An improved supervised learning defect prediction model based on cat swarm algorithm

To cite this article: Xuchen Wang et al 2018 J. Phys.: Conf. Ser. 1087 022005

View the article online for updates and enhancements.
An improved supervised learning defect prediction model based on cat swarm algorithm

Xuchen Wang, Haihua Yan, Jiayi Li
School of Computing, Beihang University, BUAA, Beijing, China
xuchen.wang@foxmail.com, yhh@buaa.edu.cn, ljyss9@163.com

Abstract. Software defect prediction has been the hot topic in the field of software engineering, for software modules that require validation and high-quality requirements, Software defect prediction provides a way to minimize nonessential software expenditures on the premise that accurate test can be performed. On the one hand, as a kind of Swarm intelligence algorithms, the Cat Swarm algorithm is a typical Swarm intelligence algorithm that appeared in recent years, and it can combine itself with machine learning algorithms. Machine learning, on the other hand, as effective ways to set up models, ensemble learning is one of which has a better performance relative to the base learning method. In integrated learning, Random Forest based on bagging method is a method with good properties. By using this model, we can learn from a small number of software modules that are known to be defective, thus achieving the goal of predicting the defects of unknown modules, and reducing the total cost of quality assurance. In this paper, we will use cat swarm algorithm to improve the decision tree and improve the prediction effect of the entire random forest, and that using some pre-processing methods to solve imbalanced and high-dimensional problems in datasets. From the results, we can see that the improved random forest method based on cat swarm algorithm using feature selection method has better effect than old one. In comparing our results with published benchmarks, the performance improvement can be clearly demonstrated.

1. Introduction
As the time of software development gradually shortened, the release of software as well as the software quality assurance is increasingly dependent on automated testing technology. If we want to get the most efficient automatic test, establishing a model of software defects is the best way, once the defect model is completed, it will be in the process of software quality assurance as much as possible to have the most effective effect and can reduce the cost for this aspect of the software development. During the process of setting up a software model, the most important concept is the metrics, the concept of metrics was firstly proposed by the Akiyama [1], he thought the number of software defects in the software development process is closely related to the number of rows with software code (LOC) and they have a linear relationship. As we see the formula below: D=4.86+0.018L, which indicates that the relationship. After that, the McCabe [2] and Halstead [3] successively put forward the McCabe complexity and Halstead complexity, more recently, more and more scholars proposed their own viewpoint metrics mainly including the changes of the software, the habits of software developers, code changes and information entropy. When using the accurate metrics to establish defect model, then developers or testers can use the model to verify and predict modules which are unknown. The basic
process of establishing a software defect model is on the figure 1.

![Diagram of model establishment process]

Fig. 1: process of establishing model

Firstly, data collection, which is based on metrics and defects largely depend on the data we collect, is the first step of setting up a software defect model and will decide what machine learning algorithms we choose at step 3. The second step is data preprocessing, mainly feature selection and imbalanced solution, will be mainly introduced in the second part. The third part, in practical application, the defect prediction model is set up the vast majority of cases based on machine learning algorithms, and based on the data, in most cases, the model is a supervised learning method, learning supervision method using module and defect metrics data with labeled data as a data set is used for training, and build the model. Besides, there are semi-supervised learning and active learning, their difference is that for the defect marking of the different training data set. Model aims to predict the unlabeled data. Obviously, the effect of supervised learning will be better if the dataset become bigger, but in reality, collection of data set is not easy, it will lead to rising costs, and costs include human resources, tools and time. In the case of small data sets, how to establish effective prediction model will be a important problem. One of the purposes of this paper is to explore how to focus on this problem. We will use a supervised learning method for defect prediction model establishing, and to evaluate the effect of the model. In the relevant literature, semi supervised learning is mainly referring to the training data set and contains a tag data and data without labels.

In order to obtain good results, in this paper, we will use the method of self-training [4][5] and EM algorithms [6] to set up the model of EM algorithms, including bagging [7] and boost [8] two kinds of classifiers, we will use bagging based on cart decision tree classifier and using the self-training [9] method to set up the model of this method, referred to as random forest [10] (Random Forest, referred to as RF).

