, MLP, RBF, ADT, DS, J48, CART, LMT, NBT, RT, KNN, Kstar, DT, Ripper, Ridor, OneR, PART, and VP	No. of techniques: 29	Filter feature ranking: 11 Filter feature subset: 6 Wrapper Subset: 12	Yes, there is an impact. Filter-based subset techniques outperform other feature selection techniques across the projects and across the classification techniques.

Muthukumaran *et al.* [49] use three classification techniques to study the impact of feature selection techniques, while Xu *et al.* [74] only study the impact on Random Forest. Since there are some classification techniques that outperform others [22], it is interesting to examine the impact of feature selection techniques on the performance of different classification techniques. Therefore, in this paper, we study the impact of feature selection techniques on the performance of most commonly used classification techniques in prior defect classification studies [22, 40, 70]

2) Studies conducted by Hall *et al.* [27], Rodrìguez *et al.* [60], and Rathore *et al.* [59] use only datasets from one domain. Since the characteristics of dataset in some specific domains influence the impact feature selection techniques as discussed in Section IV, it is important to evaluate the impact on different types of datasets.

Table VI shows that recent studies have concluded that feature selection techniques have an impact on the performance of defect classification models. However, the results of these studies are inconsistent with each other, since the studies were conducted under different conditions, e.g., datasets of different domains, using different classification and feature selection techniques. Hall *et al.* [27] and Rodrìguez *et al.* [60] show that wrapper-based feature subset techniques outperform filter-based feature ranking and feature subset techniques, whereas Muthukumaran *et al.* [49] show that filter-based feature subset techniques outperform filter-based feature ranking and wrapper-based techniques. On the other hand, Xu *et al.* [74] show that filter-based subset techniques and wrapper-based techniques outperform filter-based feature ranking techniques. Furthermore, Rathore *et al.* [59] show that filter-based feature ranking techniques outperform filter-based feature subset and wrapper-based techniques.

To address the inconsistency of the previous studies result, we performed a larger scale study to analyze the impact of 30 feature selection techniques (including a no feature configuration) on the performance of 21 classification techniques using 18 datasets from the NASA and PROMISE corpora.

VIII. Conclusions

In this study, we apply 30 feature selection techniques (11 filter-based ranking techniques, six filter-based subset techniques, 12 wrapper-based subset techniques, and a no feature selection configuration) to analyze their impact on the performance of defect classification models that are trained using 21 classification techniques on the NASA and PROMISE corpora. Our results show that:

1) FS1(correlation-based filter-subset technique with the BestFirst search method) feature selection technique outperforms other feature selection techniques, appearing in the top Scott-Knott rank for 70%-87% of the studied datasets in the PROMISE-NASA corpus.
2) FS1 outperforms other feature selection techniques, appearing in the top Scott-Knott rank for 90%-100% of the studied classification techniques in the NASA-PROMISE.
3) The impact of feature selection techniques varies across the datasets, i.e., few feature selection techniques appear in the top Scott-Knott rank of one dataset, whereas for other datasets, there is no statistical difference among the studied feature selection techniques.
4) The impact of feature selection techniques also varies across the classification techniques. There is a clear statistical difference in the impact of feature selection techniques when applied to some of the studied classification techniques, whereas we find that classification techniques like CART are not sensitive to feature selection techniques, i.e., all of the feature selection techniques appear in the same statistical rank across the NASA and PROMISE corpora.

We would like to emphasize that we do not seek to claim the generalization of our results on other datasets. However, we conclude that the choice of feature selection technique (i.e., ranking-based or subset-based) has an impact on the performance classification models. FS1 outperforms other feature selection techniques across different datasets, corpora, and classification techniques. Hence, we recommend that future defect classification studies should consider applying the FS1 (correlation-based) feature selection technique.

REFERENCES

[1] tera-promise repository:. http://openscience.us/repo/.

[2] A. Ahmad and L. Dey. A feature selection technique for classificatory analysis. *Pattern Recognition Letters*, 26(1):43–56, 2005.

