Declaration of Authorship

I, Naina, declare that this thesis titled, " Elicitation of Non Functional Requirements from Catalog using Machine Learning Techniques" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this project has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this project is entirely my own work.
- I have acknowledged all main sources of help.
- Where the project is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

[illegible] Type text here

Naina
Registration No.:41500146
School of Computer Applications
Lovely Professional University
Phagwara, Punjab, India
Date: March 2023

ELICITATION OF NON FUNCTIONAL REQUIREMENTS FROM CATALOG USING MACHINE LEARNING TECHNIQUES

Thesis Submitted For the Award of the Degree of

DOCTOR OF PHILOSOPHY

in

Computer Applications

By

Naina

41500146

Supervised By	**Co-Supervised by**
Dr. Anil Sharma	**Dr. Amardeep Gupta**

LOVELY PROFESSIONAL UNIVERSITY

PUNJAB

2023

Certificate

This is to certify that the thesis entitled "Elicitation of Non Functional Requirements from Catalog using Machine Learning Techniques", which is being submitted by Ms. Naina for the award of the degree of Doctor of Philosophy in Computer Applications from the Faculty of Technology and Sciences, Lovely Professional University, Punjab, India, is entirely based on the work carried out by her under our supervision and guidance. The work reported, embodies the original work of the candidate and has not been submitted to any other university or institution for the award of any degree or diploma, according to the best of our knowledge.

[illegible]

Dr. Anil Sharma
Professor
School of Computer Applications
Lovely Professional University
Phagwara, Punjab, India
Date:March 2023

[illegible]

Dr. Amardeep Gupta
Principal
D A V College
Amritsar
Punjab, India
Date:March 2023

Abstract

Requirement Engineering plays a significant role in software engineering. The success and failure of the software highly depends on the requirements. The requirements are broadly categorised into Functional Requirements (FRs) and Non-Functional Requirements (NFRs). FRs define a function or service that is provided by the system. FRs are used to build the application architecture of the system. FRs deal with “What a system is supposed to do”. NFRs deal with ”How a system is supposed to do”. NFRs are the expected features from the software, which is also known as quality requirements. The technological architecture of the software is derived from NFRs. NFRs are the restrictions on the functionality of the system. NFRs are equally important as FRs. But NFRs are considered second-class requirements even though the success and failure of a project depend on NFRs. Moreover, it is very crucial to elicit the NFRs during the initial phases of software development. It is present in state-of-the-art research work that Machine Learning is the key to all software engineering problems.

Machine Learning is a vital subject of research in today’s world that performs tasks without being explicitly programmed. ML algorithms are proven to be very successful in various fields of software engineering domain such as defect prediction, software reliability prediction, reusability prediction, maintenance effort prediction, requirements classification, and, requirement prediction. Among all the problems, requirements classification and prediction are considered one of the most significant problems of software engineering.

The present study presents an automated system for efficient NFRs classification and prediction. The proposed system consists of 4 layers for the prediction of NFRs significance, including the Data Acquisition (DA) layer, Feature Selection and Extraction (FSE) layer, Data Prediction (DP) layer, and Data Analysis and Visualization (DAV) layer. In the initial DA layer, various NFRs elicitation techniques are discussed, such as Questionnaires, Domain Knowledge, Brainstorming, Templates, and Scenario. The elicitation techniques can be used as a single technique as well as a combined approach. In the current layer, domain knowledge and questionnaire has explored. A questionnaire based on the educational sector software from a usability perspective is designed. It consists of various questions which are taken from the literature review and the domain experts. The questionnaire is designed iteratively and suggestions from the various IT industry experts are taken for the drafted questions. The changes they suggested were incorporated iteratively and followed till the final questionnaire was developed. Finally, a mature catalog of NFRs was attained which has a maximum number of perfect and modified NFRs. The questionnaire was distributed to academicians, IT companies personnel, Social media, and personal contacts to acquire the dataset. The Min-Max scaling is applied and data is scaled between 0 and 1. The data augmentation technique SMOTE is applied for oversampling and balancing the dataset. The FSE layer selects and extracts the significant features from the set of all features. A hybrid feature selection algorithm is designed which consists of a filter and wrapper method for the classification of the dataset into significant and non-significant groups. The significant NFRs are the essential NFRs that are having a detrimental impact on software development. The Non-Significant NFRs contain a set of parameters that do not greatly affect software development. Initially, the correlation of the filter-based approach is applied and it attained 27 features as Set-1 from all the features. Then the information gain, another filter-based approach is applied to the set of features and got 27 features as Set-2. After applying the intersection between Set-1 and Set-2 and Set-3 is achieved which consists of 22 features. Finally, the wrapper method with backward selection is applied and 17 features were achieved as the final set. Based on these significant features, the ML-based model is constructed. The algorithm for NFRINDEX is presented in the proposed model. Moreover, the current research considers the probability measure of the level of NFR significance in terms of LoNFRS, which is cumulatively quantified as the NFRs significance measure (NFRINDEX). NFRINDEX has been quantified for prediction purposes using a Multi-scaled Long Short Term Memory(M-LSTM). It combines two techniques namely Convolutional Neural Network and Long Short Term Memory. Additionally, the existence of NFRs is visualized based on a Self Organized Mapping (SOM) technique. To validate the current system, a primary dataset is collected from several IT professionals and academicians as well a secondary dataset from Open Data Umea has been explored. Numerous experiments were performed to assess the performance in terms of Classification, Root Mean Square Error, Coefficient of Determination, Reliability, Stability, and Prediction Efficiency. The average level of data completeness of the questionnaire was 87%, which is significant in the current research domain. The presented model efficiently and effectively performs the classification with 94.56% Precision, 93.67% Specificity , 93.52% F1 score, and 94.57% Recall depicting enhanced efficacy. The prediction efficiency is given in terms of Accuracy, Coefficient of Determination, and Root Mean Square Error. The current approach has attained an Accuracy of 97.04%, which is higher than the other Base-line and Ensembling techniques. The average Coefficient of Determination measures 94% for the proposed model and outperforms other techniques. The current scenario shows the least Root Mean Square error than other techniques. For the dataset of 1000, 2000, 3000, 4000, and 5000 instances, the Root Mean Square Error values are registered as 2.4%, 1.9%, 1.1%, 1.5%, and 1.8% respectively, which is considerably less than other techniques. The given model has a 94.78% reliability score which is higher than other techniques. With the increase of the dataset, the reliability score also improves, making it suitable for large datasets. The proposed system is highly stable for NFRs classification. The average value of the Mean Absolute Shift (MAS) measure used for stability analysis is 70% over 5000 data instances. The results have shown 43% as a minimum and 79% as the maximum MAS value for the presented system.

Acknowledgements

I humbly bow my head to the feet of my adored Satguru S. Gurinder Singh Dhillion Ji, to whom I present and dedicate this thesis, and thank my supervisors, family, friends, colleagues, and well-wishers for their unparalleled support. Thanks to both of my supervisors, I've been able to channel my energy in the appropriate direction because of their constant support and guidance. In the hands of Dr. Anil Sharma, a Midas touch brought out the best in my hard work. As a student of Dr. Amardeep Gupta, a man of letters and an excellent teacher, I was fortunate enough to learn from a master of his craft. As a student, I also like to thank Dr. Kuljeet Cahal, and Dr. Kawaljeet for their contributions and support. My guides, each of whom had a unique perspective and experience, were an invaluable resource.

DAV College, Amritsar Principal Dr. Amardeep Gupta has been a constant source of encouragement and inspiration for my studies. Professor Vikram Sharma, the head of the computer science department at DAV College, Amritsar, has been instrumental in helping me complete my project. I could write a book to express my gratitude to my elder sister, Dr. Loveleen Kaur, Assistant Professor at DAV College, Amritsar, who not only set me on this route of research, but also remained involved and absorbed throughout my entire project.

My husband Gagandeep Madaan deserves a particular thank you for his unwavering support and encouragement throughout my career. He's always had faith in my abilities, but never more than now. My two sons, Heranaya Madaan and Havish Madaan, gave up many of their childhood luxuries in order to help me write this thesis. I'll always feel bad that I didn't let them to treasure more of their special childhood memories.

This has been an overwhelming experience, and I am unsure how I can adequately express my gratitude to my family members.

[illegible]

Naina

Date:March 2023

Contents

List of Figures

List of Tables

Abbreviations

ANFIS	Adaptive Neuro-Fuzzy System
ANN	Artificial Neural Network
BBM	Bayesian Belief Model
CNN	Convolutional Neural Network
COD	Coefficient of Determination
DA	Data Acquisition
DAV	Data Analysis and Visualization
DL	Deep Learning
DNN	Deep Neural Network
DT	Decision Tree
DTR	Decision Tree Regression
FP	False Positive
FRs	Functional Requirements
FS	Feature Selection
FSE	Feature Selection and Extraction
GBDT	Gradient Boosted Decision Tree
GRU	Gated Recurrent Unit
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
KNN	K-Nearest Neighbor
LSTM	Long Short-Term Memory

MAS	Mean Absolute Shift
M-DL	Machine-Deep Learning
ML	Machine Learning
MLP	Multilayer perceptron
NFRs	Non Functional Requirements
Pr	Precision
RE	Requirement Engineering
RFR	Random Forest Regression
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
Se	Sensitivity
SMOTE	Synthetic Minority Over-Sampling Technique
SOM	Self Organized Mapping
Sp	Specificity
SVM	Support Vector Machine
SVR	Support Vector Regression
TP	True Positive

Dedicated to my beloved family…

Chapter 1

Introduction

Software Engineering plays a significant role in the successful development of projects and in improving software production. It has been employed to meet the requirements of the project. Requirements are responsible for the performance or failure of software. In the Software Engineering community, the term requirement has been around the 1960s [1]. According to Sommerville and Sawyer, "Requirements are a specification of what should be implemented. Such are descriptions of system property or attributes. It deals with how the system should behave or constraints on the development process of the system" [2]. According to Ross and Schoman "requirements definition must say why a system is needed, based on current or foreseen conditions, which may be internal operations or an external market. It must say what system features will serve and how the system is to be constructed" [3]. There are numerous examples of software failure which serve as evidence to establish the significance of requirements [1][4][5]. The software requirements are segregated as Functional Requirements(FRs) and Non Functional Requirements(NFRs). The FRs refer to what is to be done whereas NFRs deal with how is to be done. The inappropriate management of requirements is the main reason for software failure [6]. Moreover, inappropriate handling of NFRs leads to rejection of software [7]. One of the approaches aiming to improve the process of elicitation and prediction of requirements is Machine Learning(ML). ML is a branch of artificial intelligence that enables the system to learn and predict accurately without being explicitly programmed. It is very significant to use ML in the Requirement Engineering(RE) domain. To better understand, it is worth focusing on RE [8]. The procedure of designing a requirement specification is called RE. The appropriate specification, elicitation, and validation of the requirements of stakeholders is a very crucial activity in the software development life cycle. RE is the most important activity of software development where the various stakeholders are involved [9].

1.1 Requirement Engineering

The various authors have given different definitions of RE. According to Lamsweerde “Requirement Engineering is a structured collection of activities to explore, analyze, record, consolidate, update, and adjust the requirements, capacities, qualities, limitations, and assumptions” [10]. As per Dick, Hull, and Jackson, “ Requirement Engineering is a very critical phase of Software Engineering. It first determines the nature of the issue and then relates all subsequent development knowledge to it” [11]. The project activities are monitored and the cost-effective and satisfactory development of software is achieved by using it. RE helps to determine a consistent set of requirements before the actual development starts. Agile and Traditional software development consider RE to be very significant and crucial for software development. Agile development takes RE as an iterative process, whereas traditional software development regarded RE as preliminary activity. The process of RE can be broadly classified into different steps including Elicitation, Analysis, System Modeling, Specification, Validation, and management. The phases of RE are depicted in Figure 1.1.

1. *Requirement Elicitation*: The elicitation process refers to a gathering of requirements using different techniques including interviews, workshops, brainstorming, and questionnaire from stakeholders. In the current context, the analyst explores different requirements and assumptions for designing software development. Existing NFRs elicitation can be classified as requirement elicitation and detection. The detection process involves extracting NFRs from existing documents manually or semi-automatically. Whereas requirement elicitation helps stakeholders to consider requirements and negotiation.

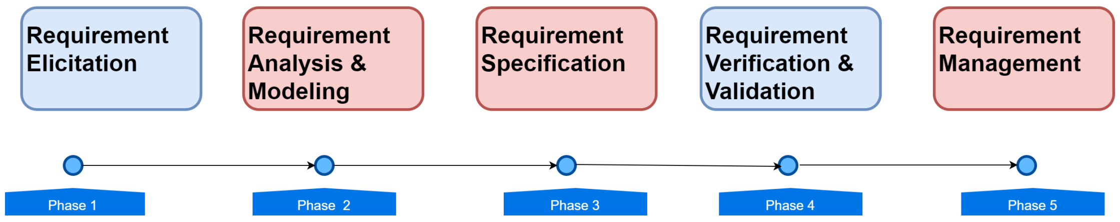

Requirement Elicitation
Requirement Analysis & Modeling
Requirement Specification
Requirement Verification & Validation
Requirement Management
Phase 1
Phase 2
Phase 3
Phase 4
Phase 5

FIGURE 1.1: Requirement Engineering Phase

2. *Requirement Analysis and Modeling*: Different graphical tools are used in the requirement analysis and modeling phase, including Flowcharts, Unified Modeling Language, and Data Flow Diagrams. It deals with designing the blueprint of the system.

3. *Requirement Specification*: System Requirement and Specification(SRS) document, is designed in requirement specification phase. The requirement specification deals with what is to be done and not about how the requirements are implemented.

4. *Requirement Verification and Validation*: It deals with the examination of the documented requirements that meet the specifications of the stakeholders. Additionally, It checks the consistency of the software model. The requirements are finalized after passing the validation process, with the goal of quality assurance. It reviews the SRS document with the actual requirements and assures the quality of the system.

5. Requirement Management: It deals with the management of changing requirements even after deployment. Moreover, It deals with the extension of requirements and changing priorities.

Therefore, RE sets up the requirements needed by the customer of software and the restrictions or constraints under which it may work [12]. Requirements may vary from a high-level abstract service specification or a framework limitation to a comprehensive practical mathematical specification [13]. FRs and NFRs are two broad categories of requirements as shown in Figure 1.2.

Requirements
Functional Requirements
Non Functional Requirements

FIGURE 1.2: Classification of Requirements

1.2 Functional Requirements(FRs)

According to IEEE “FRs are requirements that specify a function that a system or system component must be able to perform” [14]. FRs define a feature for software users. The technology issues and implementation aspects are not included in FRs. The main functionality that stakeholders expect from the software is defined FRs. The FRs are used to build the application architecture of the software. FRs describe a function or service that is provided by the software. The specification of services or functionality that software should deliver is depicted in the FRs. It is a high-level description of what the software should perform, such as the user searching the entire database or a section of it. Technical specifics, data calculations, manipulations, and software processing are examples of FRs. FRs are the essential services that the stakeholder expects from the software, such as creating, updating, and deleting accounts in the banking system [15]. FRs are the specifications that the end-user explicitly requires as the essential facilities of the system.

1.3 Non Functional Requirements(NFRs)

NFRs refer to the expected features from the software, which are known as quality requirements [16]. The technological architecture of the software is derived from NFRs. NFRs are restrictions on the functionality of the system [17]. It emphasizes how a specific process is to be done and not what the software is expected to do. NFRs are generally stated informally and often contradictory. NFRs are challenging to enforce and test before delivery. However, NFRs include utility features such as simplicity, reliability, portability, scalability, adaptability, and variability. NFRs are vital to the success and failure of IT ventures. It is mentioned in the literature that NFRs are considered second-class requirements even though the success and failure of a project depend on NFRs [18][19][20][21][22].
NFRs must be given an equal significance of consideration as FRs. Moreover, it is essential to elicit the NFRs appropriately, such as consistent, unambiguous, and complete. It is very crucial to consider the NFRs during the initial phases of development. There are different techniques available for NFRs elicitation. NFRs are hidden in FRs and are overlooked until the development of the system [23]. At the final phase of software development, NFRs are difficult to design, develop, and test which results in inferior quality and higher system repair costs. NFRs play a very crucial role in designing the architecture of every software. Therefore, NFRs are also called essential architecture requirements [24].

Table 1.1: Difference between FRs and NFRs

Functional Requirements(FRs)	**Non Functional Requirements (NFRs)**
FRs detail the functional aspect of the system, such as technical details, calculations, data manipulation, and processing.	NFRs define the performance or quality features of a system such as interoperability, efficiency, effectiveness, recovery, and accessibility.
FRs are defined and specified in the system design.	NFRs are neither standard requirements nor documented. Generally, NFRs are reported informally.
FRs are not contradictory as well as not subjective.	NFRs usually contradict each other and are subjective.
FRs are described and stated by the stakeholders.	NFRs refer to the technical requirements and are defined and expressed by technical people such as developers, team leaders, and architects.
FRs tests are used to check the working of the software.	Non-functional testing acts as an excellent way to ensure that consumer needs are being met.
FRs deal with software functionality including product features.	NFRs are well-defined in terms of software performance including product properties.

1.4 Difference between FRs and NFRs

The comprehensive definition given by Chung et al. [7] that the NFRs deal with how well the system will perform, and FRs define what the software is supposed to perform or do. FRs derive the application architecture of the software whereas NFRs derive the technical architecture. The employment of FRs is of no use without considering NFRs. The detailed difference between FRs and NFRs are discussed in Table 1.1.

The identification of NFRs can be classified into 2 techniques such as NFRs detection and NFRs elicitation technique. The NFRs detection technique refers to extracting the NFRs from existing documents like SRS, manuals, documents, etc. At the same time, the NFRs elicitation technique deals with different methods such as templates, questionnaires, guidelines, checklists, SRS, and interviews with stakeholders about the quality requirements. As a result, developing better strategies for eliciting NFRs is of interest [25].

NFRs do not have a standard definition or classification. The literature mentioned that the various authors had given different definitions, classifications, and sub-classifications of NFRs. For instance, Davis [26] recognizes that NFRs are non-behavioral types of requirements. It identifies the seven listed NFRs are portability, durability, effectiveness, human engineering, testability, comprehensibility, and modifiability. Another classification has been proposed by Kotonya and Sommerville which classified the NFRs into 3 categories, namely process, product, and external NFRs. Category 1 consists of system characteristics. Category 2 includes limitations and restrictions on the process within the system. Finally, category 3 included corporate legislation, national or international norms, and inter-operability criteria are externally relevant NFRs [27]. Glinz [12] has classified the NFRs and represented NFRs using 4 aspects: type, representation, fulfillment, and function. Another example of NFRs classification is presented by Lamsweerde [28]. Specifically, NFRs are classified as architectural constraints, quality of services (QoS), development constraints, and compliance. Various researchers have given different definitions of NFRs as depicted in Table 1.2.

Table 1.2: Different Definitions of NFRs

References	**Definitions of NFR**
Jacobson et al. [29]	Environmental and deployment restrictions, efficiency, platform dependencies, maintainability, extensibility, and durability are examples of requirements. Such a requirement defines a physical constraint on FRs.
Kotonya et al. [27]	Requirements that are not directly related to a system's features. They impose limitations on the product under production and the development process and additional requirements that the product must satisfy.
Ncube and Cornelius [30]	The behavioral characteristics must be present in the defined tasks such as performance and usability.
Robertson et al. [31]	A product property or qualities including the appearance of a product, performance, and precision properties.

Glins and Martin [32]	A service requirement is not affecting its functionality, but describing characteristics, limitations, considerations of efficiency, architecture, quality of service, environmental issues, failure, and recovery.
Wieger et al. [33]	A complete characteristics or property that software must have and the restrictions on the functionality.
Tsai et al. [34]	Requirements that define parameters for assessing the process and not specific activities of a system.
Miller et al. [35]	A description of how well a system must work non-functional specifications, characterize the system environment or system features. It defines the restrictions placed on the design of the software and qualitative qualities relating to the need for users to operate, revise, and manage changes and development.
Young and Rowland [36]	A required feature in a system, which specifies the performance of functions, also referred to as utilities in system engineering.
Huang et al. [37]	It describes primary limits on system development and behavior. NFRs define as safety, performance, availability, expansion, and portability. They play a significant role in the design of the architecture and should be addressed and defined during systems development as quickly as possible.
Davis and Alan [26]	The overall system attributes required include compact, reliable, effective, human engineering, testability, understanding, and adaptability.
Mylopoulos et al. [38]	A formal description or complete list of NFRs are not available. Global production and operating cost requirements, efficiency, reliability, maintenance, portability, ruggedness, and similar requirements have come under the NFRs.

The definition and classification given by ISO/IEC 25010 are followed in the current research. The ISO/IEC 25010 outlines the 3 quality models for software products, namely Quality in use model, Data Quality, and Product Quality models [39]. The present study focuses on the software Product Quality model, which classifies the NFRs into 8 categories as depicted in Figure 1.3 and is detailed ahead:

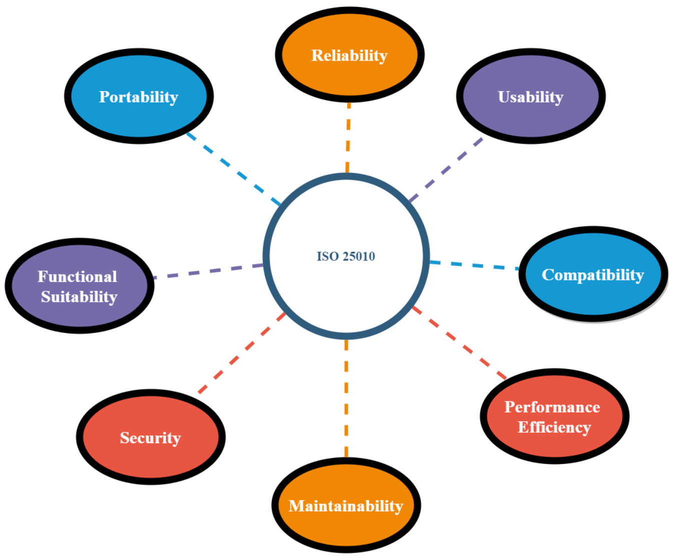


FIGURE 1.3: ISO 25010 Classification

1. *Functional Suitability* indicates the measure of the the functionality provided by the software that fulfills the stated and implicit requirements under fixed conditions. It is further classified as Functional completeness, Functional correctness, and Functional appropriateness.

   - *Functional completeness* refers to the extent to which the set of functions covers the stated task or objectives.
   - *Functional correctness* represents the intensity of the correct result and the required precision provided by the system.
   - *Functional appropriateness* refers to the degree of accomplishment of a stated objective or task by functions.

2. *Performance* efficiency refers to the measure of performance of resources used under the specified constraints. It consists of the sub-characteristics of Capacity, Resource utilization, and Time behavior [40].

   - *Capacity* refers to the degree of a maximum limit of software parameters that fulfill the requirements.

- *Resource utilization* refers to the type and quantity of resources utilized by the software to fulfill the requirements.
- *Time behavior* refers to the performance of the time-related factors like processing time, response time, and throughput rate of software while meeting the requirements.

3. *Compatibility* highlights the degree of resource exchange between systems, software, or components. It is composed of the sub-characters like interoperability and Co-existence.

    - *Interoperability* refers to the extent to which systems share information and make use of that information.
    - *Co-existence* refers to the degree of efficiency of performing the stated functions while sharing the shared resources and without deteriorating the performance of other products.

4. *Usability* refers to the standard of usage of the system efficiently to accomplish the specified objectives. It consists of the following sub-classifications as User error protection, Appropriateness, Learnability, Operability, Accessibility, and User interface aesthetics [41].

    - *Appropriateness recognizability* refers to the extent of the appropriateness of the software requirements by the user.
    - *Learnability* represents the extent to which a framework can be utilized effectively by particular clients to accomplish the objective of learning.
    - *Operability* refers to the total number of features that make the software simple to operate and control.
    - *User interface* aesthetics alludes to the level of user-friendliness while interacting with the system.

- *Accessibility* refers to the degree of usage of the system by people with a full range of features to accomplish a specified objective in the context of usability.

5. *Reliability* depicts the measure of task performance under the specified constraint of the stated time. It is composed of the following classification like Maturity, Fault Tolerance, Availability, and Recoverability.

    - *Maturity* refers to the extent to which the system fulfills the reliability requirements in usual operations.
    - *Availability* refers to the degree of accessibility of the system when it is required.
    - *Fault tolerance* alludes to the level of Operability of the system despite hardware and software faults.
    - *Recoverability* is the extent of recovery of data and regenerates the system's intended state even though the system's failure or interruption.

6. *Security* spotlights the extent of data and information protection so that other systems can appropriately access the data concerning the level of authorization. It can be sub-classified as Confidentiality, Accountability, Non-repudiation, Integrity, and Authenticity [42].

    - *Confidentiality* is the degree of accessibility of data only to the authorized ones.
    - *Integrity* refers to the degree of preventing unauthorized users from accessing and modifying the data.
    - *Non-repudiation* refers to the degree of confirmation of the events that have taken place and which is not possible to deny later on.
    - *Accountability* represents the level of traceability of actions to a unique entity.

- *Authenticity* alludes to the level to which the identity of a resource can end up being the one asserted.

7. *Maintainability* represents the point to which a system can be altered for correction, improvement, and adaptation to changing requirements and environments. Modularity, Reusability, Analysability, Modifiability, and Testability are some of the sub-qualities.

    - *Modularity* represents the degree of change in one system or component with minimal effect on other components.
    - *Reusability* denotes the degree of use of the resources in building further resources.
    - *Analyzability* is the degree of effectively identifying the system for deficiencies and also causes of failure. It also diagnoses the parts or components needed to be modified.
    - *Modifiability* represents the degree of efficiently modifying the product without deteriorating the product quality.
    - *Testability* refers to the degree of establishment of test criteria for the system, and tests can be executed to define whether the state criteria have been met or not.

8. *Portability* outlines the extent of efficient transfer of the system from one type of hardware, software, and environment to another. It can be sub-classified as Replaceability, Adaptability, and Installability [43].

    - *Adaptability* refers to the degree of efficient adoption of changing platforms like hardware, software, and usage environment.
    - *Installability* represents the degree of successfully uninstalling and installing the system under the stated environment.
    - *Replaceability*stands for replacing a software product with the same goal and in the same environment.

A quality model is an important component of a product quality assessment system. The quality model determines which quality parameters will be used while analysing the attributes of a software product. Various researchers have considered the ISO/IEC 25010 characteristics as a base for NFRs evaluation. The elicitation of FRs is different from the NFRs. A complex problem is how to get the full specifications for software development. In traditional and agile project management techniques, customers and developers employed a lot of time designing FRs. Stakeholders typically determine what they want from the framework but are not even completely conscious of NFRs. Project failure can be due to incomplete specifications. NFRs are not considered appropriate and are often overlooked throughout the software lifecycle.

1.5 Importance of NFRs

Though NFRs identify the standards adopted in development, hardware allocation, and technology selection in the system development but "NFRs treated as second class requirements and ignored until the end of the development cycle"[20] [44]. Moreover, NFRs are neglected in traditional as well as agile software development [45]. FRs are more focused than NFRs [46]. Without a well-defined set of NFRs, the development of software projects leads to failure [4]. The papers [6] [47][48][49] have shown numerous examples of project failure due to mismanagement of NFRs. For instance, Paramax′s software engineering group had developed software of more than 300,000 Lines of code, but neglecting the NFRs led to the project failure [6]. Engineers used a 4th-generation language to meet technological affordability and timeliness goals in the New Jersey Department of Motor Vehicles licensing scheme. Still, the software struggled because of performance and scalability issues [50]. According to Babar et al. [51] more than 40% of discovered faults in a US Air Force project were traced back to requirements problems. NFRs are challenging to establish and evaluate. It has been observed that non-implementation NFRs lead a project beyond the time limit and budget, and as a consequence, it could lead to the failure of a software development project. Identifying and correcting requirements errors accounts for 70% to 85% of rework costs [52]. Not properly accounting for NFRs results in the most costly and complex mistakes that can only be resolved after a project has been completed, and it is ranked as one of the top ten risks in software development [53]. It is necessary to elicit the NFRs entirely, consistently, and clearly during the initial phases of project development. It is crucial to create a more effective way of eliciting NFRs. NFRs are very important to be taken care of in the development as they influence the suitability of database, programming language, and operating system [54]. Because there is no standard for recognizing NFRs, it is a difficult task. NFRs are usually incomplete or mixed up with FRs and are much needed to elicit at the initial phases of development. Various elicitation techniques such as Brainstorming, Checklists, Questionnaires, Catalogs, and Templates are used significantly. These are used to get feedback from stakeholders. The NFR catalog aids analysts in identifying quality needs and resolving disagreements. Based on state-of-the-art research, detection techniques extract the low-level early elements in design and code. NFRs are acquired using different elicitation techniques. It requires a great deal of user engagement. Conspicuously, the identification and extraction of various NFRs must be automated. Henceforth, considering the automation of NFRs elicitation, ML can play a considerable role.

1.6 Research Challenges

- Minimal secondary data related to NFRs are available on the internet [55]. So primary datasets need to be collected from various sources [56]. The main challenge to the current research work is a lack of available datasets. Limited datasets are available which are given by researchers and academics

[57]. The shared datasets available on PROMISE repository [1] is used in numerous studies [37][58][59] [60]. Moreover, the dataset contains only 326 NFRs, which is limited in feature size for building an ML model [61]. It encourages the researchers for the collection, classification, and validation of real-life NFRs. The research in the NFRs domain can be enhanced with the shared dataset so that the researchers can perform the experiments and provide benchmarks for further work.

- According to current research work, it is found that identifying important features for NFRs is an open challenge and requires further consideration or investigation. Extracting significant features out of total features is a complex task [62]. The meaningless features significantly increase the number of false positives and affect the performance of the classifiers [63]. Moreover, limited research work has considered the hybrid feature selection techniques [64]. The literature has observed that a hybrid feature selection technique can improve the performance of ML models even in small datasets [65]. So it is indispensable to implement the hybrid feature selection for NFRs [66].

- Minimal research is done for effective elicitation and classification of the NFRs. The elicitation of NFRs is different from FRs. The elicitation and classification of NFRs are very crucial to unmask the quality attributes, constraints, and interface specifications for designing the technical architecture of the software [18]. It is beneficiary if the classification process becomes automated. It reduces mental fatigue, time, and human efforts in identifying and classifying requirements [67]. The current research focused on the elicitation and classification of NFRs.

- An efficient NFRs elicitation and prediction significantly reduce elicitation time while dealing with NFRs [18]. There are various NFRs elicitation techniques such as brainstorming, interview, template, and questionnaire [68].

[1]Data provided by DePaul University

Table 1.3: Identified and focused NFRs

ISO IEC 25010	[74]	[75]	[76]	[73]	[13]	[77]	[78]	[79]	[80]	[81]	[82]	[83]	[56]
Functional Suitability	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗
Efficiency/ Performance	✓	✗	✗	✓	✓	✗	✗	✓	✗	✗	✓	✗	✗
Compatability	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Usability	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Reliability	✓	✗	✗	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗
Security	✗	✓	✗	✓	✓	✗	✗	✓	✓	✓	✗	✓	✗
Maintainability	✓	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✓
Portability	✗	✗	✗	✓	✓	✗	✗	✗	✗	✓	✗	✗	✗

Among all the techniques the questionnaire is very economical and easy to elicit [22]. There is a lack of a standard classification of NFRs [69]. Numerous studies have worked on NFRs classification by considering the ISO/IEC 25010 [55]. It is important to classify the NFRs efficiently and find some method to predict the NFRs during the early phases of development [18].

- The limited research scope does not justify the usability perspective as it focuses more on security and other functional aspects. Usability is a very important factor as it deals with users experience with the software when users interact with it. The various factors such as efficiency, ease of use, the efficiency of use, and overall satisfaction of the user with the software. Usability is one of the factors of ISO/IEC 25010. The various research works have focused on other factors like security, portability, maintainability, and compatibility. The limited research has considered the usability factor, which is quite significant to be considered [70] [71] [72][73]. The various authors have focused on the different NFR factors of ISO/IEC 25010. Table 1.3 has shown the various NFRs focused on by different researchers. In the current work, we are focused on Usability as minimal research has focused on it.

- To the best of our knowledge, No multilayered framework is used for the prediction of NFRs. The multilayer framework will be used by researchers as a reusable template for NFRs prediction. The conceptual framework will assist that how the prediction is carried out. The framework addresses the various phases of NFRs prediction. Specifically, the layered framework will classify the whole prediction process into different layers. It will be used by other research works as a baseline.

- Reliability and Stability efficiency are very significant parameters. Minimal research works have considered these factors for NFRs prediction. Reliability is generally defined as the ability of a system to perform and maintain its functions in routine or unexpected circumstances [84]. It can be evaluated either with positive performance indicators such as in terms of accuracy or with negative performance indicators i.e in terms of the less is better, such as mean square error. Since reliability is in most cases defined qualitatively, the reliability analysis is therefore an estimate for quantitative measuring of reliability. Model-independent reliability estimates can be performed by changing the learning set and exploiting the general properties of the prediction model. It is very important to understand the reliability of the prediction system. Reliability is very significant to be considered for the evaluation of ML model [85].
  Stability represents the degree of inconsistency between different predictions made by the algorithm. It is an important measure to determine the performance of the prediction system [86]. The stability of the prediction system influences the user trust in the prediction system. A prediction system is stable if the predictions do not change strongly over a short range of data [87]. Therefore, stability and reliability are very important ML model evaluations that are required to be taken into consideration [88].

1.7 Research Motivation

The present thesis focuses on the classification and prediction of NFRs. The classification and prediction of NFRs is a very demanding domain of software engineering. Automating the task of classification and prediction helps to reduce human efforts, and fatigue and less prone to errors.

- ML has been used to solve complex practical domain problems in a variety of engineering disciplines. It enables task automation and performance efficiency. NFRs are a part of tacit knowledge and not always explicit, which makes them hard to elicit [79]. The extraction and categorization of NFRs from documents of different types are very demanding and error-prone. ML is a significant field in the current world of science that performs a task without explicitly programmed [89]. ML helps predict and prioritize the NFRs in the early phases of software development.

- The efficient ML models improve the elicitation of NFRs and help developers in the context of NFRs. Numerous ML approaches including K-Nearest Neighbor, Support Vector Machine, Random Forest, and Decision Tree have been successful in accomplishing the identification and classification process of NFRs[90] [91][92]. However, there is a scope for exploring more algorithms and techniques for predicting NFRs.

- Advances in ML algorithms, graphics processing units and open-source libraries have enabled the application of ML to address software engineering challenges. However, the usage of ML to enable decision-making during the software engineering lifecycle is very significant [93]. In the current research, we leverage ML techniques to develop an effective approach to classifying and predicting NFRs.

- Various authors have given different definitions and classifications of NFRs. Moreover, NFRs are subjective. It is very important to accumulate the all NFRs of a specific domain. So there is a need for some template or catalog which contains maximum NFRs. The mature catalog is the key to the problem. A mature catalog is an iterative way of modifying the questionnaire till the modifications become negligible and extract the maximum number of NFRs. Figure 1.4 presents the creation of a mature catalog.

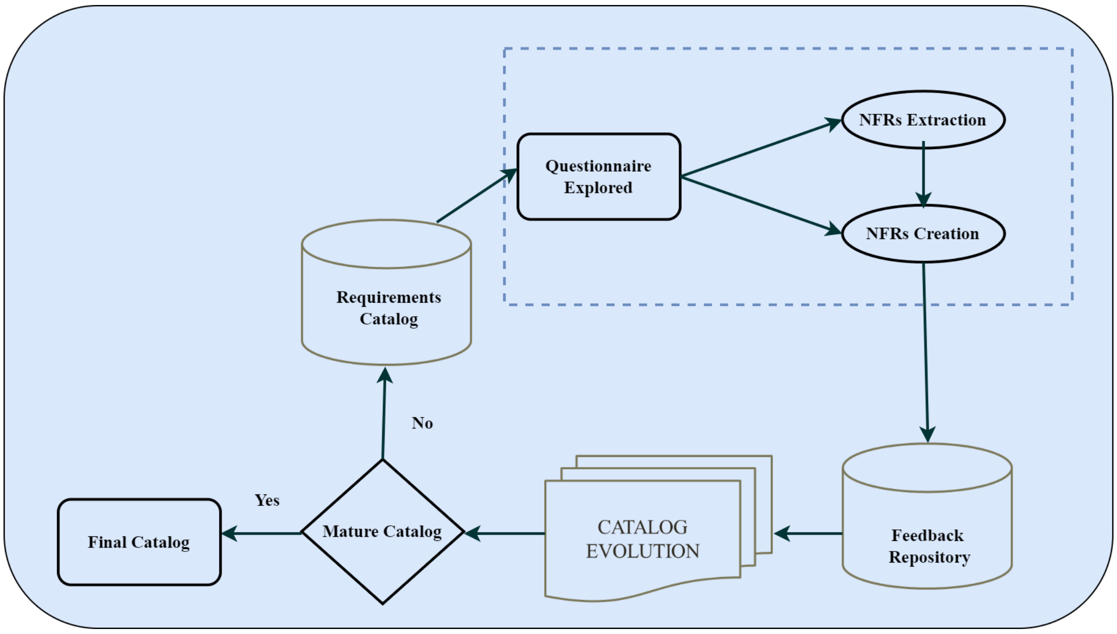


FIGURE 1.4: Creation of Mature Catalog

- According to recent research, it has been shown that finding critical NFR features is a difficult task that needs further thought or research [62]. It is difficult to separate relevant features from total features [66]. Moreover, Manually selecting features results in biasness of data. Manual feature selection is prone to errors. Additionally, the person choosing the features could have personal preferences. Automation facilitates effective feature selection. Through automated feature selection, the useful features are produced automatically and have fewer chances of biasness [94]. The classifiers performance is impacted and the number of false positives increased dramatically by the useless features. Additionally, only a small amount of research has taken into account hybrid feature selection strategies for NFRs [63]. The research has found that even with limited datasets, a hybrid feature selection strategy can enhance the performance of ML models. Implementing the hybrid feature selection for NFRs is therefore essential. It is mentioned in the state-of-the-art literature review that hybrid FS has provided better performance than the baseline filter or wrapper method [64]. A hybrid model can take the advantage of both filter and wrapper methods [65]. It is very significant to explore the hybrid FS for the present model.

1.8 Role of Machine Learning in Software Engineering

ML is a vital subject of research in today's world that performs tasks without being explicitly programmed. ML creates a function, model, or algorithm to learn and perform extractions from datasets known as training datasets [95]. A known output is utilized as an input to the training dataset. The ML can be broadly categorized into supervised, unsupervised, semi-supervised, and reinforcement learning techniques, as depicted in Figure 1.5. When historical data is available, training datasets are explored using supervised learning to extract future events. SVM, Decision Tree, and J48 are the most popular and efficient supervised learning algorithms [96]. Unsupervised learning makes use of unlabeled input data to uncover hidden patterns. K-mean clustering, K-nearest neighbor, Hierarchical clustering, and Principal component analysis, to name a few are the most popular unsupervised learning algorithms [97]. Semi-supervised learning is a type of ML that involves using a lesser amount of labeled data and a bigger quantity of unlabeled data to train. It combines unsupervised learning (with no labeled data) and supervised learning (with labeled data). Supervised learning is different from Reinforcement learning as the target data or key is included in the training data allowing the model to be trained with the right answer. On the other hand, reinforcement learning has no right or wrong response, and it determines how to complete the task. It will inevitably learn from its own experience if it does not have access to a training dataset. Artificial Neural Network (ANN) is an area of ML dealing with algorithms inspired by the function or structure of the brain.

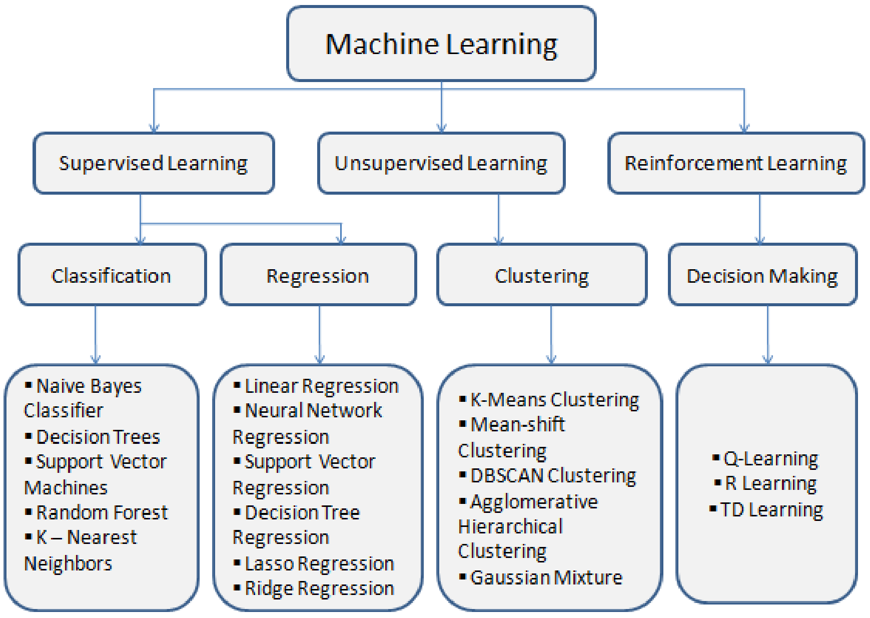


FIGURE 1.5: Machine Learning Classification

The computational discipline of ML includes Deep Learning(DL). Deeper neural networks, on the other hand, use numerous convolutions to offer a hierarchical representation of data in DL [98]. DL enables more learning capabilities, resulting in improved performance and precision. DL builds on traditional ML by adding more complexity to the model and altering the data with various functions that allow for hierarchical data representation. In supervised, unsupervised, semi-supervised, and reinforcement learning models, DL algorithms are used. The network started as a simple feed-forward neural network, but new types of networks including Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) were later added to convolutional [99]. CNN is a customized feed-forward neural network that was first used in image processing but is now widely used in a variety of industries. The current network uses a set of filters to extract features from an image automatically, resulting in a hierarchical structure of features. The filter weights are calculated directly from the training data. In most cases, CNN uses numerous convolutional layers to create a deep structure. CNN performs feature engineering automatically, reducing the time-consuming and inefficient process of doing it manually [100]. RNN was first used in Natural Language Processing, but it is now being used in various domains in a similar way to CNN. RNN is most commonly used to process sequential data with temporal dependencies. An RNN can use previous data to process new data. A key issue with RNN has been its difficulties in training with long time-dependent data (large time-series) [101]. To solve the problem, several RNN versions were developed. DL algorithms are used for NFR classification, which is the challenge of determining which of a collection of categories (subpopulations) an observation (or observations) belongs to. The term "classifier" can also refer to the mathematical function provided by a classification algorithm that transfers input data to a category [102].

