Incorporating Machine Learning Techniques to Predict Risk Assessment in Project Timeline Management

By

Maryam Aslam

00000317562

Supervisor

Engr Yawar Abbas Bangash, PhD

A thesis submitted to the faculty of Department of Computer Software Engineering, Military College of Signals, National University of Sciences and Technology, Islamabad, Pakistan, for the partial fulfillment of the requirement for the degree of MS in Software Engineering

October 2022

Declaration

I, *Maryam Aslam* declare that this thesis titled "Incorporating Machine Learning Techniques to Predict Risk Assessment in Project Timeline Management" and the work presented in it are my own and has been generated by me as a result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a Master of Science degree at NUST.
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at NUST or any other institution, this has been clearly stated.
3. Where I have consulted the published work of others, this is always clearly attributed.
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
5. I have acknowledged all main sources of help.
6. Where the thesis 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.

Maryam Aslam,
00000317562

Copyright Notice

This thesis is dedicated to *my beloved parents*

Abstract

Numerous project management applications have made extensive use of machine learning algorithms wherein risk assessment module plays an important role in project timeline management. In order to increase the likelihood that software projects will be successfully developed, research into project management applications that involve risk assessment has become increasingly popular in recent years, especially with the use of machine learning techniques to pinpoint project risk indicators before the project's development even begins.

In this research, we have applied five cutting-edge machine learning techniques namely Artificial Neural Network (ANN), Logistic Regression (LR), Naive Bayes (NB), Random Forest (RF), and K-Nearest Neighbors (K-NN) on two different risk datasets. Moreover, hyper-parameter tuning is applied using different parameters in each machine learning technique and find the optimal risk predictions in each of them. On the basis of comprehensive literature review, a methodology was proposed which made it easier for the project manager and subject matter experts to plan and reduce risks at an early stage by determining the level of risk involved. Moreover, different experiments are performed on benchmark datasets to compare the models performance based on the applied parameters. This research will assist domain experts, data analysts, developers, and researchers to better develop ML models and make effective predictions which provide guidance towards making effective decisions to meet the project's deadline and organizational goals.

Acknowledgments

First, I thank Almighty Allah for bestowing me the strength and the passion to complete this research work. Secondly, I would like to acknowledge the tireless and prompt help of my supervisor, Engr Dr Yawar Abbas Bangash. He supported me throughout the process from defining and exploring idea until its execution. His valuable help in form of critical reviews and suggestions, throughout the experimental process and thesis work, have been the major contributing factor to the success of this study. I also want to pay my special gratitude to my thesis co-supervisor Brig Dr Fahim Arif for his continuous support, monitoring and guidance. Without his assistance and dedicated involvement in every step throughout the duration, this work wouldn't have been completed. I would like to thank him very much for his understanding over this past duration. Also, I would like to thank the members of my committee; Lt Col Khawir Mahmood and Assistant Professor Dr Tauseef Rana for their support and knowledge sharing regarding the study topic. Moreover, I'm thankful to Brig Dr Asim Dilawar Bakshi for guiding me in various aspects of my thesis. Most importantly, none of this could have happened without the unceasing support and attention of my parents. I would also like to thanks my friends for their constant encouragement, care, love and support through my times of stress and excitement.

Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

ML	Machine Learning
PM	Project Management
SPM	Software Project Management
PMI	Project Management Institute
PMBOK	Project Management Body of Knowledge
ANN	Artificial Neural Network
K-NN	K-nearest Neighbour
NN	Neural Network
DT	Decision Tree
NB	Naive Bayes
LR	Logistic Regression
BNs	Bayesian Networks
RF	Random Forest
AI	Artificial Intelligence
BI	Business Intelligence

DNN	Deep Neural Network
SVM	Support Vector Machine
ANFIS	Adaptive Neuro-Fuzzy Inference Systems
HP	Hyperparameters
PCA	Principal Component Analysis
HCA	Hierarchical Cluster Analysis
WASD	Weight and Structure Determination
GMM	Gaussian Mixture Model
EM	Expectation-Maximization
SVD	Singular Value Decomposition
SGD	Stochastic Gradient Descent
MLP	Multilayer Perceptron
CSF	Cost-Sensitive Decision Forest
AODE	Average One Dependency Estimator
GBT	Gradient Boosted Trees
SRE	Software Risk Evaluation
MMRE	Mean magnitude of Relative Error
BNCC	BNs with Causality Constraints
BMMRE	Balanced Mean Magnitude of Relative Error
GS	Grid Search

Chapter 1

Introduction

1.1 Overview

Proposed work presents a comprehensive model in context of Project Management (PM) system. The main aim of the research work is to incorporate predictive analysis through Machine Learning (ML) techniques using PM modules wherein risk assessment and project timeline are the key elements. The major contribution of the work is to provide researchers with an understanding of ML techniques to develop a secure and concrete PM based platform for monitoring and tracking of tasks and milestones by keeping in view risk factors involved in projects' risk report to make effective decisions and meet various business needs.

The objective of this work is to design ML model in order to predict the probability of a project having concerns worth being highlighted in the PM risk report. The project risk report details require a substantial effort, as the analysts need to inspect reports and related documentation to determine if a project is suffered from high risk factors. So, by training a discriminative model, we will be able to prioritize the projects that statistically present a high risk profile and reduce the cost of report elaboration. We also want to explore the most significant factors that contribute to project risk like scope, managers, project risk likelihood, risk magnitude, project schedule, and risk priority etc.

In today's technical environment, organizations are emphasizing on the dynamic approach in order to compete in market. Here, the main challenge is to manage projects at run time and to achieve the entire targets within provided constraints, i.e., scope, time, quality, and budget [1]. The objective of PM is to gain client's goals and ends up

with the project successfully. The complex nature projects require an appropriate PM standard to track the risk factors being involved that effect the project progress dynamically [2]. Any discrepancy in managing the standards affect the goals of the project, which ultimately affect the organizational goals.

Today, there is a wide diversification and scope in PM based applications and their importance evolves dramatically. In order to develop such a competitive and successful applications, IT companies are emphasizing on quality oriented and secure applications with technical procedures to be implemented [3]. PM, an application of knowledge, tools and techniques can be accomplished through process of initializing, planning, executing, monitoring, leading, controlling and closing [4]. In this case, to develop a competitive PM based application with differential Business Intelligence (BI), we must know the goals and realistic objectives of the system to be developed, make a proper plan and create a schedule, identify risk assessment elements, identify activities to be performed, resources to be allocated, goals to be achieved, milestones to be tracked, and strategies to be formulated [5]. This way, organizational level processes can be improved and highlighted in risk report positively to attain targeted goals.

Risk assessment, an important factor of PM aimed to enlist occurrence of events that could influence either positively or negatively towards project goals and activities [6]. It was introduced by the Project Management Institute (PMI) and is one of the nine knowledge areas of the Project Management Body of Knowledge (PMBOK) [7]. Generally, a risk management systems are grounded on the identification and assessment of risks. However in software development, risk factors cannot be ignored at corporate business level wherein organization's stakeholders particularly risk managers are involved to analyze risks, develop policies for mitigation and control risks and thus reduce their impact by incorporating appropriate risk management tools and techniques [8].

The predictive analysis of application's risk assessment towards project timeline is the critical task for effective PM. Having an accurate project schedule, especially at the planning phase of software application, may significantly reduce the high risks that are taken during the application development. However, in some cases risk factors are high and substantially affect the project progress. Unfortunately, some existing estimation techniques may not provide required results and project timeline overruns. However, it was found by analyzing various risk factors using ML techniques may offer more accurate

results towards accomplishment of project deadline [9].

Recently, ML plays an important role and achieving remarkable breakthroughs in scientific researches and software applications. For this various ML techniques are evolving rapidly and applied depending on the nature and domain of application for better decision support. ML encompasses an adaptive mechanism which helps to predict the performance of applications through statistical and mathematical procedures [10][11]. In this instance, the increase in application development embracing ML principles lead towards development of higher numbers of intelligent analytical systems.

1.2 Problem Statement

ML is a technique to assist us in different fields and areas like pattern recognition, medical diagnosis, online transportation network and traffic prediction, crime prediction through video surveillance system, computer games, and students record prediction etc. Finding an optimal algorithm and modifying parameters to obtain the optimum model architecture are tough and time-consuming steps in the process of building an effective ML model. Different ML techniques are employed previously in different domains to address few risk factors but no such methodology has been created that co-relates ML and PM with risk assessment to predict project deadline. Therefore, to predict the risk assessment inside software projects, we have applied the following ML techniques i.e., ANN, LR, NB, RF, and K-NN to predict the accuracy of risk in any SPM application's timeline.

1.3 Aims and Objectives

This research mainly focuses to achieve following objectives:

- It provides a comprehensive analysis of current research trends in the field of PM by incorporating ML techniques.
- It provides a review of state-of-the-art ML techniques including their strengths and limitations while reviewing risk factors in PM applications.
- It provides a taxonomy for the parameter optimization techniques on the basis of comprehensive literature review.

- It provides experimental analysis to help in applying various ML approaches to analyze and evaluate the progress of projects at run time environment effectually.
- It provides a guideline to track and monitor progress of project's milestones using BI.

1.4 Research Contribution

The main contributions of this research are:

- By incorporating ML techniques, helps to analyze massive amount of data and derive predictive insights. For this, five different ML techniques are applied namely ANN, LR, NB, RF, and K-NN in this work.
- With PM, ML is helpful for increasing the potential and improve various forms of multidimensional analysis of data to yield specified results.
- In order to apply ML techniques to predict risk in two selected datasets, initially data preprocessing is performed i.e. filled empty/null values, rename features, and conversion of multivalued to binary classification in Dataset2 to enable feature engineering.
- In feature engineering one hot encoding, features selection, and standardization is performed on numerical features to bring all features to a comparable scale without distorting the variations in the values' ranges.
- In addition hyperparameter tunning is applied in each ML technique to optimize parameters and find best results in terms of accuracy.
- Implementation of different ML algorithms can be fairly compared which helps in identifying the best ML model for a given issue.
- Therefore, integration of PM with ML is chosen to improve decision-making process using BI. In this regard, BI can enable users to make fact based maximum decision value, minimum risk decisions, and provide optimal performance in a timely manner.

1.5 Summary

By keeping in view aforementioned, implementation of ML techniques can address variety of risk factors in PM. Moreover, it offers experimental analysis and provides assistance in effectively using different ML methodologies to analyze and evaluate the projects' progress in a real time scenarios. Performance measures through such kind of ML applications will be helpful for organizational strategical and tactical decision-making on project manager and team leads end. Moreover, it can reduce the amount of human labour as many ML developers are spending a lot of time in applying ML techniques and setting parameters to optimize the results especially for complex ML algorithms and large datasets.

1.6 Thesis Organization

Below mentioned thesis organization is presenting the complete structure of the thesis.

- **Chapter 1 - Introduction:**
  This chapter provides an overview of the research topic along with it's main objectives. The main research contributions are also discussed in this chapter.
- **Chapter 2 - Literature Review:**
  This chapter provides a comprehensive review of different ML techniques applied in PM based applications that involve risk assessment.
- **Chapter 3 - Proposed Methodology and Experimental Setup:**
  This chapter provides the proposed research model to predict risk assessment in project timeline management. Experimental setup is showing five ML models which includes Artificial Neural Network (ANN), Logistic Regression (LR), Naive Bayes (NB), Random Forest (RF), and K-nearest neighbors (K-NN) classification. It also discusses different parameters applied on them using benchmark datasets.
- **Chapter 4 - Results and Discussion:**
  In this chapter, results of all applied ML techniques are compared in terms of their accuracy. It also provides the analysis on the basis of the computed results.

- **Chapter 5 - Conclusion and Future Work:**

  This chapter concludes the entire research by discussing the outcome of the research along with challenges and future research directions.

The taxonomy of each chapter is shown in Figure 1.1.

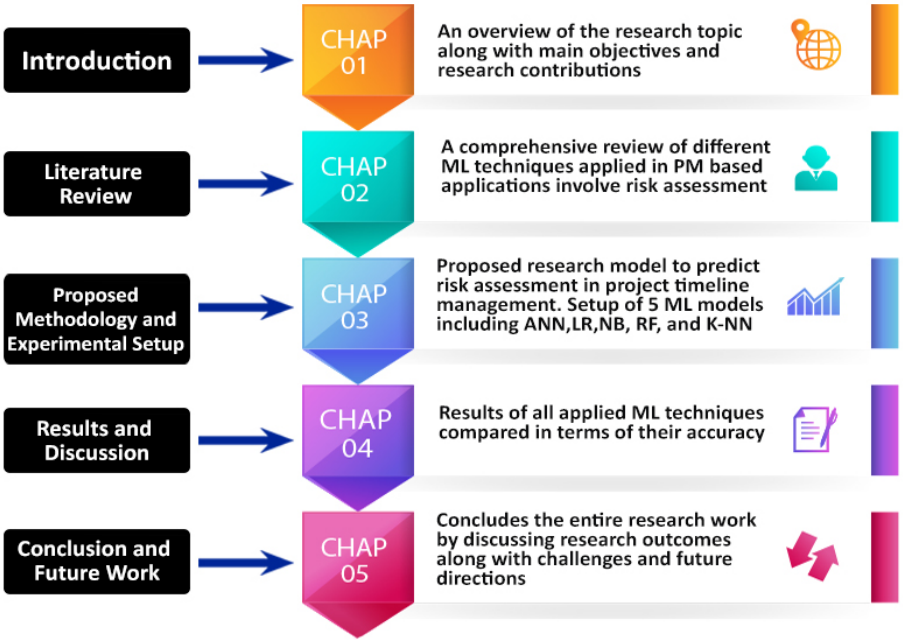

Figure 1.1: Taxonomy of chapters designed for thesis

Chapter 2

Literature Review

Previously for PM applications, different features were used to manage applications such as the classic user interface, carver matrix, eisenhower matrix, activity decomposition, and market analysis [12]. Most of the work had been done for handling the projects with limited scope but other features like risk management, contract management, team management, milestones and task management, project execution management, and prediction regarding on-time project completion are not implemented for diverse kind of PM applications.

