International Journal of Software Engineering and Knowledge Engineering
Vol , No.
© World Scientific Publishing Company

A DATA MINING APPLICATION: ANALYSIS OF PROBLEMS OCCURRING DURING A SOFTWARE PROJECT DEVELOPMENT PROCESS

RICHI NAYAK
School of Information Systems
Queensland University of Technology
Brisbane, QLD, Australia

TIAN QIU
EDS Credit Services
Adelaide, SA, Australia

Abstract

Data mining techniques provide people with new power to research and manipulate the existing large volume of data. A data mining process discovers interesting information from the hidden data that can either be used for future prediction and/or intelligently summarising the details of the data. There are many achievements of applying data mining techniques in various areas such as marketing, medical, and financial, although few of them can be currently seen in software engineering domain. In this paper, a proposed data mining application in software engineering domain is explained and experimented. The empirical results demonstrate the capability of data mining techniques in software engineering domain and the potential benefits in applying data mining in this area.

Keywords: Data Mining; Software Engineering, knowledge discovery

1. Introduction

Nowadays software projects keep growing in both scale and complexity. Improvement of the software product quality is a challenge for every software project leader. A project leader has to manage a project with several issues involved such as negotiation with customers, project planning and scheduling, code implementation, test and release. It is difficult for a project leader to precisely estimate the project duration before it starts. He can not possibly analyse all possible causes accurately if a problem is beyond his own background knowledge or previous experience.

Data Mining (DM) techniques can assist software engineers to conduct the estimation and cause analysis of a project. A project leader can make the accurate estimation of a new project by learning from the information gained by applying data mining methods to previous projects. A leader can eliminate potential problems when a similar pattern appears in the current project to the one that caused problems in previous projects.

Various DM techniques have been developed based on the works of Statistic, Artificial Intelligence, Machine Learning and Database system for knowledge discovery. DM is the process of facilitating decision-making by identifying valid, novel, potentially useful, and ultimately understandable structures in data [7]. Usually this kind of extracted knowledge is classification rules, characteristic rules, association rules, functional relationships, functional dependencies, causal rules, temporal knowledge or clusters according to the chosen DM technique and operation.

DM techniques can analyse many kinds of data such as relational, object-oriented, text, temporal, spatial, combinatorial, web and multimedia. Data mining tools and applications have generated positive results, and are continuously stimulated by exploring new areas due to the benefits brought by this technology. There are many achievements of applying data mining techniques to various areas such as marketing, medical, and financial, although few of them can be currently seen in software engineering domain. There exist several difficulties in this domain, such as hard to find a data model to put through mining process and/or no suitable mining tools. This brings out the need to investigate the efficacy of data mining techniques applied in the software engineering domain.

This paper explores the software engineering domain by applying DM techniques to a set of data collected from a software project process within a software development company. This paper starts by introducing a real software engineering problem, and data mining. After a data model is established for the underlying problem, data mining techniques and softwares such as CBA [1], C5.0 [2] and TextAnalyst [3] are introduced and experimented. Interesting findings from different data mining operations are discussed together with issues appearing during the mining process. Finally some suggestions are given about the future applications within software engineering domain.

2. Introduction to Data Mining

Data mining is the search for relationships and distinct patterns that exist in datasets but are ‘hidden’ among the vast amount of data. A data mining task involves determining useful patterns from collected data or determining a model that fits best on the collected data. Predictive modelling, clustering and link analysis are the three major data mining operations or tasks [5, 8]. Predictive modelling solves a problem by looking at the past experiences with known answers, and then projecting to new cases based on essential characteristics about the data. Clustering or segmentation solves a problem by identifying objects with similar characteristics in a data set, and thus forming the groups of similar objects. The link analysis operation establishes association between objects by finding items that imply the presence of other items in a given data set.

A data mining task includes pre-processing, the actual data mining process and post-processing. During the pre-processing stage, the data mining problem and all sources of data are identified, and a subset of data is generated from the accumulated data. To ensure quality the data set is processed to remove noise, handle missing information and transform to an appropriate format. A data mining technique or a combination of techniques appropriate for the type of knowledge to be discovered is applied to the derived data set. The last stage is post-processing in which the discovered knowledge is evaluated and interpreted. Operations such as filtering, structuring and data visualisation are utilised to represent the extracted knowledge to users in a meaningful and easily understandable manner.

Sometimes, the data mining process maintains results by iterating all the steps again for user satisfaction, or/and adapting the new information in the future (indicated by the dotted lines in Figure 1).

