Decision Support for Global Software Development with Pattern Discovery

Jack H.C. Wu
College of Professional and Continuing Education
The Hong Kong Polytechnic University
Hung Hom, Hong Kong
hcjwu@speed-polyu.edu.hk

Jacky Keung
Department of Computer Science
City University of Hong Kong
Kowloon Tong, Hong Kong
Jacky.Keung@cityu.edu.hk

Abstract—**Background: Software development process nowadays is becoming more globalized than ever before. Global Software Development (GSD) implies that the software development process is spread across countries and geographic boundaries. GSD brings challenges to software project leaders / managers because of the increase in management difficulty. As a result, utilizing data mining and machine learning techniques to provide quantitative, objective and predictive solution for project management is essential. Aim: To facilitate software project management to make decisions by mining embedded knowledge from data and providing meaningful results. Method: In this paper we propose to adopt a pattern discovery technique which has been successfully applied in the field of computational Biology. The technique discovers association patterns inherited in the data which can provide insightful information for domain experts (e.g., project leaders), therefore increasing their confidence in making decisions. We apply the technique in the software defect datasets from the NASA MDP repository to predict whether a software project is defective or not and find out important factors in the data that signaled the prediction. Results: For the tested datasets, statistically significant patterns are produced with good classification performance. The experiment results also reveal the effect of different discretization techniques on performance. Conclusions: To the best of our knowledge, this is the first study to employ the specific pattern mining technique in Software Engineering for defective software detection and the results showed the potential of such a technique in which it can provide not only good classification results but also meaningful information for project leaders to make decisions.**

Keywords-pattern discovery; association pattern; discretization; attribute clustering

I. INTRODUCTION

Software development process nowadays is becoming more globalized than ever before, especially with the help of cloud-based technology. Global software development (GSD) implies that software development process to be spread across countries and geographic boundaries. While GSD can contribute to software businesses in many ways such as reducing development cost by employing development teams in countries with lower labor cost and increasing the development efficiency by round-the-clock development [1], it also brings challenges to software project leaders / managers because of the multi-site, multi-cultural distributed environment which increases management difficulty [2, 3].

978-1-4673-9904-3/16/$31.00 ©2016 IEEE

As the complexity of the development environment grows, project leaders can no longer micro-manage every details in the management process. Noticing early signals of project failure is harder than ever before. Therefore, utilizing data mining and machine learning techniques to provide quantitative, objective and predictive solution for project management is essential [4], for examples, in effort estimation [5] and defect prediction [6].

However, most of the previously employed techniques act as a black-box which represents results that are difficult for domain experts to examine. Two main problems are: (i) they require parameters calibration for accurate prediction and (ii) they focus on estimation accuracy quantitatively while qualitative information like "What are the most influential elements in the data that lead to the particular quantitative prediction?" are difficult to obtain by the practitioners (i.e., project leaders).

In this paper we propose to adopt a pattern discovery technique which has been successfully applied in the field of computational Biology [7]. The technique was proposed to classify cancerous tissues by mining association patterns in the microarray data. We adopt the technique in this study for defective software projects detection in the datasets from the NASA MDP repository [8]. Association rule mining is the discovery of association relationships among different items in a dataset [9]. Usually it requires the user to provide *support* and *confidence* thresholds which are difficult to determine by users. Setting the thresholds incorrectly may not reveal the inherited pattern which can decrease prediction accuracy [10]. The adopted technique in this paper is highly automatic which requires minimal user intervention in terms of parameter calibration and it outputs statistically significant association patterns which are meaningful for the user to examine.

The rest of the paper is organized as follows. Section II describes the pattern discovery process which consists of three stages: (i) discretization of attributes with continuous values, (ii) attribute clustering for feature selection and (iii) statistically significant association rule mining. Classification can be done after the association rules are obtained. Section III reports the results of the technique in the datasets from the NSAS MDP repository for software defect classification. Finally, Section IV concludes the paper.

