A Novel Dataset-Similarity-Aware Approach for Evaluating Stability of Software Metric Selection Techniques

Taghi M. Khoshgoftaar  
*Florida Atlantic University*, taghi@cse.fau.edu

Huanjing Wang  
*Western Kentucky University*, huanjing.wang@wku.edu

Randall Wald  
*Florida Atlantic University*, rwald1@fau.edu

Amri Napolitano  
amrifau@gmail.com

Follow this and additional works at: [https://digitalcommons.wku.edu/comp_sci](https://digitalcommons.wku.edu/comp_sci)

Part of the Computer Engineering Commons, and the Computer Sciences Commons

**Recommended Repository Citation**

Available at: [https://digitalcommons.wku.edu/comp_sci/5](https://digitalcommons.wku.edu/comp_sci/5)

This Article is brought to you for free and open access by TopSCHOLAR®. It has been accepted for inclusion in Computer Science Faculty Publications by an authorized administrator of TopSCHOLAR®. For more information, please contact topscholar@wku.edu.
Western Kentucky University

From the SelectedWorks of Dr. Huanjing Wang

August, 2012

A Novel Dataset-similarity-aware Approach for Evaluating Stability of Software Metric Selection Techniques

Huanjing Wang, Western Kentucky University
Taghi M. Khoshgoftaar
Randall Wald
Amri Napolitano

Available at: https://works.bepress.com/huanjing_wang/19/
A Novel Dataset-Similarity-Aware Approach for Evaluating Stability of Software Metric Selection Techniques

Huanjing Wang, Taghi M. Khoshgoftaar, Randall Wald, and Amri Napolitano
{huanjing.wang@wku.edu, khoshgoftaar@fau.edu, rwald1@fau.edu, amrifau@gmail.com}

Abstract—Software metric (feature) selection is an important preprocessing step before building software defect prediction models. Although much research has been done analyzing the classification performance of feature selection methods, fewer works have focused on their stability (robustness). Stability is important because feature selection methods which reliably produce the same results despite changes to the data are more trustworthy. Of the papers studying stability, most either compare the features chosen from different random subsamples of the dataset or compare each random subsample with the original dataset. These either result in an unknown degree of overlap between the subsamples, or comparing datasets of different sizes. In this work, we propose a fixed-overlap partition algorithm which generates a pair of subsets with the same number of instances and a specified degree of overlap. We empirically evaluate the stability of 19 feature selection methods in terms of degree of overlap and feature subset size using sixteen real software metrics datasets. Consistency index is used as the stability measure, and we show that RF is the most stable filter. Results also show that degree of overlap and feature subset size do affect the stability of feature selection methods.

I. INTRODUCTION

Software metrics are collected at various states of a software development life cycle. This data is often associated with defects found both before and after the project is released. Feature selection helps separate relevant software metrics from irrelevant or redundant ones, thereby selecting the set of software metrics that are the best predictors of fault proneness for new components, modules, and releases. However, despite the elimination of possible data, feature selection can lead to more efficient and accurate classifiers [1], [2].

While feature selection is a necessary step, very little work has focused on the robustness (stability) of the feature selection methods in regards to software metrics data. The purpose of studying the stability of a feature selection technique is to determine which technique provides the feature subset that is the most robust to changes in the data. Robust techniques are less likely to provide unhelpful features because of variations in the dataset. In this study, we propose a fixed-overlap partitions algorithm to generate a pair of subsets which have the same number of instances and a specified degree of overlap (fraction of instances in common). Then the features chosen from the pair of subsets using a feature selection method are compared. The proposed algorithm is different from the approaches used by most researchers, which either generate multiple random subsamples of the original dataset and compare the features chosen from these with one another, or compare the features from the subsamples directly with the features from the original data. Our proposed algorithm improves upon these techniques because the first approach can not control the similarity between subsamples and the second approach compares features from different sizes of datasets, while our algorithm has neither of these issues.

We examine the stability of nineteen different feature selection techniques, including six commonly used feature selection techniques, Signal to Noise, SVM-RFE, and eleven threshold-based feature selection (TBFS) techniques. We evaluate the stability of a metric on a pair of subsets using the fixed-overlap partitions algorithm. Five different levels of overlap are considered in this study, as are four different feature subset sizes. The empirical validation of the stability measure was implemented through a case study of four consecutive releases of a very large telecommunications software system (denoted as LLTS), three datasets from NASA project KC1, and nine datasets from the Eclipse project.