ML algorithms have proved to be very successful in various fields of software engineering domain such as defect prediction, software reliability prediction, reusability prediction, maintenance effort prediction, and requirements classification [103]. Among all the problems the requirements classification is considered one of the most significant problems of software engineering. In the state-of-the-art work, various authors have applied different ML techniques and tools for classifying and predicting requirements. The process of identifying NFRs as well as FRs from the total set of requirements is a classification problem. Deciding whether requirements are FRS or NFRS is also a classification problem. Requirement Prioritization is also very significant to be considered. It is present in state-of-the-art work that ML is the key to all software engineering problems. The ML and DL algorithms used in the software engineering domain are discussed ahead:

1. *J48*: It is a decision tree that extracts the training dataset while keeping a range of features [104]. It is an execution of an algorithm for data categorization and determination based on the attributive values of the provided data [105]. The various authors have explored decision trees for requirement engineering problems [106][107].

2. *Support Vector Machine (SVM)*: It is also a type of classifier that may be used to create binary classifiers. Rather than specifying the input space, it relies on the geometrical properties of the given training dataset. The efficacy of the SVM is determined by the kernels that are chosen and used. It also successfully recognizes the soft margin parameters. It is capable of accurately modeling complex non-linear decisions. The SVM algorithm has been used for software reliability classification problem [108], prediction of software reliability [109], classification of NFRs [110], and predicting software defects [111].

3. *Random Forest*: Random forest selects a subset of a prepared set at random from the arrangement of Decision Trees. The final class of the test item is determined by adding the votes from several Decision Trees. Random Forest is the result of a developing ensemble of trees or a collection of extracted tree outputs [112]. Random forest algorithm has proved to be successful in various fields of software engineering such as software defect prediction [113] and classifying NFRs and FRs [114],

4. *Artificial Neural Network(ANN)*: At each layer, an ANN is made up of multiple perceptrons or neurons. Because its inputs are exclusively processed forward, an ANN is also known as a Feed-Forward neural network. They transfer data in a single path, going via multiple input nodes before reaching the output node [98]. ANN utilized for software effort estimation [115], software defect classification [116], software reliability prediction [117], and Requirement classification [118].

5. *Convolutional Neural Network(CNN)* : A multilayer perceptron variation is used in the CNN-based model, which is capable of linking or pooling together many convolutional layers. A portion of the data is captured by these convolutional layers, which are subsequently separated into rectangles and processed in a non-linear manner. Without the need for human intervention, automatically detects the relevant properties or features from data [99]. CNN proved successful in research works of software design and modeling [119], Software bugs detection [120], and RE [121].

6. *Recurrent Neural Network(RNN)*: In RNN, the processing nodes are saved and reinserted into the model as the information is not passed in one direction only. The model can predict the outcome of a layer. The nodes of the RNN model function as memory cells, allowing calculations and operations to continue. If the network forecast proves to be incorrect during backpropagation, the system self-learns and works toward the correct prediction. RNNs are useful in time series prediction because they can remember previous inputs [54]. The various research works have explored and proved RNN is very successful in classifying NFRs [122], requirement prediction [123], and classifying FRs [124].

These ML and DL algorithms can be clubbed together. The phenomenon of clubbing together is known as ensembling.

1.8.1 Ensembling

The usage of numerous classifier systems is referred to as ensembling. It is effective and versatile in solving problems across a wide range of domains. By partitioning the same training dataset and the same algorithm in different ways or utilizing various algorithms on the same dataset, the ensemble approach can employ any ML technique as its base model. It is a hybrid model that combines numerous models to create an output model. The ensemble method seeks to improve results by using a variety of models and methodologies. The phrase ensembling is described using a variety of terminologies such as classifier, combination, committee, combination, and aggregation [125]. Voting, stacking, bagging, and boosting are different ensembling methods. The techniques of ensembling are explored and proved comparatively successful by researchers in various software engineering fields such as software defect prediction [126], requirement classification [127] and software effort estimation [128].

The effectiveness of the ML algorithms even ensembling depends upon choosing the significant features out of all features.

1.8.2 Feature Selection(FS)

It is a process of picking a meaningful subset with the given variables to build a model. Its goal is to enhance accuracy by allowing ML algorithms to train faster, lower complexity, reduce overfitting, and extract datasets. FS techniques include super greedy and greedy algorithms. The FS technique reduces the number of redundant or unnecessary features, resulting in less information loss. It is utilized in domains that are complex and have a small number of data samples [129]. The FS techniques can be broadly categorized as Wrapper, Filter, and Hybrid technique [130]. The various techniques of FS have classification [131], document classification [132], and software fault prediction [133]. Furthermore, the proposed research will compare traditional ML algorithms such as SVM, K-nearest neighbor, and Random Forest with DL methods. The various DL methods such as ANN, CNN, and RNN are explored for prediction purposes.

Constructing the ML model has gone through various phases such as data preprocessing, feature selection, and ensembling to make a prediction.

1.9 Research Objectives

ML enables automated prediction of NFRs. The current research enhances the accuracy of NFRs prediction by ensembling a novel ML model. The primary goal of the proposed research is to develop a hybrid ML model for predicting NFRs effectively. The proposed development has the following subsidiary objectives:

1. To develop a mature catalog, which considers a maximum number of perfect and modified NFRs templates.
2. To propose a hybrid feature selection algorithm.
3. To predict significant NFRs using different approaches of ML.
4. The following parameters will be used to assess the effectiveness of the proposed technique.
    - Accuracy
    - Recall
    - Precision

1.10 Proposed Research Methodology

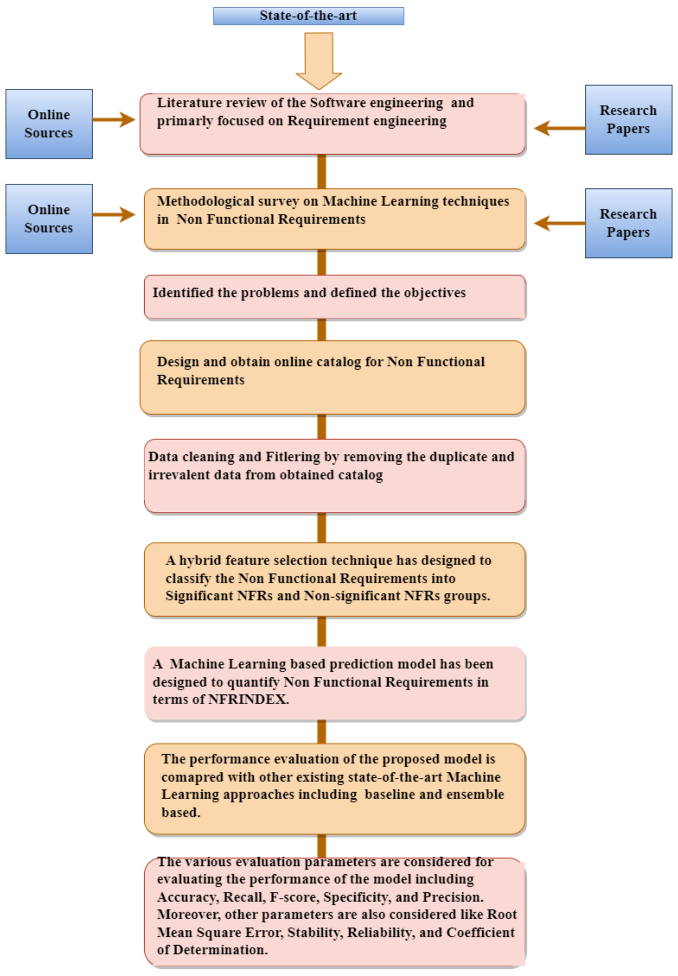

State-of-the-art
Online Sources
Literature review of the Software engineering and primarly focused on Requirement engineering
Research Papers
Online Sources
Methodological survey on Machine Learning techniques in Non Functional Requirements
Research Papers
Identified the problems and defined the objectives
Design and obtain online catalog for Non Functional Requirements
Data cleaning and Fitlering by removing the duplicate and irrevalent data from obtained catalog
A hybrid feature selection technique has designed to classify the Non Functional Requirements into Significant NFRs and Non-significant NFRs groups.
A Machine Learning based prediction model has been designed to quantify Non Functional Requirements in terms of NFRINDEX.
The performance evaluation of the proposed model is comapred with other existing state-of-the-art Machine Learning approaches including baseline and ensemble based.
The various evaluation parameters are considered for evaluating the performance of the model including Accuracy, Recall, F-score, Specificity, and Precision. Moreover, other parameters are also considered like Root Mean Square Error, Stability, Reliability, and Coefficient of Determination.

FIGURE 1.6: Research Methodology Flowchat

Research methodology is a way to solve the research problem systematically. It is a step-by-step procedure used by researchers to describe the research work. In the current research, the usability-based NFRs dataset has been from various IT professionals and academicians. The data preprocessing has been performed on data by removing the null values and duplicate records. A hybrid feature selection has been applied to the dataset to classify the data into two groups namely significant and non-significant groups. A hybrid model has been designed for NFRs prediction. The baseline and ensemble-based ML algorithms have been compared with the proposed technique for performance enhancement. The proposed technique can register improved performance in terms of Precision, Recall, Accuracy, Mean Square Error, Coefficient of Determination, Sensitivity, Specificity, Reliability, and Stability. Figure 1.6 shows the top-down approach of the presented methodology. The essential details are depicted ahead.

The proposed work, as shown in Figure 1.6, follows the mentioned methodology.

1. To achieve the first objective, the online questionnaire has been designed based on the usability perspective of ISO/IEC 25010[2] software product quality parameter of NFRs. Initially, a mature catalog is formulated using different research papers which is enhanced with the help of various IT professionals. A mature catalog is a repetitive way of enhancing the questionnaire and extracting the maximum number of NFRs so the modification becomes negligible. The questionnaire has been distributed among various IT professionals and academicians for accumulating data. Moreover, NFRs based secondary dataset has been also accumulated from different sources.

2. To attain the second objective, a hybrid feature selection algorithm has been presented for selecting the Significant NFRs from the obtained dataset. The proposed model has been compared with other baseline and ensembling techniques to determine the model's efficiency. A hybrid feature selection model has explored the merits of the filter and wrapper method. The filter-based techniques including information gain and correlation have been used in the proposed research. Moreover, the Bayesian Belief technique with backward selection has been explored as a wrapper method. Therefore, a proposed feature selection model has the advantages of both filter and wrapper techniques.

3. To achieve the third objective, a Long Short-Term Memory and Convolutional Neural Network based hybrid ML model for predicting the NFRs have

[2]https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

been employed and compared with state-of-the-art baseline and ensembling-based ML techniques.

4. Finally, to achieve the fourth objective, various parameters like Accuracy, Recall, Sensitivity, Specificity, Precision, Coefficient of Determination, Mean Square Error, Reliability, and Stability have been evaluated to determine the effectiveness of the proposed technique.

1.11 Major Contribution

NFRs are equally significant as FRs. The success and failure of any project depend on the NFRs. But it is mentioned in the literature that NFRs are not considered appropriate. So there is a need to view or predict the NFRs during the early stages of software development. The primary dataset is collected from various academic institutes and IT companies regarding usability-related NFRs. An online questionnaire is designed for generating the dataset. In the study, the significant NFRs are extracted from the total collected NFRs. The classification is done, and the percentage of NFRs considered is predicted. The current research will help the software developers to know beforehand about the percentage of NFRs considered so that they can take appropriate action in time. The success of software development depends immensely on the exact elicitation of NFRs. The current research focuses on the interest of NFRs and the ML technique. A multi-layered conceptual framework is proposed for effective NFRs prediction. Specifically, in the current system;

1. The dataset is collected from several software developers and faculty of colleges and universities engaged in developing academic-oriented software.

2. A novel M-LSTM and CNN-based prediction framework is designed for the estimation of NFRs.

3. The NFRs parameters are categorized into 2 groups Significant and Non-Significant NFRs based on a probabilistic parameter termed LoNFRS using the Bayesian Belief Model (BBM).

4. The probability of significant NFRs is quantified in terms of NFRINDEX value for prediction purposes. Based on these significant features, the ML-based model is constructed. The algorithm for NFRINDEX is presented in the proposed model. Moreover, the current research considers the probability measure of the level of NFR significance in terms of LoNFRS, which is cumulatively quantified as the NFRs significance measure (NFRINDEX). NFRINDEX has been quantified for prediction purposes using a Multi-scaled Long Short Term Memory (M-LSTM). It combines two techniques namely Convolutional Neural Network and Long Short Term Memory.

5. A state-of-the-art comparative analysis is conducted to estimate the performance enhancement of the current NFRs prediction model.

6. A color-coded technique of Self Organized Mapping (SOM) is used to visualize the outcome of NFRs significance prediction.

Furthermore, in the field of NFR prediction, the current system has several advantages. Some of the advantages are listed below:

1. The current system will find the significant NFRs considered at the early stage of software development.

2. The ML techniques based on feature selection and ensembling will automatize the prediction of NFRs. Automatic prediction of NFRs will help the software developers to consider the significant NFRs before starting the actual development.

3. The current system will visualize the results of NFRs parameters over portable devices based on the mobile platform for providing user-specific decision-making.

4. The proposed system has the potential to recommend NFRs for software development.

1.12 Thesis Organization

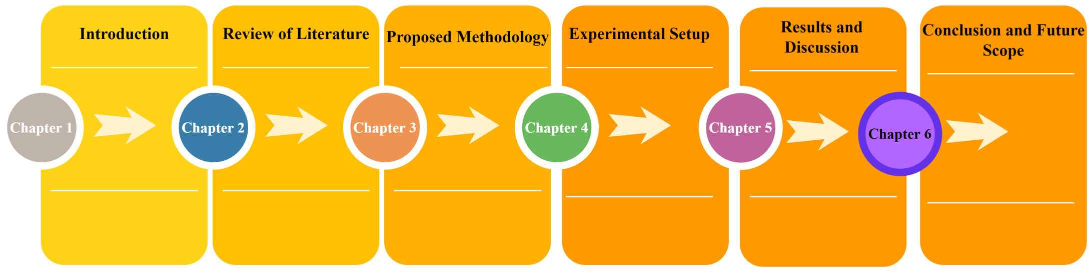


FIGURE 1.7: Chapter-wise thesis organization

The thesis is organized into 6 chapters as shown in Figure 1.7.

Chapter 1 introduces NFRs, Feature Selection, ML algorithms, and the metrics used for measuring the model accuracy in terms of Precision, Recall, and Accuracy. Chapter 2 deals with the state-of-art literature review of the NFRs, ML, and their annexure. This chapter is derived from the following papers:

- Naina Handa, Anil Sharma, "Non Functional Requirements: the emerging wisdom", International Journal of Research and Analytical Reviews, 2018. (UGC)

- Naina Handa, Anil Sharma, Amardeep Gupta "Non Functional Requirements Analysis using Data Analytics", International Journal of Advanced Science and Technology, 2019. (Scopus)

- Naina Handa, Anil Sharma, "Compact Discourse on Feature Selection", Think India Journal, 2019. (UGC)
- Naina Handa, Anil Sharma, "An Inclusive Study of Several ML Techniques", Journal of The Gujarat Research Society,2019. (UGC)
- Naina Handa, Anil Sharma, Amardeep Gupta, "An Inclusive Study of Several ML Based Non-functional Requirements Prediction Techniques" FTNCT, 2019. (Scopus)
- Naina Handa, Anil Sharma, Amardeep Gupta, "A Review of Machine Learning Techniques in Non Functional Requirements" 1st National Conference on Innovations in Applied Science and Engineering NCIASE, 2019.

Chapter 3 discusses the proposed methodology and layered framework of the model. The framework has been divided into different layers, and these are detailed in the current chapter. This chapter is particularly derived from:

- Naina Handa, Anil Sharma, Amardeep Gupta "An Elicitation of Non Functional Requirements using Questionnaire: A Layered Framework", Solid State Technology,2020. (Scopus)

Chapter 4 comprises the experimental setup for the research work. It consists of the data collection methodology, the technique followed in the current research, and the parameters used. This chapter is derived from:

- Naina Handa, Anil Sharma, Amardeep Gupta, "Non-functional Requirements Engineering Questionnaire: Novel Visions and Review of Literature" Recent Innovations in Computing, 2020. (Scopus)

Chapter 5 describes the results and discussion. The varied base and ensembling ML algorithms are applied and compared in the current chapter. The results of the proposed methodology are registered in the chapter. This chapter is derived from:

- Naina Handa, Anil Sharma, Amardeep Gupta, "Framework for prediction and classification of Non Functional Requirements: A novel vision" Cluster Computing, 2022. (SCI)
- Naina Handa, Anil Sharma, Amardeep Gupta, "Hybrid Feature Selection based Classifier for Non Functional Requirements" Mathematical Statistician and Engineering Applications, 2022. (Scopus)

Chapter 6 presents the conclusion and results of the thesis found. New research avenues are also introduced. This chapter is derived from:

- Naina Handa, Anil Sharma, Amardeep Gupta, "Framework for prediction and classification of Non Functional Requirements: A novel vision" Cluster Computing, 2022. (SCI)
  All the chapters systematically present the elicitation and prediction of NFRs through varied ML techniques.

Chapter 2

Review of Literature

2.1 Introduction

Non Functional Requirements are the essential aspect for the development of quality software. Moreover, software developers have recognised the importance of NFRs [13]. However, in state-of-the-art research works, it has been analysed that NFRs have not been given enough consideration, and are not considered adequately in the development of software [46]. Consequently, NFRs have been overlooked in software development [134]. Furthermore, it is very significant to elicit and predict the NFRs during the early phases of software development. Numerous projects have been failed by neglecting the NFRs [135] [51] [52]. NFRs are very critical for the correct architectural decisions and therefore known as architecture requirements [136].
ML plays an essential role in software engineering and particularly in NFRs [137]. ML helps to extract, predict and classify NFRs by exploring different techniques and methods [138]. The various ML techniques have been applied in different domains of NFRs and proved to be successful. Numerous research works have discussed the utilization of ML for NFRs prediction and classification [55]. It is very

important to consider ML for NFRs. Moreover, it is very significant to predict the NFRs in the early phases of software development.

Though the audit of literature affirms an annexure of ML and NFRs as an established field of research, it glares upon various constraints and unexplored areas in both domains. The literature review is divided into 7 sections. The first section specifically, Table 2.1 contains a comprehensive review of major contributions and limitations in the field of NFRs classification and prediction. The second section deals with the benchmark ISO/IEC 25010 which is used by different researchers for consideration of NFRs. We have also discussed the varied domains in which ISO/IEC 25010 is used. The third section gives a literature review regarding the different techniques of ML/DL in the context of NFRs. The fourth section elaborates upon performance metrics Table 2.5 outlines the various parameters for evaluating the performance of the ML model used by renowned researchers. The fifth section includes Table 2.6 reasoning out our choice of usability. The sixth section presents a comprehensive comparative analysis that results in tracing out our research gaps. The chapter ends with a detailed note on research gaps.

2.1.1 State-of-the-art Comparative Analysis: Major Contribution and Limitations

NFRs are very significant factors for determining the failure or success of any software system. NFRs are challenging to implement because NFRs are diverse, usually contradictory, and subjective. NFRs are ignored in traditional software development. It is crucial to unmask NFRs early during the phases of development. NFRs refer to the expected features from the software, which are known as quality requirements [16]. The technological architecture of the software is derived from NFRs. NFRs are restrictions of the functionality of the system [17]. It emphasizes how a specific process is to be done and not what the software is expected to do. NFRs are generally stated informally and often contradictory. NFRs are challenging to enforce and test before delivery. Eliciting NFRs emerges as the biggest challenge in RE. Numerous NFRs elicitation techniques are discussed in the literature. Various researchers have given different definitions and classifications of NFRs. ML has always solved complex practical domain problems in various engineering fields by automating various tasks. NFRs are not always explicit and can be a part of tacit knowledge, making them hard to elicit [30]. The extraction and categorization of NFRs from documents of different types are very demanding and error-prone. ML is an essential field of today's world of science that carries out a task without being explicitly programmed. ML is thus quite helpful in predicting and prioritizing the NFRs in the early phases of system development. The efficient ML models improve the elicitation of NFRs and help users/developers in the context of NFRs. Numerous ML approaches proved successful in accomplishing the identification and classification process of NFRs. Various researchers have used different ML techniques like Supervised, Unsupervised, and Reinforcement learning [32]. While dealing with NFRs, ML automates the process and reduces human efforts, time, and mental fatigue. ML is very rich in techniques and algorithms. The various types of algorithms including Support vector Machine, Random Forest, and Decision Tree are explored in the research focused on NFRs. The different works have used various evaluation metrics such as Precision, Recall, Accuracy, and F1-score. Table 2.1 consist a details of various ML techniques used for NFRs categorization and prediction. The dataset and performance evaluation technique used by the researchers have been discussed. This section does a literature evaluation of the major contribution and limitations of the research.

Table 2.1: Review on NFRs and ML: Major Contributions and Limitations

Ref.	**Major Contribution**	**Limitations**
Farid and Mitropolous [139]	Researchers designed a novel tool for modeling NFRs in an agile environment. the project leaders and scrum managers get facilitated by the tool for risk management.	The current work has not validated the NORMATIC tool in real-world projects specifically in an agile environment. A web-enabled version of the tool NORMATIC is another area for improvement.

Cleland-Huang et al. [37]	The research has focused on the automation of classifying NFRs. Software requirements and specification documents, meeting minutes, interview notes, and comments are used as source documents. NFRs include Legal, Availability, Look & Feel, Maintainability, Usability, Security, Scalability, Performance, and, Operability is focused. Recall, Precision, and Specificity are used to evaluate the performance.	The work has provided the initial validation and still, there is a need of evaluating whether the large training set will enhance the performance of NFRs classification in the context of various organisations, projects, and NFRs types. Moreover, it is also required to determine under what conditions retraining is necessary and the effect of it on performance.
Cleland-Huang et al. [58]	The research work primarily explored the performance, security, and, usability NFRs for classification. A novel iterative technique has been designed for classification. The authors have claimed that the present classifier can provide support to the analyst in classifying NFRs manually. It has scanned large datasets including meeting minutes, memos and survey responses efficiently to determine and classify NFRs. The current work introduced a novel approach for iteratively training a classifier to work in new domains. WEKA tool has been used for the experiment purpose. Precision, Recall, and Specificity has used for evaluating the performance.	Additional work is still required to enhance the runtime performance of the model so that the real analyst can use it.

Glinz [32]	The detailed survey has been performed on the problems with existing definitions, classification, and representation of NRs. The author has also a new facet definition of classification and segregates the requirements into representation, kind, role, and satisfaction.	Author has focused on the theoretical work. Moreover, the validation and practical usefulness of the system has taken as a topic for further investigation.
Yusop et al. [140]	The research has concluded that NFRs are not considered appropriately in web software development. The authors have conducted interviews with software developers who deal with commercial web development.	Research has concluded that the NFRs are stated informally for web development. Web developers focused more on FRs than NFRs. The authors recommended developing an intelligent tool for dealing with NFRs in the early phases of development.
Ezami [141]	Using data from the Electronic Health Records domain, the supervised approaches outperformed the unsupervised ones. The bag of words method has been tested and found to be superior to the doc2vec technique.	The key disadvantage of the present research is the requirement for a labeled dataset to conduct supervised model training. Even though there is a tonne of information from comments in open source projects in the Electronic Health Records and other fields, none of them have labels indicating whether they pertain to any of the NFR categories or not. The procedure of assigning labels to each sentence is expensive and subject to human bias. As a result, one of the potential possibilities for this work is the use of unsupervised or semi-supervised algorithms with comparable performance and less requirement for labeled data.

Martino et al. [142]	To facilitate the development of Cloud applications, an Automatic Classification model has been developed for deriving cloud service categories from requirements. It was retrieved from the PROMISE repository that was utilised for the target data.	The current research has not included Cloud Ontology. Moreover, reliability and stability parameters have not been focused on in the proposed research.
Bhowmil and Do [143]	Open source software, such as Firefox, Lucene, and Mylyn, were analysed for 50 NFRs. JIT requirements and engineering-specific tasks were uncovered in this study of how OSS project teams generate quality features from minimal initial descriptions. Following a preliminary study that focused more on implementation than requirements, this study is a follow-up. NFRs are the primary subject of this investigation.	Authors suggested considering just-in-time FRs as a future perspective. The reliability parameter is considered in research, but stability analysis is still missing in research.
Portugal et al. [144]	The paper addressed the difficulty of eliciting NFRs. NFRFinder, created by the authors, is a semi-automatic method for discovering NFRs in unstructured data. As an initial test, it was applied to structured text and used the metrics such as Recall and Precision. NFRFinder has demonstrated that NFR categorization is influenced by the context and stakeholders participating in the classification process.	NFRFinder has worked on structure text. The authors suggested working on qualitative analysis and extracting the NFRs from the unstructured text including meeting minutes, open-source software, and software comments.

Toth and Vidacs [145]	It has compared the applicability and performance of ML and NLP techniques in software engineering. Recall and precision are metrics that have been utilized. Stack Overflow and Promise repositories were used for the experiment. Research has used the Naive Bayes, SVM, Logistic Regression, and Decision Tree.	Neural Network and Recurrent Network implementations have provided a future outlook of the research.
Baker et al. [146]	One of the ML techniques utilized is an Artificial Neural Network and a Convolutional Neural Network. This study focuses on the usability, maintainability, operability, performance, security, and maintainability of the system. The F1 metric, recall, and precision have shown that the CNN outperforms categorising NFRs.	The research has opened new avenues for further research like exploring other algorithms for the same problem and domain. Specifically Recurrent Neural Network has many benefits over ANN and CNN. Therefore, RNN can be evaluated for the problem. The authors have designed a semi-automatic model for classification. The fully automatic NFRs classification model benefits the developers. Authors have focused on the lack of an NFRs dataset, the limited labeled dataset is available and explored in research. The large dataset can evaluate the model efficiency in a better way.
Abad et al. [147]	SVM is used to extract and classify requirements effectively and efficiently. Measuring performance has been done using a statistic called Precision, Recall, and Distance. Documents and interaction records are used as a dataset for the experiment.	Authors have explored only 3 datasets for the experiment. There is a need to evaluate a model on different datasets of various domains to evaluate the efficiency of the model.

Taj et al. [148]	Requirements have been categorized as either FRs or NFRs in this effort. The model provided by the authors in this research aids in the elicitation and classification of requirements. The requirements were elicited with the help of a crowdsourcing approach and input from a variety of stakeholders. To classify data, the Naive Bayes and Decision Trees have been employed in the research. The case study was used to demonstrate the model's effectiveness.	A web application for RE can be designed to elicit a response from various stakeholders. This helps in eliciting the goof number of responses over the globe.
Naseem et al. [149]	Authors have focused on software risk prediction as a critical activity. An ML-based model has been implemented for the same purpose using 10 different classifiers and Recall, Precision, F-measure, Mean absolute error, ROC and accuracy, etc.	Authors have suggested improving the accuracy by exploring more ML techniques for prediction. The current research work has missed the use of ensembling, class imbalance, and feature selection.
Rahimi et al. [127]	The research has introduced a novel ensembling-based model for classifying requirements. The various five ML algorithms include SVM, Logistic Regression, Naive Bayes, Decision tree, etc. These are combined to build the ensembling model. The model for classifying FRs has achieved 94.45% accuracy.	The present work has explored 5 ML algorithms. The research can be enhanced by evaluating other Baseline and ensemble-based algorithms. The current work has evaluated based on accuracy, However, precision and recall are very important to consider for model evaluation.

Assim et al. [150]	Artificial Neural Networks, Decision Tree, Random Tree, Linear Regression, Gaussian Processes, Random Forest different ML algorithms have been explored for predicting software defects. The proposed model has shown that the ensembling model acquires better results than the single model.	The assessment of additional ML classifiers and comparison of them can be performed as future work. To improve prediction accuracy, the research can also use more datasets during the learning and prediction procedures. Authors should also take into account practical software defect prediction.
Chung et al. [151]	Authors have performed a state-of-the-art review on the treatments of NFRs. The researchers have discussed the various definitions and classifications of NFRs. ISO/IEC 9126 is a very important classification among other classifications.	it is necessary to conduct an additional empirical study on the application of NFRs to RE as well as on the use of ethnographic studies to understand how software teams approach quality issues as requirements. The research has recognised that real projects should be used in the study, both in the lab and on industry projects. It enhances the understanding of quality issues that occur during development.
Casamayor et al. [79]	Research introduced the semi-supervised technique-based classification model for NFRs. The authors explored a semi-supervised approach to minimize the manual assignment of categories to NFRs. The feedback from the stakeholders also exploits to improve the performance of the classifier.	Authors suggested that active learning is a good approach that can be added to reduce the labeling efforts than the semi-supervised approach.

Mairiza et al. [13]	Researchers assessed the NFRs in 3 different dimensions including the relevance of NFRs in various domains, terminology and definition of NFRs, and, classification. A domain-oriented novel classification of NFRs is presented in research work. Performance, Reliability, Maintainability, Security, and, Usability are considered the most significant NFRs for the various domains.	The present work has 2 limitations such as the possible overlaps between the definitions as well as attributes of each NFR are not examined. The current work has not established a systematic hierarchy of NFRs.
Umar et al. [48]	The research work defined the various NFRs-based frameworks, recommendations, literature, techniques, and, limitations. The authors concluded that Security, Reliability, and Performance are the most common types of NFRs.	The aspect orientation of NFRs is very beneficial to reduce the complexity of software and enhancing the modularity of artifacts. But the current research has not considered the aspect orientation of NFRs.
Mirakhorli and Cleland-Huang [152]	The research covered the substantial advantages that can be attained by tracing NFRs as well as the unique traceability difficulties brought on by the cross-cutting character and interdependencies of NFRs. Authors have designed an architectural-centric approach for tracing NFRs.	One of the unsolved research topics in the Grand Challenges of Traceability has been identified as the difficulties with tracing NFRs. Researchers have focused on traceability but there is a need to understand and address traceability more deeply.
Farid and Weam [153]	The research work designed a tool namely NORMATIC for identifying and modeling NFRs. The goal of the research work is to design an NFRs framework that can be modified for an agile environment. The present work has been validated with two case studies.	The current work has the requirement of validating the framework in real-world projects based on the agile approach. It enhances the NFRs extraction, identification, and integration with FRs.

Slankas and William [154]	Authors developed NFR Locator to classify the natural language documents into 14 predetermined categories. A K-NN classifier is used to identify sentences that are similar across documents. The research uses a variety of classifiers, where the K-NN classifier was able to get the best results for NFRs classifications.	several limitations exist for the present work including the model not being able to extract the data or information contains in images and tables. Moreover, the researchers have considered the limitation that the toll is not generalized to different domains.
Rashwan et al. [110]	The research emphasized the requirement of some automation system for NFRs extraction and classification. The authors used the Support Vector Machine algorithm for the classification of NFRs. The Precision, Recall, and F-measure metrics have been used for evaluating the performance of the ML model which is implemented with the WEKA tool.	As a future work, the authors have suggested addressing the present misclassification by creating extra semantic features for the classifier. The research's long-term goal is to identify quality issues in various requirement artifacts to reduce the chances of software failure.
Chen et al. [155]	Authors suggested an automated technique that is based on the semantics of web services for directly combined NFRs and FRs. The model has been evaluated and proved effective in real-world case studies. Availability, cost, and response time are considered NFRs for web services.	Authors addressed that the work could be extended for other domains including sensor networks. State reduction methods can be explored to enhance performance.

Kopczynska et al. [156]	Researchers investigated the Structured Elicitation of Non-functional Requirements(SENoR) strategy. The technique consists of a succession of short meetings to generate new ideas driven by ISO 25010 Quality attributes. It utilizes Non-Functional Requirements Templates (NoRTs) to help the elicitation procedure. Our exploratory contextual analysis of the expense and viability of SENoR and NoRTs included 7 projects that created customized web applications.	Researchers addressed the need for further research in other organizations and initiatives, with an emphasis on increasing efficiency and lowering costs. The present research has focused on brainstorming and templates, other elicitations techniques need to be considered.
Sharma et al. [157]	Textual pattern recognition was used to detect and analyze different NFRs in the natural language text document proposed by authors. Rules, rather than keyword identification, are used in the proposed rule-based approach to detect and classify various NFRs in natural language.	The authors discussed some of the potential avenues for our future research, particularly those related to developing new rules and an ontology.
Raturi et al. [158]	Researchers developed a theoretical framework for sustainability-based requirements for the software. The research has provided a path for how to incorporate sustainability in RE.	The practical implementation of the theoretical framework is required in various domains. The evaluation of the framework should be done in real-life projects.

Rama-Dhani et al. [91]	Authors explored the requirement sentences-based classification technique by using the Fuzzy Similarity-based K-Nearest Neighbor to identify and classify NFRs from text sources. This approach takes into account semantic aspects and the measurement of semantic similarity. Summing up, it was shown that by including semantic information precision increased by 43.7% over the FSKNN's 41.47%.	The other algorithms can be considered for comparison and further approaches can be used for live projects.
Zou et al. [74]	Authors discussed the importance of NFRs for developers. The research targets to analyze the NFRs from the programming question-answer website, which gathers the views of developers. The authors have applied the Latent Dirichlet allocation and, Topic Modelling technique, in the research and concluded that the developers focus more on Usability and Reliability than on Maintainability and Efficiency. The authors have studied the millions of posts and comments on stack overflow and guide to understand the NFRs through the developers eyes.	The research has stated that Usability and Reliability get more focus than Maintainability and Efficiency. There is a need to consider factors other than usability and reliability, specifically, Maintainability and Efficiency need to be focused. The performance has been evaluated based on precision and recall. The factors like Accuracy and F-Score measures are required to be considered.

kurtanovic et al. [114]	The research has focused on the challenge of identification of requirements types based on the quality attributes such as NFRs dataset using the SVM (Support Vector Machine), which is a supervised ML technique. Research is mainly focused on usability, security, operational, and performance. In the paper, the under-sampling and over-sampling strategies have been employed to handle the problem of imbalanced classes in the dataset. The required accuracy has been achieved.	The few of the parameters including mean square error, stability, reliability, and coefficient of determination are not considered in the research.
Lu and Liang [159]	Researchers have depended upon supervised learning methods for extracting and considering NFRs from user reviews. The four classification techniques BoW, CHI2, TF-IDF, and AUR-BoW, have been joined with three ML algorithms J48 and Bagging and Naive Bayes, to classify user reviews. Bagging has been inferred as the best technique for NFRs classification. This paper has used techniques such as feature selection, ensembling technique, and bagging to achieve higher accuracy.	The users reviews are taken from iBooks and Whatsapp can, and other applications like Facebook can be considered for validating the model. The ensembling with other algorithms can be applied to get better accuracy. Class imbalance is a major problem in research that need to be considered.

Mats-Umoto et al. [160]	Researchers developed the Non Functional verification method. The authors focused primarily on usability and time-response requirements written in a natural language. The verification method has been established by extending the Requirement Frame model. In this paper, SRS written with a natural language is taken as source code.	Time response and usability are the only NFRs that are taken into consideration. The evaluation of methods for different SRS and considering different NFRs are considered future works.
Kopczynska et al. [49]	The significance of the NFRs templates for inexperienced requirements elicitors based on the e-commerce system has been addressed by the authors. The database of NFRs templates has been used in 41 projects to evaluate the maintenance effort. The paper demonstrates that the templates are superior to the ad-hoc approach to requirement elicitation and improve NFR quality.	Maintainability is the main focus of the research. Another NFRs including usability are required to be focused. Various statistics techniques are used in research but the ML can help in a better way.
Jindal et al. [59]	Authors focused on the extraction and classification of NFRs into 9 categories. A set of 15 projects containing 326 NFR descriptions developed by MS students at DePaul University are used to evaluate the models. The research has concluded that the Naive Bayes outperformed in the case of classifying the availability and maintainability category of NFRs. Using cost-benefit analysis, the researchers were able to see the benefits of the current models.	Authors suggested expanding the work by using datasets specifically academic datasets. The evolutionary algorithms or search methods are required to be explored for the classification of NFRs.

Sabir et al. [161]	Authors developed a multiclass deep learning-based model for classifying the NFRs. The research explored the various deep learning models such as Gated Recurrent Units, Artificial Neural Networks, Long Short-term Memory, and Convolutional Neural Networks. The NFRs are categorized into 5 categories including portability, reliability, maintainability, usability, and efficiency. The data augmentation techniques are discussed by the authors. The research shows that the Convolutional Neural Network outperforms.	The present model is required to be validated on the different corpus. The present augmentation technique does not work well with a small dataset. Moreover, current research does not provide a solution for class imbalance.
Khatian et al. [104]	Authors have focused on the challenge of classifying the NFRs. Supervised ML algorithms are explored and compared for the prediction including decision tree, logistic regression, k-nearest neighbor, and, naive bayes. The total number of NFRs considered is 11 including performance, usability, look and feel, legal, maintainability, security, operability, portability, fault tolerance, and availability.	The present work has only explored the mentioned supervised learning algorithms. The work can be extended with other available algorithms and comparison can be performed with new algorithms.

2.1.2 State-of-the-art Comparative Analysis: ISO / IEC 25010 Domain Specific Applications

In the state-of-the-art literature review, the various NFRs are considered by different researchers. The different definitions of NFRs are listed in Chapter 1. The classification of NFRs also plays an essential role in literature. Moreover, The classification of NFRs is critical for presenting a complete picture of the various categories of NFRs to requirements analysts and customers. Boehm et al. [162] defined a software quality tree that describes the main types of NFRs and their relationships. They classify NFRs according to their general utility. The general utility is sub-categorized into Portability, as-is utility, and maintenance are the three categories. NFRs were grouped into six sub-categories in another classification method presented by Roman [163]. It includes the "interface requirements", "performance requirements", "security requirements", "operational requirements", "economic requirements" and "lifecycle requirements". Grady et al. exhibit their work and define the NFRs as the FURPS model. It consists of attributes such as "Functionality","Usability","Reliability","Performance", and "Supportability" [164]. Mairiza et al. [13] have elicited the 252 NFR attributes from the literature and extracted 114 types of NFRs in the research work. It has categorized these NFRs into Performance, Usability, Reliability, Security, and Maintainability and subclassified these NFRs. ISO/IEC 25010 has given classification schemes based on various quality levels [165]. The classification given by ISO/IEC has considered almost all the important attributes. ISO/IEC 25010 is a world standard for evaluating software and systems quality. The standard was modified three times in 2007, 2011, and 2017 [166]. This standard is also known as the SQuaRE (Systems and Software Quality Requirements and Evaluation) model. It also relates to the software product's quality as well as ease of use., ISO/IEC 25010 was produced as an upgrade to the ISO/IEC 9126 model. According to them, the earlier model (ISO/IEC 9126) has 6 elements and 21 sub-factors. While comparing the two models, the only two actors included in the ISO/IEC model were "security" and "compatibility," together with their sub-factors [39]. ISO/IEC 25010 Starting with the top components and working down to the sub-factors, the quality aspects in this model are presented in order of relevance. The top level is divided into eight components on a lower level, each of which is further divided into subfactors. Quality requirements are identified by ISO/IEC 25010, a derivative of ISO/IEC 9126 [167]. According to this standard, the software product must have certain characteristics, as shown in Table 2.2.

Table 2.2: Categories of NFRs based on ISO/IEC 25010

Main Category of NFRs	**Sub Category of NFRs**
Security	Confidentiality, Authentication, Integrity and Accountability
Performance	Speed, Response Time, Throughput and Availability
Reliability	Completeness, Robustness, Consistency, Maturity, and Accuracy
Usability	Accessibility, User Error, Operability and Learnability
Maintainability	Modularity, Testability, Modifiability and Availability
Portability	Compliance, Adaptability, Co-existence and Replaceability
Efficiency	Time Behavior and Resource Behavior

The ISO/IEC 25010 attributes have been taken as a base for NFRs in the various application domain. ISO/IEC 25010 classification is used in e-commerce, decision support system, application development, and, the expert system as shown in Table 2.3 below.

