Ensemble Feature Selection Technique for Software Quality Classification

Huanjing Wang
Western Kentucky University
Bowling Green, Kentucky 42101
huanjing.wang@wku.edu

Taghi M. Khoshgoftaar
Florida Atlantic University
Boca Raton, Florida 33431
taghi@cse.fau.edu

Kehan Gao
Eastern Connecticut State University
Willimantic, Connecticut 06226
gaok@easternct.edu

***Abstract*—Feature selection is an important data preprocessing step in data mining applications. In this paper, we studied six filter-based feature ranking techniques and an ensemble technique using rank ordering of the features (mean or median). The best features are selected through either the individual ranker or an ensemble approach. Then the reduced data set is used to build classification models using three well-known classifiers within the domain of software quality engineering. The classification accuracy was evaluated in terms of the AUC (Area Under the Receiver Operating Characteristic Curve) performance metric. Results demonstrate that the ensemble technique performed better overall than any individual ranker and also possessed better robustness. The empirical study also shows that variations among rankers, learners and software projects significantly impacted the classification outcomes.**

I. Introduction

Software quality data (metrics) that are collected during the software development process include valuable information about a software project. Software practitioners strive to improve software quality by constructing quality prediction models using software metrics (attributes or features) and data mining techniques. Feature selection can be broadly classified 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 which collectively have good predictive power. Feature selection techniques can also be categorized as *filters*, *wrappers* and *hybrids*. *Filters* are algorithms in which a feature set is selected without involving any learning algorithm. *Wrappers* [1], [2] are algorithms that use feedback from a learner to select features. The *hybrids* [3] are combination of filters and wrappers, and thus exploit advantage from both methods. This work focuses on filter-based feature ranking.

Using a single method for feature selection may generate local optima. Ensemble methods attempt to combine multiple methods of different types instead of using a single one. These methods are an emerging area in machine learning research. Ensemble feature selection combines multiple feature subsets to get a single final feature subset [4], [5]. The final feature subset will be used for future learning. There are two types of ensemble feature selection: ensembles of multiple feature selection techniques and an ensemble of a single feature selection. Ensembles of multiple feature selection combine outcomes of various feature selection techniques. The feature selection can be feature ranking, feature subset selection, or combinations of feature ranking and feature subset selection techniques. An ensemble of a single feature selection combines various outcomes (multiple views) of a single feature selection technique, where feature selection is performed on different subsamples of the original data set, including sampling, bootstrap or n-fold cross-validation.

The focus of this paper is to evaluate six commonly used filter-based rankers (feature selection techniques) and our proposed ensemble approach. The latter consists of feature ranking techniques followed by a combination method. The six feature ranking techniques considered are: chi-square (CS), information gain (IG), gain ratio (GR), two forms of the ReliefF algorithm (RF and RFW), and symmetrical uncertainty (SU). Two separate combination methods were used in the ensemble technique. One uses mean (EM) of ranking scores, while the other uses median (ED). These methods were evaluated using software measurement data in our case study, including the four consecutive releases of a very large telecommunications system and three data sets from NASA project KC1. The software quality prediction models were built using three different classification algorithms (classifiers or learners): naïve Bayes (NB), K-nearest neighbors (KNN), and support vector machine (SVM). The classification accuracy was evaluated in terms of the AUC performance metric. The results demonstrate that the performance of filter-based rankers may be significantly influenced by the data set and learner used in classification, while the ensemble approach generally performed better than the individual rankers and presented more stable and robust behavior with respect to various data sets and different learners.

The key contributions of this research are:

- Implementation and investigation of the ensemble technique. Although six filter-based feature ranking techniques and three learners are used in the study, any number of feature ranking techniques and learners can be used. This is also the first paper to investigate ensembles of such diverse ranking techniques.
- The use of imbalanced data from real-world software systems. Such an extensive range of feature ranking and ensemble approaches for software quality prediction (and other application domains) is unique to this study.