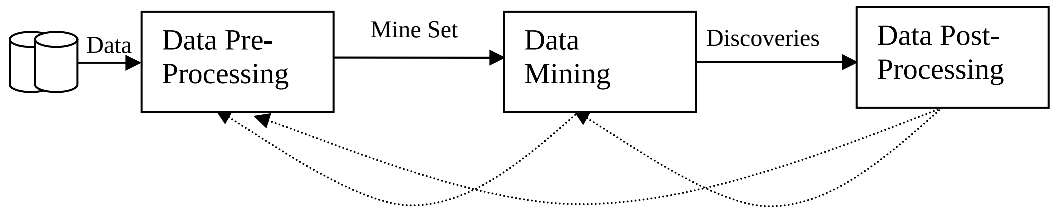


Figure 1: A Usual Data Mining Process

Attribute	Description
Category	Name of a project or department in MASC that raises the PR.
Synopsis	The main points of the issue associated with the PR.
Responsible	The person(s) who is (are) assigned to fix the problem.
Originator	The name of the person who submit the PR.
Confidential	Is the report confidential?
Submitter-ID	Identification of the originator.
Environment	The problem is related to which environment such as windows or Unix.
Release Note	Fixing comment.
Audit Trail	The progress of the process.
Arrival-Date	Date, Time & Day of submitting the PR to the MASC Intranet system.
Closed-Date	Date, Time & Day of closing the PR from the MASC Intranet system.
Description	Purely text format data about what the bug is and how it affects the whole system.
Priority	*High*: the problem should be resolved immediately. *Medium*: the problem should be resolved within 1 week. *Low*: the problem should be resolved if the project schedule permits.
State	*Open:* the PR is not analysed yet. *Active:* A valid PR and assigned to someone. *Analysed:* the PR is under investigation. *Suspended:* the PR is waiting due to certain reason. *Closed:* there is no work left for this PR. *Resolved:* the PR is successfully fixed. *Feedback:* the current Responsible person for the PR is waiting for feedbacks from a third party. According to the feedback, a PR can be further classified to *resolved, active* or *suspend* state.
Class	*Sw-bug:* the bug is from the software code implementation. *Doc-bug:* the bug is from the documents that are directly related to a software product. C*hange-request*: an enhancement request from customer. S*upport:* the bug is from tools or documents that are not directly related to a software product. *Mistaken:* incorrect in either the software or document. *Duplicate:* the problem has already been covered by another PR record.

Table 1: Description of attributes for the software engineering problem

3. Data Mining Application to a Software Engineering Problem

Every year, more than 50 software projects are carried out in MASC[*], more than ten thousands lines of code are created, and thousands pages of documents are released to various customers. In order to control any problem appearing during a project life cycle and improve the working efficiency, MASC has set up a series of processes such as Software Configuration Management, Software Risk Management, Software Project Metric Report and Software Problem Report Management. Data is collected during all these procedures. The Software Problem Report (PR) management data is chosen as the focus for this research.

A software bug-tracking system, GNATS (**A** **T**racking **S**ystem by **GN**U), is set up on MASC Intranet to collect and maintain all PRs raised from every department and individual within MASC. Currently the GNATS system stores more than 40,000 of problem reports. The GNATS system can also provide simple PR retrieval based on some categories. On the top of each PR page, a number associated with the PR is displayed together with a series information about when, who and from which project or department this PR is raised. After this, there are several fields that give details of the PR (Table 1).

3.1 *Data Mining Process: Defining Goals*

Two areas of interest that may be suitable for mining tasks are identified. The first one appears at the early Estimation and Planning stage of a software project. To ensure software products quality, MASC engineers need to make estimations on many aspects of a project. This includes (1) the number of lines of code to be developed, (2) the kinds of document to be delivered to customer, and (3) the time required to accomplish each software engineering stage for the project. Currently, although there are several tools that can help a programmer to do the implementation jobs during the software design stage, but there is few or even no tool that can be used on both estimation and project problem reasoning stage. Project team members can only give estimations based on their own experience from previous projects. If the current project is not within their familiar topics, the accuracy of the estimation may be worse. A PR (problem

[*] MASC is a division of a global telecommunication company. Due to security reason, detail of the information source is removed.

report) fixing work may become a time consuming task when a problem appears but the responsible person can not estimate the time to fix the problem. This directly affects the success of a software project. Finding a precise estimation figures on bug fixing or estimation work at the early stage of a project will bring great cost savings and accurate progress control to the whole development team as well as to the whole organization.

The second problem relates to the GNATS itself. Currently the system has limited ability of information retrieval. There is no actual database management system being implemented. If a PR is closed, it is just statically stored in GNATS and no further analysis is performed. This limits the potential benefits to software engineers who can obtain valuable information if the existing PR data is being analysed. This will be useful especially when a programmer is struggling with a bug while a resolution may already hide behind the knowledge that can be derived from the previous similar problems.

The above two problems can be relieved by utilising data mining techniques. The format of a PR data is semi-structured, and every field of it contains a simple type of data, such as short text. These features make the PR data set an ideal candidate for mining task. Mining results of the PR data will bring benefits such as accurate project estimation and planing, improved control over the PR fixing and deduction of the project cycle time.

3.2 *Data Mining Process: Data Pre-Processing*

The quality of data cannot be guaranteed even with large corporate systems. Although data mining tools have some form of noise control, the data is still need to be prepared for the mining process.

Data Field Selection There are several fields such as *Confidential, Submitter-ID, Environment, Fix, Release Note, Audit Trail, the associated project name and PR number* that are ignored during mining. These fields provide identification information about a PR, but they contain no data mining value. Instead, some of these values are used as support roles during pre-processing and post-processing stages to assist in the selection of data and a better understanding of the rules being found.

The aim of this data mining exercise is to find out useful knowledge from existing projects, all the existing projects should already be finished and all the corresponding PRs should also be closed. Otherwise, a PR can still be changed and is not stable for mining. Hence all PRs with a ***closed*** value in their *State* field are chosen.