Table 2.3: Application Domain of ISO/IEC 25010 in Requirement Engineering

Ref. No.	**Title**	**Application Area**
Iqbal et al. [168]	"An approach for analyzing ISO/IEC 25010 product quality requirements based on fuzzy logic and Likert scale for decision support systems"	Internet Banking
Peters and Aggrey [169]	"An ISO 25010 Based Quality Model for ERP Systems"	Education System
Kadi et al [170]	"Quality evaluation of cardiac decision support systems using ISO 25010 standard"	E-Health System
Hussain and Mkpojiogu [171]	"An application of the ISO/IEC 25010 standard in the quality-in-use assessment of an online health awareness system"	Online Health Awareness System

Yildiz et al. [172]	"Analysis of B2C mobile application characteristics and quality factors based on ISO 25010 quality model"	Mobile Application
Izzatillah and Millati [173]	"Quality measurement of transportation service application go-jek using iso 25010 quality model"	Transpotation Service Application
Alsareet et al. [174]	"Incorporation of ISO 25010 with machine learning to develop a novel quality in use prediction system (QiUPS)"	E-Learning
Alharthi et al. [175]	"Sustainability requirements for eLearning systems: a systematic literature review and analysis"	Education System
Wattiheluw et al. [176]	"Development of a Quality Model Based on ISO 25010 Using Fuzzy and PSO for E-commerce Websites"	E-commerce Websites
Dewi et al. [177]	"Maintainability Measurement and Evaluation of myITS Mobile Application Using ISO 25010 Quality Standard"	Academic Information System (AIS)
Panduwiyasa et al. [178]	"Accounting and Smart System: Functional Evaluation of ISO/IEC 25010: 2011 Quality Model (a Case Study)"	Accounting and Smart System
Atanacio at al. [179]	"Development and Evaluation of Rural Health Unit Record Management System with Data Analytics for Municipality of Bay, Laguna using ISO 25010"	Health System
Felipe et al. [180]	"Evaluation of user embracement software with pediatric risk classification"	Risk Classification
Pratama and Mutiara [181]	"Software Quality Analysis for Halodoc Application using ISO 25010:2011 "	Telemedicine
Shiratuddin et al. [41]	"Evaluation of e-Book applications using ISO 25010 "	Electronic Book
Dewi et al. [177]	"Maintainability Measurement and Evaluation of myITS Mobile Application Using ISO 25010 Quality Standard "	Mobile Application

Acharya and Sinha [182]	"Assessing the Quality of M-Learning Systems using ISO/IEC 25010"	M-Learning System

There are various classifications given for the NFRs in literature. But ISO/IEC 25010 is the most appropriate one and is used by most researchers. The above table shows that ISO/IEC 25010 is applied to cloud computing, classification, Data Analysis, and the Internet of Things. The current research has considered the ISO/IEC classification in current research.

2.1.3 State-of-the-art Comparative Analysis: ML/DL in NFRs

The present work focuses on the intersection of ML/DL and NFRs. ML, defined by Arthur Samuel (1959) [183] as "the ability of computers to learn without being explicitly programmed". ML is a branch of artificial intelligence that examines the research and development of a model that learns from raw data, trains the system, and provides predictions based on this trained data. In today's environment, ML is a prominent research topic that executes tasks without being explicitly programmed. In ML, a function, model, or algorithm is created to learn and perform extractions from existing datasets known as training datasets. ML has a subfield called deep learning. Deep Learning (DL) is a cutting-edge ML technique that allows computers to automatically extract, analyze, and comprehend relevant information from raw data [184]. The system learns the intricate link between input and output using a non-linear model with many hidden layers of architecture. DL has the advantage over ML in that it does not require explicitly extracted or constructed features. DL automatically retrieves features from raw data, processes them, and then makes a prediction based on them [185].

Table 2.4 has depicted the comparative analysis of various parameters such as the dataset used, ML/DL methods, evaluation metrics, type of ml algorithm, and, NFRs identified.

TABLE 2.4: Comparative Analysis of ML and NFRs

Ref.	**Focused Area**	**ML Method**	**Evaluation Metric**	**Source Doc**	**Results**	**Su**	**Sm**	**Un**	**En**	**NFR Identified**
(Canedo and Mendes, 2020)[186]	NFRs Classification	LR,SVM, MNB, K-NN	Precision, Recall, F-Measure	Text Document	F-measure= 91%	✓	✗	✗	✗	Performance, Security
(Ahmed and Daleel, 2020)[23]	Classify FRs and NFRs	Neural Network	Precision, Recall, Distance Metric	SRS Document	Precision= 90%, Accuracy= 87%,	✓	✗	✗	✗	Performance, Usability, Security
(Naseem et al., 2020)[149]	Prediction of requirements risk	KNN, Random Forest, Decision Tree, SVM	Mean absolute error, Root mean square error,	UCI, Medical, Online shopping	MAE= 89%, RMSE= 44%	✓	✗	✗	✓	Performance, Access control
(Kurtanovic and Maalej, 2017)[114]	Quality requirements	Support Vector Machine	Precision, Recall, F1	SRS, RFP archives, and User Comments	Precision= 92%, Recall= 92%	✓	✗	✗	✓	Usability, Security, Operational, Performance
(Lu and Liang, 2017)[159]	Classification of NFRs from User Reviews	Naive Bayes, J48, Bagging	Precision, Recall, F1	User Reviews from Apple App Store and WhatsApp Reviews from Google Play	F-measure= 71.8%, Precision= 71.4%, Recall= 72.3%	✓	✗	✗	✓	Reliability, Usability, Portability, Performance
(Martino et al., 2018)[142]	Requirement related to Cloud Computing field	Naive Bayes, Maximum Entropy, CNN	✗	PROMISE repository	✗	✗	✗	✗	✗	Availability, Maintainability

It is very important to discuss the various ML parameters which are used for designing the ML model for NFRs classification and prediction. The various parameters include Feature Selection and Ensembling. ML/DL works on a simple rule “if you put garbage in, you will get the garbage out”. Feature selection (FS) is the process of identifying a small number of linked features that are sufficient for learning the target concept. FS refers to the concept of feature relevance, redundancy, and complementarity (synergy). FS has been investigated by the data mining and ML/DL sectors. Features are sometimes referred to as attributes or variables [187]. Features are the attributes of the system that have been measured and designed from the original input variables. It’s a technique for extracting the most significant characteristics from the original features to reduce the dataset’s size. In data mining, feature selection is a crucial pre-processing step that identifies useful feature subsets for classification[188]. The main goal of the FS is to retrieve the optimal subset from the feature set that yields the lowest generalization error. FS can be segregated into three categories, namely Wrapper, Filter, and Embedded method.

Ensembling is a crucial component of ML. In the context of ensembling, different classifier systems are employed. It is highly successful and quite adaptable when resolving issues across a wide range of fields. Any ML technique can be used as the basis for the ensemble method, which splits the same training dataset and applies the same algorithm to different parts of the same dataset in different ways. The ensembling method combines several models to construct an output model. By combining several methodologies and models, the ensemble method seeks to produce better outcomes [189]. Voting, stacking, bagging, and boosting are different ensemble methods. Various researchers have explored the different ML parameters for NFRs.

This section contains a comprehensive review of the existing NFRs elicitation techniques. Though the review of literature affirms an annexure of ML-DL and NFRs as an established field of research, it glares upon various constraints and unexplored areas in both domains. Some of the significant research works are as follows:

Jindal et al. [59] focused on the extraction and classification of NFRs into 9 categories. A set of 15 projects containing 326 NFR descriptions developed by MS students at DePaul University are used to evaluate the models. The research has concluded that the Naive Bayes outperformed in the case of classifying the availability and maintainability category of NFRs. Using cost-benefit analysis, the researchers were able to see the benefits of the current models. Rahimi et al. [127] addressed the issue of classification of requirements. The authors have developed a novel Ensembling-based ML model. The research work explored the voted ensembling technique and coupled various ML algorithms such as Support Vector Classification, Logistic Regression, SVM, Decision Tree, and Naive Bayes. The 99.45% accuracy has been attained by the proposed model. The experiment was evaluated on the primary dataset. Slankas et al. [154] focused on the NFRs classification. The research used a K-nearest neighbor (KNN) supervised learning method and compared its performance to SVM and Nave Bayes techniques. SVM outperformed the Multinomial Nave Bayes classifier and the KNN classifier outperformed the optimum Naive Bayes classifier using a unique distance measure, according to the results of the study. The manuals, requests for proposals and system requirements, and specification documents have been used as source documents for research. Rahy and Bass [190] explored the challenge of dealing with NFRs in an agile environment. The study includes 18 interviews with agile software development practitioners from two international businesses. A technique guided by grounded theory and information flow models was utilized to compare and contrast interactions between processes in the interviews that were recorded, transcribed, and analysed. In one case study, the NFRs were managed as artifacts in the agile methodology, while the other company reverted to traditional plan-based software development methods of documentation, timeframe estimations, and safety-critical requirements. This study compares and contrasts these two methodologies in great detail. The researchers have designed a set of recommendations for handling NFRs in regulated environments by utilizing agile methodologies is the key contribution of this work. Singh et al. [61] proposed that a rule-based classification technique has been used in research to identify and categorise the requirement sentences into NFR sub-classes utilizing thematic roles. The results were checked against the PROMISE and Concordia corpora. F1 is a performance statistic that has been used in the past. NFR classifications are based on ISO/IEC quality characteristics. Mahmoud and Williams [73] detected and classified NFRs using a multi-step, unsupervised technique. To train the classifier, early systems relied on manually classifying the data. However, big data sets are not always accessible, making it difficult to achieve high accuracy. To aid in the traceability of NFRs, a method is employed to extract natural language information from source code. In the area of software requirements, semantic similarity algorithms are employed. The most logical set of required words can be generated using cluster configuration. Gnanasekaran et al. [191] used ML approaches to automatically classify distinct forms of NFRs and that is examined in this research. An author developed a recurrent neural network model which has been examined for its effectiveness in classifying NFRs into 5 separate categories such as operability, usability, maintainability, performance, and security. An average precision of 84%, a recall rate of 86%, an F1-score close to 84%, and an accuracy of 88% have been achieved in the experiments. NFRs may be classified using this method with an accuracy of 88%. The dataset for the experiment has been taken from the PROMISE repository and International Requirements Engineering Conference's 2017 Data Challenge data set for experiment purposes. Canedo and Mendes [186] proposed an ML model for categorizing the requirements into NFRs and FRs and further sub-categorizes NFRs. A multi-class classifier model has been designed and implemented with Naive Bayes, Logistic Regression, SVM, and k-Nearest Neighbors ML algorithms. The experimental simulation has been performed using a dataset from the PROMISE repository. The metrics for performance evaluation used in the research are F-measure, Precision, and Recall. The current research has attained the 91% F-measure for the classifier. Baker et al. [146] tested two deep learning techniques as Convolutional Neural Networks (CNN) and Artificial Neural Networks (ANN) to identify the NFRs into 5 categories such as usability, maintainability, performance, operability, and security. With a halved number of training samples, the ANN model was trained in four classes such as usability, operability, security, and performance. On the other hand, the CNN was trained using the entire dataset, which included all of the training samples required to meet the security requirements. According to the results, the CNN model was found to outperform the ANN model in both the Performance and Security categories. Haque et al. [92] examined the results of this study, which included a combination of seven ML techniques such as Bernoulli Naive Bayes, Gaussian Naive Bayes, Multinomial Naive Bayes, K-Nearest Neighbors, Decision Tree, Stochastic Gradient Descent SVM and Support Vector Machine and four feature extraction techniques such as Term Frequency Inverse Document Frequency at the character level and word level, and Bag of words are explored. The dataset for the experiment has been taken from the PROMISE repository. As a result, Stochastic Gradient Descent SVM and Term Frequency Inverse Document Frequency was determined to be the best combination for classifying NFRs, with the highest precision, recall, and F1 score. Baker et al. [146] analyzed that the use of ML for decision-making during SDLC is not all-encompassing because of the parameter adjustment required by the various models. It is now possible to classify NFRs using an effective methodology developed by researchers. An ANN (Artificial Neural Network) and CNN (Convolutional Neural Network) were used as a ML learning strategy. Research has focused on NFRs in terms of their, usability and maintainability. NFR classification has been analyzed using the Precision and F1 metrics, and it has been concluded that the CNN is superior. Hey et al. [192] designed a novel NoRBERT technique for categorizing requirements. A binary classifier has been implemented to categorize NFRs and FRs, whereas a multi-class classifier has been designed to subclassify the NFRs. The dataset has been taken from the PROMISE repository for simulation. NoRBERT approach has been compared with other techniques and it outperforms and attained a 94% F1 score to classify and sub-classify requirements. Rahimi et al. [193] implemented a three ensemble-based approach for the classification of requirements. Research has combined various deep learning techniques such as bidirectional long short-term memory, long short-term memory, gated recurrent unit, and a convolutional neural network. The model has been trained and tested on the PROMISE dataset. The current model has obtained an accuracy near 95% for binary classification and multi-class classifier it attained 93%. Casamayor et al. [79] proposed the semi-supervised text categorization technique developed by using the Naive Bayes algorithm to find NFRs in textual data. The pre-categorization of requirements in the previous supervised approach demands a great deal of manual effort. Using the identical standards documents, the semi-supervised strategy exhibits accuracy above 70%, which is greater than the supervised learning results. In their study, Rahimi et al. [89] looked at data mining methods for locating NFRs. Automated detection of quality issues such as usability, security, and performance from a document is achieved using a variety of data mining and ML techniques in this study. The authors tested their approach against two frameworks linked to human services and found it to be effective.

Ramadhani et al. [91] explored the requirement sentences-based classification technique by using the FSKNN (Fuzzy Similarity-based K-Nearest Neighbor) to identify and classify NFRs from text sources. This approach takes into account semantic aspects and the measurement of semantic similarity. Summing up, it was shown that by including semantic information precision increased by 43.7% over the FSKNN's 41.47%. Slankas and Williams [154] developed NFR Locator to classify the natural language documents into 14 predetermined categories. A K-NN classifier is used to identify sentences that are similar across documents.

The research uses a variety of classifiers, where the K-NN classifier was able to get the best results for NFRs classifications. Mahmoud and Williams [73] considered an unsupervised technique to identify and classify NFRs. Data classification is used to train the model in the early stages of development. The authors have determined that the classifier requires a big training data set that is not always readily available to achieve high accuracy. Riaz et al. [80] have focused on security by considering security-based Sentences. It also provides context-specific security requirements templates that aid in the translation of security-relevant statements into FRs. The authors have achieved 82.2% accuracy. Gazi et al. [194] implemented a classification scheme for NFRs to Information systems (IS), Web-based systems, and real-time systems. According to research, two significant NFRs for information systems are reliability and availability, with the accuracy, maturity, and completeness of the reliability requirements being further broken down. Aasem et al. [195] have inferred the importance of requirement prioritization as a base for decision-making in further phases. The developer must decide what to create in the principal release and the coming discharges based on prioritization. The paper focused on the weaknesses and strengths of different prioritization techniques and proposed a novel framework by combining existing approaches and methods. Aljallabi and Mansour [196] clarify appropriately that dismissing the NFRs is probably the most significant impediment in Agile. There is no standard system for the elicitation and management of NFRs. This paper shows two existing NFRs investigation methodologies. Afterward, it gives another upgraded way to examine NFRs better, consolidating their qualities and defeating their shortcomings. Maiti and Mitropoulos [45] propose a study to effectively gather NFRs Metadata from software requirements artifacts such as documents and images. The historical trending approach is utilized to foresee whether extra NFRs are disregarded and can be incorporated with FRs at the beginning of Agile software development. The objective of the examination is to enhance earlier investigations of NFRs to give viable methods to organize and foresee NFRs and their effect during the beginning times of software development. The author has suggested that there is an absence of research to assemble NFRs from pictures. There is a need for a model that fits all NFRs as these differ in nature. Existing FRs can be utilized to foresee extra NFRs are helpful for software development. Lu and Lang [159] have depended upon supervised learning methods for extracting and considering NFRs from user reviews. The four classification techniques BoW, CHI2, TF-IDF, and AUR-BoW, have been joined with three ML algorithms J48 and Bagging and Naive Bayes, to classify user reviews. Bagging has been inferred as the best technique for NFRs classification. This paper has used techniques such as feature selection, ensembling technique, and bagging to achieve higher accuracy. Martino et al. [142] evaluate that the requirement specification is a complex task concerning, cloud computing, particularly with developing stakeholders who have ever-changing needs. So, the authors have proposed automatic modeling and classification of requirements stated in natural language form. The target data has been used from the PROMISE repository. Abad et al. [138] contributed a methodology for pre-handling requirements that normalizes and standardizes requirements before applying classification algorithms and improves performance. Different ML techniques have been compared. The focused NFRs are usability, availability, or performance. Kiran and Ali [197] explored that the requirement elicitation process is very complex and critical as engineers from various locales of the world build up the framework, So it's tough to accumulate requirements for such frameworks. This paper focuses on how the procedure of requirement elicitation is completed for open-source software and the various ways utilized to streamline the process of requirement elicitation by using a variety of tools, strategies, and methodologies. Tiwari and Rathore [198] presented an approach to pick a subset of techniques for an optimal output as part of the Requirement Elicitation process. Requirement engineering is heavily influenced by three factors: people, processes, and projects. This work aims to provide significant insights into the features of diverse requirements elicitation approaches. A series of case studies will be used to evaluate and offer context for the selection of the Requirement Elicitation technique.

Asadi et al. [199] proposed the framework for automatically selecting suitable features that satisfy all types of requirements like functional, non-functional, and stakeholder constraints. This approach improves the feature model configuration through the positive and negative impact of the features. The study combined the Analytical Hierarchical Process and Fuzzy Cognitive, and then Hierarchical Task Network has applied for finding the optimal set of features. The significance of the NFRs templates for inexperienced requirements elicitors based on the e-commerce system has been addressed by Kopczynska et al. [5]. The database of NFRs templates has been used in 41 projects to evaluate the maintenance effort. The paper demonstrates that the templates are superior to the ad-hoc approach to requirement elicitation and improve NFR quality. Groen et al. [200] focused on requirements elicitation from many online reviews by using an automatic approach. The functional aspects and user feedback concerning quality issues are taken into consideration. The outcomes demonstrated that online reviews are an unexploited Big Data hotspot for quality requirements. Kopczynska and Nawrocki [156] investigated the Structured Elicitation of Non-functional Requirements(SENoR) strategy. The technique consists of a succession of short meetings to generate new ideas driven by ISO 25010 Quality attributes. It utilizes Non-Functional Requirements Templates (NoRTs) to help the elicitation procedure. Our exploratory contextual analysis of the expense and viability of SENoR and NoRTs included 7 projects that created customized web applications.

Slankas et al. [154] used Naive Bayes, A supervised ML Algorithm. As shown in the table that the Evaluation Metric for most of the models is Precision, Recall, and F Measure. The different works focused on different NFRs. Maiti and Mitropoulos [201] explored the prioritization out of the Capture Elicit and Prioritizing (CEP) approach. This research focuses on improving the prioritizing of NFRs during the starting phases of an Agile development process. The authors have used the existing framework for prioritizing the NFRs, which were already used for FRs. In the research, it is focused that the prediction of NFRs is valuable for software engineers. Zou et al. [70] discussed the importance of NFRs for developers. The research targets to analyze the NFRs from the programming question-answer website, which gathers the views of developers. The authors have applied the Latent Dirichlet allocation, i.e., the Topic Modelling technique, in the research and concluded that the developers focus more on Usability and Reliability than on Maintainability and Efficiency. The authors have studied the millions of posts and comments on stack overflow and guide to understand the NFRs through the developer's eyes. Lin and Huang [202] explored how the software designing life cycle procedures can be specific for Big Big Data processes. The researchers focus that how the NFRS is often ignored in Agile as well as in traditional approaches. Agile methods are rapidly gaining popularity, especially SCRUM and Extreme Programming have quickly delivered a system that meets FRs, but still, the NFRs are ignored. Sachdeva and Chung [134] proposed a novel approach to handling NFRs for Big data projects in an Agile environment. The contextual analyses exhibited in the paper unmistakably outline the need to present NFRs, for example, Security and Performance, right off the bat in the software lifecycle, whose absence of concerns can likewise seriously harm other essential NFRs. The authors have also suggested a need to address process-related NFRs for Big Data and cloud projects.

Mahalank et al. [203] focused on the requirement to analyze the Non-Functional requirements for making design decisions. The authors have discovered a template and checklist-based strategy for examining the NFRs for the Internet of Things-based traffic board unit. The design model choices are represented by NFRs structure parameters like cost, storage capacity, sensitivity, advancement process, reaction criteria, design complexity, and ecological effect. Investigating NFRs and further actualizing the arrangements in a design model is crucial for including the NFRs in the development process. Maiti and Mitropoulos [45] explored that the Agile software engineering methods often consider FRs for rapid delivery and

strongly neglect the NFRs. The authors have developed an automatic approach for capturing, eliciting, predicting, and prioritizing NFRs.
Winkler and Vogelsang [204] developed an automated tool for helping users by providing warnings for classifying the specification document into the FRs and NFRs. The Neural Network approach has been used for classification. The controlled experiment is done for showing the benefit of the tool. The outcomes demonstrate that the utilization of an intuitive system has given the high exactness. Open-source software, such as Firefox, Lucene, and Mylyn, were tested for 50 NFRs by Bhowmik and Do [205]. It is being studied that Following a preliminary study that was more focused on implementation than requirements, this study is a follow-up to that. On the other hand, this study focuses more on NFRs requirements. Nguyen et al. [206] addressed the Internet of Things software optimization issue concerning fulfilling NFRs in the application design phase. The author has explored that a significant challenge is to satisfy the different NFRs while designing the IoT system. A model-driven technique has been proposed to enhance an IoT application for NFRs. In the result, it is exhibit that NFRs such as power consumption and reliability can be improved significantly during the optimization process. It is suggested that parallel processing or cloud servers can be used for reducing the runtime of optimization. Arruda [15] explored that the Requirements Engineering (RE) plays a crucial role in big data and specifying the need for NFRs concerning Big Data. The proposed work is tending to the examination, particular, and documentation of information quality requirements and their association with objectives of association with Big Data frameworks just as its effect on the software development process. The questionnaires and templates are used as a data collection tool. In the research, a systematic view has been done to explore Big Data and its application areas. Noorwali et al. [207] focused on the most effective method to efficiently deal with Quality Requirements, including Big data attributes. In the paper, the methodology proposed for dissecting and determining quality requirements for Big Data Applications. The fundamental thought

is to cross a Big Data trademark with a Quality trait. This methodology consolidates three components - Big Data trademark, Quality properties, and Quality requirements depiction and guarantees that the Big Data attributes are addressed in the designing of quality requirements. Matsumoto et al. [160] developed the Non Functional verification method. The authors focused primarily on usability and time-response requirements written in a natural language. The verification method has been established by extending the Requirement Frame model. In this paper, SRS written with a natural language is taken as source code. Abad et al. [138] investigated how the different ML approaches have worked to classify requirements into FRs and NFRs and further classify NFRs into sub-categories. The research contributes to a methodology for pre-processing requirements that normalizes and regularises requirements before applying ML algorithms. It also improves the classification process. The experiment was done on 625 requirements provided by the Promise repository. The authors have discovered that there is a vast difference in the performance of various algorithms such as Latent Dirichlet Allocation, and Naive Bayes for the further classification of NFRs. Kurtanovic and Maalej [114] focused on the challenge of identification of requirements types based on the quality attributes such as the NFRs dataset using the SVM (Support Vector Machine), which is a supervised ML technique. Research is mainly focused on usability, security, operations, and performance. In the paper, the under-sampling and over-sampling strategies have been employed to handle the problem of imbalanced classes in the dataset. The required accuracy has been achieved. Ezami [141] examined how often NFRs are referenced implicitly or explicitly in the source code remarks by utilizing Natural Language Processing (NLP) methods and assess how adequately they can be extracted utilizing ML. The best execution was accomplished using the SVM classifier, with an F1 measure of 0.86%. The outcomes demonstrate that the supervised technique outflanks than unsupervised techniques. The author has compared the outcomes with past investigations and demonstrates that NFRs can be identified more precisely from source code comments contrasted with other software relics, for example, SRS and RFP archives. It is likewise discovered that bag of words performs better than doc2vec in extracting NFRs from source code reviews. Portugal et al. [144] analyzed the unstructured data to identify mining the NFRs. NFRFinder is a semi-automatic tool developed by researchers. When it was first developed, it was utilized to analyze structured text using Recall and Precision metrics as performance indicators. For example, NFRFinder has provided encouraging results in structure and demonstrated that NFR categorization is affected by the context and stakeholders participating in classification. Taj et al. [148] focused on defining requirements for FRs and NFRs.The authors have created a model to assist in the process of eliciting and classifying requirements. Crowdsourcing was used to acquire the Requirements, and many stakeholders were involved in the elicitation process. Decision Trees and Naive Bayes were used in the classification model. The case study was able to demonstrate the model's abilities. Li et al. [208] designed a novel model named as NFRNet for the extraction of NFRs.The researchers have made use of a deep neural network and PROMISE corpus for experimental purposes. The authors have shown that the proposed model outperforms with an accuracy of 91%. Toth and Vidacs [145] analyzed the interest in ML techniques and NLP to evaluate the application and execution in software engineering.ML Techniques like Naive Bayes, Support Vector Machine(SVM), Logistic Regression, and Decision Trees are used in research. The recall metric is utilized to measure performance in Precision and F1. The experimental dataset is taken from the Stack Overflow and promise repository. Salman et al. [209] proposed a methodology for clubbing requirements into the semantic cluster, which is based on the similarity of the text between the requirement statements. The agglomerative hierarchical clustering algorithm is used in the framework, and the analysis is done utilizing open-access software to accomplish the proposed work better. Various authors did a comparative study of clustering algorithms to find the keywords from clusters that define the domain knowledge. Ali et al. [204] suggested that NFRs should be handled with care and considered in the early stages of software development to avoid system failures in the future. To avoid Requirement rejection, NFRs must be addressed effectively. The authors have proposed an Agile strategy for dealing with NFR dependencies after examining all conceivable situations between NFRs. There is a solution suggested here that is both cost-effective and capable of dealing with complicated requirements.

Based on the literature survey as described previously, we critically examined the classification/clustering method used, the dataset for an experiment, and the Evaluation metric, which parameters are used to evaluate the performance of the ML model. FS and Ensembling are being used or not. Moreover, NFRs used in various research are identified. The key parameters such as Focused Area, Dataset, ML method, Evaluation Metric, Source Document, Results, FS, and type of learning such as Supervised (Su), Semisupervised (Sm), Unsupervised (Un), Ensembling(En), and NFRs explored are compared and described.

2.1.4 State-of-the-art Comparative Analysis: Performance Metric

The performance of the ML model is evaluated using various metrics. The performance of the prediction model is compared based on mathematical formulas. The different evaluation metrics are given, such as Precision, Recall, Accuracy, Sensitivity, Specificity, and F-measure. All the evaluation metrics are derived from the confusion matrix. It is very significant to consider all the metrics which cover all the aspects of model evaluation. It is very important to mention that none of the metrics can tell the full performance of the ML model. It is very important to consider the combination of different metrics. The details of the various metrics are as below:

Precision: It measures the percentage between the sum of correctly classified data and the total of data. In other words, it is performed by using the ratio of the sum of correctly classified to the sum of total classification evaluated. Mathematically, it can be written as

$$Precision = \frac{TP}{TP + FP}$$

where TP stands for the total or sum of the true positives. (TP+FP) stands for the quantity of positive whereas FP refers to the false positive. The higher the value of precision means more correctly classified samples. For instance, precision=1 tells us that all the examples or instances are correctly classified.

A recall is another significant metric. It is also known as Sensitivity. It is a ratio of positive instances that are detected correctly by the model. Mathematically, it is represented as

$$Recall = \frac{TP}{TP + FN}$$

Where TP stands for true positive and FN refers to a false negative. The recall is a ratio of the number of times a class was correctly predicted to the number of times the class appears in the test data.

The F1 measure is another metric for evaluating the model. It is also referred to as the F1 score. It is a combination of Precision and Recall. It is the harmonic mean of Recall and Precision. The harmonic mean is preferred over the normal mean. It has given more weightage to the lower values, and the mean has equally treated all values. Mathematically, it is defined as

$$F - measure = \frac{2TP}{2TP + FP + FN}$$

Accuracy refers to the proportion of correctly classified examples. It is a more accurate measure as it calculates the number of correctly classified instances on the whole. Specifically, accuracy is most suitable for a balanced target class. Mathematically, it can be defined as

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Specificity is the metric that measures a model's ability to accurately forecast the actual negatives of each of the categories it can predict. It is defined as follows:

$$Specificity = \frac{TN}{TN + FP}$$

where TN stands for true negative and FP stands for false positive.

The various research works have considered different evaluation metrics. Table 2.5 shows the various focused metrics in the different research works.

TABLE 2.5: State-of-the-art Comparative Analysis: Performance Metric

Research Work	FRs	NFRs	Precision	Recall	F-Score	Accuracy
(Canedo and Mendes, 2020)[186]	✗	✓	✓	✓	✓	✗
(Khatian et al.,2021) [104]	✓	✓	✓	✓	✗	✓
(Sabir et al., 2020)[210]	✗	✓	✗	✓	✗	✗
(Ahmed and Daleel, 2020)[23]	✓	✓	✓	✓	✗	✗
(Jindal et al., 2021)[59]	✗	✓	✗	✓	✗	✗
(Quba et al., 2021)[211]	✓	✓	✓	✓	✓	✗
(Hey et al., 2020)[192]	✓	✓	✗	✗	✓	✗
(Lu and Liang, 2017)[159]	✗	✓	✓	✓	✓	✗
(Rahimi et al., 2020)[127]	✓	✓	✓	✓	✓	✓
(Naseem et al., 2020)[149]	✗	✗	✓	✓	✗	✗

Table 2.5 presents the different combination of performancr metrics are utilised by various research works.

2.1.5 State-of-the-art Comparative Analysis: Usability NFR

The limited research scope does not justify the usability perspective as it focuses more on security and other functional aspects. Usability is a very important factor as it deals with users experience with the software when users interact with it. The various factors such as efficiency, ease of use, the efficiency of use, and overall satisfaction of the user with the software. Usability is one of the factors of ISO/IEC 25010. The various research works have focused on other factors like security, portability, maintainability, and compatibility. The limited research has considered the usability factor, which is quite significant to be considered [70] [71] [72][73]. The various authors have focused on the different NFR factors of

ISO/IEC 25010. Table 2.6 has shown the various NFRs focused on by different researchers. In the current work, we are focused on Usability as minimal research has focused on it.

Table 2.6: Identified and focused NFRs

ISO IEC 25010	[74]	[75]	[76]	[73]	[13]	[77]	[78]	[79]	[80]	[81]	[82]	[83]	[56]
Functional Suitability	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗
Efficiency/ Performance	✓	✗	✗	✓	✓	✗	✗	✓	✗	✗	✓	✗	✗
Compatability	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Usability	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗
Reliability	✓	✗	✗	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗
Security	✗	✓	✗	✓	✓	✗	✗	✓	✓	✓	✗	✓	✗
Maintainability	✓	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✓
Portability	✗	✗	✗	✓	✓	✗	✗	✗	✗	✓	✗	✗	✗

2.1.6 State-of-the-art Comprehensive Comparative Analysis

The literature review has critically evaluated the various key parameters which are depicted in Table 2.7. The parameters are Pri. data., Sec. Data. , Acc., Pre. Rec., F-Sc., FS, Hyb. FS, Foc. NFRs, SOM Visualization, Sta. Eff. and Rel. Eff. represent the Primary Dataset, Secondary Dataset, Accuracy, Recall, F-Score, Feature Selection, Hybrid Feature Selection, Focused NFRs, SOM Visualization, Stability Efficiency, and Reliability Efficiency respectively. The various focused parameters are described below:

1. Type of dataset used: The type of dataset used in research is segregated into a primary dataset and a secondary dataset. The type of dataset is mentioned in Table 2.7.

2. Evaluation metric: The different performance evaluation metrics are used in various research including Precision, Recall, F-score, and Accuracy. The type of evaluation metric used by various researchers is mentioned.

3. Feature Selection technique: FS is very important to select the significant features out of all features. The prediction of NFRs has explored baseline and Hybrid FS techniques. As mentioned in Table 2.7 that minimal studies have explored the hybrid FS. Still, there is a need to explore the different hybrid FS techniques for NFRs.

4. SOM technique explored: Table 2.7 has represented that the NFRs-based research has minimally considered the SOM visualization technique. It is a color-coded representation of results. Henceforth, it is very significant to consider SOM in research.

5. Reliability and Stability efficiency: Reliability refers to the ability of the ML model to carry out and maintain its operations under unexpected conditions. An ML model is considered to be stable if the prediction does not change strongly over the short range of data. It is very crucial to consider reliability and stability efficiency. It is depicted in Table 2.7 that minimal research has considered reliability and stability efficiency. So it is very important to consider the reliability and stability efficiency in the domain of NFRs.

6. Tool/Techniques: The classification and prediction of NFRs are done through various ML techniques including supervised and unsupervised. It has depicted that the various researchers have explored different techniques including SVM, Naive Bayes, Decision tree, RNN, ANN, and Random forest. Some research works have used the baseline techniques and others use ensembling techniques. Still, there is scope for exploring the ML algorithms and different ensembling techniques in the domain of NFRs.

   The current research has explored the primary as well as the secondary dataset. The hybrid FS technique has been used for FS. The model has been evaluated based on accuracy, precision, and recall. The reliability and stability efficiency has evaluated for the model. The SOM visualization technique has been explored for visualization.

Table 2.7: Comprehensive analysis with related studies

Ref.	Pri. Data.	Sec. Data.	Acc.	Pre.	Rec.	F-Sc.	FS	Hyb. FS	Foc. NFRs	SOM Vis.	Sta. Eff.	Rel. Eff.
(Rahimi et al., 2020) [127]	✓	✗	✓	✗	✗	✗	✓	✗	✓	✓	✗	✗
(Gnan asekaran et al., 2021) [191]	✗	✓	✓	✗	✓	✓	✓	✗	✓	✗	✗	✗
(Li et al., 2021) [208]	✗	✓	✓	✓	✓	✓	✗	✗	✓	✗	✗	✗
(Bhatia et al., 2018) [85]	✓	✗	✗	✓	✗	✗	✗	✗	✗	✓	✓	✗
(Haque et al., 2019) [92]	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗
(Baker et al., 2019) [146]	✗	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗	✗
(Jindal et al., 2021) [59]	✗	✓	✗	✗	✓	✗	✓	✗	✓	✗	✗	✗
(Naseem et al., 2021) [149]	✗	✓	✗	✓	✓	✗	✗	✓	✓	✗	✗	✗
(Behal and Singh, 2021) [88]	✗	✓	✓	✗	✗	✗	✗	✗	✗	✓	✗	✓
(Canedo and Mendes, 2020) [186]	✗	✓	✗	✓	✓	✓	✓	✓	✓	✗	✗	✗
(Ezami, 2018)[141]	✓	✗	✗	✓	✓	✓	✓	✗	✓	✗	✗	✗
Current Re-seach	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓

2.1.7 State-of-the-art Review on NFRs Elicitation

There are defined methods for eliciting FRs, but there is no standard technique for eliciting NFRs. One of the more difficult tasks in requirement analysis is eliciting NFRs. The various authors have utilized different NFRs elicitation techniques and tools for their work. The tools have been designed for NFRs identification and elicitation by researchers. The state-of-the-art research works are discussed in Table 2.8.

Table 2.8: Important research milestones in NFRs Elicitation

Ref.	**Research Contribution**	**Future Aspects**

[212]	Balushi et al. present a tool namely ElicitO as a requirement elicitation tool for eliciting precise NFRs specifications. The tool is based on ISO/IEC 25010. The tool serves as a memory aid for structuring elicitation interviews, directing requirements analysts concerning key quality considerations about a class of applications, and supporting the development of precise requirements based on features and metrics found in quality model standards. The Currently available elicitation tools including Domain Analysis, JAD, and Scenario-based elicitation have focused on the FRs. Manchester Unity Web Project has used the ElicitO for elicitation purposes.	Future work will be focused on using ElicitO in various domains to elicit NFRs precisely.
[25]	Younas et al. discussed that the NFRs are given less attention due to a lack of knowledge about NFRs. The present research has given the NFRs elicitation guidelines for users as well as for developers. The guidelines are focused on Agile-based software development. Moreover, the study has discussed how cloud computing works with agile approaches, particularly in the elicitation process. The eProcurement document has been explored for extracting NFRs in the present research.	The proposed guidelines will be used in other case studies in the future, especially in industry. Moreover, a tool will be developed for helping the stakeholders based on the present guidelines.

[153]	Farid has focused that an Agile-based method such as Scrum is not adequately identified and manage NFRs during the early phases of software development. The research has presented a NORMAP method for identifying and managing the Agile Choose Case, Agile Use case, and Agile Loose Case. The framework is dedicated to 25 specific NFRs. The method has been validated using the simulation tool as well as utilizing case studies.	Further study is required to evaluate the framework in actual agile development projects, enhance NFRs recognition and text mining, and link the framework with online, mobile, and social media technologies.
[68]	Silva et al. proposed the guidelines for NFRs elicitation. The researchers have considered the challenge of the lack of a standard technique for NFRs elicitation. The elicitation guide aims to help the developer in identifying and elicit NFRs. The guide consists of different questions along with a description and association with the issue. The elicitation guide has been used by governmental software development companies and positive results are achieved.	The research's subsequent steps aim to include a guide to various software development projects and evolve the procedure in line with user experience. Moreover, a tool will be proposed for better NFRs elicitation. Furthermore, the significant NFRs selection based on the multicriteria model will be studied in the future from the customers viewpoint.

[156]	Kopczynska and Nawrocki focused on the question that how to get a balance between the cost and effectiveness (value) of the NFRs elicitation process. The researchers have Investigated SENoR by making use of NoRTs. SENoR refers to the structured elicitation of NFRs and NoRTS is an NFRs template for NFRs elicitation. The workshop consists of brainstorming sessions conducted for each parameter of ISO 25010. It results in a stability level of 80%. The 7 projects which developed the web applications were explored for the case study of the SENoR.	In the future, other organisations and projects will be explored for further study of the cost and effectiveness of the SENoR. The other elicitation methods will be explored in real projects for NFRs elicitation.
[5]	Kopczynska et al. evaluated the usefulness of NFRs templates for inexperienced requirement elicitation. The researchers have investigated the 41 industry projects and analysed 2231 NFRs. The work discussed the usefulness of the catalog of NFRs templates.	The fastest method of browsing the NFRs catalog based on some ML technique or ontology will be discussed in the future. Furthermore, The NFRs catalog will be enriched with online feedback systems including an app store.

[213]	Veleda and Cysneiros introduced a tool that helps to implement the NFRs. This tool includes preliminary search techniques to make it easier to find potential solutions for NFRs. The NFR tool is designed in the present research as a support mechanism built on a previously established ontology to facilitate the information gathering necessary to satisfy NFRs while enabling queries on this knowledge base. It enables the requirement engineer to search the knowledge base and visualize the results in different ways.	The evaluation of the tool will be conducted based on the in-vitro experiments. Furthermore, the integration of the solutions to the model presents the entire solution to the problem.
[214]	Lopez et al. suggested a novel model helping in requirements elicitation for the development of mobile e-commerce applications (RPM-REFEMAD). It enables the elicitation of requirements based on the use of four elicitation techniques namely questionnaire, document analysis, prototyping, and brainstorming which are integrated into 10 steps. The suggested methodology was used to extract the requirements for two mobile e-commerce applications in various scenarios. A survey was created and distributed to stakeholders who took part in the evaluation of the reference process model to visualise the model's findings. The survey results showed that, in comparison to using conventional requirements elicitation methodologies, the RPM-REFEMAD model helps to elicit more high-quality NFRs.	Future work will focus on developing mobile applications for several categories of e-commerce, including B2B, C2B, and C2C.

[144]	Portugal et al. proposed a tool namely NFRFinder, a semi-automated approach for mining keywords. The tool is used to find NFRs from the unstructured text which is collected during the elicitation including meeting minutes. The tool aims to assist the requirement engineer in identifying NFRs from a huge volume of text.	The tool NFRFinder will be explored on various unstructured texts including open-source software and comments.
[215]	Handa et al. proposed a framework for the NFRs prediction system which consists of 4 layers. Layer 1 deals with NFRs elicitation using a questionnaire, layer 2 deals with the feature selection, layer 3 deals with the prediction of NFRs, and layer 4 deals with the visualization of the layer. The present research is focused on the usability perspective of the NFRs.	A recommendation system will be proposed which helps the software developers in NFRs elicitation during the early phases of software development. The other factors including security, maintainability, compatibility, and interoperability will be considered in the future.

2.1.7.1 NFRs Elicitation oriented Research Gaps

The various research gaps are identified in the NFRs elicitation domain. There is still an open question on how to elicit the NFRs as there is no standard tool for eliciting NFRs. Moreover, NFRs are subjective which made it more complex to elicit NFRs. NFRs need to address the significant issue of the quality of software systems. There are several NFRs elicitation techniques, but each has its own limitation. So, the choice of elicitation technique is an open-ended question in NFRs elicitation. As far as the users are concerned, with varying levels of competence, knowledge, and experience; they might range from novices to developers. It can be challenging to define usability requirements because neither consumers nor developers are reliable sources for this information. Developers lack sufficient knowledge to satisfy and meet these needs when dealing with usability characteristics in isolation. The majority of usability research focuses on creating better user interfaces, but a systematic method is still required to assess and model usability requirements in the early phases of software development. The requirement engineer has elicited the NFRs using various tools and techniques from the stakeholders, but still, there is a need to deal with the challenge of missing NFRs. These are the various challenges that were found during the NFRs elicitation. These gaps are outcries for some tool that helps in NFRs elicitation.

2.1.8 Research Gaps

When investigated closely, the current methods demonstrate that the vast majority of the current strategies experience the ill effects of different issues. These lapses can be categorized as given:

1. Manual NFR elicitation is difficult, tedious, and time-consuming, demanding for the application of machine learning approaches. To effectively elicit and classify the NFRs, limited research is conducted. FRs and NFRs are elicited in very diverse ways. To reveal the quality characteristics, limitations, and interface requirements for developing the technical architecture of the program, the elicitation and classification of NFRs are essential [18]. It is advantageous if the classification procedure is automated. It decreases human work, time, and mental exhaustion associated with finding and classifying needs [60].

