A Systematic Literature Review on Cross-project Defect Prediction
Abstract

**Background:** Cross-project defect prediction, which provides feasibility to build defect prediction models in the case of lack of local data repositories, have been gaining attention within research community recently. Many studies have pursued improving predictive performance of cross-project defect prediction models by mitigating challenges related to cross-project defect prediction. However there has been no attempt to analyse the empirical evidence on cross-project defect prediction models in a systematic way.

**Objective:** The objective of this study is to summarise and synthesise the existing cross-project defect prediction studies in order to identify what kind of independent variables, modelling techniques, performance evaluation criteria and different approaches are used in building cross-project defect prediction models. Further, this study aims to explore the predictive performance of cross-project defect prediction models compared to within-project defect prediction models.

**Method:** A systematic literature review was conducted to identify 30 relevant primary studies. Then qualitative and quantitative results of those studies were synthesized to answer defined research questions.

**Results:** The majority of the Cross Project Defect Prediction (CPDP) models have been constructed using combinations of different types of independent variables. The models that perform well tend to be using combinations of different types of independent variables. Models based on Nearest Neighbour (NN) and Decision Tree (DTree) appear to perform well in CPDP context. Most commonly used Naive Bayes (NB) seemed to having average performance among other modelling techniques. Recall, precision, F-measure, probability of false alarm (pf) and Area Under Curve (AUC) are the commonly used performance metrics in cross-project context. Filtering and data transformation are also frequently used approaches in the cross-project context. The majority of the CPDP approaches address one or more data related issues using various row and column processing methods. Models appear to be performing well when filtering approach is used and model is built based on NB. Further, models perform well with data transformation approach is used and model is built based on Support Vector Machine (SVM). There is no significant difference in performance of CPDP models compared with Within Project Defect Prediction (WPDP) model performance. CPDP model perform well in majority cases in terms of recall.

**Conclusion:** There are various types of independent variables, modelling techniques, performance evaluation criteria that have been used in cross-project defect prediction context. Cross-project defect prediction model performance is influenced by the way it is being built. Cross-project defect prediction still remains as a challenge, but they can achieve comparative predictive performance as within-project defect prediction models when the factors influencing the performance are optimized.
Keywords
Systematic Literature Review, Software defect prediction, modelling technique

Supervisors
Professor, Burak Turhan, Seyedrebvar Hosseini
Foreword

I would like to express sincerest thanks to my supervisors Prof. Burak Turhan and Seyedrebvar Hosseini who gave me professional guidance and valuable feedback throughout whole process of conducting systematic literature review and thesis writing. In addition, I am also thankful to Olli Pakanen who helped me in the initial phase of the literature review by evaluating primary studies.

Data analysis phase in the review was the most challenging part in the whole process since there were lots of various data to analyse. I am grateful to my supervisors who always provided valuable suggestions and motivation whenever I was trapped with some issues.

At last but not least I want to thank my husband Tharanga, my daughter Diliru and my parents who always behind me encouraging me to complete this thesis.

Dimuthu Gunarathna
Oulu, October 21, 2016
Abbreviations

SLR     Systematic Literature Review
CPDP    Cross-Project Defect Prediction
WPDP    Within-Project Defect Prediction
CP      Cross-Project
WP      Within-Project
EBSE    Evidence-based Software Engineering
SCM     Static code metrics
LOC     Lines of code
OO      Object-oriented
DTree   Decision Tree
LR      Logistic Regression
NN      Nearest Neighbour
NB      Naïve Bayes
EM      Expectation Maximization
GP      Genetic programming
MO      Multi-Objective
MODEP   Multi-Objective Defect Prediction
CODEP   Combined Defect Prediction
HISNN   Hybrid Instance Selection Using Nearest-Neighbour
TCSBoost Transfer Cost-Sensitive Boosting
CCA     Canonical Correlation Analysis
DTB     Double Transfer Boosting
TNB     Transfer Naïve Bayes
VCB-SVM Value Cognitive Boosting- Support Vector Machine
TCA     Transfer Component Analysis
HDP     Heterogeneous Defect Prediction
LACE    Large-scale Assurance of Confidentiality Environment
MDP     Metrics Data Program
# Contents

Abstract ......................................................................................................................... 2  
Foreword ......................................................................................................................... 4  
Abbreviations .................................................................................................................. 5  
1. Introduction ................................................................................................................ 9  
   1.1 Motivation ................................................................................................................ 10  
   1.2 Related Literature Reviews .................................................................................... 10  
   1.3 Research goal ......................................................................................................... 10  
   1.4 Research Method ................................................................................................... 12  
   1.5 Structure ................................................................................................................ 12  
2. Prior Research ............................................................................................................ 13  
   2.1 Software defect prediction ...................................................................................... 13  
   2.2 Within-project / within-company defect prediction .................................................. 15  
   2.1 Cross-project/ cross-company defect prediction ......................................................... 16  
3. Research Methodology .............................................................................................. 18  
   3.1 Overview of systematic literature review .................................................................. 18  
   3.2 Process of Systematic Literature review .................................................................. 20  
4. Systematic literature review ......................................................................................... 21  
   4.1 Planning the review .................................................................................................. 21  
      4.1.1 Identification of the need for a review ................................................................. 21  
      4.1.2 Specifying the research question(s) ................................................................. 21  
      4.1.3 Developing a review protocol ...................................................................... 22  
      4.1.4 Evaluating the review protocol .................................................................. 22  
   4.2 Conducting the review ............................................................................................ 22  
      4.2.1 Identification of primary studies .................................................................. 22  
      4.2.2 Selection of primary studies ......................................................................... 24  
      4.2.3 Study quality assessment ........................................................................... 25  
      4.2.4 Data Extraction .......................................................................................... 29  
      4.2.5 Data Analysis ............................................................................................... 31  
5. Results ....................................................................................................................... 35  
   5.1 Results of study selection process ............................................................................ 35  
   5.2 Results of quality assessment process .................................................................... 36  
   5.3 Overview of studies ................................................................................................ 37  
   5.4 Types of Independent variables used in CPDP ....................................................... 38  
   5.5 Modelling techniques used in CPDP .................................................................... 41  
   5.6 Performance evaluation criteria used in CPDP ....................................................... 45
5.7 Different approaches used in CPDP ................................................. 47
  5.7.1 Data related issues ................................................................. 47
  5.7.2 Column processing methods .................................................... 50
  5.7.3 Row processing methods ......................................................... 54
  5.7.4 Data processing methods in continuous models ......................... 55
  5.7.5 CPDP approaches ................................................................. 56
5.8 CPDP model performance vs WPDP model performance .................. 61

6. Analysis ....................................................................................... 63
  6.1 Types of Independent variables used in CPDP (RQ1) ...................... 63
  6.2 Modelling techniques used in CPDP (RQ2) .................................. 64
  6.3 Performance evaluation criteria used in CPDP (RQ3) ................. 65
  6.4 Different approaches used in CPDP (RQ4) .................................. 66
  6.5 CPDP model performance comparison with WPDP model performance (RQ5) ................................................................. 68

7. Discussion and Implications .......................................................... 70
  7.1 Which types of independent variables have been used in CPDP and their performance? (RQ1) .................................................. 70
  7.2 Which modelling techniques have been used in CPDP and their performance? (RQ2) .......................................................... 70
  7.3 Which evaluation criteria have been used in CPDP performance? (RQ3) ... 71
  7.4 What are the different cross-project approaches used in CPDP to yield higher performance? (RQ4) ................................................. 72
  7.5 What is the performance of CPDP models compares to performance of WPDP models? (RQ5) ................................................................. 72

8. Validity threats ............................................................................. 73
  8.1 Publication bias ...................................................................... 73
  8.2 Search term bias .................................................................... 73
  8.3 Study selection bias .................................................................. 73
  8.4 Quality assessment and data extraction ................................... 74
  8.5 Violin plots ............................................................................. 74

9. Conclusion .................................................................................... 75
References ....................................................................................... 77
Appendix A. Search Strings ............................................................. 87
Appendix B. List of primary studies .................................................... 89
Appendix C. Context data table .......................................................... 94
Appendix D. Qualitative data ............................................................ 95
Appendix E. Quantitative data ............................................................ 96
Appendix F. Modelling technique vs CPDP approach matrix ............... 98
1. Introduction

Software quality assurance has become one of the most critical and expensive phase during the development of high assurance software systems. Software defect prediction is an activity which improves the software quality by identifying defect-prone modules prior to testing, so that software engineers can optimize testing resources allocation for testing and maintenance (He, Shu,Yang, Li, & Wang, 2012; D’Ambros, Lanza & Robbes, 2010; Tosun, Bener & Kale, 2010).

Usually defect prediction models works effectively as long as there is sufficient amount of past project metrics and defect data are available to train the model (Catal & Diri, 2009; Catal, 2011; Menzies et al., 2010). The prediction models where the model is trained from its own historical data refers to as within-project defect prediction (WPDP). But in a practical scenario, many companies often lack local data repositories because either a project is totally new or data was not properly collected due to infeasible effort and cost to maintain such data repositories (Zimmermann, Nagappan, Gall, Giger, & Murphy, 2009; Turhan, Menzies, Bener & Di Stefano, 2009). In such situation, cross-project defect prediction (CPDP) gives an attractive solution by predicting defects using prediction models trained from historical data of other projects (Turhan et al., 2009; Rahman, Posnett & Devanbu, 2012; Zimmermann et al., 2009). Such data can be collected either from finished projects or publically available data repositories such as PROMISE, Apache, Eclipse, and NASA MDP (Metrics Data Program) (Turhan et al., 2009; Herbold, 2013; He, Li, Liu, Chen& Ma, 2015).

Many research have been conducted in the defect prediction area are based on the within-project (WP) context (Ostrand, Weyuker & Bell, 2005; Menzies, Greenwald & Frank, 2007; Nagappan & Ball, 2005; Lessmann, Baesens, Mues & Pietsch, 2008; Kanmani, Uthariaraj, Sankaranarayanan & Sankaranarayanan, 2007). Lately, CPDP has been gaining researchers’ attention due to the fact that feasibility of predicting defects in a software system in any company those with limited or no defect data (Zimmermann et al., 2009; Turhan et al., 2009; He et al., 2012; Watanabe, Kaiya & Kaijiri, 2008). When look at the past researches in the context of CPDP, we can see some of the results are discouraging (Briand, Melo & Wust, 2002; Turhan et al., 2009; Zimmermann et al. 2009) while few of others provide promising results on CPDP models (Singh, Verma & Vyas, 2013; Rahman et al., 2012; Nam, Jialin Pan & Kim, 2013; Lin ,Fang, Shang & Tang, 2015).

The purpose of this study is to summarize and evaluate the state of the art in CPDP research by summarising the metrics, modelling techniques, different approaches and different performance evaluation criteria used in CPDP context with a systematic literature review and synthesis of evidence when possible. Moreover this study compares the CPDP model performances against WPDP model performances.
1.1 Motivation

As mentioned in the previous section, contradictory results of CPDP researches show how challenging CPDP is. Many of the previous studies have built complex, varying CPDP models and there is no up-to-date comprehensive picture of current state of CPDP. This is one of the key drivers to systematically summarize the empirical evidence on CPDP from the existing literature and studies.

Further, to the best of thesis author’s knowledge, there is no any systematic literature review has been conducted that focuses on CPDP. This motivated to study on CPDP to provide broader knowledge to existing research pool.

1.2 Related Literature Reviews

There are two systematic reviews have been conducted on the defect prediction area (Catal & Diri, 2009; Hall, Beecham, Bowes, Gray & Counsell, 2012). In 2009, Catal and Diri have done systematic review of software fault prediction studies with a specific focus on metrics, methods, and datasets. The most recent study, conducted by Hall et al. (2012), has investigated how model performance is affected by the context, independent variables and the modelling techniques. Those two previous reviews have brought up the analysis in general defect prediction. None of them have specifically addressed the CPDP context. Therefore, to facilitate the use of CPDP, it is necessary to systematically summarize the empirical evidence from the existing literature and studies.

1.3 Research goal

By considering motivation and the prior research, the goal of this study is to summarize, analyse and assess the empirical evidence regarding: metrics, modelling techniques, different approaches and performance evaluation criteria in the context of CPDP. Further, CPDP model performance against WPDP model performance is also explored. In order to achieve this goal, five research questions were defined. Table 1 shows the research questions and the motivation behind each question.
Table 1. Research questions addressed

<table>
<thead>
<tr>
<th>Research Question</th>
<th>Motivation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which types of independent variables have been used in CPDP and their performance?</td>
<td>There are different types of independent variables (metrics) that have been used in CPDP models in past studies. Product metrics, process metrics, object-oriented (OO) metrics and organizational metrics are popular among those. Goal of this study is to synthesise different types of metrics that have been used in CPDP based on current knowledge. Also performance of each metric type is investigated. In general, performance of these metrics or metrics types has been evaluated within individual studies, but there is no comparison has been done across studies to investigate which metrics or metrics types perform best in CPDP. So it is difficult to decide which set of metrics should be used to build the model. This study also investigates which metrics perform best in CPDP based on current knowledge.</td>
</tr>
<tr>
<td>Which modelling techniques have been used in CPDP and their performance?</td>
<td>There are different modelling techniques (machine learning and Regression modelling techniques) on which model is built. Logistic Regression (LR), NB, Bayesian Networks (BN), DTree, Random Forest (RF) are the popular techniques. The goal is to synthesise which modelling techniques (learning algorithms) have been used in CPDP based on current knowledge. Most recent study by Herbold (2013) has done a comparison in performance of cross-project prediction models built on seven different modelling techniques which has been successfully used in past. His experiment is based only on data from open-source projects written in Java. So it is also worth to study which modelling technique perform best in CPDP irrespective of the project type (open source or proprietary) or programming language. Therefore comparison on performance in different modelling technique also done.</td>
</tr>
<tr>
<td>Which evaluation criteria have been used to measure CPDP performance?</td>
<td>Most of the past studies have used standard measures of precision, recall and F-measure to measure the CPDP performance. The aim is to identify and synthesise different evaluation criteria that have been used to measure CPDP performance based on current knowledge and discuss their applicability.</td>
</tr>
<tr>
<td>What are the different cross-project approaches used in CPDP to yield higher performance?</td>
<td>There are different approaches have been used to achieve better predictive performance in the context of CPDP. Filtering, instance selection and transfer learning are some of the techniques used by researchers in past studies. The aim of this question is to find out what are the different approaches that can be used in CPDP model to achieve better performance based on the current knowledge.</td>
</tr>
</tbody>
</table>
What is the performance of CPDP models compared to performance of WPDP models?

Some researchers claim that CPDP models are able to achieve performance similar to WPDP model performance or better performance than WPDP models (He et al., 2012) (Zhang, Mockus, Keivanloo & Zou, 2015; JingWu, Dong, Qi & Xu, 2015). Also there are some studies which provide evidence that the CPDP model performance is worse than WPDP models (Turhan et al, 2009; Yu & Mishra, 2012; Kamei et al., 2015). So that, there is no clear consensus about performance of CPDP models compares to WPDP models. This made the interest towards comparing the results of CPDP performance with WPDP performance.

1.4 Research Method

For the purpose of meeting the research objectives, a systematic literature review is conducted. In this study, systematic literature review guidelines provided by Kitchenham and Charters (2007) were followed.

1.5 Structure

The structure of the thesis is organized as follows. Chapter 2 presents the literature review on work related to general software defect prediction, CPDP and WPDP. Chapter 3 explains the research methodology used. Chapter 4 provides details about whole procedure of SLR followed in the study. Findings of the study present in the Chapter 5. Chapter 6 contains analysis of the study. Chapter 7 present the discussion about results and implications and then validity threats are discussed in Chapter 8. Finally, Chapter 9 concludes the study.
2. Prior Research

In order to understand the software defect prediction process and different defect prediction types, prior research were reviewed and presented in this chapter. First general defect prediction related studies are presented and then WPDP and CPDP are discussed in respective sections.

2.1 Software defect prediction

Software testing consumes time and resources (Jureczko & Madeyski, 2010). Since the distribution of defects among individual part of software is varying, applying similar testing effort to all parts of a system is not the optimal approach. Therefore, defect-prone classes should be identified and testing tasks need to be prioritized accordingly (Jureczko & Madeyski, 2010). As described in Chapter 1, this process is called software defect prediction. Software defect prediction has gained the attention of many researches and has become an active area in software engineering research. General defect prediction process is presented in Figure 1.

![General defect prediction process](image)

As presented in the Figure 1, first step of the process is labelling the defect data collected as TRUE (buggy) or FALSE (clean). These defect data are collected from actual software development projects. Then features are extracted for predicting the labels of instances. By combining labels and features of instances, training corpus is developed. Then using training corpus and general machine learner, defect prediction model is built. Finally defect prediction model performance is measured. (Kim et al., 2011). 10-fold cross validation is one of the widely used evaluation method. In this method, one portion of the available data is used as the learning set to build the model and remainder used as the test set to evaluate the performance of the model (Koru & Liu, 2005). Defect prediction model can be used to predict defect-proness at different granularities such as method, class, file, package etc. (Radjenovic, Hericko, Torkar & Zivkovic, 2013). So that before designing a prediction model, prediction target needs to be specified (Kim et al., 2011).
According to Kim et al. (2011), there are two types of defect prediction. First type is “buggy file prediction” (i.e. traditional defect prediction) which uses code features such as complexity metrics (e.g. McCabe and Halstead), process metrics (e.g. code delta, code churn) and resource metrics to predict buggy files in advance. Various defect prediction models have been built based on metrics (Nagappan, Ball & Zeller, 2006; Zhang, Zhang & Gu, 2007; Moser, Pedrycz & Succi, 2008). As expressed by Fukushima, Kamei, McIntosh, Yamashiti and Ubayashi (2014), general defect prediction models describes the relationship between module metrics as independent variable and a module status (defect prone or not) as dependent variable. Most of the traditional defect prediction studies predicting defect-proneness is too coarse grained (i.e. File or package) (Kamei et al., 2010; Kamei et al., 2013). So that traditional defect prediction approach requires further analysis in those large files and packages which is not practical in software development environment because predictions are made late in the development and the developers involved may not recall the decisions made during development. Also it is difficult to figure out which developer should do the inspection/testing since usually many developers are involved in developing same file or package. (Kamei et al. 2013; Kamei et al. 2015)

Second defect prediction type, “change classification” addresses the limitations in traditional defect prediction approach which requires analysis on large files and packages by developers. It learns buggy change patterns from revision history. Then it predicts if a new change introduces bug or not. Change classification methods predict the location of latent software bugs in changes (Kim, Whitehead & Zhang, 2008). Change history can often extract from configuration management systems which use to manage the changes in software projects (Kim et al., 2011; Kim, Zimmermann, Pan & Whitehead, 2006). There are some advantages in change classification defect prediction over traditional defect prediction. For example, predictions made are fine grained as often buggy changes are mapped to small region of the code and save the effort and inspection can be easily assigned. Also predictions are made early on so that design decisions made during development are still fresh in their minds. In past studies this type of defect prediction approaches are referred as “Just-In-Time (JIT)”. (Kamei et al. 2013; Kamei et al. 2015)

In recent years researchers have proposed numerous defect prediction approaches (Ma, Luo, Zeng & Chen, 2012; Lin et al., 2015; Nam et al., 2013; Turhan et al., 2009). Most of the defect prediction models are based on supervised learning where the model is trained with labelled datasets (Nam & Kim, 2015). Prediction models based on supervised learning either classify the software module as defect-prone/not defect-prone (Jing et al., 2015) (Lin et al., 2015; Nam et al., 2013; Turhan et al., 2009) or predict the number of defects in a software module (Jureczko & Madeyski, 2010; Chen & Ma, 2015; Bell, Ostrand & Weyuker, 2006; Cruz & Ochimizu, 2009). Typically defect prediction practices based on supervised learning are designed for a single project and apply for projects with sufficient history data (Nam & Kim, 2015). As mentioned in Chapter 1, this type of defect prediction approach usually called as WP or within-company defect prediction. More details about within-project/within-company defect prediction will be discussed in Chapter 2.2. Major limitation of supervised learning based defect prediction techniques is the difficulty of applying for new projects or projects with inadequate historical data (Nam & Kim, 2015). To mitigate this limitation,
researchers have proposed CPDP models (Herbold, 2013; Turhan et al., 2009; He et al., 2012; Rahman et al., 2012). Also semi-supervised learning based defect prediction which uses small set of labelled data with many unlabelled data for training the model has been studied in the literature (Seliya & Khoshgoftaar, 2007; Lu, Cukic & Culp, 2012). Furthermore defect prediction models on unlabelled data sets were also introduced (Catal, Sevim, & Diri, 2009; Zhong, Khoshgoftaar & Seliya, 2004; Nam & Kim, 2015).

