8-1-2010

A Comparative Study of Threshold-based Feature Selection Techniques

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

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

Jason Van Hulse
Florida Atlantic University, jvanhulse@gmail.com

A Comparative Study of Threshold-based Feature Selection Techniques

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

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

Jason Van Hulse
Florida Atlantic University
jvanhulse@gmail.com

Abstract

Given high-dimensional software measurement data, researchers and practitioners often use feature (metric) selection techniques to improve the performance of software quality classification models. This paper presents our newly proposed threshold-based feature selection techniques, comparing the performance of these techniques by building classification models using five commonly used classifiers. In order to evaluate the effectiveness of different feature selection techniques, the models are evaluated using eight different performance metrics separately since a given performance metric usually captures only one aspect of the classification performance. All experiments are conducted on three Eclipse data sets with different levels of class imbalance. The experiments demonstrate that the choice of a performance metric may significantly influence the results. In this study, we have found four distinct patterns when utilizing eight performance metrics to order 11 threshold-based feature selection techniques. Moreover, performances of the software quality models either improve or remain unchanged despite the removal of over 96% of the software metrics (attributes).

Keywords: *performance metrics, threshold-based feature selection technique, software metrics, classification.*

1 Introduction

Given a set of software metrics (independent features or attributes) the objective of feature selection is to remove irrelevant or redundant features, which can then be discarded from the analysis. Reducing the number of features in a data set can lead to faster model training and improved classifier performance. Feature selection has been widely used and thoroughly researched [6, 7, 10]. The two general categories for feature selection are *filters* and *wrappers*. Filters are algorithms in which a feature subset is selected without involving any learner. Wrappers are algorithms that use feedback from a learning algorithm to determine which feature(s) to include in building a classification model. Another categorization for feature selection techniques is *feature ranking* and *feature subset selection techniques*. Feature ranking ranks the attributes according to their individual predictive power, while feature subset selection selects subsets of attributes that collectively have good predictive power. We consider filter-based feature ranking techniques in this study.

In this paper, we present our newly proposed threshold-based feature selection techniques (TBFS) which represent a substantial extension of the FAST algorithm proposed by Chen and Wasikowski [2]. Our technique is much more general than that of Chen and Wasikowski. Their procedure calculates a ROC curve by discretizing the distribution, while ours does not require discretization, making it more precise. Furthermore, there are 11 different versions of TBFS which are based on 11 different classifier performance metrics. TBFS can also be extended to incorporate additional metrics.

The 11 threshold-based feature selection techniques are evaluated using software measurement data in our case study, including three data sets of release 3.0 of a real-world software project, Eclipse [13]. In order to evaluate the classification performance of the TBFS techniques on the smaller subsets of attributes, several classification models are built using five commonly used classifiers. Since related literature lacks general agreement on which performance metrics should be used for evaluating classification performance [9, 8], learners are evaluated using eight performance metrics. The experiments demonstrate that the choice of a performance metric can significantly influence the conclusions. In addition, we have found four distinct patterns when we use eight performance metrics to order 11 threshold-based feature selection techniques.

The main contribution of this work is the presentation of a set of novel threshold-based feature selection techniques and an assessment and comparison of these feature selection methods using eight performance metrics and five commonly used classifiers.

The remainder of the paper is organized as follows. Section 2 explains our threshold-based feature selection methodology. Section 3 describes the learners, performance

978-0-7695-4161-7/10 $26.00 © 2010 IEEE
DOI 10.1109/GrC.2010.104

Algorithm 1: Threshold-based Feature Selection Algorithm

input :
1. Data set D with features $F^j, j = 1, \ldots, m$;
2. Each instance $x \in D$ is assigned to one of two classes $c(x) \in \{fp, nfp\}$;
3. The value of attribute F^j for instance x is denoted $F^j(x)$;
4. Threshold-based feature ranking technique $\omega \in$ {BFM, OR, PO, PR, GI, MI, KS, DV, BGM, 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 $F^j \mapsto \hat{F}^j = \frac{F^j - \min(F^j)}{\max(F^j) - \min(F^j)}$;
  Calculate metric ω using attribute $\hat{F}^j$, $\omega_i(\hat{F}^j)$.

Create feature ranking $\mathbb{R}$ using $\omega_i(\hat{F}^j) \forall j$.
Select features according to feature ranking $\mathbb{R}$ and a predefined threshold.

metrics and case study data sets used in this work, and presents the experimental results. Finally, we conclude the paper in Section 4 and provide suggestions for future work.