2. The current study concentrated on the efficiency of reliability and stability. Instead of evaluating the ML model on precision, recall, and, accuracy. It is very important to consider stability and reliability. These characteristics were only considered in a few research studies. The ability of a system to carry out and maintain its operations under expected or unanticipated

conditions is the definition of reliability in general [78]. It is crucial to comprehend how reliable the prediction system is. Reliability is a crucial factor to take into account when evaluating ML models [79]. The degree of consistency between several algorithmic predictions is represented by stability. The effectiveness of the prediction system can be evaluated using this crucial metric [80]. User confidence in the prediction system is influenced by the prediction system's stability. A prediction system is stable if the predictions do not change strongly over a short range of data [81]. Henceforth, it is very significant to consider the reliability and stability metric in the ML model.

3. The performance of the classifiers is impacted by the meaningless characteristics, which dramatically increase the number of false positives [56]. Additionally, limited research has taken into account hybrid feature selection strategies [57]. A hybrid feature selection strategy has been shown in the literature to enhance the functionality of ML models [58]. Implementing the hybrid feature selection for NFRs is therefore essential [59].

4. The usability perspective is not justified by the limited research scope, which focuses more on security and other functional concerns [65]. Usability is crucial to consider because it is related to how people interact with the software. Other elements including security, portability, maintainability, and compatibility have been the topic of numerous research studies [64]. Usability is one of the components of ISO/IEC 25010 [66]. The usability element has been taken into account in the limited research, which is extremely important [67].

5. A consistent dataset for NFR elicitation is not readily available. As per our research, limited NFRs-based secondary datasets are available [48]. The scarcity of datasets is the biggest obstacle confronting NFRs-based research [50]. The publicly available dataset can advance research in the field of NFRs by enabling experiments and serving as a standard for future studies.

6. As per our research, no multilayered framework is used for the prediction of NFRs. Researchers will use the multilayer framework as a reusable template for NFRs prediction. The conceptual framework will help in carrying out the prediction. The framework covers all of the different stages of NFRs prediction. The layered structure will specifically divide the entire prediction process into various tiers. It will serve as a benchmark for other research projects.

7. NFRs are very crucial for the development of the software system. Numerous examples of project failure are mentioned in literature because of not appropriately considering the NFRs [216]. The beforehand information on significant NFRs is very important for software development. It ensures the effectiveness, performance, and usability of the software system. Automatic prediction of NFRs will help the software developers to consider the significant NFRs before starting the actual development. There is a high need for ML techniques for NFRs prediction [90].

The research shows the above gaps that have given significant motivation to design a framework for an intelligent NFRs prediction system. Research Objective 1 is formulated with Research Gaps 4 and 5. According to our research, minimal secondary datasets are available for NFRs. Henceforth, there is a need to develop a mature catalog that consists of a maximum number of NFRs. Research Objective 2 is mapped with Research Gaps 1 and 3. In Research Objective 2, a hybrid feature selection technique for NFRs has been designed based on the classification to classify the data into significant and non-significant groups. Research Objective 3 is formulated with Research Gaps 6 and 7. Objective 3 is based on designing the ML model for predicting the NFRs using various algorithms and approaches like baseline and ensembling. Objective 4 is formulated with research gap 2. In objective 4 the various evaluation parameters are considered for evaluating the performance of the model Accuracy, Recall, and Precision. Moreover, other parameters are also considered like Root Mean Square Error, Stability, Reliability, and Coefficient of Determination.

Chapter 3

Proposed Work

The conceptual view of the proposed work for ML is based on the NFRs Prediction framework. Precisely, the present model consists of 4 crucial layers, namely, as Data Acquisition layer (DA), Feature Selection and Extraction layer (FSE), Data Extraction and Mining layer (DEM), and Data Analysis & Visualization layer (DAV). In the initial DA layer, the data is collected by using varied methods. The FSE layer selects and extracts the significant features from the set of all features. In the DP layer, the different ML algorithms are applied to selected data, and the output of the ML algorithms is ensembled as the output of all ML algorithms. Finally, in the DAV layer, the output is analyzed and visualized. In the proposed model, various NFRs elicitation techniques are discussed, such as Questionnaires, Brainstorming, Templates, and Scenario. The elicitation techniques have their props and corns. One elicitation technique for one application and the other for the second application. The elicitation techniques can be used as a single technique as well as a combined approach. For instance, the output of one technique becomes the input of the next technique. A questionnaire is suitable when there is a need to gather requirements across the globe. In contrast, the interview is successful when there is a direct interaction between the analyst and the stakeholders. Brainstorming is a good one when users have less experience. The features are classified into two groups significant and non-significant, based on the Bayesian

Belief algorithm. Based on these significant features, the ML-based model can be constructed. The algorithm for NFRINDEX has also been presented in the proposed model. Finally, the outcomes of the prediction model have been visualized with the help of a color-coded scheme. This proposed framework is beneficial for NFRs elicitation and prediction. The design and detailed functionality of each layer has been provided ahead in Figure 3.1.

3.1 Data Acquisition (DA)Layer

The DA layer is the initial layer in the proposed framework of ML-based NFRs Prediction. The data can be acquired using different data elicitation techniques like Interviews, Questionnaires, Brainstorming, Use Cases Stories, Modeling Notations, and Evolutionary Development which are explained below:

1. *Interview:* It is a simple and productive way to exchange thoughts and communicate the needs of analysts and stakeholders, either conversationally or verbally. It is often used to elicit requirements with one or two people, face-to-face talk, asking questions, and logging the answers as per the demand of the conduct [217]. It is known as an important technique because of the potential to acquire detailed information. The analyst can also read the body language of the stakeholders, yet it is not easy to give the same time to all stakeholders. Only a small number of stakeholders can participate, so it is a time-consuming process. There are 3 kinds of interviews such as semi-structured, structured, and unstructured. The semi-structured and structured focus on collecting quantitative data, whereas the unstructured interview deals with acquiring the requirements through open debates between stakeholders [218].

   (a) *Structured/Close Interview*:

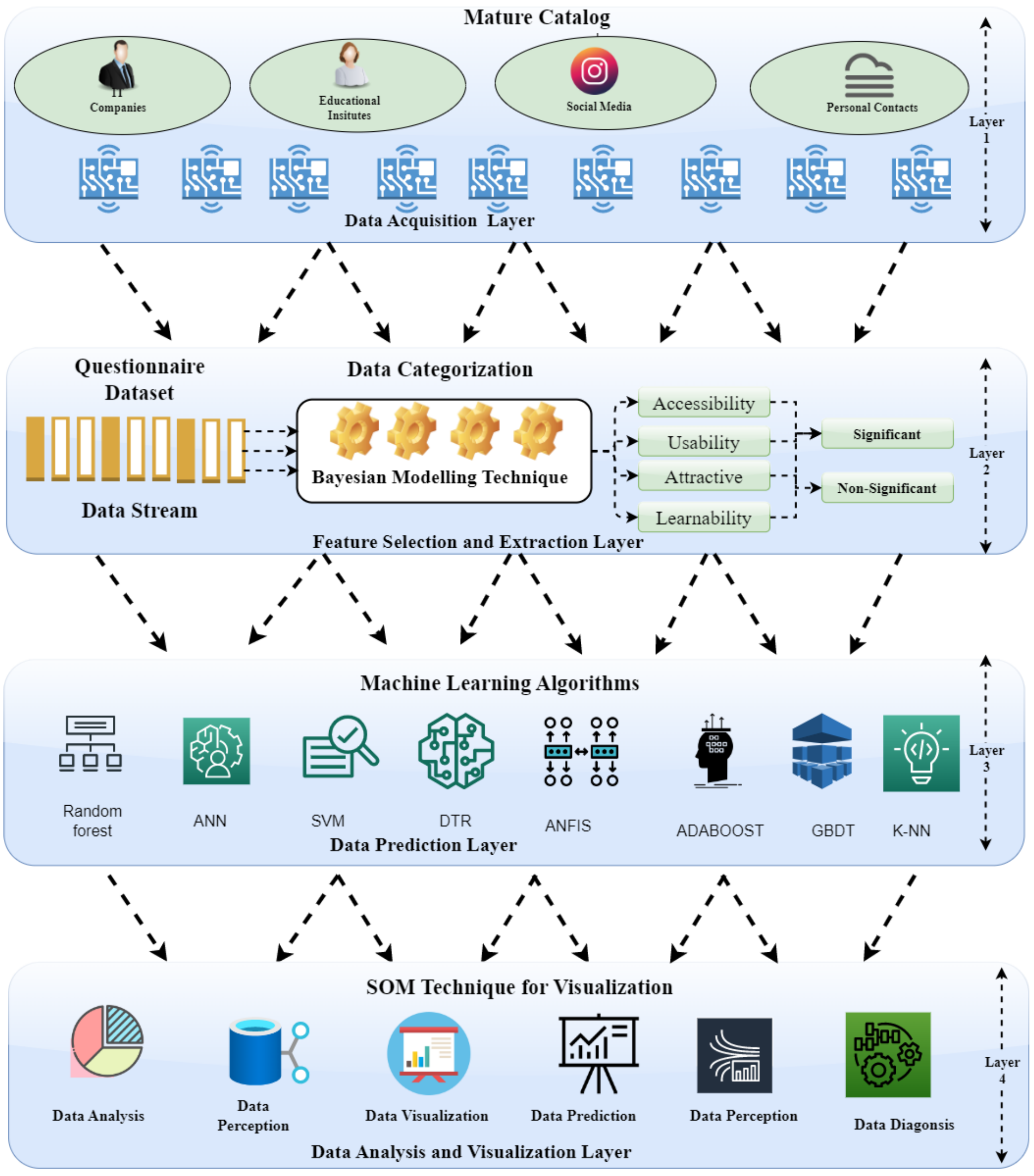


FIGURE 3.1: Proposed model for ML based NFRs Prediction

It is a quite formal approach. Structured interviews are systematic, with a pre-determined set of questions established and posed to the stakeholders. It is a well-known and successful strategy and results in quantitative data. Structured interviews are ineffective as these do not provide for new developments and expressions. Questions are posed in a prescribed manner in an already-established context. The interviewer must use caution with stakeholders, and they should be able to communicate their expertise effectively by listening to responses to the queries. In this approach, there are fewer chances of biasness as questions are predefined. Questions can be added later on to enhance the clarification. The new concepts and innovations are not addressed in this approach.

(b) *Unstructured/Open Interview:* It is an informal interview with questions that are not expected. It is an open discussion of qualitative data production between analysts and stakeholders. Some topics are overlooked in unstructured interviews, while some are detailed. Unstructured interviews are helpful if a particular community understands a specific problem in-depth. The new ideas are generated and addressed. It is a quite comfortable approach for stakeholders.

(c) *Semi-structured Interview:* It combines predefined and unforeseen issues. It is a combination of structured and unstructured interviews. This approach helps stakeholders to share innovative ideas and new experiences. The questions are not predefined in this approach, and analysts may skip the critical questions. It is also very essential to generalize the finding of the interview [219].

2. *Questionnaires:*

It is a low-cost approach to eliciting requirements. Questionnaires reach a large number of individuals in less time and at a lower cost. This approach is used to meet the demands of a broader group population community scattered over a vast region and multiple time zones. This is a successful and fast technique. It's important to pre-plan the questionnaire, and it takes due care of the information. With the help of questionnaires, the requirements can be elicited from a large number of stakeholders. It is pretty economical and there are lesser chances of biasness. This is best suited for generic kinds of software development and helps in avoiding unnecessary and repetitive information. However, the results of the questionnaires should be examined. The questionnaire's outcomes are primarily determined by two things [220]. The effectiveness and design of the questionnaire and the respondent's sincerity. A well-designed and effective questionnaire can elicit information about the user's requirements, objectives, and restrictions. A well-structured questionnaire encourages people to be candid, resulting in reliable results from many people. The data collected via questionnaires enable systematic and quantitative analysis of the generated results. The questionnaire design process is multi-staged and should be considered as such. But in this approach, the questions can be misinterpreted and are ambiguous and meaningful feedback is not acquired [221].

3. *Observation:*

   One of the prominent ethnographic techniques, observation, refers to the actual execution of the ongoing processes by the users. It works along with other techniques like an interview and task analysis. These are a bit expensive to perform and require expertise to understand and interpret. This technique varies in performance as per the skills of the user. Active or passive observation can be achieved. In Active observation, the analyst can interrupt and ask questions to the stakeholders, while In passive observation, there is no interaction between the analyst and stakeholders. It is quite a time-consuming process [66].

4. *Prototyping:*

This is an iterative method in which a toy version of the product is introduced to meet consumer needs and refined according to user feedback again and again. Prototyping can be used for elicitation when users are unaware of their needs and stakeholders need an early response. This approach can be used for other methods, such as JAD or interviews. In the development of new systems, it is functional [222]. The system's prototype gives the user an idea of the system's needs by offering him the fundamental interface design. Prototypes are designed chiefly for systems with more significant interaction between users and implement fewer internal features. This technique is used when an analyst looks for better requirements. Analysts and users have a better understanding of software systems. It is a time-consuming and expensive approach [218].

5. *Domain Knowledge:*

   Examining current and related documentation and applications is an excellent technique for collecting early requirements, understanding and capturing domain knowledge, and identifying reusable concepts and components. When a project involves replacing or modifying an existing system, these types of analyses are especially significant. The Instruction manuals and design documents for existing legacy systems and forms and files in hardcopy are used in current business processes and are an illustration of documentation that may be valuable for requirements elicitation. Upstream and downstream systems and competitors or similar solutions are frequently examined during application studies. Other elicitation approaches, such as observing the current system and interviewing current users, are usually used in these investigations. In the elicitation of requirements, domain knowledge is the kind of detailed description. This approach is generally used in conjunction with other requirements elicitation methods [198].

6. *Introspection:*

The analyst must establish requirements using the introspective technique [23] based on stakeholder's beliefs and requirements from the system. Although most analysts use it to some extent, this method is used primarily as an initial point for additional elicitation activities. The introspection is truly effective only when the analyst is well-versed in the system's domain and aims and knowledgeable about the user's business processes. When the analyst is obliged to apply this technique more frequently, such as when the stakeholders have lesser experience with the system in their workplace, facilitation introspection should be conducted using additional elicitation techniques like interviews and protocol analysis [197].

7. *Repertory Grids:*

   Stakeholders are asked to create attributes and provide values to domain entities in repertory grids [38]. As a result, the system is described as a matrix by categorising the system's constituents, identifying the instances of those categories, and assigning variables with associated values to each one. The purpose is to identify and visualize the similarities and contrasts between the various domain constituents. This degree of abstraction is unknown to the majority of stakeholders. As a result, this method is often used when eliciting requirements from domain experts. The ability of repertory grids to represent precise features of complex requirements is restricted [223].

8. *Card Sorting:*

   It is a method of generating information by which stakeholders are asked to sort cards using index cards or software packages according to the name of a domain entity. It allows user needs to be grouped and combined. In the groups in which they make sense, the participants organize cards. Sorting cards offer a detailed understanding of the user's mental model and explain how users frequently use their minds to tacitly group and sort and mark assignments and content. The stakeholders from far-flung places can participate and provide input. This is only achievable if both the analyst and the participants have a good understanding of the area. It does not provide access to content and is also not suitable for complex and large projects [224].

9. *Group Work:*

   Group work, such as collaborative sessions, is a frequent and often default strategy for eliciting requirements. Stakeholders are encouraged to participate in this methodology to satisfy project requirements. They express their preferences and needs. There is a discussion of each requirement and correct suggestions. The moderator focuses on the classes and encourages members to participate freely. Interaction in a community promotes new ideas and addresses each topic in detail. The expressed thoughts and views are simultaneously registered, enabling the participants to participate fully in the meeting [225]. The session manager ensures that particular personalities do not dominate the conversations. The composition of members and group cohesiveness are essential elements in the effectiveness of group work. Group work is successful if and only if stakeholders feel comfortable and secure in speaking openly and honestly. It is critical and complicated for all stakeholders to come together in one position simultaneously. Due to the number of diverse stakeholders engaged in the project, these meetings can be challenging to schedule. This approach provides us with reasonable quality requirements in less time [226].

10. *Brainstorming:*

    Brainstorming is a method in which members of various stakeholder groups meet informally to create as many ideas as possible without focusing on anyone in particular. The produced pictures have been registered, improper ideas are removed, and sufficient priority has been granted to ideas. It supports new ideas and expressions and promotes advanced and recent practices on current issues. When undertaking this sort of group work, it is critical to avoid delving too deeply into or analyzing ideas. The goal of brainstorming sessions isn't generally to solve serious problems or make important choices. This method is frequently used to create the project's and target system's early mission statement. One of the benefits of brainstorming is that it encourages freethinking and expression while also developing new and creative solutions to challenges [227].

11. *Requirements Workshops:*

    It refers to the structured meeting, with selected stakeholders meeting after multiple sessions to review, refine and verify specifications. To make it successful, the stakeholders need to take an active role. Moreover, facilitator expertise and stakeholder awareness are essential to succeed. It can be used with interviews, document analysis, and brainstorming for illustration. In this approach, there is quick feedback and acquired requirements in a short time. It is beneficial in generating an understanding between the analyst and stakeholders. It isn't easy to schedule the same time meeting for all stakeholders and experts [228].

12. *Document Analysis:*

    This method includes reviewing and collecting information and other relevant data from current records. This technique can be used effectively to begin the process of eliciting requirements. Human interaction is sometimes required to add or validate the data. The information collected in this technique can differ according to the available documentation and human interaction. It is used when an expert wants to research the domain data extensively. It involves analyzing documentation in the context of existing system design documents, templates, and manuals. This methodology is modified to substitute or upgrade existing systems. The key argument here is that the facts and figures analyzed must be examined to determine accuracy and meaning. It is preferred when stakeholders are not available. It is a pretty inexpensive technique and very useful for designing the questions for the interview. An analyst can use historical data to understand the system. But sometimes it provides invalid and incomplete information [229].

There are many more elicitation techniques discussed above. The various research papers have used different elicitation techniques. Moreover, It is vital to select the best approach for eliciting requirements. In the present scenario, the domain knowledge and questionnaire has used to elicit the dataset.

In the current scenario, domain knowledge, and questionnaire has considered for the research work. The iterative questionnaire has been designed for data collection from IT professionals and academicians. The various questions have been collected from the literature review and the initial questionnaire has been designed. The initial questionnaire has been distributed among the software developers and improved iteratively and the final questionnaire has been designed and distributed. The final questionnaire consists of various significant questions related to NFRs. The detail of the questions and responses are given in Table 3.1.

TABLE 3.1: Input Feature Set(NFRs based questions taken from literature review and ✗represents the questions suggested by domain expert)

Question id.	**Ref.**	**Non Functional Requirements based Questions**	**Description**	**Assigned Responses**
A1	[49]	"Are Non Functional Requirements important?"	The question is very important as the developers are asked about the significance of NFRs for IT projects.	The different options as responses are given include extremely significant, significant, less significant, and not significant.

A2	[230]	“Should the website be compatible across different devices?”	Considering usability, The compatibility of the software across various devices including tablet, mobile, and computer systems is very important.	The responses include highly relevant, relevant, Less relevant, and does not matter.
A3	[231]	“Does website work on different web browsers?”	It is important that the website opens and works effectively on different web browsers. Some technologies cause the system to crash while using visiting the website.	Highly significant, significant, less significant, and not significant at all are the various options for response.
A4	[232]	“How significant the customer integration for the website?”	It refers to built customer software solutions that are specific to a particular use case.	Highly significant, significant, less significant, and not significant at all are the various options for response.
A5	[233]	“How important is the responsiveness of the Website?”	The responsiveness refers to an approach that website development and design must act appropriately to user’s behavior based on orientation, screen size, and platform. It includes adjusting screen resolution, flexible images, and custom layout structure.	Highly relevant, relevant, Less relevant, and does not matter are the options given as responses.
A6	[234]	“How significant is page loading time?”	The time taken to load a page after a click is very important to be considered for the software. The stakeholders are asked about the convenient time to be taken by the software in loading a page.	The various options are given consist of time less than 5 seconds, time between 5sec -8 sec, time between 9sec-12 sec, the time between 13 sec -16 sec, and time between 17 sec-20 sec.
A7	✗	“How important are Cookies or Cashing in website design?”	Cashing is used to store online page resources in a browser for the long run purpose. Whereas cookies are employed to store user choices including user preferences.	The options of the responses consist of utmost important, important, and not so important.
A8	[235]	“How significant is the Water Marking of text boxes?”	Watermaking is about imposing a text or logo in text boxes. It defines the textbox answers.	The responses include very significant, significant, less significant, and not significant at all.

A9	[236]	"How important is the Page region (navigation) of the website?"	Page navigation deals with the flow from one page to another without any inconvenience. It serves as the outline of the website.	The responses include very significant, significant, less significant, and not significant at all.
A10	[237]	"How significant is the Font selection of website?"	There are various types of fonts and different sizes of fonts that are used in websites.	The responses are given as Yes, font should be big and simple, Yes, should be artistic, and font selection does not matter.
A11	[238]	"Is Website Accessibility important?"	Website accessibility ensures that potential users including those who are disabled have a positive browsing experience and can quickly access your content.	The response options including important, less important, and not important are considered.
A12	[239]	"Does a Link change its color on use ?"	Different link colors like blue, purple, indigo, and red may be used by developers for used or unused links.	The responses are collected in the form of yes, no, and maybe.
A13	[240]	"Should there be hyperlink descriptions?"	The hyperlink description provides quality information to the user about the hyperlink.	The responses include yes, no, and maybe.
A14	[241]	"Should design be dynamic?"	A dynamic website changes the kind of content that is displayed each time a person accesses it. This display varies depending on the viewer's demographics, the time of day, location, language preferences, and other elements.	There are two choices for the software design consist of dynamic and static.
A15	[242]	"Are user manuals required?"	A user manual guides the user in using the website.	The responses are yes, no, and maybe.
A16	[243]	"What is the total number of people who use the system concurrently?"	It is very important to consider the total number of users who use the system at the same time to enquire about the effectiveness of the system to handle and provide services to its users simultaneously.	The responses are given as >500, > 1000, > 2000, > 5000, and > 10000.

A17	[233]	"What is the acceptable response time after clicking any button?"	Response time helps in website monitoring.	The responses are taken as <1 second, <5 seconds, <10 seconds and <15 seconds.
A18	[244]	"What should the system support whether command line or graphical user interface?"	User interface tells about user friendliness and suggests how a user interacts with a website.	The responses are taken as Command use interface, Graphical user interface, and both.
A19	✗	"Is a detail description of the data model required?"	The detailed description includes information about tables and rows.	The responses are taken as Yes, No, and Maybe.
A20	[245]	"How important is it to have an attractive interface of the website?"	An interactive interface is required for the quality of user interaction.	The responses are taken as extremely significant, significant, less significant, and not significant.
A21	[246]	"How important is it to you that a website should not contain irritating elements?"	The irritating elements include marquees, scrolling text, and constant running animations.	The responses include extremely significant, significant, less significant, and not significant.
A22	[245]	"How significant is the sitemap for you in a website?"	A sitemap is a list of indexable pages on the website.	The responses include extremely significant, significant, less significant, and not significant.
A23	[247]	"How important is error handling to you?"	Error management makes it possible to gracefully handle both hardware and software problems and enables interrupted execution to continue.	The responses include extremely significant, significant, less significant, and not significant.
A24	✗	"How significant are the search features of the website for you?"	The easy searching of data has an impact on the usability of a website.	The responses include extremely significant, significant, less significant, and not significant.
A25	[248]	"Who is going to use the website developed by you?"	It may include occasional or regular users.	The responses include experienced users, IT professionals, and naive users.
A26	✗	"Does your website provide assistance?"	It refers to providing help and assistance like web chat to the user while using the website.	The responses include yes, and no.

A27	[249]	"Is updation of the information important for the website?"	Updation refers to the updating and maintenance of the content or data of the website.	The responses include extremely significant, significant, less significant, and not significant.
A28	[250]	"How significant is the placement of menus and links on the website?"	Menus and links are generally placed by the standard for easy recognition.	The responses include extremely significant, significant, less significant, and not significant.
A29	[246]	"How significantly navigational aids act in efficient information retrieval in your website?"	Navigational aids provide more direct assistance with site navigation.	The responses include extremely significant, significant, less significant, and not significant.
A30	[251]	"How significant your website considers users memorability?"	Memorability measures how well users can remember different functions after they have learned the functions.	The responses include extremely significant, significant, less significant, and not significant.
A31	✗	"How important is the website integration for you in the website?"	Website integration refers to integrating the data from various resources including system, application, and website.	The responses include extremely significant, significant, less significant, and not significant.
A32	[245]	"How important is users satisfaction with the website to you?"	Satisfaction refers to the level of pleasure and confidence while using the website.	The responses include extremely significant, significant, less significant, and not significant.
A33	✗	"Do you think this questionnaire covers usability requirements?"	The respondents are asked about their views on the questionnaire.	The responses including less than 50%, more than 60%, and more than 80%.

The questionnaire is distributed among IT professionals and academicians for data collection. The stakeholder's response sincerity is a key to successful elicitation. After collecting the dataset it is very important to preprocess the dataset.

3.1.1 Data Preprocessing and Normalization

Data Preprocessing is a very significant step in every ML model. It is a preliminary step, and the data get transformed and bring the data into a state that can be efficiently understood and interpreted by the ML algorithms. A dataset can be considered a group of data objects such as vectors, samples, observations, points, entities, and records. The features can be called attributes, fields, characteristics, dimensions, and variables. A data object is a collection of features. Features can be any measurable characteristics or property of a thing that is being observed. For instance, dimension, color, and shape can be taken as features of a table. These features can be broadly categorized into Categorical and Numerical and further sub-classified as Nominal, Ordinal, Interval, and Ratio as shown below in Figure 3.2.

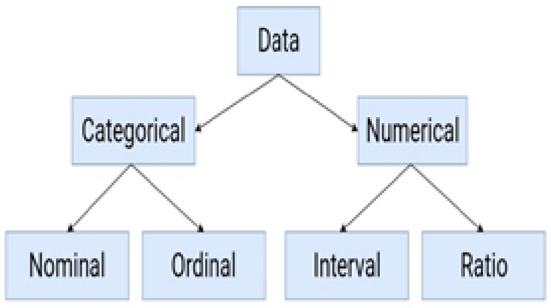


FIGURE 3.2: Categories of data

(a) Categorical variables represent grouping. The categorical features have taken their values from the fixed set of values. For instance, Boolean set: False, True is a categorical variable because the values need to be taken from the set. Another example is the months in a year. The categorical variables are sub-classified as Nominal and Ordinal.

- Nominal features are those variables that are without any implied order. The order of the values does not have any effect. For instance, The color of the bike such as Red, Blue, and Green.
- Ordinal features are those variables that have some natural implied order. For instance, The size of the dress such as Small, Large, or Extra Large.

(b) The numerical features are those features whose values are integer values. Mostly these are characterized by numbers and having the properties of numbers. The speed of the car or the walking steps are examples of numerical features.

- Interval: It represents the variables whose main feature is that they can be measured along with the continuous and have a numeric value. This method can be used to encode Fahrenheit and Celsius scales, which differ in the size of a unitś Zero values.
- Ratio: These variables can be scaled to any specific measure while keeping the meaning and ratio of their values. For illustration, length can be expressed in meters or feet, and money can be expressed in a variety of currencies.

In the present scenario, the data preprocessing and normalisation consist of scaling, removing missing values, data augmentation, and balancing the dataset. Normalization is one of the most frequently used data preparation techniques, which helps us to change the values of numeric columns in the dataset to use a common scale.

The collected responses are required to convert into numeric values. While attempting to combine the values as features during modeling, the significant difference in the scale of the numbers causes problems. Normalization solves these issues by generating new values that preserve the source data's general distribution and ratios while keeping values within a scale that is applied to all numeric columns in the model. This module provides a variety of numeric data transformation options. The values are converted to a 0-1 scale such as percentile rank. For illustration, The response consists of extremely significant, highly significant, significant, less significant, and not significant at all and is converted into numeric values such as 1, .75, .50, .25, and 0. The Extremely Relevant, Relevant, Less Relevant, Not Relevant, and Neutral are scaled as 1,.75,.50,.25, and 0. The Strongly Agree, Agree, Fair, Disagree, and Strongly Disagree are converted into 1, .75, .50, .25, and 0. The MinMaxScaler normalization has used and scaled the data into a range of 0 and 1. It is very important to scale up all features on a common scale before applying ML algorithms.

The missing values are replaced with the mode value of the attribute. The mode value represents the highest frequency value i.e. the response which has been chosen by the maximum number of respondents. The mode value is highly suitable for filling the null values in the present scenario. Data augmentation is also a part of data preprocessing.

3.1.1.1 Data Augmentation

ML models are robust and have high accuracy, the problem faced by them is data scarcity or data imbalance. Imbalanced data are a phenomenon often occurring in a real-world application where the distribution of classes is not balanced, i.e., some classes appear much more frequently than others which are called biasness in data. In such situations, ML algorithms have difficulties since they will be biased toward the majority class. The reason is that ML algorithms assume the number of measurements for each class to be approximately the same [252]. Moreover, the small dataset is again a challenge for ML models. Data augmentation technique is key to the problem. The artificial production of training data for ML through transformations is known as data augmentation. There are many data augmentation techniques are available. The most promising data augmentation technique for tabular data is Synthetic Minority Oversampling Technique (SMOTE) [253] [254] [255] [256]. Chawla et al. [257] introduced SMOTE technique which is one of the most popular oversampling techniques. SMOTE augments the data by creating synthetic data points based on the original data points. It can be seen as an advanced version of oversampling, or as a specific algorithm for data augmentation. The advantage of SMOTE is that you are not generating duplicates, but rather creating synthetic data points that are slightly different from the original data points. It is not just oversampling technique but also handles the class imbalance problem. SMOTE technique augments the data as well as balance the distribution of class by replicating and randomly increasing the minority class instances. It potentially performs better than simple oversampling and is widely used [258]. Many research experiments show that augmented data-trained classifiers perform better than original small datasets [259]. In the current research work, SMOTE technique has been used for data oversampling and balancing the classes as well. There are many variants of SMOTE technique including Borderline-SMOTE(BDSMOTE)[260] and Adaptive synthetic sampling approach for imbalanced learning (ADASYN).

BDSMOTE focuses on generating synthetic data by considering only samples that make up the border that divides one class from another. BDSMOTE is an oversampling technique derived from the technique called SMOTE. SMOTE generates synthetic data of a minority class by using the nearest neighbor of minority class data samples. But SMOTE does not consider the location of adjacent majority class data while synthesizing the data of the minority class, so the class samples can be overlapped. To address this limitation, Han et al. proposed an improved SMOTE algorithm, BorderlineSMOTE [260]. The B6orderline-SMOTE algorithm performs oversampling on instances of the minority class near the borderline. He et. al. introduced Adaptive Synthetic (ADASYN) [261]. It is a technique that is based on the SMOTE algorithm for generating synthetic data. The difference between ADASYN and SMOTE is that ADASYN implements a methodology that detects those samples of the minority class found in spaces dominated by the majority class, to generate samples in the lower-density areas of the minority class. It focuses on those samples of the minority class that are difficult to classify because they are in a low-density area. The present research

Table 3.2: Data Augmentation Results based on Accuracy

Techniques	Methods	Accuracy	Specificity	Sensitivity
SMOTE	SVM	86.81	86.65	86.43
	KNN	87.10	86.82	87.09
	BBM	88.32	87.59	87.49
BDSMOTE	SVM	86.23	85.65	85.56
	K-NN	85.10	86.82	86.10
	BBM	87.52	86.45	86.23
ADASYN	SVM	85.89	60.65	85.61
	K-NN	86.10	85.82	85.10
	BBM	87.52	87.19	86.39

explores the SVM, KNN, and BBM for the data augmentation techniques including SMOTE, BDSMOTE, and ADASYN for the classification purpose. The classification model classifies the data into significant and non significant NFRs. The comparison analysis is registered in Table 3.2.

Table 3.2 presents that the Basic SMOTE technique outperforms in comparison to other data augmentation techniques BDSMOTE and ADASYN. It is also observed that SMOTE technique performs better with BBM in terms of Accuracy, Specificity, and Sensitivity. Henceforth SMOTE technique is considered the data augmentation technique for the present research work. After augmentation, the dataset is fully available for experiment purposes. The output of this layer is the dataset which is the input to the next layer.

3.2 Feature Selection and Extraction(FSE) Layer

FSE is the second layer of the proposed model intended to select the significant features from the set of features for NFRs. The proposed research incorporates different Feature Selection techniques. The current research focuses on the parameters of ISO/IEC 25010 for eliciting the NFRs. The questionnaire is designed with the perspective of ISO/IEC 25010 parameters, such as compatibility of the system, response time, font size, and navigation. Each factor of ISO/IEC 25010 is essential to find the overall quality of the software. A questionnaire is divided into

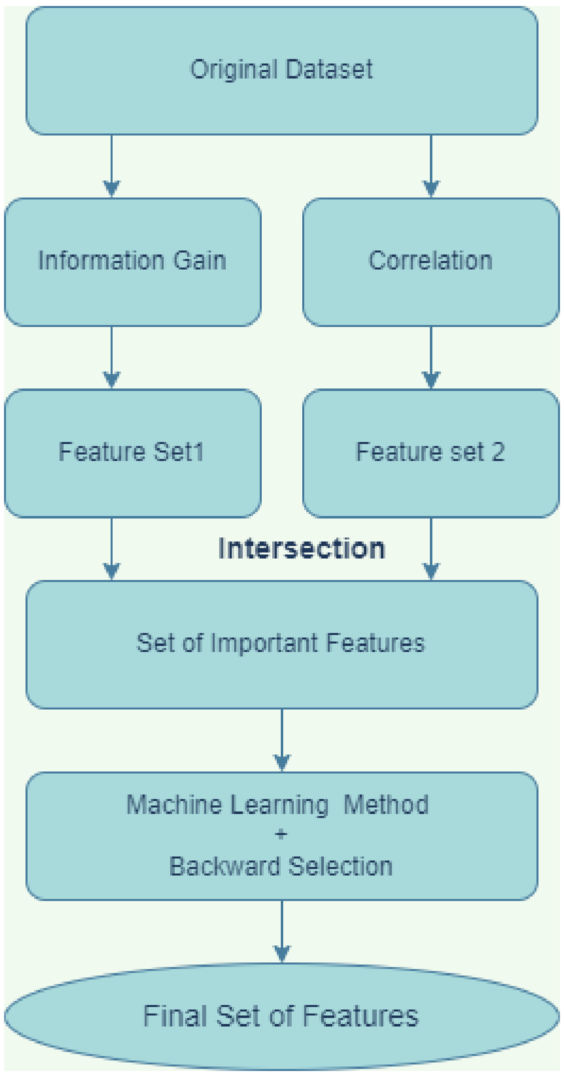


FIGURE 3.3: Proposed model for Hybrid feature selection NFRs Prediction

different sections and subsections. For instance, Usability is divided into 2 sections UX and UI. UX refers to the experience of a stakeholder with the software, and UI depicts how stakeholders interact with the software. In addition to this, each question is assigned a priority on the Likert Type Scale to determine NFRs. For instance, gauge the importance of software compatibility on a tablet, mobile, and computer system. The level of importance has been given as

1. Less significant
2. Slightly significant
3. Neutral

4. Moderately significant
5. Extremely significant

Algorithm 1 Hybrid Feature Selection Procedure

```
1: Input: Feature Set FS=f1,f2,.......,fn.
2: Output: S- the selected feature subset.
3: Select the features based on the information Gain filter method and store them
   in Feature Set-1.
4: Select the features based on the Correlation filter method and store them in
   Feature Set-2.
5: Perform intersection between Feature Set 1 and Feature Set 2 and store in
   Feature Set-3.
6: * Extract common features*
7: Provide Feature Set-3 to wrapper method as input
8: Apply wrapper method with Backward Selection method.
9: Get Set-4 as the final selected feature subset.
```

Aforementioned five-point scale analyses NFRs based on their importance. FSE performs the crucial task of analyzing data effectively based on specific thresholds. The goal of feature selection is to identify the most important features of a given problem. It is beneficial in terms of increasing computing speed and accuracy. The current study presents a hybrid FS technique for NFRs classification that incorporates two FS methods: Filter and Wrapper. Candidate features are initially chosen from the original feature set using two filter approaches, and then the intersection of these two sets is refined further using more precise wrappers. The working of the proposed Algorithm 1 is depicted in Figure 3.3. The proposed hybrid feature selection algorithm starts with two filter techniques namely information gain and correlation to remove unnecessary and redundant features. Both approaches generated features that are regarded as the most class-related features among all features. As a result, deciding on a final feature set for the wrapper technique has become a challenge. The intersection meaning AND of these two feature sets is the key to combining them. Only those features that are common in both of the feature sets are chosen when feature set1 and feature set2 are intersected. The features would be examined by the wrapper procedure using a machine learning technique. For the wrapper technique, there is a need to choose a search strategy and an ML algorithm. The sequential backward selection approach and Bayesian Belief Method is explored for research purpose. In terms of NFRs prediction, the hybrid approach is more viable. FS approaches have been applied to classify problems to pick a smaller feature set that makes the classifier more accurate and faster.

The researchers have used the Bayesian Belief Model to classify data segments. The level of NFR Significance (LoNFRS) is used as a probability measure because of its high efficiency. The Significant class event is analyzed by quantifying the usability factors into a unifying factor. The current research incorporates the level of significance of NFRs assessment as a quantification measure.

Definition 1. *Level of NFR Significance(LoNFRS): To determine the presence of significance in NFRs. LoNFRS are used as a probabilistic measure. It quantifies the existence of irregularities as indicated by stakeholders. These irregularities are significantly related.*
LoNFRS monitors the significance of NFRs by estimating the interdependent parameters. The values of NFRS form two classes of data segments, namely the Significant NFRs Group and Non-Significant NFRs Group.

1. *Significant NFRs Group:* Essential NFRs which have a detrimental impact on software development are grouped in this class. These NFRs mostly belong to the normal range of significance and are utilized to monitor the parametric values. It is crucial to examine this class for the success of software development.

2. *Non-Significant Group:* It contains a set of parameters that falls below the normal range of values and do not greatly affect software development. These

are comparatively less required data fragments during the software development process. For instance, A lower value of compatibility and responsiveness in NFRs.

3.2.0.1 ***Mathematical analysis***

Bayesian Belief Model (BBM): It is a probabilistic model for assessing a class of the particular data segment. As proclaimed earlier 2 classes (groups) are defined which are associated with various NFRs-based parameters. For mathematical analysis, let a data instance is represented by the vector $X_i = (x_1, x_2, x_3, \ldots, x_n)$where X_i signifies the i_{th} parameter of NFRs, given that all parameters are mutually related. The conditional probability of significant X_i belongs to the classY_j is defined by $P(\frac{Y_j}{x_1, x_2, x_3, \ldots, x_n})$. However, because the input parameters are large and variable values can be obtained by unique NFR attributes so that the above formulation can lead to local inconsistencies. Hence, it is possible to describe the revised BBM as

$$P(\frac{Y_j}{X_i}) = \frac{P(Y_j)P(A_i/Y_j)}{P(A_i)} \tag{3.1}$$

However, the probability of $P(Y_j)P(X_i/Y_j)$ can be revised based on the joint probability function as

$P(Y_j)P(X_i/Y_j) = P(x_1, x_2, x_3, \ldots, Y_j)$
$= P(x_1/x_2, \ldots, x_n, Y_j)P(x_2, \ldots, x_n, Y_j)$
$= P(x_1/x_2, \ldots, x_n, Y_j)P(x_2/x_3, \ldots, x_n, Y_j)P(x_3, \ldots, x_n, Y_j)$
$= P(x_1/x_2, \ldots, x_n, Y_j)P(x_2/x_3, \ldots, x_nY_j), \ldots, P(x_n - 1/x_n, \ldots, x_nY_j)$
$P(x_n/Y_j)P(Y_j)$

Furthermore, each x_i attribute of NFR parameter is understood to be independent of other attributes x_j such that $i \neq j$.Then $P(x_i/x_{i+1}, \ldots, x_n, K_j) = P(x_i/y_j)$. Accordingly, the joint probability can be represented as

$$P(Y_j) = \sum_{i=1}^{n} P(Y_j)P(X_i/Y_j) \tag{3.2}$$

$$P(\frac{Y_j}{x}) = \sum_{i=i}^{n} P(Y_J)P(x_i/Y_j)/P(x) \tag{3.3}$$

In the above equation, Y_J can represent the NFRs significant and NFRs non-significant group. Moreover, the classification of a dataset into different groups enables the efficient quantification of a dataset.