Majority of the defect prediction models in the literature are supervised learnings. Below section presents related studies on typical supervised learning based defect prediction; which is within-project/within-company defect prediction.

2.2 Within-project/within-company defect prediction

Within-project, within-company terms are used interchangeably in various past studies. Both terms refer to defect prediction models trained by available local data (Turhan et al., 2009; He et al., 2012). When the source and target data are coming from the same project, it is typically called within-project. On the other hand when the source and target data sets are from the same company, it is known as within-company defect prediction.

Catal and Diri (2009) have reviewed 74 software fault prediction papers in 11 journals and several conference proceedings with focus on metrics, methods and data bases use to build defect prediction model. The results of the review reveal that, usage of publically available data set has increased and the usage of machine learning algorithm slightly increased since 2005. Also in their study they report that method-level metrics are the main metric type used in defect prediction research area and defect prediction models are mostly based on machine learning algorithms.

Later in 2011, Catal investigates 90 software defect prediction papers published between 1990 and 2009. He categorizes papers according to the publication year. The study results further validates the results of study done by Catal and Diri (2009). Besides, they validate NB as a robust machine learning algorithm for supervised software defect prediction problems.

Hall et al. (2012) conduct a SLR to investigate how defect prediction model performance is affected by the context of the model, the independent variables used in the model, and the modelling techniques applied in the model. Their analysis is based on 208 fault prediction studies published from January 2000 to December 2010. The results of their study shows that defect prediction models based on simple modelling techniques, such as NB or LR have tendency to perform well. Further they articulate that the defect prediction models which have used combinations of independent variables perform well and the results were particularly well when feature selection has been applied to these combinations.

The majority of the studies used to synthesise the results of above mentioned reviews are related the defect prediction studies in the context on within project/within-company context. But as expressed in Chapter 1, WPDP is not
always feasible if the company does not collect and manage historical data to train the prediction model, so many researchers and practitioners attention begun to divert towards the cross-project/cross-company defect prediction. Next section presents the prior researches related to CPDP.

2.1 Cross-project/ cross-company defect prediction

Cross-company defect prediction denotes predicting defects using a prediction model trained by data from other companies. On the other hand CPDP refers to predicting defects using prediction model trained by history data from other projects. When only one project in a particular company is used to train the model, cross-company defect prediction can be consider as CPDP. (Jing et al., 2015). This section presents the past studies related to both cross-company and CPDP.

To the best of our knowledge, Briand et al. (2002) make the first attempt to build a CPDP model. They build the prediction model using a set of OO metrics collected from one java project to predict defects in another java project developed in the same environment. Their findings reveal that, though the systems stem from same development environment, applying the defect prediction models across systems is challenging. Also they propose cost-benefit model to assess the economic validity of defect prediction models and their results indicate defect prediction models provide significant benefits, even prediction models applying across systems. Later many researchers have studied about cross-project/cross-company defect prediction to make them useful in practice.

Turhan et al. (2009) have proposed a practical defect prediction technique for companies with lack of defect data. They have used Nearest-Neighbour (NN) filtering to select the similar sample from training data set. Their approach yield better prediction performance than training the prediction model using all available source data, but still the performance was lower than the within-company defect prediction. In addition, runtime complexity in NN filtering method increases with the data set size which ultimately caused for high computational cost. In their later paper, Turhan, Tosun and Bener (2011) investigate the effect of mixed (i.e. within and cross) project data on defect prediction performance. Their results show that the mixed project data based defect prediction models gain only minor improvement in the predictive performance and using CP data for defect prediction is still an open challenge.

Zimmermann et al. (2009) run 622 CP predictions for 12 commercial and open source projects. Results of their study show that usage of projects in the same domain does not help to build accurate defect prediction models. Also they suggest factors such as process, code data and domain that need to be evaluated carefully by software engineers before choosing projects for CP prediction model building.

In 2010, Jureczko and Madeyski have conducted a study which mainly focuses on clustering software projects to identify groups of software projects with similar characteristic. They use Hierarchical and k-means clustering to find groups of similar projects. Obtained clusters were investigated and ensure that the group exists through statistical analysis. They found that prediction from the models based on projects that belongs to identified group is significantly better than the all-projects models.
He et al. (2012) investigate defect prediction in CP context focusing on training data selection. They have used brute force strategy with a posteriori knowledge to select the best combination of data sets in order to yield best performance. Their results indicate that training data from same project does not always lead to better predictive performance than the training data from other projects. Furthermore they assert that the data distributional characteristics are related to prediction results.

Instead of using standard measures such as precision, recall, F-measure etc., Rahman et al. (2012) evaluate the prediction performance of CP models using new performance measure called area under the cost effectiveness curve (AUCEC). They argue that CPDP can achieve same performance as WPDP in terms of AUCEC.

A later study by He, Peters, Menzies and Yang (2013), propose another data selection method. Their data selection method selects potential training data based on the data similarity. Additionally they carry out feature selection to find features that cause the differences between training set and test set and they remove those unstable features to increase the similarity between training and test data sets. With this method, they have built the defect prediction model using data from open-source project to predict the defect in proprietary project. Their results and analysis exhibit that the proposed data selection method achieves relatively better prediction results than the NN-filtering proposed by Turhan et al. (2009).

Transfer learning is another attractive approach used by many of the past studies. Transfer learning techniques let data distribution of source and target data sets to be different (Ma et al., 2012). Ma et al. (2012) have proposed algorithm called Transfer Naive Bayes (TNB) which exploits information of all the cross-company data in training set instead of discarding some training samples. The solution given by them is to weight the instance of training data based on target set information and build the prediction model with weighted data. The results of the study exhibit TNB method provide good performance among the comparative methods while showing excellent runtime cost property. Nam et al. (2013) apply another transfer learning approach called Transfer Component Analysis (TCA) to make distribution in both source and target data similar. Additionally they propose TCA+ by extending TCA and they illustrate TCA+ significantly improve the CPDP performance. Recently Chen, Fang, Shang and Tang (2015) come up with novel algorithm called Double Transfer Boosting (DTB). DTB model first reshapes the entire distribution of cross-company data using data gravitation method and then transfer boosting method is designed with small ratio of labelled within-company data to remove negative instances in cross-company data. Their experimental results and synthesis declare that DTB provide better performance than the other tested cross-company defect prediction models and it is significantly better than within-company defect prediction models trained by limited data.

Menzies et al. (2013) propose a different methodology. Instead of predicting defects, they discuss ways to collect source data and to generate rules for minimising number of defects. Clustering method called WHERE is used for selecting source data. By eschewing learners that use statistical distributions and probability calculations to generate models, they learn the rules from WHICH learner to reduce the number of defects.
3. Research Methodology

Research methodology used to conduct this research is systematic literature review. Systematic literature review (SLR) is defined as “a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest” (Kitchenham & Charters, 2007). This section details the overview of SLR and difference between systematic review and conventional review and the stages of the SLR process.

3.1 Overview of systematic literature review

There are series of papers published by Kitchenham, Dybå and Jørgensen claiming that empirical software engineering researchers and practitioners should adopt evidence-based practice which founded in the fields of medicine and sociology. Also they assert that software engineering would benefit from evidence based approach to deal with the specific problems arising from the nature of software engineering (Kitchenham, Dyba & Jørgensen, 2004; Dybå, Kitchenham & Jørgensen, 2005). They come up with a framework for Evidence-based Software Engineering (EBSE) which derived from medical domain. EBSE aims to improve decision making in software development and maintenance by aggregating best available evidence from research to address specific research questions. The methodology aggregating empirical evidence is SLR (Kitchenham et al., 2010).

Kitchenham et al. (2004) adapt medical guidelines for SLRs to the needs for software engineering research community. Later in 2007, Kitchenham and Charter update guidelines to be consistent with social science viewpoints. SLR discovers and summarizes patterns of existing research and identifies any gaps in the current research to be filled by future research (Kitchenham & Charters, 2007). SLRs are referred as secondary studies and studies that are summarized in the particular systematic review are termed primary studies.

“Systematic review”, “Systematic literature review”, “Research synthesis” and “meta-analysis” are different terms often used to describe the process of systematically reviewing papers and synthesizing research evidence about subject matter of interest. First three terms can be used interchangeably (Dybå, Dingsoyr & Hanssen, 2007). But systematic reviews and meta-analyses are not synonymous. As defined by Pai et al. (2004), “A meta-analysis is the statistical pooling of data across studies to generate summary (pooled) estimates of effects”. Typically meta-analysis should be conducted as a part of systematic review (Pai et al., 2004). See Figure 2. In meta-analysis, research synthesis is based on the quantitative statistical method (Kitchenham & Charters, 2007).
SLRs differ from traditional or narrative reviews in being followed by well-defined and planned sequence of methodological steps according to developed a priori protocol (Biolchini, Mian, Natali & Travassos, 2005). This well-established protocol minimizes the bias in the study and ensures the repeatability of the study. Table 2 summarizes differences between traditional reviews and SLRs.

**Table 2. Differences between traditional reviews and systematic reviews (Pai et al., 2004) (Dybå et al., 2007)**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Traditional review</th>
<th>Systematic review</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formulating question</td>
<td>Usually broad in scope</td>
<td>Often addressed focused question (s)</td>
</tr>
<tr>
<td>Search strategy</td>
<td>Often search strategy is not described and usually cannot replicate</td>
<td>Clearly defined search strategy is available which rigorous, transparent and repeatable</td>
</tr>
<tr>
<td>Study selection</td>
<td>Usually not described or all studies are accepted</td>
<td>Clearly defined inclusion/exclusion criteria in order to reduce likelihood of bias</td>
</tr>
<tr>
<td>Quality assessment</td>
<td>Usually include all identified studies without excluding any based on the quality of the study</td>
<td>Quality assessment criteria is defined and exclude primary studies accordingly</td>
</tr>
<tr>
<td>Data extraction</td>
<td>Usually data extraction method not described</td>
<td>Data extracted by the defined data extraction form and it is piloted to reduce bias. Usually involved in more than one reviewer</td>
</tr>
<tr>
<td>Data Synthesis</td>
<td>Usually qualitative summary</td>
<td>Qualitative and/or quantitative analysis (meta-analysis)</td>
</tr>
<tr>
<td>Interpreting results</td>
<td>Prone to systematic bias and personal bias</td>
<td>Less prone to systematic bias and personal opinion</td>
</tr>
<tr>
<td>Heterogeneity</td>
<td>Usually dealt with a narrative fashion</td>
<td>Impact of potential sources of heterogeneity, (e.g. study type, study quality, and sample size) are graphically represented, Sources of heterogeneity are identified</td>
</tr>
<tr>
<td>Appraisal</td>
<td>Variable</td>
<td>Can be critically appraised</td>
</tr>
<tr>
<td>Interferences</td>
<td>Sometimes evidence-based</td>
<td>Usually evidence-based</td>
</tr>
</tbody>
</table>
3.2 Process of Systematic Literature review

There are several discrete stages in conducting SLR's. These stages can be grouped into three main phases: planning the review, conducting the review and reporting the review (Kitchenham & Charters, 2007; Biolchini et al., 2005). The process is depicted in Figure 3. Execution of overall SLR process involves iteration, feedback, and modifications of the defined process.

![Diagram](image.png)

**Figure 3.** Stages in systematic review process (Kitchenham & Charters, 2007)

The following sections of the thesis present all the necessary steps to carry out the SLR in the context of the research objectives of this study.
4. Systematic literature review

The systematic literature review in this study followed the guidelines proposed by Kitchenham & Charters (2007). Each phase in the SLR and their constituent stages are described in this chapter. It starts from the planning the review and leads to conducting the review and reporting the review of SLR. Results of the SLR process are presented on Chapter 5.

4.1 Planning the review

In the initial phase of the systematic review, purpose of the research and the procedures for the review to be executed were defined. This included identifying the need for a review, formulating research questions, producing review protocol and evaluating the review protocol. The stages associate with planning the review is detailed out in below sections.

4.1.1 Identification of the need for a review

The first step for this SLR (Step no 1, Figure 3) was identifying the need for systematic review. As stated in the introduction on the thesis, CPDP is challenging and there is no up-to-date comprehensive picture of the current state of CPDP. Further, no prior researchers have conducted SLR on CPDP. All these reasons revealed the need for SLR on CP context.

4.1.2 Specifying the research question(s)

Based on the recommendation from medical domain, well-built question comprises with four critical parts namely Population, Intervention, Comparison and Outcomes. It denotes by acronym PICO (Pai et al., 2004). Individual terms in PICO criteria identified in the study are given below.

- Population: Software defect prediction studies
- Intervention: Cross project defect prediction model
- Comparison: Within project defect prediction (when applicable)
- Outcomes: prediction related metrics

Within the context of this study, CPDP was studied from the point of view of the following research questions in the Table 3. Detailed justification for research questions is provided in the section 1.3.
### Table 3. Research questions

<table>
<thead>
<tr>
<th>ID</th>
<th>Research question</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQ1</td>
<td>Which types of independent variables have been used in CPDP and their performance?</td>
</tr>
<tr>
<td>RQ2</td>
<td>Which modelling techniques have been used in CPDP and their performance?</td>
</tr>
<tr>
<td>RQ3</td>
<td>Which evaluation criteria have been used in CPDP performance?</td>
</tr>
<tr>
<td>RQ4</td>
<td>What are the different cross-project approaches used in CPDP to yield higher performance?</td>
</tr>
<tr>
<td>RQ5</td>
<td>What is the performance of CPDP models compared to performance of WPDP models?</td>
</tr>
</tbody>
</table>

### 4.1.3 Developing a review protocol

The review protocol used in this study was developed by the author of the thesis. Further, other review team member also participated in developing the protocol by giving suggestions. Before finalizing the review protocol, some aspects of the protocol such as search terms, inclusion/exclusion criteria and data extraction form were piloted during its development and necessary changes were made.

### 4.1.4 Evaluating the review protocol

The review protocol in this study was reviewed and evaluated by the thesis supervisor and updated accordingly. Also protocol was iteratively enhanced during conducting and reporting stages of the SLR. Following sections detail out the element of the review protocol.

### 4.2 Conducting the review

Actual review process was commenced after the review protocol was finalized and agreed. Stages carried out in this phase elaborate in following sections.

#### 4.2.1 Identification of primary studies

The defined searched strategy in the review protocol was used to find as many as primary studies relating to research questions. Below section describes the search strategy, bibliography management and how search was documented in the SLR presented in the thesis.

**Search strategy**

There were two elements in the search strategy.

1. **Search in electronic databases using derived search string**

   The following three electronic databases were chosen for searching primary studies. These three electronic databases covered majority of the software engineering publications. Moreover, those electronic databases have been successfully used by previous literature review on software fault prediction (Hall et al., 2012).
In addition to above mentioned digital databases, below two search engines were also added by consulting with the thesis supervisor who is an expert in the field.

- Google Scholar
- Scopus

Studies were searched in electronic databases using derived search string. Following steps were used to construct the search string used in the SLR.

1. Derive search terms from research questions by identifying PICO
2. Identify search terms by checking titles and keywords in any relevant papers already known.
3. Identify alternative spellings and synonyms for search terms.
4. Construct search string using identified search terms, Boolean ANDs and ORs.

Search string constructed in the study is given below. All searches were based on the full text of the study.

(“cross-project” OR “cross project” OR “multi-project” OR “multi project” OR “multiple project” OR “cross-company” OR “cross company” OR “mixed-project” OR “mix-project” OR “mixed project” OR “mix project”) AND (defect* OR fault* OR bug* OR error*) AND (predict* OR estimat*) AND software

Search string was adapted to suit the specific requirements of different electronic databases. Adapted search strings based on different database requirements are given in the Appendix A. Pilot search was performed to verify that the search string is properly constructed to identify all relevant primary studies. Search results from the pilot study were validated against list of known papers. Electronic databases were searched on 6th of June 2015 and SLR covers studies published to the date.

2. Snowballing technique

Snowball sampling method (Wohlin, 2014) was used to identify additional primary studies. This process was carried out after the primary study selection process. There are two methods of snowballing; backward snowballing and forward snowballing. Backward snowballing technique was carried out to identify the candidate primary studies reviewing references of the primary studies selected from databases. In contrast, forward snowballing method helped to identify studies by searching the citation of the selected primary studies. This process was repeated until no further relevant papers were identified based on the inclusion criteria.

Bibliography Management

Bibliographic package Mendeley (https://www.mendeley.com/) was used to manage references obtained from search. In addition to Mendeley, Microsoft Excel was also used to store and remove duplicates of references.
Documenting the search

As stated by Kitchenham and Charters (2007), SLR needs to be well documented, so that reader can critically appraise the review. In this study, Excel was used to document the search. Electronic databases, search terms used in individual electronic database, date of search and no of studies found were reported.

4.2.2 Selection of primary studies

The process of selection of primary studies aims to identify those studies that provide evidence about the research questions defined in the SLR (Kitchenham & Charters, 2007). The selection process was carried out in the way that it is described in the protocol to reduce any bias. Inclusion and exclusion criteria presented in the Table 4 were used for the selection of primary studies. Publication year was not taken into account for exclusion of paper.