On the other hand, in recent years, in the observation and Research on ants and other gregarious colony behavior, scholars have put forward a kind of biological groups inspired by artificial intelligence to realize the model, briefly, is "simple intelligent theme through the cooperation showed the characteristics of complex intelligent behavior. In addition to the classical ant colony algorithm and particle swarm optimization [11][12], an important method cat swarm algorithm was proposed Shu-Chuan Chu[13][14] et al in 2006, the cat swarm algorithm is put forward based on the two behaviors of the cat, the lazy rest and find the target quickly capture prey are known as search mode and tracking mode, the search mode represents the global search process in the tracking mode represents the local optimization in the process of search. Although the proposed year is new, it has been applied in engineering and machine learning such as clustering algorithms [15].
In the existing software defect prediction model method, almost all methods are limited in the framework of machine learning methods and related processing, and can’t be combined with some heuristic methods, in this paper, we will combine the tracking mode of cat swarm algorithm with CART decision tree to improve every node optimization in order to reinforce the effect, to improve decision tree.

In this paper, we propose improved Cart decision tree based on cat swarm algorithm called decision-tree-with-CSO (CDT for short), so we use random forest method based on the improved decision tree, during the process in a stochastic feature selection called improved Random-Forest (EXTRF), it can be recycled as a part of the training set, and from which the random extraction part and random feature extraction process metrics retain several absolute metrics. The same before using the data, we will carry on the data feature selection method [14] for preprocessing, feature selection will remove some redundant data and small changes in data, to select some high correlation data.

In the second part, we will briefly introduce the data preprocessing method and specific content of cat swarm algorithm in the third part, we will show the overall ideas and methods, using the random forest method we performed after feature selection to establish the defect model of the fourth part, we will describe our experiments and show the overall results, related work and the discussion of the results will be carried out in the fifth part, and finally we will provide an overall conclusion.

2. feature selection technology and cat swarm algorithm
In order to improve the quality of software, software developers have put a lot of machine learning methods into the establishment of software defect prediction models. In general, common machine learning algorithms have the following:

- K nearest neighbor algorithm (KNN) [16], the kernel of the algorithm, is that if a sample is mostly in the same class within the k most contiguous samples in the feature space, the sample belongs to that category.
- Support vector machine [117] (SVM), which is based on the VC dimension theory and structural risk minimization principle of statistical learning theory.
- Logistic regression, it is a generalized linear regression analysis model, and Logistic regression is a method that can fundamentally solve the problem that the variables are not continuous variables.
- Decision tree: is mainly made up of three parts, including node, branches and leaf. The root node is most important node. The whole decision process starts with the root decision node, from top to bottom. According to the classification of the data, different results are given at each decision node.

The following figure reflects the whole process:
Depending on the scope of use and convenience, developers will use different algorithms for research and development. However, the use of the algorithm is limited by the data set. For example, when using the KNN algorithm, although the use of the KNN algorithm theory is simple, and easy to implement, but for large capacity data sets, large amount of calculation, and when the sample is not balanced, the prediction will have large deviation. Moreover, logistic regression can lead to poor prediction results when faced with multidimensional large data. The artificial neural network algorithm has a long learning time, and may fall into the local minimum, and the parameters are more, and the weights and thresholds are not easy to determine and set.

Therefore, the establishment of software defect prediction model, as in figure 1, the data set is the first step to establish the whole model, and fully understand the data set will be basis on how to use different methods to establish model. In the practical application of machine learning, the number of features usually have a lot, which may exist irrelevant features, and may also exist the characteristics of mutual dependence. In addition, generally-speaking, for all of the data, if defective data occupies a minority, there will be risk of data imbalance so that the original data set if you do not use any treatment, will likely lead to the following consequences: the larger the number of features, the longer data analysis, model training time required; in addition, the large number of features easily lead to "dimension disaster", the model will be more complex, the applicability will decline.

Dimensionality reduction [18] [19] comes from the process of signal processing. The main difference between feature selection and dimensionality reduction techniques is that the purpose of feature selection is to answer the minimum number of features needed to construct an efficient prediction model. In contrast, the dimensionality reduction technique is the minimum dimension of the data structure needed to construct a prediction model.

