BUG CLASSIFICATION IN CLOUD COMPUTING APPLICATION USING DEEP LEARNING

[1]Souvik Ghosh, [2]Anunay Ghosh

[1]Student, [2]Assistant Professor
Department of Computer Application
JIS College of Engineering
Kalyani, West Bengal, India

Abstract- **Cloud technology is not immune to defect classifying or errors (Bugs) classifying. It may come from internal system errors or hardware resource corruption. Minor bugs may cause screen freeze, unexpected messages and also may effect on usage. The unpredictable bug reports have a direct impact on the effectiveness of the bug detection model. If that bug remains unpredictable then we cannot record it as future reference. That will become more trouble. The accuracy of the system surely compromised by this problem. By manual process to review bug reports can take lots of time, and its truly hard work for testers and developers. We need to feature extraction of the dataset before to process it. After the dataset has undergone, we can put in a Machine learning algorithm to process it. This study proposes two classifiers linear regression or logistic regression classifier and Naive Bayes.**

Index Terms- Cloud **computing; Classifiers; Bugs; Deep learning.**

I. INTRODUCTION

In today's world we are all dependent on the digital platform. Every step of our work goes through a virtual environment. So, it is important to maintain those virtual platform or web application based on cloud technology. This cloud technology provides us few genuine services, among them I am discussing the SaaS (Software as a service), because I did my project over this topic. We all know that SaaS provides us ready to use applications, that can be used in various way in our real life, and we are thankful to have it. It makes our work easier and user efficient. So many organizations are depending their work on this. For these reasons we should well maintain this platform. There are lots of problems we have faced on this field, one of them internal errors, also called as bug or defects, which can come for different reasons. Manually we cannot solve this problem easily, because it's too hard to be shorted, and it will take lots of time and effort. Not only time and effort, but also it takes maintenance cost, which is too high. We need an automatic way which can solve this problem. An automatic way means here we have brought machine learning techniques basically I meant to say deep learning, which has so many model that can help us to do this. We need to find out which model will suitable for it. These models will help to classify the errors before starting to solve it, developers, those who will work to get the solutions can get help to fix the defects using these priorities. We all know that cloud technology is not immune to defects or bugs. It's also needed a virtual way to maintain this software. Machine learning provides so many patterns and algorithm that can get the problems and find out and then solve it, so that the employee or stuffs can do their own job efficiently and keep their focus on their work. To develop my idea, I am using naïve bayes classifier for this project. Let's have a theoretical look on my work.

Problem statement

There are lots of problem we have faced on this field. Doing all the maintenance work manually is not a good idea. This is totally cumbersome and hard work. It will bring cost related issues for an organization only for the maintenance purpose, and its time consuming. So, I made an idea to shift this process manual to automated. It is necessary to develop a system that can handle the whole procedure automatically, so that the works can do their work efficiently. My thesis paper is mainly focused on how we can classify these defects and manage the priorities using Machine techniques for the cloud-based Software as service applications.

Objective

- My objectives are simple, we need to develop a machine learning model that can classify those defects or error and manage the priorities, so that the developers can solve the problems and make a record for their future reference.
- Another objective is to get higher precision, recall and accuracy values from the selected classifier.
- Last objective is providing a better environment developing this model to them who are suffering from tis systematic issues and cannot do their job correctly for the lack of focus.

