MACHINE LEARNING APPLIED TO SOFTWARE TESTING

Manikkannan D[1], R. Bhumi Devi[2], V.V. Nithyaa Shri[3], B. Bala Barathy[4]

[1]Assistant Professor/CSE, [2, 3, 4]B.Tech-Student/CSE, SRM Institute of Science and Technology

***Abstract* - Software testing is an integral part of developing software which aims to identify the errors and faults. The budget involved in developing the software exceeds if the fault is not detected and corrected at the right time. Also, testing is error prone. In order to overcome such problems, an attempt is made to automate software testing by applying machine learning algorithms which will minimize the error and the cost involved in software testing. Software testing activities include test case generation and oracle testing among others. Linear regression, decision tree and random forest algorithms are the machine learning algorithms that are used.**

***Index Terms* - software testing, machine learning, linear regression, decision tree, random forest**

I. INTRODUCTION

Software testing is a process which checks whether the system under test works according to the expected behaviour. It includes execution of software component towards one or more properties. Software testing activities can be categorised as: white box and black box testing. White box testing involves testing of software's internal structure design encoding. Black box testing depends upon software requirements, specifications and design parameters. In black box testing, the input and output of the software is considered rather than the architecture of the software. Testing requires more time and resources to generate good quality output. Drawback of manual testing is that it often provides least accuracy. In order to minimize the limitations, software testing activities are automated by applying machine learning algorithms.

The software testing activities to which machine learning algorithms are applied are test case generation and oracle testing. Test case generation is a procedure for generating test suites for particular application. Test cases are considered based on the functionalities of an application. So it changes from one application to another. The idea of test case generation is to check the output against expected results. Oracle testing is the mechanism to check whether the software under test passes or fails the test case. If the actual and the predicted output are same then the oracle is said to pass the test. Oracle testing comprises of comparing the output for given test- case input to the output that the oracle determines. Generation of correct output for given input is referred to as test- oracle problem. There are different types of test-oracles. Specified oracle includes formal specification, model- based approach to generate test- oracles. One of the major setbacks of specified oracle is that formal specification model cannot recognize every behavior as it depends on abstraction. A derived test oracle derives the information from the artifacts involved in the system to compare the correct and

incorrect behavior. Pseudo- oracle is a type of derived oracle which allows using separate program to take same inputs. It checks whether the outputs can be compared without any problem. Implicit oracle uses implied information. It detects the unwanted behavior to determine the problem. One of the limitations of implied oracle is that it is prone to dependencies in environment which leads to false positive results. There might be situations in which any of the three oracle approaches cannot be used. In such cases, human input can be used as both qualitative as well as quantitative approach to generate test-oracles. Quantitative approach collects right quantity of information on system under test to decide whether to go for fit or release of the software. Qualitative approach mainly focuses on input data representation and desired output of the software under test.

Machine learning is a mechanism of learning algorithms to create rules. The process of developing a machine learning model involves feeding the training dataset to the algorithm. The algorithm fosters new set of rules depending upon the inferences included in given data. Generation of new algorithm is termed as machine learning model. Different training data can be used so that same learning algorithm can initiate different models. The machine learning algorithms can be categorised into three types: supervised learning, unsupervised learning and reinforcement learning.

Supervised learning involves usage of labelled data and required output. Labelled data described helps the algorithm to identify the rules based on which classification can be made. Unsupervised learning feeds unlabelled data to learning algorithm, where the algorithm needs to identify the patterns involved in the input. Reinforcement learning enables algorithm to communicate with the system to give back feedbacks in terms of rewards and punishments. In this paper, we will be using machine learning algorithms such as linear regression, decision tree and random forest.

Linear regression is used to determine the linear relationship between dependent variable and one or more independent variables. It is used to fit the predictive model to a set of observed data. Regression is an unpredicted change that occurs whenever the code changes. Regression happens when new features are implemented. Regression can be used to automate testing activities in order to deal with efficiency and speed constraints. Decision tree is used to classify instances by categorization from the root to leaf node in order to perform classification of instances. It can also be used to predict continuous values. When decision tree is evaluated, we must partition the data into training and testing dataset. Train- test process is repeated using cross- validation which determines the average success rate. It establishes whether testing activity will create high or low accuracy outputs. Random forest is ensemble learning approach used especially for performing classification, regression and construction of decision trees. It generates individual tree which in turn creates a forest of trees by employing bagging and random features. The prediction made will have more accuracy than provided by the individual tree. Accuracy of the model is generally low during the testing phase. Over-fitting arises when the model learns

the training data to a level such that it negatively affects the performance when modelling with the new data. Random forest overcomes these difficulties posed during testing.

The remainder of the paper is as follows. Section II presents the related work. The system design is provided in Section III. Section IV describes about the methodology followed. The results are discussed in Section V. The paper is concluded in Section VI with the summary and future scope.

II. RELATED WORK

Different machine learning algorithms have been proposed to automate software testing in order to minimize the cost and error in software testing.

