Chapter 2

Review of Literature
This chapter presents a detailed study of the software defect prediction using data mining technique.

Chidamber et.al. (1994) suggested that software development plays a dominant role in the field of information and technology and there is a continuous effort by the managers to improve the process in the software development area. To accomplish the task, several fresh approaches are implemented wherein object-oriented is the most prominent one. They have developed six design metrics. They have collected the data from the real world environment C++ and small talk, which is a popular object-oriented environment to execute the independence of these metrics. Managers may use these metrics to express the feasibility of the metrics in raising the demand for software measure and the process improvement. The object-oriented metrics suite can assist the managers to categorize in which areas of applications, thorough testing and redesigning of the system is required. The main objective of the organization is to reduce the development cost and time and these metrics suite will prove to be beneficial in measuring the process of software development. [28]

Mannila et al. (1996) mentioned that data mining and knowledge discovery in database are the most important tools to discover valuable information from huge datasets. He clearly indicated in his paper that in the 1990’s, data mining emerged as one of the most significant areas for the research and development, yet there was a total lack of analysis method for large datasets. [29]

Rahm et.al. (2000) in his paper addresses the issue of data cleaning which is the foremost part of extraction, transformation, loading (ETL) process in a data warehouse. Data cleaning is required to improve the quality of the data as it removes the inconsistency and errors in the data. There are various types of tools that are available to clean the data, but there are times when a major portion of the data needs to be clean manually that are difficult to write and maintain. The research in this area of data mining has been very negligible. Though a large variety of tools are available, however there is a complexity in the cleaning process. Therefore there is a requirement for more research activity to be done on
the design and implementation. Data cleaning is not only required in a data warehouse but in web-based information also. [30]

Adderley et al. (2007) believed that data mining includes a vast range of techniques, which provide supplementary information in an improved comprehension and construing the data in the areas of prediction, forecasting, decision support, and estimation. Data mining includes a broad range of methods such as neural network, rule induction, data visualization to observe the data within the computer system and classify operational issues by discovering useful and unidentified information. [31]

Hassan (2008) in his paper has discussed at length about the software repositories like Historical Repositories, Run-time Repositories and Code Repositories which can guide the decision processes in modern software projects while at the same time, can reveal useful, important patterns and information. [32]

Zimmermann et al. (2008) has discussed in his paper that systematic mining exposes the modules which are most prone to defects and failures. Software defects are the major cause for the degradation of the quality of the product. Bug repositories are a major source of database, which retains the history of success and failure. Bug database is a rich source of information for software failures. [33]

Hassan (2010) explored more research problems in mining software repositories and software intelligence, which further give future directions (i) finding the technique to automate and extract information from the repository (ii) to mine vital information from these repositories. He also suggested that innovative software bug prediction models need to be designed, effective software defect metrics need to be synthesized and given them as inputs to various data mining techniques for extraction classified information to predict the software defects in new software versions and also further developed methods are needed to reduce software cost overruns. [34]

Hassan et al. (2010) investigated that stored software evolution supports various aspects of software development within the industrial software development
process. To produce high quality software systems, researchers are using data mining techniques to explore the valuable data to better manage the projects and develop within time and budget. [35]

Kamei et.al.2010 describes that bug prediction technique is a significant area in software engineering field for the last three decades. The prediction result is an important measure for the software developer to control the overall software development process. The bug prediction techniques are often used to help allocate software quality assurance efforts and if applied accurately can reduce the software defect. [36]

Kursa (2010) mention his paper that Boruta is one of the important feature selection packages to locate the relevant element from a large dataset. The Feature selection technique not only improves the speed of the algorithm, but also increases the accuracy of machine learning algorithm. Boruta uses a wrapper algorithm which uses the Random Forest classification algorithm and is based on the principle of iteration and rejects the features which are irrelevant. The Wrapper algorithm is better than the filter method because among the given features, there is no direct correlation. The Wrapper method classifier is used as a black box, returning a feature ranking. The R package Boruta is available at http://CRAN.R-project.org/package=Boruta). When the R package Bouruta is run, its shows the entire confirmed variable and the rejected variable in a dataset. When the box plot is drawn Green, Blue and Red box plot represent a Z-scores of confirmed, minimal or average rejected attribute respectively. [37]

