ISSN: 2424-8460(O)
2251-2608(P)

Original Research Article

Research on feature selection techniques in software defect prediction

Zijie Zhou

Northeastern University, Boston, MA, 02108, USA

Abstract: This paper aims to study the application of feature selection technology in software defect prediction, explore the influence of different feature selection methods on the performance of prediction model, and then improve the accuracy and efficiency of software defect prediction. Three common feature selection methods were used: information gain, chi-square test and recursive feature elimination (RFE), and experimental validation was conducted using support Vector Machine (SVM), decision tree, random forest and logistic regression. The experimental results show that the combination of random forest with recursive feature elimination (RFE) method shows optimal performance on multiple indicators, with accuracy of 0.89, precision of 0.84, recall of 0.91, F1 value of 0.87 and AUC value of 0.93, which are better than the other combinations. The junction feature selection can significantly improve the accuracy and efficiency of the model in the software defect prediction. Through the appropriate feature selection methods, it can effectively reduce the redundant features, reduce the model complexity, and improve the prediction accuracy.

Keywords: Software defect prediction; Feature selection; Information gain; And chi-square test

1. Introduction

With the increasing scale and complexity of software systems, the prediction and management of software defects has become an important research topic in the field of software engineering. The existence of defects not only affects the reliability and stability of the software, but also increases the maintenance cost in the later stage. How to timely identify the potential defects in the development stage, and make effective prediction and management, has become one of the key means to improve the software quality.

2. Theoretical basis of software defect prediction

2.1. Basic concept of software defect prediction

Software defect prediction is to predict the area of defect in the software by analyzing the defect data in historical projects and combining the characteristics of the current version of the software[1]. Software defect prediction model is based on machine learning algorithm, using various characteristics of software modules (such as code complexity, developer experience, inter-module dependence, etc.) as input to predict the occurrence probability of defects. In the process of defect prediction, the construction of data set is crucial, and it is necessary to collect various data of software development, including the defect history of each software module, code characteristics, participation of developers and test results of software.

In machine learning, defect prediction belongs to the classification problem, and the goal is to classify each module or file as “defect” or “no defect”. The performance of the model is influenced by the dataset quality, feature selection and machine learning algorithms, and the feature selection process is crucial to improve the prediction accuracy[2]。

2.2. Application of common machine learning algorithms in defect prediction

The application of commonly used machine learning algorithms in software defect prediction includes decision tree, support vector machine (SVM), random forest, logistic regression, neural network, etc. These algorithms train the data sets, learn the hidden patterns in the data, and are used to predict the software defects[3]。

① The decision tree algorithm selects features recursively by building tree models to realize the classification of defect modules. Decision trees have the advantage of strong interpretability and can clearly show which features have a greater impact on defect prediction.

② Support vector machine (SVM) is a powerful classification model, especially suitable for small sample, nonlinear, and high-dimensional data sets. In software defect prediction, SVM is able to find an optimal decision boundary in the feature space to distinguish defect modules from non-defect modules.

③ Random forest is a method in ensemble learning, which builds multiple decision trees to predict. Due to its strong robustness, random forests are often used to solve the problems caused by feature redundancy and high dimensionality.

④ Logistic regression predicts the occurrence probability of module defects by constructing a probability model. Often used as a baseline model, it can deal with linear features and binary classification problems.

⑤ Deep learning algorithms show powerful performance in processing large-scale data. The application of neural networks in defect prediction mainly focuses on the scenarios that can extract complex feature patterns, especially when large amounts of data are capable of providing superior prediction ability.

2.3. Theory and significance of feature selection

Feature selection is a process of screening out the most important subset of features from the data, which aims to reduce redundant and irrelevant features in the data set, and thus improve the training efficiency, prediction accuracy and interpretability of the model[4]. The significance of feature selection is particularly important in software defect prediction, since the software development process involves a large number of features, and not all features contribute significantly to the defect prediction. The efficient feature selection can significantly reduce the model complexity, reduce the computational cost, and improve the reliability of the prediction results.