II. Related Work

This section provides a brief coverage of key ensembles feature selection works in the area of data mining and software engineering. An exhaustive discussion is avoided due to space considerations.

One of the main problems which needs to be considered when building an ensemble model is diversity. Diversity may be achieved through using different data sets, feature subsets, or classifiers. Lee [6] and Rokach et al. [7] combined outcomes of various non-ranker filter-based feature subset selection techniques. Souza et al. [8] and Loscalzo et al. [9] studied the ensemble of a single feature subset selection technique. Souza et al. applied the ensemble approach to feature selection by proposing a systematic way of combining various outcomes of a feature selection algorithm. Loscalzo et al. applied an ensemble approach to consensus feature groups.

Very limited research exists on ensemble feature ranking. Ensemble feature ranking can be classified as either ensembles of multiple feature ranking techniques or an ensemble of a single feature ranking technique. More recently, one approach was proposed by Saeys et

al. [5] using an ensemble of a single feature ranking technique. They studied an ensemble of a feature ranking technique which aggregates the top 1% of best features of the rankings. The rankers used in the study included two filter-based feature ranking techniques, SU, and RELIEF; and two embedded feature ranking techniques, Random Forests with 10 trees and SVM_RFE. Each feature ranking technique was repeatedly applied on different bootstrapped samples with 40 bags of the same data set. The classifiers used in the study included SVM, Random Forests with 50 trees and 5-NN. Their results showed that the best classification results (accuracy) were obtained by using SVM classifier and an ensemble of SVM_REF.

A recent study on ensembles of multiple feature ranking techniques was done by Olsson and Oard [4]. Their work considered combinations of three commonly-used filter-based feature ranking techniques, including document frequency thresholding, IG, and chi-square method (χ^2_{max} and χ^2_{avg}) for text classification problems. To create ensembles, two or more ranking lists were combined. They concluded that the best performing combination was χ^2_{max} and χ^2_{avg}. The experimental results showed that the ensemble approach could achieve higher peak R-precision and F_1 than a non-combined feature ranker at statistically significant levels when the classifier was built using *100*-Nearest Neighbor. The technique we propose in this work is much more general than that of Olsson and Oard. Their work built ensembles for the text classification problem, while ours does not require binary data, making it more general and useful on any numeric data set. We combined six diverse commonly used filter-based rankers instead of three. Moreover, the ensemble technique presented in this study can be used with any number of feature ranking techniques which we leave to future work.

III. Methodology

A. Filter-based Feature Ranking

Filter-based feature ranking techniques rank features independently without involving any learning algorithm. Feature ranking consists of scoring each feature according to a particular method, then selecting features based on their scores. This work used some commonly used filter-based feature ranking techniques including chi-square, information gain, gain ratio, symmetrical uncertainty, and ReliefF. For specific algorithmic details on these techniques, the reader is referred to the various cited references.

The chi-square (CS) [10] 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) [11] is the information provided about 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) [12] 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.

Relief is an instance-based feature ranking technique [13]. ReliefF is an extension of the Relief algorithm that can handle noise and multi-class data sets. 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.

B. Ensemble Method

We investigated the use of ensembles of multiple feature ranking techniques. These ranking techniques are combined to yield more stable and robust results. There are two essential steps in creating a single feature ranking list from multiple ranking lists. First, a set of different ranking lists is created using corresponding filter-based rankers and input to the next combining step; second, these ranking lists are integrated using rank ordering of the features. Diversity can be achieved by using various rankers. The combining methods used in the study include mean (EM) and median (ED). For mean combination, each feature's score is determined by the average of the ranking scores of the feature in each ranking list, while for median combination, we give each feature's combining score the median score in all ranking lists. Mean combination (average rank) has been used by Olsson and Oard [4]. We used median combination (median rank) in the paper as well.

C. Classifiers

Software quality prediction models were built with three well-known classification algorithms including naïve Bayes (NB), K-nearest neighbors (KNN), and support vector machine (SVM). These were selected because of their common use in software engineering and other data mining applications. Unless stated otherwise, we use default parameter settings for the different learners as specified in the WEKA [14] data mining tool. Parameter settings are changed only when a significant improvement in performance is obtained.