The main contribution of the present work is that we consider the stability of feature selection techniques by comparing the selected features generated from two subsets which have same number of instances and a specified level of overlap, rather than comparing separate random subsamples of the original dataset with each other or with the original datasets. This is an important distinction because without controlling the degree of overlap, it is unclear how much of the similarity comes from similarity of the datasets being compared, and because feature selection sometimes behaves differently as the size of the dataset grows. Only by controlling both factors (dataset size and degree of overlap) can the results be used to understand how the degree of perturbation in a dataset affects stability.

The rest of the paper is organized as follows. We review relevant literature on feature selection techniques in Section II. Section III provides detailed information about the 19 feature selection techniques. Section IV describes our methods in performing the study. Section V describes the datasets used in the study and presents experimental results and analysis. Finally, in Section VI, the conclusion is summarized and suggestions for future work are indicated.

II. RELATED WORK

Feature selection is a a necessary step in data mining. The main goal of feature selection is to select a subset of features that minimizes the prediction errors of classifiers. Feature selection can be broadly grouped as feature ranking and feature subset selection. Feature ranking sorts the attributes according to their individual predictive power, while feature subset selection finds subsets of attributes that collectively have good predictive power. Feature selection can also be categorized as filters and wrappers. Filters are algorithms in which a feature subset is selected without involving any learning algorithm. Wrappers are algorithms that use feedback from a learning algorithm to determine which feature(s) to include in building a classification model.

that sample size clearly affects the ranked gene lists produced by different feature selection methods.

Feature selection has been applied in many data mining and machine learning applications. However, its application in the software quality and reliability engineering domain is limited. Chen et al. [6] have studied the applications of wrapper-based feature selection in the context of software cost/effort estimation. They concluded that the reduced dataset improved the estimation. In a recent study [7] by Gao et al., a comparative investigation in the context of software quality estimation is presented for evaluating a proposed hybrid attribute selection approach, in which feature ranking is first used to reduce the search space, followed by a feature subset selection.

These works focused on the performance of models built using the selected features. However another way to evaluate a feature selection technique is robustness (stability), which has received less attention in the past. Few studies exist on the stability of feature selection algorithms. The stability of a feature selection method is normally defined as the degree of agreement between its outputs to randomly selected subsets of the same input data [8], [9]. Wang et al. [10] consider the stability of feature selection techniques by comparing the selected features before and after some instances are deleted from a dataset (or equivalently, before and after some instances are added), rather than directly comparing separate subsamples of the original dataset.

Although few works consider the impact of dataset similarity when performing perturbation experiments, one paper, by Alelyani et al. [11], does. In this paper, the authors note that without controlling for similarity, it is difficult to tell whether two feature subsets are different due to underlying stability issues with the ranker or due to differences in the datasets they were drawn from. To evaluate this, the researchers sampled 25% of the instances into one subset, and then created nine more subsets with exactly \( c \) of their instances in common with the first. The pairwise stability of the features from these subsets were evaluated as \( c \) varied from 0 to 1. They found that some algorithms were not able to outperform the inherent stability of the underlying datasets, and so should not be considered “stable” regardless of their stability performance.

Although Alelyani et al. raises an important question about the role of dataset similarity, it does not necessarily address this question to the extent it deserves. Notably, during their experiments with varying the amount of overlap between subsets, only the overlap between the first subset and the remaining nine is considered; the overlap among the nine is not, and will depend on random chance. In addition, by consistently using only 25% of the instances from their datasets (which have as few as 85 instances to start with), they discard much of their data. Finally, although their proposal to compare a ranker’s stability with the minimum stability provided by the dataset is useful, it doesn’t address the problem of selecting stable rankers for different subset sizes, degree of class balance, size of underlying dataset, or difficulty of learning of the underlying dataset. These questions and more remain open.