The dataset has been provided as input to the FSE layer. All 33 attributes are named A1, A2, A3,...., and A33 respectively. The dataset has provided the two filter methods including information gain and correlation. Information Gain(IG) can be used for feature selection, by evaluating the gain of each variable in the context of the target variable. In other words, the calculation is referred to as mutual information between the two random variables. For illustration, a classification problem defines the amount of information provided by the feature items for the classification problem. It is calculated by how much of a term can be used for the classification of information. In the current scenario, all 33 attributes are evaluated for the information gain with the target variable name as significant and non-significant NFRs. It uses 1 and 0 respectively to measure the importance of various attributes for the classification. IG is calculated by how much an attribute contributes for classification purposes. The IG for all the attributes is calculated and sorted in descending order as shown in Table 3.3 below:

Table 3.3: Information Gain Score for Feature Selection

Attributes	**IG Score**
A1	0.8659
A29	0.8421
A33	0.8413
A14	0.8258
A12	0.7119
A32	0.7064
A20	0.7809
A7	0.7784
A8	0.7614
A28	0.6610
A10	0.6596

A27	0.6162
A4	0.6010
A13	0.5658
A11	0.5489
A17	0.5364
A16	0.5358
A25	0.5289
A18	0.4364
A30	0.4329
A31	0.4269
A21	0.4214
A19	0.4010
A23	0.3845
A3	0.2489
A9	0.0291
A6	0.0261
A22	0.0581
A24	0.0735
A5	0.0486
A2	0.0365
A26	0.0389
A15	0.0264

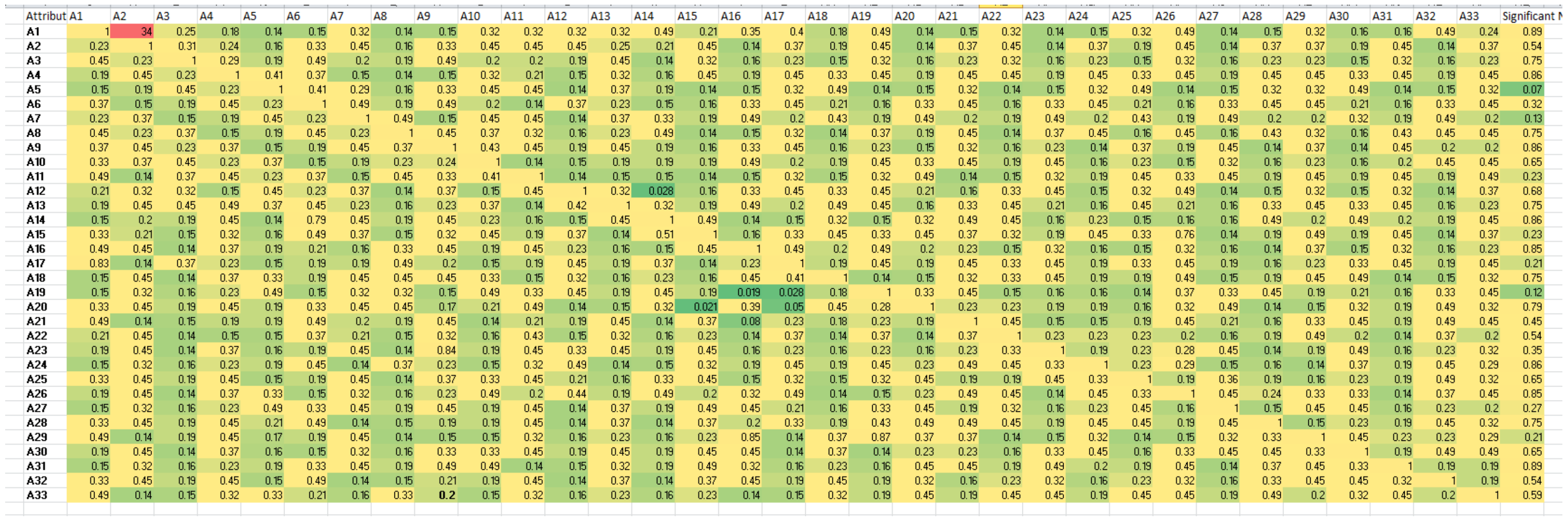
Table 3.3 registers the IG score of attributes with the target variable. It is clear from the table that the attributes A9, A6, A22, A24, A5, A2, A26, and A15 have the least IG score. Thereafter, these attributes are removed from the total attributes and the final list contains only 25 attributes.

Set-1= {A1, A3, A4, A7, A8, A10, A11, A12, A13, A14, A16, A17, A18, A19, A20, A21, A23, A25, A27, A28, A29, A30, A31, A32, A33}.

Correlation is a measure of the linear relationship between 2 or more variables. Through correlation, we can predict one variable from the other. The logic behind using correlation for feature selection is that the good variables are highly correlated with the target. Furthermore, variables should be correlated with the target but should be uncorrelated among themselves. If two variables are correlated, we can predict one from the other. Therefore, if two features are correlated, the model only really needs one of them, as the second one does not add additional information. We need to set an absolute value, for illustration 0.5 as the threshold for selecting the variables in the present scenario. If we find that the predictor variables are correlated among themselves, we can drop the variable which has a lower correlation coefficient value than the target variable. We can also compute multiple correlation coefficients to check whether more than two variables are correlated to each other. In the present scenario, the Pearson correlation has been explored and out of that, we got the heatmap. The heatmap shows the correlation of 33 attributes with target variable significant NFRs and also the correlation between different variables. The heatmap is shown in Figure 3.4: The heatmap shows that there is a high correlation between A14 and A6 i.e. 0.79. Moreover, A6 has less correlation with the target value than A14 i.e. 0.32 and 0.89 respectively. So variable A6 is removed from the total features. Similarly, A29, A15, A5, A17, and A23 are removed from the feature list. As these attributes have a high correlation with the other attributes and less with the target variable. After removing these attributes Set-2 consists of 27 attributes.

Set-2={A1, A2, A3, A4, A7, A8, A9, A10, A11, A12, A13, A14, A16, A18, A19, A20, A21, A22, A25, A27, A28, A30, A31, A32, A33}. The preliminary procedure is completed with the selection of two features subset with IG and correlation. The selection of the final feature set for the wrapper method is a problem now. The key to the problem is to combine these two feature sets by performing the intersection on these feature sets. The resultant features are the features that are common in both sets. After intersecting Set-1 and Set-2 the Set-3 is attained with 22 features.

Set-3={A1, A12, A7, A8,A10, A27, A4, A13, A11, A16, A18, A31, A21,A19, A3, A25, A27,A28, A30, A20, A32, A33 }. The main concept behind the suggested solution is to combine the accuracy of the wrapper method with the efficiency of the filter method. The resultant features in Set3 are provided as input to the wrapper method. The search method and machine learning algorithm are required to be selected for the wrapper method. The wrapper method explores the backward search method and various ML algorithms for examining the features. Backward search



Attribut	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	A25	A26	A27	A28	A29	A30	A31	A32	A33	Significant N
A1	1	34	0.25	0.18	0.14	0.15	0.32	0.14	0.15	0.32	0.32	0.32	0.32	0.49	0.21	0.35	0.4	0.18	0.49	0.14	0.15	0.32	0.14	0.15	0.32	0.49	0.14	0.15	0.32	0.16	0.16	0.49	0.24	0.89
A2	0.23	1	0.31	0.24	0.16	0.33	0.45	0.16	0.33	0.45	0.45	0.45	0.25	0.21	0.45	0.14	0.37	0.19	0.45	0.14	0.37	0.45	0.14	0.37	0.19	0.45	0.14	0.37	0.37	0.19	0.45	0.14	0.37	0.54
A3	0.45	0.23	1	0.29	0.19	0.49	0.2	0.19	0.49	0.2	0.2	0.19	0.45	0.14	0.32	0.16	0.23	0.15	0.32	0.16	0.23	0.32	0.16	0.23	0.15	0.32	0.16	0.23	0.23	0.15	0.32	0.16	0.23	0.75
A4	0.19	0.45	0.23	1	0.41	0.37	0.15	0.14	0.15	0.32	0.21	0.15	0.32	0.16	0.45	0.19	0.45	0.33	0.45	0.19	0.45	0.45	0.19	0.45	0.33	0.45	0.19	0.45	0.45	0.33	0.45	0.19	0.45	0.86
A5	0.15	0.19	0.45	0.23	1	0.41	0.29	0.16	0.33	0.45	0.45	0.14	0.37	0.19	0.14	0.15	0.32	0.49	0.14	0.15	0.32	0.14	0.15	0.32	0.49	0.14	0.15	0.32	0.32	0.49	0.14	0.15	0.32	0.07
A6	0.37	0.15	0.19	0.45	0.23	1	0.49	0.19	0.49	0.2	0.14	0.37	0.23	0.15	0.16	0.33	0.45	0.21	0.16	0.33	0.45	0.16	0.33	0.45	0.21	0.16	0.33	0.45	0.45	0.21	0.16	0.33	0.45	0.32
A7	0.23	0.37	0.15	0.19	0.45	0.23	1	0.49	0.15	0.45	0.45	0.14	0.37	0.33	0.19	0.49	0.2	0.43	0.19	0.49	0.2	0.19	0.49	0.2	0.43	0.19	0.49	0.2	0.2	0.32	0.19	0.49	0.2	0.13
A8	0.45	0.23	0.37	0.15	0.19	0.45	0.23	1	0.45	0.37	0.32	0.16	0.23	0.49	0.14	0.15	0.32	0.14	0.37	0.19	0.45	0.14	0.37	0.45	0.16	0.45	0.16	0.43	0.32	0.16	0.43	0.45	0.45	0.75
A9	0.37	0.45	0.23	0.37	0.15	0.19	0.45	0.37	1	0.43	0.45	0.19	0.45	0.19	0.16	0.33	0.45	0.16	0.23	0.15	0.32	0.16	0.23	0.14	0.37	0.19	0.45	0.14	0.37	0.14	0.45	0.2	0.2	0.86
A10	0.33	0.37	0.45	0.23	0.37	0.15	0.19	0.23	0.24	1	0.14	0.15	0.19	0.19	0.19	0.49	0.2	0.19	0.45	0.33	0.45	0.19	0.45	0.16	0.23	0.15	0.32	0.16	0.23	0.16	0.2	0.45	0.45	0.65
A11	0.49	0.14	0.37	0.45	0.23	0.37	0.15	0.45	0.33	0.41	1	0.14	0.15	0.15	0.14	0.15	0.32	0.15	0.32	0.49	0.14	0.15	0.32	0.19	0.45	0.33	0.45	0.19	0.45	0.19	0.45	0.19	0.49	0.23
A12	0.21	0.32	0.32	0.15	0.45	0.23	0.37	0.14	0.37	0.15	0.45	1	0.32	0.028	0.16	0.33	0.45	0.33	0.45	0.21	0.16	0.33	0.45	0.15	0.32	0.49	0.14	0.15	0.32	0.15	0.32	0.14	0.37	0.68
A13	0.19	0.45	0.45	0.49	0.37	0.45	0.23	0.16	0.23	0.37	0.14	0.42	1	0.32	0.19	0.49	0.2	0.49	0.45	0.16	0.33	0.45	0.21	0.16	0.45	0.21	0.16	0.33	0.45	0.33	0.45	0.16	0.23	0.75
A14	0.15	0.2	0.19	0.45	0.14	0.79	0.45	0.19	0.45	0.23	0.16	0.15	0.45	1	0.49	0.14	0.15	0.32	0.15	0.32	0.49	0.45	0.16	0.23	0.15	0.16	0.16	0.49	0.2	0.49	0.2	0.19	0.45	0.86
A15	0.33	0.21	0.15	0.32	0.16	0.49	0.37	0.15	0.32	0.45	0.19	0.37	0.14	0.51	1	0.16	0.33	0.45	0.33	0.45	0.37	0.32	0.19	0.45	0.33	0.76	0.14	0.19	0.49	0.19	0.45	0.14	0.37	0.23
A16	0.49	0.45	0.14	0.37	0.19	0.21	0.16	0.33	0.45	0.19	0.45	0.23	0.16	0.15	0.45	1	0.49	0.2	0.49	0.2	0.23	0.15	0.32	0.16	0.15	0.32	0.16	0.14	0.37	0.15	0.32	0.16	0.23	0.85
A17	0.83	0.14	0.37	0.23	0.15	0.19	0.19	0.49	0.2	0.15	0.19	0.45	0.19	0.37	0.14	0.23	1	0.19	0.45	0.19	0.45	0.33	0.45	0.19	0.33	0.45	0.19	0.16	0.23	0.33	0.45	0.19	0.45	0.21
A18	0.15	0.45	0.14	0.37	0.33	0.19	0.45	0.45	0.45	0.33	0.15	0.32	0.16	0.23	0.16	0.45	0.41	1	0.14	0.15	0.32	0.33	0.45	0.19	0.19	0.49	0.15	0.19	0.45	0.49	0.14	0.15	0.32	0.75
A19	0.15	0.32	0.16	0.23	0.49	0.15	0.32	0.32	0.15	0.49	0.33	0.45	0.19	0.45	0.19	0.019	0.028	0.18	1	0.33	0.45	0.15	0.16	0.16	0.14	0.37	0.33	0.45	0.19	0.21	0.16	0.33	0.45	0.12
A20	0.33	0.45	0.19	0.45	0.19	0.33	0.45	0.45	0.17	0.21	0.49	0.14	0.15	0.32	0.021	0.39	0.05	0.45	0.28	1	0.23	0.23	0.19	0.19	0.16	0.32	0.49	0.14	0.15	0.32	0.19	0.49	0.32	0.79
A21	0.49	0.14	0.15	0.19	0.19	0.49	0.2	0.19	0.45	0.14	0.21	0.19	0.45	0.14	0.37	0.08	0.23	0.18	0.23	0.19	1	0.45	0.15	0.15	0.19	0.45	0.21	0.16	0.33	0.45	0.19	0.49	0.45	0.45
A22	0.21	0.45	0.14	0.15	0.15	0.37	0.21	0.15	0.32	0.16	0.43	0.15	0.32	0.16	0.23	0.14	0.37	0.14	0.37	0.14	0.37	1	0.23	0.23	0.23	0.2	0.16	0.19	0.49	0.2	0.14	0.37	0.2	0.54
A23	0.19	0.45	0.14	0.37	0.16	0.19	0.45	0.14	0.84	0.19	0.45	0.33	0.45	0.19	0.45	0.16	0.23	0.16	0.23	0.16	0.23	0.33	1	0.19	0.23	0.28	0.45	0.14	0.19	0.49	0.16	0.23	0.32	0.35
A24	0.15	0.32	0.16	0.23	0.19	0.45	0.14	0.37	0.23	0.15	0.32	0.49	0.14	0.15	0.32	0.19	0.45	0.19	0.45	0.23	0.49	0.45	0.33	1	0.23	0.29	0.15	0.16	0.14	0.37	0.19	0.45	0.29	0.86
A25	0.33	0.45	0.19	0.45	0.15	0.19	0.45	0.14	0.37	0.33	0.45	0.21	0.16	0.33	0.45	0.15	0.32	0.15	0.32	0.45	0.19	0.19	0.45	0.33	1	0.19	0.36	0.19	0.16	0.23	0.19	0.49	0.32	0.65
A26	0.19	0.45	0.14	0.37	0.33	0.15	0.32	0.16	0.23	0.49	0.2	0.44	0.19	0.49	0.2	0.32	0.49	0.14	0.15	0.23	0.49	0.45	0.14	0.45	0.33	1	0.45	0.24	0.33	0.33	0.14	0.37	0.45	0.85
A27	0.15	0.32	0.16	0.23	0.49	0.33	0.45	0.19	0.45	0.19	0.45	0.14	0.37	0.19	0.49	0.45	0.21	0.16	0.33	0.45	0.19	0.32	0.16	0.23	0.45	0.16	1	0.15	0.45	0.45	0.16	0.23	0.2	0.27
A28	0.33	0.45	0.19	0.45	0.21	0.49	0.14	0.15	0.19	0.19	0.45	0.14	0.37	0.14	0.37	0.2	0.33	0.19	0.43	0.49	0.49	0.45	0.19	0.45	0.45	0.19	0.45	1	0.15	0.23	0.19	0.45	0.32	0.75
A29	0.49	0.14	0.19	0.45	0.17	0.19	0.45	0.14	0.15	0.15	0.32	0.16	0.23	0.16	0.23	0.85	0.14	0.37	0.87	0.37	0.37	0.14	0.15	0.32	0.14	0.15	0.32	0.33	1	0.45	0.23	0.23	0.29	0.21
A30	0.19	0.45	0.14	0.37	0.16	0.15	0.32	0.16	0.33	0.33	0.45	0.19	0.45	0.19	0.45	0.45	0.14	0.37	0.14	0.23	0.23	0.16	0.33	0.45	0.16	0.33	0.45	0.45	0.33	1	0.19	0.49	0.49	0.65
A31	0.15	0.32	0.16	0.23	0.19	0.33	0.45	0.19	0.49	0.49	0.14	0.15	0.32	0.19	0.49	0.32	0.16	0.23	0.16	0.45	0.45	0.19	0.49	0.2	0.19	0.45	0.14	0.37	0.45	0.33	1	0.19	0.19	0.89
A32	0.33	0.45	0.19	0.45	0.15	0.49	0.14	0.15	0.21	0.19	0.45	0.14	0.37	0.14	0.37	0.45	0.19	0.45	0.19	0.32	0.16	0.23	0.32	0.16	0.23	0.32	0.16	0.33	0.45	0.45	0.32	1	0.19	0.54
A33	0.49	0.14	0.15	0.32	0.33	0.21	0.16	0.33	**0.2**	0.15	0.32	0.16	0.23	0.16	0.23	0.14	0.15	0.32	0.19	0.45	0.19	0.45	0.45	0.19	0.45	0.45	0.19	0.49	0.2	0.32	0.45	0.2	1	0.59

FIGURE 3.4: Heatmap for correlation

is started with all features and it removes one feature at a time. Then the learning model is applied to test its result. The same process is performed iteratively till the number of features reached a predefined threshold or the test result got worse. It follows a greedy search approach by evaluating all the possible combinations of features against the evaluation criterion. The wrapper method explored the ML algorithms including BBM, DT, SVM, and KNN for feature selection, and the results are registered in Table 3.4.

TABLE 3.4: Wrapper technique comparison table based on Accuracy

No. of features	**BBM** (Accuracy)	**DT** (Accuracy)	**SVM** (Accuracy)	**KNN** (Accuracy)
22	95.89	91.54	93.78	94.23
21	95.79	91.45	93.55	94.01
20	95.72	90.89	93.13	93.61
19	95.50	92.94	93.05	93.15
18	95.45	91.23	92.17	91.56
17	95.45	89.19	91.58	90.41
16	90.22	88.25	89.67	90.14
15	90.13	88.19	87.25	89.73
14	88.87	87.12	85.46	86.45
13	85.19	81.45	83.19	84.02
12	83.45	80.63	81.12	82.54
11	81.12	80.16	78.54	80.79
10	79.56	75.38	77.47	76.71
9	73.17	70.73	69.56	71.46
8	70.23	68.78	65.27	69.86
7	68.32	65.47	65.59	67.61
6	67.19	61.97	65.87	66.75
5	65.12	60.24	63.85	63.24
4	64.59	59.76	62.56	63.35
3	60.57	59.84	61.28	59.64
2	58.79	49.52	51.92	50.03
1	55.79	45.76	51.12	52.81

The performance of the wrapper-based FS is measured in terms of Accuracy. BBM, SVM, DT, and K-NN were explored for comparative purposes. Moreover, it is worth noting that the only classification technique was changed during implementation, while the rest of the model remains unchanged. According to the results, the BMM attains the highest accuracy value. Furthermore, the BMM is more efficient than other ML techniques including DT, SVM, and K-NN. The details of the BBM are depicted in Table 3.5.

TABLE 3.5: Wrapper technique results with BBM

No. of features	**Features set**	**Accuracy**
22	{A1, A3, A4, A7, A10, A11, A12, A13, A16, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32, A33 }	95.89
21	{A1, A3, A4, A7, A10, A11, A12, A13, A16, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32 }	95.79
20	{A1, A4, A7, A10, A11, A12, A13, A16, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32 }	95.72
19	{A1, A4, A7, A10, A11, A13, A16, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32 }	95.50
18	{A1, A4, A7, A11, A13, A16, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32 }	95.45
17	{A1, A4, A7, A11, A16, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32 }	95.45
16	{A1, A4, A7, A11, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32}	90.22
15	{A1, A4, A11, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32}	90.13
14	{A1, A11, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32 }	86.87
13	{A1, A11, A17, A18, A20, A21, A25, A27, A28, A29, A30, A31, A32 }	83.19
12	{A1, A11, A17, A18, A20, A21, A25, A27, A28, A30, A31, A32 }	81.45
11	{A1, A11, A17, A20, A21, A25, A27, A28, A30, A31, A32 }	81.12
10	{A1, A11, A17, A20, A21, A27, A28, A30, A31, A32 }	79.56
9	{A1, A11, A17, A20, A27, A28, A30, A31, A32 }	73.17
8	{A1, A11, A17, A20, A27, A28, A31, A32 }	70.23
7	{A1, A11, A17, A27, A28, A31, A32 }	68.32
6	{A1, A11, A17, A28, A31, A32 }	67.19
5	{A1, A11, A17, A28, A32 }	65.12
4	{A1, A17, A28, A32 }	64.59
3	{A1, A28, A32 }	60.57
2	{A1, A32 }	58.79
1	{A1}	55.79

It is clear from Table 3.5 that the accuracy is decreasing after 17 features and the same from the 18 features. Therefore it is optimal to take these 17 features as the

final set of features. The 17 features are {A1, A4, A7, A11, A16, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32 }.
Final Set-4= {A1, A4, A7, A11, A16, A17, A18, A19, A20, A21, A25, A27, A28, A29, A30, A31, A32 }.
The details of these attributes and attributes id are given in Table 2.1. As the output of the feature selection and extraction phase, the feature set of 17 features is achieved, which is input to the next layer.

3.3 Data Prediction (DP) Layer

DEM provides a novel NFRs prediction technique for software development, which is based on computing devices' computational capabilities. The NFRINDEX-based requirement analysis is predicted using a DL-assisted Multi-scaled Long Short-Term Memory (M-LSTM) model. M-LSTM combines the 2 most popular deep learning techniques: (i) CNN and (ii) LSTM(Long Short-Term Memory) which is shown in Figure 3.4 and the flow of data is depicted in Figure 3.5. The CNN retrieves the patterns from the categorised datasets and pools them with the LSTM network for sequential learning to predict the next likely data value.

Each section of predicted data is treated individually. In the proposed technique, CNN analyses conceivable local patterns by looking at a cluster of expected events for a given window ΔT based on the data collected. The local patterns are then updated using the Rectified Linear Unit (ReLU) pattern transformation facility, which is a non-linear function.

In addition, the max-pooling algorithm has been utilized to convert the changing patterns into feature vectors. Eventually, the feature vectors are sent into an LSTM network, which analyses and predicts the next data value. The entire pattern extraction and event prediction procedure are described in detail below.

1. Convolutional Neural Network (CNN): To read the local pattern from the data occurrences, convolutional layers are integrated at the top of the network. With a window size of $2d + 1$, the filter P is built using a granule technique. The pattern extraction procedure is mathematically described as. The filter P is created using a granule approach with a window size of $2d + 1$. The pattern extraction procedure is mathematically described as

$$d_t = ReLU\left(b + \sum_{j=-b}^{b} W_j x_{t+j}\right) \tag{3.4}$$

where Wj represents the kernel of the convolution layer at the relative location j, and dt represents the filter in a specific state t. The relative position's size is determined by the window's size, which is $2d + 1$. In the current scenario window size is 3. The nonlinearity is preserved by using the ReLU activation function, which is mathematically written as ReLU(x) = max $\{0, x\}$, and the bias value is represented as b. The goal of using the rectified linear function is to improve the data pattern's strength by removing the minimal data patterns. The patterns are pooled after local extraction to turn the data into global vectors as follows:

$$\overline{d} = max\{d_t\} \tag{3.5}$$

where $\overline{d}$ signifies the global feature vectors. The element-wise multiplication operation is performed by the *max* operation. After converting the local patterns to global pattern vectors, the vectors are transferred to the proposed M-LSTM network for NFRs determination.

2. M-LSTM (Multi-scaled Long Short Term Memory): The presented Multi-scaled LSTM network integrates 3 data processing layers into a deep neural network-like structure. The model's base layer is made up of M-LSTM memory cells that are responsible for coping with the captured irregular

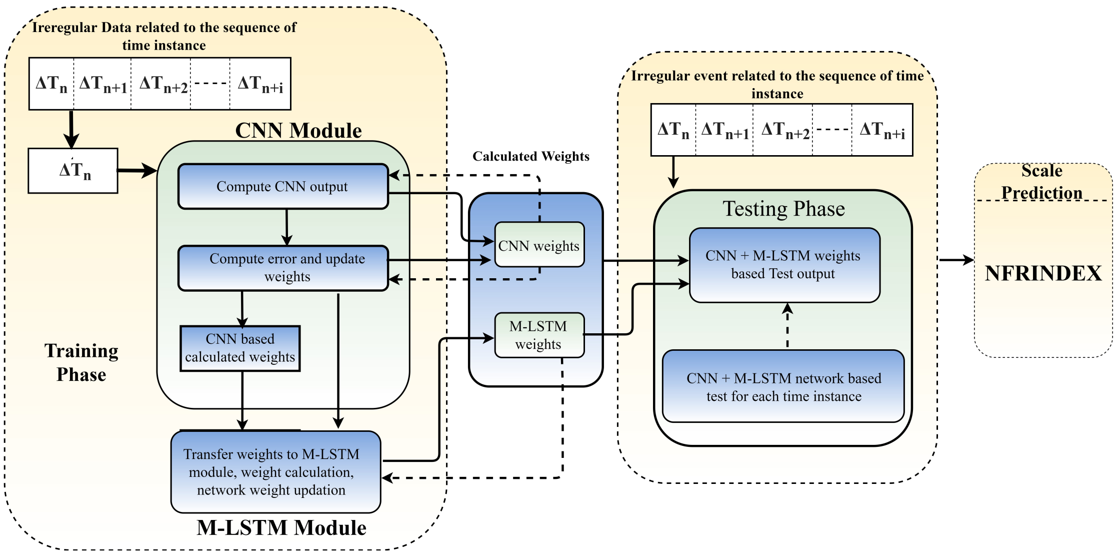

Ireregular Data related to the sequence of time instance
ΔTn ΔTn+1 ΔTn+2 ΔTn+i
ΔTn
CNN Module
Compute CNN output
Compute error and update weights
CNN based calculated weights
Training Phase
Transfer weights to M-LSTM module, weight calculation, network weight updation
M-LSTM Module
Calculated Weights
CNN weights
M-LSTM weights
Irregular event related to the sequence of time instance
Testing Phase
CNN + M-LSTM weights based Test output
CNN + M-LSTM network based test for each time instance
Scale Prediction
NFRINDEX

FIGURE 3.5: Proposed Hybrid M-LSTM and CNN network(ΔT denotes temporal NFRs value)

data-oriented patterns. The goal of using LSTM's sequential data efficiency is to solve two core issues:

(a) Evaluation of relationships between discrete inputs of the given parameters and,

(b) Inconsistent data-driven analysis.

More precisely, the sequence of patterns supplied as input is represented as $d_t = \{v_1, v_2, ..., v_n\}$, where $d_t = [x_t, \Delta t]$. Each input data sequence is represented as a feature vector, $x_t \in R^M$, where M defines the vector dimensions of the feature x_t. The difference between the current input sequence and the prior input sequence in a given module (ΔT) is denoted as D_t. M-LSTM computes the sequence of irregular states such as $h_t = \{h_1; h_2; ...; h_n\}$, where the vector is denoted as $h_t \in R^K$ and the dimensionality of the vector is specified by K.The second layer employs a multi-scaled weighted pooling function to conduct the aggregation operation on the predicted irregular states.

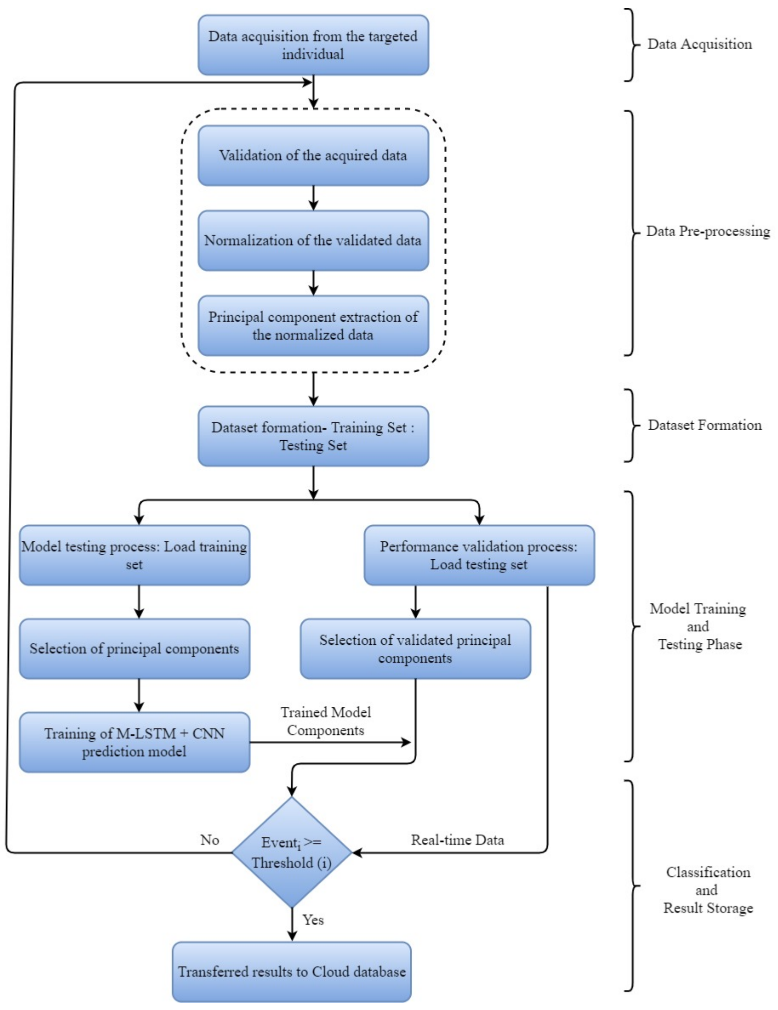

Data acquisition from the targeted individual
Validation of the acquired data
Normalization of the validated data
Principal component extraction of the normalized data
Dataset formation- Training Set : Testing Set
Model testing process: Load training set
Performance validation process: Load testing set
Selection of principal components
Selection of validated principal components
Training of M-LSTM + CNN prediction model
Trained Model Components
Event$_i$ >= Threshold (i)
No
Real-time Data
Yes
Transferred results to Cloud database
Data Acquisition
Data Pre-processing
Dataset Formation
Model Training and Testing Phase
Classification and Result Storage

FIGURE 3.6: Workflow of the Proposed Model

The aggregation operation on predicted irregular states is defined as $h = pool\{h1; h2; ...; hn\}$, in, $h \in R^K$represents the severity scale. The outcome probability is estimated by the third and topmost fully connected layers of the presented NFRs model, as shown below:

$$p(y|u_{1:n}) = p(N_y(h)) \tag{3.6}$$

(a) Parameter Relationship Analysis

The operation of M-LSTM, which is responsible for evaluating the relationship between the specified factors to evaluate the dynamics of NFRs, is thoroughly described. The M-LSTM model is also in charge of determining the significance of anomalies in the events collected in a specific module ΔT. As previously stated, the occurrences are classified as either Significant NFRs or Non-Significant NFRs. Once the significant events for the given parameters have been established, this component is used to deliver significant events as input to the M-LSTM.

$$i_t = \frac{1}{m_t}\sigma(W_i X_t + U_i h_{t-1} + b_i) \tag{3.7}$$

where the type of event is represented by m_t. The value of m_t in the proposed study will always be 1, and σ represents the sigmoid function of a vector. Because the input feature values are in charge of analysing significant NFRs, the output gate is moderated by the currently accepted parameters, which control the state of irregularity as follows:

$$O_t = \sigma(W_O X_t + U_O h_{t-1} + b_O) \tag{3.8}$$

Furthermore, The parameters chosen may have a long-term impact on NFR stability. This entails the following procedure for forgetting the previously evaluated event and determining the next possible event as shown below:

$$f_t = \sigma(W_f X_t + U_f h_{t-1} + b_f) \tag{3.9}$$

(b) NFRINDEX-oriented Analysis

The history of significant data is stored in the memory cell, which is modeled and evaluated using the LSTM's sequential data processing efficiency. However, because data changes over time, the memory cells must be updated on a regular basis. The parametric time mechanism is implemented in M-LSTM by altering the gate f as follows:

$$f_t = \sigma(W_f X_t + U_f h_{t-1} + Q_f q_{\Delta t-1:t} + b_f) \tag{3.10}$$

where the time gap between current data and previous data is determined by $q\Delta t - 1 : t$, which is additionally represented as a derived vector. In addition, Qf describes the current module's parametric weight matrix. The next possible significant NFR $ht+1$ is registered after the setup of M-LSTM units for each time step is completed. Following that, the current determined states have been utilized to forecast the next significant NFR. The static computation of NFRINDEX is estimated using Algorithm 2. As it can be seen, in the current algorithm, several NFRs are incorporated. Henceforth, if there are 'n' number of NFRs for 'm' parameters, then the computational complexity is given by O(mn). In other words, the temporal computational complexity for a large number of parameters is given by O(n^2)

3. System Training: To determine the significant NFR in the proposed study, short-term prediction operations were used. The training of short-term models is frequently used to determine long-term significant NFR. Transfer learning is the process of determining the next possible large NFR based on the data from prior events. Models are trained for short-term operations by minimising the log-loss $L = -\Sigma_t logP(y_t|u_{1:t})$, where y_t represents the input codes. The completely differentiable loss functions are used in the suggested model, which can be reduced using the traditional back-propagation technique despite its complex structure. The kernel size is 3. The input feature shape is(1,17) as there are 17 features giving input for the 1-d CNN. The shape of the initial weight matrices is 1*17 for each kernel. The output layer is generated after applying LSTM. It generates a value between 0 to 1 as prediction probability. Standard hyperparameters are used for the present research. ADAM optimizer is used to optimize the model. ADAM is an optimization algorithm, as a substitute for a classical stochastic gradient descent

system to update network weights in training data. This is used to perform optimization and is one of the best optimizers at present [262]. The linear learning method is used in this proposed solution, based on a pre-determined set of parameters.

Algorithm 2 NFRINDEX: NFR Significant Determination Procedure

```
1: Input NFR data values for n parameters and associated values$\alpha, \beta, \lambda, \delta$ are the
   associated weight.
2: Initialize NFRINDEX=Null(0)
3: Compare LoNFRS value of NFR parameter 1 with prefix threshold value.
4: if LoNFRS$_1$ $> \gamma 1$, then
5:     Add $\alpha *$LoNFRS$_1$ to NFRINDEX
6: end if
7: Compare LoNFRS value of NFR parameter 2 with prefix threshold value.
8: if LoNFRS$_2$ $> \gamma 2$, then
9:     Add $\beta *$LoNFRS$_2$ to NFRINDEX
10: end if
        *Do for n parameters*
11: Compare LoNFRS value of NFR parameter n with prefix threshold value.
12: if LoNFRS$_n$ $> \gamma_n$, then
13:     Add $\beta *$LoNFRS$_n$ to NFRINDEX
14: end if
15: Cummulative NFRINDEX =$\alpha$*LoNFRS$_1$ + $\beta$* LoNFRS$_2$ + $\lambda$*LoNFRS$_3$ +
    ......$\delta$*LoNFRS$_n$
```

The associated values$\alpha, \beta, \lambda, \delta$ are the associated weight that is decided by the domain expert. For illustration, all the attributes are given the same weightage as all are considering the same. The value can be changed according to domain and software. The threshold values are decided with the help of domain experts.

3.4 Data Analysis and Visualization (DAV)Layer

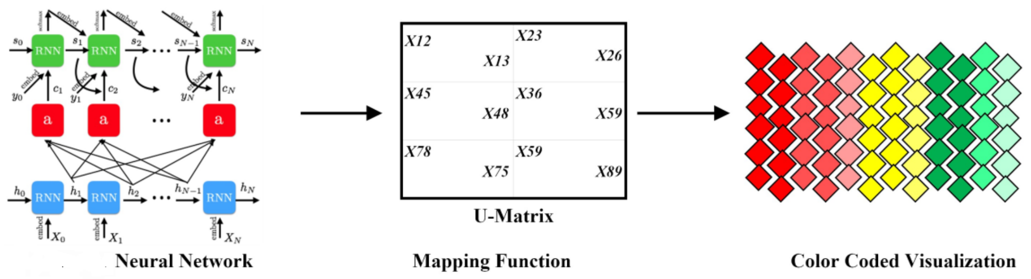


FIGURE 3.7: Self Organized Mapping technique for Visualization [87]

DAV layer is crucial in the proposed model. The Self Organized Mapping(SOM) Visualization technique is a very efficient color-coded scheme as shown in Figure 3.6. The results are visualized using this technique. The visualization of the NFRs Prediction model is a very useful tool for software developers and concerned. Specifically, LCD (Liquid Crystal Display) is required to display the user's output or results. In comparison to the quantitative display of parameters, the SOM technique enhances the predictive outcome of the model. SOM is a technique based on the color-coded interactive display. The prediction model is visualized using U-matrix for embodying the SOM technique. The paper [263] has shown the U-Matrix and its usage. The figure above shows the implementation and visualization of U Matrix utilizing the SOM technique [87]. The upper value of NFRINDEX is depicted using the red color, and lower values are identified using the green color.

3.5 Conclusion

This framework consists of 4 different layers. These layers work efficiently for data acquisition, feature selection and extraction, data prediction, and visualization. The various elicitation techniques have been discussed, such as questionnaires, interviews, Brainstorming, and templates. The Bayesian belief method has been used to classify data into significant and non-significant categories. Various baseline and ensembling methods are used for data prediction. The color-coded SOM technique is used for data visualization. This framework is quite significant for NFRs elicitation and prediction.

Chapter 4

Experimental Setup

4.1 Data Collection

There are various techniques discussed in the literature for requirement elicitation. But every method has its own merits and demerits. Some methods are suitable for one field and some for other areas. The different techniques have been explored in various research works. The use of the elicitation technique is largely determined by the resources available. When resources and funds are limited, the questionnaire is the most cost-effective method of gathering requirements because the administrative costs are significantly lower. The questionnaire can also save time by gathering information from a large number of people in a short period. The type of data that must be gathered with the questionnaire is determined by the respondent's level of knowledge and background. There is no set rule for when the questionnaire should be used to gather requirements. The questionnaire is the most suitable for our research work. The questionnaire is easy to design and distribute. Various researchers, academicians, and IT professionals can be approached using online questionnaires across the world. While designing the questionnaire, the things which are kept in mind are as follows:

- The goal of the survey has been defined.

- The survey's sample group of respondents is chosen in advance.
- Developing and preparing the questionnaire
- The Questionnaire Process is being carried out.
- Getting the data and analyzing it.
- The questions are well arranged in the questionnaire as particular questions follow general questions.
- The questions that are relevant to the main subject should be prioritized and stated at the beginning of the questionnaire.

The questions are arranged in such a way that easy questions come first. In other words, the questions are put in order of known to unknown. In our research work, the online questionnaire has been designed with the help of Google forms. Initially, The questionnaire consisted of 43 questions and different options. The questions have been taken from the other problems and challenges given in various papers regarding NFRs. As per the advice of experts and backed up by extensive literature review Usability has been shortlisted to be a factor for consideration. The questions are based on the usability perspective of NFRs. Usability is very important because it focused on the user experience. It is usually considered the use of a system and is easy to access. Usability is usually divided into UI (User Interface) and UX (User Experience). The usability factor tells about the satisfaction of the user and the experience of the user with the system. The process of designing, sharing, and visualizing has been performed in different layers. The various layers have been designed for eliciting the data using a questionnaire and analyzing the survey results. The different layers are (i) Questionnaire Designing Layer and, (ii) Questionnaire Distribution Layer. The online Google docs have been used as a tool for designing the questionnaire. Online mode is quite helpful in eliciting responses over the globe. The various phases are explained below:

1. *Questionnaire Designing Layer:* It is the initial layer of the current model. In this layer, the different NFRs-related papers are explored. The search engines like Google Scholar, IEEE, ReseachGate, and Science Direct to name a few, are explored. The keywords like Requirements, Functional Requirements, Non-Functional, Elicitation techniques, and Usability have been searched for collecting NFR-related essential questions. While designing the questionnaire the biasness of data has kept in mind. The rough drafts of the questions are designed. The suggestions of the various IT industry experts have been taken for the drafted questions. The changes they suggested were incorporated iteratively and followed till the final questionnaire was developed. A questionnaire is based on the usability perspective. It consists of various questions which are taken from the literature review and the domain expert. Already available usability-based questionnaires are taken as a base of this questionnaire generation. Chiew and Salim [245] have designed a usability questionnaire consisting of 24 questions to measure the usability of any kind of website. Mustafa and Zouabi [233] designed the usability-based questionnaire based on the websites of Jordanian universities. In the questionnaire 23 usability criteria are discussed. WAMMI is a questionnaire that consists of 20 questions that are focused on the user satisfaction perspective of usability [264]. Avouris et al. [265] designed an online questionnaire and WAMMI questionnaire for evaluating the usability perspective of web portals that are serving as an academic department. Aziz and Kamaludin [266] designed a usability-based questionnaire with 51 questions. The authors have taken many questions from WAMMI and Computer System Usability Questionnaire (CSQU) [267]. Moreover, domain knowledge is also explored for the questionnaire design. There are different departments of Lovely Professional University (LPU). The specific department namely Venture Pack of LPU manages the University Management System (UMS), computer modules, the website of LPU, LPU Touch (Application), and all technical tasks

of the university handle all the tasks. There are approximately 100 employees who are working in the Venture Pack. The head of the venture pack has been consulted for designing and modifying the questionnaire. The website development and maintenance cell of DAV Amritsar has also helped us in the designing of the questionnaire. Various IT professionals have consulted for the same. Moreover, generic prediction helps to cover a wide range of software domains. The questionnaire can be customized and updated according to the requirement of the software and client. For instance, the significance of the on-screen messages indicating important announcements and notifications can be customized for the academic software. The Likert scale has been considered for the response to the questionnaire. For instance, the options are given as responses like Very critical, Critical, Neutral, Less critical, and Not critical. Precisely, a particular question that what the respondents feel about the completeness of the questionnaire is asked. The questions related to the NFRs consist of the inquiry regarding the importance of the compatibility of the website, the significance of the page loading time, the importance of cookies and cashing, the importance of watermarking of text boxes, navigation of the website, the importance of font selection, Accessibility, Adware elements, Orphan pages, placement and content of the site map, Frequency and position of advertisement, Data confidentiality, Website security, Accuracy of content, Website focusing on a particular objective, Compatible across different browsers, Hyperlink description, and updating of contents. These questions are based on the usability perspectives of the NFRs. Usability is a very significant factor of the ISO/IEC 25010. The quality of any software is based on the usability or user interaction of the system. Table 4.1 has shown the various questions derived from literature and IT experts for designing the NFRs questionnaire.