2 Threshold-based Feature Selection Techniques

Filter-based feature ranking techniques (filters) rank features independently without involving any learning algorithm. Eleven threshold-based feature selection techniques (TBFS) were developed and implemented by our research group within Weka [12]. 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 is then paired individually with the class attribute and the reduced two attribute data set is evaluated using 11 different 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 class is imbalanced. Therefore, we propose the use of performance metrics which allow for finding the optimal threshold.

The true positive (TPR), true negative (TNR), false positive (FPR), false negative (FNR), precision (PRE), negative predicted value (NPV) [11] rates can be calculated at each threshold $t \in [0, 1]$ relative to the normalized attribute $\hat{F}^j$. The threshold-based attribute ranking techniques we propose utilize these rates as described below.

- *Best F-measure (BFM)*: is a single value metric derived from the F-measure that originated from the field of information retrieval.

$$\text{BFM} = \max_{t \in [0,1]} \frac{(1+\beta^2) \times TPR(t) \times PRE(t)}{\beta^2 \times TPR(t) + PRE(t)}.$$

  β is set to 1 in this study. The maximum F-measure (BFM) is obtained when varying the decision threshold value between 0 and 1.

- *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:

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

- *Power (PO)*: is a measure that avoids common false positive cases while giving stronger preference for positive cases [5]. Power is defined as:

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

  where $k = 5$.

- *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 [5]. The probability ratio is defined as:

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

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

$$\begin{aligned}\text{GI} \quad = \quad & \min_{t \in [0,1]} [2PRE(t)(1 - PRE(t)) \\ & + 2NPV(t)(1 - NPV(t))].\end{aligned}$$

- *Mutual Information (MI)*: measures the mutual dependence of the two random variables. 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. 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.

Table 1. Performance Metrics using NB

Data	Filter	AUC	PRC	DFM	BFM	DGM	BGM	DAM	BAM
	BFM	0.8936	0.4402	0.5144	0.5448	0.7431	0.8535	0.7668	0.8544
	OR	0.8606	0.3998	0.4900	0.5195	0.7203	0.8384	0.7496	0.8391
	PO	0.8966	0.4479	0.5168	0.5431	0.7334	0.8607	0.7603	0.8611
	PR	0.8408	0.3863	0.4803	0.5109	0.7178	0.8047	0.7472	0.8085
Eclipse 3.0-10	GI	0.8439	0.3889	0.4804	0.5089	0.7151	0.8056	0.7452	0.8089
	MI	0.8715	0.4331	0.5101	0.5364	0.7342	0.8398	0.7603	0.8414
	KS	0.8713	0.4478	0.5118	0.5420	0.7330	0.8429	0.7596	0.8446
	DV	0.8986	0.4388	0.5214	0.5454	0.7453	0.8633	0.7689	0.8635
	BGM	0.8622	0.4412	0.5133	0.5466	0.7360	0.8365	0.7618	0.8384
	AUC	**0.8988**	0.4558	**0.5336**	**0.5570**	**0.7536**	**0.8637**	**0.7755**	**0.8640**
	PRC	0.8936	**0.4573**	0.5203	0.5493	0.7395	0.8561	0.7647	0.8562
	BFM	0.8863	0.6535	0.6495	0.6689	0.7670	0.8441	0.7846	0.8442
	OR	0.8830	0.6490	0.6412	0.6697	0.7633	0.8436	0.7812	0.8438
	PO	**0.8885**	**0.6600**	0.6527	0.6704	0.7666	0.8432	0.7848	0.8438
	PR	0.8810	0.6425	0.6336	0.6619	0.7613	0.8396	0.7789	0.8398
Eclipse 3.0-5	GI	0.8813	0.6447	0.6312	0.6598	0.7598	0.8409	0.7776	0.8412
	MI	0.8845	0.6502	0.6502	0.6708	0.7671	0.8430	0.7848	0.8431
	KS	0.8862	0.6539	0.6469	**0.6759**	0.7631	0.8417	0.7817	0.8420
	DV	0.8856	0.6527	**0.6555**	0.6698	**0.7687**	0.8427	**0.7865**	0.8427
	BGM	0.8858	0.6532	0.6429	0.6746	0.7608	0.8427	0.7797	0.8431
	AUC	0.8847	0.6474	0.6484	0.6664	0.7653	0.8448	0.7834	0.8450
	PRC	0.8851	0.6542	0.6547	0.6712	0.7671	**0.8456**	0.7853	**0.8458**
	BFM	0.8129	0.6427	0.5645	0.6314	0.6594	0.7647	0.7034	0.7676
	OR	0.8091	0.6472	0.5790	0.6321	0.6777	**0.7714**	0.7136	**0.7731**
	PO	**0.8163**	**0.6583**	0.5854	0.6347	0.6792	0.7684	0.7162	0.7710
	PR	0.8107	0.6416	0.5793	0.6289	0.6804	0.7645	0.7144	0.7666
Eclipse 3.0-3	GI	0.8108	0.6418	0.5800	0.6291	0.6809	0.7649	0.7148	0.7669
	MI	0.8124	0.6441	0.5625	0.6318	0.6582	0.7637	0.7024	0.7672
	KS	0.8109	0.6373	0.5595	0.6294	0.6536	0.7619	0.7002	0.7659
	DV	0.8140	0.6436	0.5643	0.6308	0.6585	0.7650	0.7031	0.7677
	BGM	0.8123	0.6420	0.5626	0.6306	0.6565	0.7632	0.7020	0.7663
	AUC	0.8101	0.6480	0.5819	0.6344	0.6770	0.7694	0.7143	0.7718
	PRC	0.8106	0.6515	**0.5878**	**0.6379**	**0.6814**	0.7689	**0.7177**	0.7716