Another work, Haury et al. [12], considers the role of overlap when considering the stability of gene subsets. In addition to other analysis of their datasets, the researchers consider the fraction of instances in common when comparing feature lists generated from subsamples of the original data which either have 80% or 0% overlap. They also compare feature lists among four distinct (but related) datasets. They found that the similarity measures for the 0% overlap case more closely resembled the between-datasets case than did the results from the 80% overlap case. However, unlike the 0% case, where it is noted that the original data was divided into two mutually-exclusive groups (which therefore have 0% overlap), for the 80% case the two groups were generated by adding 80% of the data from the original dataset into each group, and then splitting the remaining 20% in half and putting each half into one of the groups. Thus, the 80% refers to proportion of the original data shared by the two groups, not the overlap between the two groups. This makes it difficult to generalize the approach to create datasets with arbitrarily-chosen overlaps.

### III. Filter-based Feature Selection Techniques

This work focuses on filter-based feature ranking. Filter-based feature ranking techniques rank features independently without involving any learning algorithm. We chose this class of feature selection algorithm because for large datasets, feature subset evaluation (including wrappers) can be computationally prohibitive. In this work, the feature rankers (filters) chosen can be placed into two categories: eleven threshold-based feature selection techniques (TBFS) that were developed by our research team and eight non-TBFS feature selection techniques including six commonly-used filters, signal-to-noise, and SVM-RFE. Table I contains all of the feature selection techniques used and their abbreviations.

#### A. Non-TBFS Feature Selection Techniques

The non-TBFS feature selection techniques we examine include six commonly used feature selection techniques, signal-to-noise, and SVM-RFE. The six standard, commonly-used filter-based feature ranking methods include: chi-squared [13], information gain [14], [13], gain ratio [13], two types of ReliefF [15], and symmetrical uncertainty [16], [13]. All of these feature selection methods, with the exception of signal-to-noise, are available within the WEKA [13] machine learning tool [32], and WEKAs default parameter values were used unless otherwise noted. Since most of these methods are widely known, we provide only a brief summary; the interested reader should consult with the included references for further details.

The chi-square (CS) [17] test is used to examine if there is ‘no association’ between two attributes, i.e., whether the two variables are independent. CS is more likely to find significance to the extent that (1) the relationship is strong, (2) the sample size is large, and/or (3) the number of values of the two associated features is large.

Information gain, gain ratio, and symmetrical uncertainty are measures based on the concept of entropy, which is based on information theory. Information gain (IG) [18] is the information provided about

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>Chi-squared</td>
</tr>
<tr>
<td>GR</td>
<td>Gain Ratio</td>
</tr>
<tr>
<td>IG</td>
<td>Information Gain</td>
</tr>
<tr>
<td>RF</td>
<td>ReliefF</td>
</tr>
<tr>
<td>RFW</td>
<td>ReliefF — Weight by Distance</td>
</tr>
<tr>
<td>SU</td>
<td>Symmetric Uncertainty</td>
</tr>
<tr>
<td>SN</td>
<td>Signal to Noise</td>
</tr>
<tr>
<td>SVM-RFE</td>
<td>SVM Recursive Feature Elimination</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Threshold-Based Feature Selection Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM</td>
</tr>
<tr>
<td>OR</td>
</tr>
<tr>
<td>PO</td>
</tr>
<tr>
<td>PR</td>
</tr>
<tr>
<td>GI</td>
</tr>
<tr>
<td>MI</td>
</tr>
<tr>
<td>KS</td>
</tr>
<tr>
<td>DV</td>
</tr>
<tr>
<td>GM</td>
</tr>
<tr>
<td>AUC</td>
</tr>
<tr>
<td>PRC</td>
</tr>
</tbody>
</table>
the target class attribute Y, given the value of independent attribute X. Information gain measures the decrease of the weighted average impurity of the partitions, compared with the impurity of the complete set of data. A drawback of IG is that it tends to prefer attributes with a larger number of possible values; that is, if one attribute has a larger number of values, it will appear to gain more information than those with fewer values, even if it is actually no more informative. One strategy to counter this problem is to use the gain ratio (GR), which penalizes multiple-valued attributes. Symmetrical uncertainty (SU) [14] is another way to overcome the problem of IG’s bias toward attributes with more values, doing so by dividing IG by the sum of the entropies of X and Y. These four techniques (CS, IG, GR, and SU) utilize the method of Fayyad and Irani [19] to discretize continuous attributes, and all four methods are bivariate, considering the relationship between each attribute and the class, excluding the other independent variables.