Whenever a PR is raised, a project leader will have to find answers for the following questions before taking any action: How long it will take to fix? How many people were involved? How sever the problem is (customer impact)? What is the impact of the problem on project schedule (Cost & Team priority)? and What type of the problem it is (a Software bug or a design flaw)? According to these, attributes such as Severity, Priority, Class, Arrival-Date, Closed-Date, Responsible, Synopsis (table 1) are found important to describe the characteristics of a PR. The first three attributes describes how a PR is handled within a project, the next three attributes indicate how long a PR is fixed and who were responsible, and the last one lists the content in a PR. The attribute 'class' is chosen as the target attribute in order to find out any valuable knowledge among the type of a problem and the rest of the PR attributes. For example, when a project leader comes to know of the relationship between the fix effort (time) and the PR class, he can analyse the fix effort versus the human resources available and put it in the schedule and resource plan.

The first five fields have fixed input values. *Responsible* attribute is used to calculate how many people were involved to fix the problem. Association or characteristics rules can be found by applying mining techniques on these fields. Whereas *Synopsis* field has no fixed input values. Instead, there is a lot of pure text information stored in this field briefly describing what the problem is in the associated project. It may contain what type of a project document (a piece of code or a support document) that the PR is concerned with. It can be used as a text index. Due to the nature of the values inside this field, a different mining approach, Text Mining is considered to deal with such kind of text values.

Data Cleaning Data is further investigated to identify problems, such as missing values, inconsistent values, and mistaken values using graphical tools such as histogram for frequency distribution of the values or calculating maxima, minima and mean values. Histogram plots the contribution made by each value for the (categorical) attribute, and therefore helps to identify distribution skews and invalid values. The occurrence of these problems comes with several factors such as human mistakes and evolutions of the GNATS system. An example is the use of different terminologies over the time such as ***SW-bug*** or ***sw-bug*** as an input value for *Class* field (Example **a, d** in Figure 2). A *Time-Zone* field and other new input values have been added later in the system on management request based on feedback of users after several years of system running.

Depending on whether an error in a PR can be corrected, all PRs can be grouped in two categories: recoverable and unrecoverable. In recoverable group, an error can be recovered manually or automatically by software. So the modified PRs can be included in the mining process. For example, ***SW-bug*** in *Class* field is replaced by ***sw-bug*** throughout the data. Another example is filling the data in wrong fields such as mistakenly input the date being closed in an obsolete *Completed-Date* field instead of the *Closed-Date* field.

In unrecoverable group, the error cannot be recovered precisely and the PRs with error(s) have to be discarded. An example (Example **a** in Figure 2) shows that it has its closed time even earlier than the time being raised. Some PRs even lost the part of time related values, such as Example **c** in Figure 2. An example of inconsistent values shown in Figure 2 is the *Time-Zone* field that was only added in 1998, there was no input for the *Time-Zone* field in a PR recorded before 1998.

Data Transformation Data transformation is considered, in the way of converting attributes *Arrival-Date* and *Closed-Date* to a time-period - identifying the time spent to fix a PR - taking account the additional information *Time-Zone* and *Responsible*. This transformation resulted in the *Time-to-fix* attribute with continuous values. The *Responsible* attribute has the information about personnel engaged in rectifying the problem. We assume that the derived attribute *Time-to-fix* is total time spent to fix a problem if there is only one person involved. The calculated time period from *Arrival-Date* and *Closed-Date* is than multiplied by the number of people yield from the *Responsible* attribute. In order to improve the mining quality, we have discretized this attribute with cutting points be one day (1), half week (3 days), one week (7), two weeks (14), one month (30) and one quarter (90 days), half year (180 days) and more than one year (360 days). So that the mining results are not very highly depended on the exact human resource involved but gives an approximate estimate, allowing a minor change in human resource.

```
     PR_ID|Category|Severity|Priority|Class|Arrival-Date|Close-Date|Synopsis

     a. 17358|bambam|serious|high|sw-bug|20:50 May 25 CST 1999|11:35 Mar
        24 CST 1999| STI STR register not being reset at POR

     b. 17436|bambam|serious|high|support|18:10 Mar 30 CST 1999|12:00 May
        24 CST 1999| sequence_reg varable in the RDR_CHL task is not
        defined
     …
     c. 580 |bingarra|serious|low|doc-bug|10:10 May 31 May 1996|  | In
        URDRT2 of design doc, the word 'last' should be 'first'
     …
     d. 6205 |gali|serious|medium|SW-bug|14:30 Nov 5 1997|13:14 Dec 1
```

Figure 2: Data examples from the original PR data set

During the data cleaning stage, all the recoverable errors are corrected after identification. The PR data is then transferred into several data files to meet different data mining operation formats shown in Figure 3. For unrecoverable errors, the corresponding PR records are either discarded or put in further process if the error can be ignored or irrelevant with a specific data mining operation. Example c in Figure 3 has a '?' symbol as its Time-to-fix value, it is unrecoverable error but is still available for text mining since its Synopsis field provides correct information.

```
        Severity |Priority| Time-to-fix| Class |Synopsis

     a. serious, high, 61, sw-bug, STI STR register not being reset at POR
     b. serious, high, 56, support, sequence_reg varable in the RDR_CHL
        task is not defined
     c. serious, low, ?, doc-bug, In URDRT2 of design doc, the word 'last'
        should be 'first'
```

Figure 3: Data examples ready for mining

After data pre-processing, four attributes (time-to-fix, class, severity and priority) are chosen for mining process. Out of total 40000 PRs initially selected as the data set, we are left with 11,000 PRs after choosing only the PRs with state field as closed and after applying pre-processing steps. These 11,000

PRs (depends on different mining tasks, the numbers varies a little) cover more than 120 projects within MASC from 1996 to 2000. For example 11364 PR records have been applied with text-mining tools on the valid values in their Synopsis fields, as 364 records have no time values so could not be applied with classification tools.

