Advances in Intelligent Systems Research, volume 146
2nd International Conference on Artificial Intelligence: Technologies and Applications (ICAITA 2018)

# Software Defect Prediction Based on Data Sampling and Multivariate Filter Feature Selection

Yating Lin[1] and Yiwen Zhong[2,*]
[1]College of Computer and Information Science, Fujian Agriculture and Forestry University, 350002, Fuzhou, China
[2]Key Laboratory of Smart Agriculture and Forestry Fujian Agriculture and Forestry University, Fuzhou Province University, 350002, Fuzhou, China
[*]Corresponding author

***Abstract*—In order to solve the useless feature and class imbalance problem in software defect prediction(SDP), this paper proposes a new prediction method which is based on data sampling and multivariate filter feature selection. Firstly, the sampling method re-samples the data set to achieve the data balance. Secondly, the multivariate filter algorithm selects feature and eliminates useless features such as irrelevant features and redundant features. Experimental results show that the proposed algorithm not only can effectively improve the prediction accuracy of the minority classes, but also effectively improve the overall classification performance of SDP.**

***Keywords—software defect prediction; data sampling; multivariate Filter algorithm***

## I. INTRODUCTION

With the increasing demand for software, the scale and complexity of software is getting higher and higher, and its development technology and operating environment are becoming more and more diverse. Therefore, in order to meet the software requirements, the requirements for the quality and reliability of software are also increasing. Especially in industries such as automatic control of special equipment, national military security, aerospace and other industries, a subtle hidden danger of software products may cause serious loss of life and property. Therefore, it is a key problem to improve the software quality and software reliability by accurately finding the modules with the fault-prone in software during the effective time.

The main purpose of Software Defect Prediction (SDP) is to predict whether there is a module with the fault-prone in a software product. It predicts whether a new module has a fault-prone based on the software metrics and historical defect information. These defect information can not only be used to guide the allocation of resources and repair software defects timely, but also can save the cost of software development and improve the quality of software [1-3]. There are many kinds of software metrics used for defect prediction. However, there are a large number of redundant features and irrelevant features in these features (software metrics) [4-5]. If these useless features are all deleted, the prediction model can reduce the computation time, memory consumption and improve the prediction accuracy of the model. Feature Selection (FS) [6-11] aims to select a few effective features to build an efficient prediction model. By eliminating irrelevant, redundant, or noisy features, the efficiency of the prediction model is accelerated and the accuracy of the prediction model is improved. However, the traditional FS algorithm assumes that the data set is balanced. In actual situations, SDP is an imbalance problem. In the normal software development process, the software modules with the fault-prone tend to be much less than the software modules with not-fault-prone. As a classification result of the unbalanced SDP, there may be software modules that will mistakenly predict software modules with the fault-prone as not-fault-prone. Finally, software failures and system failures in the actual operation process may cause incalculable losses.

In order to eliminate these useless features and solve the class imbalance problem, this paper proposes a data preprocessing technique that combines data sampling with FS. For example, literature [12] proposed a new method Im-IG (imbalanced-information gain) for the imbalanced problem in feature selection. Im-IG increased the weight of minority class in the information entropy calculation, in order to select features which were better for minority class. Im-IG improves the accuracy of the minority classes and solves effectively inadaptability of the IG algorithm in the imbalance problem. In literature [13], the K-means algorithm and the sampling mechanism are combined with the ReliefF and Relief algorithms to effectively solve the classification problem of imbalanced data sets. In literature [14], an algorithm based on immune clone feature selection and under-sampling ensemble was proposed to effectively solve the problem of curse of dimensionality and imbalanced classification in the process of Web spam detection. Literature [15] proposed an algorithm combining genetic algorithm and SMOTE. Firstly, features were selected by genetic algorithm. Secondly, SMOTE adds new artificial minority examples. These methods combine data sampling with FS, but there are also problems such as low prediction accuracy and low efficiency.

This paper proposes a feature selection algorithm based on data sampling and multivariate filter. Firstly, the sampling method re-samples the data set to achieve the data balance. Secondly, the multivariate filter algorithm selects feature and eliminates irrelevant features and redundant features. This paper use the multivariate filter algorithm, which not only considers the correlation between features and classes, but also considers the redundancy between features and features. It fully explains the purpose of feature selection—eliminating irrelevant features and redundant features to achieve optimal subset. In this study, we first introduced two kinds of

multivariate Filter algorithms: Correlation-based Feature Selection (CFS) and Fast Correlation-Based Filter (FCBF), and two data sampling methods: Synthetic Minority Over-sampling Technique (SMOTE), and Edited Nearest Neighbor(ENN). Finally we research the effect of combing the multivariate filter algorithm and the sampling method on the classification results.

