Detecting Fault Modules Applying Feature Selection to Classifiers

D. Rodríguez[1], R. Ruiz[2], J. Cuadrado-Gallego[1], J. Aguilar-Ruiz[2]

[1]Dept of Computer Science
The University of Alcalá
Ctra Barcelona Km 37,1
28805 Alcalá de Henares, Madrid, Spain

[2]Dept of Computer Science
University Pablo de Olavide
Ctra. Utrera km. 1
41013 Sevilla, Spain

{daniel.rodriguezg,jjcg}@uah.es, {robertoruiz,aguilar}@upo.es

Abstract

At present, automated data collection tools allow us to collect large amounts of information, not without associated problems. This paper, we apply feature selection to several software engineering databases selecting attributes with the final aim that project managers can have a better global vision of the data they manage. In this paper, we make use of attribute selection techniques in different datasets publicly available (PROMISE repository), and different data mining algorithms for classification to defect faulty modules. The results show that in general, smaller datasets with less attributes maintain or improve the prediction capability with less attributes than the original datasets.

1 Introduction

Currently, organizations can collect large amounts of data and attributes from version management systems, Integrated Development Environments and other metrics tools about modules and components. Some of these repositories have been made available through the PROMISE repository[1]. However, a large number of attributes could make more difficult the application of the collected data with techniques such as regression or classification. In this paper, we apply Feature Selection (FS) or Feature Subset Selection for identifying the most relevant attributes from several datasets from the PROMISE repository. The need of applying FS includes the following points:

- A reduced volume of data allows different data mining or searching techniques to be applied.
- Irrelevant and redundant attributes can generate less accurate and more complex models. Furthermore, data mining algorithms can be executed faster.
- It is possible to avoid the collection of data for those irrelevant and redundant attributes in the future.

Until present few authors have investigated the application of FS to software engineering datasets areas such as cost estimation or quality. Among these works, Chen et al [2] have analyzed the application of feature selection using wrappers to the problem of cost estimation. They also concluded that the reduced dataset could improve the estimation. Kirsopp and Shepperd [9] have also analyzed the application of feature subset selection to cost estimation reaching with similar conclusions.

The paper is organized as follows. Section 2 explains the background behind Feature Selection, learning Classifiers using Feature Selection and common techniques used in data mining for evaluation. Section 3 describes the experimental results using several datasets from the PROMISE repository. Finally, conclusions and future work are commented.

[1]http://promisedata.org/

1-4244-1500-4/07/$25.00 ©2007 IEEE

2 Feature Selection (FS)

FS algorithms designed with different evaluation criteria broadly fall into two categories [21, 14, 13, 3, 1, 12, 5]:

- The *filter model* relies on general characteristics of the data to evaluate and select feature subsets without involving any data mining algorithm.
- The *wrapper model* requires one predetermined mining algorithm and uses its performance as the evaluation criterion. It searches for features better suited to the mining algorithm aiming to improve mining performance, but it also tends to be more computationally expensive than filter model [11, 12].

Feature subset algorithms search through candidate feature subsets guide by a certain evaluation measure [13] which captures the goodness of each subset. An optimal (or near optimal) subset is selected when the search stops.

Some existing evaluation measures that have been shown effective in removing both irrelevant and redundant features include the consistency measure [4], the correlation measure [7] and the estimated accuracy of a learning algorithm [11].

- Consistency measure attempts to find a minimum number of features that separate classes as consistently as the full set of features can. An inconsistency is defined as to instances having the same feature values but different class labels.
- Correlation measure evaluates the goodness of feature subsets based on the hypothesis that good feature subsets contain features highly correlated to the class, yet uncorrelated to each other.
- Wrapper-based attribute selection uses the target learning algorithm to estimate the worth of attribute subsets. The feature subset selection algorithm conducts a search for a good subset using the induction algorithm itself as part of the evaluation function.

Langley [12] notes that FS algorithms that search through the space of feature subsets must address four main issues: (i) the starting point of the search, (ii) the organization of the search, (iii) the evaluation of features subsets and (iv) the criterion used to terminate the search. Different algorithms address theses issues differently.