3.3 *Data Mining Process: Data Modelling*

The GNATS system provides some simple methods to retrieve basic information from the PR data set, such as the PR numbers related to a particular person, etc. Besides this, some general data base techniques, such as SQL can also give some more useful information, i.e., the average time spent for fixing a PR in a project. But all these methods cannot perform in situations like performing queries over a large number of records with high dimensional structures, summarising a large data set to facilitate decision-making, making predictions on new data based on the existing rules, and visualising simplified extracted local structures. On the contrary, data mining techniques perform well on above situations and are able to reveal the deeper characters of the PR data. Some questions can only be answered after applying data mining techniques on the data set, such as:

- What type of project documents that needs a lot of time to have the development team fix its associated bugs in a project?
- If a PR being raised, how long should it be fixed according to the contents/values in its *Synopsis*, *Severity*, *Priority* and *Class*?

The selection of data mining operations and techniques is one of the most important things that directly affect the progress and the accomplishment of any DM applications. We have chosen:

- Prediction modelling on the time consuming patterns of the PR data and helping a project team to make estimation.
- Link analysis to discover association among the contents/values of the variables being selected.
- Text Mining to *analyse Synopsis* field.

The predictive modelling or classification task of data mining builds a model from the existing dataset by recognising the distinct characteristics of the data set. The model predicts future events based on previous data, specifying a class (or label) to each record in the dataset. There is a finite set of classes (or labels) that all the data is classified to. A supervised machine learning algorithm, that learns a model on previous or existing data, can be used for predictive modelling task. These models are developed over training and testing phases. The model is given some already known facts with correct answers during the training phase, from which the model learns to make accurate predictions. During the testing phase, the model is exposed to new data set to check the predictive capability. Three techniques namely neural induction, tree induction and bayesian classifiers are mainly used for classification data mining tasks [5]. Some other classification methods are K-nearest neighbour classifiers, case based reasoning, genetic algorithms, rough set and fuzzy set approaches [8].

Tree induction or decision tree is a suitable technique for this task. Decision tree has been quite popular in data mining due to their simplicity, efficiency and capability of dealing with a large number of training examples. The decision tree learning algorithms start by constructing a decision tree from top to bottom. Attributes are evaluated at each step to form descendant nodes. The attribute selection is based on a `statistical test' to determine how well it classifies the training examples. An internal node represents a test on an attribute and a leaf node represents a class or class distribution. Classification of unknown samples is made by tracing a path through the decision tree until a leaf node having the class prediction is reached. The maximum height of the decision tree depends on the number of attributes used to define the rules. Because the number of attributes in our problem is small, the resulting decision tree is relatively simple and thus its structure is understood easily by a human analyst.

The link analysis operation exposes samples and trends by predicting correlation of variables in a given data set that are otherwise not obvious. Association discovery builds a model to find variables implying the presence of other variables (with a certain degree of confidence and support) in the given data set. This process reveals hidden affinity among the variables i.e. which variables cause one another if a problem appears. The technique is based on counting occurrences of all possible combination of variables. Apriori and its variation algorithms [9] are most widely used.

In general, Data mining is knowledge discovery from structured databases. Text mining techniques, on the other hand, discover the knowledge from unstructured textual data. In order to discover and use the implicit structure of the texts, text mining techniques integrate some specific natural language processing to pre-process the textual data,

A suitable data-mining tool should satisfy the ease of use, low cost, ease of preparation and appropriate for the data model [6]. The C5 [2] is used for classification, CBA [1] is used for both classification and association rules mining, and TextAnalyst [3] is used for text mining.

3.4 ***Data Mining Process: Analysing the Results***

After applying several rounds of data mining operations to the data, sets of rules and results are achieved and stored. Due to the size of the results, only the main points are explained in this section.

Classification and Association Rule Mining In order to get better rules and to decrease the error rate as much as possible, several approaches are used. One approach is to stratify the data on the target using choice-based sampling rather than using random samples. Equal number of samples representing each possible value of the target attribute (‘**Class’** field) is chosen for training. The aim of this attempt is to improve the possibility of finding rules that are associated with small group of values during training. Another approach is to choose different number of PR data as training sets. We use three different training data sets. The first data set (Case 1) choses 1224 PRs from only one software project. The second one (Case 2) builds equal distributed value for a medium size of 3400 PRs (1000 PRs from each value of ‘**Class’** field) from all software projects. The third data set (Case 3) contains a large size of 5381 PRs from all software projects.

