8. CONCLUSION AND FUTURE WORK

8.1 CONCLUSION

The ability to identify software defects in the early stages of development is extremely crucial for minimizing the cost and improving the overall effectiveness of the testing process. A majority of faults in a software system are found in a few of its own components. Knowing the causes of possible defects as well as identifying the process areas that might need attention could save money, time and effort. A low-cost method for defect analysis is learning from past mistakes to prevent future ones. The defects occur in software due to various reasons. If a developer or a tester can predict the software defects properly and identify those, it can reduce the cost, time and effort at a later instance, which needs to be avoided. To mitigate the defects we need to identify potential fault-prone code, estimate the number of faults and discover the possible causes of faults.

There are two kinds of models in the literature: defect prediction and defect identification. There were different models used in the past like regression model estimations provide the exact number of faults but require lots of information and are difficult to interpret. So it is very difficult to use especially at the beginning of a project when too little information is available. On the other hand, classification models that predict possible faults are useful in the earlier stages, but they do not give the actual number of faults.

It is required to reduce the cost of software development both in the developing and maintenance phases. The proposed methodology tries to help in identifying modules that require immediate attention and because of that the reliability of the software is vastly improved by handling high priority defects first. The goals undertaken in this research are

- To propose a data mining approach to the software defect prediction process based on software metrics;
- To present architecture framework for software defect prediction process model based on the available metrics;
- To develop and test a method to predict software defects from historical data; and
- To compare different classification mining models with respect to software defect prediction performance.
We have attempted to solve the Software defect prediction problem using different Data mining (Classification) algorithms. There are several research streams in defect prediction which uses different evaluation measures for defect prediction across the literature. Lots of these methods require preprocessing of the data used. Most of these models were not actually prediction models but fitting models that investigated the correlation between metrics and the number of defects. The limitations in using these models are given below:

- the prediction model could be useful only after the product release for the purpose of quality assurance.
- it was not possible or difficult to build a prediction model for new projects or projects having less historical data
- Are the defect prediction models really helpful in industry?

In order to improve the efficiency and quality of software development, we can make use of the advantage of data mining to analyze and predict large number of defect data collected in the software development. We reviewed the current state of the art in software defect management, software defect prediction models and data mining technology. In this context the widely used metrics are source code and process metrics which deals with the complexity in the code and complexity in the process used. Researched and analyzed several software defect prediction methods based on data mining techniques and specific models (NB, Logistic, PART). Then proposed an ideal software defect management and prediction system.

Comparing different classification algorithms based on the requirements we noted that NB and Logistic algorithms were better in most of the requirements for defect prediction. PART and J48 were better in moderate number of requirements and OneR and JRip were the least. From these results, we can infer that a data preprocessor/attribute selector can play different roles with different learning algorithms for different data sets and that no learning scheme dominates, i.e., always outperforms the others for all data sets. This means we should choose different learning schemes for different data sets, and consequently, the evaluation and decision process is important.

Software quality and maintenance are the main areas in recent research approaches. Conventional methods used one pass algorithm for finding association rules with minimum support and minimum confidence. We developed a predictive classification algorithm to decrease cost of the software testing and cost estimation for software
application process. With this proposed software application model format, we increased assurance of software quality for all the properties of the data set represented in association rule mining. The implementation also supports the conclusion that a sufficient number of rules are a precondition for the high prediction accuracy to be obtained in the context of defect isolation effort prediction.

Since initial and early detection of defective software components help software experts to optimally benefit from time and resources, our model increased reliability and improved software control processes. We proposed a hybrid method to improve the precision of defective components prediction. The proposed method benefits from support vector machine algorithm, multi-layer perception neural networks and evolutionary algorithms. In this method, a new learning approach was used which uses multi-layer perception neural networks algorithm. We could achieve a significant increase in the network efficiency because of the hybrid method. The efficiency of the developed hybrid method was compared against 11 statistical and machine learning models using 3 NASA datasets. The results showed that the proposed method has higher efficiency as compared to other models. Achieving acceptable efficiency with both large and small datasets is one of the advantage of the proposed hybrid method.

A comparison framework is proposed for comparing the performance of classification algorithms in the software defect prediction. The data set of software metrics used for this research is acquired from NASA’s Metrics Data Program (MDP). The existing classification algorithms are evaluated initially based on these Datasets. The framework is comprised of 9 NASA MDP datasets, 10 classification algorithms, 10 fold cross validation model, and AUC accuracy indicator. Friedman and Nemenyi are used to test the significant difference among the AUC models and our proposed model. The experimental results shows that the SP-LR (our method) performed better in most of NASA MDP datasets. NB, NN, SVM and k* also perform well, and actually there is no statistically significant difference among them. Decision tree based classifiers tend to underperform, as well as LDA and k-NN.
8.2 SCOPE FOR FURTHER RESEARCH

- Multi-Variant clustering based classification can be used.
- Future studies could focus on comparing more classification methods and improving association rule based classification methods.
- Furthermore, the pruning of rules for association rule based classification methods can be considered.
- Hybrid method developed for defective component prediction can be extended to give a more prominent role to evolutionary algorithms so that the prediction accuracy could be increased.