Naïve Bayes classifier (NB) [15] utilizes Bayes's rule of conditional probability and is termed 'naïve' because it assumes conditional independence of the features. K-nearest neighbors (KNN) [16] classifier, also called instance-based learning, uses distance-based comparisons. The choice of distance metric is critical. KNN was built with changes to three parameters. The 'distanceWeighting' parameter was set to 'Weight by 1/distance', the 'kNN' parameter was set to '5', and the 'crossValidate' parameter was turned on (set to 'True'). The support vector machine (SVM) [17], also called SMO in WEKA, had two changes to the default parameters: the 'complexity constant c' was set to 5.0 and 'build Logistic Models' was set to true. By default, a linear kernel was used.

D. Performance Metric

Traditional performance measures such as F-measure, overall classification accuracy, or its complement, misclassification rate, are inappropriate when dealing with the classification of imbalanced data. In a domain such as software quality prediction, the number of *fp* (fault-prone) modules is much lower than the *nfp* (not fault-prone) modules. Instead, we used a performance metric that considers the ability of a classifier to differentiate between the two classes: the area under the ROC (Receiver Operating Characteristic) curve (AUC). It has been shown that AUC has lower variance and is more reliable than other performance metrics (such as precision, recall, F-measure) [18].

The AUC is a single-value measurement, whose value ranges from 0 to 1. The ROC curve is used to characterize the trade-off between hit (true positive) rate and false alarm (false positive) rate [19]. A classifier that provides a large area under the curve is preferable over a

TABLE I
SOFTWARE DATA SETS CHARACTERISTICS

Data set	#Metrics	#Modules	%fp	%nfp
SP1	42	3649	6.28%	93.72%
SP2	42	3981	4.75%	95.25%
SP3	42	3541	1.33%	98.67%
SP4	42	3978	2.31%	97.69%
KC1-5	63	145	24.83%	75.17%
KC1-10	63	145	14.48%	85.52%
KC1-20	63	145	6.90%	93.10%

classifier with a smaller area under the curve. Traditional performance metrics consider only the default decision threshold of 0.5. ROC curves illustrate the performance across all decision thresholds. A perfect classifier provides an AUC that equals 1.

IV. EXPERIMENTS

A. Experimental Data Sets

Experiments conducted in this study used software metrics and defect data collected from real-world software projects, a very large telecommunications software system (denoted as LLTS) and NASA software project KC1. LLTS contains data from four consecutive releases, which are labeled as SP1, SP2, SP3 and SP4. The software measurement data sets consist of 42 software metrics, including 24 product metrics, 14 process metrics, and four execution metrics [20]. NASA project KC1 [21] contains three data sets KC1-5, KC1-10, KC1-20. After preprocessing, each of these data sets includes 63 attributes. The seven data sets used in this work represent software projects of different sizes with different levels of imbalance. Table I lists the characteristics of the seven data sets utilized in this work.

B. Experimental Design

We first used six filter-based rankers and our proposed ensemble technique to select the subsets of attributes. We rank the features and select the top $\lceil \log_2 n \rceil$ features according to their respective scores, where n is the number of independent features for a given data set. The reasons why we select the top $\lceil \log_2 n \rceil$ features include (1) no general guidance has been found in related literature on the number of features that should be selected when using a feature ranking technique; (2) a software engineering expert with more than 20 years experience recommended selecting $\lceil \log_2 n \rceil$ number of metrics for software quality prediction; (3) a preliminary study showed that $\lceil \log_2 n \rceil$ is appropriate for various learners; and (4) a recent study [22] showed that it was appropriate to use $\lceil \log_2 n \rceil$ as the number of features when using WEKA to build Random Forests learners for binary classification in general and imbalanced data sets in particular. Subsequently, the selected features were used to build classification models.