In current dynamic environment, organizations are required to handle PM based systems with different level of uncertainties at run time. In this case, risk assessment offers a wide approach in order to deal with risks and their consequences [13]. Risk identification, risk analysis, and risk evaluation are the three levels of risk assessment that must be carried out for software projects, according to International Organization for Standardization (2018). The process of identifying potential risk components is known as risk identification. Risk analysis is a procedure to understand, define, and estimate the level of risk. Risk evaluation is the process of relating the outcomes of risk analysis with risk criteria to evaluate either the risk and its level of severity is acceptable [14].

Numerous elements or situations that may cause a major threat to the project's successful completion are considered risks in software projects. The process of identifying and categorising risk entails calculating its importance, assessing its probability of occurrence, assessing its potential impact on project performance, and formulating policies to address it. Additionally, it might be a breakthrough in management procedures to determine which operations require more attention in light of potential risks and what

decisions to make if the situation would occur [15]. Therefore, keeping in view above factors, based on the technical and integrated approach software project would maintain their sustainability by organizing the architecture for the development processes effectively.

PM monitoring and control applications are dedicated to measure progress on regular basis. In this case more expertise is required to deal with ambiguities and uncertainties of data to develop an integrated PM application. To deal with this scenario ML, an application of Artificial Intelligence (AI) has evolved significantly in the context of data analysis and computing, enabling the applications to operate in an intelligent and smart manner particularly [16]. The inventor of ML, Arthur Samuel defined ML as a "field of study that gives computers the ability to learn without being explicitly programmed" [17]. ML has emphasized on classification and prediction, in view of known features already understand and learned from the training information guidelines [18].

In many practical application areas, such as pattern recognition, medical diagnosis, online transportation network and traffic prediction, crime prediction through video surveillance system, computer games, students record prediction, facial recognition, etc., ML algorithms have demonstrated considerable importance [19]. This is primarily true in fields where managing huge datasets is necessary, and when investigating some form of consistency is essential, and a computer must adjust to changes on its own [18]. Various ML techniques, such as supervised, unsupervised, semi-supervised, and reinforcement learning, are used depending on the domain and data characteristics.

2.1 Supervised Machine Learning

By using examples provided from outside sources, algorithms that are capable of generating broad patterns and hypotheses are created through supervised ML [20]. An input is often mapped to an output via supervised learning, which is based on examples of input-output pairs. It uses a range of training samples and labelled training data to infer a function. In a task-driven method, which is when certain objectives are determined to be reached from a specific set of inputs, supervised learning is carried out [21].

Linear Regression, Artificial Neural Network (ANN), Naive Bayes (NB), Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), K-Nearest Neighbors

(K-NN) are some of the widely used techniques in supervised learning [19]. In general, these ML approaches and methods provide system's ability to automatically learn from experience and improve it without being explicitly programmed.

Unremitting variables are modelled using Linear Regression, and predictions are then made. Examples include are predicting exam results for students, estimating real estate prices, predicting changes in the price of stocks on the stock exchange, and estimating market sales [22]. Regression produces an output by adding the inputs multiplied by a set of constants. A straight line (regression line) is applied to the dataset to establish a correlation between the dependent variable Y and the independent variable X, which may be several independent variables, while relating the linear variables in the dataset is shown in Figure 2.1 .

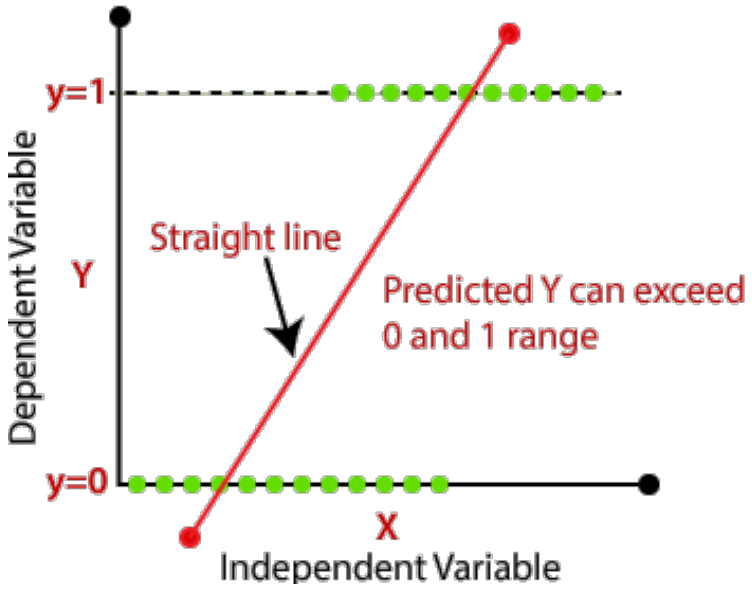

Figure 2.1: Linear Regression, creates a correlation between the dependent variable Y and the independent variable X. Variable X may have several independent variables

Applying regularizing in linear regression makes it simple to avoid overfitting. The disadvantage of linear regression is that it does not perform well with non-linear relationships. One dependent variable often depends on a number of independent variables. Basic linear regression, to put it simply, examines correlations where the input and output variables are one to one. However, the link between a single dependent (output/response) and number of independent (input/predictor) variables are dealt with in multiple linear regression [23]. The regression may not necessarily produce better

predictions if more input variables are included.

Multiple regression, partial correlation, and tabulation methods are used to create a multivariate methodology. Multivariate strategies complicate statistical methods and statistical models [22]. For statistical modelling, a large sample size is required to increase the confidence level in finding better possible outcomes. The evaluation of automobiles, demand for power, the value of real estate, quality assurance, quality control, and medical diagnostics can all be predicted using regression analysis with many variables.

LR, a method for supervised learning, provides a solid method for handling a binary classification issue. Depending on the values of input variables, it predicts the likelihood of an outcome having only two values (in terms of 0 and 1) and provides the binomial outcome, but it requires a lot of iterations and takes a while to train bulk of data [24]. LR prediction in terms of 0 and 1 is shown in Figure 2.2

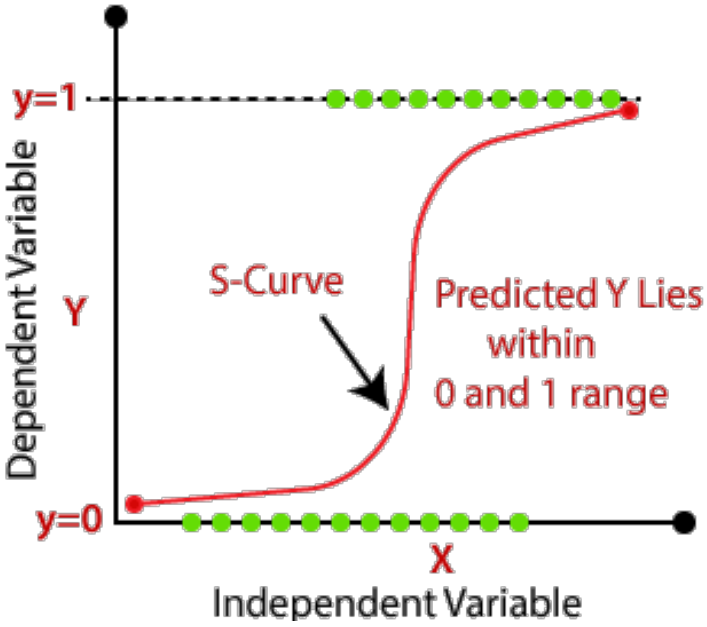

Figure 2.2: LR, predict the likelihood of an outcome having only two values i.e. 0 and 1

For instance, whether or not an email is labelled as spam is an example of a binomial result. An LR study may also yield multinomial outcomes, such as a prediction of the chosen fabric for clothing, such as Chinese, Pakistani, Italian, etc. An ordinal result, such as a product rating from 1 to 5, is another possibility. Therefore, the goal of LR is to predict a categorical target variable. Additionally, it deals with prediction values for continuous variables, such as forecasting store prices over a five-year period.

The application of LR helped to improve regularization ease, training efficiency, and computing efficiency. LR do not require any scaling of the input features. The majority of the time, this method is employed to resolve business- or industry-scale issues. However, because of its linear decision surface, tendency for over fitting, and inability to perform adequately without the identification of all independent variables, LR is unable to tackle non-linear problems. Some instances of practical application of LR includes the likelihood of contracting specific disease, cancer diagnosis, and in engineering, the probability that a product or system would fail [25].

DT, a common classification algorithm, uses continuous data partitioning depending on a predetermined parameter to solve classification and regression issues wherein data is divided into nodes and leaves respectively [21]. The decision variable in a classification tree is categorical (the outcome is expressed as a yes/no response) whereas in a regression tree it is continuous. DTs can be applied to situations like project timeline prediction, tumour diagnostic prediction, and future book use in the library etc. DT for project timeline is shown in Figure 2.3

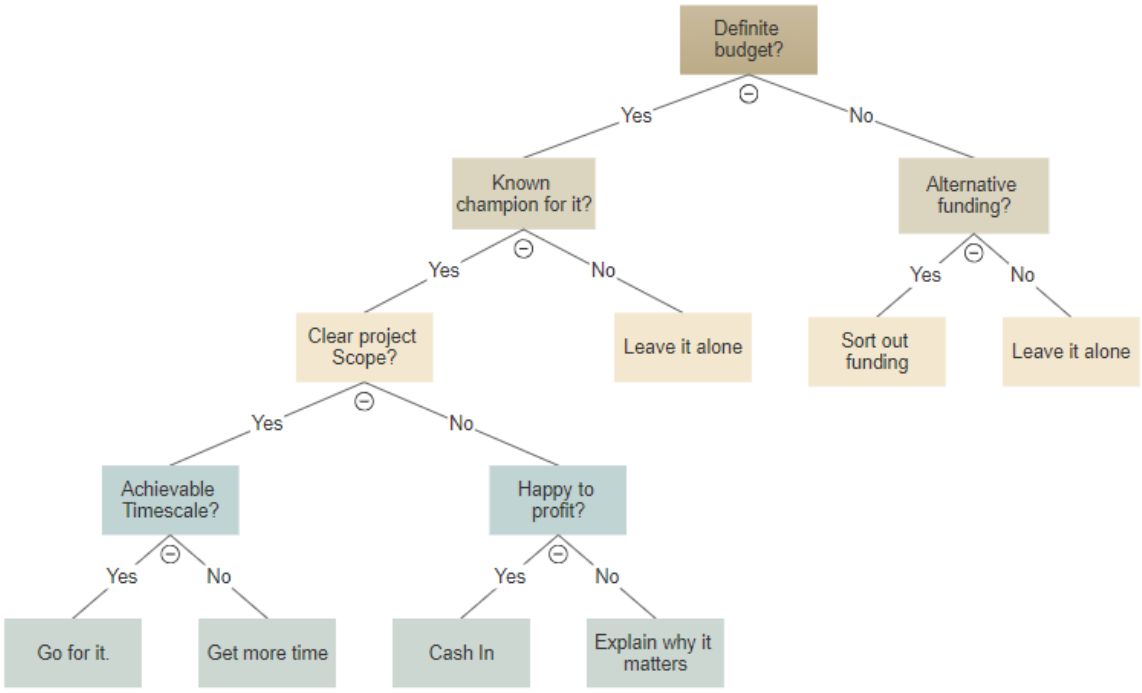

Figure 2.3: DT prediction for project timeline in PM based application

Due to the effectiveness of the tree traversal method, the DT offers simplicity in handling categorical and quantitative data, ease of interpretation, high performance, and the ability to fill missing values in attributes with the most likely values [26]. RF, based

on an ensemble modelling approach offers the solution to the over-fitting issue that DT may provide. Additionally, extreme gradient boosting (XGBoost) and extra trees (ET) models have been created to improve model performance by combining different DTs.

To solve classification and regression problems, SVM can be used. The principle of linearly separating data points by translating them from low-dimensional to high-dimensional space is the basis of SVM algorithms. Graphically, the categorization boundary is then created as a hyperplane to separate the data points is shown in Figure 2.4

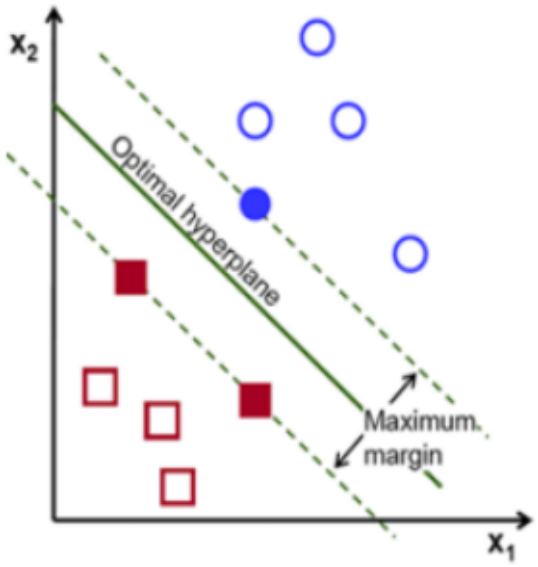

Figure 2.4: SVM Graphical Illustration, determine how similar two data points x1 and x2 are to one another

The kernel function f(x), which is employed in SVM models to determine how similar two data points x1 and x2 are to one another, can be chosen from a variety of kernel types [27]. As a result, adjusting the kernel type hyper-parameter would be essential. SVM frequently employs sigmoid kernel, radial basis function (RBF), linear, and polynomial types. The target of SVM is to correctly identify the objects from the training data set by using examples. If the right kernel function is determined, SVM can handle complex functions and both structured and semi-structured data [28]. Since SVM adopts generalisation and can scale-up with big scale data, there is less chance of over-fitting in it.