[3] G. Biau. Analysis of a random forests model. *The Journal of Machine Learning Research*, 13(1):1063–1095, 2012.

[4] A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1):245–271, 1997.

[5] J. Brank, M. Grobelnik, N. Milic-Frayling, and D. Mladenic. Feature selection using linear support vector machines. In *Proceedings of the third international conference on data mining methods and databases for engineering, finance and other fields. Bologna, Italy*, 2002.

[6] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[7] M. D. Buhmann. Radial basis functions. *Acta Numerica 2000*, 9:1–38, 2000.

[8] T. Chai and R. R. Draxler. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific Model Development*, 7(3):1247–1250, 2014.

[9] J. Cheng and R. Greiner. Comparing bayesian network classifiers. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 101–108. Morgan Kaufmann Publishers Inc., 1999.

[10] W. W. Cohen. Fast Effective Rule Induction. In *Proceedings of the International Conference on Machine Learning*, pages 115–123, 1995.

[11] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[12] M. D'Ambros, M. Lanza, and R. Robbes. An extensive comparison of bug prediction approaches. In *Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on*, pages 31–41. IEEE, 2010.

[13] M. Dash and H. Liu. Consistency-based search in feature selection. *Artificial intelligence*, 151(1):155–176, 2003.

[14] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM, 2006.

[15] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[16] N. E. Fenton and M. Neil. A critique of software defect prediction models. *Software Engineering, IEEE Transactions on*, 25(5):675–689, 1999.

[17] S. Fong, J. Liang, R. Wong, and M. Ghanavati. A novel feature selection by clustering coefficients of variations. In *Digital Information Management (ICDIM), 2014 Ninth International Conference on*, pages 205–213. IEEE, 2014.

[18] S. Fong, J. Liang, and Y. Zhuang. Improving classification accuracy using fuzzy clustering coefficients of variations (fccv) feature selection algorithm. In *Computational Intelligence and Informatics (CINTI), 2014 IEEE 15th International Symposium on*, pages 147–151. IEEE, 2014.

[19] B. R. Gaines and P. Compton. Induction of ripple-down rules applied to modeling large databases. *Journal of Intelligent Information Systems*, 5(3):211–228, 1995.

[20] K. Gao, T. M. Khoshgoftaar, and N. Seliya. Predicting high-risk program modules by selecting the right software measurements. *Software Quality Journal*, 20(1):3–42, 2012.

[21] K. Gao, T. M. Khoshgoftaar, H. Wang, and N. Seliya. Choosing software metrics for defect prediction: an investigation on feature selection techniques. *Software: Practice and Experience*, 41(5):579–606, 2011.

[22] B. Ghotra, S. McIntosh, and A. E. Hassan. Revisiting the impact of classification techniques on the performance of defect prediction models. In *Proceedings of the 37th International Conference on Software Engineering-Volume 1*, pages 789–800. IEEE Press, 2015.

[23] B. Goel and Y. Singh. Empirical investigation of metrics for fault prediction on object-oriented software. In *Computer and Information Science*, pages 255–265. Springer, 2008.

[24] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[25] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[26] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.

[27] M. Hall, G. Holmes, et al. Benchmarking attribute selection techniques for discrete class data mining. *Knowledge and Data Engineering, IEEE Transactions on*, 15(6):1437–1447, 2003.

[28] M. A. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.

[29] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1):63–90, 1993.

[30] E. Jelihovschi, J. C. Faria, and I. B. Allaman. Scottknott: A package for performing the scott-knott clustering algorithm in r. *Trends in Applied and Computational Mathematics*, 15(1):003–017, 2014.

[31] K. Kaur, K. Minhas, N. Mehan, and N. Kakkar. Static and dynamic complexity analysis of software metrics. *World Academy of Science, Engineering and Technology*, 56:2009, 2009.

[32] H. Khalid, M. Nagappan, E. Shihab, and A. E. Hassan. Prioritizing the devices to test your app on: a case study of android game apps. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 610–620. ACM, 2014.