Table 4.1: Initial 43 NFRs based questions ✗represents questions suggested by domain experts

Reference	**Non Functional Requirements based Questions**
[49]	"What is the importance of Non Functional Requirements?"
[233]	"What is the importance of Irritating elements in webpage accessibility?"
[234]	"What is the importance of Orphan pages in a webpage?"
✗	"Placement and content of site map is helpful?"
[235]	"Does various Link color matters?"
[236]	"How often should website content be updated?"
[237]	"Low Download time required downloading a webpage?"
[238]	"There should be a Back button on our webpage?"
[239]	"Webpage Respond according to userś expectations?"
[240]	"There should be Hyperlink descriptions?"
[241]	"Design should be Consistent?"
[230]	"Is it important your website is compatible across different browsers?"
[268]	"Is website related to information in it is true?"
[269]	"Is website focusing on a particular objective?"
[270]	"When a website is updated last time?"
[271]	"Website content guarantee accuracy?"
✗	"Does website works on different web browsers?"
[272]	"Website professionally designed is reliable?"
[273]	"Why is website security so important?"
[274]	"Do you need a website security?"
[275]	"How do you ensure data confidentiality?"
[276]	"Is it necessary for the system to have the same look and feel as existing apps?"
[277]	"What is the importance of Website Accessibility?"
[242]	"Are user manuals required?"
[270]	"After clicking the icon, link, or button, how long should it take to load (render) the full screen (with all images and buttons) of the application?"
[278]	"How will the system be supported, such as command-line and graphical user interfaces?"
[279]	"Should a corporate database definition policy such as a table, keys and column names to be followed?"
✓	"Is there any need for the detail description of the rows and tables of the data model?"
[245]	"How important is the attractiveness of the interface for your website?"
[247]	"How important is error handling for your website?"
✗	"Do you require a search feature on your website?"
✗	"The requirement of web assistance for your website?"
[241]	"Should design be dynamic?"
[236]	"How significant is the navigation and sitemap for a website?"
✗	"What kind of web assistance is required for the website?"
[245]	"What is the maximum time the GUI can take to respond to a user's action, such as clicking a button to advance to the next screen?"
✗	"What are the system's operating hours?"
✗	"What are the system's busiest hours when the most people use it?"
✗	"What is the total number of people who use the system?"
✗	"How many users are likely to use the system at any given time?"
✗.	"During peak hours, what is the total number of concurrent system users?"
✗.	"What is the total number of users in each of the system's geographical locations?"
✗	"How do you feel about the completion of the questionnaire?"

2. *Questionnaire Distribution Layer:*

   It is the second layer of the present model. An important task is to distribute the questionnaire and collect the response. The online questionnaire has been designed using Google Forms. The link to the questionnaire and description has been distributed using various sources. The questionnaire is shared on Facebook, and WhatsApp, with IT Companies, and academic institutes. The questionnaire has also been sent personally to people in contact who are dealing with the IT industry and the Computer Science faculty of the different colleges and universities. The academicians and IT professionals are asked to participate in the survey so that the views of a diverse group of people can be acquired. The stakeholders are asked about the perceived importance of NFRs from the perspective of ISO/IEC 25010 parameters in an online survey. Respondents were asked to consider a project they had worked on in the past and determine whether the practice was Critical, Important, or Neutral. Respondents could also choose to skip the question if it made them uncomfortable. The demographic information is also asked from respondents who are responding to the question concerning NFRs. The initial questionnaire has been designed, and proofreading has been done by the professor of DAV College, Amritsar. The data collection started in the IT department of DAV College. The minor modifications have been made, which are suggested by the staff of Computer Science of DAV College, Amritsar. Then the questionnaire has been distributed using the link to Google Docs Form. While deciding on the target population, we did not restrict ourselves to a specific domain or application. Our focus of the survey is to get a view of the NFRs based on ISO/IEC 25010. We have tried to cover various groups of people. We assumed that the respondent must know about software development. However, it is found very difficult to access the respondents of the said population. So in our research, we followed the convenient sampling method. The questionnaire has been shared on social media software development groups, Reddit, and also on LinkedIn.The questionnaire was also communicated to friends through Whatsup messenger. A direct invitation has been given to the LPU software development cell. The placement cell of LPU has sent the questionnaire to the students who are placed in companies. The various colleges and universities of Amritsar were approached for questionnaire filling. Direct e-mails have been sent to the professionals whose e-mail ids have been taken from GitHub and LinkedIn. The simulation of the proposed system is performed in the real-world dataset, which is acquired using Google forms. The NFRs dataset of 312 IT professionals and academicians is gathered for 33 attributes. The responses are taken in the form of a 5 Likert-type scale. Data preprocessing is crucial in any ML problem. In preprocessing of the dataset, the different responses are converted into numeric values. The null and missing values are also replaced with appropriate values. But still, for ML, a large number of the dataset has required. To overcome this problem, the data augmentation technique namely SMOTE has been applied for multiplying the dataset and finally getting the 5304 instances for experimental purposes. Data augmentation refers to processes that are used to increase the amount of data measured by adding slightly altered copies of previously existing data or newly created synthetic data from existing data. When training an ML, it acts as a regulariser and reduces overfitting. There is a problem with ML especially with deep learning models when the experiment is performed on a small dataset. Initially, the model predicts well and also provides good accuracy on the original dataset. But with the newer dataset, the accuracy drops. Data augmentation solves the problem of overfitting and also regularizes the data. So the augmentation technique solves the problem related to small datasets and overfitting. The population consists of software developers from diverse domains and applications. The 3 dominating domains

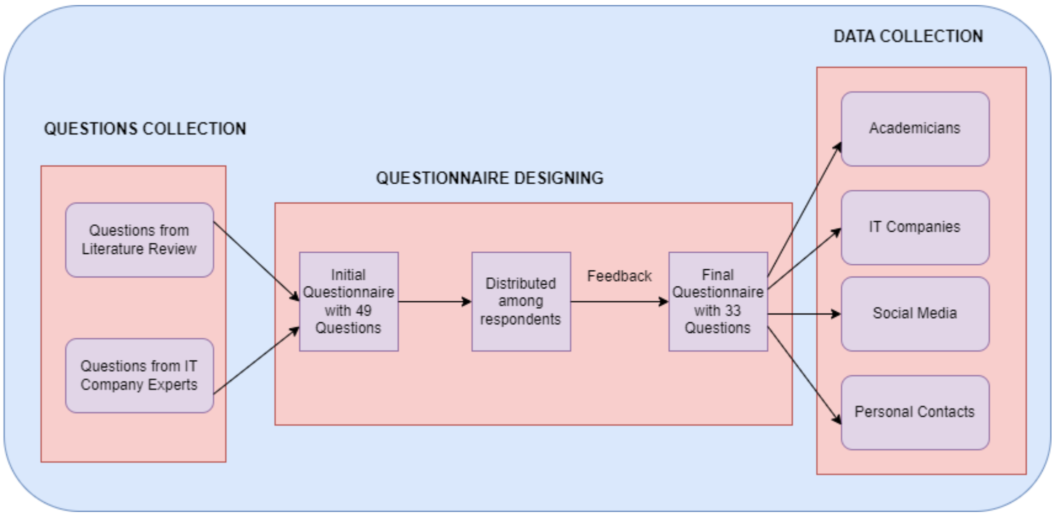

DATA COLLECTION
QUESTIONS COLLECTION
QUESTIONNAIRE DESIGNING
Questions from Literature Review
Questions from IT Company Experts
Initial Questionnaire with 49 Questions
Distributed among respondents
Feedback
Final Questionnaire with 33 Questions
Academicians
IT Companies
Social Media
Personal Contacts

FIGURE 4.1: Questionnaire Designing and Data Collection

are Educational websites, Mobile application development, and financial applications. The different Project managers, Software Designers, Testers, and Requirement Analysts participated in the survey. The participants are from India and around the globe. The respondents have experience from 1 year to 10 years. The 5 respondents have chosen another response and are not considered for the analysis. The number is relatively small and does not have any effect on the final analysis. The validity threat is that the Likert scale is subjective and perceived differently by different people. Some of the respondents have given a subjective response as a response to any other. These questions are handled manually, and one of the respondents has given his opinion of No need for NFRs at all. More than 70% of respondents have shown positive responses to the completion of the questionnaire. The different charts are used in this phase to visualize the results of the questionnaire. The bar diagram and pie charts have been explored to show the results of the different questions.

Figure 4.1 has shown the different phases of questionnaire design and data collection phases. Initially, the questions have been taken from the literature review where problems and challenges are discussed about NFRs. The software professional helped in designing questions and provided the Likert scale for the answers. The questionnaire is distributed among various academicians and IT professionals. The questionnaire has run in 4 rounds for getting more responses. In the first round, it is shared among the computer science staff of various colleges and universities of Amritsar. The data has been collected, and the questionnaire has been discussed with different experts and made a modified questionnaire. The questionnaire included participants' perspectives on the coverage of questions about NFRs from a usability view. In the second run, various IT Companies have been approached. The placement cell of LPU has helped in that and sent the questionnaire to multiple students of LPU who are in the software development field and placed in MNCs. The placement cell of DAV College has helped in filling out the questionnaire from various IT companies. The software development cell of LPU has discussed the questions with us and consulted the experts regarding the questionnaire. The questionnaire has been shared among various friends and colleagues in the IT field. Finally, the questionnaire data has been collected in good numbers.

Chapter 5

Results and Discussions

5.1 Implementation Analysis

This section discusses the experimental implementation of the proposed model for performance assessment. A laptop configuration including an Intel Core i5 processor and 32 GB RAM is required. As mentioned earlier, the presented model comprises different layers. In the initial layer, the data is collected from various stakeholders. Then, the data values are categorised into significant and non-significant values in the next phase using the BBM technique. Finally, the Hybrid M-LSTM and CNN-based M-DL model is used to estimate the NFR's significance in real-time. The experimental implementation is performed by focusing on the following objectives:

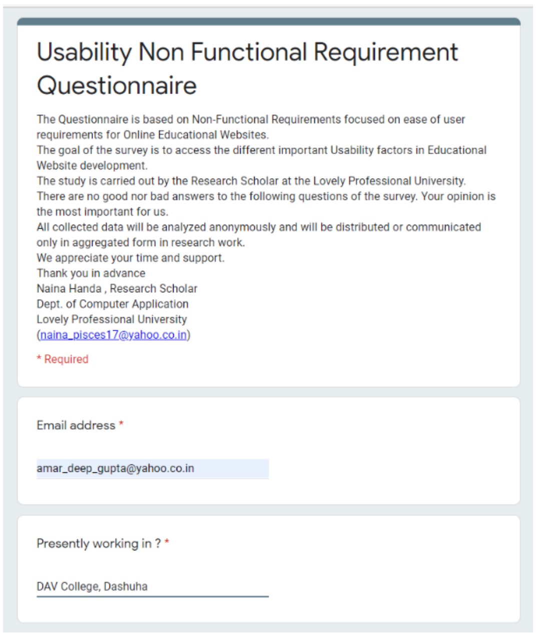
1. Evaluate the efficiency of the data completeness based on the data collected using Google forms.
2. Determine the classification efficiency of the proposed system using the BBM.
3. Estimate the Feature Selection efficiency for the identification of significant features.
4. Evaluate the NFRs prediction Accuracy of the presented system.

5. Evaluate the overall reliability of the current prediction system over a primary dataset.

6. Estimate the stability of the system to determine the durability of the proposed system.

5.1.1 Deployment Environment

The deployment of the proposed system is performed over the real-world dataset, which is acquired using Google forms. The NFRs dataset of 312 IT professionals and academicians is gathered for 33 attributes and 5304 instances. The data is collected using various online platforms such as Facebook, WhatsApp, Reddit, and Academic Institutes. The google form is also sent to the different IT societies on Facebook like research scholars, research gate, software developers groups, and software engineering groups, to name a few. The Lovely Professional University (LPU), India, has provided questionnaire-based data gathering from the computer science staff and software development department. The questionnaire consists of the different NFRs related to 33 questions, as shown in the above figure. It gathered data regarding the email id of the respondent, present workplace, and appropriateness of the questionnaire. The questions related to the NFRs consist of the inquiry regarding the importance of the Compatibility of the website, the Significance of the page loading time, the Importance of cookies and cashing, the Significance of watermarking of text boxes, the Navigation of the website, Significance of font selection, Accessibility, Adware elements, Orphan pages, Placement and content of the site map, Frequency and position of advertisement, Data confidentiality, Website security, Accuracy of content, Website focusing on a particular objective, Compatible across different browsers, Hyperlink description, and Updating of contents. These questions are based on the usability perspectives of the NFRs. Usability is a very significant factor as per ISO 25010. The quality of any software is based on the usability or user interaction of the system. The responses



Usability Non Functional Requirement Questionnaire

The Questionnaire is based on Non-Functional Requirements focused on ease of user requirements for Online Educational Websites.
The goal of the survey is to access the different important Usability factors in Educational Website development.
The study is carried out by the Research Scholar at the Lovely Professional University.
There are no good nor bad answers to the following questions of the survey. Your opinion is the most important for us.
All collected data will be analyzed anonymously and will be distributed or communicated only in aggregated form in research work.
We appreciate your time and support.
Thank you in advance
Naina Handa , Research Scholar
Dept. of Computer Application
Lovely Professional University
(naina_pisces17@yahoo.co.in)

* Required

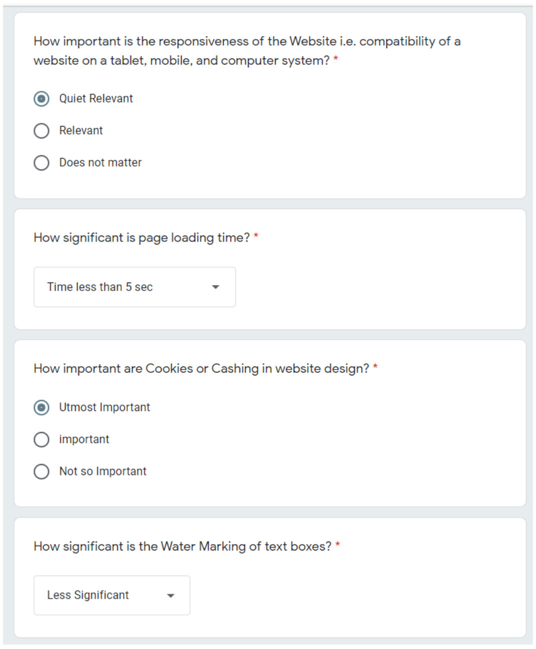
Email address *

amar_deep_gupta@yahoo.co.in

Presently working in ? *

DAV College, Dashuha

(a)



How important is the responsiveness of the Website i.e. compatibility of a website on a tablet, mobile, and computer system? *

[✓ Quiet Relevant]
Relevant
Does not matter

How significant is page loading time? *

Time less than 5 sec

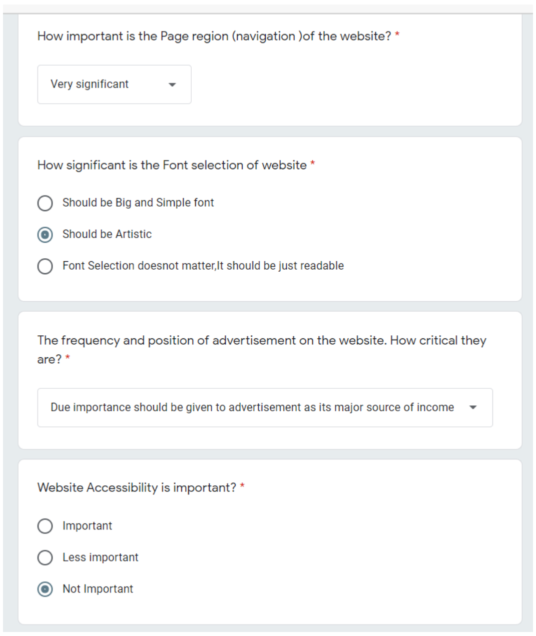
How important are Cookies or Cashing in website design? *

[✓ Utmost Important]
important
Not so Important

How significant is the Water Marking of text boxes? *

Less Significant

(b)



How important is the Page region (navigation)of the website? *

Very significant

How significant is the Font selection of website *

Should be Big and Simple font
[✓ Should be Artistic]
Font Selection doesnot matter,It should be just readable

The frequency and position of advertisement on the website. How critical they are? *

Due importance should be given to advertisement as its major source of income

Website Accessibility is important? *

Important
Less important
[✓ Not Important]

(c)

Figure 5.1: Questionnaire Form

are taken in the form of a 5 Likert-type scale. To normalise the data, it is converted into numeric data, and null values are replaced. The data augmentation technique is used for multiple instances of data. Figure 5.1 has shown a few glimpses of a questionnaire.

5.1.2 Data Completeness Efficiency

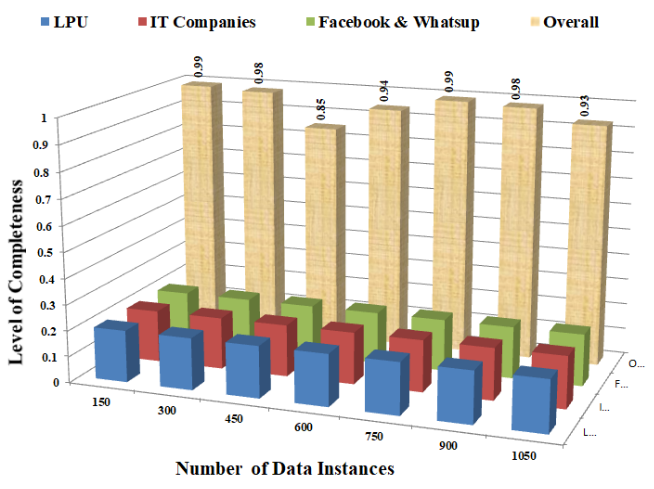


FIGURE 5.2: Level of Completeness

The dataset is collected from different IT professionals and academic institutes. The questionnaire is divided into various sections and sub-sections. A few glimpses of the questionnaire are shown in above Figure 5.1. Data completeness is a significant factor in the dataset. It refers to the number of questions answered by the respondent. The questionnaire is sent to approximately 900 persons, out of which more than 300 respondents have given a response. But many respondents did not answer all the questions, so there is a need to consider the completeness of the questionnaire. The primary 4 sources of the responses are Lovely Professional University (LPU) India, DAV College, Amritsar, Facebook, and WhatsApp Messenger. The completeness is measured on a scale from 0 to 1. Figure 5.2 shows the completeness of the questionnaire. The LPU dataset has given 95% of completeness, IT Companies have registered 89% of completeness, and Facebook and

Whatsapp have attained 78% of completeness. The results have shown that the average completeness is 87%, whereas the LPU has achieved maximum efficiency in data completeness.

5.1.3 Feature Selection Efficiency

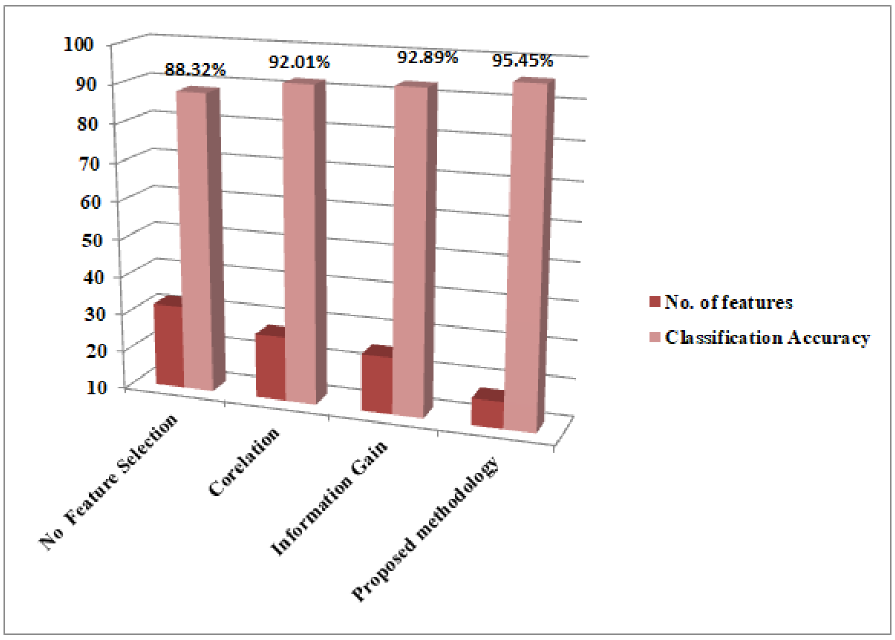


FIGURE 5.3: Feature Selection Accuracy

The hybrid feature selection algorithm has designed and evaluated the classification of NFRs into significant and non-significant groups. The performance of the proposed technique is compared with the none feature selection approach and the other 2 frequently used filter-based feature selection approaches, such as information gain and correlation. In the experiments, the prepared dataset is trained and tested to evaluate the classification accuracy of the proposed model. The key idea of the proposed method is to combine the efficiency of the filter method and the Accuracy of the wrapper method. A model has been designed where the two filter techniques, such as information gain and correlation, have been utilised as preliminary procedures for the model. The intersection of both feature sets has been computed. The resultant feature set further provides the wrapper method where the backward search technique and BBM were used as machine learning models. The results are depicted as a graph in Figure 5.3. The results have shown that when no feature selection technique has been applied, and all 33 features are considered for classification the Accuracy is 88.32%. After applying the correlation and information gain with features set 27 and 25 the Accuracy is 92.01% and 92.89%. But for the proposed model, the features are reduced to 17 and the Accuracy is 95.45%, which is relatively higher than other techniques. Hence, the proposed hybrid feature selection model is relatively suitable for the NFRs classification.

5.1.4 Classification Efficiency

Table 5.1: Classification Efficiency in terms of Precision(Pre), Recall(Rel), Specificity(Spe), F1 score(F1)

Model	BBM				DT				SVM			
Dataset	Pr(%)	Sp(%)	Rel(%)	F1(%)	Pr(%)	Sp(%)	Rel(%)	F1(%)	Pr(%)	Sp(%)	Rel(%)	F1(%)
500	93.47	92.08	92.43	92.93	92.22	91.87	92.11	92.16	92.17	91.14	92.40	92.28
1000	92.78	90.34	93.42	93.09	91.43	91.12	93.05	92.23	90.17	91.43	93.15	91.63
1500	93.65	91.56	93.63	93.64	93.13	90.55	92.16	92.64	93.07	90.98	92.35	92.70
2000	94.88	90.78	92.22	93.53	92.24	90.81	91.13	91.68	92.16	91.86	90.16	91.14
2500	93.62	91.84	92.13	92.86	93.21	92.15	92.05	92.62	93.09	92.08	92.86	92.97
3000	92.89	93.73	93.76	93.32	92.89	91.23	93.06	92.97	92.64	92.27	93.67	93.15
3500	94.75	91.87	92.11	93.41	93.67	90.05	91.86	92.75	93.13	91.68	91.49	92.30
4000	94.57	91.87	93.46	94.01	92.56	91.14	93.05	92.80	91.08	91.45	93.01	92.03
4500	93.76	92.95	93.03	93.39	92.73	92.25	92.77	92.75	92.03	92.69	92.15	92.08
5000	95.45	93.06	94.84	95.14	93.84	90.03	91.67	92.74	93.43	90.15	91.10	92.25

In the classification task of the proposed model, the Precision for a model is the number of instances correctly labelled as belonging to a significant class divided by the total number of instances labelled as belonging to a significant class. A recall is defined as the number of instances correctly classified to a significant class divided by the total number of instances that belong to the significant class. A high Precision value means that every classification done by the model is relevant but doesn't tell whether all relevant instances are classified. The higher value of Recall means all the relevant instances are classified by the model but does not say anything about how many irrelevant instances are also classified. These two metrics are not specifically useful when used in isolation. There is an inverse relationship between Precision and Recall, where it is possible to increase one at the cost of the other. The model needs high Recall in case of critical problems and high Precision when it is more focused on true positive and false positive and does not care about the false negative. The main key to the problem of high Recall or Precision is the probabilistic threshold value assigned for classifying the instances. To maximise the value of Recall, the probabilistic threshold value has been set below 0.5, like 0.2 in our case.

For illustration, greater than 0.3 is a significant class, and 0.1 is a non-significant class. This has increased the system's Recall and reduced the value of Precision. For Precision, the probabilistic threshold must be set to a higher value like 0.7 in the current model. This increases the Precision value of the model and reduces Recall's value. The Precision-Recall trade-off is fundamental when Recall is very important than Precision and vice versa. The fundamental idea behind the trade-off of Precision and Recall is that the change in threshold value for determining the class cause Recall to increase and Precision to decrease or vice versa. So, the Precision and Recall are computed at each threshold, and out of the results, the graph is plotted for finding the best trade-off point and the same threshold for the model. Figure 5.4 has plotted the graph for different values of Precision and Recall concerning various threshold values. The graph depicts that the best threshold in current research is 0.5 where the Precision and Recall cuts each other. In the classification task of the proposed model, the Precision for a model is the number of instances correctly labelled as belonging to a significant class divided by the total number of instances labelled as belonging to a significant class. A recall is defined as the number of instances correctly classified to a significant class divided by the total number of instances that belong to the significant class. A high Precision value means that every classification done by the model is relevant but doesn't tell whether all relevant instances are classified. The higher value of Recall means all the relevant instances are classified by the model but does not say anything about how many irrelevant instances are also classified. These two metrics are not specifically useful when used in isolation. There is an inverse relationship between Precision and Recall, where it is possible to increase one at the cost of the other. The model needs high Recall in case of critical problems and high Precision when it is more focused on true positive and false positive and does not care about the false negative. The main key to the problem of high Recall or Precision is the probabilistic threshold value assigned for classifying the instances. To maximise the value of Recall, the probabilistic threshold value has been set below 0.5, like 0.2 in our case. For illustration, greater than 0.3 is a significant class, and 0.1 is a non-significant class. This has increased the system's Recall and reduced the value of Precision. For Precision, the probabilistic threshold must be set to a higher value like 0.7 in the current model. This increases the Precision value of the model and reduces Recall's value. The Precision-Recall trade-off is fundamental when Recall is very important than Precision and vice versa. The fundamental idea behind the trade-off of Precision and Recall is that the change in threshold value for determining the class cause Recall to increase and Precision to decrease or vice versa. So, the Precision and Recall are computed at each threshold, and out of the results, the graph is plotted for finding the best trade-off point and the same threshold for the model. Figure 5.4 has plotted the graph for different values of Precision and Recall concerning various threshold values. The graph depicts that the best threshold in current research is 0.5 where the Precision and Recall cuts each other. The performance of the classification model is measured based on Precision (Pre), Recall (Rel), and Specificity (Spe) with the probabilistic threshold value of 0.5. Decision Tree (DT) and Support Vector Machine (SVM) are used as baseline classifiers for comparative analysis. The classification model has been implemented using the Python interface. Specifically, BBM is implemented by utilising the PyBBN Python library. Python library sci-kit-learn has been explored for SVM and DT. SVM is a supervised learning algorithm. In the current scenario, SVM with RBF kernel has been used. In the present research, we have explored the RBF (Radial Basis Function) kernel and polynomial Kernel. But in the experiment, the RBF has given more accuracy. Henceforth, we use the RBF kernel for the experiment purpose. Regularization parameter (C) =1.0.

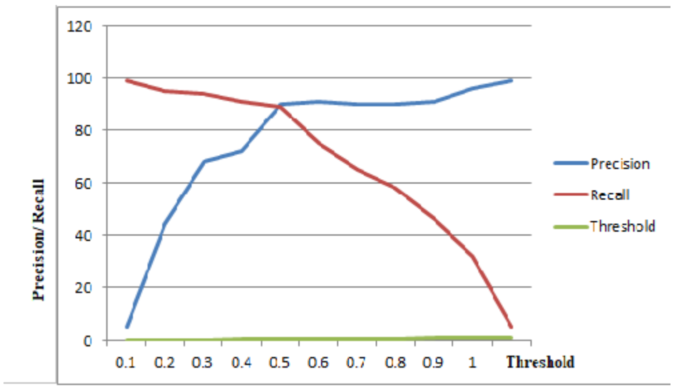


FIGURE 5.4: Recall and Precision values for different threshold

The gamma is the kernel coefficient and it has set as gamma=auto and 1/n features are used instead as the default value. DT has used the default parameters. In which class weight=none, criteria = gini, and splitter=best. Moreover, it is also worth noting that the only classification techniques were changed during implementation, while the rest of the model remained unchanged. The finding of the model for different sets of datasets has been evaluated. Python, the open-source software for performance evaluation, is used for deployment purposes. The results of the Classification efficiency are depicted in Table 5.1. The results are explained as follows:

1. For Precision analysis, the proposed model has registered a higher value of 94.56%. In comparison to this, SVM acquired the Precision value of 92.41%, and DT registered 91.67%. Therefore, it is depicted that the proposed model based on BBM is more efficient than the other classifiers.

2. According to the results, the current model can achieve the average Specificity value of 93.67%. Furthermore, the proposed approach is more efficient than the other classifiers as DT scored 91.08% and SVM attained 90.77%.

3. Recall analysis is a very significant measure of attaining the performance of the proposed model. It is necessary to mention that, in the present scenario, the current model has registered a high value of 94.57% in comparison to 91.56% by SVM and 90.78% by DT. The result is based on the current scenario, which depicts that the proposed model is quite effective and efficient.

4. For F1 score analysis, the proposed model has registered a higher value of 93.52%. In comparison to this, SVM acquired the F1 score value of 92.39%, and DT registered 92.59%. Therefore, it is depicted that the proposed model based on BBM is more efficient than the other classifiers.

The average value of Precision is 94.56%, Specificity is 93.67%, Recall is 94.57% and F1 score is 93.52%. With the threshold value of 0.5, the classification model can achieve good F1 score, Recall, and Precision values for the given dataset. The enhanced values for classification analysis can be deduced that the presented BBM technique incorporates the probabilistic classification technique. This enables effective assessment of NFR values for efficient quantification.

5.1.5 Prediction Efficiency

The Neural Designer Toolkit[1] is used for the deployment of the proposed prediction model. For the experimental implementation, the number of neurons in input layers is set to 8(number of NFRs parameters). The number of hidden layers is given as 3, and each consists of 9 neurons. Finally, the output layer consists of 1 neuron for prediction purposes. The Root Mean Square Error (RMSE) and the Coefficient of Determination(r_2) are used for specifying the association between

[1]Source:https://www.neuraldesigner.com/

the parameters during the experiment performed on the dataset. State-of-the-art prediction models are implemented on the dataset for comparative analysis, including K Nearest Neighbor (KNN), Support Vector Machine (SVM), Artificial Neural Network (ANN), Stochastic Gradient Descent with Regression (SGDR), and Decision Tree Regression (DTR). Moreover, Ensembling techniques, namely Gradient Boosted Decision Trees (GBDT), Adaptive Neuro-Fuzzy System (ANFIS), and Random Forest Regression (RFR), Adaptive Boosting (ADABOOST) are explored. It is significant to mention that the only prediction model is altered during the implementation, and the remaining model is identical. The tables below have shown the prediction efficiency results in terms of Coefficient of Determination, Accuracy, and Root Mean Square Error.

Results

1. The Accuracy of the current technique is quantified in Table 5.2. The results show that the proposed system has acquired a higher Accuracy by 97.04% compared to other Ensembling and baseline techniques with a minor variation in a dataset of 1000 instances. The proposed model has attained an Accuracy of 97.65% for the dataset consisting of 2000 instances greater than the other approaches. Moreover, the proposed approach has reached the Accuracy of 98.89%, 98.89%, and 98.90%, respectively, for the dataset of 3000, 4000, and 5000 instances. The Hybrid M-LSTM and CNN model is extremely accurate for the proposed prediction system based on the current scenario. This significant improvement is achieved due to the hybrid incorporation of the M-DL technique of M-LSTM and CNN. Moreover, it depicts that the variability of the data instances has minimal impact on the Accuracy registered by the presented model.

2. The Coefficient of Determination (r_2) is a significant statistical parameter for depicting the prediction Accuracy (Table 5.3). Mathematically Coefficient

Table 5.2: Results based on Accuracy of Models

Techniques	Methods	Dataset (1000 instances)	Dataset (2000 instances)	Dataset (3000 instances)	Dataset (4000 instances)	Dataset (5000 instances)
Base-line Techniques	SVM	88.89	89.65	92.7	93.5	93.89
	K-NN	88.10	88.82	89.10	89.69	90.63
	ANN	90.52	91.45	93.23	94.01	94.56
	DTR	89.01	89.69	90.75	92.10	93.35
	SGDR	91.98	92.10	94.73	94.69	95.10
Ensembling Techniques	GBDT	96.10	96.89	96.89	97.25	95.23
	RFR	92.70	93.25	95.10	95.56	95.98
	ADABOOST	94.75	94.75	96.84	96.45	96.01
	ANFIS	95.10	96.4	97.6	97.10	97.12
	PROPOSED	**97.04**	**97.65**	**98.89**	**98.89**	**98.90**

Table 5.3: Coefficient of Determination Results(r_2)

Techniques	Methods	Dataset (1000 instances)	Dataset (2000 instances)	Dataset (3000 instances)	Dataset (4000 instances)	Dataset (5000 instances)
Base-line Techniques	SVM	78	84	80	81	79
	K-NN	77	82	79	77	78
	ANN	80	84	80	78	78
	DTR	76	83	77	76	75
	SGDR	81	85	82	80	81
Ensembling Techniques	GBDT	88	91	89	85	83
	RFR	87	89	85	82	81
	ADABOOST	85	90	84	87	83
	ANFIS	85	93	91	89	87
	PROPOSED	**94**	**95**	**93**	**94**	**94**

of Determination(COD) can be written as

$$r^2 = 1 - \frac{SE}{ST}$$

where ST denotes the total sum of squares=S_{xx} and the SE represents the sum of squared residuals=$S_{xx} - S_{yx}^2/S_{yy}$. The values of the parameters included S_{xx}, S_{yx}, and S_{yy}, which are obtained by the testing and prediction phase and implemented over the x-y plane. The experiment has shown that

Table 5.4: Root Mean Square Error (RMSE)Results

Techniques	Methods	Dataset (1000 instances)	Dataset (2000 instances)	Dataset (3000 instances)	Dataset (4000 instances)	Dataset (5000 instances)
Base-line Techniques	SVM	3.5	4.2	3.6	3.7	3.2
	K-NN	4.3	5.7	3.9	3.7	3.8
	ANN	5.6	4.5	3.3	4.5	3.8
	DTR	5.6	2.3	4.7	2.9	4.1
	SGDR	4.1	4.5	3.8	2.7	2.8
Ensembling Techniques	GBDT	4.8	4.8	2.9	2.5	2.3
	RFR	3.9	5.2	2.7	2.2	2.1
	ADABOOST	4.5	3.1	1.9	2.7	2.3
	ANFIS	3.5	2.3	2.1	2.9	1.7
	PROPOSED	**2.4**	**1.9**	**1.1**	**1.5**	**1.8**

the COD measure of the proposed model outperforms other prediction techniques. For illustration, in the dataset of 1000 instances, the proposed model registered the COD value of 94%, which is greater than the other state-of-art approaches. In the same scenario, the COD value for the dataset of 2000, 3000, 4000, and 5000 instances for the prediction model is 95%, 93%, 94%, and 94%, respectively, which is greater than the other comparative techniques. These results have depicted that the proposed system methodology has presented a higher performance in the present scenario of NFRs significance estimation. This shows the effectiveness of utilising the M-DL technique for NFRs prediction.

3. Root Mean Square Error(RMSE) is another significant statistical measure for the performance of the prediction system. It represents the error in the prediction value of the proposed model. Table 5.4 depicts the outcome of the model in the form of RMSE. It is represented mathematically as RMSE $= \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\left(\frac{d_i - f_i}{\sigma_i}\right)^2}$ where σ_i signifies the error acquired in the training phase for the ith instance and n represents the number of instances. The RMSE value for a dataset of 1000, 2000, 3000, 4000, and 5000 instances

are registered 2.4%, 1.9%, 1.1%, 1.5%, and 1.8% respectively. Henceforth, it is registered that the prediction model is comparatively better for NFRs estimation as the minimal error is registered.

5.1.6 Reliability Analysis

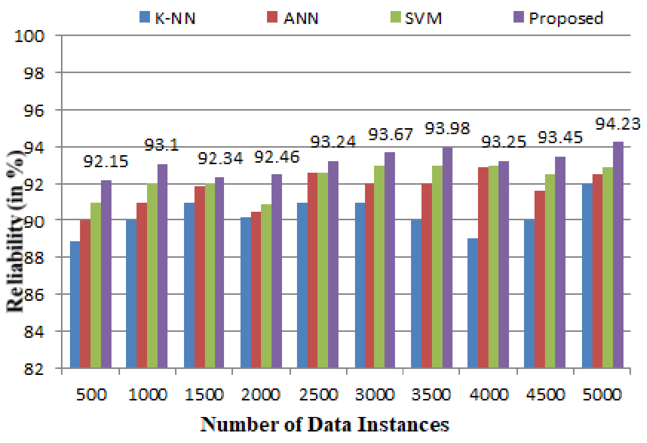


FIGURE 5.5: Reliability Analysis of Overall System

Reliability defines the ability of the software system to perform in unexpected circumstances. Reliability in terms of abnormality is an important metric for accessing the reliability of the proposed methodology. It can be evaluated by positive as well as negative performance indicators including accuracy and error respectively. Reliability can be estimated in different ways like model-independent or model-dependent reliability [280]. The model's independent reliability is estimated by changing the learning set. The model-independent reliability estimates for individual predictions, which are implemented as estimates of the prediction error as a negative performance indicator or Accuracy as a positive performance indicator [87]. These estimates are defined as metrics, of which higher values represent higher accuracy of the model and vice versa. The high value of reliability depicts that the prediction model is better to adhere to accuracy [281]. In the present scenario, the accuracy metric is explored to evaluate the reliability of the system. The experimental implementation was performed with 10% abnormal data values to compare the results for reliability among different prediction models. Figure 5.5 above indicates the reliability results of the deployments. The average reliability of the model is reported for comparative analysis with current prediction models such as SVM, ANN, and K-Nearest Neighbor(K-NN). The experiment system has depicted that with the increase in the dataset, the trends in reliability confirm the higher values for the present system compared to other prediction techniques. Results illustrate that the given model is better than other prediction techniques if the following data instances are acquired. The proposed model has a reliability score of 94.78% which is higher than the 90.50%, 91.35%, and 93.05%, respectively, for K-NN, ANN, and SVM prediction techniques. It is concluded from the findings that the proposed prediction model is highly accurate over the large dataset in comparison to other prediction models.

5.1.7 Stability Analysis

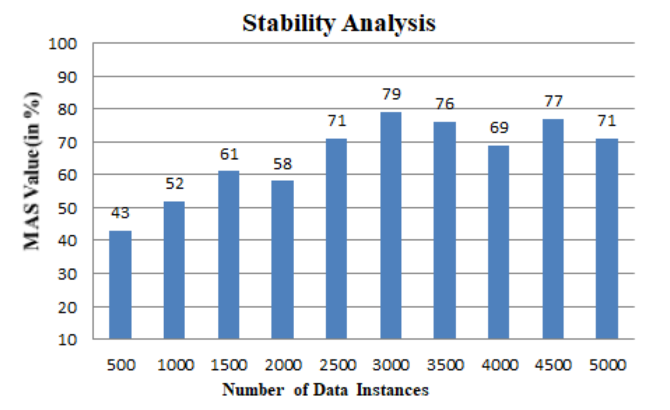


FIGURE 5.6: Stability Analysis Overall System

Stability represents the degree of inconsistency between different predictions made by the algorithm [282]. Stability is an important measure to determine the performance of the prediction system [283]. The stability of the prediction system influences the users trust in the prediction system. A prediction system is stable if the predictions it provides do not change strongly over a short range of data[85]. It evaluates and predicts the total stabilization of the model when it is implemented over a large dataset [284]. System stability can be measured with MAS(Mean Absolute Shift). Stability analysis has been utilised to estimate the prediction system for normalising behavior over large data instances. Specifically, it evaluates and predicts the model's total stabilisation when implemented over a large dataset. The Mean Absolute Shift(MAS) metric has been utilised for stability analysis in the proposed system. The range is given from 0% to 100%. The 0% depicts the minimum stability, and 100% acquires the maximal stability value for MAS. Figure 5.6 shows the outcomes of stability analysis for the predicted system. The findings depict 43% as the minimum value of MAS and 79% as the maximum value in MAS, whereas the average value is 70% over 5000 data instances. The results have shown that the proposed system is beneficial and stable for NFRs prediction.

5.1.8 Discussion

The proposed system has considered numerous aspects like Data Completeness, Efficiency, Stability Analysis, and Reliability. Conspicuously, some of the key aspects are discussed ahead.

1. The average level of data completeness of the questionnaire was 87%, which is significant in the current research domain.
2. The presented model efficiently and effectively performs the classification with 94.56% Precision, 93.67% Specificity, 93.52% F1 score, and 94.57% Recall depicting enhanced efficacy.
3. The prediction efficiency is given in terms of Accuracy, Root Mean Square Error, and Coefficient of Determination. The current approach has attained an average Accuracy of 97.04%, which is higher than the other Baseline and Ensembling techniques. The average Coefficient of Determination measures

94% for the proposed model and outperformed other techniques. The current scenario shows the least Root Mean Square error than other techniques.