SVM performance declines with large data sets as a result of an increase in training time.

However, finding an adequate kernel function is also a challenging element. SVM cannot work well when the dataset is noisy. As this technique cannot provide probabilities in its output, it is difficult to comprehend the final SVM model. On the other side, SVM helps in a variety of tasks including text categorization, face detection, handwriting recognition, and credit card fraud detection. Therefore, SVM may be tested when there are many observations and features [29].

NB, a different supervised learning method is based on the Bayes theorem. As a model this method uses a probability table, which is updated using training data [30]. In real-world applications NB is used for classification of medical conditions i.e. cancer recurrence prediction, and for other applications like weather forecasting, customer credit assessments. NB can handle binary and multi-class classification problems and performs well when using minimum training data [31]. When there are more predictors and data points, it scales linearly. However, it is challenging to apply NB directly when one of the features is required to be a "continuous variable" like time [19]. In comparison to SVM or simple LR, NB requires more runtime memory to make predictions. It is computationally rigorous, particularly for models involving several variables.

K-nearest neighbor (K-NN) is an easy to use ML technique. This technique attempts to create categories of sample data points that is supplied to it as a classification issue using a database that encompasses data points divided into various types [32]. K-NN is considered non-parametric since it cannot take into account any data distribution parameters. It is a highly flexible form of categorization technique that performs well for classes that use many modalities. K-NN, however, is not appropriate for categorizing pricey and unidentified records. The size of the training set increases as the method becomes more computationally intensive and its accuracy decreases when there are too many distracting or irrelevant features.

As managing huge bulk of data require expensive calculations, more dimensional data in K-NN cause a drop in region accuracy [33]. K-NN can be employed in the diagnosis of a variety of illnesses, the prediction of health status, the modelling and detection of biosystem chronic disorders, and the recognition of human action, and handwriting recognition [34]. For instance, if there are two categories: Category A and Category B, and we have a new data point x1, which category does this data point relates to? We require a K-NN algorithm to address this kind of issue which makes it simple to

determine the category or class of a given dataset. This scenario is shown in Figure 2.5.

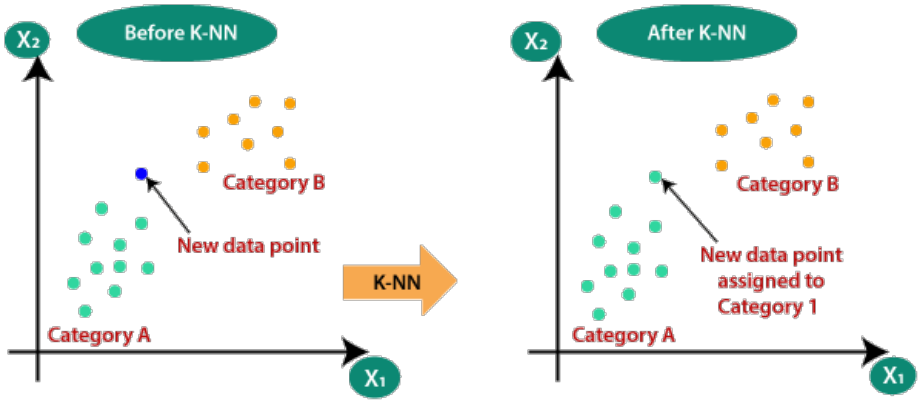

Figure 2.5: K-NN, category A and category B with data points

In recent years, ANN have gained popularity due to its superior presentation. A group of artificial neurons coupled together form the ANN [35]. Each connection is given a weight, and depending on the problem's complexity, each neuron's output is either a linear or a nonlinear function. ANNs have historically been built using the error back propagation (BP) algorithm. Three layers make up a standard ANN: an input layer, a hidden layer, and an output layer. Such neural networks acquire an assortment of input features at the input layer, which are subsequently converted into output features by tying together hidden layers, an activation function, weights, and biases [36]. The cost function is the dataset's difference function between expected output and actual output, which is used to modify the weights and biases using the gradient descent methodology. The BP algorithm then feeds the neural network with these adjusted weights and biases. In order for an ANN with multiple layers to get an ideal result, the input patterns must be repeatedly performed while the weights are being changed, is shown in Figure 2.6.

Researchers have recommended modifying the network structure to address these problems [37]. Modifying the activation function such as the cost function (Mean Square Error, Rastrigin, etc.), Weight-and-Structure-Determination (WASD), and swapping out the BP algorithm with meta-heuristic algorithms are common alterations.

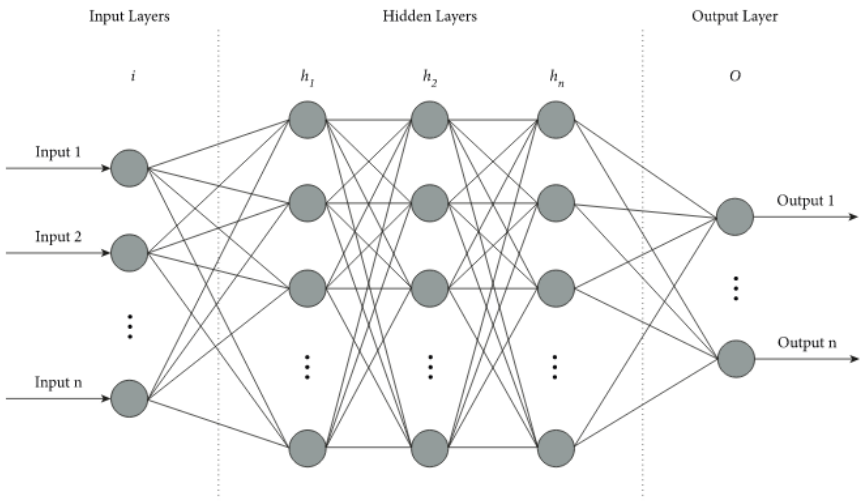

Figure 2.6: ANN, creating a network with input, hidden, and output layers

2.2 Unsupervised Machine Learning

ML techniques known as unsupervised learning algorithms use a set of input vectors (x) as training data without any associated target values. Unsupervised learning, or a data-driven technique, is used to find previously unidentified patterns in unlabeled datasets without the need for human interaction [38]. Clustering and dimensionality-reduction algorithms are the two main categories of unsupervised learning techniques. Unsupervised learning approaches include using a similarity measure to group data points and dimensionality reduction to project high dimensional data to smaller subspaces. Clustering is used in unsupervised learning to find applications for anomaly detection [39]. Two well-known examples of unsupervised learning techniques are the use of clustering e.g. to predict heart illness and the use of principal component analysis (PCA), a dimensionality reduction method e.g. to predict hepatitis disease [40].

In PM applications, unsupervised learning is typically employed before supervised learning while performing exploratory data analysis in order to generate classes based on groupings. Clustering is one example of a data compression technique that can be applied. Every dataset contains clusters, and distance-based clustering can be more successfully applied to large data sets using unsupervised feature ranking [41].

Clustering algorithms are divided into a few categories that are specifically exclusive, overlapping, hierarchical, and probabilistic. A data point may only be included in one cluster according to the grouping method is called as exclusive clustering. The

K-means approach, which divides data points into K groups based on how far apart they are from each other's centroid, is a good example of exclusive clustering [42]. The data points that fall into the same category are those that are closest to a certain centroid. Greater K values indicate smaller groupings and higher granularity whereas smaller K values indicate bigger groupings and less granularity. Market segmentation, document clustering, image segmentation, and image compression all frequently use K-means clustering [43].

Hierarchical clustering, also known as hierarchical cluster analysis (HCA), is a type of unsupervised clustering that can be categorised as either agglomerative or divisive [44]. Agglomerative clustering is a "bottom-up technique" in which data points are initially separated into distinct groupings and once one cluster has been obtained, they are progressively combined based on similarity [45]. The opposite of agglomerative clustering is referred to as divisive clustering which adopts a top down approach in which divisions between data points inside a single data cluster are made [46]. Even though divisive clustering is not frequently employed, it is important to be aware of in the context of hierarchical clustering. A dendrogram, which resembles to tree-like figure is frequently used to illustrate clustering processes, can be used to demonstrate the merging or splitting of data points at each iteration is shown in Figure 2.7

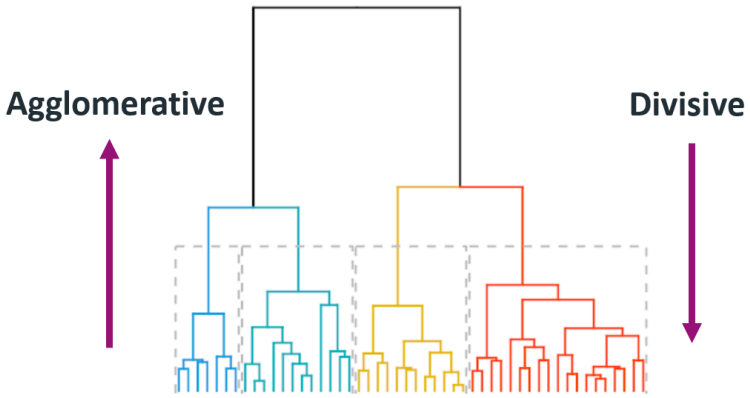

Figure 2.7: Cluster Dendogram, a tree-like figure, which is typically used to display the clustering processes

Density estimates, often known as "soft" clustering problems can be solved using an unsupervised technique called a probabilistic model. In probabilistic clustering, data points are categorised based on how likely it is that they will fit into a particular distribution. One of the most popular probabilistic clustering techniques is the Gaussian

Mixture Model (GMM), falls within the category of mixture models, which is formed by arbitrary number of probability distribution functions [47]. The main application of GMMs is to identify the Gaussian or normal probability distribution. Its illustration is shown in Figure 2.8

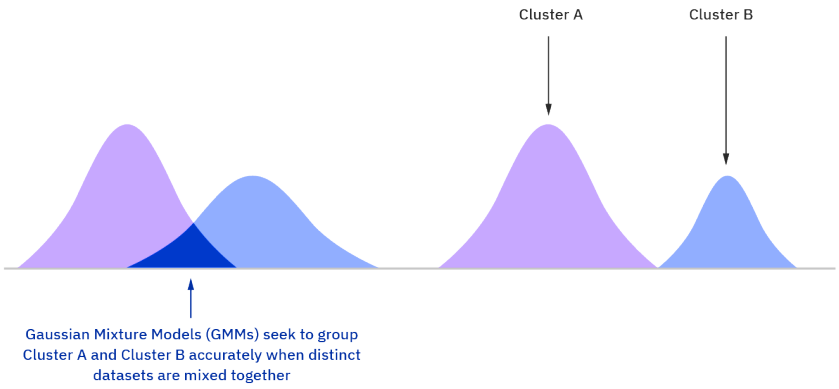

Figure 2.8: GMM, when combining different datasets trying to organize both clusters appropriately

If the variance or mean are known, we can determine which distribution a given data point relates to. Since these factors are unknown in GMMs, we presume that a hidden variable exists in order to cluster data points properly. The Expectation-Maximization (EM) algorithm is frequently used to estimate the assignment probability for a given data point to a specific data cluster. By using the final clusters, the EM algorithm is targeted to maximize the overall probability of data [48].

A rule-based approach for connecting variables in a particular dataset is known as an association rule. Huge volume of data is analyzed using association rule mining to uncover linkages and relationships. This rule displays the number of times an itemset appears in a transaction e.g. a market-based analysis serves as a common illustration [49]. Other alternative algorithms including FP-Growth, Apriori, Eclat are employed to produce association rules. Among them, Apriori algorithms have gained popularity through market basket analysis. A hash tree is used by apriori algorithms to count itemsets while they traverse the dataset in breadth-first manner. By creating a large number of candidate itemsets, the Apriori algorithm determine frequent itemsets [50].

Unsupervised techniques can be used to achieve dimensionality reduction, which refers to the techniques used to represent fewer characteristics of data in a dataset while retaining as much variance as is feasible in the original dataset [51]. The sparse latent structure decreases the number of attributes, which reduces the amount of data processing needed in the future and gets rid of redundant features. Data can be changed from one modality to another in other contexts by employing dimensionality reduction. Therefore, it effectively improves the overall performance of the ML algorithms [52].

The dimensionality reduction technique maintains the dataset's integrity as much as possible when a dataset has an excessive number of dimensions and features while lowering the number of data inputs to a suitable level. There are a few dimensionality reduction techniques that can be followed, including Principal component analysis (PCA), Singular value decomposition (SVD), and Autoencoders [53].

Principal component analysis (PCA), an established linear technique is a form of dimensionality reduction technology that use feature extraction to decrease duplication and compress datasets [54]. The direction that gain the variance of the dataset is the first principal component. The largest variance in the data is comparably found by the second principle component, but it is completely unrelated to the first principal component and produces a dimension that is orthogonal, or perpendicular to the first component. First and second principal component with 3-covariates setting is shown in Figure 2.9

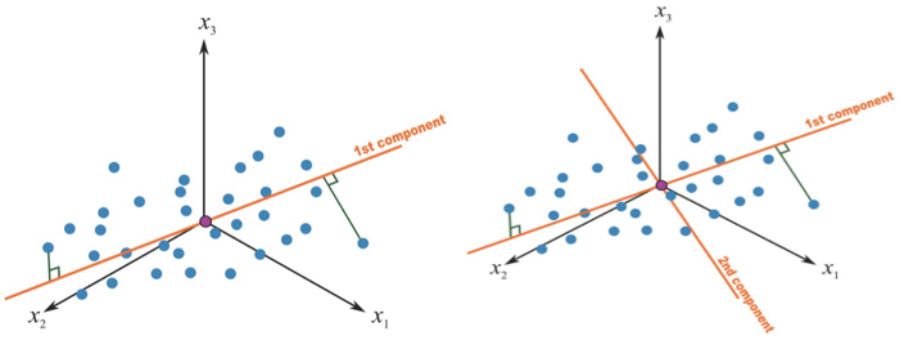

Figure 2.9: First and second principal component in a 3-covariates setting

Depending on the number of dimensions, this process is repeated, with each subsequent main component pointing in the opposite direction from the previous component with the highest variance [55].