Feature selection refers to the process of selecting subsets of related features from a given set of features. Common feature selection techniques are divided into three categories, filtering, wrapping, and embedding. The filter model is mainly based on the characteristics of the original training data and do not consider the choice of learning strategies, generally-speaking, this kind of method to sort the
data to obtain the characteristics of the highest ranking metrics based on the characteristics of the data correlation metrics. It is important to note that this sort of sorting should be able to reflect the influence factor of the data on the overall outcome. Common filtering model methods include Information Gain and CFS. The parcel model determines the criteria for evaluating the performance of the learner according to the performance of the learner to be selected. The premise of this method is to determine a prediction model, and the influence of characteristics on the result is determined by this model. Common wrapping methods includes Forward selection, and Backward selection.

For optimization problems, people are inspired from the physical or biological phenomenon and put forward many intelligent heuristic algorithm, genetic algorithm, including: according to the theory of evolution is proposed based on the idea of annealing algorithm and simulated annealing based on ant colony algorithm proposed ant colony behavior, in recent years, in view of the feline animal behavior observation, people put forward the cat swarm algorithm [20].

Cat swarm algorithm (CSO) divides cat behavior patterns into two categories: tracking mode and search mode. In order to imitate the behavior of cats in nature, part of the cat population executes search pattern, and the rest performs the tracking pattern. In search mode, copy cat's position on their own position, each copy of the change in a certain range to generate a new location, and the put results in the memory pool, the fitness value is calculated to determine the location of a point of comparison. In tracking mode, the individual cat changes its speed to update its position, adding a random perturbation to speed changes. When all cats finish searching and tracking, their fitness is calculated according to fitness function and the optimal values in the swarm are preserved. Finally, the iteration will be stopped until the result meets the requirements or reaches the maximum number of iterations.

![Fig. 3: process of cat swarm algorithm](image)

3. Method and mathematical model
The purpose of our research is to provide an effective and practical way to identify error prone modules. We propose a random forest algorithm based on CART decision trees with cat swarm algorithm to improve. In this section, we discuss the design and application of Random Forest algorithms based on feature selection.

First, let's flag the entire data set and related content. At the beginning, we use X to define the test data set which is a X (M X (P+Q)) vector, M represents the dimension of each group of data, but also can be said to be a measure of number of metrics, also, P represents the number of data sets containing the labeled, and Q is on behalf of the number of unknown defects of data sets, the relationship between
P and Q is determined by the relationship between the threshold. In addition, we use Y to represent markers, \( Y = \{ y_1, y_2, ..., y_n \} \), \( Y \), which is a one-dimensional array of two forms of \( Y \), for any \( y^f \), \( y^f = \{ 0, 1 \} \), \( 0 \) and \( 1 \) respectively represented by the data module without defect or defective. Obviously, \( Y = \{ y_{p+1}, ..., y_q \} \) The value of which is unknown and needs to be obtained using the predictive model. In addition, we use \( Z \) to define the data after preprocessing the data, \( Z \) is a \( (M'X(P' + Q)) \), where \( M', P' \) with respect to the original \( M \) and \( P \) have changed, this is because our feature selection and therefore solve the problem of imbalanced data, and relatively, the corresponding \( Y \) will have relative changes.

In this paper, we finally propose an improved decision tree algorithm based on cat swarm algorithm as base classifier to improve random forest algorithm (EXTRF) as the final algorithm. We use \( \theta(x) \) as the final prediction model, where is the \( \theta \) is the model itself, while the \( X \) is the data set required to be predicted. \( x \) is relative to the \( X \) data processing data set \( X = (M'XQ) \). The formation of prediction model is mainly based on the training data set and training data set, and a comparison between \( E \) and \( x \), will have a different proportion, generally speaking, the proportion depend on \( E / (E + x) \leq 0.5 \) to limit the threshold, we can see increasing the training set may make the training more accurate, but when it reaches a certain value, though the data set increases again, it will not make the efficiency of the model increases a lot. From the result of the \( \theta \), we tend to get the result is a floating-point value of \([0,1]\), while not the value of \( \{0,1\} \) which is boolean, such a result would allow people to understand the possibility of defects that a module may contains.

Figure 4 shows the whole process of the whole EXTRF algorithm.