It is impractical to look at all possible feature subsets, even if the size is small.They can be classified into those that add features to an initially empty set (*forward selection*) and those that remove features from an initially complete set (*backward elimination*). Hybrids both add and remove features as the algorithm progresses. Forward selection is much faster than backward elimination and therefore scales better to large data sets. A wide range of search strategies can be used: best–first, branch–and–bound, simulated annealing, genetic algorithms (see Kohavi and John [11] for a review). In this paper we use forward selection.

Recently, Yu and Liu [20] proposed a new framework of FS, fast correlation–based filter algorithm (FCBF) which uses correlation measure to obtain relevant genes and to remove redundancy. There are other methods based on relevance and redundancy concepts. It is based on the concept of Markov blanket, where M is formed by only one attribute, and gradually eliminates redundant attributes with respect to M from the first to the final attribute of an ordered list.

2.1 Learning Classifiers using FS

Many software engineering problems like defect detection, cost estimation can be viewed as classification problems. A classifier resembles a function in the sense that it attaches a value (or a range or a description) to a set of attribute values. A classification function will produce a set of descriptions based on the characteristics of the instances for each attribute. Such class descriptions are the output of the classifier's function.

In order to compare the effectiveness of FS, feature sets chosen by each technique are tested with two different and well-known types of classifiers: a probabilistic classifier (naïve Bayes) and a decision tree classifier (C4.5). These algorithms have been selected because they represent different approaches to learning and for their long standing tradition in classification studies.

- The naïve Bayes [16] algorithm uses the Bayes theorem to predict the class for each case, assuming that the predictive attributes are independent given a category. A Bayesian classifier assigns a set of

attributes $A_1, A_2, \ldots, A_n$ to a class C such that $P(C|A_1, A_2, \ldots, A_n)$ is maximum.

- C4.5 [17]. A decision tree is constructed in a top-down approach. The leaves of the tree correspond to classes, nodes correspond to features, and branches to their associated values. C4.5 uses the gain ratio criterion to select the attribute to be at every node of the tree.

2.2 Evaluation

When evaluating the prediction accuracy of the classification methods we described above, it is important not to use the same instances for training and evaluation. Feature selection methods will perform well on examples they have seen during training. To get a realistic estimate of performance of the classifier, we must test it on examples that did not appear in the training set.

A common method to test accuracy in such situations is cross-validation. To apply this method, we partition the data into *k* sets of samples, $C_1, \ldots, C_k$ (typically, these will be of roughly the same size). Then, we construct a data set $D_i = D - C_i$, and test the accuracy of f_{D_i} on the samples in C_i. Having done this for all $1 \leq i \leq k$ we estimate the accuracy of the method by averaging the accuracy over the *k* cross-validation trials.

Cross-validation has several important properties. First, the training set and the test set in each trial are disjoint. Second, the classifier is tested on each sample exactly once. Finally, the training set for each trial is $(k-1)/k$ of the original data set. Thus, for large *k*, we get a relatively unbiased estimate of the classifier behavior given a training set of size *m* [10].

Another common way to measure the goodness of data mining applications is through the $f - measure$ [19].

3 Experimental Results

In this paper, we have applied feature selection to the CM1, JM1, KC1, KC2, and PC1 datasets available in the PROMISE repository [18], to generate models for defect classification. These datasets were created from projects carried out at NASA and collected under their metrics program[2].

All datasets contain 22 attributes composed of 5 different lines of code measure, 3 McCabe metrics [15], 4 base Halstead measures [8], 8 derived Halstead measures [8], a branch-count, and the last attribute is 'problems' with 2 classes (false or true, whether the module has reported defects). For a comprehensive coverage and explanation of the metrics, we refer to Fenton and Pfleeger [6]. The number of instances, however, varies among the datasets: CM1 contains 498 instances, JM1 (10885), KC1 (2109), KC2 (522) and PC1 is composed of 1109 instances.

As stated previously, FS can be grouped into filter or wrapper depending on whether the classifier is used to find the feature subset. In this paper, for the filter model, we have used consistency and correlation measures; for the wrapper-method, two standard classifiers have been applied: naïve Bayes and C4.5 classifiers.

