Computer Standards & Interfaces 40 (2015) 15–22

Contents lists available at ScienceDirect

Computer Standards & Interfaces

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

Discriminating risky software project using neural networks

Wen-Ming Han *

Department of Information Management, Takming University of Science and Technology, 56, Sec. 1, Huanshan Rd., Taipei, Taiwan, ROC

ARTICLE INFO

Article history:
Received 1 February 2012
Received in revised form 17 November 2014
Accepted 27 January 2015
Available online 4 February 2015

Keywords:
Risks
Software project risk management
Risky project
Neural network

ABSTRACT

Early and accurate discrimination of risky software projects is critical to project success. Researchers have proposed many predictive approaches based on traditional modeling techniques, but the high misclassification rate of risky projects is common. To overcome this problem, this study proposes a typical three-layered neural network (NN) architecture with a back propagation algorithm that can learn the complex patterns of the OMRON dataset. This study uses four accuracy evaluation criteria and two performance charts to objectively quantify and visually illustrate the performance of the proposed approach. Experimental results indicate that the NN approach is useful for predicting whether a project is risky. Specifically, this approach improves accuracy and sensitivity by more than 12.5% and 33.3%, respectively, compared to a logistic regression model developed from the same database. These results imply that the proposed approach can be used for early planning of limited project/organization resources and appropriate action for risky projects that are likely to cause schedule slippage and cost overload.

Contents

* Tel.: +886 2 26585801x5263; fax: +886 2 26581726.
E-mail address: wmhan@takming.edu.tw.

http://dx.doi.org/10.1016/j.csi.2015.01.001
0920-5489/

1. Introduction

A recent industry survey has revealed that software projects can fail due to a variety of problems including cost overload, schedule slippage, requirement misunderstandings, and client dissatisfaction [1]. There is a certain risk involved for any worthwhile software project. This fact highlights the need for early identification of risky projects to enable the planning of essential risk management activities and resources during their implementation. Suitable planning of resources and actions can effectively increase the success rates of such software projects [2].

Research work on risk prediction for software projects involves two approaches: (1) predicting the overall degree of risk of a project and (2) predicting whether a project is risky. The former provides a quantifiable risk value so that project managers can discriminate between high, medium, or low risk software projects with respect to a given project risk level scale or by using clustering techniques [3–5]. The latter provides a meaningful sign (risky/not risky) as a classifier that a risk-prone project can be effectively identified early and thus aid the planning of risk management strategies [6,7]. The latter approach was considered in this study.

Numerous binary prediction approaches have been constructed by statistical techniques for classifying risk-prone projects in the literature. Examples include logistic regression [6], Bayesian classification [8], and the association rule [9]. Although the overall classification accuracy of these approaches is at an acceptable level, correctly identifying a risky project at a true-positive rate is still a challenge.

Effort investment without the premise that a risky project is correctly identified at the initial stage is ineffective [10]. For project managers, misjudging a risky project diminishes their alertness during implementation. Consequently, the failure rates of the project would increase without prior warning and thus, a great cost would be expended in controlling the crisis. In other words, the incorrect classification of not risky projects at the initial stage does not increase failure rates, even if extra effort and resources are invested.

While neural networks (NNs) are regarded as a useful and accurate model building technique, little research has been conducted using this method to investigate issues related to software project risk management. The two known applications we found focused on risk analysis [11] and risk control [12]. There are currently no studies on the use of NN for classifying projects as either risky or not risky.

In this study, an NN model with a back propagation algorithm is developed to predict how risk-prone a software project is. The remainder of this paper is organized as follows. Section 2 reviews some approaches related to project risk prediction. In Section 3, we introduce the OMRON dataset and briefly explain the basic concepts of NN. The steps for constructing our prediction approach are presented in Section 4. In Section 5, the experimental results are presented and then compared to results without the prediction model. A comparison with previous work using logistic regression is shown in Section 6. The external valuations produced are shown in Section 7. Conclusions are presented in Section 8. Finally, limitations and directions for future research are presented in Section 9.

2. Related work

The International Organization for Standardization (ISO) has published ISO/IEC 16085 [13], addressing necessary processes to continuously manage risks throughout the life cycle of a product or service. This standard can be applied by any organization to manage risk as uncertainty so that meaningful actions can be taken that reduce or eliminate the probability and/or impact of these risks. ISO/IEC 16085 suggests that a complete risk management process must consist of seven essential activities: “plan and implement risk management;” “manage the project risk profile;” “perform risk analysis;” “perform risk monitoring;” “perform risk treatment;” “evaluate the risk management process;” and “technical and management processes.” Fig. 1 shows the structural process of this standard. Beginning with an analysis of Fig. 1, it can be seen that the “perform risk analysis” activity is critical because the risk-proneness of a project should be captured to make recommendations for treatment at the initial stage.

Understanding the risk-proneness of a project is critical when trying to determine whether to invest more effort and limited project/organization resources into a new project. Hence many predictive models have been proposed to assist project managers in managing risky projects in the early stages. For example, Karolak [14] used a Bayesian probability tree approach to develop a software engineering risk management (SERIM) method for predicting the risk of software projects. The method proposes a well-defined structure with 81 risk questions, but no empirical studies or data have been used to examine or improve its performance until now.

Tiwana et al. [4] investigated the importance of six risk factors for software projects. Subsequently, they built a multiple linear regression model to yield an overall risk score which was then converted into five given project risk levels (high, moderately high, medium, moderately low, and low). Although the foundation of the model was based on a large dataset including 720 projects, the predictive performance of the model was not provided and the external validity of extra new projects was not performed.

Mizuno et al. [8] used a Bayesian classification approach to predict the risk proneness of a software project based on 40 historical projects in the OMRON company. The 10-fold cross-validation results indicated that seven projects were not predicted correctly, corresponding to 17.5% inaccuracy. Moreover, two of the seven misclassified projects were risky projects.

Takagi et al. [6] used the same dataset as Mizuno et al. to classify risky projects by logistic regression. The predictive model was constructed based on 32 projects from 1996 to 1997 and was then validated by 8 projects in 1998. Only one validation project was classified incorrectly. Unfortunately, this misclassified project was a risky project.

Amasaki et al. [9] also applied the OMRON dataset to build a predictive model. Eleven rules were extracted by the association rule, and their minimum confidence was between 0.63 and 0.91. They achieved an overall accuracy of 75% based on 12 private projects from 2003 to 2004. However, three risky projects were still classified incorrectly as not risky projects.

Three known models were developed from the OMRON dataset to build predictive models for the risky nature of a project. Unfortunately, these approaches all have the same issue that some risky projects are regarded as not risky. Therefore, there is considerable interest in using NN for the same purpose.

3. Material and methods

3.1. OMRON dataset

The OMRON database was used for this research. The dataset was collected by the Software Engineering Process Group (SEPG) of the Social Systems Solutions Business Company (SSBC) for the OMRON Corporation, to better predict risky projects (confusion-prone projects). This dataset comprises 40 valid projects completed in the past and 24 attributes (22 project risk factors and 2 project background features). These 22 project risk factors comprise five viewpoints: (1) requirements (Reqm), (2) estimations (Estm), (3) planning (Plan), (4) team organization (Team), and (5) project management activities (Prma).

A risky project is defined as exceeding the budget and lacking control during project implementation; it is represented by the