The experiments were conducted to discover the impact of (1) six commonly used filter-based rankers vs. ensemble technique; (2) three different learners; and (3) two different software projects from the software quality prediction domain. We implemented the ensemble technique in WEKA and used it for the defect prediction model building and testing process. In the experiments, ten runs of five-fold cross-validation were performed. The five results from the five folds were then combined to produce a single estimation. In total, 8,400 models were evaluated during our experiments.

C. Experimental Results

The classification models were evaluated in terms of the AUC performance metric. All the results are reported in Table II. Note that each value presented in the table is the average over the ten runs of five-fold cross-validation outcomes. The best model for each data set is indicated in underlined **boldfaced** print, while the worst performance is boldfaced ***italic***. Each value in the table is determined by three dimensions: (1) feature ranking techniques (CS, GR, IG, RF, RFW, SU, EM, and ED); (2) classifiers (NB, KNN and SVM); and (3) data sets (SP1, SP2, SP3, SP4, KC1-5, KC1-10, and KC1-20). We compared the performance of NB, KNN and SVM models built after feature selection. A total of 168 values are included in the three tables.

D. Analysis of Results

Table III summarizes the number of best cases (positive numbers) and the number of worst cases (negative numbers) for all the rankers on each individual classifier as well as for all classifiers together. From Table II and III, we can observe the following facts:

- CS performed best for KC1-10 when using the KNN classifiers but worst when using NB classifier. This demonstrates that the performance of a ranker is influenced by the selected classifier. In addition, for a given classifier (for instance NB), RF performed best for SP3 and KC1-5 but worst for KC1-20. This displays that for a given classifier, the same ranker can produce different results for different data sets. In other words, the characteristics of data sets impact on the performance of rankers.
- Among the six feature ranking techniques, GR performed worst. This can be easily seen from the ratio between its numbers of the best and worst cases. RF and RFW showed a considerable number of worst cases even though they also displayed a number of best cases. This indicates that RF and RFW had unstable performance with respect to different classifiers. CS, IG and SU presented less frequently best and also worst cases than other techniques. In fact, they displayed moderate and stable behavior. The ensemble technique compared to each individual ranker (except SU) showed lower number of worst cases (actually no worst cases at all). ED also had the second highest number of best cases among all the ranking techniques.

We also performed a two-way ANOVA test to statistically examine the various effects on performance of the classification models. The two factors were designed as follows. Factor A represents the eight feature selection techniques (CS, GR, IG, RF, RFW, SU, EM and ED); and Factor B represents the three learners (NB, KNN and SVM) used in classifications. The interaction effect A$\times$B was also considered in the ANOVA test. A significance level of $\alpha = 5\%$ was used for all statistical tests. Table IV presents the ANOVA results for the LLTS data sets. Note that the ANOVA tests were performed on the LLTS four releases together. From the table, we can see that the p-values for the main factors A and B, and the interaction term A$\times$B were less than 0.05, indicating the AUC values are not same for all groups in each of the main factors and also influenced by the interaction term A$\times$B, i.e., Factor A is different at every level of Factor B, and vice versa.

Additional multiple comparisons for the main factors and interaction term were performed to investigate the difference among the respective groups (levels). The test results are shown in Figure 1, where each sub-figure displays graphs with each group mean represented by a symbol ($\circ$) and 95% confidence interval. The summarized results reveal the following:

- For the eight feature selection methods (Figure 1(a)), the two ensemble methods on average outperformed the other six filter-based ranking techniques, among which GR, RF, RFW and SU

TABLE II
EXPERIMENTAL RESULTS IN TERMS OF AUC

(a) NB

Data	CS	GR	IG	RF	RFW	SU	EM	ED
SP1	0.7846	***0.7346***	0.7831	0.7879	**0.7882**	0.7865	0.7819	0.7822
SP2	0.8108	***0.7613***	0.8081	0.8053	0.8081	0.7729	0.8109	**0.8117**
SP3	0.8184	***0.7808***	0.8118	**0.8305**	0.8190	0.7882	0.8183	0.8145
SP4	0.7696	***0.7519***	0.7794	0.7731	0.7735	0.7592	**0.8039**	0.8031
KC1-5	0.7484	0.7489	***0.7438***	**0.7990**	0.7832	0.7468	0.7841	0.7679
KC1-10	***0.7513***	**0.7729**	0.7546	0.7585	0.7639	0.7719	0.7676	0.7705
KC1-20	0.8525	0.8669	0.8569	***0.8296***	**0.8987**	0.8531	0.8507	0.8521