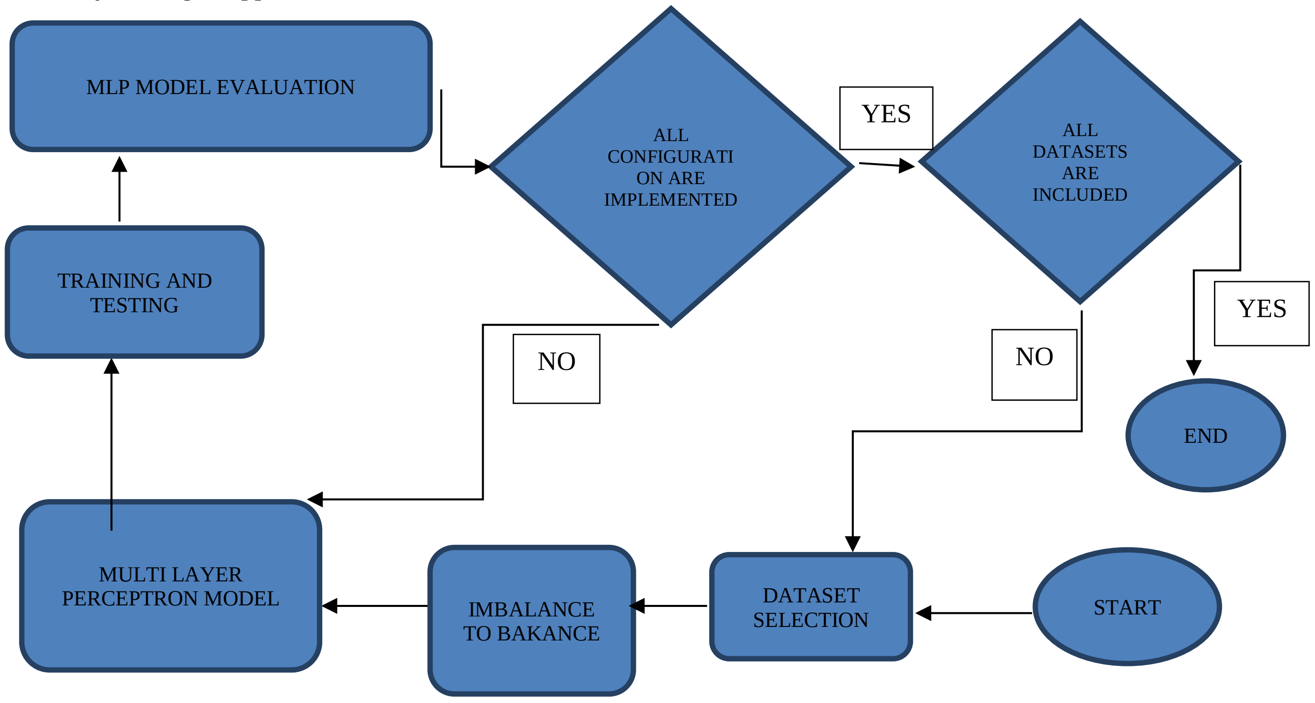
Literature review

There are multiple types of research, which has been done on bug classification of software from bug reports. There are also various tools built which can be used to identify bugs from the source code of the software and identify the priorities of those bugs. So, I have noted down a literature review on bug classification and prioritization of cloud-based application defect detection techniques. I have studied few research paper on this topic and I understand their working process, one them I am going to summarize, A developer team made a analysis of implementation problem for several bug reporting systems, this bug reporting system covering cloud computing. They have hosted typical implementations on three cloud computing services Infrastructure as a service, platform as service, software a service. They have instantiated so many virtual machines instance as enable the hardware resources. Their bug reporting architecture contains 2 models that is self-hosted bug reporting system and cloud bug reporting system architecture.

They have worked through 6 phases that is mainframe computing, pc computing, network computing, grid computing, cloud computing, internet computing.
From another research paper, we have seen they have developed a deep belief network using deep learning neural network, and then they have used long short-term memory which is subtype of the neural network. After that they have used resulting binary classifier to find the defects, they build a transformer model, which I am summarizing here, this is basically a multilayer transformer model where a input encoding and positional encoding process take place together and then multi head self-attention will work and the input values goes through the normalization, if not then it will go through the scaled dot-product attention and then it will concatenate. After normalization it will feed forward and again normalize and then give an output. Their research represents the latest progress in software error prediction using deep learning.
As conclusion, I will say the machine learning techniques provides a best result in bug classification in cloud computing and also provide a good accuracy using selected classifier. I have realized that it presents the best resolution in the whole software testing procedure. It is certainly important to use where bugs are undetectable, I mean harder to predict the class of defects.