Table 2. Performance Metrics using MLP

Data	Filter	AUC	PRC	DFM	BFM	DGM	BGM	DAM	BAM
	BFM	0.9034	0.5146	0.4143	0.5558	0.5534	0.8416	0.6510	0.8447
	OR	0.8817	0.4884	0.3994	0.5247	0.5395	0.8244	0.6428	0.8254
	PO	0.8982	0.5262	0.4333	0.5692	0.5692	0.8385	0.6596	0.8409
	PR	0.8528	0.4341	0.3556	0.4798	0.5003	0.7930	0.6222	0.7961
Eclipse 3.0-10	GI	0.8512	0.4446	0.3531	0.4828	0.4957	0.7939	0.6202	0.7974
	MI	0.8922	0.4920	0.4044	0.5312	0.5410	0.8338	0.6444	0.8367
	KS	0.8962	0.5226	0.4467	0.5555	0.5785	0.8407	0.6648	0.8431
	DV	0.9030	**0.5321**	**0.4574**	**0.5697**	**0.5866**	0.8416	**0.6699**	0.8440
	BGM	0.8945	0.5182	0.4357	0.5563	0.5684	0.8393	0.6601	0.8414
	AUC	**0.9097**	0.5289	0.4526	0.5588	0.5861	**0.8544**	0.6681	**0.8564**
	PRC	0.8981	0.5117	0.4293	0.5532	0.5662	0.8374	0.6572	0.8408
	BFM	0.9183	0.7622	0.6306	0.6977	0.7202	0.8661	0.7536	0.8667
	OR	0.9178	0.7555	0.6256	0.6944	0.7172	0.8645	0.7511	0.8651
	PO	**0.9226**	0.7606	0.6310	0.6920	0.7196	0.8614	0.7534	0.8618
	PR	0.9147	0.7520	0.6310	0.6905	0.7213	0.8566	0.7543	0.8574
Eclipse 3.0-5	GI	0.9147	0.7530	0.6337	0.6952	0.7235	0.8593	0.7559	0.8602
	MI	0.9162	0.7617	0.6286	**0.7000**	0.7187	0.8622	0.7525	0.8632
	KS	0.9194	0.7604	0.6248	0.6990	0.7193	0.8651	0.7522	0.8656
	DV	0.9217	0.7650	0.6219	0.6965	0.7122	0.8690	0.7477	0.8695
	BGM	0.9178	0.7594	0.6284	0.6982	0.7209	0.8649	0.7536	0.8654
	AUC	0.9213	**0.7674**	**0.6340**	0.6998	**0.7241**	**0.8690**	**0.7563**	**0.8695**
	PRC	0.9221	0.7657	0.6269	0.6967	0.7180	0.8671	0.7517	0.8676
	BFM	0.8778	0.7511	0.6073	0.6987	0.7027	0.8180	0.7323	0.8195
	OR	0.8742	0.7510	0.6225	0.6930	0.7098	0.8154	0.7408	0.8159
	PO	0.8848	0.7624	0.6313	0.7110	0.7146	**0.8297**	0.7443	**0.8302**
	PR	0.8796	0.7575	0.6325	0.6981	0.7228	0.8194	0.7483	0.8202
Eclipse 3.0-3	GI	0.8793	0.7577	0.6323	0.6978	0.7227	0.8190	0.7482	0.8199
	MI	0.8827	0.7604	0.6354	0.7105	0.7209	0.8253	0.7476	0.8256
	KS	0.8857	0.7633	0.6346	0.7124	0.7222	0.8254	0.7482	0.8261
	DV	0.8780	0.7512	0.6134	0.6992	0.7079	0.8182	0.7360	0.8189
	BGM	**0.8862**	**0.7657**	**0.6359**	**0.7133**	**0.7230**	0.8278	**0.7487**	0.8284
	AUC	0.8835	0.7596	0.6260	0.7087	0.7108	0.8293	0.7414	0.8296
	PRC	0.8850	0.7622	0.6335	0.7106	0.7179	0.8297	0.7463	0.8300

- *Deviance (DV)*: is the minimum residual sum of squares based on a threshold t. That is, it measures the sum of the squared errors from the mean class given a partitioning of the space based on the threshold t.