4. The given model has a 94.78% reliability score which is higher than other techniques. With the increase of the dataset, the reliability score also improves, making it suitable for large datasets.

5. The proposed system is extremely stable for NFRs classification. The average value of the Mean Absolute Shift (MAS) measure used for stability analysis is 70% over 5000 data instances. The results have shown 43% as a minimum and 79% as the maximum MAS value for the presented system.

5.2 Validation of Model

The current section discusses the experimental evaluation of the proposed model on the publically available dataset. The various performance indicators are estimated for the validation purpose of the proposed prediction model. Assessing the classification efficacy of the proposed model. Estimating the predictive efficacy of the present prediction model.

5.2.1 Dataset

The dataset is based on NFRs such as information security requirements. The NFRs such as availability, correctness, and confidentiality are considered in this dataset. The various levels, such as 1, 2, and 3, signify the level of damage if any of the said NFRs are not maintained. For illustration

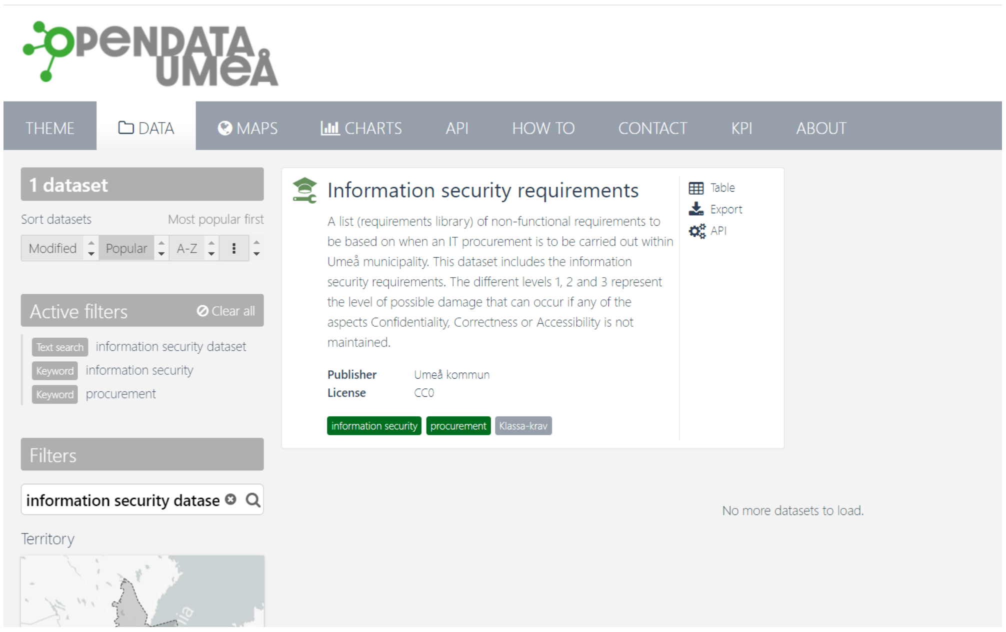
1. Level 1: Moderate injury

2. Level 2: Significant damage

3. Level 3: Serious injury

Level 1 signifies the moderate injury level and tells that it is sufficient if one of the following conditions is met: None of the participants saw significant obstacles in attaining the goals. None of them impacts critical functions at your company or elsewhere. Individuals, governments, and organisations may notice the disruption or feel a little discomfort, but there is no evidence of a financial impact. Isolated, non-sensitive personal data may be disseminated, posing a mild risk to privacy.
Level 2 represents the significant damage and states that it is sufficient if one of the following conditions is met: Activities can achieve their tasks, but there's a chance they'll have a negative impact Whether it's due to a lack of resources or the requirement to take extreme measures). Other government agencies and organisations may be impacted (the necessity to take exceptional steps, either monetarily or by the need to take extraordinary measures). It is unlikely that essential functions of your own or another organisation would be harmed. Individuals may be affected by the disruption, including considerable inconveniences or a significant economic effect. Personal information may be disseminated, posing a severe threat to privacy.
Level 3 represents severe injury. It states that one of the following conditions must be met: The organisation's activities are severely hampered. Completing the task is impossible or nearly impossible. Essential functions of your own or another company are likely to be impacted. The lives and health of individuals are affected. Sensitive personal data can be extensively disseminated, posing a significant threat to privacy.
The dataset for the experimental purpose has been taken from the Open Data Umea.[2] as shown in Figure 5.7.

[2]http://data.europa.eu/88u/dataset/https-opendata-umea-se-api-v2-catalog-datasets-informationssakerhetskrav



Text search information security dataset
Keyword information security
Keyword procurement

Information security requirements

A list (requirements library) of non-functional requirements to be based on when an IT procurement is to be carried out within Umeå municipality. This dataset includes the information security requirements. The different levels 1, 2 and 3 represent the level of possible damage that can occur if any of the aspects Confidentiality, Correctness or Accessibility is not maintained.

Publisher	Umeå kommun
License	CC0

information security procurement Klassa-krav

FIGURE 5.7: Open Data Umea

5.2.2 Experimental configuration

The simulation of the current model has been carried out on the publicly available dataset. The information-based security dataset has been acquired from the Open data Umea corpus(source: https://opendata.umea.se/.) Implementation was carried out using a computer with an Intel Core i7 processor, 4.70 GHz clock cycle, and 32 GB Ram configuration. The Synthetic Minority Over-Sampling Technique (SMOTE) is used as a data balancing and augmentation technique.

5.2.3 Classification Efficiency

The proposed model's classification efficiency is determined by estimating three key performance parameters: Precision (Pr), Recall (Rel), and Specificity (Spe). As a baseline classifier, Three classification algorithms, Bayesian Belief Method (BBM), Decision Tree (DT), and Support Vector Machine (SVM) are explored. However, it is important to note that the machine learning algorithms are changed during implementation and the rest of the model remains unchanged. The performance evaluation of the classification model is presented in Table 5.5. The results are explained as follows :

TABLE 5.5: Classification Efficiency in terms of Precision(Pre), Recall(Rel), Specificity(Spe)

Model	BBM			DT			SVM		
Dataset	Pr(%)	Sp(%)	Rel(%)	Pr(%)	Sp(%)	Rel(%)	Pr(%)	Sp(%)	Rel(%)
500	96.42	94.34	94.42	92.42	91.34	92.82	90.62	91.74	91.42
1000	94.78	95.79	93.21	94.78	93.79	91.57	91.83	90.89	92.83
1500	95.65	93.88	94.72	93.65	93.88	92.12	92.69	91.98	92.75
2000	95.15	94.34	92.82	93.15	92.94	92.22	93.19	92.34	93.82
2500	93.27	93.78	93.34	93.27	91.78	90.39	91.57	91.78	92.64
300	94.57	93.78	93.34	92.57	92.78	92.84	92.87	92.75	91.94
3500	95.21	94.43	92.21	91.78	93.73	91.21	91.56	91.43	90.66
4000	94.57	94.34	93.42	92.57	92.39	93.42	92.57	92.34	93.82
4500	95.74	94.89	94.25	93.74	92.04	91.07	92.68	91.14	92.42
5000	95.87	94.76	94.65	92.74	93.58	91.92	92.80	92.89	92.25

1. The proposed model has attained an average Precision value is 95.84 % for the considered dataset. In comparison, DT has a Precision value of 92.07 %, while SVM has 92.12 %. The suggested model based on BBM is shown to be more efficient than the other classifiers.

2. For the dataset in consideration, the proposed model is capable of obtaining an average Specificity value of 94.96% in comparison to 93.24% by DT and 92.56% by SVM. It is indicating that the proposed is considerably more efficient than the other classifiers.

3. Recall analysis is a critical component for evaluating the performance of the current model. It is worth noting that the current model has a high value of 94.78 %, compared to 92.03 % for DT and 93.05 % for SVM. The outcome shows that the current model is very efficient based on the results attained with the considered dataset.

5.2.4 Prediction Efficiency

The prediction efficacy of the proposed model is measured for the considered dataset. The current model is compared to various state-of-the-art baseline models, namely SVM, K-NN, ANN, DTR, SGDR, and ensembling models such as GBDT, RFR, ADABOOST, ANFIS. It is significant to mention that the only prediction model is altered during the implementation, and the remaining model is identical. The outcomes of the prediction model describes in terms of Accuracy, Coefficient of Determination, and Root Mean Square Error is registered in Table 5.6, Table 5.7, and Table 5.8. The results are explained ahead:

TABLE 5.6: Results of the Model in terms of Accuracy Metric

Techniques	Methods	Dataset (1000 instances)	Dataset (2000 instances)	Dataset (3000 instances)	Dataset (4000 instances)	Dataset (5000 instances)
Base-line Techniques	SVM	87.81	88.56	89.79	89.52	91.89
	K-NN	87.89	88.98	90.10	90.69	91.63
	ANN	91.23	92.65	92.64	93.12	93.16
	DTR	90.12	90.49	90.77	91.19	92.56
	SGDR	91.01	91.89	92.89	93.74	94.25
Ensembling Techniques	GBDT	95.16	94.85	94.91	93.51	94.63
	RFR	92.77	92.53	94.14	94.16	94.81
	ADABOOST	93.55	93.74	94.81	94.58	94.09
	ANFIS	94.52	94.47	95.86	95.10	95.12
	PROPOSED	**95.14**	**95.07**	**96.23**	**96.85**	**96.56**

1. Table 5.6 presents the efficacy of the proposed model in comparison to other ML techniques. From these findings, it can be seen that the average Accuracy value for the proposed model is (96.38%) over various numbers of instances. When compared to the current framework, several prediction models have a low Accuracy value. In particular, SVM has an average Accuracy of (89.57%) whereas ADABOOST and ANFIS have an average Accuracy of (94.82%) and (95.18%) respectively. Henceforth, the proposed system has the highest value of Accuracy in comparison to the other models and is therefore considerably more efficient. Moreover, it is concluded that the Hybrid M-LSTM and CNN

model is extremely accurate for the proposed prediction system based on the current scenario.

2. The Coefficient of Determination (r2) is a significant statistical parameter for depicting the prediction accuracy presents in Table 5.7. The experiment demonstrated that the presented model COD measure outperforms various prediction methods. As an instance, the suggested model registered a COD value of 92% in the dataset of 1000 instances, which is higher than the other state-of-the-art techniques. In the same scenario, the prediction model COD value for the dataset of 2000, 3000, 4000, and 5000 instances is 93%, 92%, 93%, and 93%, respectively, which is higher than the COD values obtained by the other comparison methods. These findings demonstrated that the proposed approach performed better than other techniques in the present scenario.

3. The another important statistical measure for evaluating the performance of the prediction model is Root Mean Square Error (RMSE). It represents the error in the prediction value of the proposed model. Table 5.8 depicts the outcome of the model in the form of RMSE. The RMSE value for a dataset of 1000, 2000, 3000, 4000, and 5000 instances are registered 3.1%, 2.9%, 2.1%, 2.4%, and 2.7% respectively. Henceforth, it is registered that the prediction model is comparatively better for the present scenario as a minimal error is reported.

5.2.5 Reliability Analysis

The results of the reliability efficiency are shown in Figure 5.8 for the current dataset. The average reliability results of the current model are compared with other prediction techniques like SVM, ANN, and K-NN. The experiment system showed that as the dataset grew, reliability trends confirmed the higher values for

Table 5.7: Coefficient of Determination Results(r_2)

Techniques	Methods	Dataset (1000 instances)	Dataset (2000 instances)	Dataset (3000 instances)	Dataset (4000 instances)	Dataset (5000 instances)
Base-line Techniques	SVM	74	82	79	81	80
	K-NN	76	78	77	79	78
	ANN	82	80	83	81	80
	DTR	76	81	81	82	83
	SGDR	85	87	88	89	88
Ensembling Techniques	GBDT	88	91	89	85	83
	RFR	88	89	85	81	80
	ADABOOST	84	91	90	87	89
	ANFIS	86	92	90	88	89
	PROPOSED	**92**	**93**	**92**	**93**	**93**

Table 5.8: Root Mean Square Error (RMSE)Results

Techniques	Methods	Dataset (1000 instances)	Dataset (2000 instances)	Dataset (3000 instances)	Dataset (4000 instances)	Dataset (5000 instances)
Base-line Techniques	SVM	4.3	4.2	3.9	3.8	4.1
	K-NN	4.7	4.9	4.9	5.7	5.8
	ANN	6.6	5.5	4.7	4.9	5.8
	DTR	5.2	5.3	4.7	4.9	5.1
	SGDR	4.2	4.5	4.8	4.7	4.8
Ensembling Techniques	GBDT	4.8	4.8	2.9	2.5	2.3
	RFR	3.9	5.3	5.7	5.2	5.1
	ADABOOST	4.2	4.1	5.9	5.7	5.3
	ANFIS	5.5	5.3	4.7	4.9	4.7
	PROPOSED	**3.1**	**2.9**	**2.1**	**2.4**	**2.7**

the current system when compared to other prediction techniques. Specifically, the average reliability score of the present model is 93.57% which is comparatively better than the 89.35% by SVM, 91.29% by ANN, and 90.89% by K-NN. Moreover, In the current scenario, the results describe that the present model is highly accurate over the large dataset in comparison to other prediction techniques. The experiment system showed that as the dataset grew, reliability trends confirmed the higher values for the current system when compared to other prediction techniques. Specifically, the average reliability score of the present model is 93.57%

which is comparatively better than the 89.35% by SVM, 91.29% by ANN, and 90.89% by K-NN. Moreover, In the current scenario, the results describe that the present model is highly accurate over the large dataset in comparison to other prediction techniques.

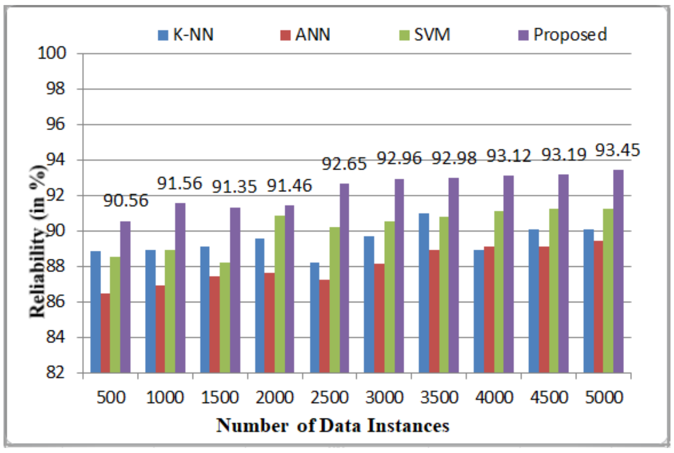


FIGURE 5.8: Reliability Analysis of Overall System

5.2.6 Stability Analysis

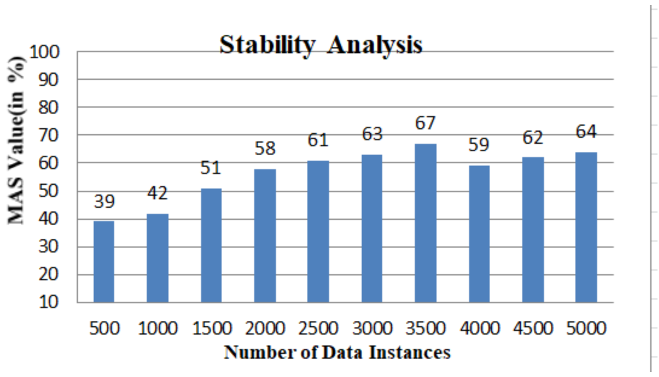


FIGURE 5.9: Stability Analysis Overall System

The prediction model for normalising behaviour over large data instances is estimated using stability analysis. In particular, it assesses and forecasts the model overall stabilisation when applied to a large dataset. The proposed system stability analysis has made use of the Mean Absolute Shift (MAS) measure. The range is presented as 0% to 100%. For MAS, 0 percent represents the minimal stability and 100 percent represents the maximum stability. The results of the stability study for the proposed system are presented in Figure 5.9. According to the results, the minimum and maximum values for MAS are respectively 39% and 67% with an average of 53% across 5000 data points. The outcomes showed the efficacy of the proposed model in the context of stability analysis for the present dataset.

5.3 Discussion and Future Scope

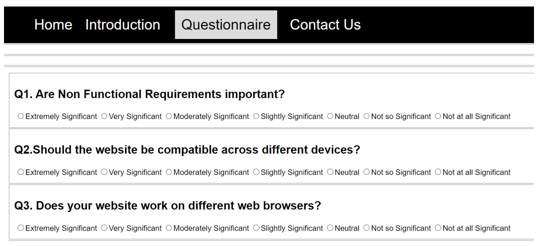
In the proposed research a website [3] is designed for NFRs elicitation. It consists of various questions on usability-based NFRs. The responses are collected based on the Likert scale. Overall the questionnaire consists of 44 questions. The range of questions tries to explore all the aspects of usability like learnability, understandability, aesthetics, accessibility, and operability to mention a few. A few glimpses of the website are given in Figure 5.10.

5.3.1 NFRs Elicitation oriented Future scope

A tool can be designed which considers all the aspects of NFRs to elicit them appropriately. The recommendation system thus designed may help the software developers in the context of NFRs.

1. The recommendation system may be designed for the NFRs elicitation system. The recommendation system will provide value. For illustration, if the value is between 0- 25% represents the NFRs consideration is very low, if the value is between 25-50%, NFRs consideration is average. If the value is

[3]http://www.nfrs-questionnaire.somee.com/WebForm1.aspx



Q1. Are Non Functional Requirements important?

○ Extremely Significant ○ Very Significant ○ Moderately Significant ○ Slightly Significant ○ Neutral ○ Not so Significant ○ Not at all Significant

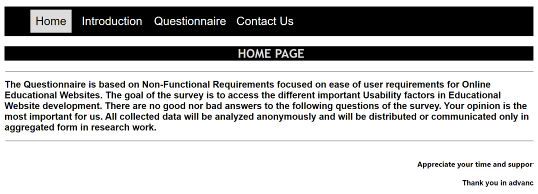
Q2.Should the website be compatible across different devices?

○ Extremely Significant ○ Very Significant ○ Moderately Significant ○ Slightly Significant ○ Neutral ○ Not so Significant ○ Not at all Significant

Q3. Does your website work on different web browsers?

○ Extremely Significant ○ Very Significant ○ Moderately Significant ○ Slightly Significant ○ Neutral ○ Not so Significant ○ Not at all Significant

(a)

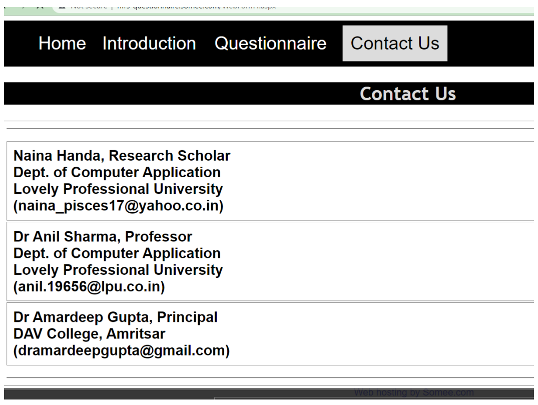


The Questionnaire is based on Non-Functional Requirements focused on ease of user requirements for Online Educational Websites. The goal of the survey is to access the different important Usability factors in Educational Website development. There are no good nor bad answers to the following questions of the survey. Your opinion is the most important for us. All collected data will be analyzed anonymously and will be distributed or communicated only in aggregated form in research work.

Appreciate your time and suppor

Thank you in advanc

(b)



Naina Handa, Research Scholar
Dept. of Computer Application
Lovely Professional University
(naina_pisces17@yahoo.co.in)

Dr Anil Sharma, Professor
Dept. of Computer Application
Lovely Professional University
(anil.19656@lpu.co.in)

Dr Amardeep Gupta, Principal
DAV College, Amritsar
(dramardeepgupta@gmail.com)

(c)

FIGURE 5.10: Glimpses of Website

between 50-75% good in the context of NFRs consideration. If the value is more than 75% represents the maximum NFRs considered for the system.

2. The prioritization of NFRs is very important to consider. Moreover, the prioritization system for NFRs may be designed in the future to assist the software developer. Developers can take help from the system and consider the NFRs according to their significance.

3. The NFRs elicitation website may be augmented with questionnaires related to other factors of ISO/IEC 25010. For illustration, the questionnaires on portability, compatibility, interoperability, and maintainability will be added to the website.

Chapter 6

Conclusion

6.1 Concluding Remarks

Software requirements are segregated as FRs and NFRs. FRs refer to what is to be done whereas NFRs deal with how is to be done. Numerous examples of project failure are mentioned in the literature due to inappropriate management NFRs. Moreover, It is very important to consider the NFRs during the initial phases of software development. The current study presents an automated system for efficient NFRs prediction. The proposed system consists of 4 layers for the prediction of NFRs significance, including the Data Acquisition (DA) layer, Feature Selection and Extraction (FSE) layer, Data Prediction(DP) layer, and Data Analysis and Visualization (DAV) layer. Moreover, the current research considers the probability measure of the level of NFR significance in terms of LoNFRS, which is cumulatively quantified as the NFRs significance measure (NFRINDEX). NFRINDEX has been quantified for prediction purposes using a Convolution Neural Network (CNN) and M-LSTM. A Self Organized Mapping (SOM) procedure is also used to visualize the existence of NFRs. A primary dataset is collected from several IT professionals and academicians to validate the proposed system.

The NFRs dataset of 312 IT professionals and academicians has been gathered and resulting in 5304 instances. Moreover, the secondary dataset from open-data umea is also considered for validation purposes. A hybrid feature selection algorithm consisting of a filter and wrapper based methods is designed for classification purposes. The filter based techniques namely Information Gain and Correlation are explored. The wrapper method based on backward selection is Explored on algorithms namely the Bayesian Belief Method, K-Nearest Neighbor, Support Vector Machine, and Decision Tree. The Bayesian Belief outperforms in terms of Accuracy. The Neural Designer Toolkit is used for the deployment of the proposed prediction model. A Multi-scaled Long Short Term Memory (M-LSTM) based prediction model has been designed for the NFRs prediction, Which is a combination of Convolutional Neural Network and Long Short Term Memory. Numerous experiments are performed to assess the performance of the present model in terms of Classification, Root Mean Square Error, Coefficient of Determination (r_2), Reliability, Stability, Feature Selection, and Prediction Efficiency. The Root Mean Square Error and the Coefficient of Determination (r_2) are used for specifying the association between the parameters during the experiment performed on the dataset. State-of-the-art prediction models are implemented on the dataset for comparative analysis, including K Nearest Neighbor, Support Vector Machine, Artificial Neural Network, Stochastic Gradient Descent with Regression, and Decision Tree Regression. Moreover, Ensembling techniques, namely Gradient Boosted Decision Trees, Adaptive Neuro-Fuzzy System, and Random Forest Regression, Adaptive Boosting are explored. It is significant to mention that the only prediction model is altered during the implementation, and the remaining model is identical. In the present research, the average level of data completeness of the questionnaire was 87%, which is significant in the current research domain. The presented model efficiently and effectively performs the classification with 94.56% Precision, 93.67% Specificity, 93.52% F1 score, and 94.57% Recall depicting enhanced efficacy. The prediction efficiency is given in terms of Accuracy, Coefficient of Determination, and Root Mean Square Error. The current approach has attained an average accuracy of 97.04%, which is higher than the other Baseline and Ensembling techniques. The Coefficient of Determination value for the dataset of 1000, 2000, 3000, 4000, and 5000 instances for the prediction model is 94%, 95%, 93%, 94%, and 94%, respectively, which is greater than the other comparative techniques. The Root Mean Square Error value for the dataset of 1000, 2000, 3000, 4000, and 5000 instances are registered 2.4%, 1.9%, 1.1%, 1.5%, and 1.8% respectively which are comparatively lesser than other techniques. The proposed model has an average reliability score of 94.78% which is higher than the 90.50%, 91.35%, and 93.05%, respectively, for K-NN, ANN, and SVM prediction techniques. With the increase of the dataset, the reliability score also improves, making it suitable for large datasets. The proposed system is highly stable for NFRs classification. The average value of the Mean Absolute Shift (MAS) measure used for stability analysis is 70% over 5000 data instances. The results have shown 43% as a minimum and 79% as the maximum MAS value for the presented system. Based on the findings, it is concluded that the proposed system outperforms other models in terms of Classification, Data Completeness, Prediction, Stability, and Reliability. Moreover, it is concluded from the results that the current classification and prediction technique is efficient for NFRs prediction.

6.2 Future Directions

The various research works have discussed the NFRs prediction and classification. But still, there is a scope for improvement in the current research field. The different significant future directions are discussed in the following section:

1. A recommendation system for software development can be designed. This system helps the software developers and other IT personnel recommend the NFRs for a software system.

2. The Prioritization of NFRs may be a significant area for exploration. It is a very significant field to prioritize the requirements even after classification. The crucial requirements can be focused on and analyzed according to their priority.

3. The algorithms used in the present paper result from an extensive literature review, and these can be further extended. In addition, novel algorithms can be added, and comparative analysis of the algorithms can open up other vistas of research.

4. The attributes focused on by the current research are generic attributes. In addition, software-oriented attributes can be added and enhanced in the catalog. This will help in enhancing the NFRs prediction.

5. The Usability parameters of ISO/IEC 25010 are considered in the present research. In addition, other parameters of ISO/IEC 25010 can be considered in future research exploration.

6. The implementation of the Self Organized Mapping over different platforms will be considered.

7. The proposed ML model will be explored on available NFRs based datasets.

REFERENCES

[1] B. Boehm and H. In, “Identifying quality-requirement conflicts,” *IEEE software*, vol. 13, no. 2, pp. 25–35, 1996.

[2] I. Sommerville and P. Sawyer, *Requirements engineering: a good practice guide.* John Wiley & Sons, Inc., 1997.

[3] D. T. Ross and K. E. Schoman, “Structured analysis for requirements definition,” *IEEE transactions on Software Engineering*, no. 1, pp. 6–15, 1977.

[4] D. R. Lindstrom, “Five ways to destroy a development project (software development),” *IEEE Software*, vol. 10, no. 5, pp. 55–58, 1993.

[5] S. Kopczyńska, J. Nawrocki, and M. Ochodek, “An empirical study on catalog of non-functional requirement templates: Usefulness and maintenance issues,” *Information and Software Technology*, vol. 103, pp. 75–91, 2018.

[6] K. K. Breitman, J. C. S. Leite, and A. Finkelstein, “The world sa stage: a survey on requirements engineering using a real-life case study,” *Journal of the Brazilian Computer Society*, vol. 6, no. 1, pp. 13–37, 1999.

[7] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, *Non-functional requirements in software engineering.* Springer Science & Business Media, 2012, vol. 5.

[8] A. Vogelsang and M. Borg, “Requirements engineering for machine learning: Perspectives from data scientists,” in *2019 IEEE 27th International*

Requirements Engineering Conference Workshops (REW). IEEE, 2019, pp. 245–251.

[9] K. Pohl, *Requirements engineering: An overview.* RWTH, Fachgruppe Informatik Aachen, 1996.

[10] A. Van Lamsweerde, “Goal-oriented requirements engineering: A guided tour,” in *Proceedings fifth ieee international symposium on requirements engineering.* IEEE, 2001, pp. 249–262.

[11] E. Hull, “K. jackson och j. dick,” *Requirements engineering, London: Springer*, 2011.

[12] M. Glinz, “Rethinking the notion of non-functional requirements,” in *Proc. Third World Congress for Software Quality*, vol. 2, 2005, pp. 55–64.

[13] D. Mairiza, D. Zowghi, and N. Nurmuliani, “An investigation into the notion of non-functional requirements,” in *Proceedings of the 2010 ACM symposium on applied computing*, 2010, pp. 311–317.

[14] I. S. C. Committee *et al.*, “Ieee standard glossary of software engineering terminology (ieee std 610.12-1990). los alamitos,” *CA: IEEE Computer Society*, vol. 169, p. 132, 1990.

[15] D. Arruda, “Requirements engineering in the context of big data applications,” *ACM SIGSOFT Software Engineering Notes*, vol. 43, no. 1, pp. 1–6, 2018.

[16] L. A. Macaulay, *Requirements engineering.* Springer Science & Business Media, 2012.

[17] K. Pohl, *Requirements engineering fundamentals: a study guide for the certified professional for requirements engineering exam-foundation level-IREB compliant.* Rocky Nook, Inc., 2016.

[18] L. M. Cysneiros and J. C. S. do Prado Leite, "Nonfunctional requirements: From elicitation to conceptual models," *IEEE transactions on Software engineering*, vol. 30, no. 5, pp. 328–350, 2004.

[19] S. Devata and A. Olmsted, "Modeling non-functional requirements in cloud hosted application software engineering," *CLOUD COMPUTING 2016*, p. 59, 2016.

[20] A. Olmsted, "Secure software development through non-functional requirements modeling," in *2016 international conference on information society (i-Society)*. IEEE, 2016, pp. 22–27.

[21] G. Robiolo, E. Scott, S. Matalonga, and M. Felderer, "Technical debt and waste in non-functional requirements documentation: An exploratory study," in *International Conference on Product-Focused Software Process Improvement*. Springer, 2019, pp. 220–235.

[22] S. Ullah, M. Iqbal, and A. M. Khan, "A survey on issues in non-functional requirements elicitation," in *International Conference on Computer Networks and Information Technology*. IEEE, 2011, pp. 333–340.

[23] I. A. O. Ahmed and M. E. E. Daleel, "Automated use case diagram generation with non-functional requirements using neural network."

[24] D. Ameller, C. Ayala, J. Cabot, and X. Franch, "How do software architects consider non-functional requirements: An exploratory study," in *2012 20th IEEE international requirements engineering conference (RE)*. IEEE, 2012, pp. 41–50.

[25] M. Younas, D. Jawawi, I. Ghani, and R. Kazmi, "Non-functional requirements elicitation guideline for agile methods," 2017.

[26] A. M. Davis, *Software requirements: objects, functions, and states*. Prentice-Hall, Inc., 1993.

[27] G. Kotonya and I. Sommerville, *Requirements engineering: processes and techniques.* John Wiley & Sons, Inc., 1998.

[28] A. Van Lamsweerde, *Requirements engineering: From system goals to UML models to software.* Chichester, UK: John Wiley & Sons, 2009, vol. 10.

[29] I. Jacobson, G. Booch, and J. Rumbaugh, “The unified process,” *IEEE software*, vol. 16, no. 3, p. 96, 1999.

[30] C. Ncube, “A requirements engineering method for cots-based systems development,” Ph.D. dissertation, City University London, 2000.

[31] S. Robertson and J. Robertson, *Mastering the requirements process: Getting requirements right.* Addison-wesley, 2012.

[32] M. Glinz, “On non-functional requirements,” in *15th IEEE International Requirements Engineering Conference (RE 2007).* IEEE, 2007, pp. 21–26.

[33] K. Wiegers and J. Beatty, *Software requirements.* Pearson Education, 2013.

[34] W.-T. Tsai, Z. Jin, P. Wang, and B. Wu, “Requirement engineering in service-oriented system engineering,” in *IEEE International Conference on e-Business Engineering (ICEBE’07).* IEEE, 2007, pp. 661–668.

[35] R. E. Miller, *The Quest for Software Requirements.* Unspecified, 2009.

[36] R. R. Young, *The requirements engineering handbook.* Artech House, 2004.

[37] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, “The detection and classification of non-functional requirements with application to early aspects,” in *14th IEEE International Requirements Engineering Conference (RE’06).* IEEE, 2006, pp. 39–48.

[38] J. Mylopoulos, L. Chung, and B. Nixon, “Representing and using nonfunctional requirements: A process-oriented approach,” *IEEE Transactions on software engineering*, vol. 18, no. 6, pp. 483–497, 1992.

[39] O. Gordieiev, V. Kharchenko, N. Fominykh, and V. Sklyar, "Evolution of software quality models in context of the standard iso 25010," in *Proceedings of the Ninth International Conference on Dependability and Complex Systems DepCoS-RELCOMEX. June 30–July 4, 2014, Brunów, Poland.* Springer, 2014, pp. 223–232.

[40] S. Ouhbi, A. Idri, J. L. F. Alemán, A. Toval, and H. Benjelloun, "Applying iso/iec 25010 on mobile personal health records." in *HEALTHINF*, 2015, pp. 405–412.

[41] N. Shiratuddin *et al.*, "Evaluation of e-book applications using iso 25010," in *2015 International Symposium on Technology Management and Emerging Technologies (ISTMET).* IEEE, 2015, pp. 114–118.

[42] N. Handa, A. Sharma, and A. Gupta, "An inclusive study of several machine learning based non-functional requirements prediction techniques," in *International Conference on Futuristic Trends in Networks and Computing Technologies.* Springer, 2019, pp. 482–493.

[43] M. Haoues, A. Sellami, H. Ben-Abdallah, and L. Cheikhi, "A guideline for software architecture selection based on iso 25010 quality related characteristics," *International Journal of System Assurance Engineering and Management*, vol. 8, no. 2, pp. 886–909, 2017.

[44] F. B. A. Ramos, A. Pedro, M. Cesar, A. A. M. Costa, M. B. Perkusich, H. O. de Almeida, and A. Perkusich, "Evaluating software developers' acceptance of a tool for supporting agile non-functional requirement elicitation." in *SEKE*, 2019, pp. 26–42.

[45] R. R. Maiti and F. J. Mitropoulos, "Capturing, eliciting, predicting and prioritizing (cepp) non-functional requirements metadata during the early stages of agile software development," in *SoutheastCon 2015.* IEEE, 2015, pp. 1–8.

[46] A. Borg, A. Yong, P. Carlshamre, and K. Sandahl, "The bad conscience of requirements engineering: an investigation in real-world treatment of non-functional requirements," 2003.

[47] F. Fellir, K. Nafil, and R. Touahni, "Analyzing the non-functional requirements to improve accuracy of software effort estimation through case based reasoning," in *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA).* IEEE, 2015, pp. 1–6.

[48] M. Umar and N. A. Khan, "Analyzing non-functional requirements (nfrs) for software development," in *2011 IEEE 2nd International Conference on Software Engineering and Service Science.* IEEE, 2011, pp. 675–678.

[49] S. Kopczyńska, M. Ochodek, and J. Nawrocki, "On importance of non-functional requirements in agile software projectsa survey," in *Integrating Research and Practice in Software Engineering.* Springer, 2020, pp. 145–158.

[50] N. Martens, "The impact of non-functional requirements on project success," *Utrecht University, Msc Thesis, Utrecht*, 2011.

[51] M. I. Babar, M. Ghazali, and D. N. Jawawi, "Systematic reviews in requirements engineering: A systematic review," in *2014 8th. Malaysian Software Engineering Conference (MySEC).* IEEE, 2014, pp. 43–48.

[52] S. Alam, S. A. A. Shah, S. N. Bhatti, and A. M. Jadi, "Impact and challenges of requirement engineering in agile methodologies: A systematic review," 2017.

[53] A. M. Davis and D. A. Leffingwell, "Using requirements management to speed delivery of higher quality applications," *Rational Software Corporation*, vol. 20, p. 2004, 1996.

[54] M. Dabbagh and S. P. Lee, "An approach for prioritizing nfrs according to their relationship with frs," *Lecture Notes on Software Engineering*, vol. 3, no. 1, pp. 1–5, 2015.

[55] M. Binkhonain and L. Zhao, "A review of machine learning algorithms for identification and classification of non-functional requirements," *Expert Systems with Applications: X*, vol. 1, p. 100001, 2019.

[56] R. Jindal, R. Malhotra, and A. Jain, "Automated classification of security requirements," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2016, pp. 2027–2033.

[57] M. S. Kumar, A. Harika, C. Sushama, and P. Neelima, "Automated extraction of non-functional requirements from text files: A supervised learning approach," *Handbook of Intelligent Computing and Optimization for Sustainable Development*, pp. 149–170, 2022.

[58] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, "Automated classification of non-functional requirements," *Requirements engineering*, vol. 12, no. 2, pp. 103–120, 2007.

[59] R. Jindal, R. Malhotra, A. Jain, and A. Bansal, "Mining non-functional requirements using machine learning techniques," *e-Informatica Software Engineering Journal*, vol. 15, no. 1, pp. 85–114, 2021.

[60] R. Malhotra, A. Chug, A. Hayrapetian, and R. Raje, "Analyzing and evaluating security features in software requirements," in *2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH)*. IEEE, 2016, pp. 26–30.

[61] P. Singh, D. Singh, and A. Sharma, "Rule-based system for automated classification of non-functional requirements from requirement specifications," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2016, pp. 620–626.

[62] H. Lu, J. Chen, K. Yan, Q. Jin, Y. Xue, and Z. Gao, "A hybrid feature selection algorithm for gene expression data classification," *Neurocomputing*, vol. 256, pp. 56–62, 2017.

[63] H. W. Park, D. Li, Y. Piao, and K. H. Ryu, "A hybrid feature selection method to classification and its application in hypertension diagnosis," in *International Conference on Information Technology in Bio-and Medical Informatics.* Springer, 2017, pp. 11–19.

[64] H.-H. Hsu, C.-W. Hsieh, and M.-D. Lu, "Hybrid feature selection by combining filters and wrappers," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8144–8150, 2011.

[65] M. Naseriparsa, A.-M. Bidgoli, and T. Varaee, "A hybrid feature selection method to improve performance of a group of classification algorithms," *arXiv preprint arXiv:1403.2372*, 2014.

[66] A. Zakari, A. A. Lawan, and G. Bekaroo, "A hybrid three-phased approach in requirement elicitation," in *International Conference on Emerging Trends in Electrical, Electronic and Communications Engineering.* Springer, 2016, pp. 331–340.

[67] W. Behutiye, P. Karhapää, D. Costal, M. Oivo, and X. Franch, "Non-functional requirements documentation in agile software development: challenges and solution proposal," in *International conference on product-focused software process improvement.* Springer, 2017, pp. 515–522.

[68] A. Silva, P. Pinheiro, A. Albuquerque, and J. Barroso, "A process for creating the elicitation guide of non-functional requirements," in *Software Engineering Perspectives and Application in Intelligent Systems: Proceedings of the 5th Computer Science On-line Conference 2016 (CSOC2016), Vol 2 5.* Springer, 2016, pp. 293–302.

[69] A. Silva, P. Pinheiro, J. Barroso, and A. Albuquerque, "A survey about the situation of the elicitation of non-functional requirements," in *2016 11th Iberian Conference on Information Systems and Technologies (CISTI).* IEEE, 2016, pp. 1–6.

[70] J. Zou, L. Xu, M. Yang, X. Zhang, and D. Yang, "Towards comprehending the non-functional requirements through developers eyes: An exploration of stack overflow using topic analysis," *Information and Software Technology*, vol. 84, pp. 19–32, 2017.

[71] J. Horkoff, "Non-functional requirements for machine learning: Challenges and new directions," in *2019 IEEE 27th International Requirements Engineering Conference (RE).* IEEE, 2019, pp. 386–391.

[72] A. Ahmad, K. Li, C. Feng, and T. Sun, "An empirical study on how ios developers report quality aspects on stack overflow," *International Journal of Machine Learning and Computing*, vol. 8, no. 5, pp. 501–506, 2018.

[73] A. Mahmoud and G. Williams, "Detecting, classifying, and tracing non-functional software requirements," *Requirements Engineering*, vol. 21, no. 3, pp. 357–381, 2016.

[74] J. Zou, L. Xu, W. Guo, M. Yan, D. Yang, and X. Zhang, "Which non-functional requirements do developers focus on? an empirical study on stack overflow using topic analysis," in *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories.* IEEE, 2015, pp. 446–449.

[75] M. Weiss and H. Mouratidis, "Selecting security patterns that fulfill security requirements," in *2008 16th IEEE International Requirements Engineering Conference.* IEEE, 2008, pp. 169–172.

[76] N. Subramanian, R. Puerzer, and L. Chung, "A comparative evaluation of maintainability: a study of engineering department's web site maintainability," in *21st IEEE International Conference on Software Maintenance (ICSM'05).* IEEE, 2005, pp. 669–672.

[77] A. S. Puspaningrum, S. Rochimah, and R. J. Akbar, "Functional suitability measurement using goal-oriented approach based on iso/iec 25010 for academics information system," *Journal of Information Systems Engineering and Business Intelligence*, vol. 3, no. 2, pp. 68–74, 2017.

[78] C. Wang, J. Li, P. Liang, M. Daneva, and M. Sinderen, "Developers' eyes on the changes of apps: An exploratory study on app changelogs," in *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW).* IEEE, 2019, pp. 207–212.

[79] A. Casamayor, D. Godoy, and M. Campo, "Identification of non-functional requirements in textual specifications: A semi-supervised learning approach," *Information and Software Technology*, vol. 52, no. 4, pp. 436–445, 2010.

[80] M. Riaz, J. King, J. Slankas, and L. Williams, "Hidden in plain sight: Automatically identifying security requirements from natural language artifacts," in *2014 IEEE 22nd International Requirements Engineering Conference (RE).* IEEE, 2014, pp. 183–192.

[81] R. Mizouni and A. Salah, "Towards a framework for estimating system nfrs on behavioral models," *Knowledge-Based Systems*, vol. 23, no. 7, pp. 721–731, 2010.

[82] A. Mahmoud, "An information theoretic approach for extracting and tracing non-functional requirements," in *2015 IEEE 23rd International Requirements Engineering Conference (RE).* IEEE, 2015, pp. 36–45.

[83] H. Sun, S. Basu, V. Honavar, and R. Lutz, "Automata-based verification of security requirements of composite web services," in *2010 IEEE 21st International Symposium on Software Reliability Engineering.* IEEE, 2010, pp. 348–357.

[84] M. Bhatia and A. Manocha, "Cognitive framework of food quality assessment in iot-inspired smart restaurants," *IEEE Internet of Things Journal*, 2020.