II. THE PATTERN DISCOVERY PROCESS

For data preprocessing, discretization of continuous attributes is necessary for better performance of machine learning algorithms [11]. It involves discretizing the attribute values into a finite number of intervals. For classification problems, the objectives of discretization are: (i) to maximize the interdependence between class labels and attribute values and (ii) to minimize the number of intervals without significant loss of class-attribute mutual dependence. We tested four discretization methods to examine their effects on the classification accuracy.

For high-dimensional datasets, attributes with higher discriminating power should be selected for classification. This process is sometimes referred as Feature Subset Selection. Feature subset selection can reduce the computational cost in finding association rules and also increase classification performance [12]. We use an information-theoretic attribute clustering algorithm which maximizes the dependence of attributes within a cluster and minimizes the dependence of attributes cross clusters [13]. Representative attributes are selected from each cluster for association rule mining. The attribute clustering algorithm has been shown to be effective in previous study [7].

After discretization and attributes selection, a pattern discovery algorithm based on adjusted residual analysis [14] is employed to discover the statistically significant associations among attribute values.

A. *Discretization Methods*

Discretization methods can be classified into supervised and unsupervised methods. Supervised methods take class information into account to find proper intervals while unsupervised methods do not. Previous research indicates that supervised methods are better than unsupervised ones [15]. In this study we tested four discretization methods in which two of them (namely equal-width and equal-frequency) are unsupervised and the other two (namely the Holte's 1R algorithm [16] and the CAIM discretization algorithm [17]) are supervised.

1) *Equal-width and Equal-frequency Methods*

Equal-width and equal-frequency are the two simplest methods for discretization. In equal-width discretization, the range which is the difference between the minimum and maximum values of an attribute is determined and then divided by the number of intervals k which its value is user-defined. It divides the attribute values into k equal width intervals. Equal-width method is not suitable for attribute values that are not distributed evenly because a lot of information will be lost after discretization.

Equal-frequency discretization sorts the attribute values in ascending order and divides the range into k (user-defined) intervals such that every interval contains the same number of values. This method could cause the same attribute value to be assigned to different intervals. In the experiments we avoid the situation by adjusting the boundaries of neighboring intervals such that each distinct value falls into exactly one interval.

2) *Holte's 1R Algorithm*

Holte [16] proposed a simple classifier that induces one-level decision trees which involves a supervised discretization process (referred as One-Rule Discretizer [15]). It sorts the attribute values in ascending order and greedily divides the values into intervals such that each interval contains only one particular class. A minimum interval size of 6 is suggested from empirical analysis.

3) *CAIM Discretization Algorithm*

For an attribute A, with class variable C and discretization variable D, the Class-Attribute Interdependence Maximization (CAIM) algorithm [17] starts with a single interval and divides it iteratively such that the CAIM value in (1) is maximized in each of the iterations.

$$CAIM(C, D \mid A) = \frac{\sum_{r=1}^{n} \frac{\max_r^2}{M_{+r}}}{n} \tag{1}$$

where n is the number of intervals, M_{+r} is the number of values in A fall within the rth interval and $\max_r$ is the maximum number of values in the rth interval that belongs to a particular class.

A larger CAIM value implies a higher interdependence between class labels and intervals. The algorithm favors discretization schemes that each interval contains a majority of values within a single class label and it does not require user-defined parameters. The CAIM algorithm is efficient and effective when comparing with other discretization methods [18].

B. *Attribute Selection Method*

In order to find attributes with high discriminating power, attribute clustering is conducted so that attributes within a cluster have high interdependence while attributes across different clusters are more independent [7]. After attribute clustering representative attributes for each cluster is selected for pattern discovery. We adopt the attribute clustering algorithm by Au et al. [13] which utilizes the interdependence redundancy measure $R(A_i{:}A_j)$ in (2) for measuring the interdependence for two attributes A_i and A_j.

$$R(A_i : A_j) = \frac{I(A_i : A_j)}{H(A_i : A_j)} \tag{2}$$

where $I(A_i{:}A_j)$ is the *mutual information* between A_i and A_j and $H(A_i{:}A_j)$ is the *joint entropy* between A_i and A_j.