(b) KNN

Data	CS	GR	IG	RF	RFW	SU	EM	ED
SP1	0.7570	***0.7139***	0.7475	0.7495	0.7489	**0.7600**	0.7545	0.7545
SP2	0.7800	0.7515	0.7721	***0.7221***	0.7255	0.7796	0.7791	**0.7802**
SP3	0.7879	***0.7298***	0.7802	0.7898	0.7704	0.7602	**0.7921**	0.7865
SP4	0.7913	***0.6853***	**0.7967**	0.7631	0.7665	0.7433	0.7897	0.7895
KC1-5	0.7915	***0.7590***	0.7749	**0.7923**	0.7782	0.7693	0.7875	0.7711
KC1-10	**0.8449**	0.8219	0.8180	0.7078	***0.6963***	0.8247	0.8246	0.8364
KC1-20	0.8659	**0.8896**	***0.8495***	0.8674	0.8867	0.8761	0.8721	0.8744

(c) SVM

Data	CS	GR	IG	RF	RFW	SU	EM	ED
SP1	0.6401	0.6532	0.6651	**0.6708**	***0.6368***	0.6632	0.6386	0.6538
SP2	**0.7060**	0.6577	0.6628	***0.6357***	0.6386	0.6572	0.6872	0.6905
SP3	0.6456	***0.6294***	0.6470	0.6341	0.6611	0.6601	0.6573	**0.7084**
SP4	0.6529	***0.6247***	0.6531	0.6248	0.6423	0.6399	0.6543	**0.6804**
KC1-5	0.7935	***0.7649***	0.7844	**0.8201**	0.7988	0.7768	0.7882	0.7819
KC1-10	0.7645	0.7712	0.7740	0.7521	***0.6798***	**0.7813**	0.7805	0.7784
KC1-20	0.8382	0.8432	0.8610	***0.8119***	**0.8824**	0.8396	0.8412	0.8398

TABLE III
PERFORMANCE SUMMARIZATION IN TERMS OF AUC

		CS	GR	IG	RF	RFW	SU	EM	ED
NB	# of best	0	1	0	2	2	0	1	1
	# of worst	-1	-4	-1	-1	0	0	0	0
KNN	# of best	1	1	1	0	1	1	1	1
	# of worst	0	-4	-1	-1	-1	0	0	0
SVM	# of best	1	0	0	2	1	1	0	2
	# of worst	0	-3	0	-2	-2	0	0	0
ALL	# of best	2	2	1	5	3	2	2	4
	# of worst	-1	-11	-2	-4	-3	0	0	0

TABLE IV
TWO-WAY ANOVA TABLES FOR LLTS DATA SETS

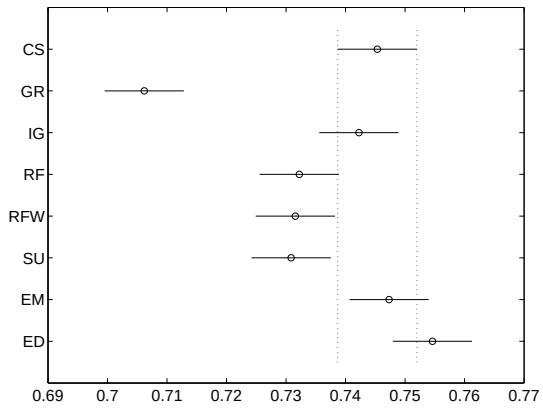
Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.1859	7	0.0266	22.95	0
B	3.2717	2	1.6358	1413.38	0
A×B	0.0534	14	0.0038	3.29	0
Error	1.0833	936	0.0012		
Total	4.5943	959			

performed significantly worse than the ensemble methods while the other two, CS and IG, performed worse. Of those six filter-based rankers, CS and IG performed best (though still worse than the ensemble techniques), and the rest fared much worse.