We use two learning engines to discover rules from the PR data set– single support CBA (labelled a in the Table 2 e.g., Case1a) and multiple support CBA (labelled b in the Table 2 e.g., Case1b). Constraints - support and confidence – are included in reporting output rules to control the quality of results. Confidence is the measure of the strength of a rule that indicates the probability of having consequence(s) in the rules provided that the rule contains certain antecedent(s). Support is the statistical significant measure of the strength of a rule indicating the number of input data supporting the rule. Since, we are usually interested in rules with worth consideration or more preferred or more certain, we set a threshold for support and confidence termed as minimum support and minimum confidence respectively. Only the rules with above minimum support and minimum confidence are considered. Setting a threshold for minimum support and confidence is a result of trial and error. If the factors are set too high, very few rules may be discovered. If the factors are set too low, too many rules may be generated with very low values. Under certain situation, attributes in the data are not likely to have uniform distributions, and many attributes are of very low frequency. Therefore a single support for all attributes may not be able to discover important rules that involve such rare attributes (very low frequency). This problem is relieved by setting multiple supports that allow user to choose different minimum supports to different attributes. Table 2 reports the classification mining results on all three cases and the associative rule mining results as Case4. The test data is 10 % of the whole data set and chosen randomly with the special consideration that each output class is representing in the test data.

	#Rules	Error rate (%)		Time cost (seconds)	
		Training data (90%)	Testing data (10%)	Training data	Testing data
Case1a	10	45.180	47.56	1.01	0.07
Case1b	9	45.180	47.56	1.04	0.09
Case2a	41	57.04	51.95	0.41	1.1
Case2b	21	59.10	58.25	0.44	1.0
Case3a	20	43.51	43.5	2.2	2.0
Case3b	15	46.5	45.1	1.6	1.9
Case4a	10	45.180	N/A	0.66	N/A
Case4b	9	45.180	N/A	1.04	N/A

Table 2: CBA Mining Results Summary

Example of an extracted classification rule is:

When a PR has a severity as non-critical and its priority is medium

Then the rule has 70.72% confidence to classify that this PR is a document type of PR with the support value of 6.5%.

The number of PRs in the training data set that satisfies the above conditions is 352; only 256 of them satisfy the above conclusion at the same time.

When the time-to-fix attribute is added to this rule, the rule becomes more stringent.

When a PR can be fixed between 21 days and 108 days, and the severity of the PR is non-critical and its priority is medium

Then the Rule has 82.703% confidence to classify that this PR is a document type of PR. with the support value of 2.7%.

The number of PRs in the training data set that satisfies the above conditions is 185; only 153 of them satisfy the above conclusion at the same time.

Based on all the rules, it can be summarised that software related bugs can be fixed within 3 days with above 75% confidence if they have high priority and are in critical condition. It may take 3 months to fix the problem if the corresponding priority and severity are graded as medium and serious. Certainly, the confidences of the rules are low, and only a small number of cases support these rules.

In general, all classification data mining operations in CBA software achieves around 46% Error rate in training data set (the lowest is 43.51%, the highest is above 59.10%). Above 51% correct prediction rate is achieved in testing data set (the lowest has 43.51%, the highest has 58.25%). Another interesting point is that the attempt to improve the accurate prediction in the way of equal-distributed target-value samples does not lead much change; there is only roughly 3% improvement over the final result. The error rates from using multiple supports are higher and the number of extracted rules is lower than those from using single support mining engine. There is no rule that has Confidence value larger than 80%, however they do describe some characters of the PR fixing patterns. Therefore they are useful for software project management in estimation bug fixing related time issues.

The CBA software is good at classification and association data mining on relation table records. It provides convenient and integrated features to help user for preparation work. The output format of the mining result is excellent, straightforward and easy for interpretation. The quality of the final result is average but not at ideal level. The error rate is still high and the stability of the system running needs to be improved.

To further improve and/or compare the quality of results obtained from CBA, the software C5 was also used to perform classification data mining with the boosting and cross validation techniques. The cross validation technique splits the whole data set into several subsets (called folds). Let each fold to be the test case and the rest as training sets in turn during training. This process may find a better result than a normal single training set mining process. Boosting is a technique for generating and combining multiple classifiers to give improved predictive accuracy. After a number of trials, several different decision trees or rule sets are combined to reduce error rate for prediction. Boosting takes a longer time to produce the final classifier, and may not always achieve better results than a single classifier approach does, especially when the training data set has noise.

	Normal mining		Mining with Boosting Techniques		Mining with cross-validation Techniques (10-fold)	
	10765 patterns	5681 Patterns	10765 patterns	5681 Patterns	10765 patterns	5681 Patterns
#Rules	51	11	N/A.	N/A	57.7	12.4
Error Rate (%) (Rules)	41.5	42.6	41.3	42.6	43.9	42.8
Error Rate (%) (Trees)	40.3	42.5	39.4	42.6	44.1	43.1
Size of tree	141	21	N/A.	N/A	121.9	17.4
Process Time (seconds)	5.6	0.2	37.7	0.4	41.1	1.1

Table 3: C5 Mining Results Summary

It should be noted that the above two techniques are not able to find any new rules. Instead, both of them try to find a better rule from existing results. Boosting only produces better results than the individual trees if the individual trees disagree with one another. For data mining operations conducted in C5, the results are summarised below in Table 3. Some example extracted classification rules are:

Rule 1: When a PR is in low priority and the time spent is around half a day (0.5 day)

Then the rule has a high probability (87.5% Confidence) to classify a bug to be a document related bug.

Rule 2: When a PR is in medium priority with non-critical severity and the time spent is around 1.1 day Then the rule has 84.6% Confidence to classify a bug to be a document related bug.