Attributes with higher dependence will have a higher value of $R(A_i{:}A_j)$. The information-theoretic algorithm is similar to k-means clustering except that (i) the centroid of each cluster is the one with highest multiple interdependence redundancy measure and (ii) the distance measure is replaced by the interdependence redundancy measure $R(A_i{:}A_j)$. The value of k is determined by optimizing the intra-group attribute interdependence which means that it does not require user-defined input. After clustering, the centroid of each cluster is selected to be the representative attribute of the whole cluster.

C. Pattern Discovery

We employ the pattern discovery algorithm for defective software project detection based on the *adjusted residual* analysis [7, 14, 19]. Intuitively, two events are dependent if their observed frequency of co-occurrences deviates significantly from the default (independence) model. The deviation is called *residual.* In order to calculate the *residual*, the *observed* and *expected* number of occurrences of a compound event has to be calculated.

For example, suppose a dataset contains T tuples with the set of attributes A and each tuple t belongs to a particular class in the set C. In second-order pattern discovery where association rules containing two attributes A_i and A_j are mined, for the tuples $t \in T$, the *observed* occurrence of the compound event where $t[A_i] = k$ where $k \in [1,\ldots, n_i]$ (n_i is the number of intervals of A_i), $t[A_i] = l$ where $l \in [1,\ldots, n_j]$ (n_j is the number of intervals of A_j) and the class of t is a particular class $c \in C$ is:

$$\left|\{t \in T \mid t[A_i] = k \cap t[A_j] = l \cap t[C] = c\}\right|$$

where $|S|$ is the cardinality of the set S. And the *expected* occurrence of the compound event which assumes independence:

$$\left|\{t \in T \mid t[C] = c\}\right| \times \frac{\left|\{t \in T \mid t[A_i] = k\}\right|}{|T|} \times \frac{\left|\{t \in T \mid t[A_j] = l\}\right|}{|T|}$$

The *standardized residue* of the particular pattern (i.e., $t[A_i]=k$, $t[A_j]=l$ and $t[C]=c$) having the *observed* and *expected* occurrences is:

$$residual = \frac{observed - \exp ected}{\sqrt{\exp ected}} \quad (3)$$

Finally, the *adjusted residual* (r) is found using the *standardized residue* divided by the square-root of the maximum likelihood estimate of its variance [7].

The *adjusted residual* (r) is an approximate standard normal distribution [20]. If $|r| > 1.96$, it means that the particular pattern is statistically significant with 95% C.I. If $|r| > 2.58$, it means that the particular pattern is statistically significant with 99% C.I. Using the value of *adjusted residual* (r), without user-defined inputs, statistically significant patterns can be discovered and a classification model can be built using the discovered patterns [14].

III. EXPERIMENT

A. Setting

We use the software defect datasets from the NASA MDP repository [8] to test the pattern discovery process discussed in the previous section. Table 1 shows the statistics of the tested datasets. All the examples in the datasets contain only continuous attributes and are classified as either defective or not defective. There are no missing values in all of the tested datasets.

For each dataset, 90% of the examples are randomly selected as the training set for pattern discovery and classification model building. The remaining 10% of the examples act as the test set in which the classification accuracy is calculated. The process is repeated 10 times for each dataset and the accuracies are averaged.

TABLE 1. STATISTICS OF THE TESTED DATASETS

Dataset	# features	# examples	# defective
CM1	37	344	42 (12.2%)
KC1	21	2,096	325 (15.5%)
MW1	37	264	27 (10.2%)
PC1	37	759	61 (8%)
PC5	38	17,001	503 (3%)

B. Effect of Discretization Methods

Table 2 shows the results of using the four discretization methods discussed in Section II-A. For equal-width and equal-frequency (equal-freq) methods, an input parameter k is required to determine the number of intervals. For Holte's 1RD and CAIM algorithms, the number of intervals is determined automatically.