Relief is an instance-based feature ranking technique [20]. ReliefF is an extension of the Relief algorithm that can handle noise and multi-class datasets. When the ‘weightByDistance’ (weight nearest neighbors by their distance) parameter is set as default (false), the algorithm is referred to as RF; when the parameter is set to true, the algorithm is referred to as RFW.

Signal-to-noise ratio is a measure used in electrical engineering to quantify how much a signal has been corrupted by noise. It is defined as the ratio of the signal’s power to the noise’s power corrupting the signal. The signal-to-noise (S2N) ratio can also be used as a feature ranking method [21]. For a binary class problem (such as $f_p$, $nfp$), the equation for signal to noise is:

$$S2N = \frac{\mu_p - \mu_n}{\sigma_p + \sigma_n}$$

where $\mu_p$ and $\mu_n$ are the mean values of that particular attribute in all of the instances which belong to a specific class, either $P$ or $N$ (the positive and negative classes). $\sigma_p$ and $\sigma_n$ are the standard deviations of that particular attribute as it relates to the two classes, respectively. If one attribute’s expression in one class is quite different from its expression in the other, and there is little variation within the two classes, then the attribute is predictive. The larger the S2N ratio, the more relevant a feature is to the dataset [22].

Support Vector Machines (SVM) as classifiers use a linear discrimination function in order to make a classification. SVM-RFE extends the SVM classifier to perform feature ranking. A linear classifier is trained and the features are ranked according to their weights derived from the support vectors (a small number of critical boundary instances). These weights are used for recursive feature elimination for Support Vector Machines (SVM-RFE). At each iteration one or more features with the lowest score (weight) are eliminated. The process is repeated until a predefined number of features remains. SVM-RFE is implemented in the WEKA tool as SVMAttributeEval [13]. Unfortunately, the recursive portion of SVM-RFE is infeasible for large datasets because it requires repeatedly training the SVM classifier a large number of times. Due to this and based on the results of a previous work [23], we only use the initial rankings (i.e., without feature elimination) for our experiment.

### B. Threshold-Based Feature Selection Techniques

Eleven threshold-based feature selection techniques (TBFS) were developed and implemented by our research group within WEKA [13]. The procedure is shown in Algorithm 1. First each attribute’s values are normalized between 0 and 1 by mapping $F^j$ to $\hat{F}^j$. The normalized values are treated as posterior probabilities. Each independent attribute (software predictor variable) is then paired individually with the class attribute (fault-prone or not-fault-prone label) and the reduced dataset is evaluated using eleven different classifier performance metrics based on a set of posterior probabilities. In standard binary classification, the predicted class is assigned using the default decision threshold of 0.5. The default decision threshold is often not optimal, especially when the relative class distribution is imbalanced. Therefore, we propose the use of performance metrics that can be calculated at various points in the distribution of $F^j$. At each threshold position, the values above the threshold are classified as positive, and negative otherwise. We then consider swapping the positive and negative, i.e. values about the threshold are classified as negative, and positive otherwise. Whichever direction of the positive and negative labeling produces the more optimal attribute values is used. In a binary classification problem such as fault-prone (positive) or not-fault-prone (negative), there are four possible classification outcomes. The threshold-based feature ranking technique utilizes the classification rates as described below.

- **F-measure (FM)**: is a single value metric derived from the F-measure that originated from the field of information retrieval [13].

$$FM = \max_{t \in [0,1]} \left( 1 + \beta^2 \right) \frac{TPR(t) \times PRE(t)}{\beta^2 \times TPR(t) + PRE(t)}$$

$\beta$ is set to 1 in this study.

- **Odds Ratio (OR)**: is the maximum value of the ratio of the product of correct (true positive rate times true negative rate) to incorrect (false positive rate times false negative rate) predictions. The odds ratio is defined as:

$$OR = \max_{t \in [0,1]} \frac{TPR(t)}{FPR(t)} \left( \frac{TNR(t)}{FNR(t)} \right)$$

- **Power (PO)**: is a measure that avoids common false positive cases while giving stronger preference for positive cases [25]. Power is defined as:

$$PO = \max_{t \in [0,1]} \left( \frac{TNR(t)}{FNR(t)} \right)^k - (FNR(t))^k$$

where $k = 5$.