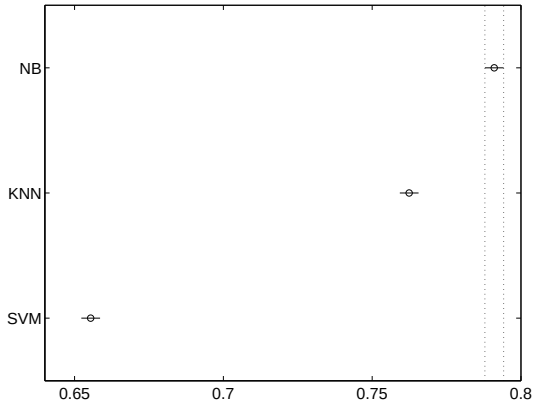
- For the three learners (Figure 1(b)), their classification performances were significantly different from each other. NB performed best followed by KNN, then SVM.
- For interaction A×B, 24 groups produced by three learners combined with eight feature selection techniques are presented. The ensemble methods are highlighted with thick lines on the 95% interval in Figure 1(c). It can be seen that Factor A was different at every level (group) of Factor B. For example, GR performed significantly worse than the other seven methods when the KNN learner was used, but did significantly worse than six of them when NB was used and only one of them when SVM was used. This demonstrates that the interaction has great impact on the results.

Table V presents the ANOVA result for the three KC1 data sets. The result demonstrates that there was no significant distinction between any pair of the group means for Factor A since the p-value (0.38) was greater than 0.05, while an obvious difference existed in at least one paired of group means for Factor B, because the p-value was zero. For interaction A×B, the conclusion was similar to the one obtained from the LLTS data sets, namely, Factor A was different at every level of Factor B. The multiple comparison tests are presented in Figure 2. The results also demonstrate that the ensemble methods on average were better than the other individual filter-based feature ranker, though not significant. In addition, the KNN learner significantly outperformed the other two learners, NB and SVM for the KC1 data sets. No distinction was found between the performances of the NB and SVM classifiers. In addition, from Figure 2(c), we can observe that the ensemble methods were more stable than the other individual ranking techniques. For instance, RFW presented the best performance among the eight feature selection techniques when the NB learner was used to build models, but showed the worst performance when the KNN and SVM learners were used to do so.

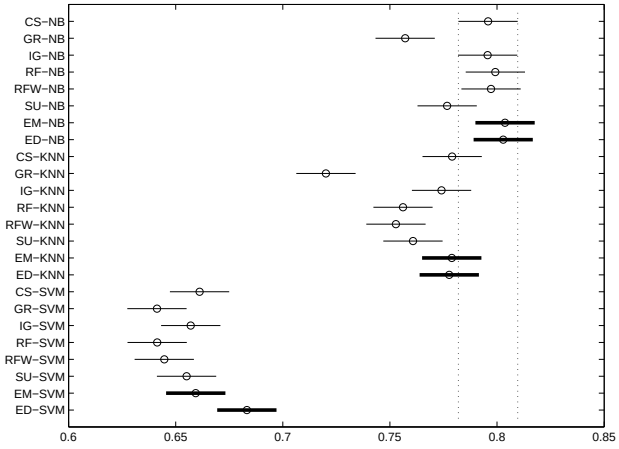
In summary, the ensemble technique with two different combination methods (mean and median), compared to each individual ranker, consistently performed very well, similar or superior to the other methods. In addition, the results indicate that the ensemble technique had better robustness. It is worthwhile to note that type of the ranker, classifier and software project (data set) played a key role for building