Another dimensionality reduction technique is singular value decomposition (SVD),

which factors a matrix A into three low-rank matrices. Many applications like PageRank algorithm used by Google, and Netflix's recommender system, an image compression make use of SVD [56]. Formula used for SVD, A = USVT where S is a diagonal matrix, and its values are considered singular values of matrix A, and U and V are the orthogonal matrices. SVD is frequently used for data search and image processing [57].

Most often used for dimensionality reduction and unsupervised pre-training of feedforward NNs, an autoencoder for ANN compresses and encodes representations of data. Because of its higher versatility, it is frequently chosen [58]. They are often trained using BP and stochastic gradient descent (SGD) approaches and developed using approximation functions. When autoencoders compress the input layer before reconstructing it in the output layer, the hidden layer specifically acts as a bottleneck. When training NN, autoencoders learn the input data function by reproducing the input at the output (a process known as encoding/decoding). In essence, a straightforward autoencoder uses recurrent patterns that are similar to low-dimensional representation of the input data [59]. Autoencoders is shown in Figure 2.10 representing its input, hidden, and output layers.

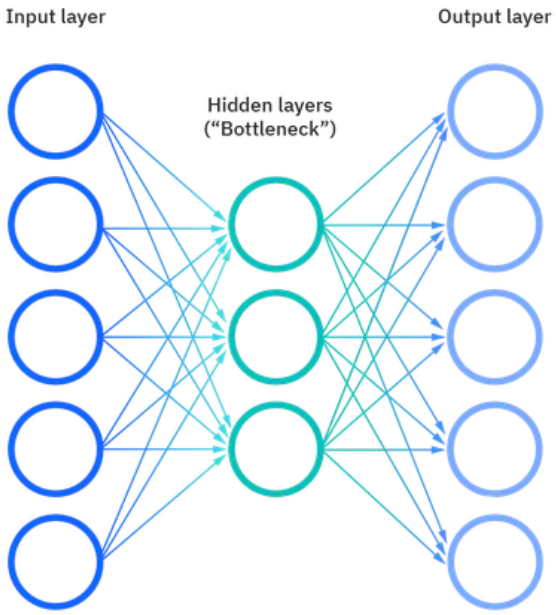

Figure 2.10: Autoencoders with input, hidden and output layer wherein hidden layer act as main bottleneck

Autoencoders came in different forms that have been effectively used in an extensive

range of applications like natural language processing (NLP), speech recognition, and computer vision [60]. It is determined that model-based collaborative filtering, unsupervised ML, and the categories utilized in it are used to improve prediction accuracy and accelerate processing.

2.3 ML and PM

Earlier, researches had been done on web based applications linked with ML techniques like web application attacks detection using ML techniques [61], effort estimation of web-based applications using ML techniques [62], implementation of ML algorithms into embedded systems with web application user interface [63], and survey on detection and prevention of web vulnerabilities in ML context [64]. ML techniques are evolved in projects and play an important role in web application's measured and statistical work, i.e., modules progress [65] which helps to analyze, predict and evaluate the project's progress, enhance the adaptability level for organizational processes towards continuous development. Pasupuleti et al. suggested applications of ML techniques in software PM have been broadly developed and applied in various fields such as healthcare, finance, manufacturing, electronic commerce, automotive industry, and entertainment. Significant improvements in software platforms for ML development have served as the foundation for these practical applications [66].

According to Shaikh et al., ML projects that apply predictive analytics, it is important for a project manager to "understand the experimental nature of discovery and development while balancing the requirements against organizational constraints to ensure that experimentation drive towards a solution-oriented deliverable that can benefit the organization" [12]. Strong communication abilities and understanding of the effects of changes are needed to balance the competing stakeholder interests. The project manager must also be able to assemble group of individuals with divergent priorities and assure that they are working towards a unified objective.

The main responsibilities of PM, from an operational standpoint are to establish milestones and coordinate tasks and activities, communicate with stakeholders, identify and recruit skilled personnel, and ensure effective delivery of projects based on planned strategy [67]. Uysal et al. commended that many contemporary organizations pro-

duce business value through initiatives to adapt to the dynamic and cutthroat business environment, according to the perspective of digital transformation. PM plays a significant part in the organization and growth of corporate innovation processes since it has implemented effective technologies to promote innovation [68].

To complete the project's objectives within the imposed timeline is the main challenge facing by the project manager. The hardest part of their task may generally be determining the exact timeline for project completion. The management of software project timelines can be progressive significantly with the use of ML. For this, Wei Wang et al. tested ML with PM software like ClickUp to predict potential user behavior [69]. Based on this, Kumeno et al. suggested that the project timeline management utilizing ML technology should be able to predict and assign tasks to the appropriate team members, automatically tag users that are pertinent to them, visualize notifications and updates based on their relevance to a particular user, and from the aforementioned criteria predict and determine the deadlines and adjust task time estimates accordingly [70].

Given that many project metrics can be categorized as soft characteristics that are difficult to quantify, it can be difficult to assess how predictable a project's metrics are. Additionally, each project is distinct, which makes it challenging to develop a universal strategy for quantification. Despite these challenges, a general measure has been created based on discussions with the experts for every critical project metric [71]. When the prediction tool is created and a dataset is acquired, the specified measures and metrics are used as input. The prediction tool then eventually produce a classification that can be either success or failure. Additionally, it allows for the effective management of project execution and the preservation of the expertise of experts within businesses. Anie Bermudez et al. claimed that when ML is used for project evaluation, companies are able to adapt to changing management styles as a result of their capabilities and ongoing progress [72].

According to Marchinares et al. by applying an intelligent PM tools and ML techniques, the users may be given the right guidance, which makes it much easier for managers to extract critical success factors and make effective decisions and enable the software being developed to expand into additional markets [73]. The timeline management of projects still has a lot of flaws in small and medium-sized businesses today. Using con-

ventional project schedule management techniques, it is challenging to deliver finished projects to customers within the specified time [69]. By incorporating ML techniques into software PM, managers can receive useful guidance that will enable them to plan the software development cycle rationally and, ultimately, deliver finished projects to customers within the specified time.

2.4 ML and PM with Risk Assessment

The practical ML approach towards PM software deals with the goal to assist project manager to develop project plans, identify, and overcome risks in an early stage of the project. According to M. N. Mahdi et al., project risk assessment using ML is more effective in minimizing the loss of the project, increasing the likelihood that the project will succeed, offering a different ways to effectively reduce project failure probabilities, increasing the output ratio for growth, and facilitate analysis on software fault prediction based on accuracy [9].

In previous researches, ML techniques were applied to assess risk in various domains. Like in banking sector including loan risk, credit risk, liquidity risk, market risk, and operational risk. Dataset was created for the construction industry, and ANN approach was applied to give evidence-based decision-making, utilising interdependent, active, and dynamic risk factors for formulating suitable project risk management strategies. The study's findings assisted managers and decision-makers in determining the risk associated with managing forthcoming events under atypical circumstances, which increased operational effectiveness. Additionally, it functioned as a valuable knowledge repository for planning and predicting the effects of risk driven policies, which helped to significantly increase the robustness of sustainable business operations [13] [74].

At industrial area, deep neural network (DNN) was applied against cyber-crimes to assess security risk measures to protect system networks, hardware, and software from digital attacks. The result of the predictive analysis ensured cybersecurity professionals to make proactive decisions in order to prevent cyber-attacks and security threats. In transportation area, ML helps to predict routes and recommend to follow different paths. Eventually, these learning-based data-driven models help to improve the traffic flow, enhance the usage and efficiency of sustainable transportation modes, and decrease real-

world disruption by modeling and visualizing future changes. The established model could predict risk increase (or decrease) with respect to the change in route system, in order to provide proper support for decision making process [19]. Moreover, there is involvement of ML techniques in health care decision making i.e. to analyze complex data, medical reports, and medical images to identify risk factors which became cause of different disease [75].

ML, the predominantly area of AI has been a key element of digitalization solutions that has drawn a significant attention in the digital arena. Various ML techniques are currently being utilized to support risk assessment phases such risk identification, risk analysis, and risk evaluation. Jeevith Hegde et al. has compiled work done on different ML methods for engineering risk assessment wherein ANN, SVM, DT, NB are used in various applications' sectors like automotive, aviation, construction, cyber security, energy, environmental engineering, healthcare, geology, maritime, railway, road safety, oil and gas, urban planning, and workplace safety [76]. Both historical and real-time data are being applied in different articles to develop ML models capable of providing inputs to traditional risk assessment techniques. With over 20% of articles published, the automotive sector is in the forefront of the use of ML for risk assessment. The construction industry is closely behind with over 15% of articles published. The most widely used ML algorithm for risk assessment is artificial neural networks (ANNs), followed by support vector machines (SVMs). More than 70% of the publications constructed the ML model using historical information, while more than 20% use real-world scenarios. One-fourth of the proposed techniques have been applied in a real-time situation, and around half of them use a case-study based approach to develop ML models. It was concluded, that the proposed ML is most frequently used to support the risk identification phase of risk assessment. However, the various safety regulatory agencies must address ways to confirm the use of ML in risk assessment.

In 2018, Zain Shaukat et al. worked on the dataset that contains required data from four various SRS sources that have been verified by IT professionals. The proposed dataset's primary goal was to offer a data source for risk decision support systems in order to improve risk prediction at the outset of the software development life cycle [77]. This dataset contains 5 levels of risks, 14 attributes, and 299 instances. Various ML approaches are used on the proposed dataset to estimate the project's risk. The dataset was validated using a K-NN classifier, and the accuracy rate was 52.8%. Three

preprocessing filters were additionally used to increase accuracy. Because most of the values in the proposed dataset are categorical, both normalization and standardization produced results with equal accuracy. In addition, k-fold cross validation is applied with K-NN 10 folds. Various aspects of software project development such as software risk prediction, requirement and risk prioritization, effort estimation, and cost estimation can be supported by the suggested dataset.

In 2021, Zain Shaukat et al. worked on empirical assessment of ML methods for software requirements risk prediction [78]. Work was done on same aforementioned dataset wherein different ML techniques were applied like K-NN, Naive Bayes (NB), Decision Tree (DT), and Random Forest(RF) for which 58.19%, 90.97%, 97.99%, and 83.27% accuracy was gained respectively. For each classifier, 10-fold cross validation was performed to assess the performance of the model. The proposed dataset can be expanded by new requirement specifications for further software projects. Additionally, it might create new challenges for predicting risks throughout the requirement gathering process.

Another research work was done using ML models which encompasses the risk features of the software project. The objective of this study is to offer a methodology for software requirement risk prediction utilizing a requirements risk dataset. To identify a better solution for risk prediction in software requirements, the suggested model is benchmarked against seven different ML techniques. This dataset contains 5 levels of risks, 13 attributes and 253 instances. A model is proposed based on AdaBoostMl and J48 (ABMJ) for risk prediction in software requirements [79]. The outcomes of the proposed ABMJ are compared with seven different ML techniques. The accuracy found with ABMJ was 97.62%, Cost-Sensitive Decision Forest (CSF) with 77.47%, Naive Bayes (NB) with 90.9%, Multilayer Perceptron (MLP) with 62%, Decision Tree (J48) with 96.83%, Average One Dependency Estimator (AODE) with 90%, Random Forest (RF) with 82.6%, and Support Vector Machine (SVM) with 62% of accuracy. The 10-fold cross validation was performed against each classifier to assess the effectiveness of the model.

Based on the results of the aforementioned research project, it was determined that several ML techniques are compared to the suggested ABMJ using a few performance evaluation indicators. However, there is a chance that new approaches may exhaust the tactics that have been developed if they are added. Using the most recent performance,

evaluation metrics resulted in improved outcomes than the existing findings, which is very supporting.

Mohammad Ibraigheeth et al. worked on the risk assessment of software project using ML approaches [80]. The purpose of research was to develop an evidence-based risk assessment model that use historical failure data from several past software projects as a training data to effectively assess software project risks using ML. Six core failure factors that were used in this work are unrealistic objectives, staff technical problems, lack of users involvement, instability of requirements, problems in the used technology, and management problems in the project. For developing the model, six techniques were applied namely, NB, LR, DT, SVM, ANN and adaptive neuro-fuzzy inference systems (ANFIS). The accuracy provided by NB is 68%, LR with 71%, DT with 75%, SVM with 83%, ANN with 67%, and ANFIS with 82%. To demonstrate the high prediction performance and the ability to precisely pinpoint the contributing factors to software risk, cross-validation analysis was carried out. The performance results demonstrated that the best average prediction performance is provided by SVM and ANFIS. The NB and ANN model has the low average prediction performance. The remaining LR and DT models seemed to be reasonably accurate and deliver respectable performance outcomes.