The experiments were conducted using algorithms implemented in the WEKA environment [19], either using the Explorer or the Experimenter tools provided. We must take into account that the proper way to conduct a cross–validation for feature selection is to avoid using a fixed set of features selected with the whole training data set, because this induces a bias in the results. Instead, one should withhold a subset of instances, select features, and assess the performance of the classifier with the selected features using the left out examples. The results reported in this section were obtained with ten runs, each run is a ten-fold cross-validation, i.e., in one run, a feature subset was selected using the 90% of the instances, then, the accuracy of this subset was estimated over the unseen 10% of the data. This was performed 10 times, each time proposing a possible different feature subset. In this way, estimated accuracies, selected attribute numbers and time needed were the result of a mean over ten ten–cross–validation samples.

Table 1 shows the average number of attributes selected using correlation, consistency or the wrapper method. From the result, we can conclude that the wrapper method selects smaller sets of attributes on average but it is the more expensive computationally.

Table 2 shows the average percentage of correctly classified instances using WEKA's Experimenter tool. The

[2] http://mdp.ivv.nasa.gov/

Table 1: No. of attributes selected

				Wrapper	
	CFS	CNS	FCBF	C4.5	NB
CM1	5.24	1.30	1.09	1.02	1.09
JM1	8.01	19.99	1.00	3.29	1.84
KC1	7.77	17.61	1.00	2.97	1.92
KC2	5.52	13.27	1.12	1.78	2.26
PC1	4.63	10.67	1.32	1.67	1.14

statistical *two tailed paired t-test* [19] with $\alpha = 0.05$ was used to investigate whether the estimation with the subset was statistically significant when compared with the original set of attributes (orig). From the results, we can conclude that in general the feature subset improves the accuracy of the estimation, being the wrapper method superior to the filter method. The C4.5 and naïve Bayes classifiers using the wrapper method obtained 3 statistically significant improved models.

The wrapper is a good option as either improves the accuracy or when the accuracy is not improved is because the models generated are very simple (the number of selected attributes is very low).

In this case, with ten-cross validation, Table 3 shows the number of times that an attribute has been selected out of the 10 times that the algorithm was run using the CFS (correlation) with sequential search. We have selected this algorithm based on correlations because it obtained good result with all classifiers. It possible to analyse this table in 2 dimensions. First, analysing columns, i.e, attributes selected for each dataset, and second, analyzing row, the number of times that an attribute is selected across different datasets. For example, with JM1, 7 attributes were always selected (`loc`, `ev(g)`, `iv(g)`, `i`, `IOComment`, `IOBlank` and `IOCodeAndComment`), 4 times the algorithm selected `v(g)` and 6 times `branchCount`. On the row dimension, it can be concluded that the attribute `i` is is relevant and provides important information for classification, attribute `i` has been always selected in 4 datasets (CM1, JM1, KC1 and KC2) and 5 times in the PC1 dataset. In the KC2 dataset, the attribute *loc* has been selected 4 times, and the attribute *v(g)* was never selected.

4 Conclusions and Future Work

Feature Selection (FS) can be grouped into filter or wrapper depending on whether the data mining classifier algorithm is used to select attributes. In this paper, FS has been applied to 5 different datasets from the PROMISE Repository to defect faulty modules. The results show that in general, the smaller datasets maintain the prediction capability with a lower number of attributes than the original datsets. Also the wrapper mode is better than the filter mode but it is more computationally expensive.

We will extend this work to further datasets and different software engineering problems such as estimation. In addition, a possible problem with the analyzed datasets is that some of the datasets are unbalanced, i.e., many more modules were classified as non-defective than defective. Current research works on how to balance dataset can be applied before the techniques described here.

Acknowledgements

We would like to thank the Spanish Ministry of Science and Technology for supporting this research (Project CICYT TIN2004-06689-C03).

References

[1] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, 1997.

[2] Z. Chen, T. Menzies, D. Port, and B. Boehm. Finding the right data for software cost modeling. *IEEE Software*, 22:38–46, 2005.

[3] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analisys*, 1(3):131–56, 1997.

[4] M. Dash, H. Liu, and H. Motoda. Consistency based feature selection. In *Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, pages 98–109, 2000.

[5] J. Doak. An evaluation of feature selection methods and their application to computer security. Technical Report CSE-92-18, University of California, Department of Computer Science, Davis, CA, 1992.

[6] N. E. Fenton and S. L. Pfleeger. *Software metrics: a Rigorous & Practical Approach.* International Thompson Press, 1997.