Azeem and Usmani(2011) in their paper has pointed out that Software bug repository is the main resource for fault prone modules. There are different data mining algorithms which are used to extract fault prone modules from these repositories. The team for Software development tried to increase the software quality by decreasing the number of defects as much as possible. In this paper different data mining techniques are discussed for identifying fault prone modules as well as compare the data mining algorithms to find out the best algorithm for defect prediction. [38]
Thomas (2011) proposed the usage of statistical topic models such as Latent Dirichlet Allocation (LDA) to automatically discover structure in software repositories since these repositories contain unstructured and unlabeled text that is rather difficult to analyze with conventional techniques. This paper addresses the challenges of applying topic models to software repositories. [39]

Wang et.al. (2011) indicates that in data mining, feature selection technique plays an extremely essential role. Through the usage of feature selection method, software defects and risk estimation may improve in the classification model and the redundant and irrelevant data gets removed. They have applied six filters bank rankers in three large software projects. They have built a classification model in SVM, NB, KNN, LR, MLP learners. They have calculated the AUC performance metric. They concluded that the data sets play a significant role in evaluating the results of performance of rankers i.e. feature selection technique. They also concluded that in future tasks, the experiment can be conducted on a different dataset in different domains like software engineering and other application domains. They also mentioned that in their future work, they would consider the strident and disproportion data.[40]

Dhiauddinet.al. (2012) has proposed the prediction model for defects in system testing. The main purpose of this prediction model is to make a quality indicator of the system whenever any system is entering into a testing stage. They have applied regression analysis in a selected metrics to predict the defect in a testing phase. They have also explained that to declare the model as fine, the prediction should fall between the maximum and minimum range. The analysis of the P-values must be less than 0.5. The value of R squared must be more than 85%. The Adjusted R squared must be more than 85%. [41]

Liao et.al. (2012) have done the literature survey from 2000 to 2011 to determine how data mining techniques and their application have developed in the past decade. They have concluded that data mining technique is gaining popularity in problem-centric and expertise-oriented applications. They also suggested that data mining technique can also be used as an alternative method in social science
methodology. Data mining techniques are based on the principle of affording novel comprehension and capacity to continue change, which will be the core of data mining applications in future.[42]

Suresh et.al.(2012) describe metrics as a unit of measurement in software engineering in which a system or process possess a given attribute. There are several metrics that are available from the conventional to the object-oriented methodology. Traditional metrics, like cyclomatic and size complexity are used for software complexity and object-oriented complexity like Chidamber and Kemerer metric suites are used for system reliability. Software metrics are beneficial to evaluate the complexity, software reliability and fault proneness of the system. CK metrics are the best indicators for fault proneness. Regression analyses are utilized to determine the relationship between the values of the metrics and bugs associated in the class. The DIT and NOC cannot be used for fault detection. LOC, LCOM and WMC are the best indicators for the system reliability. [43]

Jain et.al.(2013) said that data mining addresses all the techniques and processes involved in discovering interesting patterns hidden inside large data sets, which help in the decision making process.[44]

Goyal et.al. (2013) in their paper has clearly stated that the combinations of metrics are better as there are no solo set of metrics exist, which can be applied to projects for fault prediction model. They develop an efficient predictive model involving interaction metrics. They stated in their paper that the dimension of the bug database also gets reduced through combinational interaction among metrics before applying stepwise regression and also develop mechanism for finding infeasible sub-set of interacting terms. The Regression technique is used to analyze the relationship between the dependent or response variable or independent or predictor variables. A total of 17 metrics derived from the dataset taken were used in isolation and in combination. In their paper, single version approaches of bug prediction were taken and have considered 6 CK (Chidamber and Kemerer) metrics and 11 OO (Object-Oriented) metrics, for five software
modules i.e. Eclipse jdt core, Eclipse pde UI, Equinox Framework, Lucene and Mylyn. A Step-Wise Regression (SWR) was applied and further through feature selection method, the metrics was reduced and the value of adjusted $R^2$ fairly increased with the interaction of the metrics. In this paper the scope was limited action effects in the context of linear regression.[45]