Rule 3: When a PR is in low priority and the time spent for fixing is around 1 week

Then the rule has 83.3% Confidence to classify a bug to be a software bug.

In general, all data mining operations achieves around 42% Error rate in rules from the training set (the lowest is 40.3%, the highest is 43.9%). Similar error rate value is achieved for the generated trees in testing data set (the lowest is 39.4%, the highest is 44.1%). Both of the rates are better than CBA results. Noticeably, if comparing the results from cross validation mining approach in C5 to the results of CBA, they are roughly at the same level considering CBA uses a similar mining algorithm as used by C4.5, which is the previous version of C5. The time efficiency of C5 is also better than CBA. Another thing should come into mind that boosting or cross validation is not able to find new rules, confirming that the aim of this techniques are to find a more accurate rule from existing results.

The UNIX version of C5 is graded as excellent at classification data mining on relation table records. It provides convenient way and general format to help user for preparation work. The pure text output of the mining result is simple and in details, with some difficulty for interpretation due to the use of the formula-like presentation for the final mining result. The speed of C5 is quicker than of CBA. The quality of the final result is also better than CBA. Although the above rules cannot guarantee of full correctness, they do give a project team some valuable information to help them in project estimation and bug fixing activities.

Text Mining in PR data In order to find valuable knowledge from thousands lines of text, we categorise the pure text into several document types based on certain background knowledge. For example, the following are PR values copied directly from the original PR data set.

serious high change-request 3.95 "SRS - missing requirements " *serious high change-request 5.05 "SRS - ability to turn off sending of SOH*

serious high change-request 5.92 "RT_014_SRS_3.0.0"

serious high change-request 6.7 "SRS - misc changes needed"

serious high change-request 15.71 "Changes needed to SCMP_2.0.0"

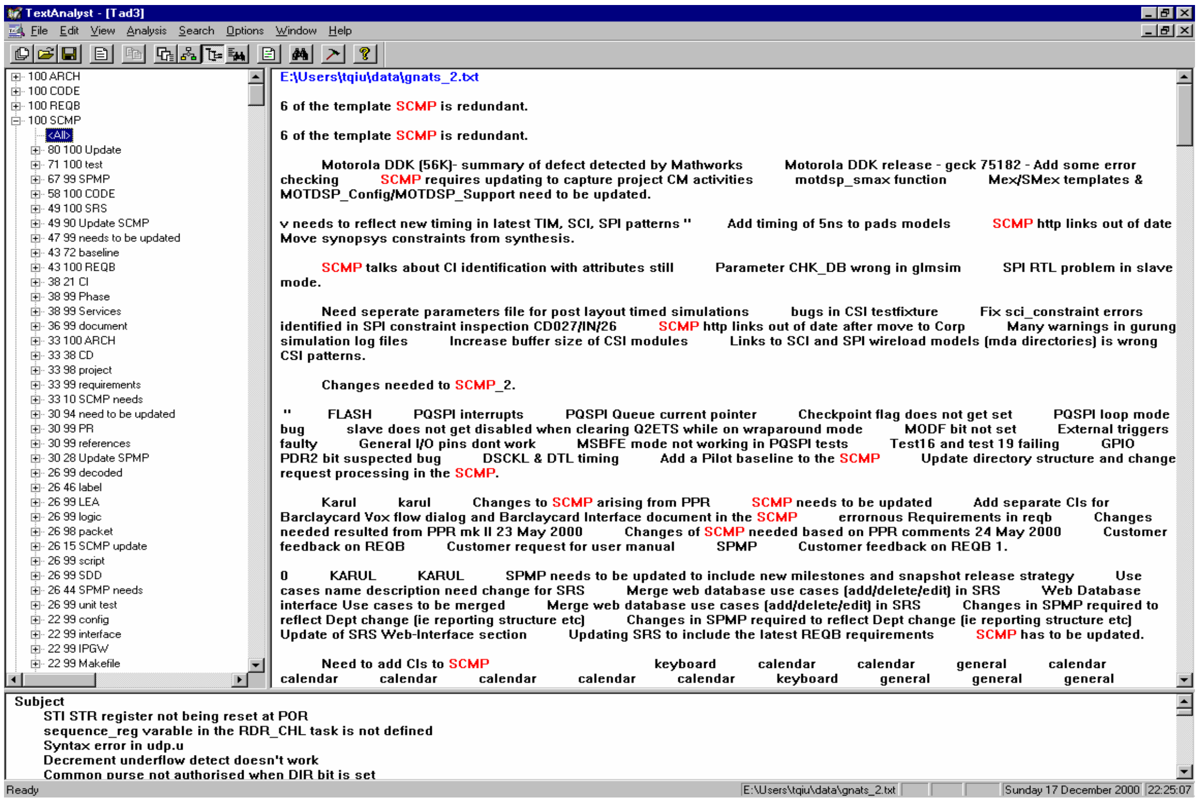
The pure text value of **synopsis** field appears on the most right side between two double quotes in PRs. It can be observed that 'SRS ' document (Software Requirement Specification) is the main target in 4 PRs, but SCMP (Software Configuration Management Plan) only appears once. To reveal the deep character of the PR data, it is not enough to only investigate what happens if the class, time or priority of a PR changes. It is also necessary to study what happens if the PR talking about different sources. Based on the above example, the analysis can reveal that SRS document costs more time than a SCMP document when fixing PRs. The question is: can this information be a valuable rule and apply in general? To answer this kind of question, we need a method and a tool that is able to automatically summarise the pure text data and extract some valuable rules.

