Empirical Evaluation of Cost Overrun Prediction with Imbalanced Dataset

Masateru Tsunoda, Akito Monden, Jun-ichiro Shibata and Ken-ichi Matsumoto
Graduate School of Information Science
Nara Institute of Science and Technology
Nara, Japan
{masate-t, akito-m, junichiro-s, matumoto}@is.naist.jp

Abstract*—To prevent cost overrun of software projects, it is necessary for project managers to identify projects which have high risk of cost overrun in the early phase. So far, discriminant methods such as linear discriminant analysis and logistic regression have been used to predict cost overrun projects. However, accuracy of discriminant methods often becomes low when a dataset used for predict is imbalanced, i.e. there exists a large difference between the number of cost overrun projects and non cost overrun projects. In this paper, we compared accuracy of linear discriminant analysis, logistic regression, classification tree, Mahalanobis-Taguchi method, and collaborative filtering, by changing the percentage of cost overrun projects in the dataset. The result showed that collaborative filtering was highest accuracy among five methods. When the number of cost overrun projects and non cost overrun was balanced in the dataset, linear discriminant analysis was second highest accuracy, and when it was not balanced, Mahalanobis-Taguchi method was second highest among five methods.

Keywords-biased data; failure prone project; Collaborative Filtering; Mahalanobis-Taguchi method; risk management

I. INTRODUCTION

Recently, software is widely used as a part of infrastructure of the our daily life such as banking system and air traffic control system, while software size and cost (i.e. development effort) became extremely larger than ever. As a result, one single overrun project can cause serious damage to the profit of a software development company. Therefore, prevention of cost overrun became extremely important today.

One effective way to prevent cost overrun is to identify the project which has high risk of cost overrun (project failure) in the early phase of the project [10][15] so that countermeasures can be performed. To predict the project result (project failure), discriminant methods such as linear discriminant analysis or logistic regression has been used [10][15][16]. On a discriminant method, the project result is set as dependent variable, and its value (i.e. cost overrun or not) is predicted from independent variables which are known at prediction point of time. Usually, project manager's answers for questionnaires related to risk factors (for example, the question is "Insufficient explanation of the requirements" [15]) are used as independent variables for project result prediction model [10][15][16]. The model is built from past projects' data, and current project's data is input as independent variables to predict the project result.

However, accuracy of discriminant methods often becomes low when imbalanced dataset is used for prediction [5]. The imbalanced dataset means that there exists a large difference between the number of cost overrun projects and non cost overrun projects. For example, in the company whose organizational maturity level is high (e.g. CMMI (Capability Maturity Model Integration) level is over 2), there would be less cost overrun projects, and that makes the percentage of cost overrun projects low.

In this paper, we focus on Mahalanobis-Taguchi method and collaborative filtering to apply cost overrun prediction. Mahalanobis-Taguchi method is used as one of the techniques for quality control of the manufacturing industry. It builds a model using only normal cases (i.e. non cost overrun projects), and predicts the project result based on the distance from the normal case group. Therefore, it is expected that the model is not affected by imbalance of the dataset. On the other hand, collaborative filtering is originally used for the item (books or music) recommender system. Collaborative filtering is based on k-nearest neighbor algorithm, as the analogy based estimation method [13]. Roughly speaking, collaborative filtering finds projects similar to the target project, and makes prediction based on values of dependent variable of similar projects. We applied it to cost overrun prediction since the dataset used for prediction is similar to the dataset treated by collaborative filtering.

We analyzed accuracy of discriminant methods when the percentage of cost overrun projects and that of non cost overrun are imbalanced. In the experiment, we changed the percentage of cost overrun projects by deleting cost overrun projects in the dataset whose data was collected in a software development company, and predicted the project result with linear discriminant analysis, logistic regression, classification tree, Mahalanobis-Taguchi method, and collaborative filtering. The result of the experiment makes practitioners choose discriminant methods more appropriately.

In what follows, Section II explains discriminant methods used in the experiment. Section III describes procedure of the experiment, the evaluation criterion of the method, and the dataset used in the experiment. Section IV shows results of the experiment and discusses it. Section V introduces related works. In the end, Section VI concludes the paper with a summary.

II. Discriminant Methods

The discriminant method builds a prediction model using a dataset which includes finished projects whose dependent variable is already known (i.e. cost overrun or not). The result of an unfinished project is predicted by the model.

