PREDICTION OF DESIGN IMPACTFUL CHANGES IN MODERN CODE REVIEW USING MACHINE LEARNING

Amirthlingam.P

219183U

Thesis/Dissertation submitted in partial fulfillment of the requirements for the degree

MSc in Computer Science

Department of Computer Science and Engineering

Faculty of Engineering

University of Moratuwa

Sri Lanka

2024

DECLARATION

I declare that this is my own work and this thesis/dissertation does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other University or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date: 29/07/2024

The above candidate has carried out research for the PhD/MPhil/Masters thesis/dissertation under my supervision. I confirm that the declaration made above by the student is true and correct.

Name of Supervisor: Prof. Indika Perera

UOM Verified Signature

Signature of the Supervisor: Date: 29/07/2024

DEDICATION

This thesis is dedicated to my loving parents, whose unwavering love, encouragement, and support have been my guiding light throughout this academic journey. Your belief in me has been a constant source of inspiration, driving me to overcome challenges and strive for excellence. This achievement is as much yours as it is mine, and I am forever grateful for your presence in my life.

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my supervisor, Prof. Indika Perera, for the invaluable guidance, support, and patience throughout the entire process of conducting this research and writing this thesis. The expertise, encouragement, and constructive feedback were instrumental in shaping this work. I am also thankful to University of Moratuwa for providing the necessary resources and facilities for this study. Additionally, I extend my appreciation to the staff and faculty members whose assistance and expertise enriched my academic journey.

I am indebted to my family for their unwavering love, understanding, and encouragement. Their constant support sustained me during moments of doubt and challenge. Lastly, I extend my heartfelt thanks to all my friends who stood by me during this endeavor, offering encouragement, motivation, and occasional distractions when needed. This thesis would not have been possible without the support and contributions of the aforementioned individuals and institutions. However, any errors or shortcomings in this work are solely my responsibility.

ABSTRACT

In today's era most of the companies are using modern code review as one of the most useful technique to monitor the software changes continuously and to improve the software quality. The major purpose for this activity is to detect the design impactful changes as soon as possible that prevent the design degrading of the software after each and every code review.

Anyhow, the modern code review techniques are not enough to prevent the design degradation even after the degradation symptoms results in the rejection of changes done in the code. The design degradation occurs when smells occur due to the symptoms of bad structural decisions that occurs by a change. Technical and Social features may influence the design degradation when it comes to code reviews.

There is no any research that investigates the benefits of the prediction of design impactful changes to the stakeholders of code review using the social and technical features. Using the data provided by the Code Review Open Platform (CROP), an open-source dataset that links code review data to software changes I have used a machine learning approach to predict the design impactful changes with social and technical features in modern code review.

At the end I have evaluated the machine learning algorithm and extracted number of social and technical features and assessed how those features influences the code review. According to the literature review I have chosen Gradient Boosting algorithm. Using technical features leads to more accurate predictions. Anyhow, the social features that are available before the start of a review also be a good aspect for a highly precise prediction. Therefore we can say that technical and social prediction models can be used in the inspection of the changes done in software.

Expanding on the use of modern code review techniques to detect design impactful changes, the limitations of these techniques, the need for incorporating social and technical features, and the implementation of machine learning algorithms for prediction. Modern code review techniques are widely adopted by companies to monitor software changes continuously and enhance software quality. One of the primary objectives of code review is to detect design impactful changes promptly to prevent software design degradation after each code review iteration.

Despite the effectiveness of modern code review, it may not always suffice to prevent design degradation. Even when changes exhibiting degradation symptoms are rejected during code review, design degradation can still occur. This degradation typically arises from poor structural decisions manifested as code smells introduced by changes. Both technical and social factors play significant roles in influencing design degradation during code review. Technical features encompass aspects like code complexity, cohesion, and coupling, while social features include factors such as developer experience, communication dynamics among team members, and team collaboration.

To address the challenge of predicting design impactful changes, machine learning approaches are employed. Utilizing the Code Review Open Platform (CROP) dataset, which links code review data to software changes, machine learning algorithms are trained to predict design impactful changes based on a combination of social and technical features. Among various machine learning algorithms, Gradient Boosting is chosen for its effectiveness in handling complex datasets and producing accurate predictions. Through rigorous evaluation, it's found that incorporating technical features leads to more precise predictions. However, social features available before the start of a code review also prove to be valuable in enhancing prediction accuracy.

In conclusion, the integration of social and technical features into machine learning models facilitates more accurate prediction of design impactful changes during code review. By leveraging both technical and social aspects, stakeholders can gain insights into the potential impact of software changes on design quality, thereby improving the overall effectiveness of code review processes.

Keywords: modern code review, machine learning, design changes

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
NSD	Count of code snippets removed during the code modification process
NSU	Count of code snippets modified during the code modification process
NSA	Count of code snippets inserted during the code modification process
NFA	Count of files inserted during the code modification process
NFD	Count of files removed during the code modification process
NLA	Count of lines inserted during the code modification process
NLD	Count of lines removed during the code modification process
CHURN	Number of lines interested to and deleted in the change of code.
IMPR	The code change description contains "improve" word
BUG	The code change description contains "bug" word
REFC	The code change description contains "refactor" word
FEAT	The code change description contains "feature" word
DOC	The code change description contains "document" word
ML	Description's word count of the change of code
NLANG	Programming language count in the change of code
ME	Changed code's distribution across the files during the code modification process
NFT	Count of types of Files in the modified code

NCF	Count of files modified in the modified code
NMD	Number of directories changed in the change of code
FD	Number of developers involved in the code change
FM	Number of modification times of the code change file before
NWIC	Total word count of the general comments in the change of code
PWIC	NWIC weighted by using the general comment count
NWGC	Total word count of the inline comments in the change of code
PWGC	NWGC weighted by using the inline comment count
NIC	Inline comment count that are made by the reviewers
NGC	General comment count that are made by the reviewers
DL	Sum of Inline comments and General comments count that are made by the reviewers
NR	The count of code changes in the past that are assigned to the code owner whose code changes to be inspected
NC	The count of code modifications in the past that are submitted by the person who owns the code
NRC	The count of code modifications in the past 120 days that are submitted by the person who owns the code
NDC	The count of code modifications in the past that are submitted by the person who owns the code that contains at least a folder that has been impacted by the code changes
MR	The merge rate of code changes in the past that are submitted by the person who owns the code
RMR	The merge rate of code changes in the past 120 days that are submitted by the person who owns the code

DMR	The merge rate of code changes in the past that are submitted by the person who owns the code that contains at least a folder that has been impacted by the code changes
SB	The total of all pairs shortest path which go with v as a fraction
SKC	The maximal subgraph which consists of degree k's nodes or more
SD	The centrality of degree for node v that fraction of nodes which are connected through v
SCLUST	The geometric weight of subgraph edges as average
SCLOS	The total count of all the distances to all the other nodes as inverse
SE	The centrality of the neighbors based centrality of a node

LIST OF APPENDICES