Table 4. Inclusion and Exclusion criteria

<table>
<thead>
<tr>
<th>Inclusion criteria</th>
<th>Described research on CPDP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Peer reviewed research paper</td>
</tr>
<tr>
<td></td>
<td>Most comprehensive and recent study for repeated publications of the same study</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Exclusion criteria</th>
<th>Grey literature</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Full text is not available</td>
</tr>
<tr>
<td></td>
<td>Studies not written in English</td>
</tr>
<tr>
<td></td>
<td>Conference version of the study for those having both the conference and journal versions</td>
</tr>
</tbody>
</table>

Before performing study selection process, inclusion and exclusion criteria were piloted by two researchers including thesis author, on randomly selected 15 primary studies from initial set of papers. Agreement between researchers was measured by Pairwise inter-rater reliability. Cohen’s Kappa statistic (Kohen, 1960) was calculated. According to the Kappa statistics vs strength of agreement specified by Landis and Koch (1977), level of agreement was determined. Further, few random papers were assessed by the supervisor and cross checked the results to validate the process. Based on the pilot study inclusion and exclusion criteria were clarified and revised. The second iteration in the pilot study resulted in 100% agreement between researchers which is an acceptable level to start the primary study selection procedure.

As depicted in Figure 4, at first, duplicate primary studies were discarded and then the study selection process was performed in two phases:

1. Exclusion of primary studies by manually scanning the title and abstract
2. Exclusion of primary studies based on the full text.
Two researchers conducted the selection process on initial set of primary studies. Primary studies were individually evaluated and then any disagreement was resolved to reduce the possible researcher bias. In the situation, where agreement could not be reached among two researchers, then the supervisor was consulted to resolve these disagreements. Cohen’s Kappa coefficient was calculated for the process of evaluating total set of studies as well. Process was replicated till the agreement reached 100% between researchers.

![Study selection process](image)

**Figure 4.** Study selection process

### 4.2.3 Study quality assessment

In addition to inclusion and exclusion criteria, the quality of each primary study was assessed based on the defined quality checklist. The quality checklist questions were developed by considering the suggestions given by Kitchenham and Charters (2007) and Hall et al. (2012). Main goal of the quality assessment of the study was to filter out primary studies reported sufficient information to analyse and answer the defined research questions.

The assessment process was piloted by applying assessment criteria to randomly selected five papers. Two researchers (including thesis author) involved in pilot study. The quality assessment checklist was refined according to the results of the pilot study. The quality assessment criteria built with six stages described below.
Stage 1: Prediction

At this stage, it is checked that the study report is a defect prediction. To achieve the goal of the SLR, it is important to choose the papers which build the actual defect prediction model. There are some papers which investigate the nature of relationship between metrics and defects (Singh, Chaudhary & Verma, 2011; Briand, Wust, Daly & Porter, 2000). Further, prediction model should be trained and tested on different datasets (Song, Jia, Shepperd, Shi & Liu, 2011). Table 5 describes the prediction criteria applied.

Table 5. Prediction Criteria

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Criteria definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is a prediction model building reported?</td>
<td>The study must report defect prediction model building. It should not be just a study that evaluates the relationship between metrics and defect data.</td>
</tr>
<tr>
<td>Is the prediction model tested on unseen data?</td>
<td>The defect prediction model should be tested with unseen data to check how well the prediction model performed. Unseen data refers to data not used to generate defect prediction model (Menzies et al., 2007). Performance of a defect prediction model can be evaluated when the exact output of unseen data is acknowledged. This can be done with traditional holdout validation or cross validation technique.</td>
</tr>
</tbody>
</table>

Stage 2: Context

The second quality assessment stage was applied to ensure sufficient context data are reported in the study. Reporting sufficient context data in the study are important when interpreting findings. Same contextual criteria used by the Hall et al. (2012) were utilized in this study with few modifications. Maturity was removed from context criteria. Many of the CPDP studies are based on NASA data sets located in either NASA MDP or PROMISE repository and maturity of those data sets was not given in included studies and no maturity information could be found from their repository documentation, so that, applying maturity criteria would have resulted in only a handful of studies being analysed. Moreover, application domain and size criteria also modified as in the Table 6. To pass this stage, context information listed in the Table 6 should be reported in the study.
Table 6. Context Criteria

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Criteria definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source of data</td>
<td>Source of data used in the study must be given. For an example, whether the data are from industrial, open source, academic, NASA, PROMISE etc. If the study has used publicly available data like NASA or PROMISE, data set names, versions must be given. So that contextual data can be accessible via public domain and it is not mandatory to report context data on those studies.</td>
</tr>
<tr>
<td>Size</td>
<td>The size of the system being studied must be given. It could be in KLOC (Thousands of Lines Of Code), no of classes, no of Instances, or an indication to get an idea about the size of the system</td>
</tr>
<tr>
<td>Application domain</td>
<td>The application domain of the system being studied must be given. (E.g. customer support). If the model used data from different systems, application domain should be given at least for more than half of the system.</td>
</tr>
<tr>
<td>Programming language</td>
<td>Programming language(s) of the system being studied must be given.</td>
</tr>
</tbody>
</table>

Stage 3: Prediction model building

Defect prediction model building data are crucial in this SLR to answer research question RQ1 and RQ2. Independent variable, dependent variable and its granularity and modelling technique used to build the model should be reported in the study. Table 7 presents the criteria applied to guarantee required model building information are reported.

Table 7. Prediction model building Criteria

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Criteria definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Are the Independent variables (metrics) which used to build the model clearly reported?</td>
<td>Independent variables (or predictor variables) used to build the model should be clearly defined. For an example, independent variables could be size and complexity, coupling, cohesion, code delta, code churn etc.)</td>
</tr>
<tr>
<td>Is the dependent variable clearly reported?</td>
<td>It must be clear that the dependent variable, i.e. output of the prediction model is defined. The model can predict if the code unit is faulty or non-faulty (i.e. categorical dependent variable) or predict the number of defects in a code unit (i.e. continuous dependent variable). Also, some studies classify unit code into different defect severity levels (Radjenović et al., 2013).</td>
</tr>
<tr>
<td>Is the granularity of the dependent variable reported?</td>
<td>The granularity of the dependent variable, the level at which defect predictions were made must be clearly mentioned in the study. For an example, defects predict in terms of, defects per method, per class, per package, per file etc.</td>
</tr>
</tbody>
</table>
Is the modelling technique on which the model was built clearly reported?

It must be distinguished which modelling technique being used in the study. For example, statistical (NB, LR) and machine learning language (DTree, SVM, K-nearest neighbour (K-NN) etc.)

**Stage 4: Approach**

To answer RQ4, it is necessary to clearly report any used CP approach in the study. Criteria in stage 4 were applied only for studies reported CP approaches. In the remaining studies, the criteria given at this stage was made not applicable. Table 8 shows the criteria in quality assessment stage 4.

**Table 8. Approach**

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Criteria definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Has the cross-project approach presented in the study clearly reported?</td>
<td>Some studies utilize different CP approaches to yield better predictive performance in CPDP. For an example, NN-filtering, EM-clustering, transfer learning, etc. If the study has used any CP approach, it must be clearly reported in the study.</td>
</tr>
</tbody>
</table>

**Stage 5: Data**

Data and how the data being acquired should be reported on the study. Those criteria are essential to ensure reliability of the prediction model. Table 9 denotes the criteria applied in this stage.

**Table 9. Data Criteria**

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Criteria definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is the fault data extraction process described?</td>
<td>How fault data are obtained must be clearly described. If the study has used private data set, it is necessary to mention how data were extracted. For an example fault data was extracted from a Version Control System (VCS) or Subversion (SVN). If the data was obtained from publicly available repository (e.g. PROMISE repository), reference or any evidence must be presented to understand the way data was obtained.</td>
</tr>
<tr>
<td>Is metric(s) (Independent variable) data extraction process adequately described?</td>
<td>The process by which independent variable (metric) was obtained must be stated. For an instance, any tool used to extract metrics data should be mentioned or if the study has used history data, process of metric collection must be given. In the case of using publicly available data reference or any evidence must be presented.</td>
</tr>
</tbody>
</table>
**Stage 6: Predictive performance**

Assessing the performance of the prediction model is a vital task to validate the model’s usefulness. In addition, it helps detailed comparison among other models (Jiang, Cukic & Ma, 2008). There are various performance criteria (performance metrics) in the literature. Moreover, predictive performance measures differ depending on the model type; continuous or categorical. Categorical models report their performance results on the basis of whether a code unit is defect-prone or not defect-prone. Continuous models report their performance results in terms of the number of faults predicted in a unit of code. Code unit can be a package, class, file, module, etc. To be able to answer the RQ4 study should satisfy the criteria shown in the Table 10.

**Table 10. Predictive Performance Criteria**

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Criteria definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is the performance evaluation criteria used clearly reported?</td>
<td>How defect prediction model performance was measured must be reported. For an example, performance of categorical model results can be measured by confusion matrix or AUC or continuous model results can be measured by error measures.</td>
</tr>
<tr>
<td>Has predictive performance values been reported clearly?</td>
<td>Predictive performance values must be clearly presented.</td>
</tr>
</tbody>
</table>

Studies those report any specific approach in their study, all the criteria defined in each stage should be satisfied to pass the quality assessment. Studies do not report any specific CP approach, criteria at all stages except stage 4 should be satisfied to pass the quality assessment. If any of the criteria fails in any stage, the primary study was considered as failed in the quality assessment and no further analysis was done in that study. Status of the particular study was denoted by (Status: Pass/Fail). If the study fail, then the stage which is failing also given. For an example, if the paper A1 is failing in stage 3, then it was denoted by (Status: Fail, Stage: 3).

Quality assessment checklist with 5 stages was applied to all selected set of primary studies by two researchers (including the thesis author) individually. All disagreements of the quality assessment results were discussed among two researchers, and then the consensus was reached eventually. There were few cases where agreement could not be reached among two researchers and those were then sent to the supervisor for moderation.

**4.2.4 Data Extraction**

Data extraction form was designed to collect the data from primary studies with the intention of answering research questions. Data was extracted from each study which passed all quality assessment criteria. Data extraction form was piloted on a sample of five randomly selected primary studies to assess completeness and usability issues. Two researchers (including the thesis author) were involved in pilot study. The structure of the data extraction form was updated based on the pilot study.
Data extraction form mainly consists of following three data sets. They are context data, qualitative and quantitative data. These data sets are described below.

**Context of study**

Data represent the context of each study were extracted in terms of: the aim of the study with details of systems used in the study such as application domain(s), project name(s), size(s) and source(s). If the study had addressed any data related issues such as difference in data distribution, class imbalance problem, etc., then those were also gathered with the aim of the study.

**Qualitative data**

For each primary study, short summary of main findings and conclusions reported by the author(s) were extracted.

**Quantitative data**

All the defect prediction model performance values for models and their model variants reported in the studies were extracted. Quantitative data were extracted as two different sets according to how the study reports their results, either via categorical or continuous dependent variables. There were some studies which report both categorical and continuous results. For those studies, only one set of results was extracted based on the way in which the majority of the results were presented by those studies. Data extracted for both categorical and continuous models are presented in below sections.

**Categorical models:**

For each study reported categorical model, following data was extracted for both CPDP and WPDP models. Dependent variable granularity, independent variable(s), performance evaluation criteria(s) used, modelling technique(s) used, data set (s), and indication of performance value are related to cross-project/within-project (“C” for CPDP and “W” for WPDP) and form of the performance value ( e.g. original, average and median). If the performance values were given in their original form the form of the performance value marked as “Original”. If the average performance values were given it is mentioned as “Average”. Similarly, “Median” is marked as the form of the performance value, for those given median performance values. Moreover, this table reports all the performance values of each model using any performance evaluation criteria given in the study. For an example, if the study had evaluated performance using multiple criteria such as probability of detection, probability of false alarms and balance, all those values were gathered. Moreover, additional performance values were calculated and derived from the reported values where it is possible. Appendix H shows the formulas used during calculations. Data set column filled as to show both training data and test data set used in the study. For an example, it denoted as Training data (Source) set → Test (Target) data set. Approach used to build the prediction model also extracted only for CPDP models. Additionally if the study mentioned any information relate to how specific predictive performance metric was chosen for model performance measuring, those information also gathered on the performance evaluation criteria(s) used column for CPDP models.
Continuous models:
For each study reported continuous model, similar set of information was gathered as for categorical models. The performance values of each model were gathered with all given performance evaluation criteria. For an example, some studies reported RMSE (Root Mean Square Error) while other reported R2 or Nagelkerke’s R2.

The total set of papers included in the data extraction phase divided between two researchers (including the thesis author) and both researchers extracted data from each set of papers independently. Extracted data by each researcher were held in tables, one excel file per study. After researchers completed data extraction for set of allocated studies, then each researcher double checked the data extraction forms filled by other researcher to ensure all necessary data were extracted accurately. Incongruities and errors were discussed. When disagreements could not be resolved among researchers, then supervisor was consulted to resolve those disagreements. Also data were extracted by both the researchers from randomly selected 3 papers to guarantee that researchers extracted data in a consistent manner. Final agreed data was then saved into a single excel file for further use during the data analysis phase.

4.2.5 Data Analysis

In the data analysis phase of the SLR, data were synthesised in a manner appropriate for answering defined research questions. In this study both qualitative and quantitative data were scrutinized and assessed. As denoted by (Brereton et al., 2007), even quantitative data being gathered, it may not be possible to perform meta-analysis of software engineering studies because the reporting protocols vary so much from study to study. Studies in this SLR were also highly dissimilar in terms of context, approaches, and variables. So that meta-analysis was not considered. In this study qualitative and quantitative data were combined to generate a better interpretation of data.

Violin plots (Hintz & Nelson, 1998) were used in order to graphically analyse the performance of different modelling techniques, different types of independent variables and CP approaches reported in studies. In addition to violin plots, bar charts and tables were used for summarizing and presenting data.

Violin plots were drawn using only categorical data. Each violin plot includes data only where at least two models have used particular factor (For an example, a particular modelling technique such as Naïve Bayes). The solid horizontal lines in the plots indicate the mean value. Continuous data reported by three studies were not used to draw violin plots as the performance measures used were dissimilar and unable to convert to comparable measures.

**Determining predictive performance metrics use for comparison**

There was no common performance metric(s) used by all selected studies reported categorical models to compare their performance values in relation to certain factor (e.g. independent variables, modelling techniques, CPDP approaches. Therefore, studies reported performance values with same performance metric were compared
together. This rule is applicable to all the plots drawn to compare performance in relation to various factors.

Form of predictive performance values

The studies reported categorical models had presented their predictive performance values in different forms. For an example, some studies had built prediction model by training the model with numerous project data sets and reported the average performance of those models. On the other hand, there were few studies repeated/iterated some steps in the model building process and then reported average performance values of models built in each iteration. There were another set of studies which had presented median model performance values. In addition to average and median predictive performance values, some studies had given performance values as they were. In the rest of the thesis this kind of performance values refers as original performance values. When drawing violin plots, studies presented performance values in the form of original and average performance values were combined and drawn together. Similarly, studies presented performance values in the form of original and median performance values were combined and drawn together. This rule is applicable to all the plots drawn to compare performance values in relation to different types of independent variable, modelling technique and CPDP approaches. The form of performance value given in a particular study can be found from the column named “Performance values are given by” in the categorical model data table in Appendix E.

Performance comparison: Types of Independent variables

Three sets of plots were drawn to compare performance of different types of independent variables. First set of plots were drawn using f-measure, precision and recall performance values (either reported on the study or calculated by researchers). Second set of plots were drawn using performance values reported in terms of AUC. For the third set of plots, recall, probability of false alarm and balance were utilized. Further, violin plots were drawn separately based on the form of the performance values presented in the study.

Performance comparison: Modelling techniques and approaches

A matrix was prepared for creating plots to compare different modelling techniques and different CPDP approaches. The matrix can be found from the Appendix F. Columns on the matrix represent the modelling techniques or optimization techniques used in studies reported categorical models. Columns were divided into 5 categories such as “Base learners”, “Learners with boosting”, “Genetic programming (GP)”, “Multi-objective (MO) – optimization” and “Other ensembles”.

- **Base learners:** Studies used base learners without any optimization technique
- **Learners with boosting:** Studies used learning algorithm with boosting (e.g. proposed approaches such as DTB, Transfer Naïve Bayes (TNB) etc.)
- **Genetic Programming:** Studies followed genetic programming approach
- **Multi-objective optimization**: The studies used MO approach to provide software engineers the capability of balancing between different objectives.

- **Other ensembles**: Studies used ensemble approaches such as bagging, composite algorithms etc.

When the study reports multiple prediction models which are based on different modelling technique/optimization technique, such studies were added under each modelling technique/optimization technique utilized. Rows on the matrix represent the different approaches used by studies reported categorical models. The approaches used by each study were mainly categorized into 6 groups. They are “Filtering”, “Data transformation”, “Clustering”, “Mixed data” and “Feature selection”.

- **Filtering**: The studies used filtering approach (e.g. NN-filtering). More information about filtering can be found from section 5.7.3

- **Data transformation**: The studies used transformation technique to gain better performance in CPDP models. Section 5.7.2 gives more information about data transformation.

- **Clustering**: Studies used various clustering algorithm such as Expectation Maximization (EM) clustering, NN clustering etc. based approaches. Section 5.7.6 details out clustering approach.

- **Mixed data**: The studies used a mix of CP data and WP data to train the model. More details about the mixed data approach can be found in the section 5.7.6

- **Feature selection**: The studies selected subset of features when building prediction model. Section 5.7.2 provides more information about feature selection

- **Other approaches**: The studies used approaches other than the aforementioned approaches. For an example, studies used simple data processing methods (e.g. data sampling, data normalization etc.) and studies not utilized any specific approach

There were 3 cases which needed to be considered when adding study in a particular approach category row. The first case is, when the study reported multiple models with different approaches, and then the study was added in all relevant approach categories. The second case is, when proposed CPDP approach by study consisted of multiple approaches, then the study was added in all relevant approach category. When drawing plot for comparing learning algorithms/optimization techniques, those studies reported in multiple approach categories (Case 2) were counted only once. The third case is, if the proposed approach by any study used one approach as part of the main approach, then study reported only in the main approach category.
Forest plot

The forest plot was used to represent the results of quantitative summary of comparing CPDP model performance against WPDP model performance using precision, recall and f-measure performance metrics. As stated by Kitchenham and Charters in 2007, forest plot present the means and variance of the difference for each study. The line of the plot denotes the standard error of the difference. The box on the plot represents the mean difference and size of the box proportional to the no of subjects in the study. Similar approach defined by Kitchenham and Charters (2007) was used to draw the forest plot. Mean values were calculated from the difference of WPDP and CPDP model performance and then standard errors of difference were calculated. In this study no of subjects are the no of data rows.
6. Results

This section presents the results of the SLR in the study. First, the results of study selection process and quality assessment process along with the overview of selected primary studies are presented. In subsequent sections, the findings are arranged into different themes based on research questions (i.e. different types of independent variables, modelling techniques, performance evaluation criteria and CPDP approaches). The last section of this chapter presents results of comparison CPDP model performance with WPDP model performance.

5.1 Results of study selection process

Figure 5 shows that initial search in the databases found 1889 studies. After discarding the duplicate studies, 962 studies left for further assessment. Applying inclusion and exclusion criteria for the title and the abstract of each paper diminished the pool of papers for full-text reading to 41.

---

As mentioned in the section 4.2.2, inter-rater agreement analysis was conducted to determine the agreement between two researchers. The Cohen Kappa statistic was calculated for both pilot and actual study evaluation process. According to the...
Kappa statistics vs strength of the agreement specified by Landis and Koch (1977), the agreement between two researchers for the exclusion of primary studies based on the title and abstract in the pilot study was “moderate” (0.59). See Table 11.