Andre Oliveira Sousa has created a research work on software projects assessing risks through ML approaches [81]. The dataset for this work is offered by Strongstep, one of the project's partner companies, as a number of spreadsheets containing various project-related data in their project management software SCRAIM. This data set comprises of 18 projects, 140 risk items, and 13 variables. The data was split into two sets: one for hyperparameter tuning validation (40 instances), and the second set (100 instances), which was added to the first set for model training and testing. The dataset was converted from multi-class to binary classification for many test and train cases and tests were run on both of them.

Six different ML algorithms were tested, three of which were deemed interpretable i.e. Gradient Boosted Trees (GBT), RF and NB and three of which were deemed non-interpretable i.e. ANN, SVM, and K-NN with the target to compare their outcomes and determining whether or not the common performance trade-offs in non-interpretable algorithms are justified by the reduced amount of information that can be obtained.

After the conversion from a multi-class to a binary classification problem, it was found that the results of the interpretable models in the risk impact and risk likelihood tests matched and frequently exceeded those of the non-interpretable models. In these tests, two algorithms in particular saw significant declines: Complement NB with 27% in the risk likelihood model testing and ANN with 61% in the risk impact model tests. However, SVM gave the best accuracy for the predictions of risk impact level which is 69%.

M. Hanefi Calp et al. has worked on software projects' risk estimation using ANN [82]. In this research, 20 software projects, and 45 risk factors were used wherein 30% of data is used for testing, and 70% kept for training purpose. Evaluation metrics provided the test accuracy of 99.3% and training accuracy of 99.7%. The suggested model as it took into account the project scope, was found robust enough to predict the risk variables that may be present in software projects in various fields.

Yong Hu et al. applied SVM and ANN ML approaches to establish a model for risk evaluation in project development [83]. A vector of software risk indicators collected through interviews with 30 experts was the model's input, and the project's final result is its output. 120 actual software projects were used as the source of the data for modelling purpose. In dataset, project was classified as "successful", "challenged" or "failed". The accuracy found by applying NN was 70% and SVM with 80% of accuracy. The study and management of project risks were found to benefit from an intelligent risk appraisal methodology.

Wen-MingHan worked on discriminating risky software project using NN which comprises of 40 projects and 22 attributes in OMRON database [84]. In this work, 80% of data was used for training and remaining 20% for testing. Two levels of risk were involved in it i.e. risky or not risky. The main aim of this research was to predict how risk factors are intricate in projects. Two ML techniques were applied in this work i.e. NN and LR which has gained the accuracy of 82.2% and 87.5%. According to experimental findings, the NN method was used to determine the risk factors that are involved in. In particular, this approach outperformed LR model created from the same database in terms of sensitivity and accuracy by more than 33.3% and 12.5% respectively.

Osamu Mizuno et al. led a research work in which a risk-management technique i.e. Software Risk Evaluation (SRE) was used to identify risk for a software development

project wherein risk taxonomy table was employed for systematically identifying project risks [85]. In dataset, project was classified as either runaway or success. Bayesian classifiers were used for experiment purpose and found an accuracy of 82.5% with 10-fold cross validation technique. A model was created for empirical evaluation using actual project data has shown that 33 out of 40 projects were predicted appropriately and reflected the risk factors involved in the projects.

Another research was conducted on probabilistic software risk assessment and estimation model for software projects [86]. The primary aim of the study was to make software risk analysis and determine the relationship between risk indicators and projects. An empirical experiment has been carried out using data collected from software projects used by an organization in order to evaluate the model. In dataset, three levels of risk factors are classified as low, medium, and high risk for which 27 risk factors are assessed in 12 software projects. Bayesian classifiers were used and found value of Mean Magnitude of Relative Error (MMRE) with 0.0384 and Balanced Mean Magnitude of Relative Error (BMMRE) with 0.03911 which indicated good estimation accuracy.

YongHu et al. conducted a research work from 302 software projects collected through questionnaires using 10-fold cross validation for testing [87]. Bayesian networks (BNs) tool is used for various risk management techniques for software development projects has been investigated. In this instance, classification of project's performance based on risks identified as "low" or "high" is used. ML techniques applied in this work are DT and NB which has provided the accuracy of 70.86% and 72.85% respectively. For the risk analysis of software development projects, a model utilizing BNs with causality constraints (BNCC) has been presented. It was proved that the suggested model can not only uncover causalities in accordance with the expert knowledge but also perform well to predict other algorithms, such as LR.

A research work was done by M. Zavvar et al. in 2017 with 530 samples of a data set created from information of software development projects wherein 70% of data was used for training and 30% for testing purpose [88]. In this work, project risk has classified either as low risk and high risk and ML technique SVM was incorporated which found an accuracy of 99.51%. It was determined that the proposed approach reduced developer work and increased accuracy in identifying the harmful risk factors.

Yong Hu et al. in another research has formulated a model for risk identification and collected actual cases from software development companies [89]. For this, 120 projects, and 64 risk factors were extracted wherein 16.7% used for testing and 83.3% of data used for training constraint. In order to evaluate the model performance, two ML algorithms ANN and SVM were compared in which 75% and 85% accuracy was found respectively with GridSearch cross validation which indicated that SVM has achieved better performance in risk prediction model.

In 2015, T. Christiansen et al. has designed a research for which data was obtained from 70 software projects through questionnaires. Based on project risk classification either as risk or non-risk, multiple LR technique was employed consisted of causal risk factors and risk stratification analyses [90]. The main aim of the work was to anticipate and minimize risk through risk factors during software development processes. LR with 90% accuracy resulted in leading the development strategies and highlighted main issues pertaining to manage, control and reduce the risk constraints.

Analytical work was performed by thefonseca in which project data was found from the Microsoft Project Server database. The goal of this effort was to create an ML model that would forecast the possibility that a project would have concerns significant enough to be included in the PM risk report [91]. From 70 projects, 20 features, and by training a discriminative model, projects were prioritized that presented a high risk profile and compact the cost of report elaboration. Most significant factors that contributed to project risk like scope, managers, and seasonality etc. were also explored in this work. ML techniques like SVM, RF, K-NN, and GaussianNB were applied and found that the proposed model is capable to detect a high percentage of risk projects i.e. 70% on test data. Hyperparameter tuning was applied wherein GridSearchCV with cv value 6 is used to predict the performance of the model.

ML along with PM has offered a wide range of applications to be developed on different platforms. It has supported web and software applications in a multi-dimensional way i.e. efficient modularity, high scalability, and usability. Therefore, implementing ML and PM using interactive BI architecture has led to highly efficient and scalable business system [92]. BI is a framework architecture for organizing data into proper business model wherein information management and technology components are used to shape BI systems into data analytics and reporting purpose [93].

Decision making through BI is of two types i.e.traditional and operational BI. In preceding researches, traditional BI was used which includes low level processes with limited user access. In contrast, operational or dynamic BI is used for real time and fact based decision making in organizations at strategic and tactical levels. The role of BI has become more prevalent and affecting the way information is perceived, applied and evaluate [94]. As output organizations can decide, measure, and manage the overall progress and performance of projects at an optimum level. Hence, the target of BI is to gain the maximum courses of action at minimum time.

The BI applications are based on fact based knowledge to enhance business oriented understanding. Both BI and ML has facilitated each other and perform in conjunction at an optimal level to utilize bulk of data for meeting business needs. Here, ML offer mathematical and predictive modeling, statistical analysis, performance management, and advanced reporting features whereas BI applications used to collect, analyze, integrate, and perform certain calculations to provide output which provide guidance for appropriate strategies and decision making [95]. Therefore, both ML and BI aim to improve business understanding and management to compete in a dynamic environment.

2.5 Literature Review Table

The literature review table contains all previous research works that has been done to predict risk assessment in PM applications in shown in Table given below.

2.6 Summary

ML approaches in PM applications are gaining widespread development and used in numerous fields at corporate level. However, connectivity of ML and PM with risk estimation policies are required to be formulated to incorporate risk management strategies in organizations to reduce the level of risk by understanding and practising the effectiveness of ML models in various applications. In this domain, and by keeping in view the aforementioned research works, a proposed model will assist in PM applications to make concrete decisions and successfully achieve the organisational objectives. The proposed methodology and experimental setup on five ML techniques are detailed in Chapter 3.

Chapter 3

Proposed Research Methodology and Experimental Setup

3.1 Introduction

Research on ML techniques with BI integration is yet to be done in web based PM system. For medium and large size organizations, management of projects is quite important in order to keep track of all tasks through some software application platform. To optimize business-level activities and to assess risk in applications, BI tools and processes are used to transform data into information and information into fact-based knowledge [96]. Moreover, ML and BI will provide a guidance to assist in decision making to deal with high risk factors in a very effective manner.The proposed model has elaborated this scenario in Section 3.2.

For experimental setup, before making predictions on a new collection of data, ML models must first be trained on it. To increase the accuracy and predictability of the models, the data that was used to train them was first reviewed and examined for consistency and quality. Different data preparation procedures that are used in the application before using the data to train the models are discussed in Sections 3.3 and 3.4 of this chapter. The characteristics of the data set that was utilized to train the ML models are detailed in Section 3.5.

3.2 Proposed Methodology

On the basis of two benchmarked datasets, a methodology is proposed to predict risk assessment in project timeline management. The attributes in both the risk datasets are those that are connected to the risks and specifications of the software project. The suggested model will allow the project manager and domain experts to easily predict and mitigate the risks at earlier stages before the development phase by using the set of PM-based application requirements as input to estimate the level of risk involved.

Both the datasets are analyzed and compared with the main risk factors like project, risk, project schedule, risk likelihood, impact, report count, dimension and priority etc. Moreover, models are evaluated based on the risk consequences that are occurred in the PM applications. In order to make risk predictions, five different ML techniques were applied and discussed in Section 3.5 and Chapter 4. Proposed methodology is validated and their outcomes are identified in order to make effective decision making which ultimately effect the project's timeline. In addition, ML provides a guidance to assist in decision making whereas BI helps to convert data in meaningful information. The proposed methodology illustrated in Figure 3.1 has shown complete flow.

Keeping in view advancements towards science and technology, ML techniques and proliferation in BI provides the core information in proposed model and permit multidimensional analysis of data via different parameters and procedures adopted in ML techniques which helps to predict the maximum possible outcomes in the form of accuracy. Furthermore, operational BI is involved with concrete specifications, project performance measures, dynamic configurations, and real time alerts to deal with risk factors in run time scenarios. This way, PM applications facilitate project managers to monitor overall project progress dynamically and check whether the project will be completed and meeting the deadlines with no lags and delays.

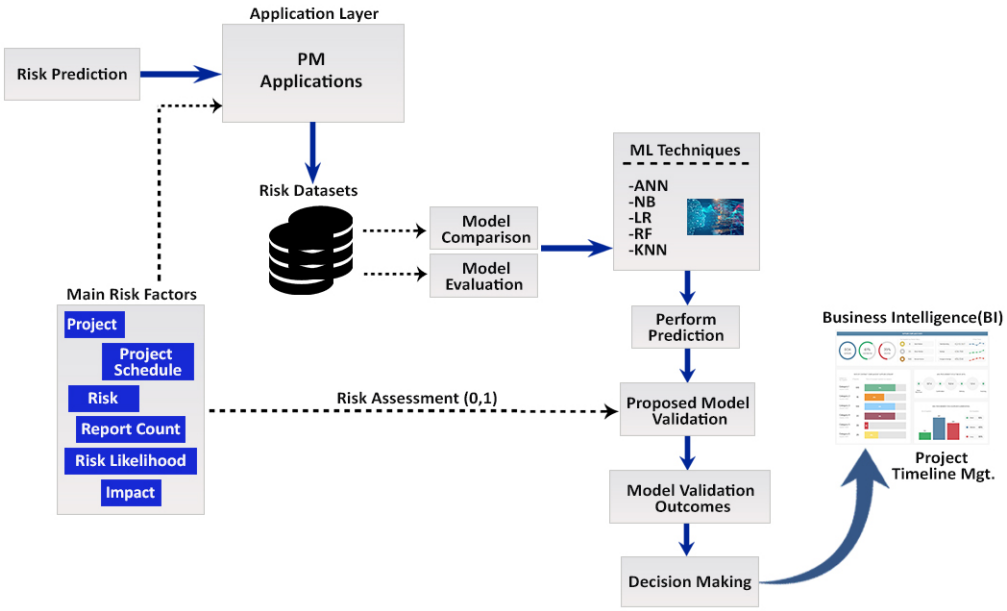

Figure 3.1: Proposed methodology to predict risk assessment in project timeline management. Main elements are risk model analysis, comparison, evaluation, perform prediction, validation, and decision making to meet project timeline goals

In order to analyze both the risk datasets, various experiments were performed to train the models and to find the accuracies in them. Its explanation is given in Section 3.3.

3.3 Datasets for Experimental Work

The dataset is made up of many different pieces of data, however it may be used to train an algorithm to look for predicted patterns throughout the entire dataset. Two datasets were taken in this work to make a comparison for experimental work. Dataset1 is used to predict risk factors that are involved in project risk report and determine either project is suffered from high or low risk factor [91]. Projects with a statistically high risk profile are given priority, and the cost of report formulation is decreased by training the discriminative model. Previous model in which risk act as a central class is shown in Figure 3.2.