## II. RELATED WORK

### A. The Multivariate Filter Algorithm

Both CFS and FCBF are based on the correlation to deal with the relationship between features and classes, features and features. Symmetrical uncertainty (SU) is often used to determine the degree of uncertainty between features and classes, features and features. The SU between the *i*-th feature $f_i$ and class label $C$ as:

$$SU(f_i, C) = 2\frac{IG(f_i, C)}{H(f_i) + H(C)} \tag{1}$$

where IG($f_i$,$C$) is the information gain between the *i*-th feature $f_i$ and class label $C$

$$IG(f_i, C) = H(f_i) - H(fi \mid C) \tag{2}$$

where H($f_i$) is the entropy of $f_i$ and H($f_i$ |$C$) is the entropy of $f_i$ after observing $C$:

$$H(f_i) = -\sum_j \rho(x_j)\log_2(\rho(x_j)) \tag{3}$$

$$H(f_i \mid C) = -\sum_k \rho(c_k)\sum_j \rho(x_j \mid c_k)\log_2(\rho(x_j \mid c_k)) \tag{4}$$

CFS[16] is a simple filter algorithm that ranks feature subsets based on correlation-based evaluation functions. The criterion of the evaluation function is to contain subsets of features that are highly correlated with the class and uncorrelated with each other which can be calculated using the following formula:

$$M_s = \frac{k\overline{r_{cf}}}{k + k(k-1)\overline{r_{ff}}} \tag{5}$$

$$r_{cf} = (1/k) * sum(SU(f_i, C)) \tag{6}$$

$$r_{cf} = (1/(k*(k-1))) * sum(SU(f_i, f_j)) \tag{7}$$

where $M_S$ is the merit of the current subset of features, $k$ is the number of features, $\overline{rcf}$ is the mean of the correlations between each feature and the class variable, and $\overline{rff}$ is the mean of the pairwise correlations between every two features.

FCBF is one of the classical methods for dealing with feature selection of discrete data sets[17]. The algorithm uses SU to measure the correlation between two features. The basic principle is described as follows: A threshold is set. If the correlation between the feature and the class is too low (below the threshold), the feature is removed as an irrelevant feature. If the correlation between two features is too large and exceeds the correlation between the two features and the class, there is redundancy between the two features, and the features with low correlation with the class are deleted.

### B. Data Sampling Method

The data sampling method is one of the important ways to deal with the problem of imbalanced data classification. Its implementation methods are mainly divided into two categories: under-sampling for samples from majority classes and over-sampling for samples from minority classes. The main idea is to reduce the impact of data imbalance on the classifier by reducing or adding some samples to achieve the purpose of data balance.

SMOTE refers to the establishment of artificial data using the similarities between the existing samples from minority classes in the feature space to achieve the purpose of data balance. ENN refers to the deletion of samples from majority classes that differ from the sample categories of two or more of the three nearest neighbors of those categories to achieve the purpose of data balance.

## III. ALGORITHM DESCRIPTION

The algorithm for combining the sampling methods (SMOTE and ENN) with the CFS algorithm is described in Table I. The algorithm for combining the sampling methods (SMOTE and ENN) with the FCBF algorithm is described in Table II.

TABLE I. CFS ALGORITHM COMBINED WITH SAMPLING METHOD

| |
|---|
| Input : Training data set S = {($F$ , $C$)}, $F$ = {($f_1$ , $f_2$ , $f_3$ , ... , $f_n$)} , Number of samples $m$ , $F_t$ = [ ] |
| Output : $F_{best}$ |
| 1. Use the SMOTE or ENN to convert an imbalanced data set into balanced data set |
| 2. For $i$ = 1 to $n$ do |
| 3. Calculated $M_s(f_i)$ according to formula (5) |
| 4. $f$ = argmax($M_s(f_i)$) |
| 5. $F_t$.append($f$) |
| 6. End for |
| 7. For $j$=1 to $n$ do |
| 8. If $f_j$ not in $F_t$ |
| 9. $f$ = argmax($M_s(F_t + f_j)$) |
| 10. $F_t$.append($f$) |
| 11. End for |
| 12. Repeat 7 |
| 13. $F_{best}$ = $F_t$ |

TABLE II. FCBF ALGORITHM COMBINED WITH SAMPLING METHOD

Input : Training data set S = {($F$ , $C$ )}, $F$ = {($f_1$ , $f_2$ , $f_3$ , ... , $f_n$)} ,
Number of samples $m$ , Threshold $\delta$ , $F_t$ = [ ]
Output : $F_{best}$

