Chapter 6

Conclusion/
Future Scope
6.1 Conclusion

6.1.1 Summary

The objective of the software bug prediction is tracing of bug constituents in software well before software testing. The bug prediction techniques were applied in all the software modules of CK and OO metrics. Bug prediction results in decreased cost of development costs, time, increased customer satisfaction and more reliable software. Therefore, bug prediction techniques are considered significant in order to accomplish important to achieve optimum quality of and refrain from repeating past mistakes. The dataset used in this paper is from the Promise Repository, which is well-recognized and a publicly accepted repository. The experiment was conducted using software modules, such as Ant, Ivy, Tomcat, Berek, Camel, Lucene, POI, Synapse and Velocity using Feature selection techniques: Boruta, regsubsets and FSelector (Random Forest, Information Gain, Linear Correlation, and Rank Correlation). The objective of the Research was achieved by locating the optimal software metrics, optimal machine learning models and preventive measures that can be used to prevent defects in future software version releases. The future scope of the research is also explained in brief towards the last section of the conclusion.

6.1.2 Conclusion Drawn

One of the objectives of the research is to find out the most optimal software metrics for the software bug prediction. It is analyzed from the result that the most optimal metrics are RFC, LOC and WMC and the least significant is NOC and DIT.

The main objective of the research however is to locate the best machine learning model for software bug prediction. To get the most optimal machine learning model various performance parameters like accuracy, mean square error, R-Squared and correlation were computed. The performance parameter was calculated by taking all the 21 CK_OO metrics as defined in Table 1.1 and with only the most relevant metrics selected by Feature Selection method.
The conclusion derived after the experiment and result analyses to predict the software bug are as follows:

(i) It is always desirable to apply feature selection method and to work with only the most important features/metrics, as more complex the data, more is the likelihood of defects, which will in turn lead to a reduced efficiency of the system and more time consumption. The Table 6.11 to 6.14 shows that the result was either better or negligible when only optimal software metrics was applied as compared to all the software metrics in machine learning model to predict the software bug.

(ii) The framework has been proposed by applying different machine learning model using all the metrics and with only selected metrics. It was analyzed that the Support Vector Machine model (SVM) outperformed as compared to other machine learning models having maximum accuracy (83%) and minimum mean square error (0.7%) for predicting the software bug.

6.1.3 Measures to prevent defect in software release

i. The earlier the defect is detected, the cost involved is reduced and the resources are fully utilized. Moreover it becomes much easier to rectify the defect during the initial stage. The Figure 6.1 also justifies that it is better to detect the bug at an early stage in software development life cycle phases.

ii. Complexity metrics are better predictors of fault potential in comparison to other well-known historical predictors of faults, i.e. prior modifications and prior faults.

iii. Self-review the code has a major contribution in preventing the software defect.

iv. Proper Defect Tracking system needs to be implemented.

v. Files that have anti-patterns tend to have a higher density of bugs than others as anti-patterns can increase the bugs in the future. Anti-patterns can be removed from systems using refactoring.
vi. Providing software developers with bug categorizing recommendations helps and supports them in cost/benefit analysis.

6.2 Future Scope

Research is a continuing process and there is always a scope of enhancement and innovative learning. To explore the finest machine learning model for software defect prediction, the experiment was conducted using six types of machine learning models. In future, more dimensions can be carried out to reduce the software defects in a system.

(i) Other types of machine learning model for regression like Cubist, M5p, FOBA etc can be applied and compared to discover the best machine learning model for software defect prediction.

(ii) New models can be proposed to enhance the accuracy and to minimize the software bug in a system

(iii) A new framework can be proposed if the dataset are available in a requirement phase and it can be compared with the testing phase to locate the software bug/defect in a system.

(iv) The data mining techniques are applied on historical software data by taking the dataset from the publicly available domain for prediction of software defect. For future work, datasets can be taken from some segment of the industry and can be compared with the historical dataset to analyze the result.
Figure 6.1: Defect Percentage in SDLC Phase