- *Best Geometric Mean (BGM)*: 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:

$$\text{BGM} = \max_{t \in [0,1]} \sqrt{TPR(t) \times TNR(t)} \qquad (1)$$

- *Area Under ROC (Receiver Operating Characteristic) Curve* (AUC): has been widely used to measure classification model performance [4]. 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.

In this study, AUC, PRC, BFM, and BGM serve as both aids to the feature ranking process and the final inductive algorithm evaluation process (see Section 3.2).

3 Experiments

3.1 Classifiers

Classifiers are built with five well-known classification algorithms [12] including naïve Bayes (NB), multilayer perceptron (MLP), k-nearest neighbors (KNN), support vector machine (SVM) [3] and logistic regression (LR). These were selected because of their common use in data mining applications. Unless stated otherwise, the default parameter settings are used for the learners as specified in Weka [12]. Parameter settings were changed only when a significant improvement in performance based on preliminary experimentation was obtained. For the KNN classifier, 5 neighbors were used and the *distanceWeighting* parameter was set to "Weight by 1/distance". For the MLP learner, *hiddenLayers* was changed to 3 to define a network with one hidden layer containing three nodes, and *validationSetSize* was changed to 10 to cause the classifier to leave 10% of the training data aside to be used as a validation set to determine when to stop the iterative training process. For SVM, the complexity constant *c* was changed from 1.0 to 5.0 and *buildLogisticModels* was enabled.

3.2 Classifier Performance Metrics

Eight performance metrics are used in the study including AUC, PRC, DFM (Default F-measure corresponds to a decision threshold value of 0.5), BFM (Best F-Measure which is the largest value of F-measure when varying the decision threshold value between 0 and 1), DGM (Default Geometric Mean), BGM (Best Geometric Mean), DAM

Table 3. Performance Metrics using KNN

Data	Filter	AUC	PRC	DFM	BFM	DGM	BGM	DAM	BAM
	BFM	0.9125	0.4544	0.4138	0.5176	0.5699	0.8600	0.6569	0.8616
	OR	0.8935	0.4485	0.3630	0.4743	0.5237	0.8446	0.6316	0.8451
	PO	**0.9206**	0.4668	0.4010	**0.5318**	0.5607	**0.8724**	0.6518	**0.8736**
	PR	0.8653	0.3797	0.3077	0.4014	0.4644	0.8145	0.6030	0.8166
Eclipse 3.0-10	GI	0.8683	0.3955	0.3360	0.4094	0.4889	0.8131	0.6153	0.8162
	MI	0.8993	0.4484	0.4049	0.4982	0.5620	0.8458	0.6532	0.8466
	KS	0.9004	0.4614	0.4030	0.5128	0.5614	0.8498	0.6519	0.8505
	DV	0.9165	0.4631	0.3951	0.5211	0.5558	0.8635	0.6493	0.8651
	BGM	0.8955	0.4546	0.4029	0.5056	0.5578	0.8425	0.6500	0.8430
	AUC	0.9193	**0.4825**	0.4055	0.5110	0.5613	0.8695	0.6523	0.8702
	PRC	0.9206	0.4672	**0.4241**	0.5076	**0.5853**	0.8715	**0.6656**	0.8728
	BFM	0.9229	0.7456	0.6312	0.6728	0.7353	0.8482	0.7620	0.8494
	OR	0.9180	0.7229	0.6211	0.6681	0.7253	0.8436	0.7549	0.8450
	PO	0.9202	0.7391	0.6267	0.6654	0.7321	0.8445	0.7597	0.8457
	PR	0.9157	0.7214	0.6147	0.6613	0.7222	0.8377	0.7520	0.8387
Eclipse 3.0-5	GI	0.9161	0.7235	0.6146	0.6620	0.7227	0.8387	0.7522	0.8402
	MI	0.9223	0.7431	0.6519	0.6755	0.7489	0.8454	0.7731	0.8471
	KS	0.9221	0.7374	0.6488	0.6771	0.7509	0.8448	0.7740	0.8461
	DV	0.9240	0.7452	0.6336	0.6746	0.7351	**0.8512**	0.7623	**0.8525**
	BGM	**0.9246**	**0.7460**	**0.6524**	**0.6852**	**0.7516**	0.8493	**0.7749**	0.8513
	AUC	0.9235	0.7421	0.6490	0.6773	0.7448	0.8494	0.7702	0.8508
	PRC	0.9224	0.7357	0.6250	0.6713	0.7330	0.8484	0.7599	0.8493
	BFM	0.8889	0.7368	0.6481	0.6854	0.7440	0.8101	0.7611	0.8108
	OR	0.8851	0.7262	0.6539	0.6899	0.7459	0.8118	0.7638	0.8125
	PO	**0.8981**	0.7521	**0.6717**	**0.7145**	**0.7624**	**0.8263**	**0.7766**	**0.8271**
	PR	0.8847	0.7183	0.6466	0.6826	0.7429	0.8117	0.7603	0.8124
Eclipse 3.0-3	GI	0.8845	0.7183	0.6466	0.6820	0.7429	0.8110	0.7603	0.8118
	MI	0.8898	0.7402	0.6515	0.6944	0.7451	0.8164	0.7626	0.8170
	KS	0.8921	0.7495	0.6501	0.7025	0.7440	0.8206	0.7617	0.8214
	DV	0.8915	0.7377	0.6515	0.6912	0.7466	0.8164	0.7633	0.8169
	BGM	0.8930	0.7508	0.6571	0.7072	0.7506	0.8238	0.7668	0.8244
	AUC	0.8951	**0.7562**	0.6645	0.7009	0.7535	0.8190	0.7701	0.8197
	PRC	0.8974	0.7516	0.6709	0.7100	0.7610	0.8248	0.7757	0.8257