Pranali Mahadik et al. [4] proposed three techniques that can be applied to automate test case generation, namely random based, search based and symbolic execution. Random based approach provides a fast and easy to generate large number of test cases for small and simple programs. This technique fails if it does not use the available information to generate test case. Search based approach uses optimization technique and the main drawback with this technique was it creates large search space. Symbolic execution technique tries to create algebraic expression and maps it with symbolic path constraint. As a large number of paths need to be analyzed, symbolic execution technique has scalability issues.

Rajesh Kumar Sahoo et al. [3] proposed a system in which test data are generated using random based approach and then it is optimized it by applying algorithms such as artificial bee colony, particle swarm optimization and harmony search algorithms. Withdrawal operation in bank ATM is the case study on which test case generation and optimization are applied. On comparing the results of all the algorithms that are used, it was found that bee colony algorithm produces optimal result with more accuracy.

Although automating software testing has increased the effectiveness of testing and reduced the cost, there is however need of manual work. Dionny Santiago et al. [2] proposed an approach where artificial intelligence and machine learning technologies are used to generate test cases which will learn directly from the human testers. Perceiving web application state and text generation for web app testing were the two applications to which machine learning algorithms were applied. Random forest, J48 decision tree, k- nearest neighbor, support vector machine and Bayesian network algorithms were used for generating and executing test cases for perceiving web application state while long short- term memory algorithm was applied for generating and executing test cases for text generation for web app testing. The main problem was to raise the level of abstraction.

Vinicius H.S. Durelli et al. [1] proposed a system where machine learning algorithms are applied to automate software testing activities which include oracle construction, test- case generation, refinement and evaluation and predicting the cost of testing related activities. The algorithm that is used varies depending upon the software testing activity. The

most commonly used algorithms include decision tree and artificial neural network. The main drawback was found to be the need of substantial amount of dataset and the quality of the data, which plays a key role in the functioning of machine learning algorithms.

III. SYSTEM DESIGN

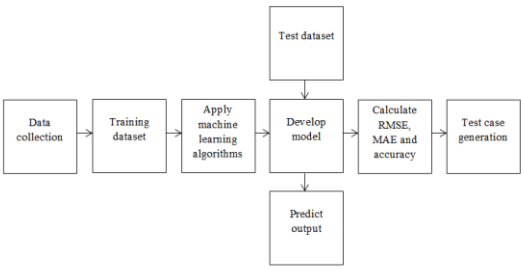

Fig. 1. Block diagram

Fig.1 demonstrates the process involved in developing the model. The block diagram represents the key steps involved in the development of the proposed model. The key phases are data collection, developing the model, evaluation and test case generation.

IV. METHODOLOGY

A. Data Collection

Dataset selection is an important task because the machine learning algorithm that is to be applied depends on it. A dataset for regression analysis is selected. The dataset that is used in this proposed model predicts the river water quality. The features are the five indicators of river water quality which are dissolved oxygen, ammonium ions, nitrite ions, nitrate ions and the biochemical oxygen demand, all measured in mg/cub dm. The target value is water quality. The dataset is saved in .csv format.

B. Regression Model

After the collection of dataset, the features and the target values are separated. Also, the entire dataset is split into training and test dataset where training dataset comprises of 70% of the dataset and test dataset comprises of 30% of the dataset. The model is trained on the training dataset by applying linear regression, decision tree and random forest algorithms. After training, the model is allowed to predict the output for the test dataset.

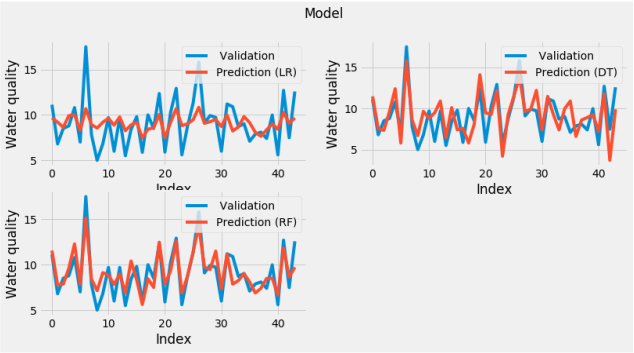

Fig. 2. Comparing the observed and predicted values of the model trained using linear regression, decision tree, random forest

C. Evaluation

The model is then evaluated by calculating root mean square error, mean absolute error and accuracy.

Root mean square error is used to compute the difference between the predicted and the observed value. It is calculated as square root of quadratic mean of differences between predicted value and observed value.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

where,

$\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n$ are predicted values

$y_1, y_2, \ldots, y_n$ are observed values

n is the number of observations

Mean absolute error is used to measure the difference between two continuous variables. It is calculated as the average magnitude of errors in prediction without considering its direction.

$$\text{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

where,

$\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n$ are predicted values

$y_1, y_2, \ldots, y_n$ are observed values

n is the number of observations

Accuracy determines how well the model predicts the correct value for a given observation.