Singh and Salarai (2013) in their study has correctly pointed out that there has been a tremendous growth in the demand for software fault prediction during recent years. In their paper, Levenberg-Marquardt (LM) algorithm based neural network tool is used for the prediction of software defects at an early stage of the software development life cycle. It helps to minimize the cost of testing which minimizes the cost of the project. The methods, metrics and datasets are used to find the fault proneness of the software. The study used data collected from the PROMISE repository of empirical software engineering data. This dataset uses the CK (Chidamber and Kemerer) OO (object-oriented) metrics. The accuracy of Levenberg-Marquardt (LM) algorithm based neural network is compared with the polynomial function-based neural network predictors for detection of software defects. Their results indicate that the prediction model has a high accuracy.[46]

Okutanet.al. (2014) also explains in their paper that there are a large number of software metrics available for software defect prediction. It is always advisable to work with smaller sets of important metrics and focus only on the important metrics for the software defect prediction. They have used Bayesian network to check the influence among software metrics and defect proneness. In combination with the metrics used in Promise Repository; they have defined two more metrics, Source Code Quality Metrics (LOCQ) and a Number Of Developers (NOD). For their experiment, they have selected nine datasets from the Promise Repository. They derived that RFC, LOC, and LOCQ are more effective to defect proneness whereas NOC and DIT are less effective and unreliable. Their future work will include other software metrics and process metrics to determine the best metrics used for defect prediction and emphasis has been given to work with a smaller set of software metrics.[47]
Palatse and Nandedkar (2015) in their research have found that Software testing plays a vital role in software development especially when the software developed is mission, safety and business critical applications. Prediction of a modules info fault-prone and non fault prone prior to testing is one of the cost effective technique. Predicting a safe module as faulty increases the cost of projects by more cautious and better-test resources allocation for those modules, whereas prediction of faulty code as fault free code end up in under-preparation and may leave modules untested this may cause accidental failure and lead towards massive loss . They have presented a novel fault prediction technique that reduces the probability of false alarm (pf) and increases the precision for detection of faulty modules. The general expectation from a predictor is to get very high probability of false alarm (pf) to get more reliable and quality software product. Software Reliability is becoming an essential attribute of any software system. It is a significant factor in software quality since it quantifies software failures. Software defect prediction models have gained considerable importance in achieving high software reliability. Software defect prediction model helps in early detection of faults and contribute to their efficient removal and producing a reliable software system. This paper presents the survey on existing data mining techniques used for prediction of software defects.[48]

Parameswari (2015) in his paper has described that Software Defect Prediction plays an important role in the field of software quality and software reliability. The costs of finding and correcting software defects have been the most expensive activity during both software development and software maintenance. The accurate prediction of defect prone software modules can help the software testing effort, reduce costs, and improve the software testing process by focusing on fault-prone module. Various metrics in software like Cyclomatic complexity, Lines of Code have been calculated and effectively used for predicting faults. Techniques like statistical methods, data mining, machine learning, and mixed algorithms, which were based on software metrics associated with the software, have also been used to predict software defects. Different data mining techniques are discussed for identifying fault prone modules as well as by comparing the data
mining algorithms to find out the best algorithm for defect prediction. [49]

Prasad, Florence and Arya (2015) in their study has depicted that Software quality is a field of study and practice that describes the desirable attributes of software products. Software quality metrics are a subset of software metrics that focus on the quality aspects of the product, process, and project. The software defect prediction model helps in early detection of defects and contributes to their efficient removal and producing a quality software system based on several metrics. The main objective of paper was to help developers identify defects based on existing software metrics using data mining techniques and thereby improve the software quality. In their paper, various classification techniques are revisited which are employed for software defect prediction using software metrics in the literature. [50]

