Weighted k Nearest Neighbor Using Grey Relational Analysis to Solve Missing Value

Desepa Isna Ulumi and Daniel Siahaan

Abstract—Software defect prediction model is an important role in detecting the most vulnerable component error software. Some research have been worked to improve the accuracy of the prediction defects of the software in order to manage human, costs and time. But previous research used specific dataset for software defect prediction model. However, there is no a generic dataset handling for software defect prediction model yet. This research proposed improvements to the results of the software defect prediction on the merged dataset, which is called generic dataset, with a number of different features. In order to balance the number of features, each dataset should be filled with a missing value. To fill the missing values, Weighted k-Nearest Neighbor (WkNN) method was used. Then, after missing values were filled, Naïve Bayes was used to classify the selected features. This research needed to obtain a set of features which was relevant, then performed a feature selection method. The results showed that by using seven NASA public MDP datasets, Naïve Bayes with Information Gain (IG) or Symmetric Uncertainty (SU) feature selection presented the best balance value.

Keywords—Software defect, NASA public MDP, weighted KNN, Naïve Bayes.

I. INTRODUCTION

Software defect prediction model can predict defect-prone modules from the basis defect or non-defect labels [1]. A software metric which contains measurements is used to identify potentially defect modules, so that it can minimize the cost and time in a project. Because the testing process is focused on the most vulnerable modules of defect.

Various kinds of methods to solve the software defect prediction problem have been proposed. One of them implicates by using Greedy Forward Selection (GFS) feature selection method and the Ensemble Learning Classification technique [2]. This methods use six datasets NASA public MDP and can resolve imbalance data and redundant features. Ensemble Learning Classification is classification technique by calculating mean from some others classification techniques. However, software defect prediction model is built for specific dataset. Another method uses Defect Prediction by using Relational Association Rule (DPRAR) for classification method [3]. This approach uses three steps. The first step is pre-processing that determines the dependence between the feature and the target output. The second step is calculating Spearman rank correlation coefficients, which have no effect deleted. The last is training phase by using a DPRAR algorithm which focuses on identifying the relationship between two software metrics (features). If relevant features are used at the next step, the model testing phase which has been built, should be tested per dataset. The results were compiled by using several accuracy measurements, namely probability of detection (pd), specificity, precision, and Area Under the ROC curve (AUC) area. But, the model software defect prediction can only be used for specific dataset.

Another method uses five selection methods and Cluster Based Classification (CBC) with seven datasets NASA public MDP [4]. Each datum has a domain. Each domain has a number of different features and similar features. A feature is a property or characteristic in data which have various value, either from one object to another or from one time to another. Gain Ratio (GR), Information Gain (IG), One-R (OR), Relief-F (RFF) and Symmetric Uncertainty (SU) are categorized as five feature selection methods. These selection features generate the best performance in information feature combination. This combination gives the best result rather than previous methods in terms of accuracy of software defect prediction. Combination of CBC classification and IG feature selection is better compared with others feature selection methods. However, although the method gives the best result, but it also less efficient. The reason is the method do one-on-one processing based on number of features of each dataset with different software prediction models. Therefore, modelling the software defect prediction for generic datasets with different number of features are necessary to be done.

In this paper, we propose a new software defect prediction model using generic datasets with different number of features. To solve the differences in the number of features, this study proposes how to fill the missing value with Weighted k-Nearest Neighbor (WkNN), so that, the proposed method can improve the software defect model for another multiple dataset.

II. METHOD

Software defect prediction in this paper consists of fifth steps. In the general method in this study is presented in figure 1.

A. Sequence feature selection

In the first stage, the researcher order the feature that have less missing value after reduce redundant data. Redundant feature is feature that have same value and class [5]. The highest redundant feature is PC2.

B. Weighted k Nearest Neighbor (WkNN)

Nearest Neighbor (NN) is used to identify data points that are not yet classified [6]. Distance is evaluated from all training to testing data. The lowest distance value is called nearest neighbor. k-Nearest neighbor has some advantages such as easy to learn, resistance to noisy training data, effective if the training data is large [6]. But the k-Nearest Neighbor (kNN) method has memory limitations, complex computing, slowly running process.
and gullible with irrelevant features. This technique is easy to implement but the k value affects the result. So T. Bailey and A. K. Jain modified the kNN by weighting and named weighted kNN (wkNN). WkNN is a method that evaluates the distance based on the value of n and the weight of each calculated value. The advantage of wkNN is to overcome the limitations of kNN by adding weight to each k, using all training samples not just k values, and suitable to be implemented in all datasets. In general, the weighted kNN process can be seen in figure 2. Data are divided two groups, complete and incomplete data. The next step calculate the nearest neighbor that calculate the distance of complete and incomplete data. Nearest neighbor is obtained by equation (1):  

\[
distance (x_i,x_j) = \sqrt{\sum_{l=1}^{n}(x_{il} - x_{jl})^2}
\]  