1. Use the SMOTE or ENN to convert an imbalanced data set into balanced data set
2. For $i$ = 1 to $n$ do
3. Calculated $SU(f_i, C)$ according to formula (1)
4. If $SU(f_i, C) \geq \delta$
5. $F_t$.append($f$)
6. End for
7. Sorting features in $F_t$ in descending order
8. For $j$ = 1 to $n$ do
9. For $k$ = $j$ +1 to $n$ do
10. Calculated $SU(f_j , f_k)$
11. If $SU(f_j, f_k) \geq SU(f_k, C)$
12. $F_t$.remove($f_k$)
13. End for
14. End for
15. $F_{best}$ = $F_t$

## IV. Analysis of Experimental Results

### A. Experiment Settings

In the experiment, we selected three data sets from NASA: CM1, PC3, KC3, as shown in Table III. Three learners are used: Naive Bayes (NB), MultiLayer Perceptron (MLP), support Vector Machines (SVM). Using AUC as a performance evaluation criterion, the greater the AUC, the better the classification performance. These experimental results takes the average of 5 runs of 5-fold CV. The results (in terms of AUC) of CFS and FCBF algorithm, each used along with two sampling methods, are reported in Table IV, in Table V, Table VI and Table VII. Figure I, Figure II, Figure III and Figure IV show the average of CFS and FCBF algorithm, each used along with two sampling methods.

TABLE III. NASA DATA SET

| Data Set | Language | System | # Modules | % Defective |
|---|---|---|---|---|
| CM1 | C | Spacecraft instrument | 327 | 12.84 |
| PC3 | C | Flight software for each orbiting satellite | 1077 | 12.44 |
| KC3 | Java | Collect and process satellite data | 194 | 18.56 |

TABLE IV. PERFORMANCE OF COMBING CFS AND SMOTE

| Data | Learner | Original data | SMOTE | CFS | SMOTE _CFS |
|---|---|---|---|---|---|
| CM1 | NB | 0.6762 | **0.6800** | 0.6682 | 0.6610 |
| | MLP | 0.3581 | 0.3581 | 0.3230 | **0.6491** |
| | SVM | 0.4572 | 0.5161 | 0.4859 | **0.6528** |
| PC3 | NB | 0.7341 | 0.7289 | 0.7253 | **0.7593** |
| | MLP | 0.3174 | 0.3073 | 0.3954 | **0.7433** |
| | SVM | 0.6297 | 0.6427 | 0.5716 | **0.7787** |
| KC3 | NB | 0.5999 | 0.5992 | **0.6528** | 0.6479 |
| | MLP | 0.4569 | 0.4569 | 0.4321 | **0.4941** |
| | SVM | 0.4931 | 0.5023 | 0.4860 | **0.5035** |

TABLE V. PERFORMANCE OF COMBING FCBF AND SMOTE

| Data | Learner | Original data | SMOTE | FCBF | SMOTE _FCBF |
|---|---|---|---|---|---|
| CM1 | NB | 0.6762 | 0.6800 | 0.6477 | **0.6792** |
| | MLP | 0.3581 | 0.3581 | 0.6309 | **0.6828** |
| | SVM | 0.4572 | 0.5161 | 0.4731 | **0.6364** |
| PC3 | NB | 0.7341 | 0.7289 | **0.7425** | **0.7425** |
| | MLP | 0.3174 | 0.3073 | **0.7425** | 0.7359 |
| | SVM | 0.6297 | 0.6427 | 0.5575 | **0.6875** |
| KC3 | NB | 0.5999 | 0.5992 | 0.5778 | **0.6083** |
| | MLP | 0.4569 | 0.4569 | 0.5854 | **0.6173** |
| | SVM | 0.4931 | 0.5023 | 0.5163 | **0.6210** |

TABLE VI. PERFORMANCE OF COMBING CFS AND ENN

| Data | Learner | Original data | ENN | CFS | ENN_CFS |
|---|---|---|---|---|---|
| CM1 | NB | 0.6762 | 0.6680 | 0.6682 | **0.6753** |
| | MLP | 0.3581 | 0.3581 | 0.3230 | **0.3598** |
| | SVM | 0.4572 | 0.4582 | 0.4859 | **0.4894** |
| PC3 | NB | 0.7341 | 0.7304 | 0.7253 | **0.7446** |
| | MLP | 0.3174 | 0.3174 | 0.3954 | **0.5416** |
| | SVM | 0.6297 | 0.6321 | 0.5716 | **0.6913** |
| KC3 | NB | 0.5999 | 0.5676 | 0.6528 | **0.6541** |
| | MLP | 0.4569 | 0.4569 | 0.4321 | **0.4902** |
| | SVM | 0.4931 | 0.4943 | 0.4860 | **0.5070** |

