Abstract—Fault-proneness of a software module is the probability that the module contains faults. A correlation exists between the fault-proneness of the software and the measurable attributes of the code (i.e. the static metrics) and of the testing (i.e. the dynamic metrics). 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. This paper introduces the evaluation of the fault proneness of modules in open source software system using k-NN clustering algorithm based on Object-Oriented metrics. The contribution of this paper is that it has used Metric values of JEdit open source software for generation of the rules for the classification of software modules in the categories of faulty and non-faulty modules and thereafter empirically validation is performed. The results show that the proposed approach can be used satisfactorily for finding the fault proneness in object oriented software components.

Keywords—k-NN Clustering Algorithms, Software Fault, Classification, Object Oriented Metrics, CK Metrics.

I. INTRODUCTION

As the complexity and the constraints under which the software is developed are increasing, it is difficult to produce software without faults. Such faulty software classes may increase development & maintenance cost, due to software failures and decrease customer’s satisfaction [11]. Prediction of fault-prone modules provides one way to support software quality engineering through improved scheduling and project control. There are many metrics and technique available for investigate the accuracy of fault prone classes which may help software organizations for planning and performing testing activities. Being able to measure the fault-proneness of software can be a key step towards steering the software testing and improving the effectiveness of the whole process.

The k-NN clustering algorithm is being successfully applied for solving classification problems. It is therefore important to investigate the capabilities of this algorithm in predicting software quality. In order to perform the analysis we validate the performance of the k-NN clustering method for dataset derived from open source software JEdit [15]. The 274 classes in this data were developed using Java language. In this study, we investigate the capability of k-NN clustering Algorithm in predicting faulty classes. We investigate the accuracy of the fault proneness predictions using object oriented design using metrics suite given by Chidamber and Kemerer [6] and used in [2] for fault prediction. By using K-NN clustering Algorithm technique on fault prone classes may enable the software organizations for planning and performing testing by focusing on accuracy of fault prone classes. This may result in significant improvement in software quality. The objectives of the study may be described as follows:

• To find the structural code and design attributes of software systems
• Developing the Fault Prediction model using the k-NN clustering algorithm

The contributions of the paper are summarized as follows:

First, open source software systems are analyzed. These systems are developed with different development methods than proprietary software. In previous studies mostly proprietary software were analyzed. Second, we examine k-NN clustering method to predict the faulty classes with better accuracy.

The paper is organized as follows: section 2 discusses the related work, Section 3 explains about the empirical data collection and section 4 describes the k-NN clustering based methodology. The result of the study is given in section 5 and conclusions of the research are presented in section 6 of the paper.

II. RELATED WORK


Khoshgaftaar [10] introduced the use of the neural networks as a tool for predicting software quality. Saida Benlarbi [17] surveyed that the basic premise behind the development of object-oriented metrics is that they can serve, as early predictors of classes that contain faults or that are costly to maintain.

Runeson & Magnus [16] proposed that in striving for high-quality software, the management of faults plays an important
role. The faults that reside in software products are not evenly distributed over the software modules; some modules are more fault-prone than others.

Briand [5] extracted 49 metrics to identify a suitable model for predicting fault proneness of classes. The system under investigation was medium sized C++ software system developed by undergraduate or graduate students. The eight systems under study consisted of a total of 180 classes.

A. Mahaweearat [13] analyzed that to remain competitive in the dynamic world of software development, organizations must optimize the usage of their limited resources to deliver quality products on time and within budget.

P. Bellini [3] compared Fault-Proneness Estimation Models that are developed using logistic regression and the discriminate analyses.


Yan Ma [21] suggested that accurate prediction of fault prone modules in software development process enables effective discovery and identification of the defects.

G. Pai [14] validated the public domain NASA data set as used in their study to predict fault proneness models with respect to two categories of faults: high and low. In this also used the same data set using a Bayesian approach to predict fault proneness models. Aggarwal. [2] validated object-oriented metrics to predict faulty classes.


S.K. Dhiman [7] presented a Genetic Algorithm Based Classification Approach for Finding Fault Prone Classes. In this paper Genetic algorithm approach can be used for finding the components.

