Optimizing: The code smell using multilabel classification approach

[1]Er. Manpreet Kaur, [2]Dr. Daljeet Singh

[1]Research scholar, [2]Assistant Professor, Department of Computer Science and Engineering, Guru Nanak Dev Engineering College, Ludhiana, Punjab, India.

[1]manpreet030@gmail.com, [2]gndecds@gmail.com

Abstract: Code Smell Detection is a part of software engineering. It is coming under the software testing field. Various types of codes smell available and different methods used to detect the codes smell in the source code. Code smell refers to a mistake in the source code that shows the bad quality of the software that is increases the maintenance cost and also increases complexity. Code smell shows the negative image of the whole quality of the system. The code smell detection is most important that motivates researchers to invent different techniques to find out the code smells in systems. Refactoring provides high quality, improve performance, decrease cost, implementation and also provide an easy way to the development of software. Code smells are like features of the software that describe a code or design problem in which software not easy to understand and maintain. The detection tools are defined in the literature that provide different results. Machine learning methods have been proposed learning and features of affected code and affected bad smells. From various research papers observed that detect only one type of smell. In the proposed work detect two types of code smells with the help of multilabel classification.

***Keywords — Code Smells, Machine Learning Techniques, Bad code Smells*.**

I. INTRODUCTION

Code Smell is an impurity in the source code. Code smell also shows the bad image of the whole quality of the system. Therefore, the code smell detection is most important that motivate researchers to invent different techniques to find out the code smells in systems. Refactoring is a process in which overcome code-smell-related problems [1]. Refactoring provides high quality, improve performance, decrease cost, implementation and also provide an easy way to the development of software. Code smells are like features of the software that describe a code or design problem in which software not easy to understand and maintain [2], [3]. The code smell detection tools are defined in the literature that provide diff erent results. Machine Learning Methods have been proposed learning and features of affected code and non-affected by bad smells. In this paper, multilabel classification techniques used to detect the code is smelled or not by one or more code smells. Long method and feature envy are two code smells perform the task of code smell detection [4]. Code smells define potential problems in design or programming level problems in source code. Code Smell detection is a wide area for the researcher to find the best way to handle the code smell related problems. Code Smells reduce the quality of the software. Maintenance cost increase due to code smells in the source code [3]. In this paper discuss the implementation of the multilabel Classification Technique. Due to Bad Smells difficulties occur in source code. In this paper, two methods are used to perform a specific task that is long method and Feature Envy. Accuracy are 95% and 98% respectively [4]. Code Smells Changes the order of the system's design that increases the maintenance cost. Code Smells gives warning about various problems such as bad smells like god method etc. In Code Smells Refactoring is main important term that increases the quality, improves performance, decreases cost and easier software development process [5].

A. Bad Smells in Programming Code

Bad Smells is like bugs or errors in code. Code Smells like problems in code that occur during programming. Due to Bad Smells refactoring is must important to or re-examine the whole code of programming. Characteristics are rectified by using refactoring [6].