We evaluated accuracy of five types of discriminant methods for predicting the project result. Linear discriminant analysis, logistic regression and classification tree are widely used as discriminant methods in the software engineering field [6][9][15]. In addition, we applied collaborative filtering and Mahalanobis-Taguchi method to predicting the project result, for they are expected to fit to the dataset.

A. *Linear discriminant analysis*

Linear discriminant analysis makes a line which divides a dataset into two groups based on a dependent variable. The model of the line is:

$$y = a_1 x_1 + \cdots a_n x_n + b \tag{1}$$

In the model, y is a discriminant score, x_n are independent variables, a_n are regression coefficients, and b is an intercept. The dependent variable is predicted by whether the discriminant score is plus or minus.

B. *Logistic regression*

Logistic regression predicts a dependent variable based on a logistic function. The model of logistic regression is:

$$y = \frac{1}{1 + e^{-(a_1 x_1 + \cdots a_n x_x + b)}} \tag{2}$$

In the model, y indicates probability of the dependent variable, x_n are independent variables, a_n are regression coefficients, and b is an intercept. The dependent value is predicted as probability. For example, when the value y is 0.7, probability of that predicted project will belong to one group is 70%.

C. *Classification tree*

Classification tree predicts a dependent variable by a tree structure model which has leafs and nodes. Each leaf indicates the predicted value of the dependent variable, and each node has a condition related to one of independent variables. Based on independent variables, a path on the model is chosen, and it predicts the dependent variable.

There are some algorithms to build classification tree. For example, CART (classification and regression trees) algorithm uses Gini index as a model building criterion, and ID3 (Iterative Dichotomiser 3) and C4.5 algorithms use information gain. In this paper, we used CART implemented on R [12].

D. *Mahalanobis-Taguchi method*

Mahalanobis-Taguchi method [14] was proposed by Taguchi, and it is used as one of the techniques for quality control of the manufacturing industry. Mahalanobis-Taguchi method assumes that although normal cases (projects) are similar to other normal cases, abnormal cases are not similar to each other because they became abnormal cases for different reasons. That is, Mahalanobis-Taguchi method assumes that when cost overrun is not occurred, the reason is similar to other non cost overrun projects, but when cost overrun is occurred, the reason of cost overrun is different from other cost overrun projects. This assumption was inspired by the sentence in Anna Karenina, a novel written by Tolstoy. It said "Happy families are all alike; every unhappy family is unhappy in its own way." Based on the assumption, Mahalanobis-Taguchi method builds a model using only normal cases (i.e. non cost overrun projects). Mahalanobis-Taguchi method is used in few software engineering researches [1].

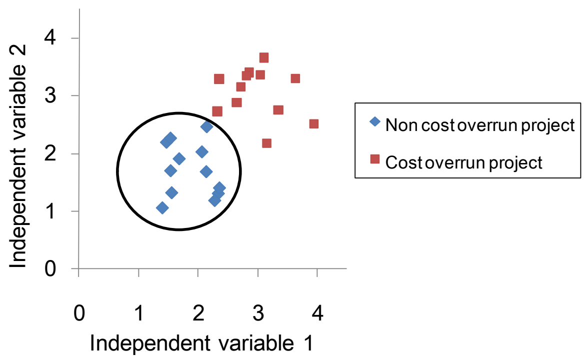

Figure 1. An example of the model of Mahalanobis-Taguchi method.

Mahalanobis-Taguchi method predicts a case (project) as abnormal one (cost overrun project) when the case is distant from a normal case group (non cost overrun project group). The distance is calculated by:

$$D_a^2 = \frac{1}{k} \sum_{ij} r_{ij} \left(\frac{m_{ai} - \overline{m}_i}{s_i} \right) \left(\frac{m_{aj} - \overline{m}_j}{s_j} \right) \tag{3}$$

In the equation, D_a indicates Mahalanobis distance of predicted project, k is the number of independent variables in the dataset, r_{ij} is the inverse correlation matrix, m_{ai} is the value of *i-th* independent variable of the predicted project, s_i is the standard deviation of *i-th* independent variable, and $\overline{m}_i$ is the average of *i-th* independent variable. Values of r_{ij}, s_i ,and $\overline{m}_i$ are calculated based on normal cases in the dataset. If D_a is greater than a certain threshold, the project is predicted as a cost overrun project. Fig. 1 illustrates an example of the model of Mahalanobis-Taguchi method. In the figure, the border line indicates the threshold of Mahalanobis distance. If predicted project is out of the border line area, the project is predicted as cost overrun project.