III. EMPIRICAL EVALUATION

This section presents the goal of the study (section A), empirical data collection (section B).

A. Goal

The goal of the study is to evaluate the performance of k-NN clustering method. The performance is evaluating using OO metrics and CK metrics for predicting faulty classes using open source software.

B. Empirical Data Collection

We used JEdit open source software in this study [15]. JEdit is a programmer’s text editor developed using Java language. JEdit combines the functionality of Window, Unix and MacOS text editors. It was released as free software and the source code is available on [20]. JEdit includes 274 classes. The number of developers involved in this project was 144. The project was started in 1999. The number of bugs was computed using SVC repositories. The release point for the project was identified in 2002. The log data from that point to 2007 was collected. The header files in C++ were excluded in data collection. The word bug or fixed was counted. Details on bug collection process can be found in [19, 17].

The following metrics are used in the classification process:

- Coupling Between Objects (CBO)
- Lack of Cohesion (LCOM)
- Number of Children (NOC)
- Depth of Inheritance (DOI)
- Weighted Methods per Class (WMC)
- Number of Public Methods (NPM)
- Lines of Code (LOC)

IV. K-NN CLUSTERING ALGORITHM BASED CLASSIFICATION TECHNIQUE

The k-nearest neighbor (kNN) method is usually used to carry out classifications. The classification is performed by using a reference data set which contains both the input and the target variables and comparing the unknown which contains only the input variables to that reference set. The distance of the unknown to the k nearest neighbors determines its class assignment by either averaging the class numbers of the k nearest reference points or by obtaining a majority vote from them.

The k-nearest neighbor algorithm (kNN) is part of supervised learning that has been used in many applications in the field of data mining, statistical pattern recognition and many others.KNN is a method for classifying objects based on closest training examples in the feature space. An object is classified by a majority vote of its neighbors. K is always a positive integer. The neighbors are taken from a set of objects for which the correct classification is known.

The algorithm on how to compute the k-nearest neighbors is as follows:

1. Determine the parameter k is number of nearest neighbors beforehand. This value is user defined.
2. Calculate the distance between the query-instance and all the training data samples.
3. Any distance algorithm can be used for calculating distance among instances.
4. Sort the distances for all the training samples and determine the nearest neighbor based on the kth minimum distance.
5. Since this is supervised learning, get all the Categories of training data for the sorted value which fall under k.
6. Majority of nearest neighbors is used as the prediction value.

With help of k-NN clustering algorithm, classification of the software components into faulty/fault-free systems is performed.

A. Performance Evaluation Measures

Let a is the number of modules that actually have no fault and classifier predicts no defects in those modules, b is the number of modules that actually have defects and classifier predicts no defects in those modules, c is the number of modules that actually have no defects and classifier predicts no defects in those modules, and d is the number of modules that actually have defects and classifier predicts defects in those
modules.

Accuracy
It indicates approximate of measurement results to the true value, precision to the repeatability or reproducibility of the measurement [9]. The accuracy is the proportion of true results (both true positives and true negatives) in the population.

Probability of detection
Probability of detection is the probability of system failure [7]. That means that the probability of a component B failing given that a component A has already failed is the same as that of B failing when A has not failed.

Probability of false alarm
Intuitively probability of false alarm is the fraction of buggy execution that raises an alarm.

V. RESULTS & DISCUSSION

The DIT, CBO, RFC, NPM and LOC metrics are selected from the set of eight metrics. The predicted model is applied to 274 classes of the dataset. The percentage of Accuracy, Probability of Detection, Probability of False alarm is recorded for the fault dataset. The Accuracy percentage is calculated as 82.84%. Thus, it is concluded that the prediction model based on k-NN clustering method delivers high Accuracy, high Probability of Detection and low Probability of False Alarm.

Hence, k-NN clustering can be used in constructing models for predicting faulty classes as tested on the open source object oriented systems. However, more similar studies should be carried out to confirm the acceptability of the predicted model.

REFERENCES