II. Workflow of model

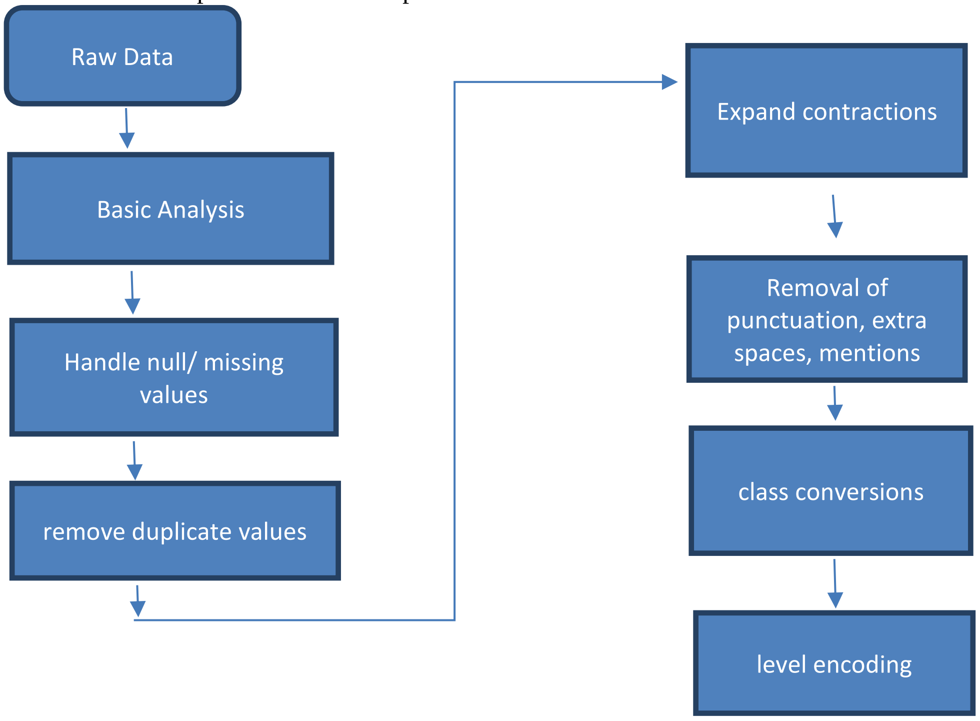
The model is simple and easy to understand. Let us discuss, at first, we need to collect dataset which is predicted as have some errors. Then we will start a dataset selection process where we take place our balance imbalance dataset splitting, imbalance datasets are need to be balanced to get the best precision and accuracy. After that it will go through the multi-layer perceptron model, where it will process using several hidden layers H=1, H=2, H=3 and neural networks. After that training and testing will take place to process the data, using k fold cross validation approach, where our K value is 4, that means every iteration will run 4 times, when 1st fold doing his job other will take rest. As an example, when in 1st iteration 1st fold is under testing, other fold will be under training. Now next step is most important which is MLP model Evaluation. And then two types of checking will be done 1st one is we will check all the configurations are implemented or not that mean all hidden layer configuration H=5, H=15, H= 20 or H=10, H=15 or H=10, H=15, H=30. If no then it go back to the MLP model generation process again, or if all configuration has done then next checking will take place that is all the data sets are applied or not, if not then it go back to the data set selection step again, which was our first step, or if all data sets are applied successfully then will it comes to end. The whole process is a model.
We need to find most suitable classifier at first. After getting the classifier, it will run on the transformed data to build a bug classification model. After finishing the model training, it is ready to perform evaluation process on dataset. Calculating different metrics like Avg Accuracy, F measure, precision, recall I will get the result. Then this model is ready to launch in cloud environment to classify the bug of application.



Dataset processing and performance evaluation

We all know that a dataset is a collection of information of a virtual platform and, these data quality is also important to build a good model. In today's word technology it's highly important to predict and categorize defects to make it more efficient. And it's really challenging to do, In my research.
Here the dataset errors are shown as a bar plot. These mistakes cause performance issues, as it may crash, errors over resource utilization. When we need to check the performance evaluation, we should remember the cross-validation technique. It is a most important feature of machine learning technique. This model is trained for k-4 cross validation, it means the process will run k times, until the whole iteration is done. Every fold of k is crucial. The result comes from every fold is acts as the final performance score. The value of K = 4 means to say there are 4 iterations and 4 folds. As an example, if 1st iteration is taking place, then other iteration remains stop and when 1st fold is under testing then other folds are under training. In this way whole testing and training over the selected dataset will be processed.

Dataset preprocessing is nothing but manage the data to use in another procedure. It is an important part of data mining, because machines cannot understand real-world data. It seems insufficient and inaccurate for the machines.There are two steps to be followed in data prepossessing. First basic analysis will be held, after that missing values will be checked, and handle it to complete the incomplete data. Here maybe we can find duplicate data, we need to remove them to help the removal of stop word. We also need to remove the hashtags symbols, because of so many unnecessary symbols this type error will occur. Fortunately, my selected data is free from the problems, no missing values. In machine learning many issues occur for the redundant data. To get prevention from this issue we need to remove those duplicate data and stop words.