Table 4. Performance Metrics using LR

Data	Filter	AUC	PRC	DFM	BFM	DGM	BGM	DAM	BAM
	BFM	0.9127	0.5661	0.4838	0.6060	**0.6241**	0.8616	**0.6908**	0.8623
	OR	0.8745	0.5025	0.4041	0.5416	0.5628	0.8326	0.6531	0.8341
	PO	0.9106	**0.5679**	0.4740	**0.6062**	0.6138	0.8611	0.6849	0.8621
	PR	0.8705	0.4422	0.3819	0.4778	0.5485	0.8237	0.6448	0.8252
Eclipse 3.0-10	GI	0.8687	0.4486	0.3833	0.4839	0.5470	0.8257	0.6441	0.8268
	MI	0.9047	0.5407	0.4686	0.5687	0.6087	0.8527	0.6814	0.8531
	KS	0.9050	0.5617	0.4741	0.5728	0.6098	0.8575	0.6818	0.8593
	DV	**0.9159**	0.5659	0.4742	0.5981	0.6125	**0.8651**	0.6840	**0.8658**
	BGM	0.9045	0.5605	**0.4882**	0.5801	0.6230	0.8525	0.6902	0.8545
	AUC	0.9097	0.5617	0.4724	0.5677	0.6045	0.8604	0.6788	0.8607
	PRC	0.9072	0.5502	0.4650	0.5796	0.5997	0.8589	0.6762	0.8595
	BFM	0.9406	0.7952	0.6697	0.7297	0.7564	0.8854	0.7800	0.8860
	OR	**0.9446**	0.7968	0.6768	0.7299	0.7638	**0.8874**	0.7857	**0.8879**
	PO	0.9375	0.7891	0.6423	0.7192	0.7367	0.8817	0.7644	0.8818
	PR	0.9379	0.7918	0.6765	0.7273	0.7642	0.8821	0.7859	0.8827
Eclipse 3.0-5	GI	0.9372	0.7912	**0.6780**	**0.7316**	**0.7660**	0.8809	**0.7873**	0.8814
	MI	0.9383	**0.7971**	0.6636	0.7227	0.7516	0.8859	0.7762	0.8860
	KS	0.9349	0.7907	0.6579	0.7141	0.7479	0.8824	0.7733	0.8827
	DV	0.9411	0.7944	0.6671	0.7251	0.7544	0.8837	0.7784	0.8841
	BGM	0.9376	0.7963	0.6600	0.7201	0.7483	0.8832	0.7738	0.8837
	AUC	0.9394	0.7905	0.6685	0.7303	0.7577	0.8872	0.7807	0.8875
	PRC	0.9421	0.7911	0.6686	0.7227	0.7577	0.8853	0.7807	0.8857
	BFM	0.9097	0.7822	0.6361	0.7314	0.7182	0.8479	0.7461	0.8480
	OR	0.9061	0.7816	0.6440	0.7243	0.7235	0.8408	0.7506	0.8409
	PO	0.9095	0.7863	0.6437	0.7288	0.7224	0.8471	0.7500	0.8472
	PR	0.9053	0.7868	0.6561	0.7238	0.7348	0.8410	0.7588	0.8413
Eclipse 3.0-3	GI	0.9054	**0.7869**	**0.6566**	0.7241	**0.7352**	0.8408	**0.7591**	0.8411
	MI	0.9076	0.7807	0.6382	0.7270	0.7186	0.8412	0.7469	0.8412
	KS	0.9077	0.7805	0.6356	0.7263	0.7164	0.8445	0.7453	0.8446
	DV	**0.9115**	0.7830	0.6368	0.7335	0.7191	0.8478	0.7467	0.8479
	BGM	0.9092	0.7821	0.6382	0.7260	0.7188	0.8456	0.7470	0.8457
	AUC	0.9089	0.7828	0.6456	0.7294	0.7258	0.8477	0.7520	0.8478
	PRC	0.9088	0.7826	0.6472	**0.7344**	0.7273	**0.8509**	0.7531	**0.8510**