TABLE 2. CLASSIFICATION ACCURACIES USING DIFFERENT DISCRETIZATION METHODS

Method	*k*	CM1	KC1	MW1	PC1	PC5
equal-width	2	0.80	0.79	0.77	0.70	**0.79**
	3	0.82	0.79	0.83	0.68	0.77
	4	0.80	0.80	0.83	0.70	0.78
	5	**0.83**	0.79	0.77	0.68	0.77
equal-freq	2	0.68	0.59	0.70	0.67	0.73
	3	0.80	0.63	0.74	0.69	0.74
	4	0.77	0.56	0.81	0.69	0.76
	5	0.82	0.63	0.85	0.73	0.75
1RD	auto	0.77	0.73	**0.86**	**0.80**	0.74
CAIM	auto	**0.83**	**0.82**	**0.86**	0.74	**0.79**

From the results, the CAIM generally performs better than others in classification accuracy. This is consistent with previous research results [18]. We use the CAIM discretization to generate the results of the following experiments.

C. Number of Selected Attributes

The number of selected attributes is directly related to the number of clusters (k) formed in the attribute clustering algorithm discussed in Section II-B. The algorithm automatically selects the best number of clusters by calculating the multiple interdependence redundancy measures (2) for k clusters and selecting the k which produced the maximized value.

Fig. 1 shows the values of the multiple interdependence redundancy measure for different values of k in the five tested datasets. From Fig. 1, the best values of k are less than 11 for all tested datasets. Given that the least number of features for the datasets is 21 (KC1), the attribute clustering algorithm can eliminate at least half of the attributes from the datasets.

D. Significant Patterns Discovered

Table 3 shows some of the first-order statistically significant patterns mined for each tested datasets. "Y" means defective while "N" means not defective. From the rules, one can observe that features such as "HALSTEAD_EFFORT" and "HALSTEAD_ERROR_EST" are important signals to

investigate to see whether an example is defective. Besides quantitative prediction values, the patterns can provide qualitative information for the project leaders to make educated decisions. Furthermore, when using 1RD or CAIM algorithms for discretization, input parameters are not necessary from the user. This can enhance the utilization of data mining techniques by practitioners.

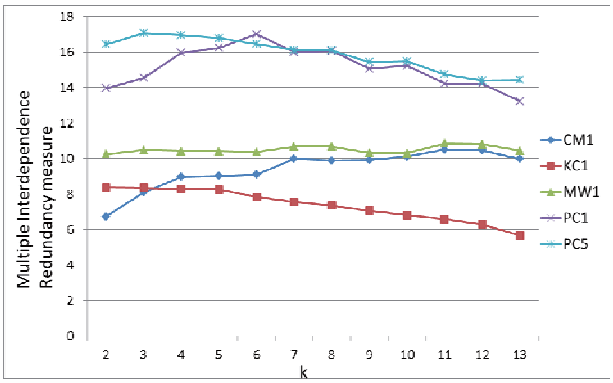

Fig. 1. Multiple interdependence redundancy measures for different values of *k* in the tested datasets.

TABLE 3. EXAMPLES OF STATISTICALLY SIGNIFICANT PATTERNS MINED FOR EACH DATASET

Dataset	Rule	*r*
CM1	NUM_OPERATORS ≥ 316 (Y)	4.0
	HALSTEAD_EFFORT ≥ 33,930.4 (Y)	3.8
KC1	BRANCH_COUNT ≥ 7 (Y)	5.3
	HALSTEAD_ERROR_EST ≥ 0.37 (Y)	4.8
MW1	MODIFIED_CONDITION_COUNT ≥ 9 (Y)	5.2
	EDGE_COUNT ≤ 66 (N)	3.9
PC1	HALSTEAD_EFFORT ≥ 86,270 (Y)	5.0
PC5	HALSTEAD_ERROR_EST ≥ 0.13 (Y)	6.5
	CYCLOMATIC_COMPLEXITY ≥ 15 (Y)	6.2

IV. CONCLUSIONS

To the best of our knowledge, this is the first study to employ the adjusted residual pattern mining technique in Software Engineering for defective software project detection and the results show that the potential of such a technique by providing not only good classification results but also meaningful information for project leaders to make decisions. The technique is highly automated which requires minimal user intervention in terms of parameters calibration. We will continue further investigation on the technique and compare it with other data mining techniques in future work.