**Table 11.** Reviewer’s agreement on the exclusion of primary studies in pilot study based on inclusion and exclusion criteria

<table>
<thead>
<tr>
<th></th>
<th>Reviewer 1</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accepted</td>
<td>Declined</td>
</tr>
<tr>
<td>Accepted</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Declined</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>Total</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 12 shows the agreement between the two researchers for assessing a full set of papers. The agreement between two reviewers was “Substantial” (0.69). There were 4 cases where agreement could not be reached by 2 researchers and those were given to supervisor to make the final decision on the inclusion of the papers.

**Table 12.** Reviewer’s agreement on the exclusion of full set of primary studies based on inclusion and exclusion criteria

<table>
<thead>
<tr>
<th></th>
<th>Reviewer 1</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accepted</td>
<td>Declined</td>
</tr>
<tr>
<td>Accepted</td>
<td>31</td>
<td>6</td>
</tr>
<tr>
<td>Declined</td>
<td>21</td>
<td>904</td>
</tr>
<tr>
<td>Total</td>
<td>52</td>
<td>910</td>
</tr>
</tbody>
</table>

In the final set, 28 studies were remained after filtering out studies based on full-text. Additional 6 and 12 relevant primary studies were found and added to the list from the backward snowballing and forward snowballing technique respectively, so that, whole process resulted in 46 studies included in this SLR. See Appendix B.

### 5.2 Results of quality assessment process

Quality assessment criteria were applied to an included set of 46 CPDP primary studies. 24% of the studies failed to meet the quality assessment criteria, so that only 35 studies remained for data extraction phase. Table 13 shows the number of studies failed in each quality assessment stage. As in the Table 13, five studies failed on the stage 6. Three of the studies failed in stage 6 due to not reporting predictive performance values clearly in the study. Since the performance comparison was carried out to answer some research questions, reporting predictive performance criteria along with the performance values in each study was crucial in this SLR. In general, it is found out, no common performance criteria were followed by studies in the literature and it caused for difficulty in comparing performance values reported among studies.

**Table 13.** Results of applying quality assessment criteria

<table>
<thead>
<tr>
<th>Quality assessment stage</th>
<th>No of failed studies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1: Prediction</td>
<td>0</td>
</tr>
</tbody>
</table>
Stage 2: Context | 2
Stage 3: Prediction model building | 2
Stage 4: Cross-project approach | 1
Stage 5: Data | 0
Stage 6: Predictive performance | 5
Failed in more than 1 stage | 1

Five studies out of total 35 studies passed through quality assessment did not contribute to the data extraction process. Those papers are listed in Table 14 with the reason why the data were not extracted from those studies. Finally 30 studies were used in this SLR. Overview of studies is presented in the section 5.3.

**Table 14.** Papers not contributed to data extraction process

<table>
<thead>
<tr>
<th>Paper identifier</th>
<th>Reason for not extracting data</th>
</tr>
</thead>
<tbody>
<tr>
<td>A17, A20, A22</td>
<td>Performance values were given in plots. Authors of the papers were contacted to get the performance values of each plot. But no data were received.</td>
</tr>
<tr>
<td>A30, A31</td>
<td>Papers are still in the pre-print servers (not published). These two studies are related to CPDP and they were kept in the list, though they could be excluded based on inclusion and exclusion criteria. They were kept in the list to see if they were to be published by the time of extracting data.</td>
</tr>
</tbody>
</table>

### 5.3 Overview of studies

The majority of the CPDP studies were published during the past 5 years (Figure 6). It indicates that CPDP have been gaining interest in many researchers recently. Moreover, it is clear that many studies related to defect prediction model building in CP context started to publish with the study by Briand et al. in 2002. 43% of the studies were published in 2015, indicating that more contemporary studies were included in this SLR. Regarding the publication source, there were slightly more primary studies (53%) were published in journals than conferences (47%).

![Figure 6](image)

**Figure 6.** The distribution of studies over the years

Out of total 30 studies, 27 studies reported their performance values via categorical dependent variables and remaining 3 studies reported the performance values via continuous dependent variables. Extracted full set of data from 30 studies can be
found from the Appendix C, D and E. Appendix C consists of context data and Appendix D and E consists of qualitative data and quantitative data respectively (summarized in categorical model table and continuous model table). Summary of main findings and conclusions reported on studies present in the qualitative data table in Appendix D. The quantitative data consists of predictive performance values of individual models reported in the 30 finally included studies. Quantitative data are presented in two tables named categorical table and continuous table in Appendix E.

5.4 Types of Independent variables used in CPDP

Many different independent variables had been used in the 30 finally included lists of studies. As in the Categorical model table and continues model table in Appendix E, independent variables used in individual studies was categorized into different predefined sets of categories. Table 15 summarises the different types of independent variables used in the studies reported categorical models. As shown in Table 15, many categorical models had used a combination of different metric types, which is 85% of the studies. SCM (Static code metrics) + LOC (Lines of code) and OO + SCM + LOC were the common combinations used in literature. There were 5 studies (A13, A24, A37, A40 and A41) where multiple combinations were used for model building. Those combinations of metrics types had formed based on the training and test data sets used in each model. For an example, study A41 (Jing et al., 2015) had used SCM + LOC for defect prediction model building where the model was trained with NASA data set and test on SOFTLAB/ReLink data set. In addition to SCM + LOC combination, A41 also had used OO + SCM + Process + LOC combination of metrics types for models trained on AEEM data set and tested on SOFTLAB/ReLink data set. Four studies had used only one type of independent variable. It is noted that LOC had been used as a part of combination of metric types in 23 (85%) studies. Also object-oriented metric type had been used as a part of combination of metrics types in 55% of the studies. Three studies reported continuous models had mainly used process and OO metrics types when constructing models.

Table 15. Metrics used in studies

<table>
<thead>
<tr>
<th># of metrics</th>
<th>Type of metric</th>
<th>Studies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single type of metric</td>
<td>Source code text</td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td>OO</td>
<td>A7</td>
</tr>
<tr>
<td></td>
<td>SCM</td>
<td>A46</td>
</tr>
<tr>
<td></td>
<td>Process</td>
<td>A39</td>
</tr>
<tr>
<td>Combination of types of metrics</td>
<td>OO + LOC</td>
<td>A2, A8, A11</td>
</tr>
<tr>
<td></td>
<td>SCM + LOC</td>
<td>A4, A14, A15, A16, A20, A25, A26, A43</td>
</tr>
<tr>
<td></td>
<td>OO + SCM + LOC</td>
<td>A3, A5, A18, A23, A27, A42, A44</td>
</tr>
<tr>
<td>More than one combination of types of metrics</td>
<td></td>
<td>A13, A24, A37, A40, A41</td>
</tr>
</tbody>
</table>

Three sets of plots were drawn to compare performance of different types of independent variables. First set of plots were drawn using f-measure, precision and recall performance values (either reported on the study or calculated by researchers). Such models were reported in 13 studies. Second set of plots were
drawn using performance values reported in 10 studies in terms of AUC. For the third set of plots, recall, probability of false alarm and balance were utilized. Such models were reported in 11 studies. As it can be noted sum of the studies used for drawing plots are not equal to the total no of included studies, which is 27. It happened because there were some studies reported their performance values using more than one performance metric set used to draw plots. For an example, A25, A39 and A40 all studies had reported AUC along with f-measure, precision and recall, so that these studies were used to draw the plots in both first and second sets. This expanded the capability of performance comparison among large no different types of independent variables within one plot. Two studies reported only f-measure values were not taken for drawing plots. Whole process resulted in 25 studies reported categorical models being used for comparison of independent variables.

Figure 7 and 8 present the violin plots for model performance in relation to types of independent variables used in each study. Violin plots drawn using f-measure, precision and recall values reported as either average or original values are in Figure 7. Violin plot made using either average or original AUC performance values reported studies can be found from Figure 8. The rest of the plots are in Appendix K. Labels on the plots were taken from the “Independent variable categorization” column on the categorical model table in Appendix E.
Figure 7. Performance of different types of independent variables in terms of f-measure, precision and recall (Performance values are given as either original or average values)
There was a difference in performance between defect prediction models using different types of independent variables. Defect prediction models used combination of metrics types seemed performing well. As illustrated in the Figure 7, Figure 8 and other figures available in Appendix K, OO+LOC, OO+SCM+LOC and OO+SCM+Process+LOC combinations of metrics types seemed perform well. When considering models built upon only single type of independent variable, it showed that the source code text metric type used in the study done by Mizuno and Hirata (2014) equally performed well as OO metric type. In their study, they had used tokens extracted from specific parts of modules (e.g. code and comment) as source code text with fault prone filtering approach.

![Figure 8. Performance of different types of independent variables in terms of AUC (Performance values are given as either original or average values)](image)

Process metrics were not performing well. Kamei et al. (2015) had used process metrics such as NF (Number of modified files), Entropy, Relative churn, Relative LT (Lined of codes in a file before the change), FIX (Whether or not the change is a defect fix) to build defect prediction models and performance of those models had contributed to low performance of process metrics. The model used SCM such as complexity based metrics seemed perform better than models based on process metrics.

5.5 Modelling techniques used in CPDP

Various different modelling techniques had been used by the 27 studies reported categorical models. The majority of studies within that set of studies had used only base learner to build the model. There were few studies which had used various optimization techniques along with modelling technique. Appendix F shows how the studies were grouped according to the way of learner been used when building the defect prediction model.

NB was the most commonly used modelling technique on the studies reported categorical models which was 52%. LR had been the next frequently used modelling technique. Base learners such as DTree, SVM and RF also had been used to build prediction models in significant amount of studies.
There were 6 studies which had used different learners with boosting (e.g. A18, A25, A26, A42 and A44) when building their proposed CPDP approach. Moreover, A43 also used NB with boosting just to compare the performance of their proposed approach. Studies A18, A25, and A44 had used NB and boosting in their proposed CPDP approaches. Study done by Zhang, Lo, Xia and Sun in 2015 had made use of both NB and DTree with boosting as part of their proposed approach. Ryu, Choi and Baik (2014) had explored the applicability of SVM learner with boosting in their study.

There was only one study, which had chosen GP-based process to build optimal defect prediction models within the included set of studies reported categorical models (Liu, Khoshgoftaar & Seliya, 2010) (A14). Besides, A11 is the only study which proposed MO defect prediction model based on different forms of modelling techniques such as LR and DTree (Canfora et al., 2015).

Few studies were grouped under “Other ensemble” on the matrix in Appendix F (e.g. A4, A5, A8, A42 and A43). Ensembles of weak models were used by set of researchers in their study to avoid overfitting to the training data set (Uchigaki, Uchida, Toda & Monden, 2012) (A4). Models related to the Combined Defect Prediction (CODEP) in the study A8 were added under “Other ensemble”. CODEP assumed that the defect-proness of a particular class can be computed as combination of all the defect-proness probabilities obtained by the different defect prediction models (Panichella, Oliveto & Lucia, 2014). In the study A8, Panichella et al. combined 6 different learning algorithms using two underlined learning algorithms; LR and BN. In a similar manner Zhang et al. in 2015 (A40) investigated 7 composite algorithms which integrate various modelling techniques. Few defect prediction models reported on A5 grouped under “Other ensembles”. Models in A5 were built using DTree with generated train-test-result instances (He et al., 2012). The research conducted by Ryu, Jang and Baik (2015) (A43) followed a different ensemble approach. They proposed Hybrid Instance Selection Using Nearest-Neighbour (HISNN) method which performs a hybrid classification selectively learning local knowledge and global knowledge.

Figure 9 and 10 show the model performance in relation to the modelling techniques used. Remaining plots can be found from the Appendix L. As per the plots, overall, models based on NN and DTree seemed to perform well. The models used LR is not seem to be related to prediction models performing well. Furthermore, RF was also having low performance. Studies using SVM had relatively better performance than both RF and LR. Most commonly used NB seemed to having average performance among other modelling techniques.

Prediction models built upon learners with boosting had not performed as well as might be expected. Many studies reported that the models based on learners with boosting related to good performance (e.g. A18, A25, A26, and A44). Models categorized under “Other ensembles” also performed well, specifically with performance metrics such as AUC and balance. Similarly models used MO-optimization (e.g. A11) also seemed to be related to models performing well. Models used MO-optimization achieved best performance value with AUC where the plot was drawn with either median or original AUC values.
Figure 9. Performance of modelling techniques in terms of f-measure, precision and recall.
(Performance values were given as either average or original values)

When considering 3 studies reported continuous models, it was noticed that, all the studies used one or more type of regression models such as LR, Bayesian Ridge Regression (BRR), Support Vector Regression (SVR), Nearest Neighbours
Regression (NNR), Decision Tree Regression (DTR) etc. As mentioned in the 4.2.5 section, performance comparison on modelling techniques was not carried out for continuous models due to unavailability of common performance metric among studies.

*Figure 10.* Performance of modelling techniques in terms of balance, recall and probability of false alarm (Performance values were given as either median or original values)
5.6 Performance evaluation criteria used in CPDP

Various kind of performance evaluation criteria were used to gauge the performance of different CPDP models. These performance metrics were employed to evaluate and compare the performance of prediction models developed using numerous independent variables, modelling techniques and CP approaches. Table 16 presents the performance metrics used in the 27 studies which reported categorical models. Table 16 consists of description of performance metrics and definition/formula for performance metrics. Last column of the table shows the studies in which they were used. Majority of performance metrics in the Table 16 are compound measures which calculated by combining values of confusion matrix. Details of confusion matrix can be found from Appendix G. Some less commonly used performance metrics were grouped as miscellaneous. The miscellaneous group includes Type I Error (FP), Type II Error (FN), Error Rate, G-mean2, H-measure, PF (Difference between Precision and ratio of faulty units).

Few studies had used only one performance metric (e.g. A4, A24, A37, A46) to evaluate and compare the model performance. Three out of those 4 studies had used AUC and remaining study had used f-measure. It was noticed that majority of studies had utilised more than one performance metric.

Cost related measures such as inspection cost (A8, A11), cost effectiveness (A44), expected cost of misclassification (A14) were used in few studies. Those studies had argued that how important those measures for quality assurance work. Inspection cost used LOC to approximate the effort needed to analyse the classified faulty units. On the other hand cost effectiveness estimated the no of faulty units found when a developer inspect first n% of line of codes. Expected cost of misclassification had been calculated using no of type I errors, no of type II errors along with cost of type I misclassification and cost of type II misclassification.
Table 16. Performance evaluation criteria used in studies reported categorical models

<table>
<thead>
<tr>
<th>Performance evaluation criteria</th>
<th>Description</th>
<th>Definition/Formula</th>
<th>Studies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>Proportion of units correctly predicted as faulty</td>
<td>( \frac{TP}{TP + FP} )</td>
<td>A1, A2, A3, A5, A7, A11, A15, A19, A23, A39, A40</td>
</tr>
<tr>
<td>Recall (pd-probability of detection, Sensitivity, TPR-True Positive Rate)</td>
<td>Proportion of faulty units correctly classified</td>
<td>( \frac{TP}{TP + FN} )</td>
<td>A1, A2, A3, A5, A7, A11, A13, A15, A18, A20, A23, A25, A26, A27, A39, A40, A41, A43, A44</td>
</tr>
<tr>
<td>Accuracy</td>
<td>Proportion of correctly classified units</td>
<td>( \frac{TN + TP}{TN + FN + FP + TP} )</td>
<td>A1, A15</td>
</tr>
<tr>
<td>Probability of False Alarm (pf, False positive rate, Type I error rate)</td>
<td>Proportion of non-faulty units incorrectly classified as fault-prone</td>
<td>( \frac{FP}{FP + TN} )</td>
<td>A7, A13, A18, A20, A25, A26, A27, A40, A41, A43, A44</td>
</tr>
<tr>
<td>AUC (Area Under Curve)</td>
<td>Effectiveness of the model calculated by measuring the area under the ROC (Receiver Operating Characteristic curve). ROC curve is produced when combination of recall (y-axis) and Pf (x-axis) are plotted.</td>
<td></td>
<td>A4, A8, A16, A25, A26, A37, A39, A40, A46</td>
</tr>
<tr>
<td>F-measure</td>
<td>Harmonic mean of precision and recall</td>
<td>( \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} )</td>
<td>A1, A3, A5, A24, A25, A39, A40, A41, A42</td>
</tr>
<tr>
<td>G-measure</td>
<td>Harmonic mean of recall and (1-pf)</td>
<td>( \frac{2 \times pd \times (1 - pf)}{(pd + (1 - pf))} )</td>
<td>A18, A27, A40</td>
</tr>
<tr>
<td>Balance</td>
<td>Euclidean distance from the real (pd, pf) point to (pd=1, pf=0) in the ROC curve</td>
<td>( 1 - \sqrt{\frac{(0 - pf)^2 + (1 - pd)^2}{2}} )</td>
<td>A13, A20 A43, A44</td>
</tr>
<tr>
<td>MCC</td>
<td>A balanced measure of all true and false positives and negatives.</td>
<td>( \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN)}} )</td>
<td>A18, A40, A41</td>
</tr>
<tr>
<td>Cost related measures</td>
<td>Include Inspection cost, cost effectiveness and expected cost of misclassification</td>
<td></td>
<td>A8, A11, A14, A42</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>Type I Error (FP), Type II Error (FN), Error Rate, G-mean, H-measure, PF,</td>
<td></td>
<td>A1, A14, A26, A44</td>
</tr>
</tbody>
</table>
A depiction of the number of studies used each performance evaluation criteria is in Figure 11. According to the Figure 11, Recall was the most commonly used performance metric followed by Probability of False Alarm, Precision, F-measure and AUC.

![Figure 11. Studies using different performance evaluation criteria for CPDP models](image)

Three studies which reported continuous models had used Nagelkerke’s R², P (Precision) and RMSE (Root Mean Square Error). Also one study had used the statistical z-test. Description and definition of these measures can also be found from Appendix I.

5.7 Different approaches used in CPDP

There were different CPDP approaches being used in the literature. When summarising different CPDP approaches in the studies, it was observed that, researchers had used different data processing methods to address various data related issues. Following sub sections present those data related issues and how those issues were addressed by utilizing different data processing methods. Moreover, different CPDP approaches and their performance comparison across studies are also presented.

5.7.1 Data related issues

It was noticed that the majority of the studies reported categorical models had addressed different data related issues such as difference distribution in source and target data sets, class imbalance issue, highly skewed data, etc. Table 17 lists those data related issues and how those issues were addressed by using different data processing methods in each study. The last column of the table presents the studies which addressed each issue. When a study had given a specific name to their
proposed approach then those approach names are also captured in the last column of the table. See the values given in a bracket.

Four studies (e.g. A1, A5, A7 and A46) which had not addressed any specific data related issues were not added to Table 17. Below subsections detail out those commonly addressed data related issues by studies.