[85] M. Bhatia and S. K. Sood, "An intelligent framework for workouts in gymnasium: M-health perspective," *Computers & Electrical Engineering*, vol. 65, pp. 292–309, 2018.

[86] M. Bhatia, "Game theory based framework of smart food quality assessment," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 12, p. e3926, 2020.

[87] M. Bhatia, S. Kaur, S. K. Sood, and V. Behal, "Internet of things-inspired healthcare system for urine-based diabetes prediction," *Artificial Intelligence in Medicine*, vol. 107, p. 101913, 2020.

[88] V. Behal and R. Singh, "Personalised healthcare model for monitoring and prediction of airpollution: machine learning approach," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 33, no. 3, pp. 425–449, 2021.

[89] M. Rahimi, M. Mirakhorli, and J. Cleland-Huang, "Automated extraction and visualization of quality concerns from requirements specifications," in *2014 IEEE 22nd international requirements engineering conference (RE).* IEEE, 2014, pp. 253–262.

[90] K. M. Habibullah and J. Horkoff, "Non-functional requirements for machine learning: understanding current use and challenges in industry," in *2021 IEEE 29th International Requirements Engineering Conference (RE).* IEEE, 2021, pp. 13–23.

[91] D. A. Ramadhani, S. Rochimah, and U. L. Yuhana, "Classification of non-functional requirements using semantic-fsknn based iso/iec 9126," *Telkomnika*, vol. 13, no. 4, p. 1456, 2015.

[92] M. A. Haque, M. A. Rahman, and M. S. Siddik, "Non-functional requirements classification with feature extraction and machine learning: An empirical study," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*. IEEE, 2019, pp. 1–5.

[93] C. Mohan and S. Nagarajan, "An improved tree model based on ensemble feature selection for classification," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 27, pp. 1290–1307, 2019.

[94] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," *Data classification: Algorithms and applications*, p. 37, 2014.

[95] P. Tahmasebi, F. Javadpour, and M. Sahimi, "Data mining and machine learning for identifying sweet spots in shale reservoirs," *Expert Systems with Applications*, vol. 88, pp. 435–447, 2017.

[96] I. Kavakiotis, O. Tsave, A. Salifoglou, N. Maglaveras, I. Vlahavas, and I. Chouvarda, "Machine learning and data mining methods in diabetes research," *Computational and structural biotechnology journal*, vol. 15, pp. 104–116, 2017.

[97] T. Le, C. Le, H. D. Jeong, S. B. Gilbert, and E. Chukharev-Hudilainen, "Requirement text detection from contract packages to support project definition determination," in *Advances in Informatics and Computing in Civil and Construction Engineering*. Springer, 2019, pp. 569–576.

[98] M. Mishra and M. Srivastava, "A view of artificial neural network," in *2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014)*. IEEE, 2014, pp. 1–3.

[99] L. C. Yan, B. Yoshua, and H. Geoffrey, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[100] C.-W. Huang, C.-T. Chiang, and Q. Li, "A study of deep learning networks on mobile traffic forecasting," in *2017 IEEE 28th annual international symposium on personal, indoor, and mobile radio communications (PIMRC)*. IEEE, 2017, pp. 1–6.

[101] K. Dutta, P. Krishnan, M. Mathew, and C. Jawahar, "Improving cnn-rnn hybrid networks for handwriting recognition," in *2018 16th international conference on frontiers in handwriting recognition (ICFHR)*. IEEE, 2018, pp. 80–85.

[102] S. Selvin, R. Vinayakumar, E. Gopalakrishnan, V. K. Menon, and K. Soman, "Stock price prediction using lstm, rnn and cnn-sliding window model," in *2017 international conference on advances in computing, communications and informatics (icacci)*. IEEE, 2017, pp. 1643–1647.

[103] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. IEEE, 2019, pp. 291–300.

[104] V. M. Khatian, Q. A. Arain, M. Alenezi, M. O. Raza, F. Shaikh, and I. Farah, "Comparative analysis for predicting non-functional requirements using supervised machine learning," in *2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA)*. IEEE, 2021, pp. 7–12.

[105] A. S. Poonia, C. Banerjee, A. Banerjee, and S. Sharma, "Aligning misuse case oriented quality requirements metrics with machine learning approach," in *Soft Computing: Theories and Applications*. Springer, 2019, pp. 687–692.

[106] V. Vaithiyanathan, K. Rajeswari, K. Tajane, and R. Pitale, "Comparison of different classification techniques using different datasets," *International Journal of Advances in Engineering & Technology*, vol. 6, no. 2, p. 764, 2013.

[107] R. R. Maiti, A. Krasnov, and D. M. Wilborne, "Agile software engineering & the future of non-functional requirements," *Journal of Software Engineering Practice*, vol. 2, no. 1, pp. 1–8, 2018.

[108] A. Kumar, "Measuring software reusability using svm based classifier approach," *International Journal of Information Technology and Knowledge Management*, vol. 5, no. 1, pp. 205–209, 2012.

[109] Y. Kamei, A. Monden, and K.-i. Matsumoto, "Empirical evaluation of svm-based software reliability model," in *Proc. 5th ACM-IEEE Int'l Symposium on Empirical Software Engineering (ISESE2006)*, vol. 2. Citeseer, 2006, pp. 39–41.

[110] A. Rashwan, O. Ormandjieva, and R. Witte, "Ontology-based classification of non-functional requirements in software specifications: a new corpus and svm-based classifier," in *2013 IEEE 37th Annual Computer Software and Applications Conference*. IEEE, 2013, pp. 381–386.

[111] J. A. Reshi and S. Singh, "Predicting software defects through svm: an empirical approach," *arXiv preprint arXiv:1803.03220*, 2018.

[112] M. Pal, "Random forest classifier for remote sensing classification," *International journal of remote sensing*, vol. 26, no. 1, pp. 217–222, 2005.

[113] D. R. Ibrahim, R. Ghnemat, and A. Hudaib, "Software defect prediction using feature selection and random forest algorithm," in *2017 International Conference on New Trends in Computing Sciences (ICTCS)*. IEEE, 2017, pp. 252–257.

[114] Z. Kurtanović and W. Maalej, "Automatically classifying functional and non-functional requirements using supervised machine learning," in *2017 IEEE 25th International Requirements Engineering Conference (RE)*. IEEE, 2017, pp. 490–495.

[115] A. B. Nassif, L. F. Capretz, and D. Ho, "Estimating software effort using an ann model based on use case points," in *2012 11th International Conference on machine learning and applications*, vol. 2. IEEE, 2012, pp. 42–47.

[116] B. Dhanalaxmi, G. A. Naidu, and K. Anuradha, "Adaptive pso based association rule mining technique for software defect classification using ann," *Procedia Computer Science*, vol. 46, pp. 432–442, 2015.

[117] Q. Hu, Y.-S. Dai, M. Xie, and S. Ng, "Early software reliability prediction with extended ann model," in *30th Annual International Computer Software and Applications Conference (COMPSAC'06)*, vol. 2. IEEE, 2006, pp. 234–239.

[118] D. K. Meena, D. M. Manimekalai, and S. Rethinavalli, "Implementing neural fuzzy rough set and artificial neural network for predicting pcos," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 3, no. 12, pp. 6722–6727, 2015.

[119] Y. Yang, X. Xia, D. Lo, and J. Grundy, "A survey on deep learning for software engineering," *ACM Computing Surveys (CSUR)*, 2020.

[120] Q. Umer, H. Liu, and I. Illahi, "Cnn-based automatic prioritization of bug reports," *IEEE Transactions on Reliability*, vol. 69, no. 4, pp. 1341–1354, 2019.

[121] T. Hidayat and S. Rochimah, "Nfr classification using keyword extraction and cnn on app reviews," in *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*. IEEE, 2021, pp. 211–216.

[122] M. A. Rahman, M. A. Haque, M. N. A. Tawhid, and M. S. Siddik, "Classifying non-functional requirements using rnn variants for quality software development," in *Proceedings of the 3rd ACM SIGSOFT International Workshop on Machine Learning Techniques for Software Quality Evaluation*, 2019, pp. 25–30.

[123] K. Kaur and P. Kaur, "Sabdm: A self-attention based bidirectional-rnn deep model for requirements classification," *Journal of Software: Evolution and Process*, p. e2430, 2022.

[124] R. Chatterjee, A. Ahmed, and P. R. Anish, "Identification and classification of architecturally significant functional requirements," in *2020 IEEE Seventh International Workshop on Artificial Intelligence for Requirements Engineering (AIRE).* IEEE, 2020, pp. 9–17.

[125] Z.-H. Zhou, "Ensemble learning," in *Machine Learning.* Springer, 2021, pp. 181–210.

[126] R. S. Wahono, "A systematic literature review of software defect prediction," *Journal of Software Engineering*, vol. 1, no. 1, pp. 1–16, 2015.

[127] N. Rahimi, F. Eassa, and L. Elrefaei, "An ensemble machine learning technique for functional requirement classification," *Symmetry*, vol. 12, no. 10, p. 1601, 2020.

[128] E. Kocaguneli, T. Menzies, and J. W. Keung, "On the value of ensemble effort estimation," *IEEE Transactions on Software Engineering*, vol. 38, no. 6, pp. 1403–1416, 2011.

[129] D. W. Opitz, "Feature selection for ensembles," *AAAI/IAAI*, vol. 379, p. 384, 1999.

[130] B. Venkatesh and J. Anuradha, "A review of feature selection and its methods," *Cybernetics and Information Technologies*, vol. 19, no. 1, pp. 3–26, 2019.

[131] M. W. Mwadulo, "A review on feature selection methods for classification tasks," 2016.

[132] D. Morariu, R. Cretulescu, and M. Breazu, "Feature selection in document classification," in *The fourth international conference in romania of information science and information literacy, ISSN-L*, 2013, pp. 2247–0255.

[133] C. Catal and B. Diri, "Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem," *Information Sciences*, vol. 179, no. 8, pp. 1040–1058, 2009.

[134] V. Sachdeva and L. Chung, "Handling non-functional requirements for big data and iot projects in scrum," in *2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence.* IEEE, 2017, pp. 216–221.

[135] M. Younas, M. A. Shah, D. N. Jawawi, M. K. Ishfaq, M. Awais, K. Wakil, and A. Mustafa, "Elicitation of nonfunctional requirements in agile development using cloud computing environment," *IEEE Access*, 2020.

[136] F. Khan, S. R. Jan, M. Tahir, S. Khan, and F. Ullah, "Survey: dealing non-functional requirements at architecture level," *VFAST Transactions on Software Engineering*, vol. 9, no. 2, pp. 7–13, 2016.

[137] F. Khomh, B. Adams, J. Cheng, M. Fokaefs, and G. Antoniol, "Software engineering for machine-learning applications: The road ahead," *IEEE Software*, vol. 35, no. 5, pp. 81–84, 2018.

[138] Z. S. H. Abad, O. Karras, P. Ghazi, M. Glinz, G. Ruhe, and K. Schneider, "What works better? a study of classifying requirements," in *2017 IEEE 25th International Requirements Engineering Conference (RE).* IEEE, 2017, pp. 496–501.

[139] W. M. Farid and F. J. Mitropoulos, "Normatic: A visual tool for modeling non-functional requirements in agile processes," in *2012 Proceedings of IEEE Southeastcon*. IEEE, 2012, pp. 1–8.

[140] N. Yusop, D. Zowghi, and D. Lowe, "The impacts of non-functional requirements in web system projects," *International Journal of Value Chain Management*, vol. 2, no. 1, pp. 18–32, 2008.

[141] S. Ezami, "Extracting non-functional requirements from unstructured text," Master's thesis, University of Waterloo, 2018.

[142] B. Di Martino, J. Pascarella, S. Nacchia, S. A. Maisto, P. Iannucci, and F. Cerri, "Cloud services categories identification from requirements specifications," in *2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA)*. IEEE, 2018, pp. 436–441.

[143] T. Bhowmik and A. Q. Do, "Refinement and resolution of just-in-time requirements in open source software and a closer look into non-functional requirements," *Journal of Industrial Information Integration*, vol. 14, pp. 24–33, 2019.

[144] R. L. Portugal, T. Li, L. Silva, E. Almentero, and J. C. S. do Prado Leite, "Nfrfinder: a knowledge based strategy for mining non-functional requirements," in *Proceedings of the XXXII Brazilian Symposium on Software Engineering*, 2018, pp. 102–111.

[145] L. Tóth and L. Vidács, "Comparative study of the performance of various classifiers in labeling non-functional requirements," *Information Technology and Control*, vol. 48, no. 3, pp. 432–445, 2019.

[146] C. Baker, L. Deng, S. Chakraborty, and J. Dehlinger, "Automatic multi-class non-functional software requirements classification using neural networks,"

in *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2. IEEE, 2019, pp. 610–615.

[147] Z. S. H. Abad, V. Gervasi, D. Zowghi, and B. H. Far, "Supporting analysts by dynamic extraction and classification of requirements-related knowledge," in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 2019, pp. 442–453.

[148] S. Taj, Q. Arain, I. Memon, and A. Zubedi, "To apply data mining for classification of crowd sourced software requirements," in *Proceedings of the 2019 8th International Conference on Software and Information Engineering*, 2019, pp. 42–46.

[149] R. Naseem, Z. Shaukat, M. Irfan, M. A. Shah, A. Ahmad, F. Muhammad, A. Glowacz, L. Dunai, J. Antonino-Daviu, and A. Sulaiman, "Empirical assessment of machine learning techniques for software requirements risk prediction," *Electronics*, vol. 10, no. 2, p. 168, 2021.

[150] M. Assim, Q. Obeidat, and M. Hammad, "Software defects prediction using machine learning algorithms," in *2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI)*. IEEE, 2020, pp. 1–6.

[151] L. Chung and J. C. S. do Prado Leite, "On non-functional requirements in software engineering," in *Conceptual modeling: Foundations and applications*. Springer, 2009, pp. 363–379.

[152] M. Mirakhorli and J. Cleland-Huang, "Tracing non-functional requirements," in *Software and systems traceability*. Springer, 2012, pp. 299–320.

[153] W. M. Farid, "The normap methodology: Lightweight engineering of non-functional requirements for agile processes," in *2012 19th Asia-Pacific Software Engineering Conference*, vol. 1. IEEE, 2012, pp. 322–325.

[154] J. Slankas and L. Williams, “Automated extraction of non-functional requirements in available documentation,” in *2013 1st International workshop on natural language analysis in software engineering (NaturaLiSE).* IEEE, 2013, pp. 9–16.

[155] M. Chen, T. H. Tan, J. Sun, Y. Liu, J. Pang, and X. Li, “Verification of functional and non-functional requirements of web service composition,” in *International Conference on Formal Engineering Methods.* Springer, 2013, pp. 313–328.

[156] S. Kopczyńska and J. Nawrocki, “Using non-functional requirements templates for elicitation: A case study,” in *2014 IEEE 4th International Workshop on Requirements Patterns (RePa).* IEEE, 2014, pp. 47–54.

[157] V. S. Sharma, R. R. Ramnani, and S. Sengupta, “A framework for identifying and analyzing non-functional requirements from text,” in *Proceedings of the 4th international workshop on twin peaks of requirements and architecture.* ACM, 2014, pp. 1–8.

[158] A. Raturi, B. Penzenstadler, B. Tomlinson, and D. Richardson, “Developing a sustainability non-functional requirements framework,” in *Proceedings of the 3rd International Workshop on Green and Sustainable Software*, 2014, pp. 1–8.

[159] M. Lu and P. Liang, “Automatic classification of non-functional requirements from augmented app user reviews,” in *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*, 2017, pp. 344–353.

[160] Y. Matsumoto, S. Shirai, and A. Ohnishi, “A method for verifying non-functional requirements,” *Procedia Computer Science*, vol. 112, pp. 157–166, 2017.

[161] M. Sabir, E. Banissi, and M. Child, “A deep learning-based framework for the classification of non-functional requirements,” in *Trends and Applications in Information Systems and Technologies: Volume 2 9*. Springer, 2021, pp. 591–601.

[162] B. W. Boehm, J. R. Brown, and M. Lipow, “Quantitative evaluation of software quality,” in *Proceedings of the 2nd international conference on Software engineering*, 1976, pp. 592–605.

[163] G.-C. Roman, “A taxonomy of current issues in requirements engineering,” *IEEE computer*, vol. 18, no. 4, pp. 14–23, 1985.

[164] R. B. Grady and D. L. Caswell, *Software metrics: establishing a company-wide program*. Prentice-Hall, Inc., 1987.

[165] I. E. Commission, *Software Engineering-Product Quality*. ISO/IEC, 2001, vol. 9126.

[166] N. Afreen, A. Khatoon, and M. Sadiq, “A taxonomy of softwares non-functional requirements,” in *Proceedings of the second international conference on computer and communication technologies*. Springer, 2016, pp. 47–53.

[167] J. P. Miguel, D. Mauricio, and G. Rodríguez, “A review of software quality models for the evaluation of software products,” *arXiv preprint arXiv:1412.2977*, 2014.

[168] H. Iqbal and M. Babar, “An approach for analyzing iso/iec 25010 product quality requirements based on fuzzy logic and likert scale for decision support systems,” *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 12, pp. 245–260, 2016.

[169] E. Peters and G. Aggrey, “An iso 25010 based quality model for erp systems,” *Advances in Science, Technology and Engineering Systems*, vol. 5, no. 2, pp. 578–583, 2020.

[170] I. Kadi, A. Idri, and S. Ouhbi, "Quality evaluation of cardiac decision support systems using iso 25010 standard," in *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA).* IEEE, 2016, pp. 1–8.

[171] A. Hussain and E. O. Mkpojiogu, "An application of the iso/iec 25010 standard in the quality-in use assessment of an online health awareness system," *Jurnal Teknologi*, vol. 77, no. 5, pp. 9–13, 2015.

[172] E. Yildiz, S. Bilgen, G. Tokdemir, N. E. Cagiltay, and Y. N. Erturan, "Analysis of b2c mobile application characteristics and quality factors based on iso 25010 quality model," in *International Conference on Mobile Web and Information Systems.* Springer, 2014, pp. 261–274.

[173] M. Izzatillah, "Quality measurement of transportation service application go-jek using iso 25010 quality model," *Simetris: Jurnal Teknik Mesin, Elektro dan Ilmu Komputer*, vol. 10, no. 1, pp. 233–242, 2019.

[174] O. Alshareet, A. Itradat, I. A. Doush, and A. Quttoum, "Incorporation of iso 25010 with machine learning to develop a novel quality in use prediction system (qiups)," *International Journal of System Assurance Engineering and Management*, vol. 9, no. 2, pp. 344–353, 2018.

[175] A. D. Alharthi, M. Spichkova, and M. Hamilton, "Sustainability requirements for elearning systems: a systematic literature review and analysis," *Requirements Engineering*, vol. 24, no. 4, pp. 523–543, 2019.

[176] F. H. Wattiheluw, S. Rochimah, C. Fatichah, and K. Z. Abidin, "Development of a quality model based on iso 25010 using fuzzy and pso for e-commerce websites," in *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON).* IEEE, 2020, pp. 250–254.

[177] M. R. Dewi, N. Ngaliah, and S. Rochimah, "Maintainability measurement and evaluation of myits mobile application using iso 25010 quality standard," in *2020 International Seminar on Application for Technology of Information and Communication (iSemantic).* IEEE, 2020, pp. 530–536.

[178] H. Panduwiyasa, M. Saputra, Z. Azzahra, and A. Aniko, "Accounting and smart system: Functional evaluation of iso/iec 25010: 2011 quality model (a case study)," in *IOP Conference Series: Materials Science and Engineering*, vol. 1092, no. 1. IOP Publishing, 2021, p. 012065.

[179] M. L. Atanacio and L. L. Lacatan, "Development and evaluation of rural health unit record management system with data analytics for municipality of bay, laguna using iso 25010," *International Journal of Recent Technology and Engineering*, vol. 8, no. 3, pp. 3915–3919, 2019.

[180] G. F. Felipe, F. E. T. Lima, L. P. Barbosa, T. M. M. Moreira, E. S. Joventino, V. S. Freire, and L. B. d. A. Mendonça, "Evaluation of user embracement software with pediatric risk classification," *Revista brasileira de enfermagem*, vol. 73, 2020.

[181] A. A. Pratama and A. B. Mutiara, "Software quality analysis for halodoc application using iso 25010: 2011," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 8, 2021.

[182] A. Acharya and D. Sinha, "Assessing the quality of m-learning systems using iso/iec 25010," *International Journal of Advanced Computer Research*, vol. 3, no. 3, p. 67, 2013.

[183] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of research and development*, vol. 3, no. 3, pp. 210–229, 1959.

[184] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning.* MIT press, 2016.

[185] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[186] E. Dias Canedo and B. Cordeiro Mendes, "Software requirements classification using machine learning algorithms," *Entropy*, vol. 22, no. 9, p. 1057, 2020.

[187] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Machine learning proceedings 1992.* Elsevier, 1992, pp. 249–256.

[188] M. Dash and H. Liu, "Feature selection for classification," *Intelligent data analysis*, vol. 1, no. 1-4, pp. 131–156, 1997.

[189] V. Gupta, A. Mehta, A. Goel, U. Dixit, and A. C. Pandey, "Spam detection using ensemble learning," in *Harmony Search and Nature Inspired Optimization Algorithms.* Springer, 2019, pp. 661–668.

[190] S. Rahy, J. Bass *et al.*, "Managing non-functional requirements in agile software development," *IET Software*, 2021.

[191] R. K. Gnanasekaran, S. Chakraborty, J. Dehlinger, and L. Deng, "Using recurrent neural networks for classification of natural language-based non-functional requirements." in *REFSQ Workshops*, 2021.

[192] T. Hey, J. Keim, A. Koziolek, and W. F. Tichy, "Norbert: Transfer learning for requirements classification," in *2020 IEEE 28th International Requirements Engineering Conference (RE).* IEEE, 2020, pp. 169–179.

[193] N. Rahimi, F. Eassa, and L. Elrefaei, "One-and two-phase software requirement classification using ensemble deep learning," *Entropy*, vol. 23, no. 10, p. 1264, 2021.

[194] Y. Gazi, M. S. Umar, and M. Sadiq, "Classification of nfrs for information system," *International Journal of Computer Applications*, vol. 115, no. 22, 2015.

[195] M. Aasem, M. Ramzan, and A. Jaffar, “Analysis and optimization of software requirements prioritization techniques,” in *2010 International Conference on Information and Emerging Technologies.* IEEE, 2010, pp. 1–6.

[196] B. M. Aljallabi and A. Mansour, “Enhancement approach for non-functional requirements analysis in agile environment,” in *2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE).* IEEE, 2015, pp. 428–433.

[197] H. M. Kiran and Z. Ali, “Requirement elicitation techniques for open source systems: A review,” *International Journal of Advanced Computer Science and Applications, Pakistan*, pp. 330–334, 2018.

[198] S. Tiwari and S. S. Rathore, “A methodology for the selection of requirement elicitation techniques,” *arXiv preprint arXiv:1709.08481*, 2017.

[199] M. Asadi, S. Soltani, D. Gasevic, M. Hatala, and E. Bagheri, “Toward automated feature model configuration with optimizing non-functional requirements,” *Information and Software Technology*, vol. 56, no. 9, pp. 1144–1165, 2014.

[200] E. C. Groen, S. Kopczyńska, M. P. Hauer, T. D. Krafft, and J. Doerr, “Users the hidden software product quality experts?: A study on how app users report quality aspects in online reviews,” in *2017 IEEE 25th International Requirements Engineering Conference (RE).* IEEE, 2017, pp. 80–89.

[201] R. R. Maiti and F. J. Mitropoulos, “Prioritizing non-functional requirements in agile software engineering,” in *Proceedings of the SouthEast Conference.* ACM, 2017, pp. 212–214.

[202] S.-J. Huang *et al.*, “The design of a software engineering life cycle process for big data projects,” *IT Professional*, 2017.

[203] S. N. Mahalank, K. B. Malagund, and R. Banakar, “Non functional requirement analysis in iot based smart traffic management system,” in *2016 International Conference on Computing Communication Control and automation (ICCUBEA).* IEEE, 2016, pp. 1–6.

[204] J. Ali, M. S. Shoukat, and M. Faisal, “Determining interdependencies among nfrs in agile environment to reduce conflicts.” *ICST Trans. Mobile Communications Applications*, vol. 3, no. 13, p. e3, 2018.

[205] A. Q. Do and T. Bhowmik, “Refinement and resolution of just-in-time requirements in open source software: a case study,” in *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW).* IEEE, 2017, pp. 407–410.

[206] X. T. Nguyen, H. T. Tran, H. Baraki, and K. Geihs, “Optimization of non-functional properties in internet of things applications,” *Journal of Network and Computer Applications*, vol. 89, pp. 120–129, 2017.

[207] I. Noorwali, D. Arruda, and N. H. Madhavji, “Understanding quality requirements in the context of big data systems,” in *Proceedings of the 2nd International Workshop on BIG Data Software Engineering.* ACM, 2016, pp. 76–79.

[208] B. Li, Z. Li, and Y. Yang, “Nfrnet: A deep neural network for automatic classification of non-functional requirements,” in *2021 IEEE 29th International Requirements Engineering Conference (RE).* IEEE, 2021, pp. 434–435.

[209] H. Eyal Salman, M. Hammad, A.-D. Seriai, and A. Al-Sbou, “Semantic clustering of functional requirements using agglomerative hierarchical clustering,” *Information*, vol. 9, no. 9, p. 222, 2018.

[210] M. Sabir, C. Chrysoulas, and E. Banissi, “Multi-label classifier to deal with misclassification in non-functional requirements,” in *World Conference on Information Systems and Technologies.* Springer, 2020, pp. 486–493.

[211] G. Y. Quba, H. Al Qaisi, A. Althunibat, and S. AlZubi, "Software requirements classification using machine learning algorithms," in *2021 International Conference on Information Technology (ICIT).* IEEE, 2021, pp. 685–690.

[212] T. H. Al Balushi, P. R. F. Sampaio, D. Dabhi, and P. Loucopoulos, "Elicito: a quality ontology-guided nfr elicitation tool," in *Requirements Engineering: Foundation for Software Quality: 13th International Working Conference, REFSQ 2007, Trondheim, Norway, June 11-12, 2007. Proceedings 13.* Springer, 2007, pp. 306–319.

[213] R. Veleda and L. M. Cysneiros, "Towards a tool to help exploring existing non-functional requirements solution patterns," in *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW).* IEEE, 2017, pp. 232–239.

[214] D. García-López, M. Segura-Morales, and E. Loza-Aguirre, "Improving the quality and quantity of functional and non-functional requirements obtained during requirements elicitation stage for the development of e-commerce mobile applications: an alternative reference process model," *IET Software*, vol. 14, no. 2, pp. 148–158, 2020.

[215] N. Handa, A. Sharma, and A. Gupta, "Framework for prediction and classification of non functional requirements: a novel vision," *Cluster Computing*, vol. 25, no. 2, pp. 1155–1173, 2022.

[216] M. Shahid and K. A. Tasneem, "Impact of avoiding non-functional requirements in software development stage," *American Journal of Information Science and Computer Engineering*, vol. 3, no. 4, pp. 52–55, 2017.

[217] T. Iqbal and M. Suaib, "Requirement elicitation technique:-a review paper," *Int. J. Comput. Math. Sci*, vol. 3, no. 9, pp. 1–6, 2014.

[218] U. Sajjad and M. Q. Hanif, “Issues and challenges of requirement elicitation in large web projects,” 2010.

[219] P. Sharmila and R. Umarani, “A walkthrough of requirement elicitation techniques,” *International Journal of Engineering Research and Applications*, vol. 1, no. 4, pp. 1583–1586, 2011.

[220] D. Mishra, S. Aydin, A. Mishra, and S. Ostrovska, “Knowledge management in requirement elicitation: Situational methods view,” *Computer Standards & Interfaces*, vol. 56, pp. 49–61, 2018.

[221] C. Ribeiro, C. Farinha, J. Pereira, and M. M. da Silva, “Gamifying requirement elicitation: Practical implications and outcomes in improving stakeholders collaboration,” *Entertainment Computing*, vol. 5, no. 4, pp. 335–345, 2014.

[222] N. C. Pa and A. M. Zin, “Requirement elicitation: identifying the communication challenges between developer and customer,” *International Journal of New Computer Architectures and their Applications (IJNCAA)*, vol. 1, no. 2, pp. 371–383, 2011.

[223] H.-H. Wu and J.-I. Shieh, “Applying repertory grids technique for knowledge elicitation in quality function deployment,” *Quality & Quantity*, vol. 44, no. 6, pp. 1139–1149, 2010.

[224] M. A. Abbasi, J. Jabeen, Y. Hafeez, D. Batool, and N. Fareen, “Assessment of requirement elicitation tools and techniques by various parameters,” *Software Engineering*, vol. 3, no. 2, pp. 7–11, 2015.

[225] A. Poth and A. Riel, “Quality requirements elicitation by ideation of product quality risks with design thinking,” in *2020 IEEE 28th International Requirements Engineering Conference (RE)*. IEEE, 2020, pp. 238–249.

[226] Z. Wang, C.-H. Chen, P. Zheng, X. Li, and L. P. Khoo, “A graph-based context-aware requirement elicitation approach in smart product-service systems,” *International Journal of Production Research*, vol. 59, no. 2, pp. 635–651, 2021.

[227] E. Kahan, E. Insfran, M. Genero, and A. Oliveros, “Studying the influence of empathy maps on brainstorming for requirements elicitation: A quasi-experiment,” 2021.

[228] M. Suhaib, “Investigation and analysis of the requirement engineering in software development process and its systematic requirements elicitation approach,” *International Journal of Scientific and Technology Research, ISSN: 2277*, vol. 8616, pp. 2723–2726, 2020.

[229] D. Gobov and I. Huchenko, “Requirement elicitation techniques for software projects in ukrainian it: An exploratory study,” in *2020 15th Conference on Computer Science and Information Systems (FedCSIS).* IEEE, 2020, pp. 673–681.

[230] J. G. Ochin, “Cross browser incompatibility: reasons and solutions,” *International Journal of Software Engineering & Applications (IJSEA)*, vol. 2, no. 3, pp. 66–77, 2011.

[231] H. X. Lin, Y.-Y. Choong, and G. Salvendy, “A proposed index of usability: a method for comparing the relative usability of different software systems,” *Behaviour & information technology*, vol. 16, no. 4-5, pp. 267–277, 1997.

[232] R. Rohrbeck, F. Steinhoff, and F. Perder, “Virtual customer integration in the innovation process: Evaluation of the web platforms of multinational enterprises (mne),” in *PICMET’08-2008 Portland International Conference on Management of Engineering & Technology.* IEEE, 2008, pp. 469–478.

[233] S. H. Mustafa and L. F. Al-Zoua'bi, "Usability of the academic websites of jordan's universities an evaluation study," in *Proceedings of the 9th International Arab Conference for Information Technology*, 2008, pp. 31–40.

[234] S. Sahni and S. K. Dubey, "Web usability: Issues, challenges and solutions," *International Journal of Advanced Engineering Research and Science (IJAERS)*, vol. 1, pp. 61–66, 2014.

[235] W. A. Alberts and T. M. Van Der Geest, "Color matters: Color as trustworthiness cue in web sites," *Technical communication*, vol. 58, no. 2, pp. 149–160, 2011.

[236] G. W. Tan and K. K. Wei, "An empirical study of web browsing behaviour: Towards an effective website design," *Electronic Commerce Research and Applications*, vol. 5, no. 4, pp. 261–271, 2006.

[237] Y. Akgül, "Quality evaluation of e-government websites of turkey," in *2016 11th Iberian conference on information systems and technologies (CISTI)*. IEEE, 2016, pp. 1–7.

[238] A. Cockburn and B. McKenzie, "What do web users do? an empirical analysis of web use," *International Journal of human-computer studies*, vol. 54, no. 6, pp. 903–922, 2001.

[239] G. M. Ping Zhang, "User expectations and rankings of quality factors in different web site domains," *International journal of electronic commerce*, vol. 6, no. 2, pp. 9–33, 2001.

[240] R. Verborgh, T. Steiner, D. Van Deursen, J. De Roo, R. Van de Walle, and J. G. Vallés, "Capturing the functionality of web services with functional descriptions," *Multimedia tools and applications*, vol. 64, no. 2, pp. 365–387, 2013.

[241] M. Masse, *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces.* " O'Reilly Media, Inc.", 2011.

[242] R. L. Chafin, "User manuals: What does the user really need?" in *Proceedings of the 1st annual international conference on Systems documentation*, 1982, pp. 36–39.

[243] A. Ahmad, A. Hussain, O. H. Flayyih, W. Abdulwahab, and M. I. Sabri, "Utilizing wammi components to evaluate the usability of e-commerce website," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 9, no. 2-11, pp. 139–143, 2017.

[244] J. Nielsen, *Usability engineering.* Morgan Kaufmann, 1994.

[245] T. K. Chiew and S. S. Salim, "Webuse: Website usability evaluation tool," *Malaysian Journal of Computer Science*, vol. 16, no. 1, pp. 47–57, 2003.

[246] K. Hornbæk, "Current practice in measuring usability: Challenges to usability studies and research," *International journal of human-computer studies*, vol. 64, no. 2, pp. 79–102, 2006.

[247] A. Assila, H. Ezzedine *et al.*, "Standardized usability questionnaires: Features and quality focus," *Electronic Journal of Computer Science and Information Technology*, vol. 6, no. 1, 2016.

[248] K. Finstad, "The usability metric for user experience," *Interacting with Computers*, vol. 22, no. 5, pp. 323–327, 2010.

[249] Y. Lee and K. A. Kozar, "Understanding of website usability: Specifying and measuring constructs and their relationships," *Decision support systems*, vol. 52, no. 2, pp. 450–463, 2012.

[250] J. Nielsen and K. Pernice, *Eyetracking web usability.* New Riders, 2010.

[251] A. M. Lund, "Measuring usability with the use questionnaire12," *Usability interface*, vol. 8, no. 2, pp. 3–6, 2001.

[252] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano, “Experimental perspectives on learning from imbalanced data,” in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 935–942.

[253] Y.-T. Kim, D.-K. Kim, H. Kim, and D.-J. Kim, “A comparison of oversampling methods for constructing a prognostic model in the patient with heart failure,” in *2020 international conference on information and communication technology convergence (ICTC)*. IEEE, 2020, pp. 379–383.

[254] S. Picek, A. Heuser, A. Jovic, S. Bhasin, and F. Regazzoni, “The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations,” *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 1, pp. 1–29, 2019.

[255] S. Chatterjee, M. Mastalerz, A. Drobniak, and C. Ö. Karacan, “Machine learning and data augmentation approach for identification of rare earth element potential in indiana coals, usa,” *International Journal of Coal Geology*, vol. 259, p. 104054, 2022.

[256] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla, “Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary,” *Journal of artificial intelligence research*, vol. 61, pp. 863–905, 2018.

[257] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[258] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, “Understanding data augmentation for classification: when to warp?” in *2016 international conference on digital image computing: techniques and applications (DICTA)*. IEEE, 2016, pp. 1–6.

[259] M. A. B. Hameed and Z. Alamgir, “Improving mortality prediction in acute pancreatitis by machine learning and data augmentation,” *Computers in Biology and Medicine*, vol. 150, p. 106077, 2022.

[260] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-smote: a new over-sampling method in imbalanced data sets learning,” in *International conference on intelligent computing.* Springer, 2005, pp. 878–887.

[261] H. He, Y. Bai, E. A. Garcia, and S. Li, “Adasyn: Adaptive synthetic sampling approach for imbalanced learning,” in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence).* IEEE, 2008, pp. 1322–1328.

[262] A. Kumar, S. Sarkar, and C. Pradhan, “Malaria disease detection using cnn technique with sgd, rmsprop and adam optimizers,” in *Deep learning techniques for biomedical and health informatics.* Springer, 2020, pp. 211–230.

[263] A. Ultsch, “Maps for the visualization of high-dimensional data spaces,” in *Proc. Workshop on Self organizing Maps*, 2003, pp. 225–230.

[264] J. Kirakowski and N. Claridge, “Wammi-web usability questionnaire,” 2013.

[265] N. Avouris, N. Tselios, C. Fidas, and E. Papachristos, “Website evaluation: A usability-based perspective,” in *Panhellenic Conference on Informatics.* Springer, 2001, pp. 217–231.

[266] N. S. Aziz and A. Kamaludin, “Using pre-test to validate the questionnaire for website usability (qwu),” in *2015 4th International Conference on Software Engineering and Computer Systems (ICSECS).* IEEE, 2015, pp. 107–111.

[267] J. R. Lewis, “Computer system usability questionnaire,” *International Journal of Human-Computer Interaction*, 1995.

[268] V. A. Zeithaml, A. Parasuraman, and A. Malhotra, “Service quality delivery through web sites: a critical review of extant knowledge,” *Journal of the academy of marketing science*, vol. 30, no. 4, pp. 362–375, 2002.

[269] N. Bonnardel, A. Piolat, and L. Le Bigot, “The impact of colour on website appeal and users cognitive processes,” *Displays*, vol. 32, no. 2, pp. 69–80, 2011.

[270] B. Hernández, J. Jiménez, and M. J. Martín, “Key website factors in e-business strategy,” *International Journal of information management*, vol. 29, no. 5, pp. 362–371, 2009.

[271] V. Moustakis, C. Litos, A. Dalivigas, L. Tsironis *et al.*, “Website quality assessment criteria.” in *ICIQ*, 2004, pp. 59–73.

[272] R. Islam, M. Ahmed, and M. H. Alias, “Application of quality function deployment in redesigning website: a case study on tv3,” *International Journal of Business Information Systems*, vol. 2, no. 2, pp. 195–216, 2007.

[273] I. Alsmadi and E. Abu-Shanab, “E-government website security concerns and citizens’ adoption,” *Electronic Government, an International Journal*, vol. 12, no. 3, pp. 243–255, 2016.

[274] J. Grossman, “The state of website security,” *IEEE Security & Privacy*, vol. 10, no. 4, pp. 91–93, 2012.

[275] G. Kelly and B. McKenzie, “Security, privacy, and confidentiality issues on the internet,” *Journal of Medical Internet Research*, vol. 4, no. 2, p. e12, 2002.

[276] H. Heitkötter, S. Hanschke, and T. A. Majchrzak, “Evaluating cross-platform development approaches for mobile applications,” in *International Conference on Web Information Systems and Technologies*. Springer, 2012, pp. 120–138.

[277] K. A. Harper and J. DeWaters, “A quest for website accessibility in higher education institutions,” *The Internet and Higher Education*, vol. 11, no. 3-4, pp. 160–164, 2008.

[278] S. Jo, T. Kim, V. G. Iyer, and W. Im, “Charmm-gui: a web-based graphical user interface for charmm,” *Journal of computational chemistry*, vol. 29, no. 11, pp. 1859–1865, 2008.

[279] D. West, “Global e-government, 2007,” 2007.

[280] E. Štrumbelj, Z. Bosnić, I. Kononenko, B. Zakotnik, and C. Grašič Kuhar, “Explanation and reliability of prediction models: the case of breast cancer recurrence,” *Knowledge and information systems*, vol. 24, no. 2, pp. 305–324, 2010.

[281] M. Bhatia, S. Kaur, and S. K. Sood, “Iot-inspired smart toilet system for home-based urine infection prediction,” *ACM Transactions on Computing for Healthcare*, vol. 1, no. 3, pp. 1–25, 2020.

[282] G. Adomavicius and J. Zhang, “Stability of recommendation algorithms,” *ACM Transactions on Information Systems (TOIS)*, vol. 30, no. 4, pp. 1–31, 2012.

[283] A. Aljumah, A. Kaur, M. Bhatia, and T. Ahamed Ahanger, “Internet of things-fog computing-based framework for smart disaster management,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 8, p. e4078, 2021.

[284] A. Manocha and M. Bhatia, “A novel deep fusion strategy for covid-19 prediction using multimodality approach,” *Computers and Electrical Engineering*, vol. 103, p. 108274, 2022.

Publications

Journals

- Naina Handa, Anil Sharma, “Non Functional Requirements: the emerging wisdom”, International Journal of Research and Analytical Reviews, 2018. (UGC)
- Naina Handa, Anil Sharma, Amardeep Gupta“Non Functional Requirements Analysis using Data Analytics”, International Journal of Advanced Science and Technology, 2019. (Scopus)
- Naina Handa, Anil Sharma, “Compact Discourse on Feature Selection”, Think India Journal, 2019. (UGC)
- Naina Handa, Anil Sharma, “An Inclusive Study of Several Machine Learning Techniques”, Journal of The Gujarat Research Society,2019. (UGC)
- Naina Handa, Anil Sharma, Amardeep Gupta, “An Elicitation of Non Functional Requirements using Questionnaire: A Layered Framework”, Solid State Technology, 2020. (Scopus)
- Naina Handa, Anil Sharma, Amardeep Gupta, “Hybrid Feature Selection based Classifier for Non Functional Requirements” Mathematical Statistician and Engineering Applications, 2022. (Scopus)

- Naina Handa, Anil Sharma, Amardeep Gupta, “Framework for prediction and classification of Non Functional Requirements: A novel vision” Cluster Computing, 2021. (SCI)

Conferences

- Naina Handa, Anil Sharma, Amardeep Gupta, “An Inclusive Study of Several ML Based Non-functional Requirements Prediction Techniques”FTNCT, 2019.
- Naina Handa, Anil Sharma, Amardeep Gupta, “A Review of Machine Learning Techniques in Non Functional Requirements” 1st National Conference on Innovations in Applied Science and Engineering NCIASE, 2019.
- Naina Handa, Anil Sharma, Amardeep Gupta, “Non-functional Requirements Engineering Questionnaire: Novel Visions and Review of Literature” Recent Innovations in Computing, 2020.