(Default Arithmetic Mean), and BAM (Best Arithmetic Mean). The arithmetic mean uses the arithmetic mean of the true positive rate and true negative rate.

Note that the metrics used to measure the performance of the classifiers is completely independent from the metrics in the TBFS algorithm. For example, AUC is used both to select the most predictive subset of features using TBFS and AUC is also used to evaluate the classifiers constructed using this set of features. However the AUC-based TBFS technique can also be evaluated using other classifier performance metrics such as BAM or DGM.

3.3 Data Sets

Experiments conducted in this study use software metrics and defect data collected from release 3.0 of a real-world software project, the Eclipse project [13]. 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 λ, which separates fp from nfp packages by classifying packages with λ or more post-release defects as fp and the remaining as nfp. We chose three post-release defects thresholds $\lambda \in \{10, 5, 3\}$ to determine the defective instances. The derived data sets contain 208 attributes and 661 instances. We use these thresholds to obtain three different levels of class imbalance for Eclipse data sets. The proportions of fp modules of the three data sets are 6.2%, 14.83% and 23.75%.

3.4 Experimental Results and Analysis

In the experiments, ten runs of five-fold cross-validation are performed. For each of the five folds, one fold is used as test data while the other four are used as the training data. First, we rank the attributes using the 11 threshold-based feature ranking techniques separately. Once the attributes are ranked, the top $\lceil \log_2 208 \rceil = 8$ (there are 208 independent features) attributes are selected (as well as the class attribute) to yield final training data. Selecting eight attributes was deemed reasonable for these experiments, and space considerations keep us from presenting results with other parameter choices. This final training data is then used to build the classification model, the resulting model is applied to the test-fold, and the eight performance metrics are calculated.

The classification models are evaluated in terms of the eight performance metrics separately. All the results are reported in Table 1 through Table 4 (the detailed results for SVM are omitted for space considerations). The classifier performance metrics are provided in the columns, and the individual TBFS filters are listed as rows. Again note that AUC is listed both as a filter and performance metric, but the context should be clear given the explanations provided above. 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 **boldfaced** print. Filter performance varies depending on both the learner and the performance metric. Table 5 presents the number of times each filter performed best relative to each performance metric, summarized across all five classifiers and three data sets together. No particular TBFS filter domi-

Table 5. Summary of Optimal TBFS Filter

Filter	AUC	PRC	DFM	BFM	DGM	BGM	DAM	BAM	Total	% of Total
BFM	0	1	1	0	2	1	2	1	8	6.7%
OR	1	0	0	0	0	2	0	2	5	4.2%
PO	7	3	1	4	1	4	1	4	25	20.8%
PR	0	0	0	0	0	0	0	0	0	0.0%
GI	0	1	2	1	3	0	3	0	10	8.3%
MI	0	1	1	1	0	0	0	0	3	2.5%
KS	0	0	0	2	0	0	0	0	2	1.7%
DV	2	1	2	1	2	2	2	2	14	11.7%
BGM	2	3	4	2	3	0	3	0	17	14.2%
AUC	3	4	2	1	2	4	2	4	22	18.3%
PRC	0	1	2	3	2	2	2	2	14	11.7%

nates the others, but generally speaking, we can conclude that PO and AUC are most often the best technique, while PR, KS, MI and OR are rarely optimal.

A two-way ANOVA [1] was performed for each of the eight performance metrics separately. The two factors are Factor A, in which eleven threshold-based rankers were considered, and Factor B, in which five classifiers were included. In this ANOVA test, the results from all three data sets were taken into account together. A significance level of $\alpha = 5\%$ was used for all statistical tests. The ANOVA results are presented in Table 6. Focusing on Factor A, the test results indicate that for each performance metric, there was a significant difference between the average values of the 11 TBFS methods. For all eight performance metrics, the p-value is less than 5%, although the significance varies substantially among metrics (e.g., PRC in Table 6(b) compared to AUC in Table 6(a)). Multiple comparison tests were conducted on Factor A, since this study mainly focuses on the attribute selection techniques and their classifier performance evaluation. Both ANOVA and multiple comparison tests are implemented in MATLAB. An exhaustive discussion of Factor B is avoided due to space consideration.