**Table 17. Data related issues addressed by studies**

<table>
<thead>
<tr>
<th>Issues addressed</th>
<th>How issue been addressed</th>
<th>Used in papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class imbalance issue</td>
<td>Data sampling</td>
<td>A18 (DTB), A26 (VCB-SVM), A39 (JIT), A44 (TCSBoost)</td>
</tr>
<tr>
<td></td>
<td>Equally Weighting instances</td>
<td>A23</td>
</tr>
<tr>
<td></td>
<td>Selective learning strategy</td>
<td>A43 (HISNN)</td>
</tr>
<tr>
<td>Difference in distributions in source and target projects (Effect of heterogeneity between different projects)</td>
<td>Transformation</td>
<td>A2 (Metric compensation), A18 (DTB), A24 (TCA+), A25 (TNB), A40 (Universal model), A41 (CCA+), A44 (TCSBoost)</td>
</tr>
<tr>
<td></td>
<td>Filtering</td>
<td>A13, A20 (Burak filter), A43 (HISNN)</td>
</tr>
<tr>
<td></td>
<td>Data normalization</td>
<td>A4, A8 (CODEP), A11 (MODEP), A14, A23, A24 (TCA+), A26 (VCB-SVM), A41 (CCA+), A44 (TCSBoost)</td>
</tr>
<tr>
<td></td>
<td>Metrics matching</td>
<td>A37 (HDP)</td>
</tr>
<tr>
<td>Highly skewed data</td>
<td>Logarithmic transformation</td>
<td>A4, A13, A20 (Burak filter), A43 (HISNN)</td>
</tr>
<tr>
<td>Privacy</td>
<td>Reduces the amount of data shared by using multi-party data sharing</td>
<td>A27 (LACE 2)</td>
</tr>
<tr>
<td>Irrelevant features and redundant features</td>
<td>Feature selection</td>
<td>A3 (TOP K), A15, A37 (HDP), A42 (Composite algorithm)</td>
</tr>
<tr>
<td>Missing feature values</td>
<td>Using average value imputation schema</td>
<td>A16</td>
</tr>
<tr>
<td>Large number of possible values of the continuous feature</td>
<td>Discretization</td>
<td>A16, A25 (TNB)</td>
</tr>
<tr>
<td>Collinearity among metrics</td>
<td>Remove highly correlated metrics /Normalization approach</td>
<td>A39 (Just-In-Time)</td>
</tr>
<tr>
<td>Noise of data sets</td>
<td>Outlier removing</td>
<td>A43 (HISNN)</td>
</tr>
<tr>
<td></td>
<td>Noise filtering</td>
<td>A27 (LACE 2)</td>
</tr>
</tbody>
</table>

**Class imbalance problem**

Usually in software defect data sets, number of defective modules is much smaller than the non-defective modules. This problem is termed as *class imbalance* (Ryu, Choi & Baik, 2014; Ryu et al., 2015). Class imbalance problem greatly impact on the performance of classification model (Ryu et al., 2015) as well as suitability of predictive performance measures. Table 16 illustrates the influence in class
imbalance issue over different performance measures. As denoted by Hall et al. in 2012, when a classifier is trained using imbalance dataset, learning from minority class is not an easy task. In such situation, classifier does not predict defective modules and it predicts all data points as belonging to majority class which are non-defective modules. The class imbalance problem was addressed by several studies using different methods such as data sampling (e.g. A18, A26, A39, and A44), equally weighting instances (e.g. A23) and selective learning strategy (e.g. A43). In the equally weighting instance method, the instances of the source data are weighted such that the total weight of the defect-prone instances equals the total weight of the not defect-prone instances. This process is used by Herbold (2013) to decrease the possibility of bias on predicting defect-prone instances. In selective learning strategy, instance selection strategy at different regions of instance space is adopted (Ryu et al., 2015). This method initially proposed by Raman & Ioerger (2003). The commonly used technique by studies; data sampling is described in section 5.7.3.

Difference in distributions in source and target projects

Different distribution of metrics in the source and target datasets is another challenge in CPDP. In some studies this issue was introduced using term data heterogeneity in software projects (A11, A37 and A41). As expressed by Canfora et al. in 2015 (A11), software projects are often heterogeneous because, they exhibit different software metric distributions. Contextual factors such as size, domain, programming language, etc. significantly affect the project’s heterogeneity (Canfora, et al., 2015).

Many machine learning algorithms work well under an assumption that the source and target data are drawn from the same feature space and the same distribution (Pan & Yang, 2010). This assumption typically holds for WPDP but it might not hold for CPDP (Nam et al., 2013). Due of this reason CPDP always does not work (Turhan et al., 2009; Zimmermann et al., 2009). In the literature, various methods such as transformation, filtering, feature matching and metric normalization were introduced to provide solution for different distribution in source and target data. More information about commonly used methods in studies such as transformation, filtering and metric normalization can be found from 5.7.2 and 5.7.3 sections.

Highly skewed data

As presented in the Table 17, few studies had addressed the issue with skewed feature data. Skewed feature data cause for ineffectiveness of data mining process (Menzies et al., 2007). Several studies (e.g. A4, A13, A20 and A43) addressed this issue by transforming feature values to their logarithms. A study done by Menzies et al. in 2007 stated that spreading highly skewed data sets evenly across the space from the minimum and maximum values significantly improve the effectiveness of data mining, because the distribution of log-transformed feature values fits better to the normal distribution assumption. Section 5.7.2 provides more information about log transformation.
**Irrelevant features and redundant features**

Irrelevant and redundant feature data affect the performance of learning algorithms. A3 A15 and A37, A42 studies had used feature selection technique to identify and remove irrelevant and redundant features. This process reduces the dimensionality of data and improves the effectiveness of the learning algorithm (Kotsiantis, Kanellopoulos & Pintelas, 2006). Section 5.7.2 presents the more details about feature selection methods used in the studies.

**Large number of possible values of the continuous feature**

Features taking numerical values (integer or real) or features with a linearly ordered range of values are termed as *continuous* features (Fayyad & Irani, 1993). Classification learning algorithms focus on learning in nominal feature spaces. But such learning algorithms could not be useful in many classification tasks that involved in continuous features (Dougherty, Kohavi & Sahami, 1995). Due to this reason, these continuous features should first discretize. Process of discretizing continuous features was used by Ma et al. in 2012 (A25) and Ma, Zhang, Chen, Zhao and Baesens in 2014 (A16). More information on discretization can be found from section 5.7.2.

**Noise of data**

Excessively deviate features from usual feature sets are known as outliers (Kotsiantis et al., 2006). Outliers from the source dataset distribution are regarded to be the noise of the source data set (Ryu et al., 2015). Performance of learning algorithm depends on the quality of the data used. When the classifier built from noisy data, accurate would be less (Teng, 1999). Consequently, these noise data should be removed. Outlier removing (A43) and noise filtering (A27) techniques were used by several studies to deal with the noise of the data.

Data related issues in the aforementioned paragraphs were addressed by utilizing various data processing methods in the set of included studies reported categorical models. Data processing methods used in each study were grouped into two main categories, namely, column processing methods and row processing methods based on which component that the data processing was conducted. Studies that had carried out any processing on features/metrics were listed under column processing methods (e.g. Transfer learning, feature selection) and studies those had done any data processing on instances (e.g. class, file, and module), were listed under row processing methods (e.g. filtering, data sampling). Then the studies were further categorized into sub groups under the column and row processing methods. Most commonly used data processing methods are described in below sections 5.7.2 and 5.7.3. Data processing methods used in three studies reported continuous studies are presented separately in section 5.7.4.

**5.7.2 Column processing methods**

Column processing techniques used in each proposed CPDP approach are listed in the Table 18 along with the study in which that particular technique is used. Commonly used column processing methods in the Table 18 are described in below sub sections.
Table 18: Column processing methods used in studies

<table>
<thead>
<tr>
<th>Column processing method</th>
<th>Studies used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformation</td>
<td>A2, A18, A24, A25, A40, A41, A44</td>
</tr>
<tr>
<td>Feature selection</td>
<td>A3, A15, A37, A42</td>
</tr>
<tr>
<td>Feature matching</td>
<td>A37</td>
</tr>
<tr>
<td>Metrics normalization</td>
<td>A8, A4, A11, A14, A23, A24, A26, A41, A44</td>
</tr>
<tr>
<td>Logarithmic transformation (Log-filtering)</td>
<td>A4, A13, A20, A43</td>
</tr>
<tr>
<td>Addressing missing feature values</td>
<td>A16</td>
</tr>
<tr>
<td>Discretization</td>
<td>A16, A25</td>
</tr>
<tr>
<td>Normalization for minimizing collinearity</td>
<td>A39</td>
</tr>
</tbody>
</table>

Transformation

As shown in the Table 18, transformation/transfer learning technique was used in several studies to overcome the effect of heterogeneity between source and target data. In some studies effect of heterogeneity between source and target data stated as distribution difference between source and target project sets (Chen et al., 2015; Nam et al., 2013).

When looking into each study, which used transformation methodology, it can be noticed that most of the transformation methodologies aimed to extract common knowledge from one data set and transfer it to another, and then defect prediction model trained on that transferred knowledge. For an example A18, A25 and A44 had modified source data using re-weighting data and then trained the model with modified source data (See Table 18). Ma et al. in 2012 (A18) proposed a novel CPDP method called TNB which is an instance-transfer approach by considering that prediction should be related to the distributional characteristics of the data sets being used. TNB model was trained on weighted training data sets according to the similarities between the source and target data (Ma et al., 2012). DTB proposed by Chen et al. in 2015 (A25) was another addition to transfer learning. DTB process first reshares whole distribution of the source data set using data gravitation to fit target data sets, and then transferred boosting algorithm employs a small amount of labelled WC data to eliminate negative instances of CC data (Chen et al., 2015). Transfer Cost-Sensitive Boosting (TCSBoost) approach proposed by Ryu, Jang and Baik in 2015 (A44) had used the similar transfer learning approach used by Ma et al. in 2012 and Chen et al. in 2015 i.e. re-weighting the source data to fit to the target data.

The study A2 and A41 had followed a different approach than the A18, A25 and A44. Study A2 had adjusted target data using the metric compensation method to make target data similar to the source data (Watanabe et al., 2008). CPDP study, A41 by Jing et al. in 2015, had made the data distribution of target data set similar to that of source data using transfer learning method named Canonical Correlation Analysis (CCA) based on unified metric representation (UMR).
Table 19. Transformation approaches in CPDP

<table>
<thead>
<tr>
<th>Transformation approach</th>
<th>Studies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modify source data to fit target data</td>
<td>A18, A25, A44</td>
</tr>
<tr>
<td>Modify target data similar to source data</td>
<td>A2, A41</td>
</tr>
<tr>
<td>Modify both source and target data distribution</td>
<td>A24, A40</td>
</tr>
</tbody>
</table>

The study conducted in 2013 by Nam et al. (A24) proposed another transfer learning approach using TCA which is a feature extraction technique for transfer learnings to make feature distributions in source and target projects similar. They had transformed both source and target data to the same latent feature space, and then built models on the latent feature space. Universal defect prediction model proposed by Zhang et al. in 2015 (A40) also made use of another variant of transformation called context-aware rank transformation to address the issue in difference in distribution of predictors (e.g. metrics). The universal defect prediction model was built from a large set of diverse projects. There were lots of variations in the distribution of metrics across projects. Therefore, they clustered the projects based on the similarity of the distribution of metrics, and derived the rank transformations using quantiles of metrics for a cluster. Then universal model was trained on transformed data. Study A40 also modified both source and target data in their transformation approach.

Feature selection

Feature selection is another column processing method used by few CPDP related studies (e.g. A3, A15, A37 and A42) in the included set of studies. Features are the metrics that use to build defect prediction models. As denoted by He et al. in 2015 (A3), performing a feature selection method is a sensible method to deal with large no of features. The feature selection method identifies a subset of features which can deliver better predictive performance. Feature selection can be classified into two; filters and wrappers. In the filter approach, irrelevant features are removed from the feature set before it is used by the learning algorithm. On the other hand wrappers use feedback from a learning algorithm to decide which feature(s) to include in building a classification model (He et al., 2015). Both A3 and A37 had used filtering approach for feature selection. In the study A3, feature selection was conducted with two different algorithms called CfsSubsetEval evaluator and GreedyStepwise in Weka (which is a collection of machine learning algorithms for data mining tasks) to select features from the original data set. Further, they used TOPK to represent the Top-k metrics determined by the number of occurrences of different metrics to further reduce the dimension of original data. Then they selected minimum subset of features by removing redundant metrics. Based on the predictive performance values achieved by CPDP models built, they recommended using filtering approach, if only high recall value is required. If higher recall and higher f-measure are required together, then they recommended using TOPK. Further, they suggested using a minimum feature subset if appropriate precision or high f-measure is required.

The study done by Nam and Kim in 2015 (A37), had used various feature selection approaches such as gain ratio, chi-square, relief-F, and significance attribute
evaluation. Their results revealed that feature selection approaches can remove irrelevant features to build a better defect prediction model. Yu and Mishra (2012) (A15) performed score test to find out significant predictors from the data sets they used to build the prediction model. The study A42 also had applied feature selection technique to select a subset of relevant features to improve the prediction performance, but they had not mentioned their feature selection approach in detail (Zhang et al., 2015).

**Metric normalization**

As shown in Table 18, normalization had been widely used in the CPDP studies. Two types of normalizations, namely the min-max normalization and z-score normalization (also known as data standardization) were used. Value ranges of metric variables are varied (Kamei, Monden, Matsumoto, Kakimoto & Matsumoto, 2007). So that majority of studies (e.g. A4, A8, A11, A23, A26, and A41) had used z-score normalization to reduce the data coming from different projects to the same interval [0, 1]. Ryu et al. (2015) (A44) also used the z-score normalization, but they had not mentioned the reason for using that.

\[ Z = \frac{X - \mu}{\sigma} \]  

(1)

Z-score normalization can be performed by using formula (1). In this formula, \( \mu \) is mean value of \( X \) and \( \sigma \) is standard deviation of \( X \). Study done by Nam et al. (2013) (A24) used both z-score normalization and min-max normalization (can be performed by using formula 2) to give all features of data set an equal weight. \( X_{min} \) and \( X_{max} \) are the minimum and maximum values of \( X \) respectively.

\[ Z = \frac{X - X_{min}}{X_{max} - X_{min}} \]  

(2)

All data sets were normalized and scaled in A14 to address the variations in data set size, but they have not mentioned which normalization method was used (Liu et al., 2010).

**Logarithmic transformation**

According to thesis author’s knowledge, for the first time in the CPDP context, Turhan et.al in 2009 (A20) had used the logarithmic transformation to minimize skewness of metrics data. This transformation replaces each numeric value \( N \) with \( \log(N) \). Following the recommendations given by Turhan et al. in 2009, several studies (e.g. A13, A43) published later in 2009 also used logarithmic transformation as a data processing method. A study done by Uchigaki et al. (2012) (A4) also converted metric value into its logarithm as a part of their normalization process.

**Discretization**

The large number of possible feature values cause for slow and ineffective process of inductive machine learning algorithms (Kotsiantis et al., 2006). As a remedy for
aforementioned issue, some studies have used discretization, which is a process of reducing the number of possible feature values of the continuous feature (Kotsiantis et al., 2006). Both the studies A16 and A25 had used Minimum Description Length (MDL) based discretization schema proposed by Fayyad and Irani in 1993. The supervised discretization algorithm proposed by Fayyad and Irani in 1993, used entropy minimization heuristic for discretizing the range of continuous-valued features to multiple intervals.

5.7.3 Row processing methods

Row processing techniques used in the proposed CPDP approach in each study reported categorical models are listed in the Table 20 along with the study in which that particular technique is used. Commonly used row processing methods; filtering, data sampling are described in below sub sections.

Table 20. Row processing methods used in studies

<table>
<thead>
<tr>
<th>Row processing method</th>
<th>Studies used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filtering (NN-filtering)</td>
<td>A13, A18, A20, A27, A43</td>
</tr>
<tr>
<td>Noise filtering</td>
<td>A27</td>
</tr>
<tr>
<td>Outlier detection</td>
<td>A43</td>
</tr>
<tr>
<td>Weighting instances</td>
<td>A23</td>
</tr>
<tr>
<td>Data sampling</td>
<td>A18, A26, A39, A44</td>
</tr>
<tr>
<td>Clustering</td>
<td>A23</td>
</tr>
<tr>
<td>No processing</td>
<td>A1, A5, A7, A46</td>
</tr>
</tbody>
</table>

Filtering

Filtering method used by several CPDP studies to address the issue in different distribution in source and target data sets. According to thesis author’s best of knowledge, Turhan et al. in 2009 (A20) are the earliest set of researchers who used filtering method in CPDP. Many of the studies in the CPDP literature have referred this filtering method as “Burak filter/relevancy filtering”. The main idea behind the filtering method was to collect similar instances together in order to construct a source data set that is similar to test data (Turhan et al., 2009). Burak filter was based on k-NN algorithm. This k-NN algorithm finds out most similar k samples from the source data set for every instance in a target data set according to their Euclidean distance. In Burak filter, k = 10 is picked.

By following recommendations by Turhan et al. in 2009, later their study (A13) published in 2013 also used the filtering method when building CPDP model (Turhan, Mısırlı & Bener, 2013). Recently published study by Chen et al. (A18) also took the advantage of filtering method to avoid irrelevance samples in the source data set. The similar filtering approach was carried out by Peters, Menzies and Layman in 2015. In their study, “Best (K)” procedure introduced by
Kocaguneli, Menzies, Bener and Keung in 2012 was utilised for tuning k to determine the best source data set for each test set.

Relevancy filtering method proposed by Turhan et al. was able to achieve low PF rate compared with other CP models not applying relevancy filtering. The approach HISNN employed filtering to further reduce pf and achieve high pd value (Ryu et al., 2015). In the HISNN approach, NN filtering based on a search ring of Min-Ham radius around each test data instance was performed. So that, only the source data instances inside each search ring remained in the source dataset.

**Data sampling**

In order to provide balance distribution of the data set, typically data sampling involves in modifying imbalance data sets by some mechanism (He & Garcia, 2009). As mentioned by Yap et al. (2014), there are three approaches to handling imbalanced data sets. In data sampling, imbalance data sets are handled in data level. There are two basic sampling techniques; Over-sampling and under-sampling (Yap et al., 2014).

In the over-sampling process, minority class samples are randomly duplicated. On the other hand, in the under-sampling, majority classes are randomly discarded to achieve equal distribution with the minority class (Yap et al., 2014). Both over-sampling and under-sampling methods were utilized in A26 and A44. The study, A18 had used only data over-sampling and A39 had used only data under-sampling to achieve better CPDP model performance.

Synthetic minority over-sampling (SMOTE) had been used in several CPDP studies (A18 and A44) as an oversampling technique. SMOTE generates new artificial minority class instances synthetically based on feature similarities to deliver more balanced class distribution (Chawla, Bowyer, Hall & Kegelmeyer, 2002). Tomek links introduced by Tomek in 1976 was used as an under-sampling method in the study by Ryu et al. in 2015 (A46).