(1)

where \(x_i\) is incomplete instance and \(x_j\) is complete instance. Each incomplete instance is obtained by equation (2):  

\[
x_{ij} = \frac{\sum_{p=1}^{k} x_{jp}}{k}
\]  

(2)

where \(x_{ij}\) target instance \(X_i\), \(p\) is feature \(p\) in instance \(X_i\) and top \(k\) is based on \(\{X_1, X_2, ..., X_k\}\). According [7], before entering the next stage the data needs to be normalized. Normalized is obtained by equation (3):  

\[
x_i(j) = \frac{x_i(j) - \min_{n=1}^{n} [x_i(j)]}{\max_{n=1}^{n} [x_i(j)] - \min_{n=1}^{n} [x_i(j)]}
\]  

(3)

Where \(x_i(j)\) is data \(i\) feature \(j\), \(\min_{n=1}^{n} [x_i(j)]\) is minimum value of each feature and \(\max_{n=1}^{n} [x_i(j)]\) is maximum value of each feature. Where \(x'_i(j)\) is feature \(j\) and shows the greatest value of each feature. Then calculate the distance with matrix form as we can see an equation (4):  

\[
\Delta_{oi}(j) = x'_o(j) - x'_i(j)
\]  

(4)

where \(\Delta_{oi}(j)\) is distance incomplete instance and complete instance after normalization and \(x'_i(j)\) is the value after normalization. Grey relational coefficient is calculate that to know the relationship of the ideal and actual experimental results as we can see an equation (5):  

\[
GRC \gamma_{oi}(j) = \frac{\Delta_{min} + \rho \Delta_{max}}{\Delta_{oi}(j) + \rho \Delta_{max}}.
\]  

(5)

GRC \(\gamma_{oi}(j)\) is grey relational coefficient, \(\rho \in [0,1]\) is a commonly defined coefficient \(\rho = 0.5\) [7][8], \(\Delta_{min}\) is minimum value on \(\Delta_{oi}(j)\) and \(\Delta_{max}\) is maximum value on \(\Delta_{oi}(j)\). Then grey relational grade is calculated with mean value of grey relational as we can see an equation (6):  

\[
GRG(Y,X_i) = \frac{1}{m} \sum_{k=1}^{m} GRC \gamma_{oi}(j)
\]  

(6)

where \(m\) is amount of feature. The higher \(GRG(Y,X_i)\) the correlation between \(Y\) and \(X_i\) is getting stronger. The stronger the correlation, the greater the weight gain. In most cases, the weight of each nearest neighbor is defined as follows (7):  

\[
w_j = \frac{1}{d_j}
\]  

(7)

where \(d_j\) is distance instance \(j\) and target instance \(i\). Filling missing value is obtained by equation (8):  

\[
x_{ip} = \frac{\sum_{j=1}^{n} w_j x_{jp}}{\sum_{j=1}^{n} w_j}
\]  

(8)

where \(x_{ip}\) is missing value form \(X_i\) instance.

C. K Cross Fold Validation

In this study, ten cross fold validation is used.

D. Naïve Bayes Classification

After filling the missing value, the next step is naïve bayes classification. The iteration can stop if the balance value \(n\) is less than the balance value \(n-1\).

E. IG, GR, OR, SU, RFF Feature Selection Method

Some features from the previous step are selected that use five feature selection method. The approach of feature selection method used in this research is filter. Filter is feature selection method based on feature rank [5].