D. Test Case Generation

For each feature, a random value is generated based on its range and is used as the input. The algorithm with high accuracy and least error is used to predict the output for this randomly generated input. This combination of input along with the corresponding output forms a test case.

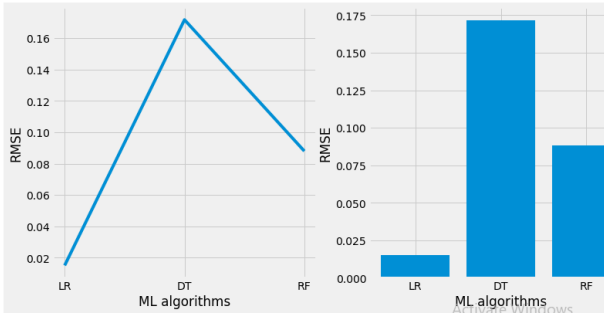

Fig. 3. RMSE of linear regression, decision tree, random forest

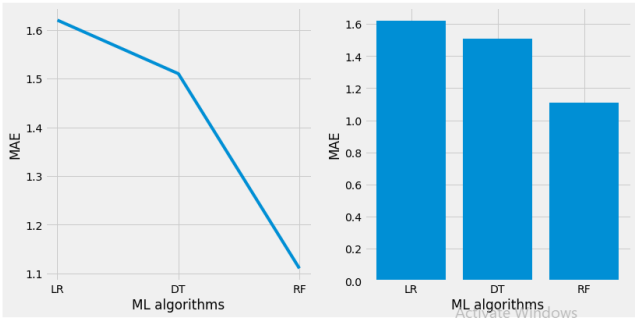

Fig. 4. MAE of linear regression, decision tree, random forest

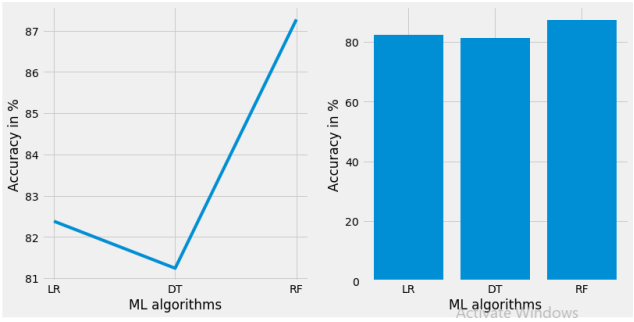

Fig. 5. Accuracy of linear regression, decision tree, random forest

V. RESULTS

The root mean square error of linear regression, decision tree and random forest is shown in Fig. 3. It can be observed that linear regression has the least root mean square error of 0.015 while decision tree has the highest root mean square error of 0.171. The root mean square error of random forest is 0.087.

Fig. 4 shows the mean absolute error of linear regression, decision tree and random forest. It can be observed that random forest has the least mean absolute error of 1.11 degrees while linear regression has the highest mean absolute error of 1.62 degrees. The mean absolute error of decision tree is 1.51 degrees.

The accuracy of linear regression, decision tree and random forest is shown in Fig. 5. It can be observed that random

forest has highest accuracy of 87.27% while decision tree has the least accuracy of 81.24%. The accuracy of linear regression is 82.38%.

VI. CONCLUSION AND FUTURE WORK

This paper aimed to automate software testing by applying machine learning algorithms to minimise the cost and error involved in testing. The testing activity determines the machine learning algorithm that is to be used. Linear regression, decision tree and random forest algorithms were used for test case generation. Random forest algorithm performs better when compared with other algorithms with respect to root mean square error, mean absolute error and accuracy and hence is used for test case generation.

In future, random forest algorithm can be explored further by applying it to other testing activities.

REFERENCES

[1] Vinicius H. S. Durelli, Rafael S. Durelli, Simone S. Borges, Andre T. Endo, Marcelo M. Eler, Diego R. C. Dias, Marcelo P.Guimar Aes, "Machine Learning Applied To Software Testing: A Systematic Mapping Study", IEEE Transactions on Reliability, vol. 68, issue 3, pp. 1189- 1212, 2019, doi: 10.1109/TR.2019.2892517

[2] Dionny Santiago, Tariq M. King, Peter J. Clarke, "AI- Driven Test Generation: Machines Learning From Human Testers", 2018, doi: 10.25148/etd.fidc007028

[3] Rajesh Kumar Sahoo, Deeptimanta Ojha, durga Prasad Mohapatra, Manas Ranjan Patra, "Automated Test Case Generation And Optimization: A Comparative Review", International Journal of Computer Science & Information Technology (IJCSIT), vol. 8, no. 5, Oct. 2016, doi: 10.5121/ijcsit.2016.8502

[4] Pranali Mahadik, Debnath Bhattacharyya, Hye-jin Kim, "Techniques For Automated Test Cases Generation: A Review", International Journal of Software Engineering and its Applications 10(12):13-20, Dec. 2016, doi: 10.14257/ijseia.2016.10.12.02