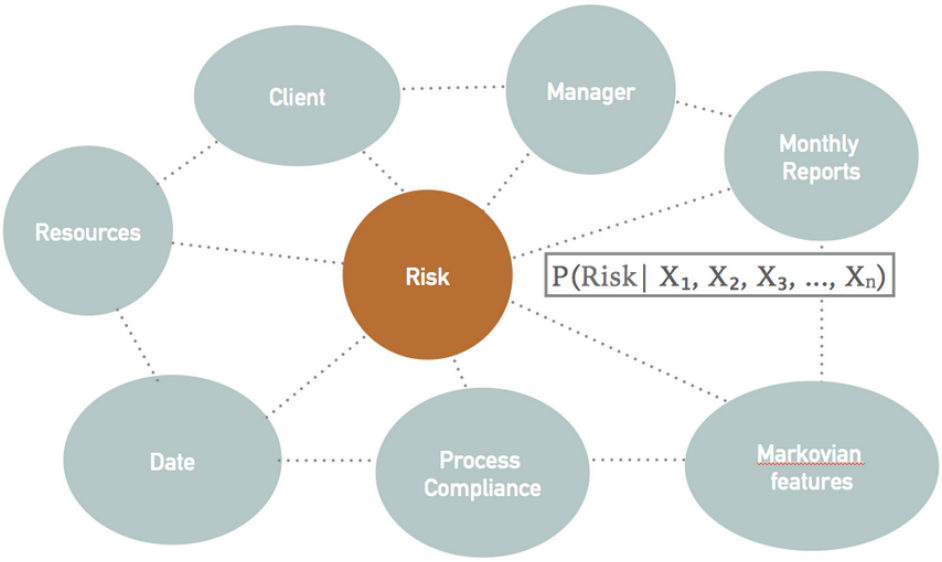

Figure 3.2: Previous model in which risk act as a central class is based on binary classification

The Dataset1 was found from the Chamber of Deputies' corporate projects' wherein data was extracted from Microsoft Project Server. It contains total 1022 records with 20 attributes and is based on seventy projects. A dependent variable (main class) "risk" is based on binary classification.

Dataset2 is used during the requirement gathering phase to determine software risk factors [78]. This dataset comprehends specification from the Software Requirement Specification (SRS) of 4 different open source projects and comprises of total 299 records with 14 attributes. A dependent variable (main class) "Risk" is based on multivalued classification i.e. ranges from 1 to 5.

Both of the datasets were analyzed completely and checked numerical data, categorical values, time series data, and text data for preprocessing before applying ML techniques. These techniques were carried out using the Python libraries Numpy, Pandas, and Scikit-Learn. The Numpy and Pandas packages provided useful utility functions for common activities like loading the data set and extracting the required variables, while Scikit learn was used to create the ML models and perform data preparation tasks before the models were trained, such as encoding and standardization.

3.4 Data Preprocessing

Any data mining and ML-oriented challenge must be successfully preprocessed in order to be solved. Data preprocessing was performed to remove impurities wherein data cleaning technique was adopted in Dataset2 as few of the feature names were creating conflicts with data samples. In Dataset1, there were empty/null values exists against one feature i.e. Compliance, for which median of categorical values was taken and incorporate values in that feature. This way both datasets became organized by removing anomalies or duplicate records to improve the quality of data and efficiency of the models. Both models' strengths and weaknesses became prevalent by comparing not just their predictions but also the class probability estimations for risk effect class. As Dataset2 was multivalued as discussed in Section 1.4 , but in order to compare both datasets effectively the dependent variable "Risk" was encoded to "0" and "1" wherein Risk values 1 and 2 are encoded to "0" (low risk) and Risk values 3,4 and 5 are encoded to "1" (high risk).

Feature engineering was applied wherein categorical values in both datasets are encoded using a one hot encoding. The range of the values of the numerical independent variables was also scaled using feature scaling. This is an important phase because if one attribute's range of values shifts noticeably more than another attribute's, the former will become dominant the latter in the data set and therefore creating an anomaly and lower the models' ability to predict performance of the models. For this, StandardScaler was applied on numerical features to scale all values between min and max so that they fall within a range from minimum to maximum values.

Moreover, to select which of the remaining independent features will be used to train both models, feature selection was also carried out. For this, one of the important part of the entire ML process was performed wherein both datasets were split into training and testing datasets. The training dataset was used to train the models, and the testing dataset was used to evaluate how well both the models have performed is discussed in next Section 3.5.

3.5 Experimental Work with ML Techniques

Numerous experiments have been performed to put theory into practice. Each ML model with its default configurations are trained and assessed as baseline models at the initial stage. At this point, train-test split was applied to see how well the learned model will generalize to new data. Later, configurations are changed to make multiple tests and thereafter, each ML model is assessed with the set parameters and selected techniques to provide comparison in terms of their accuracy. Each ML approach is configured to reduce the model error on the validation dataset using a fixed set of hyper-parameters for each ML model. A cross validation score and GridSeachCV are applied with different cv values to estimate the performance of the ML algorithms. Different line graphs are created from given range of values to check test and train accuracies in the form of a series of data points on X and Y coordinates for the models. Furthermore, the results of each model is visualized in the form of a confusion matrix using Heatmap via matplotlib and seaborn. In addition, a classification report is also created to evaluate the effectiveness and performance of the predicted model using results from the confusion matrix.

The train-test split, a model validation procedure was used to estimate the performance of ML algorithms when they are used to make predictions on data. Model trained on the training set i.e. “x_train” and “y_train” and Model tested on the testing set i.e. “x_test” and “y_test” and evaluated the performance. Each time the code is executed, the identical train test split is reproduced using a pseudo-random number parameter in a random state. However, to test how effectively the model is generalised to updated data, "random state" is set to "0". The train test split and random state on different ML techniques applied in Dataset1 and Dataset2 are shown in tables 3.1 and 3.2 respectively.

In Dataset1 and Dataset2, while creating an ANN, “ann” abject was initialized layers are created. In network 1 input layer, 1 hidden layers and 1 output layer was created initially. For further tests in network, 1 input layer, 2 hidden layers and 1 output layer was created. All the layers are created by using Dense class which is part of layers’ module. Each class has accepted two inputs i.e. units and activation. In Dataset1, input layer has contained 32 neurons, hidden layers encompasses 12 neurons in each layer and for the second input activation function “relu (rectified linear unit)” was used for input and hidden layers. In Dataset2, input layer has included 16 neurons, 2 hidden layers encompasses 14 neurons in each layer and for the second input activation function again

Table 3.1: Finalized values of train-test split method and random state in Dataset1 after experiments

ML Techniques	Train Size	Test Size	Random State
ANN	0.7	0.3	2
LR	0.9	0.1	101
NB	0.75	0.25	0
RF	0.7	0.3	1
K-NN	0.8	0.2	0

Table 3.2: Finalized values of train-test split method and random state in Dataset2 after experiments

ML Techniques	Train Size	Test Size	Random State
ANN	0.8	0.2	42
LR	0.7	0.3	42
NB	0.75	0.25	0
RF	0.7	0.3	2
K-NN	0.8	0.2	0

"relu" was used for input and hidden layers. In order to avoid overfitting, regularization using dropout and learning rate are used to enhance Deep learning model's accuracy. An output layer was employed to provide specified output wherein Dense class is again used. As it's a binary classification, only one neuron was allocated for output result and activation function "Sigmoid" was used in both the datasets. A method "compile" was used to compile the network which accepted three inputs i.e. optimizer, loss, and metrics. While making experiments "RMSProp" and "adam" optimizers are used to check which optimizer provide the optimum output. Loss specified the loss function for which "binary crossentropy" was used as binary classification and to compute the performance, "accuracy" is used as a performance metric.

After compilation, “fit” method is used to train the ann. The “fit” method accepted 4 inputs here. “x_train” as feature matrix for the training dataset and “y_train” as dependent variable vector for the training dataset. A batch size was used to see how many observations would be there in the batch. For this, different values are tested for batch size with default and other values. Epochs are used to check how many times NN will be trained for which 20, 50, 90,100, and 150 epoch values are tested in both the datasets. By carrying out the cross-validation process repeatedly provided average result against all folds and runs has shown that the average predicted result provided more accurate assessment of the model underlying an average performance on the dataset.

In Dataset1 and Dataset2, while apply NB classifier, we have applied a technique namely GaussianNB. After splitting, we trained the GaussianNB on the training set. In both datasets for GaussianNB, cross validation score was applied with different cv levels wherein for each iteration, the data is repeatedly separated into a testing and training set. The testing set is used to compute the scores once the estimator has been trained on the training data. The cross validation score's input consists of an estimator (using the fit and predict method), the cross-validation object, and the input dataset.

In order to implement LR in both Dataset1 and Dataset2, a pipeline method from sklearn pipeline, an instance of pipeline is constructed after the train-test split. The GridSearchCV class in Python Sklearn is used to implement the grid search. In this case the fit, predict, and score method used in the class provided the best model constructed with the most ultimate hyperparameters. GridSearchCV received the instance of the pipeline through the estimator. To pass the parameter grid to GridSearchCV using param grid, a JSON array with the grid's parameters erected and all the values in it are tested.

The accuracy score in LR is determined by using the scoring parameter, which is set to "accuracy". On the GridSearchCV instance containing training data and a corresponding label, the method fit is called. The following characteristics are utilized to obtain critical information when the GridSearchCV estimator is fitted where “best score” provided the score for the best model that is built using the most effective configuration of the hyperparameters, “best params” gave the most ideal hyperparameters that is utilized to obtain the best model, and “best estimator” provided the best model constructed with

the most optimal hyperparameters.

In order to implement RF in Dataset1 and Dataset2, different hyperparameters like "max depth", "min samples leaf", and "n estimators" are applied and tested with different values to enhance the predictive capability of the models. Moreover, "n jobs", "random state", and scoring are applied and checked to make the models faster. Here, "n jobs" with 1 and -1 is tested to permit single or multiple processors and "random state" controlled the randomness of the sample. In both the datasets, hyperparameter tuning for RF is performed using GridSearchCV and fit the models. Additionally, features importance was also checked to see which risk feature reflects the most in order to make optimum predictions are shown in Chapter 4. A graph is created showing the best accuracies and error rates. A forest is created based on the best estimator value with a given "max depth" and "min sample leaf".

Using the training set, we created and trained the K-NN model. Three parameters are used in the model creation. Number of neighbors "n neighbors" was set with different values, means which neighborhood points are required for classifying a given point, metric as "minkowski" and "p" indicates the Manhattan Distance wherein "p" with 1, Euclidean Distance wherein "p" as 2, and Cheybchev Distance where "p" is dedicated with infinity value. However, we have tested p with 1 and 2 values in both the datasets. After model creation, we predicted the output for the test set. The actual and predicted values are compared and our model is evaluated using confusion matric and accuracy score by comparing the actual and predicted test values.

3.6 Summary

By incorporating ML techniques, helps to analyze massive amount of data and derive risk predictive insights. ML is now playing an important role towards the emergence and progressively development of the applications [97]. In above-mentioned methodology perspective, multidimensional analysis of data is performed by using effective BI techniques like predictive modeling, analytics, and model visualization to monitor project's progress successfully. Therefore, integration of PM with ML is selected to enhance decision-making process utilizing BI.

All five ML techniques were tested, which provided their respective test and train accuracies. Before training the models, data preprocessing i.e. filled empty/null values in Dataset1, renaming of attributes and conversion of multivalued to binary conversion in Dataset2 was performed to enable feature engineering in both the datasets i.e one hot encoding, features selection, and standardization. Results and discussions made for each ML techniques on Dataset1 and Dataset2 are found in Chapter 4.

The Experimental setup of ML techniques applied in Dataset1 and Datset2 are found in Section 3.5 and final results are found in Chapter 4, Figure 4.45 and Figure 4.46. Further results are also demonstrated in Chapter 4 from Table 4.1 to 4.6 respectively.

Chapter 4

Results and Discussion

The results of using various ML techniques are examined and contrasted in this chapter. The main objectives of the work completed for this dissertation were to examine the state-of-the-art in research on risk assessment in software projects and its impact on project timeline management, particularly with the application of ML techniques to improve risk management processes, and the development of a software module capable of predicting the effects of various risk factors in a new software project before implementation phase gets initiated.

To analyze the results of our performed experiments, the evaluation techniques and performance measures are set up. We have used accuracy as our performance measure to compare the results of different ML techniques.

4.1 Results and Discussions

Five ML techniques evaluated on both the Dataset1 and Dataset2 are given below.

4.1.1 ANN

When generating an ANN in Datasets 1 and 2, an “ann” abject was created using a definite class of Keras called Sequential. After compilation using “adam” optimizer and binary_crossentropy, the ann is trained using the "fit" approach. The batch size of 40 in Dataset1 and 32 in Dataset2 was found to be the most practical out of all the numbers examined. Finally, 145 epochs for Dataset1 and 100 epochs for Dataset2 were employed. The accuracy of both datasets improved and the loss is decreased in each

epoch. Moreover, k-fold cross validation is used with a cv value of "10" to gauge how well an ANN algorithm is performing.

In Dataset1, ANN presented the test accuracy of 99.34% and 99.56% of train accuracy. The dropout value 0.2 is used for input and hidden layers to avoid over-fitting and improve the model. In addition, k-fold cross validation attained the test accuracy 99.2% and train accuracy 99.4% respectively. A confusion matrix of predictions made by ANN tested for this model is shown in Figure 4.1.