- Various Types of Bad Smells (symptoms) Code

 DOI : 10.35291/2454-9150.2020.0090

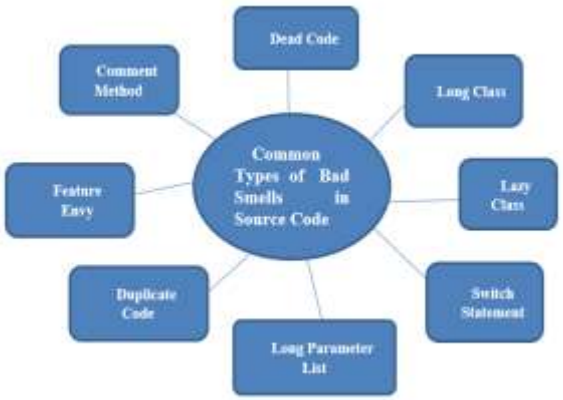

Fig. 1. Types of Bad Smells

- Dead Code: - Code that is not used frequently but it placed in source code. Dead code generates memory consumption-related problems.
- Switch Statement: - Multiple cases creates in the switch statement. Sometime it maybe can't handle. It is also generating bad smells.
- Duplicate Code: - When code written in source code. Sometimes the same type of code placed in source code. Then duplicate code type smell occurs.
- Long Parameter List: - Parameter list used limited because a large number of parameters generate complexity in the source code.
- Feature Envy: - Code features used in another place of code.
- Lazy Class: - Various methods is null and this type of class not widely used for work.
- Comments: - In source code, comments are more as compared to line of code in source code.
- Long Class: - Class contains various instances and a huge amount of code of lines.

II. LITERATURE REVIEW

Thirupathi Guggulothu et al. in this researcher discuss about code smells that are characteristics of software that find the design or code-related problems. Due to those problems' software cannot easy to understand and also increase maintenance costs. In this Literature proposed code smells detection tools provide different results. Smells that occur in code presented as subjectively. Machine Learning Methods or Techniques have been proposed to learn and classified the code that affected or not find from the source code. In previous research, Machine learning techniques used to detect only one type of smell in the source code. In this paper, the multilabel classification method used to detect smells more than one type from a source code. Code elements are affected by multiple smells. In this paper two types of datasets used for performing the task and then result converted into the multilabel dataset. At the end check performance compared with existing performance parameters [4]. Muhammad Ilyas Azeem et al. this paper presents machine learning methods used for detecting the code smells form the source code. Authors study the various machine learning methods based on four types of perspectives. Various types of bad smells (bugs or symptoms) considered and find the more usable algorithm for code smells. Commonly used algorithms for code smells are SVM (Vector Machine) and decision tree [7]. Pritiksha Sharma et al. this paper describes detecting the bad code smells known as code smell. Bad Code smells are detecting in object-oriented programming. Refactoring used to re-examine the code. Code smell detecting the result correct code detection applying for enhance the source code quality. Different types of code smell detection tools are proposed based on particular characteristics. The main motive of this work to find the various differences based on consequences. Maintainability index used for different object-oriented programming. BFOA optimization method is used for this proposed work [8]. Dario Di Nucci et al. this paper represents discuss code smells detection methods have been represented. The authors show the outputs of these methods work as subjective and method of detection. For recent work Machine Learning Techniques used for detecting the code smells have been proposed by authors, solve the various issues of detecting tools subjectivity to the learner's ability of affected code and affected source code [9]. This work provides a new wide-area for detecting code smells. In this instance are smelly by one type of code smell include in every dataset then it considered only case. For training and testing to the machine, the learner dataset is used. In this proposed work different types of datasets containing bad smells that affected by more than one type of code smell. Francisco Charte et al. this paper researchers discuss and imbalanced learning processes in the multilabel classification focusing on developing more than one goal. The first one goal represents the special measurement to fetch the unbalanced level of Multilabel Datasets. By using this measurement, we able to concluding the Multilabel Datasets that are imbalanced. The second type of goal is to develop various types of algorithms for decreasing the imbalance. Two types of approaches are used to separate into two grouped based on minority and majority of the instances. For each method or approach, two algorithms are used that is Random Under sampling and Random Oversampling provide results according to requirements [10].

III. MACHINE LEARNING TECHNIQUES

Machine Learning Method is like a platform that is used by the computer system to perform specific tasks [11]. The most widely used machine learning methods or techniques

 DOI : 10.35291/2454-9150.2020.0090

are discussed in the literature. Supervised machine learning technique is the best method that is used variables for predicting this type of variables work as independently. The huge amount of data is used for analysis [12], [13]. For code smell detection machine learning techniques widely used. It provides the best result. Machine learning methods help to improve the performance and accuracy.