E. *Colaborative filtering*

Originally, collaborative filtering is used for the recommender system which estimates users' preferences to recommend items such as books or music. Collaborative filtering presumes "Users who have similar preferences like similar items." Few software engineering researches used collaborative filtering for prediction [8].

Collaborative filtering uses $m\times n$ matrix shown in Table 1. In the matrix, $Proj_i$ is *i-th* project, Q_j is *j-th* independent variable, v_{ij} is a value of Q_j of $Proj_i$, and y_i is the value of the dependent variable. We presume $Proj_a$ is predicted project, and $\hat{y}_a$ is the predicted value of y_a. Procedures of collaborative filtering consist of the three steps described below.

Step 1 (normalization): Since a dependent variable and independent variables have different ranges of value, this step makes the ranges [0, 1]. The value v'_{ij}, normalized the value of v_{ij} is calculated by:

$$v'_{ij} = \frac{v_{ij} - \min(Q_j)}{\max(Q_j) - \min(Q_j)} \tag{4}$$

In the equation, $\max(Q_j)$ and $\min(Q_j)$ denote the maximum and minimum value of Q_j respectively.

Step 2 (similarity computation): This step computes similarity $\mathrm{Sim}(Proj_a, Proj_i)$ between the predicted project p_a and other projects p_i by:

$$\mathrm{Sim}(Proj_a, Proj_i) = \frac{\sum_{h=1}^{m}(v'_{ah} - \overline{Q}'_h)(v'_{ih} - \overline{Q}'_h)}{\sqrt{\sum_{h=1}^{m}(v'_{ah} - \overline{Q}'_h)^2}\sqrt{\sum_{h=1}^{m}(v'_{ih} - \overline{Q}'_h)^2}} \tag{5}$$

In the equation, $\overline{Q}'_h$ is average of Q_h based on v'_{ij}. With the equation, the value v'_{ij} which is higher than average shows positive values, and lower than average shows negative values to sharpen differences between projects. The range of the value of $\mathrm{Sim}(Proj_a, Proj_i)$ is [-1, 1].

Step 3 (computation of predicted value): The predicted value is computed by weighted average of the independent variable of similar projects. Formally, the predicted value is computed by:

$$\hat{y}'_a = \frac{\overline{v}'_a + \sum_{h\in Sumprojects}\mathrm{Sim}(Proj_a, Proj_h)(y'_h - \overline{v}'_h)}{\sum_{h\in Sumprojects}\mathrm{Sim}(Proj_a, Proj_h)} \tag{6}$$

In the equation, *Simprojects* denotes the set of k projects (*neighborhoods*) which have top similarity with $Proj_a$. The neighborhood size k affects prediction accuracy. The value $\hat{y}'_a$ is the normalized value of $\hat{y}_a$. The value $\overline{v}'_h$ is the average of v'_{ih} included in $Proj_h$. On the recommender system, collaborative filtering uses users' ratings for items. However, some people tend to rate every item as high, and on the other hand, some do as low. Hence, this equation uses difference from average of each people's rating. We applied this algorithm to predict the project result, because our dataset seems to have similar characteristic.

TABLE I. MATRIX USED BY COLLABORATIVE FILTERING

	Result	Q_1	Q_2	…	Q_j	…	Q_n
$Proj_1$	y_1	v_{11}	v_{12}	…	v_{1j}	…	v_{1n}
$Proj_2$	y_2	v_{21}	v_{22}	…	v_{2j}	…	v_{2n}
…	…	…	…		…		…
$Proj_i$	y_i	v_{i1}	v_{i2}	…	v_{ij}	…	v_{in}
…	…	…	…		…		…
$Proj_m$	y_m	v_{m1}	v_{m2}	…	v_{mj}	…	v_{mn}

III. EXPERIMENT

A. Overview

In the experiment, to clarify proper discriminant methods for predicting the project result, we evaluated accuracy of discriminant methods when the percentage of cost overrun projects and that of non cost overrun were imbalanced. Using 28 projects data collected in a software development company, we changed the percentage of cost overrun projects by deleting projects in the dataset from 50.0% (14 cost overrun projects) to 6.7% (1 cost overrun project), and applied discriminant methods. The Methods used in the experiment were linear discriminant analysis, logistic regression, classification tree, Mahalanobis-Taguchi method, and collaborative filtering.