(a) Factor A



(b) Factor B

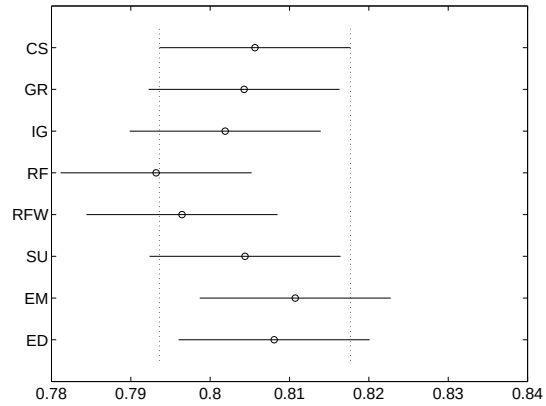


(c) Factor A×B

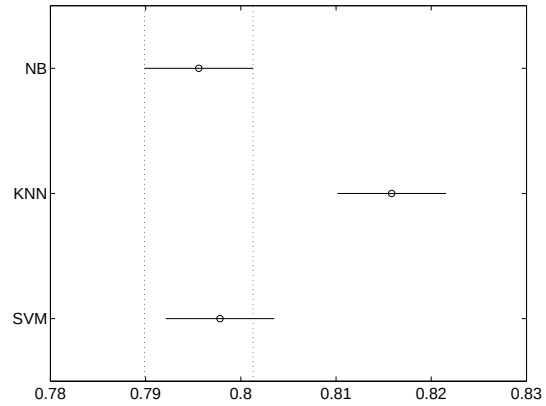
Fig. 1. LLTS: Multiple Comparison in Terms of AUC

TABLE V
TWO-WAY ANOVA TABLES FOR KC1 DATA SETS

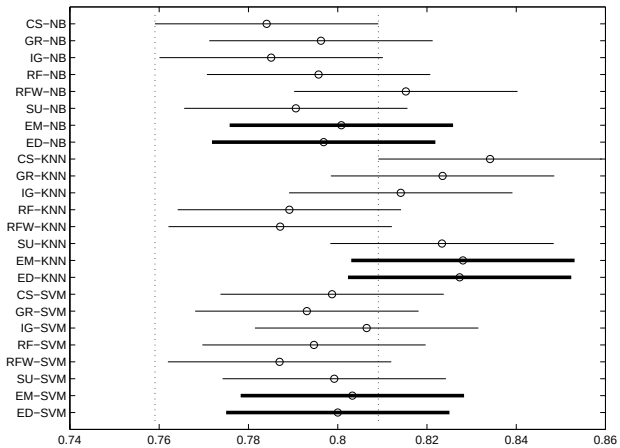
Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.0213	7	0.0030	1.07	0.3816
B	0.0593	2	0.0296	10.43	0
A×B	0.0752	14	0.0054	1.89	0.0242
Error	1.9764	696	0.0028		
Total	2.1322	719			



(a) Factor A



(b) Factor B



(c) Factor A×B

Fig. 2. KC1: Multiple Comparison in Terms of AUC

a software defect prediction model. Our recent studies [20], [23] also demonstrate that the reduced feature subsets can have better or similar prediction performance compared to the complete set of attributes (original dataset).

V. Conclusion

Filter-based ranker (feature selection) is one of the commonly used methodologies for feature selection. This work has presented detailed experiments using six frequently used rankers and our proposed ensemble approach and applied them to a very large telecommunications software system with four releases and a NASA project KC1 which includes three data sets. The classification models were built using NB, KNN and SVM learners. The classification accuracy was evaluated in terms of the AUC performance metric.

The experimental results show that the performance of rankers (feature selection techniques) may be significantly influenced by the data set and learner used in classification. It is frequently seen that one ranker performs well in a given data set for a particular classifier but becomes very bad when used on a different data set or with a different classifier. This study proposed and investigated an ensemble technique with two different combining methods using rank ordering of the features (mean or median) and results demonstrate that the ensemble technique performed better overall than any individual ranker and also possessed better robustness. The empirical study also shows that variations among rankers, learners and software projects significantly impacted the classification outcomes.

Future work may include experiments using additional data sets from the software engineering domain as well as from other application domains.