There are few studies (See Table 20) which had not used any data processing in their proposed CPDP approach (e.g. A1, A5, A7 and A46). Both the studies A1 and A7, investigated applicability of different types of metrics in CPDP instead of proposing any specific CPDP approach (Mizuno & Hirata, 2014) (Singh et al.,2013). The study A1 had used source code text and the study A7 had used OO metrics to build the CPDP models. Similarly, study done by Singh and Verma in 2015 (A46) also investigated CPDP at an early stage, focusing on design metrics. A study done by He et al. (2012), built the prediction models with different training data sets selected through different approaches which were not involved in any data processing method. Further, when look at the both Table 18 and Table 20, it can be noted that some studies have made use of both column and row processing methods (e.g. A18, A23, A26, A43 and A44) while others take advantage of either one of them.

**5.7.4 Data processing methods in continuous models**

Studies reported continuous model building had used Principal Component Analysis (PCA) (A31) and Log transformation (A35) as column processing
methods. PCA was used in the study done by Nagappan, Ball and Murphy in 2006 to avoid problems due to multicollinearity which occurs when the metrics are strongly correlated. They selected four uncorrelated linear combinations of metrics using PCA and then model was built with those four principal components.

Simple log transformation was employed in A35 to make the metrics more comparable among projects using below formula (3). In the formula, \( x \) is a particular value of metrics, \( \hat{x} \) is the transformed value of \( x \), and \( b \) is the difference between the median of the transformed dataset and the median of the dataset to be transformed (Cruz & Ochimizu, 2009).

\[
x = \log(\hat{x} + 1) + b
\]

All available releases from most appropriate data set were used as training set in the study done by (Chen & Ma, 2015), then best outcome among the other projects were reported. Detail about selecting most appropriate data set was not given in the study.

### 5.7.5 CPDP approaches

Various CPDP approaches were used in the studies from which data were extracted. The majority of studies had given specific names for their proposed approach such as TNB, DTB, VCB-SVM (Value Cognitive Boosting- Support Vector Machine), etc. Those approaches comprised with one or more data processing methods listed in Table 18 and 20. CPDP approaches used in the studies reported categorical models were categorized into 6 main categories. Those are filtering, data transformation, clustering, mixed data, feature selection and other approaches. Approach category names were defined as the data processing method name for those approaches which used column/row processing methods mentioned in the 5.7.2 and 5.7.3 as their main CPDP approach. Appendix F gives overview of studies which used each approach. As per the approach categorization done, 74% of the models reported on studies were grouped under “Other approach”. Those models had used simple data processing methods such as data sampling, data normalization, log transformation, etc. Closely similar percentage of studies had reported models built using filtering (30%) and data transformation (33%) approaches. There were only three studies which had used mixed data for building models and four studies which had used the feature selection approach.

Approaches which are defined as the row/column processing method name are already discussed in 5.7.2 and 5.7.3. Remaining approach categories are detailed out on below sections.

**Clustering**

Multiple variations of clustering were used by few studies. The EM clustering method was one variant of clustering used by Herbold (2013) in study A23. By utilizing EM algorithm, he created clusters of the characteristic vectors of the candidate source data joined with the characteristic vector of the target data. He, then, selected those data sets as source data that are located in the same cluster as the target training data. NN-clustering was another type of clustering used in A24.
In NN-clustering, target data were selected as similar to source data with favour to distributional characteristics (Herbold, 2013).

WHERE is another clustering algorithm for finding a software artefact with similar attributes. In the study A11, Canfora et al (2015) used ‘local’ predictor based on WHERE clustering method which was initially proposed by Menzies, Butcher, Marcus, Zimmermann and Cok (2011) to compare the performance of their proposed MO defect prediction model.

**Mixed data**

In the mixed data approach, CP data are mixed with WP data and then the defect prediction model is constructed with the mix of CP and WP data (Turhan et al., 2013). In A13, Turhan et al. (2013) investigated the applicability of mixed data approach. They found that, when there is limited project history data, defect prediction models based on mixed data are worth to use, since they perform well as full WP data based models (Turhan et al., 2013).

DTB, the approach proposed by Chen et al. (2015) (A18) also used CP data mixed with 10% of WC data to train the model. Ryu et al. in 2015 explored the usefulness of mixed data in a situation where class imbalance issue also taken into consideration. In their experiment they trained the proposed model with 5, 10 and 25% of WP data with CP data and they found that there is no significant performance difference in the case of using 25 and 5% of WP data as well as 25 and 5% of WP data.

**Other approaches**

Appendix F illustrates that the majority of the categorical models reported in the included set of studies used approaches other than filtering, data transformation, clustering, mixed data, feature selection. Models reported on those studies had used various basic column and row data processing methods such as data normalization, logarithmic transformation, data sampling etc. There were some studies which they had not used any specific approach. Moreover it was noticed that majority of the models categorized under “Other approaches” group had been built to compare the predictive performance of proposed CPDP approach by the study. For an example, study done by He et al. (2015) (A3) had built prediction model using most suitable data sets selected in an exhaustive way to compare the performance of the prediction models built using their proposed feature selection CPDP approach.

Similarly, in A18, A25 and A43, NB modelling technique based simple models were built to compare the proposed approaches such as DTB, TNB and HISNN respectively (Chen et al., 2015) (Ma et al., 2012) (Ryu et al., 2015). The only study (A23) which reported a model trained with all available equally weighted training data also added to “Other approach” category.

The prediction models which did not use any data processing method also added to “Other approach” column (e.g. A2, A20, A24 and A37). Those models were utilized to compare the performance against the proposed approach by each study. All the models reported on studies such as A4, A5, A8, A11, A16, A14, A26 and A39 added also under “Other approaches”. They were using simple data processing methods rather than any specific approach given in the matrix in Appendix F.
Further, models reported on studies A1, A5, A7 and A46 were mentioned under “Other approach”, since those models had not used either any specific approach stated in the matrix or any other simple data processing method.

**Performance of CPDP approaches**

The performance of the approaches was compared under a specific modelling technique. Figure 12 and 13 present the performance of different CPDP approaches in models built based on NB and LR respectively. Performance values given as either median or original were used to draw those violin plots. Figure 14 illustrates the performance of different CPDP approaches on SVM based models. Violin plot on Figure 14 was built with combination of average and original performance values. Remaining violin plots are given in Appendix M.

Figure 13 shows that prediction models used filtering approach seemed perform well than the mixed data approach in NB based models. According to figure 14, the data transformation approach is also seemed to be performing well in SVM based models. All Figures 12, 13 and 14 evidence that the models categorized under “Other approaches” have less performance than the models used specific CPDP approach.

On contrast to performance of “Other approaches” models in Figures 12, 13 and 14, and Figure 29 and Figure 31 in Appendix M provide a good sign of performance of models categorized under “Other approaches”. Models based on NB categorized under “Other approaches” had performed better than the both filtering and data transformation. Also Figure 31 shows models based on LR categorized under “Other approaches” perform better than data transformation. Among the models made use of specific CPDP approaches, “Data transformation” approach seemed to be related to good performance.
Figure 12. Performance of CPDP approaches with modelling technique NB in terms of balance, recall and probability of false alarm (Performance values are given as either median or original values)
Figure 13. Performance of CPDP approaches with modelling technique LR in terms of AUC (Performance values are given as either median or original values)

Figure 14. Performance of CPDP approaches with modelling technique SVM in terms of f-measure (Performance values were given as either average or original values)
5.8 CPDP model performance vs WPDP model performance

In the total 27 included primary studies reported categorical models, there were 14 studies which compared their proposed CPDP model performance with WPDP model performance. Out of these 14 studies, there were 11 studies which had reported their performance with all 3 precision, recall and f-measure metrics. From that pool of 11, 9 studies which had reported either original or average performance values were used to create forest plots.

Figure 15 illustrates the forest plot for a quantitative summary of precision, recall and f-measure drawn using the performance difference between WPDP and CPDP models. As shown in Figure 15, there is no significant difference in performance between WPDP and CPDP. Moreover, it can be noted that the majority of the models reported in the studies has achieved better WPDP performance than the CPDP performance with precision and f-measure while a considerable amount of models reported in the studies have gained better CPDP performance than WPDP performance with recall.
Figure 15. Forest plots for analysis of performances of CPDP models vs WPDP models in terms of f-measure, precision and recall

Next chapter provides the detail analysis of results for each research question.
7. Analysis

This section provides answers to five defined research questions by synthesizing both qualitative and quantitative data collected in the data extraction phase. Following the similar approach used by Hall et al. (2012), when comparing model performance in relation to various factors (e.g. types of independent variable, modelling technique), model performance is discussed in two ways. First, performance within individual studies is discussed to figure out major impact on predictive performance of the model within a study and then model performance across studies is compared to get a bigger picture of how well the model performs across studies. Similar approach was followed for comparison for CPDP performance against WPDP performance.

6.1 Types of Independent variables used in CPDP (RQ1)

Various independent variables used in categorical models can be mainly categorized as traditional metrics (e.g. size and complexity metrics), process metrics (e.g. code delta, code churn) and OO metrics (e.g. coupling, cohesion and inheritance source code metrics). In addition to above 3 categories, there was a one study in the included set of studies which had used source code text (e.g. Mizuno & Hirata, 2014) (A1).

As mentioned in the section 4.2.5 section, 25 studies which reported categorical models were used to compare the performance of different types of independent variables. Analysis of CPDP model performance across 25 studies showed that source code text and OO metric types perform equally well. Adding LOC with OO as well as adding LOC with SCM improved predictive performance of the model. Defect prediction model performance comparison across the 25 studies revealed that process metrics have relatively poor performance. However, Kamei et al. (2015) (A39) asserted that the defect prediction models built with process metrics in the form of code changes have more advantages over traditional defect prediction models where predictions are made at a fine granularity. But their results showed that the WPDP models built with code changes outperforms CPDP models built with those metrics in terms of AUC, precision, recall and f-measure.

CPDP defect prediction models built only with OO metric type performed better than the other CPDP models built with only one metric type (e.g. source code text, process). The study conducted by Singh et al. (2013) (A7) had stated that the performance of cross company model built with Chidamber and Kemerer (CK) OO design metrics is comparable to within company model performance when consider overall results of their research. Moreover, Singh and Verma (2015) revealed that design phase metrics which are not belonging to OO category such as node_count, edge_count, and Mc-Cabe cyclomatic complexity measures etc. can be used as predictor variables in the CPDP context where no previous fault data are available.

The analysis of models in the 25 studies discovered that the models built with a combined range of different metric types perform best. For an example, Watanabe et al. (2008) (A2) and Canfora et al. (2015) (A11) used OO metrics together with LOC. Jing et al. (2015) (A41) also used a combination of OO, SCM, process
metrics types along with LOC. A combination of OO, SCM and LOC metrics types was utilised by several studies (e.g. A3, A5, A23 and A40). Combine the context factors with a range of variety independent variables also seemed to improve predictive power (Zhang et al., 2015). The study done by Zhang et al. (2015) showed that adding context factors such as programming languages, presence of issue tracking systems, the total LOC, the total number of files, the total number of commits, and the total number of developers further increases the predictive performance.

The usage of feature selection techniques on the set of independent variables appeared to improve prediction model performance (He et al., 2015, A3) (Nam & Kim, 2015, A37). Studies A15 and A42 also had taken the advantage of feature selection technique.

### 6.2 Modelling techniques used in CPDP (RQ2)

NB is the most commonly used modelling technique in CPDP context. Many studies in the included set of pool had constructed multiple models based on the different modelling techniques. Further, they had reported performance of different models built based on various modelling techniques.

When looked at the individual studies separately, it was not possible to see a clear picture of which modelling technique performs best in the CPDP context. Singh and Verma (2015) (A7) reported that J48 which is a C4.5 decision tree perform overall well than the other classifiers such as NB, SVM, RF, NN and DT. Further, they revealed that NB classifier performs better in CPDP context than in WPDP context in terms of precision. A study done by He et al. (2012) (A5) reported that best prediction results are provided by predictor based on J48. He et al. (2015) (A3) reported that simple classifiers (e.g. NB and LR) perform well in CPDP context with respect to the overall performance. Additionally, they asserted that, NB and BN are relatively stable when there are different metric sets in source and target data sets are available and DT and SVM are suitable in CPDP when higher f-measure values is needed. Liu et al. (2010) (A14) reported that GP-based models perform better than the non-GP models by comparing GP-based model performance against 17 non-GP models.

When considering individual studies, it was shown that the CPDP models based on learners with boosting provide some promising CPDP model performance values. For an example, Chen et al. (2015) (A18) reported that DTB algorithm based on NB and boosting improve the CPDP model performance by reducing negative samples in CP data. Research conducted by Ma et al. in 2012 revealed that TNB which used NB with boosting is more accurate in terms of AUC. Further, a study done by Ryu, Choi & Baik (2014) revealed that SVM with boosting can better classify defect-prone code units than existing CPDP methods not prepared for the class imbalance (e.g. model built with NB and boosting) and the class imbalance method not prepared for the CP learning (e.g. model built with SVM and boosting). Similarly Ryu et al. (2015) asserted that TCSBoost based on NB and boosting achieves better overall performance.
Detailed analysis of model performance across 27 studies reported categorical models revealed that CPDP model performance may associate with the modelling technique used. Further, modelling techniques such as NN and DTree seemed to be performing relatively well in CPDP context. GP-based models performed better than the majority of non-GP based models (e.g. RF, LR, CRM 114, SVM, NB etc.). Moreover, comparative analysis revealed that the models based on learners with boosting relate to low performance.

### 6.3 Performance evaluation criteria used in CPDP (RQ3)

Analysing performance evaluation criteria across 27 studies reported categorical models disclosed that the majority studies had computed confusion matrix and calculated various compound performance measures based on the matrix to evaluate the performance of prediction models. Few studies had employed only AUC to measure the model performance (e.g. A4, A16, A37 and A46).

<table>
<thead>
<tr>
<th>Performance evaluation criteria</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td></td>
<td>Influenced by imbalance data set</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Threshold-sensitive</td>
</tr>
<tr>
<td>Recall</td>
<td>Does not influence by imbalance data set</td>
<td>Threshold-sensitive</td>
</tr>
<tr>
<td>Accuracy</td>
<td></td>
<td>Influenced by imbalance data set</td>
</tr>
<tr>
<td>AUC</td>
<td>Does not influence by imbalance data set</td>
<td>Ignores the cost-effectiveness of the model</td>
</tr>
<tr>
<td>Probability of False Alarm</td>
<td>Does not influence by imbalance data set</td>
<td></td>
</tr>
<tr>
<td>F-measure</td>
<td>Trade-offs between precision and recall</td>
<td>Threshold-sensitive</td>
</tr>
<tr>
<td>G-measure</td>
<td>Does not influence by imbalance data set</td>
<td></td>
</tr>
<tr>
<td>Balance</td>
<td>Does not influence by imbalance data set</td>
<td></td>
</tr>
<tr>
<td>MCC</td>
<td>Does not influence by imbalance data set</td>
<td></td>
</tr>
<tr>
<td>Cost related measures</td>
<td>Give an insight on the effort required to analyse the identified defect-prone classes which helps to prioritize QA work and allocate resources</td>
<td></td>
</tr>
</tbody>
</table>

Table 21. Pros and Cons of performance evaluation criteria

Almost all the studies in the included set had proposed single-objective defect prediction models. Single-objective defect prediction simply recommends a set or a ranked list of likely faults-prone code units (Canfora et al., 2015). Canfora et al. (2015) (A11) reported that Multi-Objective Defect Prediction (MODEP) allows to achieve better cost effectiveness than single-objective predictors in both WP and CP contexts. MODEP tries to achieve a compromise between cost and effectiveness. Through this MODEP approach, software engineers can select predictors achieving compromise between number of likely fault-prone code
The majority of the studies reported categorical models had given justification for selecting various predictive performance criteria and only a few studies (e.g. A2, A3 and A4) had not defended their selected performance metrics. Some studies had selected particular performance metric(s), since they were the most commonly used performance metrics in the literature (e.g. A5, A7, A25, A39, A41 and A46). For an example, a study conducted by He et al. (2012) (A5) and Singh et al. (2013) (A7) had chosen f-measure as one of their performance metrics, reasoning that it has been widely used in the field of Information Retrieval (IR) and Data Mining. Similarly, AUC was used in the study A46 since it has been extensively used as a performance measure in other fields of research, such as communication (Radar), machine learning, financial and medical data mining application.

Remaining studies, which is 70% of the total set of studies reported categorical models had explained the process of choosing performance metrics with theoretical arguments. Those theoretical arguments were based on recommendations by authors of previous studies. For an example, the study A20, by Turhan et al. (2009) had used recall, Probability of false alarms and balance instead of using accuracy, precision. They had further mentioned that accuracy and precision are poor indicators of performance for data where the percent of defective class is very small. Those recommendations had initially given by the authors of previous studies (Menzies et al., 2007; Menzies, Dekhtyar, Distefano & Greenwald, 2007).

As shown in the Table 21, precision and accuracy are influenced by imbalance data sets. A13, A20, A25 and A27 studies had avoided using these two performance metrics due to their unbalance nature of data sets, i.e. no of defective modules is much smaller than the non-defective modules in the data set.

Further, some studies (e.g. A4, A16, A37, and A46) preferred to use only AUC or AUC with other performance metrics (A26), since it does not influence by class imbalance data sets. Moreover, unlike other traditional measures (e.g. precision, recall and f-measure), AUC is threshold-insensitive (A4, A37, A8) (Uchigaki et al., 2012; Nam & Kim, 2015; Panichella et al., 2014), so that AUC can be drawn by both the true positive rate and the false positive rate on various prediction threshold values. It makes the process of comparing performance values reported in the defect prediction literature (Nam & Kim, 2015) easy. See Appendix J for more information on threshold-sensitivity.

### 6.4 Different approaches used in CPDP (RQ4)

Analysing 27 categorical models reported studies revealed that a variety of approaches have been proposed in CPDP context. Various data related issues were addressed by those proposed approaches. Class imbalance issue and different data distribution in source and target data were the most commonly addressed issues by proposed CPDP approaches. Approaches such as DTB (A18), VCB-SVM (A26), TCSBoost (A44), JIT defect prediction model (A39) had used data sampling method to deal with the class imbalance issue.
Some proposed approaches (e.g. Burak filter (A20)) had used data from projects with similar distributions of predictors to the target project as source data. This process eliminated the issue with difference data distribution in source and target data sets. Further, some other proposed approaches had made an effort to transform predictors either in training or target projects or transform predictors in both training and target projects to make them more similar in their distribution. Approaches such as metric compensation (A2), DTB (A18), TCA+ (A24), VCB-SVM (A26), Large-scale Assurance of Confidentiality Environment 2 (LACE2) (A27), Universal model (A40), CCA+ (A42) and TCSBoost (A44) had used data transformation. In addition to data transformation, DTB (A18) and TCSBoost (A44) approaches had utilised mixed data when building CPDP models.

Turhan et al. (2013) (A13) had investigated the usage of mixed data in CPDP context. There was only one study in the included set, which proposed clustering (e.g. NN- and EM-clustering) as their CPDP approach. Feature selection had also been used by few studies. Heterogeneous Defect Prediction (HDP) approach proposed by Nam and Kim (2015) had conducted feature selection and feature matching to build a prediction model between projects with heterogeneous metric sets. Study A3, A15, A42 had also used feature selection as a part of their proposed CPDP approach.