TABLE VII. PERFORMANCE OF COMBING FCBF AND ENN

| Data | Learner | Original data | ENN | FCBF | ENN_FCBF |
|---|---|---|---|---|---|
| CM1 | NB | 0.6762 | **0.6680** | 0.6477 | 0.6451 |
| | MLP | 0.3581 | 0.3581 | 0.6309 | **0.6438** |
| | SVM | 0.4572 | 0.4582 | 0.4731 | **0.4656** |
| PC3 | NB | 0.7341 | 0.7304 | **0.7425** | **0.7425** |
| | MLP | 0.3174 | 0.3174 | **0.7425** | **0.7425** |
| | SVM | 0.6297 | **0.6321** | 0.5575 | 0.6181 |
| KC3 | NB | 0.5999 | 0.5676 | 0.5778 | **0.6134** |
| | MLP | 0.4569 | 0.4569 | 0.5854 | **0.6420** |
| | SVM | 0.4931 | 0.4943 | 0.5163 | **0.5320** |

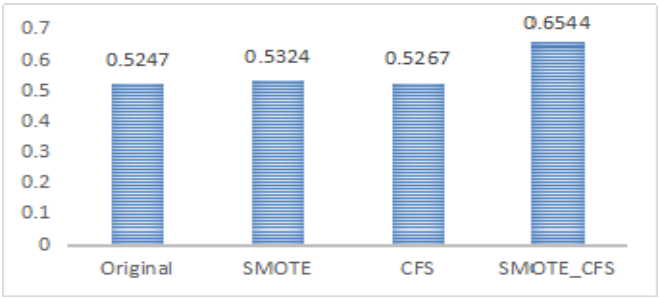


FIGURE I. PERFORMANCE OF COMBING CFS AND SMOTE

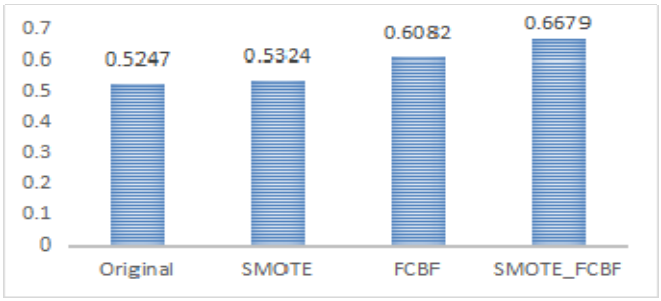


FIGURE II. PERFORMANCE OF COMBING FCBF AND SMOTE

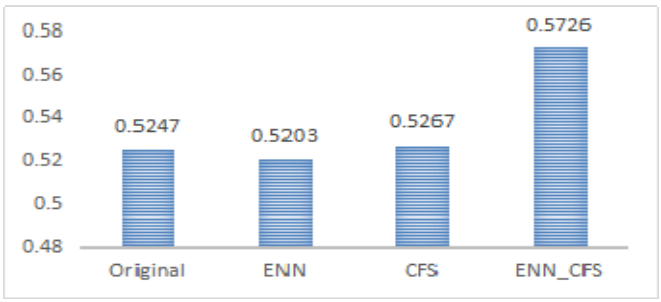


FIGURE III. PERFORMANCE OF COMBING CFS AND ENN

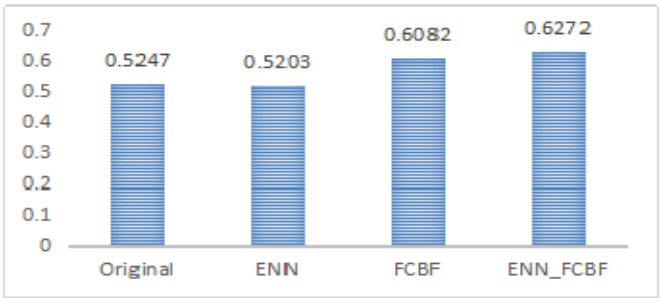


FIGURE IV. PERFORMANCE OF COMBING FCBF AND ENN

### *B. Experimental Comparison and Analysis*

From the tables and figures, when CFS was used, SMOTE _CFS displayed significantly better performance than SMOTE and CFS in most cases. Similarly, in many cases, ENN_CFS displayed significantly better performance than ENN and CFS in most cases. When FCBF was used, SMOTE_FCBF demonstrated better or similar performance than SMOTE and FCBF in most cases. Similarly, ENN_FCBF did too. This shows that the combination of the sampling method and the multivariate filter algorithm improves the re-sampling data set classification performance and classifier recognition rate for the minority classes.

