Decision Support Systems 72 (2015) 11–23

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

Cost-sensitive and ensemble-based prediction model for outsourced software project risk prediction

Yong Hu [a,*], Bin Feng [b], Xizhu Mo [b], Xiangzhou Zhang [c], E.W.T. Ngai [d], Ming Fan [e], Mei Liu [f]

[a] Institute of Business Intelligence and Knowledge Discovery, Guangdong University of Foreign Studies, Sun Yat-sen University, Higher Education Mega Center, Guangzhou 510006, PR China
[b] School of Management, Guangdong University of Foreign Studies, Higher Education Mega Center, Guangzhou 510006, PR China
[c] School of Business, Sun Yat-sen University, No. 135, Xingang Xi Road, Guangzhou 510275, PR China
[d] Department of Management and Marketing, The Hong Kong Polytechnic University, Kowloon, Hong Kong, PR China
[e] Foster School of Business, University of Washington, Seattle, WA 98195-3226, USA
[f] Department of Internal Medicine, Division of Medical Informatics, University of Kansas Medical Center, Kansas City, KS 66160, USA

ARTICLE INFO

Article history:
Received 10 June 2014
Received in revised form 15 January 2015
Accepted 3 February 2015
Available online 11 February 2015

Keywords:
COSENS
Outsourced software project
Risk management
Ensemble
Cost-sensitive
Risk prediction

ABSTRACT

Nowadays software is mainly developed through outsourcing and it has become one of the most important business practice strategies for the software industry. However, outsourcing projects are often affiliated with high failure rate. Therefore to ensure success in outsourcing projects, past research has aimed to develop intelligent risk prediction models to evaluate the success rate and cost-effectiveness of software projects. In this study, we first summarized related work over the past 20 years and observed that all existing prediction models assume equal misclassification costs, neglecting actual situations in the management of software projects. In fact, overlooking project failure is far more serious than the misclassification of a success-prone project as a failure. Moreover, ensemble learning, a technique well-recognized to improve prediction performance in other fields, has not yet been comprehensively studied in software project risk prediction. This study aims to close the research gaps by exploring cost-sensitive analysis and classifier ensemble methods. Comparative analysis with T-test on 60 different risk prediction models using 327 outsourced software project samples suggests that the ideal model is a homogeneous ensemble model of decision trees (DT) based on bagging. Interestingly, DT underperformed Support Vector Machine (SVM) in accuracy (i.e., assuming equal misclassification cost), but outperformed in cost-sensitive analysis under the proposed framework. In conclusion, this study proposes the first cost-sensitive and ensemble-based hybrid modeling framework (COSENS) for software project risk prediction. In addition, it establishes a new rigorous evaluation standard for assessing software risk prediction models by considering misclassification costs.

1. Introduction

Software project development is typically risky [37]; the Standish Group reported that the success rate is only 37% [56]. Software failure is primarily attributed to management issues (65%) [19]. Furthermore, outsourced software projects are riskier than in-house ones because they involve multiple stakeholders and decision-makers (i.e., contractors and customers). Hence, there is a great need of intelligent risk prediction models that are effective in improving the evaluation of success probability [14,62].

Minimizing misclassification cost rather than misclassification errors is more important because varied errors have different costs [8]. For example, in direct marketing, it costs more to classify a willing customer as a reluctant one, because serving a willing customer is valuable, whereas serving a reluctant one wastes time and resources [38]. Similarly in the medical diagnosis of heart disease, misclassifying a patient as a sick one only loses the fixed cost spent on angiographic test, whereas misclassification of a sick patient as a healthy one means losing the chance to administer timely treatment and puts the life of the patient at greater risk [8]. Misclassification cost is one of the most important types of costs in a whole range of costs in cost-sensitive learning studies [8,36].

To date, all existing risk prediction models for software projects assume that misclassification costs are equal, which is inconsistent with the practice. If a failure-prone project is regarded as success-prone, the corresponding losses may be immeasurable and irreparable after considerable resources have been invested, which is undoubtedly costlier than misclassifying a success-prone project as a failure. Therefore, it is imperative to addresses the unequal misclassification cost issue. Actually, cost-sensitive models have been studied in the related field of software defect prediction [20,34,55]. Software defect prediction focuses on

* Corresponding author. Tel.: +1 973 596 6597; fax: +1 973 596 5777.
E-mail addresses: henryhu200211@163.com (Y. Hu), fengbin_ace@foxmail.com (B. Feng), moxizhu@foxmail.com (X. Mo), zhxzhou@mail2.sysu.edu.cn (X. Zhang), eric.ngai@polyu.edu.hk (E.W.T. Ngai), mfan@uw.edu (M. Fan), meiliu@kumc.edu (M. Liu).

http://dx.doi.org/10.1016/j.dss.2015.02.003
0167-9236/

programming errors, whereas software project risk prediction emphasizes on overall project outcome. To the best of our knowledge, a cost-sensitive model has not been comprehensively investigated for software project risk. Hence, our first research question is how to construct an effective cost-sensitive model to predict software project risk.