1) Information Gain

Information gain is one of the feature selection techniques that is able to assess the importance of features by measuring class related [9]. Generally the information gain can change the value of the uncertainty of an information (entropy) into a measure of the value of information to be obtained. The value of information gain is obtained by equation (9):  

\[
IG(\text{Class}|\text{Attribute} = H(\text{Class}) - H(\text{Class}|\text{Attribute})
\]  

(9)

where \(H\) is the entropy. It is assumed that \(A\) is all features and classes dependent on all training. Example value \((a, y)\) with \(y \in \text{Class}\) defines the value of the specific instance for the feature \(a \in A\), \(V\) represents the set of features i.e. \(V = \{\text{value}(a, y) | a \in A \land y \in \text{Class}\}\). The IG formula on each \(a \in A\) feature is defined as follows (10):  

\[
IG(\text{Class}, a) = H(\text{class}) - \sum_{v \in V} \frac{|\{y \in \text{Class} | \text{value}(a, y) = v\}|}{|\text{class}|} \times H(y|\text{Class}|\text{value}(a, y) = v)
\]  

(10)

2) Gain Ratio

Gain ratio modifies the information gain technique by taking into account the number of results obtained by the feature test condition [10]. The value of gain ratio is obtained by equation (11):  

\[
GR(\text{Class}, a) = \frac{IG(\text{class}, a)}{H(a)}
\]  

(11)

where \(H(a)\) is obtained by equation (12):  

\[
H(a) = -\sum_{v \in V} \frac{|\{y \in \text{Class} | \text{value}(a, y) = v\}|}{|\text{class}|} \times \log_2\left(\frac{|\{y \in \text{Class} | \text{value}(a, y) = v\}|}{|\text{class}|}\right)
\]  

(12)

All the notations in the gain ratio formula are the same with IG.

3) One R

One R is built a feature called one rule for each feature in the dataset [5]. Algorithm One-R is defined as follows [11]:
For each feature $f$.

For each value $v$ from the domain of $f$

Select an instance set with feature $f$ having a value of $v$

It is assumed that $c$ is the class that has the highest frequency

Apply "if feature $f$ has value $v$ then class is $c$" for feature $f$

Output rules with the highest classification accuracy.

4) Symmetric Uncertainty

Symmetric Uncertainty (SU) also compensates the IG bias against features with a more different value and normalizes the value in range 0 to 1 [10]. The value of symmetric uncertainty is obtained by equation (13):

$$SU(Class, a) = 2 x \frac{H(Class) + H(a)}{H(Class + a)}$$

The equation similar with IG and GR.

5) Relief F

Relief F is a feature selection techniques which evaluate several times and gives weighted value for each feature based on feature ability to differentiate each class and get the features which the weighted value fulfill the threshold value according to relevan features [11]. The Relief F algorithm is shown as below:

**Input:** a training set $D$, the number of iteration $m$, the number of nearest neighbors $k$, the number of features $n$, predefined feature weight threshold $\delta$.

**Output:** feature subset $S$ constituted by features whose weights are all greater than the weight threshold $\delta$.

**Step 1:** Let $S=\emptyset$, set all feature weights $W(F_t)=0$, $t = 1, 2, ..., n$.

**Step 2:** For $j=1$ to $m$ do

1. select a sample $R$ from $D$ randomly.
2. find out $k$ nearest neighbors $H_i$ ($i = 1, 2, ..., k$) from the same class and $k$ nearest neighbors $M_i(C)$ ($i = 1, 2, ..., k$) from each different class $C$.
3. For $t=1$ to $n$ do

$$W(F_t) = W(F_t) - \frac{diff(R_i, R_j)}{\sum_{k=1}^{m} W(F_t)} + \sum_{c\in Class R}$$

**Step 3:** For $t=1$ to $n$ do

If $W(F_t) > \delta$ then add fitur $(F_t)$ to $S$.

In (1), $P(C)$ is the probability distribution of class $C$, $Class(R)$ is the category $R$ belongs to, $M_i(C)$ denotes the $i$ Near Miss of $R$ in class $C$.

$diff(R_i, R_j)$ denotes the difference between $R_i$ and $R_j$ on $F_t$. If $F_t$ is discrete:

$$diff(R_i, R_j) = \begin{cases} 0; & R_i[F_t] = R_j[F_t] \\ 1; & R_i[F_t] \neq R_j[F_t] \end{cases}$$

If $F_t$ is continue:

$$iff(R_i, R_j) = \frac{|R_i[F_t] - R_j[F_t]|}{max(F_t) - min(F_t)}$$

The last step is naïve-bayes classification and the process is the same with the fourth step.

**III. Conclusion**

In this research, based on our experiments and analysis, Naïve Bayes with Information Gain (IG) and Symmetric Uncertainty (SU) feature selection presented the best balance value which is 0.4959. It is proven that not all features are used in this research. In addition, our proposed method can also improve the performance of software defect prediction with the best result.
Table V. Confusion Matrix

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Actual</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Defect</td>
<td>TP</td>
<td>FP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Non-defect</td>
<td>FN</td>
<td>FP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

REFERENCES