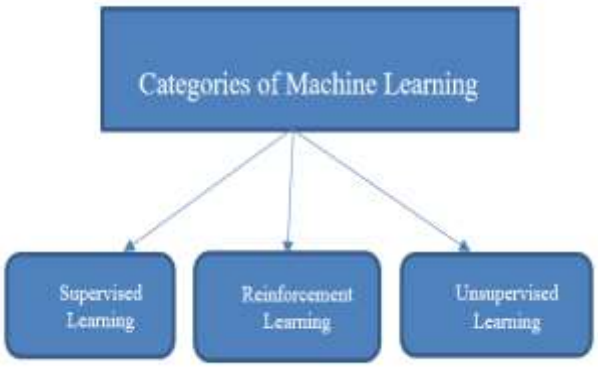

Fig. 2. Categories of Machine Learning

- Support Vector Machine**:** - It is a Supervised machine learning technique. It is most commonly used for classification and regression. The main aim of the support vector machine is minimizing the error and maximize the geometric margin [11].
- Random Forest Algorithm**:** - It is a supervised learning technique. It based on the regression tree and classification. Each tree depends upon the random vector values and experiment perform separately on each tree [14].

IV. MULTILABEL CLASSIFICATION MACHINE LEARNING TECHNIQUE: -

It is part of supervised machine learning technique. It is providing a relationship between class labels and instance. Build an algorithm to model automatically access the new instance labels. Various problems contain in multilabel classification are- Multilabel classification, binary, multiclass [15]**.** Multilabel Classification provides a suitable path to gain knowledge from various instances that are related to predictive classes. Domain of the code smells, code elements are instances and combinations of set of levels are called code smell. Various types of code smells exist in the code element. Previous existing techniques or approaches detected one type of smell, but this proposed method detects various types of code smells.

- Multilabel Classifier it provides facility to access information from the instances. It describes the relationship between levels.
- Binary Classifier is used to detect only one type of code smell [4], [16].

A. Application of Multilabel Classification

Fig. 3. Applications of Multilabel Classification

- Medical Diagnosis: - Patients record maintain for this purpose need to design various multi labels. Various electronics equipment's are used to record the health-related data [16].
- Multimedia Classification: -Mostly single types of labels detect the single instances at a time. But with the help of multilabel classification detects the multiple emotions at the same time by using music. Various multimedia like text, music, audio, and videos, etc. are used to show the emotion with multiple label [17].
- Text Categorization: - Text Categorization is the main difficult task for information retrieval. Text categorization plays an important role in business. It is coming under a supervised machine learning technique. Various companies are used for the division of companies on behalf of economic functionality. For economic functionality and categorization handle with help of multilabel and other machine learning techniques [18],[19].
- Semantic Scene Classification: - Classification problems arise due to various classes that are accessed by each other. Scene classification used a multilabel model that consists of a number of labels. Scene means something we view or seen [20][21]**.**

Multilevel dataset converted to one type of label problems and obtained result with appropriate one label of classifier. This process is known Problem Transformation method. Algorithm Adoption Method is used to handle one label of classifier. Binary Release is also called the baseline approach. Binary release and label power set two methods used problem transformation method [22]. Various algorithms involved binary released and label power set. Algorithm used for select label dependence [23]. This type of classifier obtained the best result and performance of code smell. Observing from various existing studies 76% accuracy achieved and detect only one type of code smell. But the proposed system detects two types of smells by

 DOI : 10.35291/2454-9150.2020.0090

using the same instances with 91% accuracy. The chain classifier is used as an alternative approach for transforming the problems of multilabel classification.