[33] Y. Khan and O. Khararah. A systematic review on the relationships between mood/qmood metrics and external software quality attributes. Technical report, Technical report, Department of Information and Computer Science, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, 2014.

[34] T. M. Khoshgoftaar, M. Golawala, and J. V. Hulse. An empirical study of learning from imbalanced data using random forest. In *Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE International Conference on*, volume 2, pages 310–317. IEEE, 2007.

[35] I. Kononenko. Estimating attributes: analysis and extensions of relief. In *European conference on machine learning*, pages 171–182. Springer, 1994.

[36] A. G. Koru, D. Zhang, K. El Emam, and H. Liu. An investigation into the functional form of the size-defect relationship for software modules. *IEEE Transactions on Software Engineering*, 35(2):293–304, 2009.

[37] S. Kotsiantis, K. Patriarcheas, and M. Xenos. A combinational incremental ensemble of classifiers as a technique for predicting students′ performance in distance education. *Knowledge-Based Systems*, 23(6):529–535, 2010.

[38] N. Landwehr, M. Hall, and E. Frank. Logistic model trees. *Machine Learning*, 59(1-2):161–205, 2005.

[39] B. Lemon. *The Effect of Locality Based Learning on Software Defect Prediction*. PhD thesis, West Virginia University, 2010.

[40] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch. Benchmarking classification models for software defect prediction: A proposed framework and novel findings. *IEEE Transactions on Software Engineering*, 34(4):485–496, 2008.

[41] M. Lichman. UCI machine learning repository, 2013.

[42] H. Liu and H. Motoda. *Feature Extraction, Construction and Selection: A Data Mining Perspective*, volume 453. Springer

Science & Business Media, 2012.

[43] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In *ICTAI*, pages 388–391, 1995.

[44] W.-Y. Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):14–23, 2011.

[45] H. Lu, B. Cukic, and M. Culp. Software defect prediction using semi-supervised learning with dimension reduction. In *Automated Software Engineering (ASE), 2012 Proceedings of the 27th IEEE/ACM International Conference on*, pages 314–317. IEEE, 2012.

[46] T. Menzies, J. S. Di Stefano, M. Chapman, and K. McGill. Metrics that matter. In *Proceedings of the 27th Annual NASA Goddard Software Engineering Workshop (SEW-27'02)*, page 51. IEEE Computer Society, 2002.

[47] T. Menzies, J. Greenwald, and A. Frank. Data mining static code attributes to learn defect predictors. *IEEE Transactions on Software Engineering*, 33(1):2–13, 2007.

[48] N. Mittas and L. Angelis. Ranking and clustering software cost estimation models through a multiple comparisons algorithm. *IEEE Transactions on Software Engineering*, 39(4):537–551, 2013.

[49] K. Muthukumaran, A. Rallapalli, and N. Murthy. Impact of feature selection techniques on bug prediction models. In *Proceedings of the 8th India Software Engineering Conference*, pages 120–129. ACM, 2015.

[50] M. F. Oliveira, R. M. Redin, L. Carro, L. d. C. Lamb, and F. R. Wagner. Software quality metrics and their impact on embedded software. In *Proceedings of the 5th International Workshop on Model-based Methodologies for Pervasive and Embedded Software, 2008*, pages 68–77. IEEE Computer Society, 2008.

[51] A. Panichella, R. Oliveto, and A. De Lucia. Cross-project defect prediction models: L'union fait la force. In *Proceedings of the Software Evolution Week-IEEE Conference on Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE)*, pages 164–173. IEEE, 2014.

[52] C.-Y. J. Peng and T.-S. H. So. Logistic regression analysis and reporting: A primer. *Understanding Statistics: Statistical Issues in Psychology, Education, and the Social Sciences*, 1(1):31–70, 2002.

[53] F. Peters, T. Menzies, and A. Marcus. Better cross company defect prediction. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 409–418. IEEE Press, 2013.

[54] S. L. Pfleeger. Software metrics: progress after 25 years? *IEEE Software*, (6):32–34, 2008.