Some basic data processing methods such as data normalization, logarithmic transformation, etc. had been used by the majority of proposed CPDP approaches. Those are some general data processing methods that can also be applied in other contexts as well. There were few studies (e.g. A1, A7 and A46) which had investigated some other various factors (e.g. performance with specific metric type) on model building rather than proposing a new CPDP approach.

When looked into individual study, it was noted that the authors attempted to justify the predictive performance of their proposed CPDP approach by comparing the proposed approach with few state-of-the-art CPDP approaches. For an example, He et al. who proposed DTB model in 2015 compared DTB model with other famous approaches in the CPDP context, such as TNB, NN-filtering and mixed data approach. They stated that, DTB performs statistically significantly better than all the other models in terms of g-measure.

Similarly, in A25, predictive performance of TNB approach was compared with filtering approach and found that TNB outperforms the filtering in the aspect of both AUC and f-measure. A study conducted by Jing et al. (2015) had compared their proposed CPDP approach with various state-of-the-art approaches such as CCA+ with TCA+, filtering, TNB. Some of their experimental results indicated that CCA+ is superior to state-of-the-art CCDP methods compared in terms of three widely-used measures namely pd, pf and f-measure. Even though some studies had provided comparison on few CPDP approaches, there was no clear consensus on which CPDP approach perform best.

Detailed analysis of model performance across studies supported to get a clear picture of performance of models with various CPDP approaches. The findings of the detailed analysis suggested that the performance of the model also links to the approach used. Overall, comparative analysis revealed that approaches which used data transformation perform well. Specifically CCA+ (A41), TNB (A25) and
universal model (A40) were related to good performance. The low performances of data transformation approach were caused by the various conditions used during model building. For an example, in A41, various TNB models were built with different conditions such as one-to-one (use all modules in only one project as the source data) Many-to-one (use all modules in multiple projects as the source data) etc. Moreover, in A24, various performance values were available for TCA depending on the data normalization method used.

Proposed approaches which had used feature selection also performed well with the LR learning algorithm. HDP proposed by Nam and Kim (2015) (A37) and the minimum feature subset selection approach proposed by He et al. (2015) caused for the good performance of the feature selection approach.

6.5 CPDP model performance comparison with WPDP model performance (RQ5)

No clear conclusion could make on CPDP model performance against WPDP model performance by just looking at the individual studies reported comparative performance of CPDP and WPDP model performance. There were some studies which supported for better WPDP performance against CPDP performance while some of them supported for better CPDP performance against WPDP performance.

Studies A15 and A39 reported WPDP have better model performance than CPDP performance. A study done by Yu and Mishra (2012) (A15) also revealed that WPDP model performance in terms of accuracy, precision, and recall is better than CDPD performance. Similarly, Kamei et al. (2015) (A39), a set of researchers who introduced JIT defect prediction models showed that WP JIT models outperform CP JIT models not only with f-measure, recall and precision but also with AUC.

Studies, A1, A3, A5, A11, A23, A40 and A41 had reported that CPDP model performance is better than WPDP performance with specific performance metric (s) (e.g. recall, AUC) or modelling technique (e.g. NB, J48). Study done by He et al (2012) (A5) asserted that source data from same project does not always lead to better prediction results and source data from other projects may provide better predictive performance results when most suitable source data set is used. Herbold in 2013 (A23) discovered that the CPDP performs better than within project predictions in terms of recall, but the CPDP model performance is worse in terms of precision. Singh et al. (A7) revealed that the performance of CPDP model based on NB and CK metrics better than the WPDP model performance. Moreover, they have showed performance on CPDP models based on J48 is comparable to the WPDP model performance.

Mizuno and Hirata (2014) (A1) revealed that WPDP models which use source code text as an independent variable achieve better precision than CPDP models. In contrast, they found that CPDP models show better recall. Study done by Zhang et al. (2015) (A40) mentioned that overall, there are clear differences in the performance of the universal model which made using large set of diverse projects and WPDP models with all measures they used except AUC. Specifically, the universal model yielded lower precision, larger false positive rate. But universal model achieved higher recall than WPDP models. Further they asserted that universal models as effective as WPDP models in terms of AUC which is
independent of cut-off values. CCA+, the approach for heterogeneous CCDP (HCCDP) also obtained either better or comparable prediction results as compared with WPDP results (Jing et al., 2015) (A41). The results of the study conducted by He et al. in 2015 (A3) indicated that WPDP models capture higher precision than CPDP models, while CPDP models achieve higher recall or F-measure than WPDP models. Canfora et al., (2015) (A11) revealed that, CP predictions are worse than WP predictions in terms of precision and recall. However MODEP introduced by them achieved a better cost-effectiveness than single-objective predictors trained with WP strategy. Watanabe et al. (2008) also reported WPDP performance values, but there was no comparison made with CPDP performance.

Analysing CPDP model performances with WPDP model performances across studies disclosed that overall WPDP models achieve better performance than CPDP models. Majority of WPDP models reported on studies had achieved good performance in terms of f-measure and precision. Majority of CPDP models reported on studies performed well in terms of recall.
8. Discussion and Implications

This section discusses the findings of each research questions on the SLR. Further, finding of SLR compare with existing studies.

7.1 Which types of independent variables have been used in CPDP and their performance? (RQ1)

Various different independent variables had been used when building prediction models in the context of CP. Those independent variables can be mainly categorized as traditional metrics, process metrics and OO metrics. In addition to those categories, source code text was also used as independent variables. Recent SLR on machine learning techniques in the general software defect prediction conducted by Malhotra (2015) reveals that there are only a few amount of studies (7%) which uses hybrid metrics, i.e. combination of traditional and OO metrics when building defect prediction models, but the results of this SLR show that there is a trend of using the combination of independent variables in the CPDP context. The majority of CPDP models had used a combination of metrics such as SCM + LOC and OO+ SCM+LOC rather than using only one type of independent variables.

Performance of CPDP models used a wide combination of metrics types performed better than the CPDP models used only one type of metric. OO and source code text metric types performed well in the CPDP context while process metrics did not particularly relate to good predictive performance. These findings are similar to what Hall et al. (2012) observes in their study. So that it can be mentioned that these findings are valid not only to CP context but also to general defect prediction context. With regard to performance of process metrics, finding of this SLR differ from the findings of the recent SRL on software fault prediction metrics conducted by Radjenović, Heričko, Torkar and Živkovič (2013). According to Radjenović et al. (2013), process metrics are successful at finding post-release defects while SCM metrics are not.

7.2 Which modelling techniques have been used in CPDP and their performance? (RQ2)

Variety of modelling techniques had been used in CPDP models. It was found that NB is the most commonly used modelling technique in the CPDP studies. The reason for NB being used commonly could be that it is a well understood and relatively simple technique. The study conducted by Malhotra (2015) reveals that NB has been frequently used not only CPDP context but also in general defect prediction context. In addition to NB, LR also been used in considerable amount of CPDP studies. Also, there is a trend of using learners with boosting in CPDP context.

Although Hall et al. (2012) assert that NB and LR based models perform comparatively well in general defect prediction models, results of this SLR showed that those learners may not be the best selection for CPDP models. NN and DTree
seemed to be performing relatively well in CPDP context. Moreover Malhotra (2015) asserts that RF performs the best among other machine learners such as NB, BN and MLP in general defect prediction. But in CP context, RF may not be a good option due to its low performance. Also GP-based models performed better than the majority of non-GP based models ascertaining the results obtained by Liu et al. (2010). Moreover, according to overall analysis, performance of models based learners with boosting is relatively low while CPDP models used MO-optimization performs relatively well.

7.3 Which evaluation criteria have been used in CPDP performance? (RQ3)

Various evaluation criteria had been used to measure the predictive performance of CPDP models. The majority of CPDP studies had used compound performance measures such as recall, precision, f-measure and probability of false alarm calculated based on confusion matrix and AUC. A study done by Malhotra (2015) reveals that, not only in CPDP context, but also in general defect prediction, recall, precision, f-measure and AUC have been most frequently used to measure the model performance. Additionally, cost related measures such as inspection cost, cost effectiveness and expected cost of misclassification also been used as performance metrics in CPDP context. MODEP seems promising approach in CPDP to achieve a compromise between cost and effectiveness of a model.

It was clear that the class distribution is one key factor which should take into consideration when selecting a performance metric in CPDP context. As revealed by Hall et al. (2012), selecting a performance metric depending on the class distribution of the training data set not only valid for CPDP context but also for general defect prediction context.

One challenge during performance comparison in relation to different factors such as independent variable, modelling technique and CPDP approach was finding a uniform set of measures for comparison. Due to that reason, performance comparison was done within group of studies which reported performance values with similar performance metric(s). This made the limit for getting overall picture of performance of certain factor among all studies. As recommended by Hall et al (2012), it is believed that usefulness of having uniform measure such as confusion matrix from which majority of other measures can be calculated in defect prediction studies in general.

Also it is highly recommended to provide predictive performance values in the study rather than only providing figures/plots of performance values. In this SLR, 3 studies (e.g. A17, A19, and A21) which passed quality assessment criteria were eliminated from data extraction phase due to unavailability of performance values on the study.
7.4 What are the different cross-project approaches used in CPDP to yield higher performance? (RQ4)

There had been various CPDP approaches proposed by studies. The majority of these proposed approaches comprised with one or more data processing methods such as row processing methods (e.g. data sampling, filtering, clustering etc.) and column processing methods (data transformation, data normalization, feature selection etc.). Class imbalance issue and different data distribution in source and target data sets were the commonly addressed data related issues by data processing methods in CPDP models. Filtering and data transformation were commonly used in model building in CP context. Also filtering approach seemed to be a good CPDP approach with simple learning algorithm like NB. For models based SVM, data transformation might be a good option than the NN or EM clustering. Feature selection approach for LR based models also good choice in CPDP context.

7.5 What is the performance of CPDP models compares to performance of WPDP models? (RQ5)

There were lots of arguments presented regarding to CPDP model performance against WPDP model performance in the existing literature. Some CPDP studies revealed that CPDP model performance is adequate and comparable to the WPDP model performance while some studies provided no evidence for CPDP has better performance than WPDP model performance, but the findings of this SLR showed that there is no significant difference in performance between CPDP models and WPDP models. Overall CPDP model performance is lower than the WPDP model performance. But CPDP models performed well in terms of recall. Various different CPDP approaches proposed by studies may have supported to achieve comparable CPDP model performance to WPDP model performance, so that it is fair to mention that CPDP is a viable option when there is no historical data to train the model.
8. Validity threats

Validity threats can influence the accuracy of research in a negative manner, so that it is important to identify the handle these threats. Below sections provides validity threats identified during the SLR.

8.1 Publication bias

Publication bias denotes the issue publishing positive results over negative results (Kitchenham & Charters, 2007). During the process of SLR, grey literature was excluded by making the assumption that good quality grey literature studies appear as journal or conference paper. Moreover, as stated by Kitchenham et al. (2010), grey literature is not being published because of publication bias, which takes place when studies presenting negative results are not published. They claim that this is not a problem anymore in software engineering field by giving examples of recently published meta-analysis studies reported fairly negative results.

8.2 Search term bias

Finding all relevant primary studies is always a challenge in any SLR. To address this issue, powerful search strategy was prepared as detailed out in the section 4.2.1 to find out as many as relevant primary studies. Wide search string was constructed with different search terms identified by checking titles and keywords in relevant papers already known. Also alternative spellings and synonyms for search terms were added by consulting the supervisor who is an expert in the area being studied. Search string was applied to full text of the paper. Moreover search string was piloted and resulting studies were compared with an already known list of papers and altered the search string accordingly. In addition to automated search of six electronic databases, additional search strategy, snowball sampling was also carried out to find out maximum no of relevant studies. All these facts were provided a confidence that the majority of the key studies were included.

8.3 Study selection bias

Study selection process was carried out in 2 phases. In the first round, studies were excluded based on the title and the abstract independently by two researchers. Pilot study of the selection process was conducted to place a foundation for better understanding of inclusion/exclusion criteria. Potential disagreements were resolved during the pilot study and inclusion/exclusion criteria were refined. Inter-rater reliability was evaluated to mitigate the threat emerged from researcher’s personal subjective judgment. Agreement between two researchers was “substantial” for selecting relevant papers from the full set of papers. The selection process was repeated until the agreement between two researchers achieves 100%. When the researchers could not make a decision on including a study, thesis supervisor was contacted. In the second phase studies were excluded based on the full text. Due to this well-established study selection process, it is so unlikely that any relevant study was missed. But the final exclusion of studies based on the full text was carried out by only one researcher, so that there is a possibility that study may have been erroneously excluded.
8.4 Quality assessment and data extraction

It is possible that some studies were missed which should have been included into the final set of 30 studies. Two researchers independently assessed the quality of each study. Quality assessment criteria were piloted and modified the criteria based on the pilot study results. Inputs from supervisor also were taken in the cases where two researchers could not come to an agreement on studies. All aforementioned actions mitigated the risk that missing any relevant study.

Due to the time constraints, data were extracted by one researcher and re-checked by another researcher. There were a few misunderstandings on how data should be extracted from studies. After the pilot study of data extraction, researchers were able to discuss and resolve those misunderstandings.

8.5 Violin plots

Some studies contributed data from many models to one violin plot, whereas other studies contributed data from only one model. As it can be noted number of data rows were varied in plots used to compare performance in relation to various factors. For an example, in model performance comparison in relation to modelling technique, RF plot was drawn using 223 data rows while DTree plot was drawn only using only 8 data rows. This may skew the results.
9. Conclusion

The main objective of this thesis was to summarise and synthesise the existing CPDP studies in order to identify what kind of different independent variable types, modelling techniques, performance evaluation criteria and approaches are used in building cross-project defect prediction models. Moreover, this study aimed to explore the predictive performance of cross-project defect prediction models compared to within-project defect prediction models. In order to fulfil the thesis objective, systematic literature review was conducted to answer 5 defined research questions. After a comprehensive analysis by following a systematic series of steps and assessing the quality of the studies, 35 studies were identified. By analysing the data extracted from 30 studies, research questions were answered. Main findings obtained from SLR are concluded sequentially according to defined research questions.

- Majority of the CPDP models constructs using combinations of different types of independent variables. The models used combinations of metric types perform well (e.g. metrics sets such as OO+LOC, OO+SCM+LOC and OO+SCM+Process+LOC). Metric types such as OO and source code text perform well in CP context while process metrics show comparatively low performance.
- NN and DT are the modelling techniques which perform well in CPDP. Most commonly used Naive Bayes (NB) seemed to be having average performance among other modelling techniques.
- Recall, precision, f-measure and probability of false alarm and AUC are the most frequently used performance metrics on CPDP context.
- Filtering and data transformation are also frequently used approaches in the cross-project context. The majority of the CPDP approaches address one or more data related issues using various row and column processing methods. Models seem to be performing well when filtering approach is used and model is built based on NB. Further, Models perform well with data transformation approach is used and model is built based on Support Vector Machine (SVM).
- There is no significant difference in performance of CPDP models compared with WPDP model performance. CPDP models perform well in majority cases in terms of recall.

Cross-project defect prediction model performance is influenced by the way it is being built. The predictive performance of the model is impacted by the independent variables used by the model, modelling technique on which CPDP models were built and the CPDP approach followed when building the model. Cross-project defect prediction still remains as a challenge, but they can achieve comparative predictive performance as within-project defect prediction models when the factors influencing the performance are optimized. The results of the SLR contribute to the existing literature by presenting:

- A set of 46 studies related to CPDP models. Researchers can use these studies as a foundation for future researches in cross-project (CP) context.
• A subset of 35 primary studies which passed through defined selection and quality criteria that permit these studies to be reliably analysed by software researcher or practitioner who intended to develop defect prediction model in (CP) context to make effective decisions.

• A synthesis of the current state-of-the-art in CPDP as reported in 30 studies.

There are some limitations which could impact on the scientific values of the thesis. One limitation is that, uncertainties appeared during the data analysis phase due to lack of knowledge in machine learning. For an example, when filling the modelling techniques vs CPDP approaches matrix, sometimes it was difficult to decide the best place that the model should appear. Majority of doubts was resolved with the discussion with supervisor and advisor. But still there could be some misjudgement in the analysis. Another limitation is that a simple approach was utilised to compare performance values. The model performance in relation to various factors (e.g. types of independent variables used, modelling techniques used, and CPDP approaches used etc.) was represented using violin plots. When comparing model performance in relation to a particular factor, other interacting factors which are likely to support for the performance of the model were neglected. For an example CPDP model performs well with a particular independent variable and a modelling technique than any one factor alone. More thorough analysis is needed to investigate factors impact on model performance. Furthermore, violin plots do not show the direction of any relationship between model performance and various model factors. For an example, it was not investigated that whether the CPDP approach perform well because it was used in a good model or whether model performs well because it used any particular CPDP approach. So that more comprehensive analysis should be conducted in the future to investigate factors affecting on predictive performance of the model.
References


assurance. *IEEE Transactions on Software Engineering, 39*(6), 757 - 773. doi:10.1109/TSE.2012.70


Pan, S., & Yang, Q. (2010). A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 22(10), 1345 - 1359. doi:10.1109/TKDE.2009.191


Zimmermann, T., Nagappan, N., Gall, H., Giger, E., & Murphy, B. (2009). Cross-project defect prediction: a large scale experiment on data vs. domain vs. process. Proceedings of the the 7th joint meeting of the European software
engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering (pp. 91-100). New York, NY, USA: ACM. doi:10.1145/1595696.1595713
Appendix A. Search Strings

ACM Digital Library

(("cross project" or "cross-project" or "multi-project" or "multiple project" or "mix-project") and (defect* or fault* or bug* or error*) and (predict* or estimat*) and (software))

IEEE Explorer

(("cross-project" OR "multi-project" OR "multiple project" OR "cross-company" OR "mix-project") AND (defect* OR fault*) AND (predict* OR estimat*) AND "software")

(("cross-project" OR "multi-project" OR "multiple project" OR "cross-company" OR "mix-project") AND (bug* OR error*) AND (predict* OR estimat*) AND "software")

ISI Web of Science

TS=("cross project" or "cross-project" or "multi-project" or "multiple project" or "cross-company" or "mix-project") AND (defect* or fault* or bug* or error*) AND TS=(defect* or fault* or bug* or error*) AND TS= (predict* or estimat*) AND TS = software

Google Scholar

"cross project" OR "cross-project" OR "multi project" OR "multi-project" OR "multiple project" OR "cross-company" OR "cross-company" OR "mix project" OR "mix-project" OR "mixed-project" "defect prediction"

"cross-project" OR "cross project" OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross-company" OR "mix-project" OR "mix project" OR "mixed-project" "fault prediction"

"cross-project" OR "cross project" OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross-company" OR "mix-project" OR "mix project" OR "mixed-project" "error prediction"

"cross-project" OR "cross project" OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross-company" OR "mix-project" OR "mix project" OR "mixed-project" "bug prediction"

"cross-project" OR "cross project" OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross-company" OR "mix-project" OR "mix project" OR "mixed-project" "defect estimation"

"cross-project" OR "cross project" OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross-company" OR "mix-project" OR "mix project" OR "mixed-project" "fault estimation"
"cross-project" OR "cross project" OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross company" OR "mix-project" OR "mix project" OR "mixed-project" OR "mixed project" "error estimation"