V. PROCESS OF MULTILEVEL CLASSIFICATION FOR CODE SMELL DETECTION

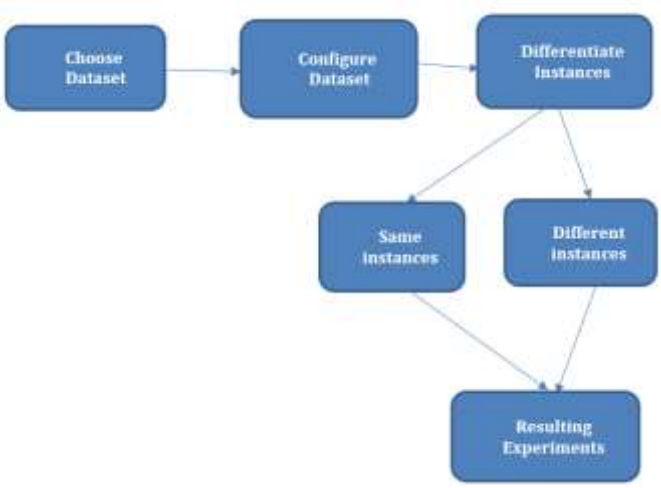

Fig. 4. Process of MLC

- Choose Dataset: - Collect the dataset for analysis of the smells in the source code for multilabel classification, long method and Feature Envy datasets consider for analysis. · Configure Datasets: - It takes instances as inputs some amounts. Then single label dataset adds with the multilabel dataset. The 10-fold method is used to find the performance classification. Instances are check and then it divides into parts or groups correct and incorrect.

- Differentiate Various Instances: - This step divides the dataset in common and uncommon of instances in the code. Create matrices for different instances. Two methods are used to remove various bugs or errors in the multilabel classification. Various mathematics calculations perform to complete the task. Accuracy, Hamming Loss, and Exact match ratio, etc [4]. Performance not improve with the help of dissimilar instances by using machine learning techniques. Long method and feature envy two methods of bad smells are detected in the source code by using multilabel classification techniques. Differentiate the affected and non- affected instances and find the ratio between them.

A. Accuracy:- Predict the correct labels w.r.t various labels in every instances.

B. Hamming Loss:- Predict the incorrect labels and find the unpredictable labels.

C. Exact Match Ratio:- Predict the labels from the instances.

A. Resulting experiment: - Basic Measurements are done by using attributes, labels, and various instances. This is used for a single label dataset. The classifier chain is used to maintain and improve the binary relevance. Predicting data or instance of every classifier add into the dataset as new characteristics [21]. The label Power set takes instances. Every label combination of class is a single type. Labels by the powerset of values of every class [19]. For check, the performance validation is applied which is called Tenfold cross-validations. Trained Multilabel datasets' features are shown as statistically. Performance parameters are checked increases the accuracy compare with existing results. Accuracy for feature envy 98% and long methods accuracy is 95% [4]. Affected instances called positive and non - affected instances called negative. Density is the ratio of average no. of active label and no. of labels in the dataset.

VI. CONCLUSION

We observering from various codes smell regarding research paper that detects bad codes smell by using various tools and techniques. For Code smell detection various machine learning techniques are used. Common instances are used to combined for dataset and remove the dissimilar instances. Multilabel classification is one of the most used techniques of machine learning to detect the code smells from object-oriented programming code. Multilabel classification is used to detect two types of code smells. Existing previous researches were done to detect only one type of smell in the source code. Multilabel classification choose the dataset then finds the affected and non-affected bad smells from the source code and common instances are avoid dissimilar data. The chain classifier provides the best result as compared to long-chain that improves the performance and accuracy.

VII. FUTURE SCOPE

Purposed method detects only two types long method and feature envy codes smell. Limitation of this machine learning approach to detect only two types of code smells from the instances in the source code. It can be expanded further research with the help of any other machine learning or optimization techniques. Improving the performance parameters and accuracy.

REFERENCES

[1] A. Al-Shaaby, H. Aljamaan, and M. Alshayeb, "Bad Smell Detection Using Machine Learning Techniques: A Systematic Literature Review," *Arab. J. Sci. Eng.*, no. 0123456789, 2020.

[2] R. Roy, R. Stark, K. Tracht, S. Takata, and M. Mori, "Continuous maintenance and the future – Foundations and technological challenges," *CIRP Ann. - Manuf. Technol.*, vol. 65, no. 2, pp. 667–688, 2016.