References

[1] W. Altidor, T. M. Khoshgoftaar, and J. V. Hulse, "An empirical study on wrapper-based feature ranking," in *Proceedings of 21st IEEE International Conference on Tools with Artificial Intelligence*, Newark, NJ, USA, Nov. 2-5 2009, pp. 75–82.

[2] K. Gao, T. M. Khoshgoftaar, and A. Napolitano, "Exploring software quality classification with a wrapper-based feature ranking technique," in *Proceedings of 21st IEEE International Conference on Tools with Artificial Intelligence*, Newark, NJ, USA, Nov. 2-5 2009, pp. 67–74.

[3] H. Wang, T. M. Khoshgoftaar, K. Gao, and N. Seliya, "Mining data from multiple software development projects," in *Proceedings of 9th IEEE International Conference on Data Mining - Workshops*, 2009, pp. 551–557.

[4] J. O. S. Olsson and D. W. Oard, "Combining feature selectors for text classification," in *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, New York, NY, USA, 2006, pp. 798–799.

[5] Y. Saeys, T. Abeel, and Y. Peer, "Robust feature selection using ensemble feature selection techniques," in *ECML PKDD '08: Proceedings of the European conference on Machine Learning and Knowledge Discovery in Databases - Part II*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 313–325.

[6] K. Lee, "Combining multiple feature selection methods," in *Mid-Atlantic Student Workshop on Programming Languages and Systems(MASPLAS'02)*, 2002, pp. 12.1–12.9.

[7] L. Rokach, B. Chizi, and O. Maimon, "Feature selection by combining multiple methods," in *Advances in Web Intelligence and Data Mining*, 2006, pp. 295–304.

[8] J. T. de Souza, N. Japkowicz, and S. Matwin, "Stochfs: A framework for combining feature selection outcomes through a stochastic process," in *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2005, pp. 667–674.

[9] S. Loscalzo, L. Yu, and C. Ding, "Consensus group stable feature selection," in *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, 2009, pp. 567–576.

[10] A. C. Cameron and P. K. Trivedi, *Regression Analysis of Count Data*. Cambridge University Press, 1998.

[11] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[12] M. A. Hall and G. Holmes, "Benchmarking attribute selection techniques for discrete class data mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 6, pp. 1437 – 1447, Nov/Dec 2003.

[13] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Proceedings of 9th International Workshop on Machine Learning*, 1992, pp. 249–256.

[14] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. Morgan Kaufmann, 2005.

[15] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in *Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence*, vol. 2, San Mateo, 1995, pp. 338–345.

[16] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine Learning*, vol. 6, no. 1, pp. 1573–0565, January 1991.

[17] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[18] Y. Jiang, J. Lin, B. Cukic, and T. Menzies, "Variance analysis in software fault prediction models," *Software Reliability Engineering, International Symposium on*, vol. 0, pp. 99–108, 2009.

[19] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, June 2006.

[20] K. Gao, T. M. Khoshgoftaar, and H. Wang, "An empirical investigation of filter attribute selection techniques for software quality classification," in *Proceedings of the 10th IEEE International Conference on Information Reuse and Integration*, Las Vegas, Nevada, August 10-12 2009, pp. 272–277.

[21] A. G. Koru, D. Zhang, K. E. Emam, and H. Liu, "An investigation into the functional form of the size-defect relationship for software modules," *IEEE Trans. Software Eng.*, vol. 35, no. 2, pp. 293–304, 2009.

[22] T. M. Khoshgoftaar, M. Golawala, and J. Van Hulse, "An empirical study of learning from imbalanced data using random forest," in *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence*, vol. 2, Washington, DC, USA, 2007, pp. 310–317.

[23] H. Wang, T. M. Khoshgoftaar, K. Gao, and N. Seliya, "High-dimensional software engineering data and feature selection," in *Proceedings of 21st IEEE International Conference on Tools with Artificial Intelligence*, Newark, NJ, USA, Nov. 2-5 2009, pp. 83–90.