B. Dataset

We used the questionnaire about the software project as the dataset for prediction. Project data in the dataset were collected in the 2000s. The questionnaire is originally used for project management. Although details of the questionnaire does not described due to confidential, it is similar to the questionnaire shown in a software project management guidebook [4], which consists of questions related to 9 knowledge areas of PMBOK (Project Management Body of Knowledge) [11]. Similar questionnaires are also used in other project result prediction researches [10][15][16]. For example, one of the questions is "If new or unexperienced technologies are used in the project, is the project plan sufficient to cope with them? [4]" (Note that it is not entirely equal to the question in our dataset). Questions are rated as "high risk", "middle risk", "low risk", or "unrelated" based on probability of project failure. If new technology is used but the project plan is insufficient, the question is rated as "high risk". If new technology is not used, it is rated as "unrelated".

We used these questions as independent variables of prediction models. Only questions whose answers were clear at early phase of the project were used as independent variables. Ratings "high risk", "middle risk", "low risk", or "unrelated" were converted to numerical values (4, 3, 2, and 1) before applying discriminant methods.

Cost overrun was set as the dependent variable. We defined cost overrun as overrun of actual cost from estimated cost. Projects whose cost overrun were greater than certain threshold were defined as cost overrun projects, and values of their dependent variable were set as 1. Other projects were defined as non cost overrun projects, and values of their dependent variable were set as 0. Note that the threshold is not disclosed in this paper because of confidential. Although there are more than 100 projects in the dataset, cost overrun projects are fairly fewer than non cost overrun projects. Therefore, we randomly extracted projects from the dataset to adjust the balance of cost overrun and non cost overrun projects.

The dataset contains 120 questions. When the number of independent variables is much greater than the number of cases (projects), it is difficult to build a prediction model appropriately (curse of dimensionality). So when correlation coefficient between the project result and the question was greater than or equal to 0.2, we used the question as the independent variable. Moreover, we eliminated questions which have missing values, to avoid influence of them. As a result, 7 questions were selected for independent variables.

C. *Evaluation criterion*

We used area under the curve (AUC) [2] as the evaluation criterion of discriminant methods. AUC is recently used to evaluate discriminant methods in software engineering researches, for it is more appropriate criterion for discriminant methods than other criteria like F1 score [7]. The value range of AUC is [0, 1], and higher AUC means that prediction accuracy of the method is high. AUC is defined as the area under the receiver operating characteristic (ROC) curve. ROC curve is drawn by changing threshold and calculating true positive rate and false positive rate. These rates are calculated by:

$$\text{True Positeve Rate} = \frac{TP}{TP + FN} \tag{7}$$

$$\text{False Positive Rate} = \frac{FP}{FP + TN} \tag{8}$$

Definitions of *TP* (true positive), *FN* (false negative), *FP* (false positive), and *TN* (true negative) are shown in Table 2. Although high true positive rate and false positive rate means high accuracy, there is tradeoff between them, and they depend on a threshold. For example, if prediction is done by logistic regression and the threshold is set as 0, true positive rate is very high but false positive rate is very low. AUC can evaluate performance of discriminant methods independently from the threshold.

D. *Exprrimantal Procedure*

We changed the percentage of cost overrun and non cost overrun projects, and predicted the project result according to the following procedure. To avoid biased results, the procedure was repeated 10 times.

1. 14 cost overrun projects and 14 non cost overrun projects were randomly selected from the dataset to make a learning dataset (i.e. the learning dataset contains 28 projects).

TABLE II. *DEFINITIONS OF TP, FN, FP, AND TN*

		Actual value	
		True	False
Predicted value	True	*TP*	*FP*
	False	*FN*	*TN*

2. A test dataset was made in the same way (the test dataset did not include as same projects as the learning dataset).
3. Prediction models of five discriminant methods were built using the learning dataset.
4. Each model was applied to the test dataset to predict the project result, and the evaluation criterion (AUC) was computed.
5. One cost overrun project was randomly deleted from the learning dataset.
6. Step 3 to 5 were repeated until the number of cost overrun projects in the learning dataset was equal to 1.