---

**Algorithm 1: Threshold-based Feature Selection Algorithm**

input:
1. Dataset $D$ with features $F^j$, $j = 1, \ldots, m$;
2. Each instance $x \in D$ is assigned to one of two classes $c(x) \in \{f_p, nfp\}$;
3. The value of attribute $F^j$ for instance $x$ is denoted $F^j(x)$;
4. Threshold-based feature ranking technique $\omega \in \{FM, OR, PO, PR, GI, MI, KS, DV, GM, AUC, PRC\}$;
5. A predefined threshold: number (or percentage) of the features to be selected.

output:
Selected feature subsets.

for $F^j, j = 1, \ldots, m$ do

- Normalize $F^j$ to $\hat{F}^j = \frac{\max(F^j) - \min(F^j)}{\max(F^j) - \min(F^j)}$;
- Calculate metric $\omega$ using attribute $\hat{F}^j$.
Create feature ranking $R$ using $\omega(\hat{F}^j)[\cdot]$.
Select features according to feature ranking $R$ and a predefined threshold.
• **Probability Ratio (PR):** is the sample estimate probability of the feature given the positive class divided by the sample estimate probability of the feature given the negative class [25]. The probability ratio is defined as:

\[
PR = \max_{t \in [0, 1]} \frac{TPR(t)}{FPR(t)}
\]

• **Gini Index (GI):** measures the impurity of a dataset [26]. GI for the attribute is then the minimum Gini index at all decision thresholds \( t \in [0, 1] \).

\[
GI = \min_{t \in [0, 1]} [2PRE(t)(1 - PRE(t)) + 2NPV(t)(1 - NPV(t))].
\]

• **Mutual Information (MI):** measures the mutual dependence of the two random variables [27]. High mutual information indicates a large reduction in uncertainty, and zero mutual information between two random variables means the variables are independent.

• **Kolmogorov-Smirnov (KS):** utilizes the Kolmogorov-Smirnov statistic to measure the maximum difference between the empirical distribution function of the attribute values of instances in each class [28]. The larger the distance between the distribution functions, the better the attribute is able to distinguish between the two classes. It is effectively the maximum difference between the curves generated by the true positive and false positive rates as the decision threshold changes from 0 and 1.

• **Deviance (DV):** is the residual sum of squares based on a threshold \( t \). It measures the sum of the squared errors from the mean class given a partitioning of the space based on the threshold \( t \). As deviance represents errors, the minimum value is considered optimal.

• **Geometric Mean (GM):** is a single-value performance measure that ranges from 0 to 1 which is calculated by finding the maximum geometric mean of \( TPR \) and \( TNR \) as the decision threshold is varied between 0 and 1:

\[
GM = \max_{t \in [0, 1]} \sqrt{TPR(t) \times TNR(t)}
\]

• **Area Under ROC (Receiver Operating Characteristic) Curve (AUC):** has been widely used to measure classification model performance [29]. AUC is a single-value measurement that ranges from 0 to 1. The ROC curve is used to characterize the trade-off between true positive rate and false positive rate. In this study, ROC curves are generated by varying the decision threshold \( t \) used to transform the normalized attribute values into a predicted class.

• **Area Under the Precision-Recall Curve (PRC):** is a single-value measure that originated from the area of information retrieval. The area under the PRC ranges from 0 to 1. The PRC diagram depicts the trade-off between recall and precision.

**IV. METHODOLOGY**

The stability of feature selection methods are evaluated using the consistency index (IV-C) calculated based on various feature subset size (IV-B) using rankings from pairs of data subsamples with controlled degree of overlap (IV-A).

**A. Fixed-overlap Partitions**

Many approaches have been used to test the stability of feature ranking techniques. Some take random subsamples from the original dataset and compare the features chosen on these subsamples with each other; others compare the features chosen on the subsamples with those chosen from the original dataset. The first of these approaches has a known flaw: it does not control for the degree of overlap between the subsamples being compared (instead leaving this to random chance). This makes it difficult to determine whether the similarity between feature subsets is due to the similarity of the underlying datasets or is a property of the feature selection technique used. The second approach is somewhat limited in scope: although it is useful for observing stability in the case of adding or removing instances from a dataset, its use of two datasets of different sizes can impact how well the results generalize to other perturbation scenarios. Neither is able to evaluate how similar the feature ranking will be for two datasets which are equal in size and have a known degree of overlap. To address this, we propose a novel algorithm, the Fixed-Overlap Partitions Algorithm (Algorithm 2), which will create two new subsets that have the desired properties while also being as large as possible for the given degree of overlap. Note in this algorithm that \( c \), the desired degree of overlap, can vary from 0 to 1, including the endpoints. A choice of \( c = 0 \) will find two entirely disjoint subsets, which will each contain half of the instances from the original dataset. On the other hand, \( c = 1 \) will create two copies of the original dataset which share all instances. This is generally not an interesting case to study, but is permitted by the algorithm.