TextAnalyst tool has been used to do the text-mining job in this research. It builds up a semantic network for the investigation over the PR data. Each element of the semantic network is characterised by a weight value and a set of relationship of this element to other elements in the network. Every relationship between elements is also assigned a weight value. The semantic network can then provide a concise and accurate summary of the analysed text.

The good thing is that, TextAnalyst does not need the user to specify any predefined rules for building the semantic network. It can automatically create the semantic network based on the structure, vocabulary and volume of the analysed text. As seen from the simple graphical user interface window in TextAnalyst (Figure 4), a semantic network tree of the PR data is shown in the upper-left window. The network contains a set of the most important words or word combinations, called *concepts*, extracted from the PR data set. The relations among those concepts together with the semantic weights of concepts and relations are also shown. The values of the weights range from 0 to 100, which correspond to the probability that the associated concept is characteristic for the whole PR data. Let us take the second concept "**test**" below "**SCMP**" as an example. "71"is the weight of the relation between the concept "**test**" and its parent concept "**SCMP**". "100" is the semantic weight of "**test**" itself. "+" sign means "**test**" node can be further opened to view related concepts in the network, and the "-" sign beside "**SCMP**" means the "**SCMP**" node is already opened.

The upper-right window shows all the text related to a concept if the user clicks the corresponding concept on the upper-left window with the text displayed in red colour. Currently all pure text value related to concept "**SCMP**" are shown in the window. The third window on the bottom contains the whole value of **synopsis** in pure text form.

Figure 4: Text Analyst Mining: An interface



```
100 ARCH
100 CODE
100 REQB
100 SCMP
  <All>
  80 100 Update
  71 100 test
  67 99 SPMP
  58 100 CODE
  49 100 SRS
  49 90 Update SCMP
  47 99 needs to be updated
  43 72 baseline
  43 100 REQB
  38 21 CI
  38 99 Phase
  38 99 Services
  36 99 document
  33 100 ARCH
  33 38 CD
  33 98 project
  33 99 requirements
  33 10 SCMP needs
  30 94 need to be updated
  30 99 PR
  30 99 references
  30 28 Update SPMP
  26 99 decoded
  26 46 label
  26 99 LEA
  26 99 logic
  26 98 packet
  26 15 SCMP update
  26 99 script
  26 99 SDD
  26 44 SPMP needs
  26 99 unit test
  22 99 config
  22 99 interface
  22 99 IPGW
  22 99 Makefile
```

```
E:\Users\tqiu\data\gnats_2.txt

6 of the template SCMP is redundant.

6 of the template SCMP is redundant.

        Motorola DDK (56K)- summary of defect detected by Mathworks        Motorola DDK release - geck 75182 - Add some error
checking        SCMP requires updating to capture project CM activities        motdsp_smax function        Mex/SMex templates &
MOTDSP_Config/MOTDSP_Support need to be updated.

v needs to reflect new timing in latest TIM, SCI, SPI patterns "        Add timing of 5ns to pads models        SCMP http links out of date
Move synopsys constraints from synthesis.

        SCMP talks about CI identification with attributes still        Parameter CHK_DB wrong in glmsim        SPI RTL problem in slave
mode.

        Need seperate parameters file for post layout timed simulations        bugs in CSI testfixture        Fix sci_constraint errors
identified in SPI constraint inspection CD027/IN/26        SCMP http links out of date after move to Corp        Many warnings in gurung
simulation log files        Increase buffer size of CSI modules        Links to SCI and SPI wireload models (mda directories) is wrong
CSI patterns.

        Changes needed to SCMP_2.

 "      FLASH        PQSPI interrupts        PQSPI Queue current pointer        Checkpoint flag does not get set        PQSPI loop mode
bug        slave does not get disabled when clearing Q2ETS while on wraparound mode        MODF bit not set        External triggers
faulty        General I/O pins dont work        MSBFE mode not working in PQSPI tests        Test16 and test 19 failing        GPIO
PDR2 bit suspected bug        DSCKL & DTL timing        Add a Pilot baseline to the SCMP        Update directory structure and change
request processing in the SCMP.

        Karul        karul        Changes to SCMP arising from PPR        SCMP needs to be updated        Add separate CIs for
Barclaycard Vox flow dialog and Barclaycard Interface document in the SCMP        errornous Requirements in reqb        Changes
needed resulted from PPR mk II 23 May 2000        Changes of SCMP needed based on PPR comments 24 May 2000        Customer
feedback on REQB        Customer request for user manual        SPMP        Customer feedback on REQB 1.

0        KARUL        KARUL        SPMP needs to be updated to include new milestones and snapshot release strategy        Use
cases name description need change for SRS        Merge web database use cases (add/delete/edit) in SRS        Web Database
interface Use cases to be merged        Merge web database use cases (add/delete/edit) in SRS        Changes in SPMP required to
reflect Dept change (ie reporting structure etc)        Changes in SPMP required to reflect Dept change (ie reporting structure etc)
Update of SRS Web-Interface section        Updating SRS to include the latest REQB requirements        SCMP has to be updated.

        Need to add CIs to SCMP                          keyboard        calendar        calendar        general        calendar
calendar        calendar        calendar        calendar        calendar        keyboard        general        general        general
```

```
Subject
    STI STR register not being reset at POR
    sequence_reg varable in the RDR_CHL task is not defined
    Syntax error in udp.u
    Decrement underflow detect doesn't work
    Common purse not authorised when DIR bit is set
```

```
Ready                    E:\Users\tqiu\data\gnats_2.txt    Sunday 17 December 2000    22:25:07
```