Decision tree algorithm is the base learner of RF algorithm, which is also the decisive factor of RF's efficiency and performance, in a variety of decision tree model, compared with the ID3 algorithm or C4.5 algorithm, we choose to use the CART decision tree, the main reason is that the former algorithm using information gain for feature selection and calculating the information gain may be complex, besides, ID3 algorithm uses a relatively complex multi tree, also when in

```plaintext
//Input: Raw dataset X,Y
//Output: Prediction result
Data pre-process-Step : feature selection +
balance-solution
1. Input : X,Y;
2. X',Y'=Feature-select(X,Y)
3. X'',Y''=balance-solution(X',Y')
4. Output:X'',Y''
```

Random forest algorithm based on improved decision tree Based
on cat swarm:
1. Input : X,Y
2. loop until reach to demanded num of trees
3. D= selectMetric-by-Datafeature(X,Y)
4. X',Y'=randomFetch(X,Y,D)
   //Random extraction of quantity and measurement element
   //extract by metric will fetch D firstly
5. tree[i]=decisionTree(X',Y')//
6. end loop
7. forest=combination strategy(tree[i])/generated tree will
   adopt a combination strategy and create forest
8. output:forest //for prediction

Fig. 4: process of entire EXTRF algorithm

the face of large data, the speed of building the decision tree will be very slow, resulting in a decline in the efficiency of the entirety. In the CART tree, we introduce the Gini coefficient as the reference of feature selection, relative to the information gain, the calculation efficiency will be greatly improved,
and we introduce two points of the CART tree, making the computation efficiency improved, when facing the two-classification problem, if the output probability belongs to the first kind is \( p \), then the expression of Gini coefficient is:

\[
\text{Gini}(p) = 2p(p-1) \quad (1)
\]

Using the cat swarm algorithm, we use the cat swarm algorithm's track patterns, following we signifying description of the tracking mode, and the tracking model is used to simulate the cat tracking target. Update the cat's position by changing each of the cat's one-dimensional data.

A. Speed update. The best location for the entire cat population, also the current optimal solution, the speed of each cat is recorded as \( V_i = \{v_{i1}, v_{i2}, \ldots, v_{im}\} \), and we change the speed according to a formula (2).

\[
v_{i,d}(t+1) = v_{i,d}(t) + \alpha \times (v_{\text{best},d}(t) - v_{i,d}(t)) \quad (2)
\]

\( v_{i,d}(t+1) \) means that the updated \( i \) cat is at the speed of dimension \( d \), \( M \) is the dimension, and \( c \) is a constant. According to the different conventions, \( r \) is a random number.

B. Judge the reasonableness of the data

C. Update parameter

There are some common points between the cat swarm algorithm and decision tree, the aim is all to find an optimal value, and they all have a n-dimensional data, such as the metrics and the speed, although metrics can’t be changed like speed, but as long as we control changes in a range for metrics, then we can think it is the same, change can also be used to determine the reasonableness of mechanism to limit data. And when we find the optimal solution we want, that is, the optimal metric unit at two, we can assume that the tracking can be done. Of course, the mechanism of judgment is to use the Gini coefficients of the cart tree.

Before the establishment of the random forest, we need to preprocess the data set, including unbalance problem solution and feature selection to reduce data redundancy. To solve the data imbalance is relatively simple, when a large gap on the proportion between datasets whether have defects or not, we use sampling method, and the data set whether contains a defect or not tends to be the same. The last section shows that the dimensionality reduction technique can obtain a lower dimension, but also can destroy itself every metric properties of data sets so we choose to use the feature selection algorithm to preprocess the data, among three methods of the last section, we choose the feature selection technique as feature selection method in this paper.

Any supervised learning algorithm in random forests can serve as a base learner. However, we have imposed some restrictions on choice. First of all, learners should have shown competitiveness in software fault modeling domains. In this study, we selected decision tree as basal learners. The random forests can provide similar MDS similarity measure. In the process of software quality prediction, random forest can provide good performance consistent from beginning to end [24], [25] one of the reasons lies in its robustness to noise, for example, it has differences with the boosting method.

Our EXTRF’s process are showed in the figure 5.
4. Experiment

4.1 NASA software data set

All the data in our experiments comes from the production system. The NASA [10] metric metadata library provides metrics that describe software behavior from 13 different projects. We chose several of the typical projects as our subjects, each of which has more than 1000 modules. Table 1 provides a simple description of these items, and the ERROR_IDENTIFY metric in the data set has been removed, so the KC1 contains 22 metric elements, and the PC1, PC3 and PC4 contains 41 metric element. The last column of the dataset marks whether the module contains defects, which can be either one or more.