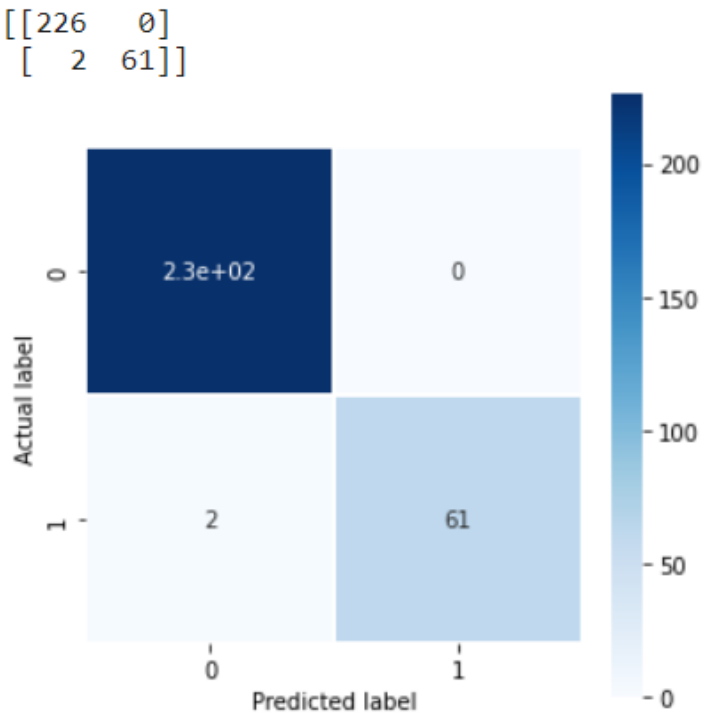

Figure 4.1: ANN Confusion Matrix of Dataset1

The confusion matrix has shown 287 correct predictions and 2 incorrect predictions. In classification report, 0.0 indicated "low risk" and 1.0 indicated "high risk" class. ANN's report is shown in Figure 4.2.

Precision indicated that the predicted model has owned a large impact of correctly predicted positive risk outcome i.e. 100%. Recall has shown 97% of the risk factors, the model correctly predicted the outcome. The f1-score i.e. harmonic mean of precision and recall is 98% which is the optimum value. However, the support value has shown an occurrence how risk belonged to each class in the test dataset i.e. 226 with low risk and 63 having a high risk effect.

	precision	recall	f1-score	support
0.0	0.99	1.00	1.00	226
1.0	1.00	0.97	0.98	63
accuracy			0.99	289
macro avg	1.00	0.98	0.99	289
weighted avg	0.99	0.99	0.99	289

Figure 4.2: ANN Classification Report of Dataset1

The model accuracy depicted the ANN's test and train accuracy. Approximately on 85 epochs, 77.5% of test and train gave same output and after that both of accuracies continued to increase above 99% is shown in Figure 4.3.

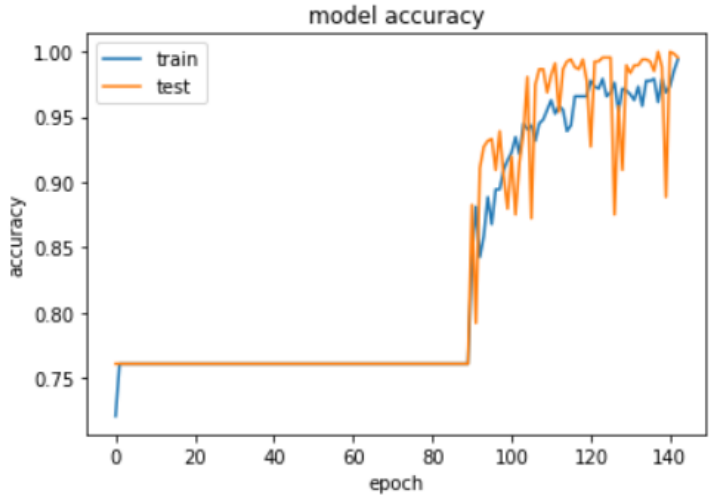

Figure 4.3: ANN model's epoch vs accuracy plot on the train and test Dataset1

The model loss against train and test is shown in graph, indicated that loss is reducing at comparable values and reached to 0.1 on around 140 epochs, which is shown in Figure 4.4.

The aggregated accuracy and loss validation illustrated that accuracy and validation accuracies both were increased one after the other with comparable values. Similarly, loss and validation loss reduced with analogous values and as the value of accuracy and validation increased, loss and validation loss got decreased. This is shown in Figure 4.5.

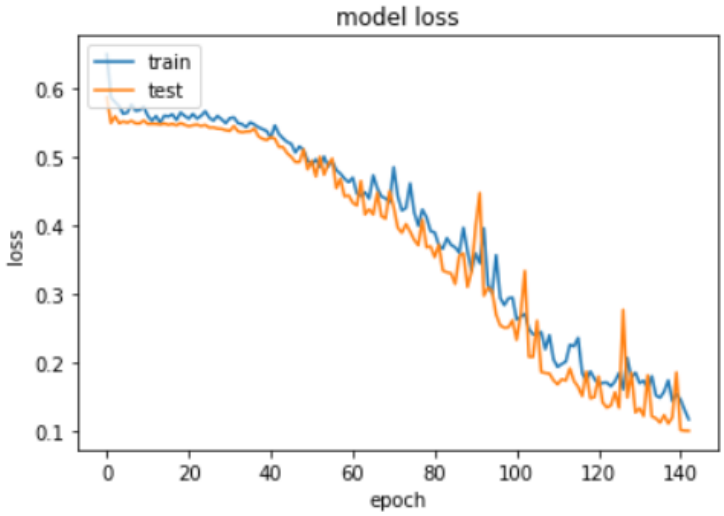

Figure 4.4: ANN model's epoch vs loss plot on the train and test Dataset1

Figure 4.5: ANN model's accuracy and loss plot on the train and validation in Dataset1

In Dataset 2, ANN presented testing accuracy of 99.2% and training accuracy of 99.58%. Regularization is used to avoid over-fitting in which The dropout value 0.2 is used in input and hidden layers and LR value 1e-4 with compile function to improve ANN. Furthermore, k-fold cross validation is used with cv value "10" and presented testing accuracy of 98.9% and 99.3% training accuracy respectively. The summary of prediction outcomes is displayed in confusion matrix is illustrated in the Figure 4.6. It has shown 59 correct predictions and 1 incorrect predictions.

In classification report, precision indicated that the predicted model has a large impact of correctly predicted positive risk outcome i.e. 100%. Recall has shown 97% of the risk

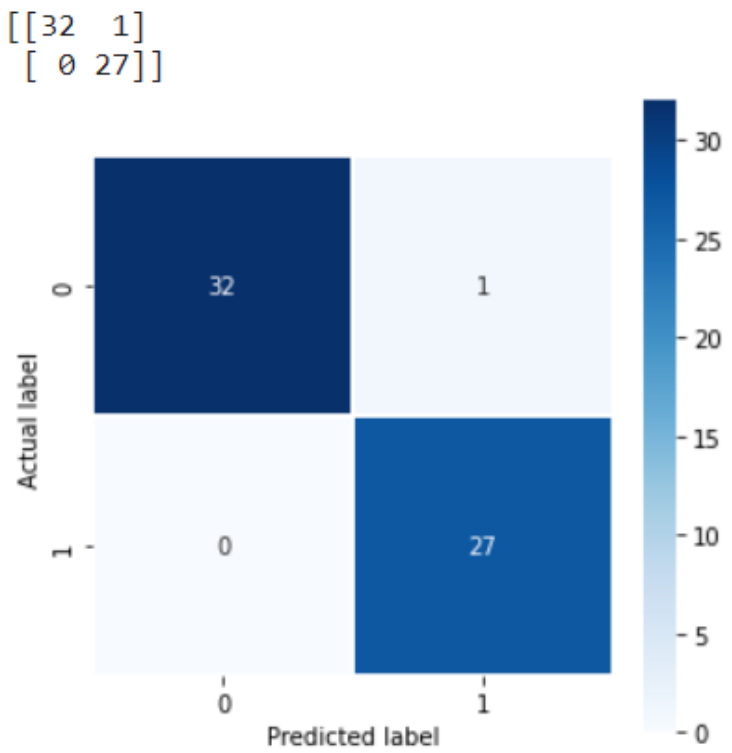

Figure 4.6: ANN's model confusion matrix of Dataset2

factors, the model correctly predicted the outcome. The f1-score is 98% which is the optimum value. However, the support value has shown an occurrence how risk belonged to each class in the test dataset i.e. 33 with low risk and 27 having a high risk effect. It is shown in Figure 4.7.

	precision	recall	f1-score	support
0.0	0.99	1.00	1.00	33
1.0	1.00	0.97	0.98	27
accuracy			0.99	60
macro avg	1.00	0.98	0.99	60
weighted avg	0.99	0.99	0.99	60

Figure 4.7: ANN's model classification report of Dataset2

As the number of epochs increased, both test and train accuracy increased. After 20 epochs, both accuracies continued to increase at comparable level. The model accuracy is shown in Figure 4.8 indicated the test and train accuracy.

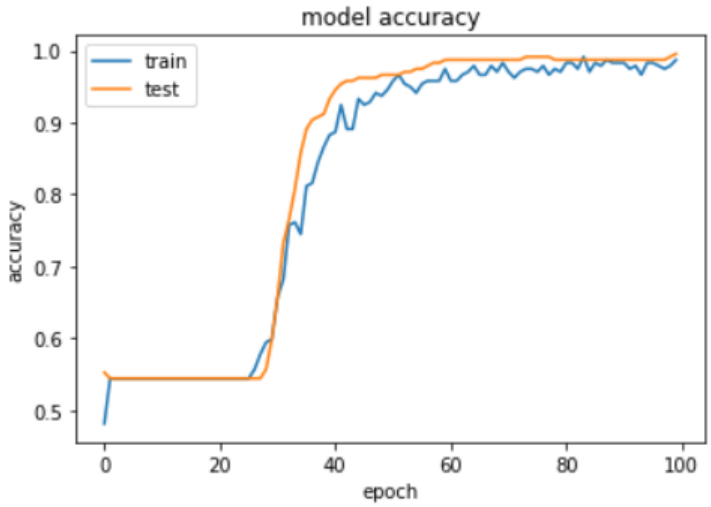

Figure 4.8: ANN model's epoch vs accuracy plot on the train and test Dataset2

The model loss against train and test is plotted, indicated that loss is reduced from 0.7 to 0.1 as the number of epochs increased till the model's execution ended is shown in Figure 4.9.

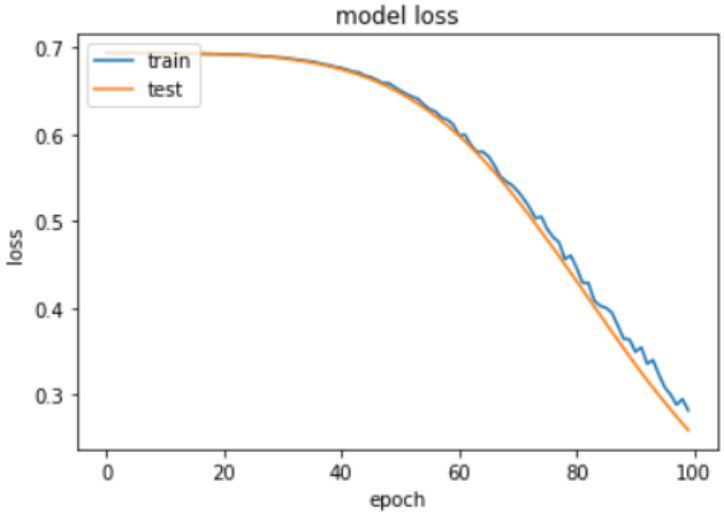

Figure 4.9: ANN model's epoch vs loss plot on the train and test Dataset2

The aggregated accuracy and loss validation of the model is shown in Figure 4.10. It illustrated that accuracy and validation accuracies both were increased with comparable values and after 20 epochs both of the accuracies continued to increase. Similarly, loss and validation loss reduced with similar values and reached to 0.1 and as the value of accuracy and validation accuracy increased, loss and validation loss decreased.

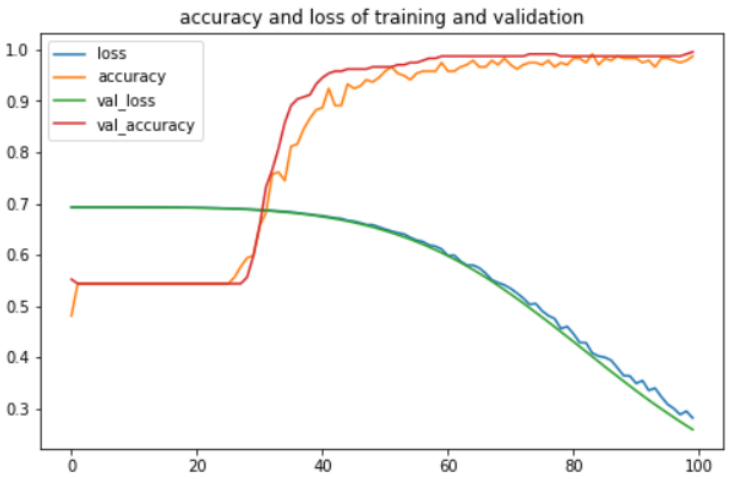

Figure 4.10: ANN model's accuracy and loss plot on the train and validation in Dataset2

4.1.2 LR

In Dataset1, numerous values against parameter grid are tested and found "best params" values of 0.5 and 10 which presented the optimal results "best score" with "best estimator". The random state of 101 is used with penalty "L2" and solver "lbfgs". By using the fit method LR gave the test accuracy of 98.96% and train accuracy of 99.55%. The GridSearchCV was applied with cv value "5" and attained the test accuracy of 96.89% and train accuracy of 99.4%. The line graph presented the accuracy and loss with varying number of "param grid" as shown in Figure 4.11 below.