Text mining is applied on a total 11226 cases. An interesting relation is researched with "SCMP", Software Configuration Management Plan, a support document in every MASC project. Since SCMP is not a main design document for a project with just tens pages, it has never been considered as a trouble making item. But amazingly, it is clear from the figure that SCMP has 71 percentage probability of appearing in test related PR records, and 58 percentage probability of appearing in Code related PR records. This result is even higher than the result associated with SRS ("Software Requirements Specification", a main development document directly related to software). Although we can not say SCMP causes more problems than SRS, but the higher appearance of SCMP inside test related PR definitely shows a warning. It is worth for software engineers to be more careful when dealing with SCMP document, and hence reducing the total cost of fixing SCMP related PRs. Another analysis shows that a test related PR has even a higher weight (36, 100) in document related bugs than a SRS related PR (35, 99) does. Which suggests a better project management should not only focus on the quality of product related documents, but also pay attention on the quality of testing related documents.

In general, TextAnalyst is a useful tool to perform analyses on data in pure text format. Its automatic mechanism of semantic network creation is very attractive, and saves time for data preparation.

Existing problems in performing mining The error rates of testing data sets in both CBA and C5 are higher than expected. Although several approaches are attempted to reduce the error such as uniform distribution of values, cross validation, boosting, different size of training set, etc. Unfortunately, the average error rate is only fallen down by 5% from 47% to 42%. The best result is 9% decrease from 46% to 37%. These results indicate that some amount of noise is still existent in data after dealing with the noise during the pre-processing phase.

For example, the relationship between PRs and human resources within a particular project plays a great impact. The time needed to fix a bug is different depending upon the actual human resources available. We have used only the attribute '*Responsible*' to indicate the human resource available. Truly, the relationship with the human resources available for past projects whose data was analysed is needed to use time patterns to help project leaders to predict time consummations more accurately. The use of additional data source 'Change Request data set' that records all customer request process data may rectify this problem.

Another reason is a non-uniform value distribution. For example, there are only 342 PRs with *change-request* value in the data set, compared to more than 5900 PRs related to *sw-bug*. Any potential rule associated to *change-request* can be heavily affected due to the presence of large size group with other values. Again the use of additional data sources together with the PR data set can rectify this problem. We also attempted to use neural network techniques, but it is difficult to interpret the outputs.

4. Future Directions and Conclusion

During the life cycle of a software project development, there are many problems such as negotiation with customers, project estimation, project planning and scheduling have to be dealt efficiently. Resolutions to these problems are time consuming and costly. This paper considers the use of data mining techniques to analyse these problems, and further find valuable rules to reduce the effort of fixing those.

The data mining result may bring some relief of the difficulties in planing, estimation and bug fixing activities for software project teams. The time patterns rules may help a project leader to understand knowledge in numeric values about the PR fixing process, the leader can then estimate or predict time consummations more accurately than before. Another finding suggests that bug fixing efforts have more probabilities to be spent on test related PR. This could cost the project team a lot of time in fixing non-product-related problems. By reducing such type of PR, the total time in bug fixing is then reduced and project efficiency will be improved.

Several data mining operations are executed on a data set collected from the software engineering process under a real software business environment. Some useful rules and background knowledge are extracted on the time patterns of the PR fixing and the relationship between the content and the type of a PR in the form of association rules, classification rules or semantic trees. Results of the application indicate that data mining techniques are capable in software engineering domain even though the scale of the data mining task is limited. As in many other domains, the benefits and capabilities brought by data mining in software engineering domain are worth of further investigations.

References

1. CBA, http://www.comp.nus.edu.sg/~dm2/
2. C5.0, http://www.rulequest.com/see5-info.html
3. TextMiner, http://www.megaputer.com/company/index.html
4. Tlearn, http://crl.ucsd.edu/innate/tlearn.html
5. P.Cabena, P. Hadjinian, R. Stadler, J. Verhees, and A. Zanasi, “*Discovering Data Mining: From Concept to Implementation*”, ISBN 0-13-743980-6, 1997
6. H. Edelstein, “*Mining for gold. (Selecting data mining tools)*”, Information Week, April 21, 1997 n627 p53 (6).
7. U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “'From Data Mining to Knowledge Discovery: An Overview”, In Advances in Knowledge Discovery and Data Mining. Eds. U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAI Press, 1996
8. J. Han and M. Kamber, “*Data mining: concepts and techniques*”, ISBN 1-55860-489-8, 2001
9. C. Westphal, and T. Blaxton, “Data Mining Solutions: Methods and Tools for Solving Real-World Problems”, ISBN 0471-253847, John Wiley & Sons, Inc, 1998.
10. J Furnkranz and M. Kamber, “First-Order Knowledge Discovery in Database”, In Applied Artificial Intelligence, 0883-9514/98, 1998