The performance of the threshold-based filters was ranked from best to worst for each performance metric as shown in Table 7. Each filter is labeled with a superscript. The filters labeled with the same superscripts implies that they were from same performance group, in which no statistically significant difference was found between filters. Some findings can be summarized from Table 7: (1) Four distinct groups of results were found when we order 11 filters based on eight performance metrics (over all the classifiers built): (a) AUC, BAM, and BGM; (b) PRC; (c) BFM; and (d) DFM, DAM, and DGM. (2) Among the 11 threshold-based feature selection techniques, the performance of AUC-based filter performed best overall. PR- and GI-based filters performed worst, followed by OR regardless of performance metrics. While PO was most often the optimal technique (Table 5), this is somewhat offset by relatively worse performance in other situations.

Table 8 presents performances of the defect classification models built with the complete set of features. Comparing these results to Tables 1 through 4, classification models

Table 6. Analysis of Variance

Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.09	10	0.009	11.9	0
B	0.9073	4	0.2268	300.00	0
Error	1.2362	1635	0.0008		
Total	2.2336	1649			

(a) AUC

Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.2857	10	0.0286	1.85	0.0481
B	3.901	4	0.9753	63.11	0
Error	25.2659	1635	0.0155		
Total	29.4527	1649			

(b) PRC

Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.2075	10	0.0208	2.06	0.0249
B	0.1645	4	0.0411	4.08	0.0027
Error	16.4901	1635	0.01009		
Total	16.8621	1649			

(c) DFM

Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.2642	10	0.0264	4.14	0
B	0.9797	4	0.2449	38.38	0
Error	10.4327	1635	0.0064		
Total	11.6766	1649			

(d) BFM

Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.1263	10	0.0126	2.18	0.0165
B	0.4417	4	0.1104	19.09	0
Error	9.458	1635	0.0058		
Total	10.026	1649			

(e) DGM

Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.0908	10	0.0091	10.61	0
B	0.5093	4	0.1273	148.73	0
Error	1.3997	1635	0.0009		
Total	1.9998	1649			

(f) BGM

Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.0453	10	0.0045	1.95	0.0348
B	0.1497	4	0.0374	16.13	0
Error	3.7929	1635	0.0023		
Total	3.9879	1649			

(g) DAM

Source	Sum Sq.	d.f.	Mean Sq.	F	p-value
A	0.0847	10	0.0085	10.54	0
B	0.4847	4	0.1212	150.76	0
Error	1.3141	1635	0.0008		
Total	1.8835	1649			

(h) BAM

Table 7. Rank of Filters (Factor A)

Metric	Rank of TBFS Filters (best $\longrightarrow$ worst)										
AUC	AUC^{a}	DV^{ab}	PRC^{ab}	PO^{ab}	BFM^{abc}	MI^{bcd}	KS^{bcd}	BGM^{cd}	OR^{d}	GI^{e}	PR^{e}
PRC	AUC^{a}	PRC^{a}	PO^{a}	DV^{a}	KS^{a}	BFM^{a}	BGM^{a}	MI^{ab}	OR^{abc}	GI^{bc}	PR^{c}
DFM	AUC^{a}	BGM^{ab}	PRC^{ab}	BFM^{ab}	DV^{ab}	KS^{ab}	PO^{ab}	MI^{abc}	OR^{bcd}	GI^{cd}	PR^{d}
BFM	PO^{a}	PRC^{ab}	DV^{ab}	AUC^{ab}	BFM^{ab}	BGM^{ab}	KS^{ab}	MI^{ab}	OR^{bc}	GI^{c}	PR^{c}
DGM	AUC^{a}	BGM^{ab}	PRC^{ab}	BFM^{ab}	DV^{ab}	PO^{ab}	MI^{ab}	KS^{ab}	OR^{bc}	GI^{c}	PR^{c}
BGM	AUC^{a}	PRC^{a}	PO^{ab}	DV^{ab}	BFM^{abc}	KS^{abcd}	MI^{bcd}	BGM^{cd}	OR^{d}	GI^{e}	PR^{e}
DAM	AUC^{a}	BGM^{ab}	PRC^{ab}	BFM^{ab}	DV^{ab}	PO^{abc}	MI^{abc}	KS^{abc}	OR^{bcd}	GI^{cd}	PR^{d}
BAM	AUC^{a}	PRC^{a}	PO^{ab}	DV^{abc}	BFM^{abc}	KS^{abcd}	MI^{bcd}	BGM^{cd}	OR^{d}	GI^{e}	PR^{e}