The percentage of cost overrun projects was fixed as 50.0% in the test dataset. When a model of Mahalanobis-Taguchi method was built, step 5 and 6 were not performed because Mahalanobis-Taguchi method does not use cost overrun project to build the model, and hence accuracy of the model is not affected by the percentage. The neighborhood size of collaborative filtering was set as 5. When using linear discriminant analysis and logistic regression, both variable selection model and non variable selection model were built, because these discriminant methods have commonly-used variable section method.

IV. RESULTS AND DISCUSSION

Results of the experiment are shown in Fig. 2. In the figure, the virtual axis indicates AUC and the horizontal axis indicates the number of cost overrun projects. AUC is average of 10 results of the experiment. Regardless of the number of cost overrun projects, collaborative filtering showed highest accuracy among five discriminant methods.

Table 3 shows one of 10 prediction results by collaborative filtering when number of cost overrun projects in the learning dataset was 1 (AUC was 0.77). As shown in the table, collaborative filtering chose similar projects appropriately. Although projects P27 and P28 did not have similar projects whose actual values were 1, predicted values were 1. This is because when average of independent variables is high, predicted value of the dependent variable is also high by (6). Note that even if average of independent variables is high, predicted value of the dependent variable can be low as shown in Table 4. Equation (6) reflects the assumption that in particular projects, dependent variable can be low in spite of high average of independent variables. Actually, average values of independent variables of P08, P10, and P11 were same as P28, but predicted values were different from P28.

The reason of high accuracy of collaborative filtering would be that both our dataset and a dataset used by a recommender system based on collaborative filtering have

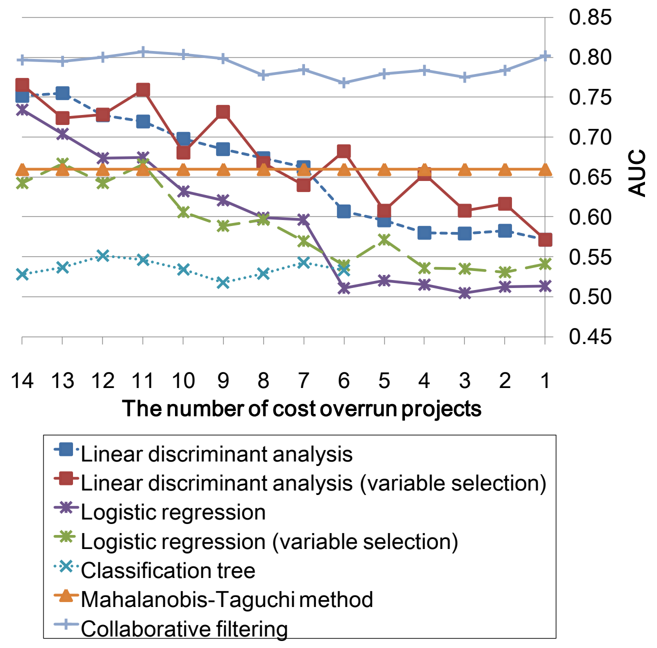

Figure 2. Relationship between accuracy and the number of cost overrun projects.

TABLE III. *PREDICTION RESULTS BY COLLABORATIVE FILTERING*

	Predicted value	Actual value	Actual value of similar projects				
P01	0	0	0	0	0	0	0
P02	0	0	0	0	0	0	0
P03	1	0	0	1	0	0	0
P04	0	0	0	0	0	0	0
P05	1	0	0	1	0	0	0
P06	0	0	0	0	0	0	0
P07	0	0	0	0	0	0	0
P08	0	0	0	0	0	0	0
P09	0	0	0	0	0	0	0
P10	0	0	0	0	0	0	0
P11	0	0	0	0	0	0	0
P12	1	0	0	1	0	0	0
P13	1	0	0	1	0	0	0
P14	0	0	0	0	0	0	0
P15	1	1	0	0	0	1	0
P16	1	1	1	0	0	0	0
P17	1	1	1	0	0	0	0
P18	1	1	0	1	0	0	0
P19	1	1	0	1	0	0	0
P20	0	1	0	0	0	0	0
P21	0	1	0	0	0	0	0
P22	1	1	0	1	0	0	0
P23	0	1	0	0	0	0	0
P24	1	1	0	0	1	0	0
P25	1	1	0	0	0	1	0
P26	0	1	0	0	0	0	0
P27	1	1	0	0	0	0	0
P28	1	1	0	0	0	0	0