Our second research question is to determine an effective ensemble method for prediction model construction. Many researchers in other fields have applied classifier ensemble methods, which can generate predictions that are theoretically [46] and empirically [7,39] better than those of individual classifiers. However, all existing prediction models of software project risk are based on individual classifiers [1,48,61], and sufficient ensemble models have not been developed. Moreover, a consensus has not been reached on the ideal ensemble method (e.g. majority voting, bagging, boosting, and Borda count) [7,11]. Thus, we aim to examine: do these methods significantly affect the performance of a cost-sensitive prediction model for outsourced software project risk, and, if so, which is the most effective? (We compared three prevalent ensemble methods, namely majority voting, bagging and boosting.)

To investigate the two research questions above, we first explore the framework of the cost-sensitive and ensemble-based method (COSENS) for software project risk prediction, and various cost-sensitive classifier ensembles were constructed. Furthermore, we compare these ensembles with several individual classifiers using T-test to determine its statistical significance.

Remainder of this paper is structured as follows: in Section 2, we discuss related works on software project risk prediction, cost-sensitive learning, and ensemble methods. In Section 3, we describe the methodology, including the research design and the evaluation methods. In Section 4, we conduct experiments and analyze the results. In Section 5, we present the research implications, limitations, and future works.

2. Related works

2.1. Prediction of software project risks

In this section, we summarize the studies on the intelligent prediction of software project risks over the past 20 years. Existing studies primarily adopt Bayesian network (BN), random forest (RF), neural network (NN), logistic regression (LoR), fuzzy logic, genetic algorithm (GA), and DT. Table 1 compares representative studies in terms of their methods used and their contributions/findings.

Table 1 indicates that the basic objective of such research is to help software managers improve prediction accuracy and evaluate success rate. However, such models do not consider cost sensitivity. In addition, the abovementioned models are commonly based on individual classifiers, and, the prediction accuracy of models may improve with the ensemble method [46]. The current study combines cost-sensitive learning and ensemble methods to construct an appropriate prediction model of software project risk.

2.2. Cost-sensitive learning

To minimize misclassification costs, Elkan [5] proposed the cost-sensitive learning method. This method is widely applied in the field of software engineering for predicting software faults [63], software defects [20], and software quality [34].

Jiang and Cukic [63] applied cost-sensitive learning to predict software faults. This method can help project managers choose a suitable model that reveals explicit information on misclassification costs. Zheng [20] used three cost-sensitive boosting algorithms to boost NN for software defect prediction. The experiment results revealed that the algorithm based on threshold moving outperformed the two other algorithms based on weight updates. Seliya and Khoshgoftaar [34] introduced cost-sensitive learning into software quality prediction using two prevalent DT algorithms (C4.5 and RF). They comprehensively examined six cost-sensitive learning methods and concluded that random undersampling method is ideal.

However, existing studies regarding the prediction model of software project risks do not consider the cost-sensitive issue. Therefore, the introduction of the cost-sensitive learning method may guide project managers in this field substantially.

2.2.1. Cost matrix

We typically use a confusion matrix (shown in Table 2) for the binary classification problem (e.g. failure-prone and success-prone projects). The matrix is divided into four parts: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). For this problem, we define the failure-prone project as a positive sample. TP and TN represent the number of correct predictions for failure-prone and success-prone projects, respectively. FP represents the number of incorrect predictions that misclassify a success-prone project as failure-prone, whereas FN stands for the number of incorrect predictions that misclassify a failure-prone project as success-prone.

Table 1
Research on the prediction of software project risk.

Representative studies	Method	Contribution/finding
Hu et al. [62]	BN with causality constraints	The predictions of the BN model with causality constraints are more accurate than those of the LoR, DT, NB, and BN models.
Procaccino et al. [22]	BN	The predictions of the BN-based model are more accurate than the naïve model (71% to 40%).
Hu et al. [59]	Many-to-many actionable knowledge discovery based on RF	Both risk analysis and risk planning are considered in software project risk management. The RF model was more effective than the C4.5, Naïve Bayes (NB), and BN models.
Zhang et al. [15]	NN	The predictions of the NN model are more accurate than those of the traditional LoR model with respect to the construction of an effective early warning system for project escalation.
Verner et al. [24]	LoR	This study used LoR to predict project success of three groups of projects under different culture context. There is a significant difference in prediction accuracy among these groups due to cultural, schedule-related factors and technical skill levels.
Takagi et al. [66]	LoR	This model adopted LoR to predict confused projects and achieved a high prediction accuracy of 87.5%.
Siwani and Capretz [18]	Fuzzy logic	The proposed framework based on fuzzy logic can enhance estimation accuracy to increase the probability of project success.
Sharma and Banwari [9]	DT	The generated decision rules can effectively predict project outcomes for project managers. These managers can then minimize risks through the appropriate actions.
Reyes et al. [14]	GA, BN	The model based on the evolution of GA over Bayesian method indicated great superiority in success ratio prediction and cost evaluation.
Amasaki et al. [45]	Association rule	This study applied the association rule method to determine runaway projects. The rules generated by this method can generate relatively accurate predictions (75%).