"cross-project" OR "cross project" OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross company" OR "mix-project" OR "mix project" OR "mixed-project" OR "mixed project" "bug estimation"

**Scopus**

ALL("cross-project" OR "cross project" OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross company" OR "mix-project" OR "mix project" OR "mixed-project" OR "mix project") AND ALL(defect* OR fault* OR bug* OR error*) AND ALL(predict* OR estimat*) AND software
<table>
<thead>
<tr>
<th>No</th>
<th>Reference</th>
</tr>
</thead>
</table>


<table>
<thead>
<tr>
<th>A29</th>
<th>He, P.; Li, B.;&amp; Ma, Y. (2014). <em>Towards Cross-Project Defect Prediction with Imbalanced Feature Sets</em>. Cornell University Library. (Status: Pass)</th>
</tr>
</thead>
</table>
## Appendix C. Context data table

<table>
<thead>
<tr>
<th>Aims of the paper</th>
<th>Application</th>
<th>Project</th>
<th>Size (KLOC)</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Investigated how to find an appropriate prediction model in the fault prone Brio method that predicts faults using tokens from source code modules.</td>
<td>Command-line tool Ant (1.4.1, 1.5, 1.6, 1.7)</td>
<td>463</td>
<td></td>
<td>Open source</td>
</tr>
<tr>
<td>Development</td>
<td>Eclipse (2.0, 2.1, 3.0)</td>
<td>673</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test editor</td>
<td>JIDE (3.2, 4.0.4, 4.1, 4.2, 4.3)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indexing and search technology</td>
<td>Lucene (2.0, 2.2, 2.4)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>API</td>
<td>PoL (1.5, 2.0, 2.5, 2.0)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creation and maintenance of open-source software</td>
<td>Velocity (1.4, 1.5, 1.6)</td>
<td>162</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ETL processor</td>
<td>Kalan (2.4, 2.5, 2.6)</td>
<td>162</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parse</td>
<td>Xerces (1.2, 1.3, 1.4)</td>
<td>467</td>
<td></td>
<td>Open source</td>
</tr>
<tr>
<td>Investigates the possibility of applying the prediction model, which is generated based on one project, to other projects, and have proposed compensation techniques for applying to other projects.</td>
<td>Text editor Sakura Editor</td>
<td>131</td>
<td></td>
<td>Open source</td>
</tr>
<tr>
<td>Validation of the feasibility of the predictor built with a simplified metric set for software defect prediction in different scenarios, and to investigate practical guidelines for the choice of training data, classifier and metric subset of a given project.</td>
<td>Text editor JIDE (3.2, 4.0, 4.1, 4.2, 4.3)</td>
<td>274</td>
<td></td>
<td>Open source</td>
</tr>
<tr>
<td>Aims of the paper</td>
<td>Command-line tool Ant (1.3, 1.4, 1.5, 1.6, 1.7)</td>
<td>501</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integration framework</td>
<td>camel (1.0, 1.2, 1.4, 1.6)</td>
<td>311</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dependency manager</td>
<td>svt (1.1, 1.4, 1.6)</td>
<td>174</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text editor</td>
<td>JIDE (3.2, 4.0, 4.1, 4.2, 4.3)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indexing and search technology</td>
<td>Lucene (2.0, 2.2, 2.4)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>API</td>
<td>poi (1.5, 2.0, 2.5, 2.0)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enterprise Service Bus (ESB)</td>
<td>synapse (1.0, 1.1, 1.2)</td>
<td>125</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creation and maintenance of open-source software</td>
<td>Velocity (1.4, 1.5, 1.6)</td>
<td>162</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ETL processor</td>
<td>Kalan (2.4, 2.5, 2.6)</td>
<td>162</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parse</td>
<td>Xerces (1.2, 1.3, 1.4)</td>
<td>467</td>
<td></td>
<td>Open source</td>
</tr>
<tr>
<td>Proposes prediction technique called &quot;an ensemble of simple regression models&quot; to improve the prediction accuracy of cross-project prediction.</td>
<td>Spacecraft instrument CM1</td>
<td>17</td>
<td></td>
<td>Industrial</td>
</tr>
<tr>
<td>Have addressed the issue with difference in distribution in metrics (using normalization + step 1: algorithm transformation; step 2: Z-score transformation).</td>
<td>-</td>
<td>457</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aims of the paper</td>
<td>Command-line tool Ant (1.3, 1.4, 1.5, 1.6, 1.7)</td>
<td>501</td>
<td></td>
<td>Open source</td>
</tr>
<tr>
<td>Integration framework</td>
<td>camel (1.0, 1.2, 1.4, 1.6)</td>
<td>311</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dependency manager</td>
<td>svt (1.1, 1.4, 1.6)</td>
<td>174</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text editor</td>
<td>JIDE (3.2, 4.0, 4.1, 4.2, 4.3)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indexing and search technology</td>
<td>Lucene (2.0, 2.2, 2.4)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>API</td>
<td>poi (1.5, 2.0, 2.5, 2.0)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enterprise Service Bus (ESB)</td>
<td>synapse (1.0, 1.1, 1.2)</td>
<td>125</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creation and maintenance of open-source software</td>
<td>Velocity (1.4, 1.5, 1.6)</td>
<td>162</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ETL processor</td>
<td>Kalan (2.4, 2.5, 2.6)</td>
<td>162</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Investigates defect predictions in the cross-project context focusing on the selection of training data.</td>
<td>Command-line tool Ant (1.3, 1.4, 1.5, 1.6, 1.7)</td>
<td>501</td>
<td></td>
<td>Open source</td>
</tr>
<tr>
<td>Integration framework</td>
<td>camel (1.0, 1.2, 1.4, 1.6)</td>
<td>311</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dependency manager</td>
<td>svt (1.1, 1.4, 1.6)</td>
<td>174</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text editor</td>
<td>JIDE (3.2, 4.0, 4.1, 4.2, 4.3)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indexing and search technology</td>
<td>Lucene (2.0, 2.2, 2.4)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>API</td>
<td>poi (1.5, 2.0, 2.5, 2.0)</td>
<td>211</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enterprise Service Bus (ESB)</td>
<td>synapse (1.0, 1.1, 1.2)</td>
<td>125</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creation and maintenance of open-source software</td>
<td>Velocity (1.4, 1.5, 1.6)</td>
<td>162</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ETL processor</td>
<td>Kalan (2.4, 2.5, 2.6)</td>
<td>162</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 16. Context data (Location of full table: https://drive.google.com/drive/folders/0BwCx6zvzZhi7RERVRHMcEh1NGs)
## Appendix D. Qualitative data

<table>
<thead>
<tr>
<th>Paper</th>
<th>Main findings and conclusions</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>Intra-project prediction yield better precision than cross-project prediction. But recall values are better in cross-project prediction.</td>
</tr>
<tr>
<td>A2</td>
<td>In the case of similar domain and similar size, it is possible to reuse the prediction model between languages, but precision/recall is not so high. Compensation based on metric value is effective, and it improves the recall value. Reuse direction has much effect on recall values.</td>
</tr>
<tr>
<td>A3</td>
<td>WPDP models generally capture high precision, whereas CPDP models always achieve higher recall and F-measure. There is no observable improvement when increasing the number of training data sets in WPDP. Therefore, for the training data of defect prediction, quality is more important than quantity. In WPDP, Naïve Bayes provides high recall and Decision Table maintains stable results for different metric sets. In CPDP, Naïve Bayes and Bayesian Network are relatively stable for different metric sets, and Decision Table and SVM become suitable as selections for high F-measure values. Different rules are proposed for WPDP and CPDP depending on the required performance. Simple classifiers (e.g., Naïve Bayes) tend to perform better on the whole when using a simplified metric set for defect prediction in all three scenarios. LOC, CBO, and LCOM are considered to be suitable components of the minimum metric subset. Metric subset CBO + LOC, as a subset of CBO + LOC + LCOM, not only has the highest Coverage value among the combinations with two metrics, but also can achieve a similar result that has no statistically significant difference compared with CBO + LOC + LCOM in terms of the Wilcoxon signed-rank test. CBO + LOC could be an alternative choice of the minimum metric subset for CPDP from a practical point of view, even though the combination CBO + LOC + LCOM has been validated as the minimum metric subset from a theoretical point of view. The implications of using a simplified metric set for defect prediction are effective reduction of the cost of data acquisition and processing by sacrificing a little bit of performance.</td>
</tr>
<tr>
<td>A4</td>
<td>Proposed method outperformed conventional multivariate logistic regression model in terms of AUC. Also proposed method is much more practically useful than the conventional method.</td>
</tr>
<tr>
<td>A5</td>
<td>In the best cases, prediction results provided by training data from other projects are acceptable on the average level. Training data from the same project doesn’t always lead to better prediction results than training data from other projects. To the contrary, for more than half of the releases, training data from other projects (most suitable training sets in scenario A) may provide better prediction results. The distributional characteristics of data sets are informative for training data selection. In addition, by constructing a decision tree on train-test-result instances generated from available historical data using the approach proposed above, we may find out cross-project training data for projects with prediction results comparable with the prediction results provided by data in the same project.</td>
</tr>
<tr>
<td>A7</td>
<td>When training is done by the NASA project for open source software then the false alarm rate (f) decreases and for probability of detecting (pd) increases when training is done by open source software for NASA project. In the case of different domain and different size, it is possible to reuse the prediction model between languages and projects of different companies. CK based object oriented metric value is effective for cross project fault prediction. In overall performance evaluation 148 has performed better than the other classifiers in pd, recall and F-measure. Results in terms of precision using CK metrics for fault prediction of cross company is better than the within company in case of null learner and is competitive in case of 349.</td>
</tr>
</tbody>
</table>

**Figure 17.** Qualitative data (Location of full table: https://drive.google.com/drive/folders/0BwCx6zxZhi1M3E2d2dXdXlqdm8)
### Appendix E. Quantitative data

<table>
<thead>
<tr>
<th>Paper</th>
<th>Independent variable granularity</th>
<th>Independent variable categorization</th>
<th>Performance evaluation criterion</th>
<th>Data set (Training data set → Test data set)</th>
<th>Causal Project/complex project (Variable y) (E/C/W)</th>
<th>Approach</th>
<th>Approach categorization</th>
<th>Performance values are given by (average, median, original)</th>
<th>AUC</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>Class level</td>
<td>Frequency of particular words</td>
<td>Precision, Recall, F-Measure, Accuracy</td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using single project</td>
<td>Other approaches</td>
<td>Average</td>
<td>0.358</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JdF</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Average</td>
<td>0.249</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>Lyric</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Average</td>
<td>0.303</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Average</td>
<td>0.285</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.514</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.013</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.307</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.614</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.723</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.677</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.569</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.459</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CRF 116</td>
<td>JnC</td>
<td>C</td>
<td>Prediction using all other projects</td>
<td>Other approaches</td>
<td>Original</td>
<td>0.172</td>
</tr>
</tbody>
</table>

**Figure 18.** Quantitative data: Categorical models (Location of full table: https://drive.google.com/drive/folders/0BwCx6zxZhjIlc25XZWPveEJubm8)
<table>
<thead>
<tr>
<th>Performance values values</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Approach</td>
<td>R2</td>
<td>Adjusted R2</td>
<td>Precision (Percentage of correctly predicted instances to the total number of test data)</td>
<td>RMSE</td>
</tr>
<tr>
<td>Component1</td>
<td>0.783</td>
<td>0.759</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component2</td>
<td>0.573</td>
<td>0.502</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component3</td>
<td>0.665</td>
<td>0.626</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 19.** Quantitative data: Continuous models (Location of full table: https://drive.google.com/drive/folders/0BwCx6zxZhjIlc25XZWpveJubm8)
## Appendix F. Modelling technique vs CPDP approach matrix

![Modelling technique vs CPDP approach matrix](https://drive.google.com/drive/folders/0BwCx6zxZhjIIVGowLUdwVkpYd00)

**Figure 20.** Modelling technique vs CPDP approach matrix (Location of full table: https://drive.google.com/drive/folders/0BwCx6zxZhjIIVGowLUdwVkpYd00)
Appendix G. Measuring predictive performance

This section presents the summary of gauging predictive performance of defect prediction models based on previous studies, (Bowes, Hall & Gray, 2012; Ostrand & Weyuker, 2007). The measurement of predictive performance is often based on the analysis of data in a confusion matrix shown in the below Table 22.

**Table 22. Confusion Matrix**

<table>
<thead>
<tr>
<th>Observed fault-prone</th>
<th>Predicted fault-prone</th>
<th>Predicted non fault-prone</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TP (True Positive)</td>
<td>FN (False Negative)</td>
</tr>
<tr>
<td>Observed non fault-prone</td>
<td>FP (False Positive)</td>
<td>TN (True Negative)</td>
</tr>
</tbody>
</table>

Confusion matrix reports four kinds of defect prediction results which are described below.

- True Positive (TP) /Type I Error: No of non-faulty units (files, modules or packages) classified as fault-prone
- False Positive (FP) /Type II Error: No of faulty units classified as non-faulty
- True Negative (TN): No of units correctly classified as faulty
- False Negative (FN): No of units correctly classified as non-faulty

Compound measures can be calculated by combining values from the confusion matrix. Recall, precision, probabilities of false alarm, accuracy, f-measure, g-measure, balance, MCC are examples for compound measures.
Appendix H. Calculating precision, recall and f-measures when error rates are provided

Wide variety of performance measures were reported in studies. For an example some studies had reported precision and recall while some other studies reported using recall and probability of false alarms. So that performance values need to be converted to some common metrics in order to compare their performance across studies. Calculation equations given in this section were taken from the study by Hall et al. (2012). Below section shows how to reconstruct a form of a confusion matrix where values are frequency of each instance:

\[ 1 = TP + TN + FP + FN \]  
(1)

Frequency of true class \( d \),

\[ d = TP + FN \]  
(2)

Then it is possible to calculate \( TP, FP, TN \) and \( FN \) as follows:

Given \( pf \) and \( d \):

\[ TN = (1 - d)(1 - pf) \]  
(3)

\[ FP = (1 - d)pf \]  
(4)

Given \( pd \) and \( d \):

\[ TP = d.r \]  
(5)

\[ FN = d(1 - d) \]  
(6)

Given type II error \( t_2 \), \( pf \) and \( d \) already have (1), (3) and (4)

\[ FN = \frac{pf(1 - d)t_2}{(1 - t_2)} \]  
(7)

\[ TP = 1 - FN - TN - FP \]  
(8)

Given Error rate \( er \), type II error \( t_2 \) and \( pf \):

\[ d = 1 - \frac{er(1 - t_2)}{pf} \]  
(9)

Then using (3), (4), (7) and (8), \( FP, FN \) and \( TP \) can be calculated. Now precision, recall and f-measure can be calculated:

\[ Precision = TP / (TP + FP) \]  
(10)
Recall = \frac{TP}{TP + FN} \quad (11)

F - measure = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} \quad (12)

There were considerable no of studies which had given model performance values using pd, pf. Also those studies had given balance of faulty and non-faulty units in data sets. Though they had given balance of faulty and non-faulty units in the training and testing data sets, it was not able to compute actual d value, because full data sets were not used when building defect prediction model. Some studies had selected test set randomly and also some studies had repeated experiment multiple times. Due to this reason, precision and recall values were not able to re-compute in those studies.
Appendix I. Performance evaluation criteria used in three studies reported continuous models

Table. 23. Performance evaluation criteria used in studies reported continuous models

<table>
<thead>
<tr>
<th>Performance Metric</th>
<th>Description</th>
<th>Definition</th>
</tr>
</thead>
</table>
| Nagelkerke’s $R^2$ | Percentage of correctly predicted instances to the total number of test data | $1 - e^{-LR/n}$  
Where LR is the likelihood ratio chi-square for the complete model and ‘n’ is the number of observations in the dataset. |
| Precision (P) | Measures the difference between the values predicted by a model and the values actually observed | $\frac{N_{\hat{r}=r}}{N}$ |
| RMSE | Measures the difference between the values predicted by a model and the values actually observed | $\sqrt{\frac{\sum_{i=1}^{N} (\hat{r}_i - r_i)^2}{N}}$  
Where $N$ is the number of instances in test data, and $N_{\hat{r}=r}$ indicates the number of instances whose predicted values ($\hat{r}$) are equal to their real values ($r$). |
| Z-test | | |
Appendix J. Threshold sensitivity

Defect prediction performance values reported in the literature are hard to compare since those prediction results are coming from the different cut-offs of prediction thresholds (Mende, 2010).

Cut-off probability (prediction threshold) is used to decide whether a code unit should be classified as fault-prone or not fault-prone (Lessmann et al., 2008; Song et al., 2011). Performance metrics such as recall, precision and f-measure which depend on the values of TP, FP, TN and FN in the confusion matrix require binary decision from the defect prediction model. But majority of the classification techniques use in the defect prediction model building give the probability of a code unit being fault-prone rather than producing binary decision. Due to that, cut-off probability is used to discretize this continuous probability into a binary decision. Any entity with a defect proneness probability greater than cut-off would be considered by the model as fault-prone, otherwise the model would consider the code unit as not fault-prone (Rahman et al., 2012; Lessmann et al., 2008).

In ROC curve is plotted by using recall on the y-axis and Pf on the x-axis on various prediction threshold values (Nam & Kim, 2015). Curve denotes the precision/recall pairing for all possible cut off values between 0 and 1. The area under the ROC curve (AUC) is a useful measure of model performance. A perfect model will have AUC of 1.0. A random prediction yields AUC of 0.5. In AUC, random predictor always has the value 0.5 irrespective of TP value. So that AUC is independent from threshold setting (Rahman et al., 2012).
Appendix K. Performance comparison plots for different types of independent variables

Figure 21. Performance of different types independent variables in terms of f-measure, precision and recall (Performance value are given as either original or median values)
Figure 22. Performance of different types of Independent variables in terms of AUC (Performance value are given as either original or median values)
Figure 23. Performance of different types of Independent variables in terms of balance, recall and probability of false alarm (Performance values were given as either average or original values)
Figure 24. Performance of different types of Independent variables in terms of balance, recall and probability of false alarm (Performance values were given as either median or original values)
Appendix L. Performance comparison plots for Modelling techniques

![Performance comparison plots](image)

**Figure 25.** Performance of modelling techniques in terms of balance, recall and probability of false alarm (Performance values were given as either average or original values)
Figure 26. Performance of modelling techniques in terms of f-measure, precision and recall (Performance values were given as either median or original values)
Figure 27. Performance of modelling techniques in terms of AUC (Performance values were given as either average or original values)

Figure 28. Performance of modelling techniques in terms of AUC (Performance values were given as either median or original values)
Appendix M. Performance comparison plots for CPDP approaches

Figure 29. Performance of CPDP approaches with modelling technique NB in terms of f-measure, precision and recall (Performance values are given as either average or original values)
Figure 30. Performance of CPDP approaches with modelling technique NB and boosting in terms of Balance, recall and probability of false alarm (Performance values are given as either average or original values)
Figure 31. Performance of CPDP approaches with modelling technique LR in terms of f-measure (Performance values are given as either median or original values)