Attribute Selection in Software Engineering Datasets for Detecting Fault Modules

Daniel Rodriguez[1] R Ruiz[2] J Aguilar[2] J Cuadrado[1]

[1]University of Alcalá, Madrid, Spain
[2]Pablo de Olavide Univeristy, Seville, Spain

Euromicro 2007 - Software Management Session

Outline

Outline

Motivation

In this work, we apply Feature Subset Selection (FSS) to several datasets publicly available (PROMISE repository), and different classifiers to improve the detection of faulty modules. FSS is important in different ways:

- 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.
- we can avoid the collection of data for those irrelevant and redundant attributes in the future.

Feature Subset Selection (FSS)

FSS algorithms search through candidate feature subsets guide by a certain evaluation measure which captures the goodness of each subset. An optimal (or near optimal) subset is selected when the search stops. There are two possibilities when applying FSS:

- 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.

Outline

Classifiers used with FSS in this Work

- The IB1 is a nearest-neighbor (K-NN) classifier. To classify a new test sample, all training instances are stored and the nearest training instance to the test instance is to be found (this class is retrieved to predict the class of the test instance).
- C4.5. A decision tree where the leaves of the tree correspond to classes, nodes correspond to features, and branches to their associated values.

Evaluation

In this work, we have used the following evaluation techniques:

- Cross-validation (CV) Partition the data into k sets of samples, $C_1, \ldots, C_k$ (typically, 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.

F-measure

Another common way to measure the goodness of data mining applications is through the $f - measure$, which is defined as:

$$f - measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

where
$Precision = tp/(tp + fp)$ and
$Recall = tp/(tp + fn)$

Outline

Datasets

We have applied feature selection to the CM1, JM1, KC1, KC2, and PC1 datasets available in the PROMISE repository.

- Publicly available at: `http://promisedata.org/repository/`
- All datasets contain 22 attributes composed of 5 different lines of code measure, 3 McCabe metrics, 4 base Halstead measures, 8 derived Halstead measures, a branch-count, and the last attribute is 'problems' with 2 classes (false or true, whether the module has reported defects)

Outline

Running of the Experiments

All algorithms were run using the WEKA environment, either using the Explorer or the Experimenter tool.

- Results were obtained with 10 runs, each run is a 10-fold CV, 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. Therefore, estimated accuracies and the no. of selected attributes were the result a mean over 10 10–CV.

Outline

No. of Attributes Selected

Table: No. of attributes selected

			Wrapper	
	CFS	CNS	C4.5	IB1
CM1	5.24	1.30	1.02	1.83
JM1	8.01	19.99	3.29	2.91
KC1	7.77	17.61	2.97	1.94
KC2	5.52	13.27	1.78	2.16
PC1	4.63	10.67	1.67	1.58

Percentage of correctly classified

Table: Percentage of correctly classified

	C4.5				IB1			
Dataset	Orig.	CFS	CNS	WRP	Orig.	CFS	CNS	WRP
CM1	88.05	89.30	89.82	90.16◦	84.98	83.59	83.86	86.44
JM1	79.73	80.83◦	79.72	80.78◦	75.91	75.54	75.91	72.01
KC1	84.04	84.54	84.22	84.80	83.35	82.77	83.59	73.80
KC2	81.19	83.64	82.18	84.44◦	78.62	77.64	78.10	76.45
PC1	93.63	93.17	93.10	92.87	91.87	91.42	91.18	85.96

◦, statistically significant improvement

F-measure

Table: F-measure

	C4.5				IB1			
Dataset	Orig.	CFS	CNS	WRP	Orig.	CFS	CNS	WRP
CM1	0.94	0.94	0.95	0.95◦	0.92	0.91	0.88	0.92
JM1	0.88	0.89◦	0.88	0.89◦	0.85	0.85	0.85	0.79
KC1	0.91	0.91	0.91	0.92	0.90	0.90	0.90	0.78
KC2	0.88	0.90	0.89	0.91◦	0.86	0.86	0.86	0.84
PC1	0.97	0.96	0.96	0.96	0.96	0.95	0.95	0.89

◦, statistically significant improvement

No. of Times an Attribute is Selected

Table: 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
lOCode: Halstead's line count	1	0	6	1	1
lOComment: Halstead's lines of comments	10	10	9	2	9
lOBlank: Halstead's blank lines	4	10	10	2	10
lOCodeAndComment	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

Conclusions and Future Work

- FS obtains better accuracy than the datasets using all attributes
  - The wrapper model is superior to the filter model
    - It either improves the accuracy, or when the accuracy is not improved, the models generated are much simpler, i.e., low no. of attributes; e.g., for the JM1 dataset, the accuracy using IB1 wrapper is 72.01% compared with 79.73% when using all attributes, however, it just needs less than 3 attributes for an *acceptable* accuracy.
  - The drawback is that the wrapper model is computationally expensive (some runs took around a week to finish).
- From the SE point of view, FS finds important attributes (for these datasets, FS has removed most derived attributes.
- Future work: other datasets with more classes, how to deal and analyze unbalanced datasets, etc.