ACKNOWLEDGEMENTS

This research is supported by the City University of Hong Kong research funds (Project No. 7200354, 7004222, 7004474).

REFERENCES

[1] J. D. Herbsleb and D. Moitra, "Global software development," *Software, IEEE,* vol. 18, pp. 16-20, 2001.

[2] A. Avritzer, D. Paulish, Y. Cai, and K. Sethi, "Coordination implications of software architecture in a global software development project," *Journal of Systems and Software,* vol. 83, pp. 1881-1895, 10// 2010.

[3] A. Begel and N. Nagappan, "Global Software Development: Who Does It?," in *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, 2008, pp. 195-199.

[4] L. C. Briand, V. R. Basili, and W. M. Thomas, "A pattern recognition approach for software engineering data analysis," *Software Engineering, IEEE Transactions on,* vol. 18, pp. 931-942, 1992.

[5] K. Srinivasan and D. Fisher, "Machine learning approaches to estimating software development effort," *Software Engineering, IEEE Transactions on,* vol. 21, pp. 126-137, 1995.

[6] T. Menzies, J. Greenwald, and A. Frank, "Data Mining Static Code Attributes to Learn Defect Predictors," *Software Engineering, IEEE Transactions on,* vol. 33, pp. 2-13, 2007.

[7] A. K. C. Wong, W. H. Au, and K. C. C. Chan, "Discovering High-Order Patterns of Gene Expression Levels," *Journal of Computational Biology,* vol. 15 (6), pp. 625-637, 2008.

[8] "NASA MDP Software Defect Data Sets, http://http://nasa-softwaredefectdatasets.wikispaces.com/."

[9] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *SIGMOD Rec.,* vol. 22, pp. 207-216, 1993.

[10] J. Han, J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*: Morgan Kaufmann, SanFrancisco, CA, 2011.

[11] T. Elomaa and J. Rousu, "Efficient Multisplitting Revisited: Optima-Preserving Elimination of Partition Candidates," *Data Min. Knowl. Discov.,* vol. 8, pp. 97-126, 2004.

[12] T. Menzies, O. Mizuno, Y. Takagi, and T. Kikuno, "Explanation vs Performance in Data Mining: A Case Study with Predicting Runaway Projects," *JSEA,* vol. 2, pp. 221-236, 2009.

[13] W.-H. Au, K. C. C. Chan, A. K. C. Wong, and Y. Wang, "Attribute Clustering for Grouping, Selection, and Classification of Gene Expression Data," *IEEE/ACM Trans. Comput. Biol. Bioinformatics,* vol. 2, pp. 83-101, 2005.

[14] A. K. C. Wong and W. Yang, "Pattern discovery: a data driven approach to decision support," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on,* vol. 33, pp. 114-124, 2003.

[15] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," presented at the ICML-95, 1995.

[16] R. C. Holte, "Very Simple Classification Rules Perform Well on Most Commonly Used Datasets," *Mach. Learn.,* vol. 11, pp. 63-90, 1993.

[17] L. A. Kurgan and K. J. Cios, "CAIM Discretization Algorithm," *IEEE Trans. on Knowl. and Data Eng.,* vol. 16, pp. 145-153, 2004.

[18] K. J. Cios, W. Pedrycz, R. W. Swiniarski, and L. Kurgan, *Data Mining A Knowledge Discovery Approach*: Springer Science+Business Media, LLC, NY, USA, 2007.

[19] A. K. C. Wong and W. Yang, "High-order pattern discovery from discrete-valued data," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 9, pp. 877-893, 1997.

[20] S. Haberman, "The Analysis of Residuals in Cross-Classified Tables," *Biometrics,* vol. 29, pp. 205-220, // 1973.