Vashisht, Lal and Suresh chandar (2015) in their research paper has pointed out the fact that number of approaches has been proposed for effective and accurate prediction of software defects, yet most of these have not found widespread applicability. The main objective in this communication is to provide a framework which is expected to be more effective and acceptable for predicting the defects in multiple phases across software development lifecycle. The proposed framework is based on the use of neural networks for predicting defects in software development life cycle. Further, in order to facilitate the easy use of the framework by project managers, a software graphical user interface has been developed that allows input data (including effort and defect) to be fed easily for predicting defects. The proposed framework provides a probabilistic defect prediction approach where instead of a definite number, a defect range (minimum, maximum, and mean) is predicted. The claim of efficacy and superiority of proposed framework is established through results of a comparative study, involving the proposed framework and some well-known models for software defect prediction. [51]

Arvinder Kaur, Kamaldeep Kaur and Chopra (2016) in their research article has depicted that there are many approaches for predicting bugs in software systems.
A popular approach for bug prediction is using entropy of changes as proposed by Hassan (2009). Their paper uses the metrics derived using entropy of changes to compare five machine learning techniques, namely Gene Expression Programming (GEP), General Regression Neural Network, Locally Weighted Regression, Support Vector Regression (SVR) and Least Median Square Regression for predicting bugs. Four software subsystems: mozilla/layout/generic, mozilla/layout/forms, apache/httpd/modules/ssl and apache/httpd/modules/mappers are used for the validation purpose. The data extraction for the validation purpose is automated by developing an algorithm that employs web scraping and regular expressions. The study suggests GEP and SVR as stable regression techniques for bug prediction using entropy of changes. [52]

Halili, and Rustemi (2016) has pointed out that with the rapid development of technology, there are various sophisticated software which enable them to solve problems in various spheres of our life. With the introduction of sophisticated software, there is a need also for new terms where it will be stored these data because we know that software cannot function if it does not have the most significant part and that is database. They have introduced the terms Big Data, Data Warehouses, Data Mining and their classification and done analysis for Regression technique (linear and multiple regression). Regression as technique although is predictive technique, but based on analyzes conducted to reach the conclusion most scientists, have concluded that the reliability percentage is around 95%. Through their paper they have demonstrated this scale of reliability through c examples.[53]

Rong, Li and Cui (2016) in their article has pointed out that software defect prediction is not only crucial for improving software quality, but also helpful for software test effort estimation. It has been observed that 80% of the fault happens in 20% of the modules. Therefore, there is a need to find out the most error prone modules accurately and correct them in time to save time, money and energy. There is one method that is Support vector machine (SVM) which is an advanced classification method that fits the defection classification. However, studies show that, the value of parameters of SVM model has a remarkable influence on its
classification accuracy and the selection process lacks theory guidance that makes the SVM model uncertainty and low efficiency. In their paper, a CBA-SVM software defect prediction model is proposed, which take advantage of the non-linear computing ability of SVM model and optimization capacity of bat algorithm with centroid strategy (CBA). Through the experimental comparison with other models, CBA-SVM is proved to have a higher accuracy. [54]

Periasamy and Mishbahulhuda (2017) have mentioned in their paper that Software defect prediction work focuses on the number of defects remaining in a software system. The software defect prediction model helps in early detection of defects and contributes to their efficient removal and producing a quality software system based on several metrics. A prediction of the number of remaining defects in an inspected is fact can be used for decision making. An accurate prediction of the number of defects in a software product during system testing contributes not only to the management of the system testing process but also to the estimation of the product’s required maintenance. Defective software modules cause software failures, increase development and maintenance costs, and decrease customer satisfaction. It strives to improve software quality and testing efficiency by constructing predictive models from code attributes to enable a timely identification of fault-prone modules. The main objective of paper is to help developers identify defects based on existing software metrics using data mining techniques and thereby improve the software quality. In this paper, we will discuss data mining techniques that are association mining, classification and clustering for software defect prediction. This helps the developers to detect software defects and correct them. [55]