Table 8. Performance of Full Data Sets

Data	Learner	AUC	PRC	DFM	BFM	DGM	BGM	DAM	BAM
Eclipse 3.0-10	NB	0.8685	0.3304	0.4710	0.4868	0.7666	0.8228	0.7811	0.8238
	MLP	0.7697	0.3836	0.4000	0.4539	0.5772	0.7435	0.6598	0.7526
	KNN	0.7325	0.3565	0.1959	0.4203	0.3364	0.7322	0.5556	0.7388
	SVM	0.8030	0.4067	0.4169	0.4506	0.6172	0.7680	0.6809	0.7697
	LR	0.6596	0.1993	0.2995	0.3355	0.5935	0.6600	0.6527	0.6764
Eclipse 3.0-5	NB	0.8648	0.5113	0.5707	0.5934	0.7274	0.8104	0.7485	0.8111
	MLP	0.8422	0.6065	0.5750	0.6113	0.7072	0.8042	0.7383	0.8069
	KNN	0.8114	0.5880	0.5156	0.5828	0.6239	0.7694	0.6869	0.7729
	SVM	0.8897	0.6741	0.6150	0.6489	0.7498	0.8369	0.7691	0.8373
	LR	0.7632	0.4294	0.4974	0.5208	0.7016	0.7348	0.7198	0.7388
Eclipse 3.0-3	NB	0.8119	0.5684	0.5475	0.6025	0.6617	0.7426	0.6972	0.7471
	MLP	0.8076	0.6408	0.5640	0.6102	0.6776	0.7483	0.7091	0.7534
	KNN	0.7770	0.5828	0.5016	0.5546	0.6299	0.7201	0.6711	0.7218
	SVM	0.8673	0.7216	0.6385	0.6667	0.7364	0.7953	0.7548	0.7959
	LR	0.7452	0.4984	0.5245	0.5431	0.6804	0.6960	0.6909	0.7035

built using smaller subsets of attributes selected with TBFS had better performances than those built with a complete set of attributes.

4 Conclusion

Numerous feature selection techniques have been proposed in the data mining and machine learning domains. The aim of feature selection is to remove irrelevant and redundant features with the primary objective of improving classifier performance. In this study, we present our newly proposed threshold-based feature selection techniques and compare their performance to classifiers constructed without the use of feature selection. Three data sets from a real-world software project were used, with different levels of class imbalance. Classification models were constructed using five commonly used methodologies and are evaluated using eight performance metrics. Threshold-based feature selection using the AUC parameter was shown to provide generally good performance, though the optimal filter often varied depending on the classifier, dataset, and performance metric. OR-, GI-, and PR-based filters in particular did not perform as well as the other filters. Furthermore, we have found four distinct patterns when utilizing eight performance metrics to order 11 threshold-based feature selection techniques. A final inference is that even after removing 96% of the available software metrics, the classification models were not adversely affected; in fact, in 95% of the cases the results were better.

Future work will involve conducting additional empirical studies with data from other software projects and application domains. Additional experiments should also be conducted to analyze the impact of the number of selected features.

References

[1] M. L. Berenson, M. Goldstein, and D. Levine. *Intermediate Statistical Methods and Applications: A Computer Package Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2 edition, 1983.

[2] X.-w. Chen and M. Wasikowski. Fast: a roc-based feature selection metric for small samples and imbalanced data classification problems. In *KDD '08: Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, pages 124–132, New York, NY, USA, 2008. ACM.

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

[4] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, June 2006.

[5] G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.

[6] 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*, pages 272–277, Las Vegas, Nevada, August 10-12 2009.

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

[8] N. Seliya, T. M. Khoshgoftaar, and J. Van Hulse. A study on the relationships of classifier performance metrics. In *Proceedings of 21st IEEE International Conference on Tools with Artificial Intelligence*, pages 59–66, 2009.

[9] M. Sokolova, N. Japkowicz, and S. Szpakowicz. Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation. *AI 2006: Advances in Artificial Intelligence*, pages 1015–1021, 2006.

[10] 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*, pages 83–90, Newark, NJ, USA, Nov. 2-5 2009.

[11] G. M. Weiss and F. Provost. Learning when training data are costly: the effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, (19):315–354, 2003.

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

[13] T. Zimmermann, R. Premraj, and A. Zeller. Predicting defects for eclipse. In *ICSEW '07: Proceedings of the 29th International Conference on Software Engineering Workshops*, page 76, Washington, DC, USA, 2007. IEEE Computer Society.