**Algorithm 2: Fixed-Overlap Partitions**

input : Original dataset \( S \) with \( N \) instances 
: \( c \), the fraction of instances the two subsampled datasets should have in common \( (0 \leq c \leq 1) \)
output: Datasets \( S_1 \) and \( S_2 \) which have \( c \) of their instances in common while being identical in size and as large as possible for the given \( c \).

Let \( d = 1/(2 - c) \) (i.e., \( c = (2d - 1)/d \))

\( S_1 \) and \( S_2 \) start out empty
Randomly select \( dN \) instances from \( S \) and add them to \( S_1 \)
Randomly select \( cdN \) instances from \( S_1 \) and add them to \( S_2 \)
Take all instances in \( S \) which are not in \( S_1 \) and add them to \( S_2 \)

There are three properties which must be guaranteed when selecting these subsets: that they contain the same number of instances, that they have the specified degree of overlap, and that they are as large as possible while the first two properties hold true (since there is no reason to discard instances if they could be used to improve feature selection or classification). Based on Algorithm 2, we can see that \( S_1 \) contains \( dN \) instances. To find the number of instances in \( S_2 \), we note that two steps add instances to that dataset: one adds \( cdN \) instances and the other adds the instances not included in \( S_1 \) (e.g., \((1-d)N\) instances). Working from here and using the definition of \( d \) in the algorithm, we have:

\[
|S_2| = cdN + (1 - d)N \\
= \left(\frac{2d - 1}{d}\right) dN + (1 - d)N \\
= (2d - 1)N + (1 - d)N \\
= 2dN - N + N - dN \\
= dN
\]

Thus, we have \(|S_1| = |S_2| = dN\), satisfying the first property. As for the second property, recall that \( S_1 \) and \( S_2 \) share precisely \( cdN \) instances; thus, they have \( cdN/dN = c \) of their instances in common, as desired. For the third property, observe that adding any instances to either \( S_1 \) or \( S_2 \) would necessarily increase the fraction of overlap (since these would have to be instances already found in the other
subsampled dataset). Thus, $S_1$ and $S_2$ are the largest datasets which are identical in size and have an overlap of precisely $c$.

In this study, the degree of overlap is chosen from the set \{0, 0.25, 0.5, 0.7, 0.85\}. A choice of $c = 0$ will find two completely different subsets with $0.5N$ instances and no overlap while a choice of $c = 0.85$ will generate two subsets with $0.87N$ ($0.87 \approx 1/(2 - 0.85)$) instances.

### B. Feature Selection

First the features are ranked according to their relevance to the class using 19 different feature selection techniques separately. These ranked lists are found for all the data subsamples created by applying Algorithm 2 30 times to each of the original datasets for each of the specified degrees of overlap. The next step is to select a subset consisting of the most relevant features. In this study, four subset sizes were used, namely 3, 4, 5, and 6. These numbers were deemed reasonable after some preliminary experimentation conducted on the corresponding datasets [30].

#### C. Stability Measure

To assess the robustness of feature selection techniques, past works have used different similarity measures, such as Hamming distance [31], correlation coefficient [32], consistency index [8], and entropy [33]. Among these four similarity measures, consistency index is the only one which takes into consideration bias due to chance. Because of this, in our work the consistency index was used as the stability measure. The consistency index was defined by Kuncheva et al [8], and is computed between two feature subsets as follows. Let $T_i$ and $T_j$ be subsets of features, where $|T_i| = |T_j| = k$. The consistency index $\theta$ is obtained as follows:

$$IC(T_i, T_j) = \frac{d_n - k^2}{k(n-k)},$$

where $n$ is the total number of features in the dataset, $d$ is the cardinality of the intersection between subsets $T_i$ and $T_j$, and $-1 < IC(T_i, T_j) \leq +1$. The greater the consistency index, the more similar the subsets are.

