Chapter X

A Statistical Framework for the Prediction of Fault-Proneness

Yan Ma, West Virginia University, USA
Lan Guo, West Virginia University, USA
Bojan Cukic, West Virginia University, USA

Abstract

Accurate prediction of fault-prone modules in software development process enables effective discovery and identification of the defects. Such prediction models are especially valuable for the large-scale systems, where verification experts need to focus their attention and resources to problem areas in the system under development. This chapter presents a methodology for predicting fault-prone modules using a modified random forests algorithm. Random forests improve classification accuracy by growing an ensemble of trees and letting them vote on the classification decision. We applied the methodology to five NASA public domain defect datasets. These datasets vary in size, but all typically contain a small number of defect samples. If overall accuracy maximization is the goal, then learning from such data usually results in a biased classifier. To obtain better prediction of fault-proneness, two strategies are investigated: proper sampling technique in constructing the tree classifiers, and threshold adjustment in determining the “winning” class. Both are found to be effective in accurate prediction of fault-prone modules. In addition, the chapter presents a
Introduction

Early detection of fault-prone software components enables verification experts to concentrate their time and resources on the problem areas of the software system under development. The ability of software quality models to accurately identify critical faulty components allows for the application of focused verification activities ranging from manual inspection to automated formal analysis methods. Software quality models, thus, help ensure the reliability of the delivered products. It has become imperative to develop and apply good software quality models early in the software development life cycle, especially for large-scale development efforts.

The basic hypothesis of software quality prediction is that a module currently under development is fault prone if a module with the similar product or process metrics in an earlier project (or release) developed in the same environment was fault prone (Khoshgoftaar, Allen, Ross, Munikoti, Goel, & Nandi, 1997). Therefore, the information available early within the current project or from the previous project can be used in making predictions. This methodology is very useful for the large-scale projects or projects with multiple releases.

Accurate prediction of fault-prone modules enables the verification and validation activities focused on the critical software components. Therefore, software developers have a keen interest in software quality models. To meet the needs of developers, fault-prone prediction models must be efficient and accurate.

Many modeling techniques have been developed and applied for software quality prediction. These include: logistic regression (Basili, Briand, & Melo, 1996); discriminant analysis (Khoshgoftaar, Allen, Kalaichelvan, & Goel, 1996; Munson & Khoshgoftaar, 1992); the discriminative power techniques (Schneidewind, 1992); optimized set reduction (Briand, Basili, & Hetmanski, 1993); artificial neural network (Khoshgoftaar & Lanning, 1995); fuzzy classification (Ebert, 1996); genetic algorithms (Azar, Precup, Bouktif, Kegl, & Sahraoui, 2002); and classification trees (Gokhale & Lyu, 1997; Khoshgoftaar & Seliya, 2002; Selby & Porter, 1988; Troster & Tian, 1995). Following more than a decade of research, Fenton and Neil (1999) critically evaluated software quality models. They proposed the Bayesian belief networks (BBNs) as the most promising method for further consideration. In the Bayesian networks, nodes (variables) are interrelated by links representing causal relationships. The BBNs predict an event (here an occurrence of a fault-prone module) based on an uncertainty measure computed by the Bayesian inference methods. The ability of the BBNs in modeling complex inter-relationship among the variables cannot be achieved by the conventional methods such as multi-variate analysis. The introduction of the BBNs to software quality prediction was certainly a positive step forward. However, the drawbacks of the BBNs have also been recognized by the researchers. First, the BBNs connect variables based on causal relationship. However, not all variables are causally related. Software quality predictors are generally software metrics, such as McCabe metrics, Halstead metrics, or...
Machine Learning for Biometrics
www.igi-global.com/chapter/machine-learning-biometrics/37003?camid=4v1a