A. Experimental design

We intend to use a conventional approach to build software defect models, and our building process focuses on the following issues:

- The impact of cat swarm algorithms on decision tree performance;
- The influence of data preprocessing methods on model establishment;
- The impact of improved random forest algorithms on the overall model, and the impact of data preprocessing methods on the performance of the overall stochastic forest.

In view of the above three parts, in this paper, we will go through the following several tests for experiments, aiming at the first point, we will compare the common decision tree (a) (DT) and (b) improved decision tree using the cat swarm algorithm (CDT), for second and third point, (c) improved Decision tree based on feature selection (CDT.FS) and (d) improved random forest(EXTRF) are experimented to come to the conclusion. In addition to the above methods, the number of labeled data and unlabeled data ratio is a very important point, and in our experiment, proportion of label training set in all major settings in the data, 2% to 50%, specifically, we will be in the range of arbitrary choice part of the data set as training data, we select several specific proportion including 2%, 5%, 10%, 25% and 50%. For each corresponding method of each set of data sets, we will perform 10 independent experiments to obtain the average of the relevant data values. And the average output is displayed as a result.

In addition to the above required parameter data, some other parameters also need to be quantified. The first is the use of cat swarm algorithm, the formula (2) includes r and c and the value of S, the first r value will be determined by a random value [0,1], and c value will be determined by feature selection.
in data correlation coefficient, although it is a fixed value, each measure the element coefficients are different, the value of s are unified and fixed, the value of the objective is to make the change of the data is not too large, thereby resulting in lost data authenticity. For the elements in the feature selection, select the number of eigenvalues is also a need to determine the elements, through the test, we can learn that choosing 50% elements as the feature metric may be the better choice, both to ensure the authenticity of the data, and reduce data redundancy, eliminate the noise therefore making efficiency improved. Finally, in the experiments on EXTRF, there are several points that need to be identified, the number of trees in the random forest, and the number of invariant eigenvalues. After investigation, we believe that the 100 numbers ensure both efficiency and quick results in time.

We will evaluate the performance of our algorithm by measuring the area under the ROC curve (AUC).–We know that AUC has a lower change value and has better reliability relative to other software defect model evaluation standards. As we can see, the value of AUC will be between the threshold of [0,1], and it can therefore make possible the probability between TP and FP between different thresholds. We will compare the values of these different thresholds and map them into the line chart, and we can see that the correct trend of the line can better reflect the trend of the results.

In addition to AUC and ROC, the classification model evaluation is a very effective and intuitive method for evaluating the quality of the defect model. In order to evaluate the classification prediction model, the following prediction outputs are defined.

(a) Accuracy, recall rate, F value, support value

TP (True Positive): defect instances are predicted as defects
FP (False, Positive): no defect instances are predicted as defects
TN (True, Negative): no defect instances are predicted to be free of defects
FN (False, Negative): the defect instance is predicted to be free of defects

Through the above outputs, the evaluation criteria are defined as follows: Accuracy (accuracy) represents the ratio of the number of instances that are correctly classified in the entire test instance; accuracy = (TP+TN)/C.

The precision ratio (precision) represents the proportion of positive instances that are correctly classified in the number of numbers that are divided into positive cases; precision = TP/(TP+FP).

The recall rate (recall) represents the proportion of positive cases that are correctly classified in the actual positive case: recall = TP/P.

The F evaluation value is the mean value of precision and recall (harmonic, mean); F = (2 * (precision * recall)) / (precision + recall). C = P+N

(b) Accuracy-score: accuracy, whether the fraction is correctly predicted (default) or wrong (normalize=False).