[3] A. Güzel and Ö. Aktaş, "A Survey on Bad Smells in Codes and Usage of Algorithm Analysis," vol. 5,

 DOI : 10.35291/2454-9150.2020.0090

no. 6, pp. 114–118, 2016.

[4] T. Guggulothu, "Code Smell Detection using Multilabel Classification Approach," 2019.

[5] "M. Fowler, K. Beck, J. Brant, W. Opdyke, D. Roberts, Refactoring: Improving the design of existing programs (1999)." .

[6] Y. Zhang and A. Haghani, "A gradient boosting method to improve travel time prediction," *Transp. Res. Part C Emerg. Technol.*, vol. 58, pp. 308–324, 2015.

[7] M. I. Azeem, F. Palomba, L. Shi, and Q. Wang, "Machine learning techniques for code smell detection: A systematic literature review and meta-analysis," *Inf. Softw. Technol.*, vol. 108, no. 4, pp. 115–138, 2019.

[8] P. Sharma and E. Arshpreet Kaur, "Design of testing framework for code smell detection (OOPS) using BFO algorithm," *Int. J. Eng. Technol.*, vol. 7, no. 2.27, p. 161, 2018.

[9] D. Di Nucci, F. Palomba, D. A. Tamburri, A. Serebrenik, and A. De Lucia, "Detecting code smells using machine learning techniques: Are we there yet?," *25th IEEE Int. Conf. Softw. Anal. Evol. Reengineering, SANER 2018 - Proc.*, vol. 2018-March, pp. 612–621, 2018.

[10] F. Charte, A. J. Rivera, M. J. del Jesus, and F. Herrera, "Addressing imbalance in multilabel classification: Measures and random resampling algorithms," *Neurocomputing*, vol. 163, pp. 3–16, 2015.

[11] I. Xplore, "Samuel, A.L., 1959. Some studies in machine learning using the game of checkers. IBM Journal of research and development, 3(3), pp.210-229."

[12] F. A. Fontana, M. V Mäntylä, M. Zanoni, and A. Marino, "Comparing and experimenting machine learning techniques for code smell detection," *Empir. Softw. Eng.*, pp. 1143–1191, 2016.

[13] "Alpaydin, E., 2014. Introduction to machine learning. MIT press." .

[14] L. E. O. Breiman, "Random Forests," pp. 5–32, 2001.

[15] M. I. Azeem, F. Palomba, L. Shi, and Q. Wang, "Machine learning techniques for code smell detection: A systematic literature review and meta-analysis," *Inf. Softw. Technol.*, vol. 108, no. 4, pp. 115–138, 2019.

[16] T. Baumel, J. Nassour-Kassis, R. Cohen, M. Elhadad, and N. Elhadad, "Multi-Label Classification of Patient Notes a Case Study on ICD Code Assignment," pp. 409–416, 2017.

[17] A. Ma, I. Sethi, and N. Patel, "Multi-Label Classification Method for Multimedia Tagging," *Int. J. Multimed. Data Eng. Manag.*, vol. 1, no. 3, pp. 57–75, 2010.

[18] S. Alsaleem, "Automated Arabic Text Categorization Using SVM and NB," vol. 2, no. 2, pp. 124–128, 2011.

[19] P. M. Ciarelli and E. Oliveira, "Multi-label text categorization using a probabilistic neural network," *Int. J. ...*, vol. 1, pp. 133–144, 2009.

[20] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognit.*, vol. 37, no. 9, pp. 1757–1771, 2004.

[21] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Multi-label machine learning and its application to semantic scene classification," *Proc. 2004 Internaional Symp. Electron. Imaging*, vol. 37, no. 9, pp. 1–12, 2004.

[22] S. Godbole and S. Sarawagi, "Discriminative methods for multi-labeled classification," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3056, 2004, pp. 22–30.

[23] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Mach. Learn.*, vol. 85, no. 3, pp. 333–359, 2011.

 DOI : 10.35291/2454-9150.2020.0090