3. Feature selection method

3.1. Information gain method

Information gain (Information Gain, IG) is an entropy-based (Entropy) feature selection method by evaluating its contribution to the classification of datasets. In the software defect prediction, the information gain can measure the importance of each feature for the defect prediction, and select the features with a large information gain[5]. The core idea of information gain is to select features that can minimize the uncertainty of the dataset.

The information gain is calculated by the following formula:

$$IG(A) = H(D) - \sum_{v \in A} \frac{|D_v|}{|D|} H(D_v)$$

A is the feature, v is the possible value of the feature, D is the dataset, D_v is the subset of the feature under

the value.

$H(D)$ represents the entropy of the dataset D, defined as:

$$H(D) = -\sum_{i=1}^{k} p_i \log_2 p_i$$

p_i is the probability of category D in the dataset i, and k is the number of categories

The information gain method compares the information gain values of different features and selects the features with the largest information gain for further training and prediction.

3.2. Chi-square test method

The Chi-square test (Chi-square Test) is a statistical-based method for assessing the independence between a feature and the target variable. Through the chi-square test, it can quantify the strength of the relationship between each feature and the defect prediction target, and select the features with a large correlation degree with the target variable. In software defect prediction, the chi-square test is often used to screen for defect-related features to improve the predictive power of the model.

The chi-square value is calculated as follows:

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

O_i is the observed frequency, E_i is the expected frequency, and n is the number of feature categories.

The expected frequency is calculated using the following formula:

$$E_i = \frac{(R_i \times C_i)}{N}$$

R_i is the observed frequency, C_i is the expected frequency, N is the number of feature categories.

The chi-square test determines the significance of each feature by comparing the difference between observed and expected frequencies, and then screened for features that would significantly contribute to the defect prediction.

3.3. Recursive feature elimination (RFE)

Recursive feature elimination (RFE) is a model-based feature selection method that eliminates the poor features recursively until the optimal feature subset is obtained. RFE improves the model performance by assessing the importance of each feature, retaining the features that contribute the most to the model and reducing the impact of irrelevant features on the model.

The basic process of RFE is as follows: ① trains a preliminary model and calculates the importance of all features (usually feature-based weights or other performance measures); ② removes the least important features; ③ retrains the model on the remaining features and repeats the process until the stop condition is met (e. g., the predetermined number of features).

The iterative process of the RFE can be described by the following formula:

$$F_{k+1} = F_k - \text{Remove}\left(F_{\text{least important}}\right)$$

F_k is the feature subset in the k iteration, $F_{least\ important}$ is the least important feature in the current model. The iterative process continues until the size of the feature subset reaches the predetermined target.

RFE method by gradually removing redundant features to obtain a streamlined and excellent feature set,

which can effectively improve the effect of the defect prediction model.

4. Simulation and simulation experiment

4.1. Selection and processing of experimental data sets

Multiple publicly available software defect datasets were used in the experiment containing various trait data collected during development by different software projects.

When selecting the experimental data set, the following several factors are considered. ① Scale of data set, select multiple data sets, covering projects of different sizes, from small open source projects to large enterprise applications; the diversity of ② features, each data set contains multiple features, which can represent all aspects of the software development process, including the code structure, the complexity of modules, and the frequency of changes in the development process. ③ Accuracy of defect annotation, and the verified data set with clear defect annotation is selected to ensure the reliability of the experimental results.

After the following preprocessing steps, the ① data is cleaned and deleted to ensure the quality of data; ② data normalize the numerical features to eliminate the influence of different feature dimensions; split ③ data, the data set is used at 70%, and the remaining 30% is used for testing to ensure the independence of training and test data.

The main data sets used in the experiment include: ① NASA MDP Data set: the data set contains code features and defect records of multiple open source software projects, which is the benchmark data set commonly used in the field of software defect prediction; ② Kaggle software defect data set: contains code features and defect annotation of multiple projects, which is suitable for the study of classification problems.

4.2. The process of feature selection and model training

① Data preprocessing, the selected datasets were cleaned, normalized, and divided into training and test sets. The pre-processing of the dataset ensures that each feature is on the same dimension, avoiding the effects of scale differences between features.