Table 2: Percentage of correctly classified

	C4.5					NB				
Dataset	Orig.	CFS	CNS	WRP	FCBF	Orig.	CFS	CNS	WRP	FCBF
CM1	88.05	89.30	89.82	90.16○	89.84	84.84	87.15○	89.70○	90.02○	89.34○
JM1	79.73	80.83○	79.72	80.78○	80.89○	80.42	80.37	80.34	80.67	80.87○
KC1	84.04	84.54	84.22	84.80	84.83	82.46	82.73	82.30	85.38○	85.04○
KC2	81.19	83.64	82.18	84.44○	83.58	83.62	83.60	83.97	83.24	83.93
PC1	93.63	93.17	93.10	92.87	92.91	89.00	90.05	88.74	92.89○	91.96○

○, statistically significant improvement

Table 3: F-measure

	C4.5					NB				
Dataset	Orig.	CFS	CNS	WRP	FCBF	Orig.	CFS	CNS	WRP	FCBF
CM1	0.94	0.94	0.95	0.95○	0.95	0.91	0.93○	0.95○	0.95○	0.94○
JM1	0.88	0.89○	0.88	0.89○	0.89○	0.89	0.89	0.89	0.89○	0.89○
KC1	0.91	0.91	0.91	0.92	0.92	0.90	0.90	0.90	0.92○	0.92○
KC2	0.88	0.90	0.89	0.91○	0.90○	0.90	0.90	0.90	0.90	0.91
PC1	0.97	0.96	0.96	0.96	0.96	0.94	0.95	0.94	0.96○	0.96○

○, statistically significant improvement

[7] M. Hall. *Correlation-based Feature Selection for Machine Learning*. PhD thesis, University of Waikato, Department of Computer Science, Hamilton, New Zealand, 1999.

[8] M. H. Halstead. *Elements of software science*. Elsevier Computer Science Library. Operating And Programming Systems Series; 2. Elsevier, New York ; Oxford, 1977.

[9] C. Kirsopp and M. Shepperd. Case and feature subset selection in case-based software project effort prediction.

[10] R. Kohavi and G. John. Automatic parameter selection by minimizing estimated error. In *12th Int. Conf. on Machine Learning*, pages 304–312, San Francisco, 1995.

[11] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 1-2:273–324, 1997.

[12] P. Langley. Selection of relevant features in machine learning. In *Procs. Of the AAAI Fall Symposium on Relevance*, pages 140–144, 1994.

[13] H. Liu and H. Motoda. *Feature Selection for Knowlegde Discovery and Data Mining*. Kluwer Academic Publishers, London, UK, 1998.

[14] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. on Knowledge and Data Eng.*, 17(3):1–12, 2005.

[15] T. J. McCabe. A complexity measure. *IEEE Transactions on Software Engineering*, 2(4):308–320, December 1976.

[16] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[17] J. R. Quinlan. *C4.5: Programs for machine learning*. Morgan Kaufmann, San Mateo, California, 1993.

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

[19] I. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2 edition, 2005.

[20] L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–24, 2004.

[21] L. Yu and H. Liu. Redundancy based feature selection for microarry data. In *10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2004.

Table 4: No. of Times an Attribute is Selected

	CFS-sf				
	CM1	JM1	KC1	KC2	PC1
loc: McCabe's line count of code	6	10	1	4	1
v(g): McCabe "cyclomatic complexity"	0	4	4	0	2
ev(g): McCabe "essential complexity"	0	10	2	10	0
iv(g): McCabe "design complexity"	7	10	1	0	0
n: Halstead total operators + operands	0	0	1	2	0
v: Halstead "volume"	0	0	0	0	0
l: Halstead "program length"	0	0	0	0	0
d: Halstead "difficulty"	1	0	8	2	0
i: Halstead "intelligence"	10	10	10	10	5
e: Halstead "effort"	0	0	4	0	0
b: Halstead	1	0	0	0	0
t: Halstead's time estimator	0	0	2	0	0
IOCode: Halstead's line count	1	0	6	1	1
IOComment: Halstead's lines of comments	10	10	9	2	9
IOBlank: Halstead's blank lines	4	10	10	2	10
IOCodeAndComment	0	10	0	4	10
uniq-Op: unique operators	6	0	1	8	0
uniq-Opnd: unique operands	4	0	7	10	4
total-Op: total operators	0	0	0	0	1
total-Opnd: total operands	0	0	4	0	0
branchCount: of the flow graph	0	6	8	0	0