(c) Results and analysis

Figure 6 depicts the performance of the fault prediction models on the KC1. The graphs mainly show us the AUC, and accuracy-score and F-measure about the project data. As we can see in the graph, the X axes represent the size of the labeled data used for model training. As mentioned before, each value presented in the graphs is the average results over ten times of the corresponding experiment. By observing, we can see that CDT defeat DT on all different kinds of evaluation indicators. Also, from the data, we find that CDT has got a brilliant result. As we see, we use a user-defined model rather than the other existing model so as to change the base code and process the experiment. From the table, we can find that our main change to the decision tree has made a great
First International Conference on Advanced Algorithms and Control Engineering  
IOP Publishing  
IOP Conf. Series: Journal of Physics: Conf. Series 1087 (2018) 022005  
doi:10.1088/1742-6596/1087/2/022005

Fig. 6: Performance results of KC1

Table I

| Data   | N modules | % Faulty | N features | Project description          | Language |
|--------|-----------|----------|------------|------------------------------|----------|
| KC1    | 2109      | 13.9%    | 22         | Storage management           | C++      |
| PC1    | 1109      | 6.59%    | 41         | Flight software              | C        |
| PC3    | 1563      | 10.43%   | 41         | Flight software              | C        |
| PC4    | 1458      | 12.24%   | 41         | Flight software              | C        |

Table II depicts the performance of the four fault prediction models on the KC1, PC1, PC3 and PC4. We can see that for KC1 and PC4, CDT shows a great result, but as to PC2 and PC3, CDT just owns a little effective. Besides, all of the four datasets can benefit a lot from the process of feature selection. And EXTRF always get the best result. Besides, from table III, we can find that our methods can be better than others while for PC4, our method isn’t the better one.

4.2 Threats to Validity

Following are several potential threats to the validity with respect to the experiments.

1) Difficulties of parameters’ settings: In our research, there are many parameters needing to be set and it is a little difficult to choose a proper value. As we know, adaptive value will lead to a great result and there aren’t formulas or rules which can decide the value.

2) Limitations to dataset: In our research, we only choose the NASA datasets to be our experiments’ dataset. So we can’t come to a conclusion that our algorithm can adapt to all of the dataset.

5. Related work

5.1 Methods
In traditional supervised learning, if we want to establish a prediction model, all data should be labeled before learning, and classifiers are then trained on these labeled data. When there is a portion between sample labeled and unlabeled, an effective way to solve this problem is to introduce semi-supervised learning [23–25]. An initial hypothesis is firstly learned from the labeled data, and then is refined through the unlabeled ones by certain automatic labeling strategy.

There are many methods proposed by different researchers. For examples, Huihua Lu et al. [26] implement a [27] semi-supervised learning approach. It predicts the labels of unlabeled data, and then combines the data to refine initial model. Results show that the approach perform significantly better than one of the most effective supervised learning approach - Random Forest - especially when few modules with known fault content are available. Qing He et al. [28]

| Dataset | ratio | Classifiers |
|---------|-------|-------------|
|         |       | DT | CDT | CDT.FS | EXTRF |
| KC1 2%  | 0.527 | 0.6617 | 0.6723 | 0.7823 |
| 5%     | 0.5509 | 0.6058 | 0.6813 | 0.7991 |
| 10%    | 0.5366 | 0.6222 | 0.6958 | 0.7967 |
| 25%    | 0.5412 | 0.6138 | 0.6991 | 0.787 |
| 50%    | 0.55  | 0.6372 | 0.7063 | 0.8011 |
| PC1 2%  | 0.5982 | 0.662 | 0.723 | 0.8102 |
| 5%     | 0.5516 | 0.6045 | 0.7182 | 0.8294 |
| 10%    | 0.6011 | 0.601 | 0.7056 | 0.8532 |
| 25%    | 0.5993 | 0.6174 | 0.7271 | 0.8725 |
| 50%    | 0.62  | 0.6637 | 0.7562 | 0.8933 |
| PC3 2%  | 0.5836 | 0.588 | 0.6887 | 0.8102 |
| 5%     | 0.577 | 0.6163 | 0.7289 | 0.8369 |
| 10%    | 0.5756 | 0.6314 | 0.7361 | 0.8571 |
| 25%    | 0.5942 | 0.6286 | 0.7536 | 0.8692 |
| 50%    | 0.6078 | 0.6512 | 0.7665 | 0.8863 |
| PC4 2%  | 0.584 | 0.6567 | 0.7685 | 0.8823 |
| 5%     | 0.5913 | 0.6378 | 0.7762 | 0.893 |
| 10%    | 0.6025 | 0.6544 | 0.7784 | 0.9078 |
| 25%    | 0.6135 | 0.6544 | 0.7863 | 0.9101 |
| 50%    | 0.6302 | 0.6838 | 0.7903 | 0.9213 |

| Dataset | KC1 | PC1 | PC3 | PC4 |
|---------|-----|-----|-----|-----|
| Menzies [21] | NULL | 0.48 | 0.8 | 0.98 |
| Lessmann [22] | 0.9 | 0.82 | 0.97 | 0.78 |

5.2 Metrics

If we want to set up a good prediction model, we firstly need enough training data, on which there are different kinds of metrics, commonly, includes Mccabe and Halstead metrics. With the further development of research, researchers have come up with some other kinds of metrics.