[55] R. L. Plackett. Karl pearson and the chi-squared test. *International Statistical Review/Revue Internationale de Statistique*, pages 59–72, 1983.

[56] D. Pyle. *Data preparation for data mining*, volume 1. Morgan Kaufmann, 1999.

[57] J. R. Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[58] J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[59] S. S. Rathore and A. Gupta. A comparative study of feature-ranking and feature-subset selection techniques for improved fault prediction. In *Proceedings of the 7th India Software Engineering Conference*, page 7. ACM, 2014.

[60] D. Rodríguez, R. Ruiz, J. Cuadrado-Gallego, and J. Aguilar-Ruiz. Detecting fault modules applying feature selection to classifiers. In *Information Reuse and Integration, 2007. IRI 2007. IEEE International Conference on*, pages 667–672. IEEE, 2007.

[61] D. Rodríguez, R. Ruiz, J. Cuadrado-Gallego, J. Aguilar-Ruiz, and M. Garre. Attribute selection in software engineering datasets for detecting fault modules. In *Software Engineering and Advanced Applications, 2007. 33rd EUROMICRO Conference on*, pages 418–423. IEEE, 2007.

[62] L. Sehgal, N. Mohan, and P. S. Sandhu. Quality prediction of function based software using decision tree approach. In *Proceedings of the International Conference on Computer Engineering and Multimedia Technologies (ICCEMT)*, pages 43–47, 2012.

[63] A. Shaik, C. Reddy, B. Manda, C. Prakashini, and K. Deepthi. Metrics for object oriented design software systems: a survey. *Journal of Emerging Trends in Engineering and Applied Sciences*, 1(2):190–198, 2010.

[64] S. Sheng and C. X. Ling. Hybrid cost-sensitive decision tree. In *Knowledge Discovery in Databases: PKDD 2005*, pages 274–284. Springer, 2005.

[65] M. Shepperd, Q. Song, Z. Sun, and C. Mair. Data quality: Some comments on the nasa software defect datasets. *IEEE Transactions on Software Engineering*, (9):1208–1215, 2013.

[66] J. S. Shirabad and T. J. Menzies. The promise repository of software engineering databases. *School of Information Technology and Engineering, University of Ottawa, Canada*, 24, 2005.

[67] S. Shivaji, E. J. Whitehead, R. Akella, and S. Kim. Reducing features to improve code change-based bug prediction. *IEEE Transactions on Software Engineering*, 39(4):552–569, 2013.

[68] Q. Song, Z. Jia, M. Shepperd, S. Ying, and J. Liu. A general software defect-proneness prediction framework. *IEEE Transactions on Software Engineering*, 37(3):356–370, 2011.

[69] C. Tantithamthavorn. ScottKnottESD: The Scott-Knott Effect Size Difference (ESD) Test. https://cran.r-project.org/web/packages/ScottKnottESD/index.html, 2016.

[70] C. Tantithamthavorn, S. McIntosh, A. E. Hassan, and K. Matsumoto. Automated parameter optimization of classification techniques for defect prediction models. In *Proceedings of the 38th International Conference on Software Engineering*, pages 321–332. ACM, 2016.

[71] T. Wang, W. Li, H. Shi, and Z. Liu. Software defect prediction based on classifiers ensemble. *Journal of Information & Computational Science*, 8(16):4241–4254, 2011.

[72] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[73] I. H. Witten, E. Frank, L. E. Trigg, M. A. Hall, G. Holmes, and S. J. Cunningham. Weka: Practical machine learning tools and techniques with java implementations. *Proceedings of ANNES'99 International Workshop on emerging Eng*, 1999.

[74] Z. Xu, J. Liu, Z. Yang, G. An, and X. Jia. The impact of feature selection on defect prediction performance: An empirical comparison. In *Software Reliability Engineering (ISSRE), 2016 IEEE 27th International Symposium on*, pages 309–320. IEEE, 2016.

[75] L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *ICML*, volume 3, pages 856–863, 2003.