### V. Experiments

In order to test the stability of our nineteen feature selection techniques we employed sixteen different software metrics datasets. With each feature ranker and dataset combination, we used five different levels of overlap partitions along with four different numbers of features chosen.

#### A. Dataset

Experiments conducted in this study used software metrics and defect datasets collected from real-world software projects, including a very large telecommunications software system (denoted as LLTS) [34], the Eclipse project [35], [36], and NASA software project KC1 [37].

LLTS contains data from four consecutive releases, which are labeled as SP1, SP2, SP3, and SP4. The software measurement datasets consist of 42 software metrics, including 24 product metrics, 14 process metrics, and four execution metrics [34]. The dependent variable is the class of the program module: fault-prone ($fp$) or not fault-prone ($nfp$). A program module with one or more faults is considered $fp$, and $nfp$ otherwise.

From the PROMISE data repository [36], we also obtained the Eclipse defect counts and complexity metrics dataset. In particular, we use the metrics and defects data at the software package level. The original data for the Eclipse packages consists of three releases denoted 2.0, 2.1, and 3.0 respectively. We transform the original data by: (1) removing all nonnumeric attributes, including the package names, and (2) converting the post-release defects attribute to a binary class attribute: fault-prone ($fp$) and not fault-prone ($nfp$). Membership in each class is determined by a post-release defects threshold $t$, which separates $fp$ from $nfp$ packages by classifying packages with $t$ or more post-release defects as $fp$ and the remaining as $nfp$. In our study, we use $t \in \{10, 5, 3\}$ for release 2.0 and 3.0 while we use $t \in \{5, 4, 2\}$ for release 2.1. These values are selected in order to have datasets with different levels of class imbalance. All nine derived datasets contain 209 attributes. Releases 2.0, 2.1, and 3.0 contain 377, 434, and 661 instances respectively.

The NASA project, KC1 [37], includes 145 instances containing 95 attributes each. After removing 32 Halstead derived measures, we have 63 attributes. We used three different thresholds to define defective instances, thereby obtaining three structures of the preprocessed KC1 dataset. The thresholds are 20, 10, and 5, indicating instances with numbers of defects greater than or equal to 20, 10, or 5 belong to $fp$ class. The three datasets are named KC1-20, KC1-10, and KC1-5.

Table II lists the characteristics of the 16 datasets utilized in this work, which exhibit different distributions of class skew (i.e., the percentage of $fp$ modules).

#### B. Experimental Design

The experiments were conducted to discover the robustness (stability) of 19 rankers and the impact of feature subset size and degree of overlap partitions. There are three main factors which can be examined to observe their effects on stability: degree of overlap, number of features used, and choice of filter. We first generate 2400 pairs of subsets (16 original datasets $\times$ 5 levels of overlap $\times$ 30 repetitions). For each combination of feature selection method and pair of data subsets, we applied the feature selection method, generated a ranking list, and then selected a subset of attributes. In this study, the top three, four, five, and six features are selected according to their respective scores. Consistency index is then computed for each pair of feature subsets. In total, 182400 consistency index (16 original datasets $\times$ 5 levels of overlap $\times$ 30 repetitions $\times$ 19 ranker $\times$ 4 feature subsets) values are computed.

#### C. Results and Analysis

Tables III through VII contain the results from the experiments, with each table holding partition overlap constant and showing the results for all feature rankers and numbers of selected features separately. Note that each value presented in the tables is the average over the thirty repetitions across all datasets. The top value for each level of feature subset size is highlighted with boldfaced print and the
The smallest value is in italicized print. From these tables, we can observe the following facts: (1) Among the 19 rankers, RF shows the most stable filters in terms of consistency index. The most unstable ranker by far was GR. (2) Stability increases as feature subset size increases. The only exception to this trend seems to be SVM-RFE, which loses stability slightly as the number of features increases. (3) As the degree of overlap increases, the stability of the rankers increased. This trend persists for different numbers of features chosen.