② Feature selection. For each data set, the information gain method, chi-square test method and recursive feature elimination (RFE) are applied to feature selection.

③ Model training: train multiple machine learning models on the subset after feature selection, including Support Vector Machine (SVM), decision tree, random forest, logistic regression, etc.; each model is trained with selected features and uses cross-validation to adjust the model hyperparameters to optimize the model performance.

④ Performance evaluation, evaluating the predictive performance of each model on the test set, and calculating and comparing the evaluation metrics under different feature selection methods. The main evaluation indicators used include accuracy (Accuracy), precision (Precision), recall (Recall), F1 values, and AUC values.

During the process of model training, the mathematical formulas of the training and test of the feature subsets obtained by different feature selection methods can be expressed as:

$$\hat{y} = f(X_s, \theta)$$

$\hat{y}$ is the output predicted by the model; X_s classification label for defects or no defects); f is a subset of selected features generated based on feature selection methods (information gain, chi square test, or RFE) It is a classification model that represents models such as support vector machines, decision trees, random forests, etc. θ

is a parameter of the model, including hyperparameters and weights learned during the training process.

The indicators used for the performance evaluation are as follows:

① precision (Accuracy)

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

② Accuracy (Precision)

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

③ recall (Recall)

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

④ F1 value (F1-Score)

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

⑤ AUC value (Area Under Curve): The AUC value represents the classification performance of the model at different thresholds, which is mathematically expressed as:

$$AUC = \int_0^1 \text{TPR}(t)\text{dFPR}(t)$$

The TPR indicates the recall rate, The FPR indicates the false - positive case rate.

5. Discussion

5.1. Improvement effect of feature selection on model performance

Feature selection is a key link to improve the performance of the software defect prediction model. Feature selection techniques such as information gain method, Chi-square test method and recursive feature elimination (RFE) can effectively screen out the features closely related to defect prediction and avoid the interference of irrelevant features to the model.

5.2. Suitability analysis of feature selection on different datasets

Different datasets differ in the applicability of feature selection. the information gain method performs well on the data set with large data volume and independent features, and can effectively quantify the contribution of features to defect prediction, while the chi-square test method is more effective when there is obvious statistical relationship between features and defect category. Recursive feature elimination (RFE) is especially suitable for high-dimensional data sets, which can improve the robustness and prediction accuracy of the model by gradually eliminating unimportant features. The advantages of RFE are particularly evident with the small or high noise of the data set, and the selection of an appropriate feature selection method should be adjusted according to the size and feature distribution of the dataset.

5.3. Relationship between feature selection and the prediction efficiency of software defects

Feature selection not only affects model performance, but also has important effects on prediction efficiency. By reducing the number of features, feature selection is able to reduce the computational complexity and accelerate the training and prediction process of the model. In practical applications, especially in the defect

prediction of large-scale software projects, the high efficiency of feature selection can effectively shorten the training time and improve the ability of real-time prediction.

6. Conclusion

Feature selection technology can not only improve the accuracy of software defect prediction, but also optimize the computational efficiency. Future studies can further combine different feature selection methods and machine learning algorithms to explore more universal and efficient defect prediction models, so as to provide a more scientific solution for software quality assurance.

References

[1] Zhang Jian, Jiang Hong. Research on unbalanced data classification algorithm in software defect prediction [J]. Information Technology, 2024, (12): 149-158 + 166.

[2] Wang Yue, Li Yong, Zhang Wenjing. Active learning method for software defect prediction [J]. Modern Electronic Technology, 2024,47 (20): 101-108.

[3] Tang Yu, Dai Qi, Yang Zhiwei, et al. Research on software defect prediction algorithm based on optimized random forest [J]. Computer Engineering and Science, 2023,45 (05): 830-839.

[4] Li Huilai, Yang Bin, Yu Xiuli, et al. Comparison of the interpretability of the software defect prediction model [J]. Computer Science, 2023,50 (05): 21-30.

[5] Cheng Hui. Research on software defect prediction techniques based on feature selection and ensemble learning [D]. Hangzhou Dianzi University, 2017.