If the dataset remains missing values, it may affect the prediction of the model and is preform wrong results. We can also fill those missing values using machine learning, using machine learning techniques we can find those values.

Tools

Here I have used 2 classifier techniques. Naive Bayes and Logistic regression or linear regression. And to comparative analysis there are metric are calculated those are accuracy, precision, recall, and F1-score. This measurement will help to find out the best classifier we can use for our model.

To evaluate the performance of the classification model there are two metrics that have been used, training and testing. This is important to note that we have to score better in precision, recall, F1 score also, not only accuracy. Because only the high accuracy is not enough to prove that the model will give the best outcomes.

These are the following formulas to calculate these metrics.

1. Precision = TP / (TP + FP)
2. Recall =TP / (TP + FN)
3. F1 score = 2*(Precision * Recall) / (Precision + Recall)
4. Accuracy = (TP + TN) / (TP + TN + FP + FN)

For the node activation Sigmoid function is used, that is shown below,

$$f(x) = 1 / (1-e^{-x})$$

Results analysis

Naive Bayes classifier outcomes are shown below.

Outcomes from Naive Bayes.

No.	Dataset	Term frequency-IDF	Bag of words
1.	Balance dataset	Training: 92.55	Training: 92.00
		Testing: 83.54	Testing: 84.11

It can be shown that the Naive Bayes classifier performed well and obtained training accuracy and test accuracy 92.00 %and 84.11% respectively. I have shown the confusion matrix below.

No.	Precision	Recall	F1-score
0	0.96	0.97	0.97
1	0.75	0.70	0.72
2	0.94	0.98	0.95
3	0.67	0.66	0.64
accuracy			0.88

Classification report of Naive Bayes

Here we can see that the score of recall and precision is around 67 to 95 but row number 4 and 5 have shown the 99 to 100% recall. This is really a good result.

Logistic regression is fully supervised learning. It always shows the output in binary form is like yes or not. The outcome is shown below.

Outcomes from logistic regression

No.	Dataset	Term frequency-IDF	Bag of words
1.	Balanced dataset	Training: 91.64	Training: 91.53
		Testing: 86.62	Testing: 87.33

From the table we can see that the result from the logistic regression is not as expected. The highest score of training is 91.64% and the testing result is 86.62%. Below I have shown the confusion matrix of logistic.

No.	precision	recall	F1-score
0	0.98	0.97	0.97
1	0.76	0.72	0.74
2	0.91	0.97	0.96
3	0.74	0.64	0.67
accuracy			0.88

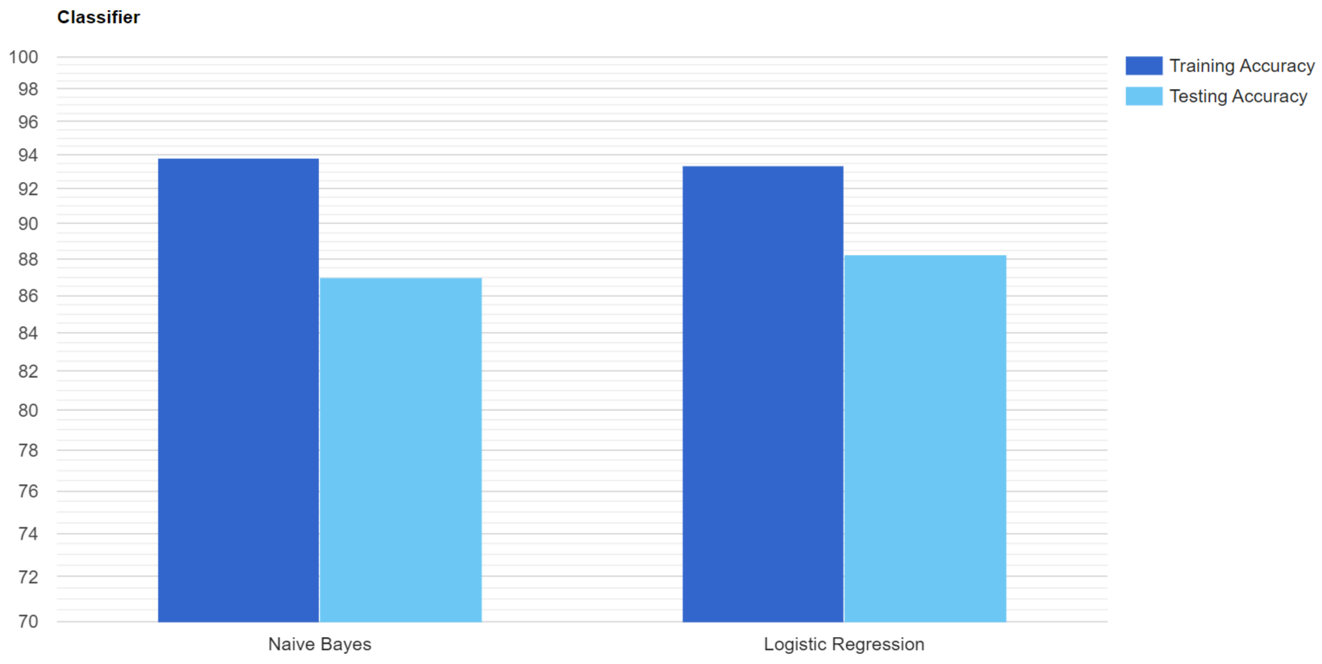
Classification report of logistic regression