We also conducted an Analysis Of Variance (ANOVA) F test [38] to statistically examine the various effects on the performances of the stability. A K-way ANOVA can be used to determine if the means in a set of data differ when grouped by multiple factors. If they do differ, one can determine which factors or combinations of factors are associated with the difference. In this study, a two-way ANOVA test was used. The ANOVA model for stability includes two factors: Factor A represents 19 feature subset selection strategies (rankers), and Factor B represents the five degrees of overlap. In addition, the interaction AxB was also considered in the ANOVA test. For our ANOVA tests, the results from all 16 datasets with feature subset size six were taken into account together.
TABLE VIII
ANALYSIS OF VARIANCE

<table>
<thead>
<tr>
<th>Source</th>
<th>Sum Sq.</th>
<th>d.f.</th>
<th>Mean Sq.</th>
<th>F</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>416.35</td>
<td>18</td>
<td>23.13</td>
<td>517.9</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>919.56</td>
<td>4</td>
<td>229.89</td>
<td>5147.29</td>
<td>0</td>
</tr>
<tr>
<td>A×B</td>
<td>15.37</td>
<td>72</td>
<td>0.213</td>
<td>4.78</td>
<td>0</td>
</tr>
<tr>
<td>Error</td>
<td>2032.36</td>
<td>45505</td>
<td>0.045</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Total</td>
<td>3383.64</td>
<td>45599</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

The ANOVA model can be used to test the hypothesis that the stability for the main factors are equal against the alternative hypothesis that at least one mean is different. If the alternative hypothesis (i.e., that at least one mean is different) is accepted, numerous procedures can be used to determine which of the means are significantly different from the others. This involves the comparison of two means, with the null hypothesis that the means are equal. In this study, we performed the multiple comparison tests using Tukey’s honestly significant difference criterion [38]. All tests of statistical significance utilize a significance level α of 5%.

The ANOVA results are presented in Table VIII. From the table, we can see that all the p-values were zero, indicating that for the main factors and their interaction, the alternate hypothesis is accepted, namely, at least two group means are significantly different from each other. Multiple comparison tests for the main factors were performed to investigate the differences among the respective groups (levels). Both the ANOVA and multiple comparison tests were implemented in MATLAB.

The multiple comparison results are presented in Figure 1, displaying graphs with each group mean represented by a symbol (○) and the 95% confidence interval as a line around the symbol. Two means are significantly different if their 95% confidence intervals are disjoint, and are not significantly different if their intervals overlap. Some findings can be summarized from these tables and figures.

- For Factor A (feature selection techniques), we can divide the 19 rankers into seven groups, \{GR\}, \{PR\}, \{GI, SU, SVM, OR\}, \{CS, IG, KS, FM, GM, DV, MI\}, \{PO, S2N\}, \{AUC, PRC\}, \{RFW, RF\}, with the groups ordered by their performances from worst to best. The rankers from different groups performed significantly different, while the rankers from same group performed similarly (were not significantly different from one another). For the last group, RF performs best.
- For Factor B, the degree of overlap significantly affects the robustness of feature selection techniques. Higher overlap between two datasets produces more stable ranking by the feature selection techniques. This works for all feature selection techniques.

VI. CONCLUSION

In this paper, we empirically study the stability of feature selection techniques to find those techniques which produce the most reliable and trustworthy features. Instead of comparing features chosen from pairs of random subsamples (without controlling the underlying similarity) or comparing random subsamples with the original dataset (resulting in datasets having different numbers of instances), we propose a novel fixed-overlap partitions algorithm to generate a pair of subsets with the same size and a specified degree of overlap. By controlling these two factors, we ensure that the variation between the datasets being compared is precisely at the desired level, and the only difference between the datasets is this variation. Thus, our results directly show the effects of the given degree of perturbation on feature ranking stability.

We consider nineteen feature selection methods, five different levels of overlap, and four different feature subset sizes in our evaluation. Our findings show that RF and RFW are the most stable rankers regardless of level of overlap and feature subset size, followed by PRC, AUC, and S2N. GR is the least stable filter. They also show that, in general the bigger the feature subset size, the more stable the rankers are. Our work provides guidance to software practitioners as to which feature selection methods are more stable. This is also useful for other domains in which feature selection is used for preprocessing data.

Future work may include experiments using additional datasets from other software engineering and non-software engineering domains, and experiments with other ranking techniques and classifiers for building classification models.

REFERENCES