Taek Lee et al. [29] implement a new kind of metric called MIMs, MIMs mainly consist of micro interaction metrics. These metrics almost originate from develops’ editing habits such as ‘NumEditEvent’, which means the number of editing. The results show that these kinds of metrics can lead to a better efficiency. In addition, Feng Zhang et al. [30] introduce some statistic data as a new kind of metric.
6. Conclusions and future work

Software defect prediction is one of the most widely studied problems in software engineering literature [31]. Of all the models setting-up methods, in generally, supervised learning is mostly favored. But the drawback of supervised learning is obvious, the more data it has, the better the efficiency will be. Therefore, there appear semi-supervised learning and active learning [32]. Though there more and more methods or new metrics showing up, most of the algorithm are limited to the machine learning domain.

With the appearance of swarm intelligence algorithm, in this paper, we research the similarity between two kinds of algorithms and find out their common points, we choose cat swarm intelligence algorithm to be reference and use heuristic method to set up a new style of decision tree model.

Our results show that cat swarm intelligence algorithm can typically improve decision tree, therefore improving the efficiency of the random forest algorithm. Besides, we added the feature selection pre-processing strategy to the supervised learning algorithm, CDT.FS, it also shows a great assistance to the final results. The entire methods can reach to a pretty good efficiency.

From this study, we find that software fault prediction can combine traditional machine learning methods with heuristic methods. And our methods need to do more to consummate the whole method. Our future work will concentrate on three points:

1) How to more efficiently decide the parameter inside the whole method.
2) Applying more datasets to our method to see the results.
3) Using some other methods such as clustering algorithm to make our method more efficient.

References
[1] F. Akiyama, “An example of software system debugging,” in IFIP Congress (1), pp. 353–359, 1971.
[2] T. McCabe. A complexity measure. Software Engineering, IEEE Transactions on, SE-2(4):308–320, Dec 1976.
[3] H. Maurice. Halstead Elements of Software Science (Operating and programming systems series) [M]. Elsevier Science Inc. New York, NY, USA, 1977:53-56.
[4] C. Rosenberg, M. Hebert, and H. Schneiderman, “Semisupervised self-training of object detection models,” in Seventh IEEE Workshop on Applications of Computer Vision, 2005, pp. 29–36.
[5] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training.” Morgan Kaufmann Publishers, 1998, pp. 92–100.
[6] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, “Text classification from labeled and unlabeled documents using em,” in Machine Learning, 1999, pp. 103–134.
[7] L. Breiman. Bagging predictors. Machine Learning, 1996, 24(2): 123-140
[8] Y. Freund, R.E. Schapir. Experiments with a new boosting algorithm. In: Proceedings of the 13th International Conference on Machine Learning, Bari, Italy, 1996, 148-156
[9] C. Rosenberg, M. Hebert, and H. Schneiderman, “Semi-supervised selftraining of object detection models,” in Proceedings of the Seventh IEEE Workshops on Application of Computer Vision (WACV/MOTION’05) - Volume 1 - Volume 01, WACV-MOTION ’05, (Washington, DC, USA), pp. 29–36, IEEE Computer Society, 2005.
[10] L. Breiman, “Random forests,” Machine learning, vol. 45, no. 1, pp. 5–32, 2001.
[11] Sangita Roy, Samir Biswas, Sheli Sinha Chaudhuri; Nature-Inspired Swarm Intelligence and Its Applications[J]. I.J. Modern Education and Computer Science, 2014, 12, 55-65.[DOI:10.5815/ijmecs.2014.12.08]
[12] G. Beni and J. Wang, Swarm intelligence in cellular robotic systems. In NATO Advanced Workshop on Robots and Biological Systems, II Ciocco, Tuscany, Italy, 1989.
[13] Shu-Chuan Chu,Pei-wei Tsai,Jeng-Shyang Pan,Cat Swarm Optimization[A].9th Pacific Rim International Conference on Artificial Intellignece.Berlin,Germany:Springer Verlag,2006:854-858
[14] Shu-chuan Chu,Computational intelligence basedon the behavior of cats[J].International
Journal of Innovative Computing, Information and Control, 2007, 3(1): 163-173.