similar characteristics (they are rating data). Moreover, accuracy of the method kept high when the percentage of cost overrun projects got low. Collaborative filtering does not build a model before prediction but uses prepared equation (lazy learning). As a result, when collaborative filtering fits well to a dataset, accuracy is not considered to be affected very much by imbalance of the dataset. Therefore, collaborative filtering is most suitable for predicting the project result, no matter what the percentage of cost overrun projects is.

When the percentage of cost overrun projects and non cost overrun projects was balanced, accuracy of Mahalanobis-Taguchi method was lower. One of the reasons may be that Mahalanobis-Taguchi method does not use cost overrun projects to build a prediction model, and consequently, available information amount is less than other discriminant methods. However, when the percentage of cost overrun projects was lower than 30.0% (6 cost overrun projects), accuracy of Mahalanobis-Taguchi method was second among five discriminant methods. This would be because other discriminant methods were overly affected by non cost overrun projects, and as a result, accuracy of Mahalanobis-Taguchi method was comparatively higher. Although Mahalanobis-Taguchi method is not fit well for predicting the project result, it can be a candidate of prediction methods when the dataset is imbalanced.

Accuracy of classification tree was the lowest in the discriminant methods except the percentage of cost overrun projects was 30.0% (6 cost overrun projects). Thus, classification tree is not suitable for project result prediction. Also, classification tree failed to build a model when the percentage of cost overrun projects was low. When the percentage of cost overrun projects was lower than 39.1% (9 cost overrun projects), classification tree failed to build a model in some cases of 10 repeated experiments, and when the percentage was lower than 30.0%, it failed to build models in all cases. The result means that classification tree is greatly affected by imbalance of a dataset.

Accuracy of linear discriminant analysis was higher than logistic regression at any percentage of cost overrun projects. In linear discriminant analysis models, accuracy of models applied variable selection was almost higher than not applied. In logistic regression models, when the percentage of cost overrun projects was high, accuracy of models applied variable selection was lower than not applied, but when the percentage was low, accuracy of models applied variable selection was higher. So affects of the percentage of cost overrun projects for variable selection is considered to be different for discriminant method types.

V. RELATED WORK

There are some researches about project result prediction such as cost overrun [10][15][16]. For instance, Takagi et al. [15] proposed delivery delay project prediction method using logistic regression and a questionnaire about the project. However, these researches did not compare accuracy of discriminant methods, changing the percentage of failure projects in a dataset, and therefore they did not clarify which method is better when a dataset is imbalanced.

In the software engineering field, there are few researches using Mahalanobis-Taguchi method or collaborative filtering. Aman et al. [1] proposed the prediction method which identifies program modules whose

TABLE IV. *RELATIONSHIP BETWEEN PREDICTED VALUE AND SIMILAR PROJECTS*

Similar projects (Learning dataset)		Predicted project (Test dataset)	
Dependent variable	Average of independent variable	Average of independent variable	Predicted dependent variable
Low	High	High	Low
Low	Low	High	High

modification effort is high by Mahalanobis-Taguchi method. Motomura et al. [8] proposed cost overrun project prediction using collaborative filtering. Again, these researches did not clarify accuracy of discriminant methods when a dataset is imbalanced.

As contrasted to Mahalanobis-Taguchi method, positive unlabeled learning builds a prediction model without negative cases (i.e. use only cost overrun projects). Hata et al. [3] proposed applying positive naive Bayes to predict fault prone modules. One of our future research issues is to apply positive naive Bayes to project result prediction.

Kamei et al. [5] proposed applying oversampling to an imbalanced dataset before predicting fault prone modules. Oversampling duplicates one group whose number of cases is smaller than the other group, to align imbalance of the dataset. The other future issue of our research is applying oversampling and comparing accuracy of discriminant methods, changing the percentage of cost overrun projects.