Final analysis

The selected balanced datasets are built by these two classifiers, Naive Bayes and logistic regression. Below I've shown the result accuracies.

Obtained accuracy of these classifiers.

No.	Classifier	Training Accuracy	Test Accuracy
1	Naive Bayes	93.84%	87.01%
2	Logistic regression	93.37%	88.27%



Comparing both classifiers, we can see that the Naïve Bayes classifier presented a better score than the logistic regression. Both training and testing accuracy is higher. And other metric results recall, accuracy, and F1 score is greater than the logistic regression.

CONCLUSION

In the old days we handled several problems which occurred on cloud platform, but as we are stepping upwards in today's world, we are seeing a lots of more challenging problems on this platform only because of using more and depending our technology on this platform so we must upgrade our thinking and work for better solutions. This inspiration made us more focused on using machine learning technologies in this field. My study elaborates that machine learning algorithms can provide better performance and work effectively for classification of bugs. Though it has large amount of data on cloud computing, ML techniques can handle accurately and identify the class of bugs with higher accuracy. This procedure improves the resolution of bud detection. Thus, we can say machine learning techniques for cloud platforms provide an effective way to work in future research, which makes us satisfied that the bug classification problems can be resolved with positive potential in cloud computing.

FUTURE WORK

There are so many fields in this topic to be improved in future. Transferring the knowledge of machine learning techniques from previous reports to new reports, which can make it a more time-saving and strategic way to work efficiently. It will provide a better environment in cloud computing systems. There is technique called domain adaptation which can be used to classify bugs when training and testing data mismatch occurs. Another feature calls visual feedback, which can be used in future. We don’t need to write bug reports; it will serve a visual display of the results.

REFERENCES:

1. Thota, M.K.; Shajin, F.H.; Rajesh, P. Survey on software defect prediction techniques. Int. J. Appl. Sci. Eng. 2020, 17, 331–344.
2. Umer, Q.; Liu, H.; Sultan, Y. Emotion Based Automated Priority Prediction for Bug Reports. IEEE Access 2018, 6, 35743–35752.
3. Waqar, A. Software Bug Prioritization in Beta Testing Using Machine Learning Techniques. J. Comput. Soc. 2020, 1, 24–34.
4. Hai, T.; Zhou, J.; Li, N.; Jain, S.K.; Agrawal, S.; Dhaou, I.B. Cloud-based bug tracking software defects analysis using deep learning. J. Cloud. Comp. 2022, 11.
5. Catolino, G.; Palomba, F.; Zaidman, A.; Ferrucci, F. Not all bugs are the same: Understanding, characterizing, and classifying bug types. J. Syst. Softw. 2019, 152, 165–181.
6. Kukkar, A.; Mohana, R.; Nayyar, A.; Kim, J.; Kang, B.-G.; Chilamkurti, N. A Novel Deep-Learning-Based Bug Severity Classification
7. Technique Using Convolutional Neural Networks and Random Forest with Boosting. Sensors 2019, 19, 2964.