[15] Budi Santosa, Mirsa KCenca, Ningrum, Cat swarm optimization for clustering [A]. International Conference on Soft Computing and Pattern Recognition, Piscataway, USA: IEEE Computer Society, 2009: 54-59.

[16] ML Zhang, ZH Zhou, ML-KNN: A lazy learning approach to multi-label learning. Pattern Recognition, 2007, 40 (7): 2038-2048.

[17] C. Cortes, V. Vapnik. Support-vector networks. Machine Learning, 1995, 20 (3): 273-297.

[18] Andreas, Michael, 5, Hofmann, and H., “Interactive Data Visualization with Multidimensional Scaling.” Tech. Rep., Mar. 2004.

[19] M. Brucher, C. Heinrich, F. Heitz, and J.-P. Armspach, “A metric multidimensional scaling-based nonlinear manifold learning approach for unsupervised data reduction.” EURASIP J. Adv. Sig. Proc.

[20] G. Beni and J. Wang, Swarm intelligence in cellular robotic systems. In NATO Advanced Workshop on Robots and Biological Systems, Il Ciocco, Tuscany, Italy, 1989.

[21] T. Menzies, J. Greenwald, and A. Frank, “Data mining static code attributes to learn defect predictors,” IEEE Transactions on Software Engineering, vol. 33, pp. 2-13, 2007.

[22] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, “Benchmarking classification models for software defect prediction: A proposed framework and novel findings,” Software Engineering, IEEE Transactions on, vol. 34, pp. 485–496, 2008.

[23] O. Chapelle, B. Schölkopf, A. Zien, et al., “Semi-supervised learning,” MIT press Cambridge, 2006.

[24] M. Seeger, “Learning with labeled and unlabeled data,” Institute for Adaptive and Neutral Computation University of Edinburgh, 2000.

[25] X. Zhu, “Semi-supervised learning literature survey,” Tech. Rep. 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[26] H. Lu, B. Cukic, and M. Culp, “A semi-supervised approach to software defect prediction,” in Computer Software and Applications Conference (COMPSAC), 2014 IEEE 38th Annual, pp. 416–425, July 2014.

[27] G. Haffari and A. Sarkar, “Analysis of semi-supervised learning with the yarowsky algorithm,” in 23rd Conference on Uncertainty in Artificial Intelligence (UAI), 2007.

[28] Qing He, Beijun Shen, Yuting Chen; Software Defect Prediction Using Semi-supervised Learning with Change Burst Information[C]; 2016 IEEE 40th Annual Computer Software and Applications Conference, 2016: 113-122.

[29] Taeck Lee, Jaechang Nam, Donggyuun Han, SunghunKim, and Hoh Peter In; Developer Micro Interaction Metrics for Software Defect Prediction[C]; 2016 IEEE Transactions on Software Engineering, 2016: 1-22.

[30] Feng Zhang, Ahmed E. Hassan, Member, IEEE, Shane McIntosh, Member, IEEE, and Ying Zou, Member, IEEE; The Use of Summation to Aggregate Software Metrics Hinders the Performance of Defect Prediction Models[C]; 2016 IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 2016: 1-16 (http://ieeexplore.ieee.org/document/7539677/)

[31] Hall T, Beecham S, Bowes D, Gray D, Counsell S. A systematic literature review on fault prediction performance in software Engineering[J]. IEEE Trans. on Software Engineering, 2012, 38(6): 1276–1304. [doi: 10.1109/TSE.2011.103]

[32] Huihua Lu, Ekrem Kocaguneli and Bojan Cukic; Defect Prediction between Software Versions with Active Learning and Dimensionality Reduction[C]; 2014 IEEE 25th International Symposium on Software Reliability Engineering, 2014: 312-322.