VI. CONCLUSIONS

In this paper, we compared accuracy of discriminant methods, changing the percentage of cost overrun projects. We predicted cost overrun projects by linear discriminant analysis, logistic regression, classification tree, Mahalanobis-Taguchi method, and collaborative filtering, using the questionnaire about the software project. The result showed that collaborative filtering was highest accuracy among five discriminant methods. When the dataset was not very imbalanced, linear discriminant analysis was second highest in the methods, and when it is imbalanced, Mahalanobis-Taguchi method was second highest. Classification tree was not fitted to project result prediction. Our future works are to apply positive naive Bayes and oversampling to predict the project result, changing the percentage of cost overrun projects.

ACKNOWLEDGMENT

This work is being conducted as a part of the StagE project, The Development of Next-Generation IT Infrastructure, and Grant-in-aid for Young Scientists (B), 22700034, 2010, supported by the Ministry of Education, Culture, Sports, Science and Technology. We are deeply grateful to Mr. Yutaka Fukuchi (Hitachi Ltd.), Mr. Tetsuya Yonemitsu (Hitachi Ltd.), and people who cooperated for data collection. Also, we would like to thank Dr. Naoki Ohsugi for offering the collaborative filtering tool.

REFERENCES

[1] H. Aman, N. Mochiduki, and H. Yamada, "A Model for Detecting Cost-Prone Classes Based on Mahalanobis-Taguchi Method," IEICE transactions on information and systems, vol.E89-D, no.4, pp.1347-1358, 2006.

[2] J. Hanley, and B. McNeil, "The meaning and use of the area under a receiver operating characteristic (ROC) curve," Radiology, no.143, pp.29-36, 1982.

[3] H. Hata, O. Mizuno, and T. Kikuno, "Application of Machine Learning Without Negative Examples to Fault-Prone Module Detection," Proc. Software Engineering Symposium 2009 (SES 2009), pp.133-138, 2009 (in Japanese).

[4] IPA/SEC Japan, Visualization of IT Project (lower process edition). p.211, Nikkei Business Publications, 2006.

[5] Y. Kamei, A. Monden, S. Matsumoto, T. Kakimoto, and K. Matsumoto, "The Effects of Over and Under Sampling on Fault-Prone Module Detection," Proc. International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), pp.196-204, 2007.

[6] T. Khoshgoftaar, and N. Seliya, "Comparative Assessment of Software Quality Classification Techniques: An Empirical Case Study," Empirical Software Engineering, vol.9, no.3, pp.229-257, 2004.

[7] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking Classification Models for Software Defect Prediction: A Proposed Framework and Novel Findings," IEEE Transactions on Software Engineering, vol.34, no.4, pp.485-496, 2008.

[8] T. Motomura, T. Kakimoto, M. Tsunoda, N. Ohsugi, A. Monden, and K. Matsumoto, "Prediction of Project Cost Overrun Based on Collaborative Filtering," Technical report of IEICE. SS, vol.105, no.229, pp.35-40, 2005.

[9] N. Nagappan, and T. Ball, "Static analysis tools as early indicators of pre-release defect density," Proc. International Conference on Software Engineering (ICSE 05). pp.580-586, 2005.

[10] J. Procaccino, J. Verner, S. Overmyer, and M. Darter, "Case study: factors for early prediction of software development success," Information and Software Technology, vol.44, no.1, pp.53-62, 2002.

[11] Project Management Institute, A Guide to the Project Management Body of Knowledge: (Pmbok Guide). p.459, Project Management Institute, 2008.

[12] R Development Core Team, R: A language and environment for statistical computing. R Foundation for Statistical Computing, 2009. http://www.R-project.org.

[13] M. Shepperd, and C. Schofield, "Estimating software project effort using analogies," IEEE Transactions on Software Engineering, vol.23, no.12, pp.736-743, 1997.

[14] G. Taguchi, R. Jugulum, The Mahalanobis-Taguchi Strategy: A Pattern Technology System. Wiley, 2002.

[15] Y. Takagi, O. Mizuno, and T. Kikuno, "An Empirical Approach to Characterizing Risky Software Projects Based on Logistic Regression Analysis," Empirical Software Engineering, vol.10, no.4, pp.495-515, 2005.

[16] J. Verner, W. Evanco, and N. Cerpa, "State of the practice: An exploratory analysis of schedule estimation and software project success prediction," Information and Software Technology, vol.49, no.2, pp.181-193, 2007.