## V. CONCLUSION

In order to solve the useless feature and class imbalance of SDP, this paper proposes a multivariate filter algorithm based on data sampling. The experimental results show that the proposed algorithm not only can effectively improve the prediction accuracy of the minority classes, but also improves the overall classification performance. It can be seen that in the proposed algorithm, the sampling method converts the imbalanced data set into a balanced data set, which solves the imbalanced classification problem; the multivariate filter algorithm also eliminates irrelevant and redundant features to some extent. This effectively improves the predictive performance on imbalanced data sets.

## ACKNOWLEDGMENT

This work is supported by the special fund for scientific and technological innovation of Fujian Agriculture and Forestry University (No. CXZX2016026, No. CXZX2016031).

## REFERENCES

[1] He P, Li B, Liu X, et al, An empirical study on software defect prediction with a simplified metric set, Information & Software Technology, vol. 59, 2015, pp. 170-190.

[2] Erturk E, Sezer E A, A comparison of some soft computing methods for software fault prediction. Expert Systems with Applications, vol. 42, April 2015, pp.1872-1879.

[3] Khoshgoftaar T M, Gao K, Napolitano A, et al, A comparative study of iterative and non-iterative feature selection techniques for software defect prediction. Information Systems Frontiers, vol. 16, May 2014, pp. 801-822.

[4] Gao K, Khoshgoftaar T M, Wang H, et al, Choosing software metrics for defect prediction: an investigation on feature selection techniques, Software—practice & Experience, vol. 41, May 2011, pp.579-606.

[5] Hall T, Hall T, Christianson B, et al, The jinx on the NASA software defect data sets// International Conference on Evaluation and Assessment in Software Engineering, ACM, 2016, pp. 13.

[6] Armah G K, Luo G, Qin K, et al, Applying Variant Variable Regularized Logistic Regression for Modeling Software Defect Predictor, 2016.

[7] Gao K, Khoshgoftaar T M, Napolitano A, An Empirical Investigation of Combining Filter-Based Feature Subset Selection and Data Sampling for Software Defect Prediction, International Journal of Reliability Quality & Safety Engineering, vol. 22, June 2015.

[8] Shivaji S, Akella R, Kim S, Reducing Features to Improve Code Change-Based Bug Predictiop, IEEE Transactions on Software Engineering, vol. 39, April 2013, pp. 552-569.

[9] T. M. Khoshgoftaar, K. Gao and A, Napolitano, An empirical study of feature ranking techniques for software quality prediction, International Journal of Software Engineering & Knowledge Engineering, vol. 22, February 2012, pp. 161-183.

[10] Huanjing Wang, Taghi M. Khoshgoftaar, Amri Napolitano, An Empirical Investigation on Wrapper-Based Feature Selection for Predicting Software Quality, International Journal of Software Engineering & Knowledge Engineering, vol. 25, January 2015, pp. 93-114.

[11] Gao K, Khoshgoftaar T M, Wang H, Exploring filter-based feature selection techniques for software quality classification, International Journal of Information & Decision Sciences, vol. 4, 2012, pp. 217-250.

[12] You M, Chen Y, Li G, Im-IG: a novel feature selection method for imbalanced problems, Journal of ShanDong University( Engineering Science), vol. 40, May 2010, pp. 123-128.

[13] Jian X, Han S, Cui C, Relief feature selection algorithm on unbalanced datasets, Journal of Data Acquisition and Processing, vol. 31, April 2016, pp. 838-844.

[14] Lu X, Chen M, Wu Z, et al, Web spam detection based on immune clonal feature selection and under-sampling ensemble, Journal of Computer Application, vol. 36, July 2016, pp. 1899-1903.

[15] Lu H, Zhang J, Ma X, et al, Study of Over-Sampling Method Based on Feature Selection, Journal of Telecommunications Science, vol. 28, January 2012, pp. 87-91.

[16] Hall M A, Correlation-Based Feature Selection for Machine Learning, 1999, pp. 19.

[17] Andrzejewski D, Zhu X, Craven M, Incorporating Domain Knowledge into Topic Modeling via Dirichlet Forest Prior// International Conference on Machine Learning. Proc Int Conf Mach Learn, 2009, pp. 25.