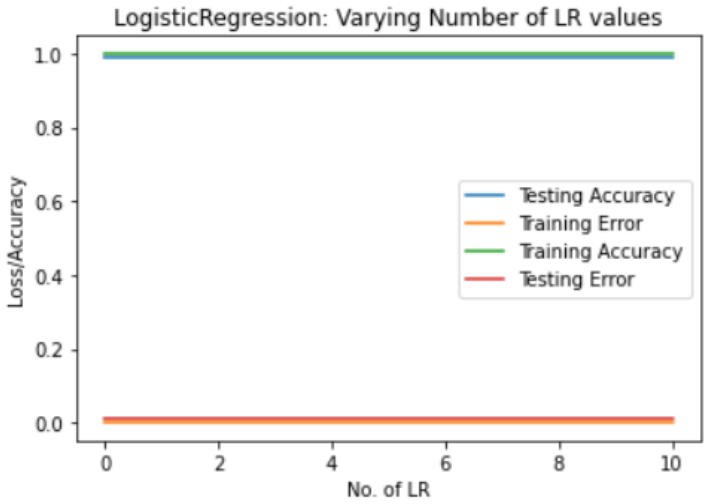

Figure 4.11: LR model's loss and accuracy plot on the train and test Dataset1

LR confusion matrix using heatmap is illustrated in the Figure 4.12 below which has

shown 96 correct predictions and 1 incorrect prediction.

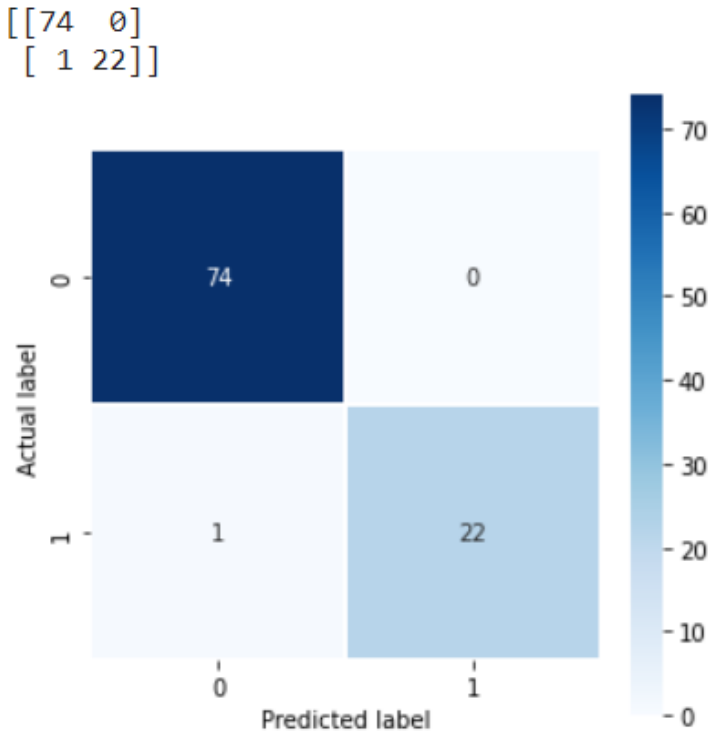

Figure 4.12: LR's model confusion matrix of Dataset1

In classification report, precision has shown that the predicted model has a significant influence on accurately predicting a favorable risk outcome. Recall revealed that the model properly predicted 96% of the risk factors. The f1-score is 98% which is relatively high. The support value, however, demonstrated an instance of how risk belonged to each class in the test dataset, with 74 having low risk and 23 having a high risk effect.It is shown in Figure 4.13.

	precision	recall	f1-score	support
0	0.99	1.00	0.99	74
1	1.00	0.96	0.98	23
accuracy			0.99	97
macro avg	0.99	0.98	0.99	97
weighted avg	0.99	0.99	0.99	97

Figure 4.13: LR's model classification report of Dataset1

Receiver Operating Characteristic (ROC) represented a probability curve for many classes. It indicated in terms of the predicted probability, how well the model can differentiate between the specified risk classes. Below Figure 4.14 has depicted a perfect ROC curve indicated the LR performance with True Positive and False Positive rate exceeding from 0 to 1.

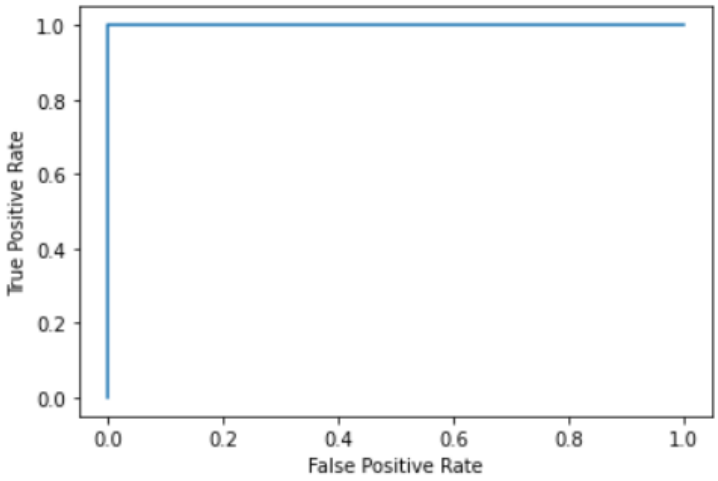

Figure 4.14: LR's model ROC plot on Dataset1

In Dataset2, a range of values are evaluated against the parameter grid and the "best params" value was found to be 0.5, which provided the test accuracy of 98.8% . A random state 42, solvers "lbfgs" and "newton-cg", and penalty "L2" is used. Using the fit approach, LR produced test and train accuracy values of 98.8% and 100%, respectively. The GridSearchCV was applied with cv level 5 and attained the test accuracy and train accuracy of 98.5% and 99.8% respectively. As shown in the Figure , the line graph displayed the accuracy and loss with changing numbers of "param grid". Figure 4.15 below.

LR confusion matrix using heatmap is illustrated in the Figure 4.16 has shown 89 correct predictions and 1 incorrect prediction.

In classification report, precision has shown that the predicted model had a significant influence on accurately predicting a favorable risk outcome i.e. 100%. Recall revealed that the model properly predicted the result for 98% of the risk factors. The f1-score, which is 99%, is more or less at 100%. The support value demonstrated an instance of how risk belonged to each class in the test dataset, with 49 having low risk and 41 having a high risk effect is shown in 4.17.

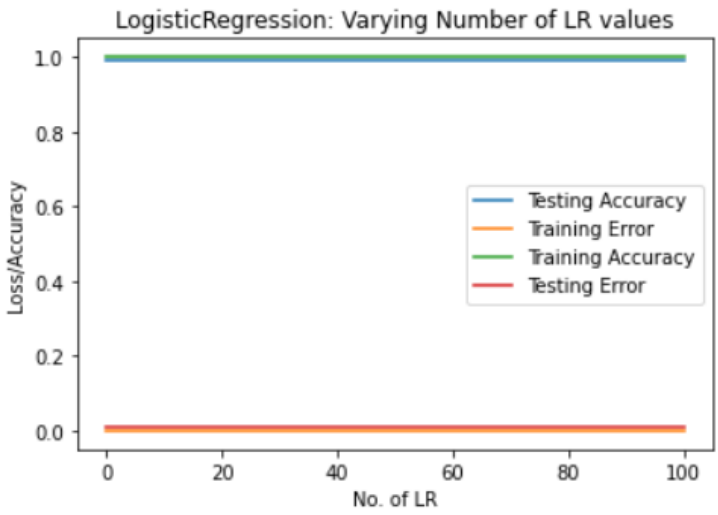

Figure 4.15: LR model's loss and accuracy plot on the train and test Dataset2

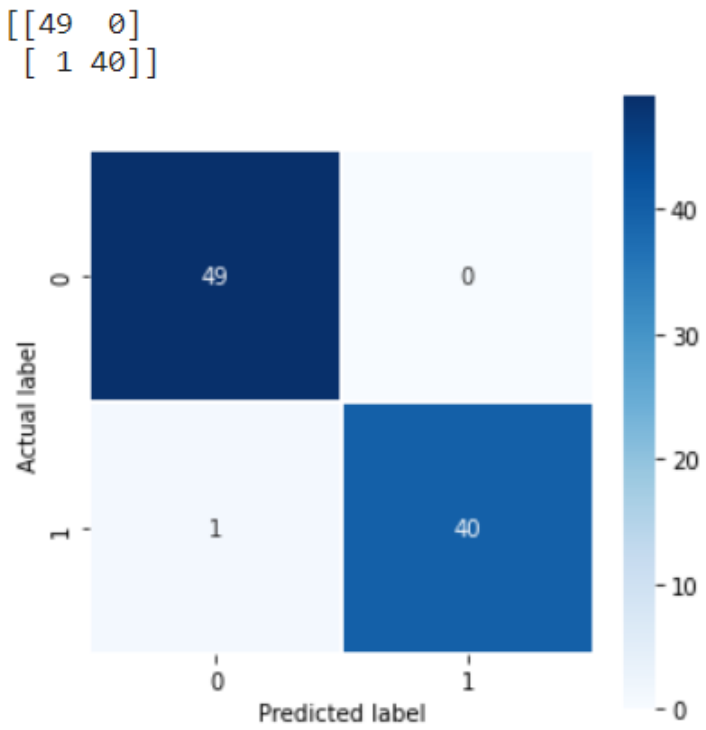

Figure 4.16: LR's model confusion matrix of Dataset2

	precision	recall	f1-score	support
0	0.98	1.00	0.99	49
1	1.00	0.98	0.99	41
accuracy			0.99	90
macro avg	0.99	0.99	0.99	90
weighted avg	0.99	0.99	0.99	90

Figure 4.17: LR's model classification report of Dataset2

The ideal ROC curve in the graph depicted the LR performance with a True Positive and False Positive predicted probability rate greater than 0 to 1. It is shown in Figure 4.18

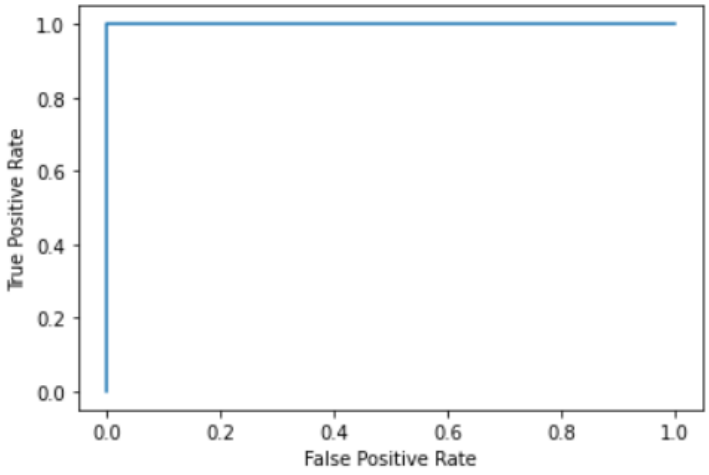

Figure 4.18: LR's model ROC plot on Dataset2

4.1.3 NB

In Dataset1, NB classifier including GaussianNB presented the testing accuracy of 99.1% and 99.3% of train accuracy. Cross validation was applied with cv value "6" and attained 96.2% and 98.8% value of the test and training accuracy respectively.

A confusion matrix made for NB classifier for the model is shown using heatmap is illustrated in the Figure 4.19 has shown 239 correct predictions and 2 incorrect model predictions.

In NB's classification report, precision revealed that the predicted model has a large impact on correctly anticipating a positive risk outcome which is 95%. Recall showed

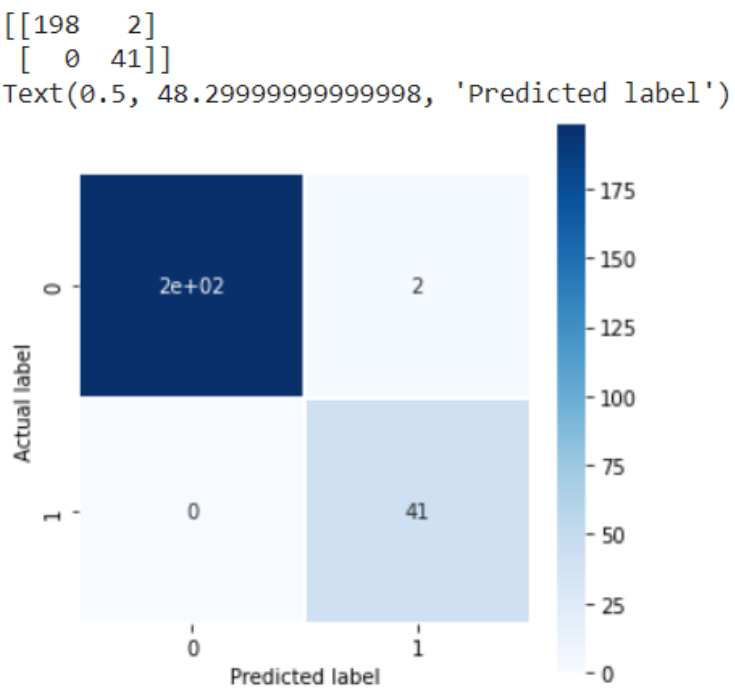

Figure 4.19: NB's model confusion matrix of Dataset1

that for all of the risk factors, the model accurately predicted the outcome. In general, the f1-score, which is 98% is nearer to 100%. With 200 having low risk and 41 having a high risk effect, the support value illustrated how risk belonged to each class in the test dataset is shown in Figure 4.20.

	precision	recall	f1-score	support
0	1.00	0.99	0.99	200
1	0.95	1.00	0.98	41
accuracy			0.99	241
macro avg	0.98	0.99	0.99	241
weighted avg	0.99	0.99	0.99	241

Figure 4.20: NB's model classification report of Dataset1

In Dataset2, NB classifier including GaussianNB presented the testing accuracy of 98.6% and 99.56% of train accuracy. Cross validation was applied with cv level "6" and gained 95.9% and 99.1% value of the testing and training accuracy respectively.

A confusion matrix of predictions made by NB for the model is shown in Figure 4.21, indicating 74 correct predictions and 1 incorrect prediction.

```
[[44  1]
 [ 0 30]]
Text(0.5, 48.29999999999998, 'Predicted label')
```

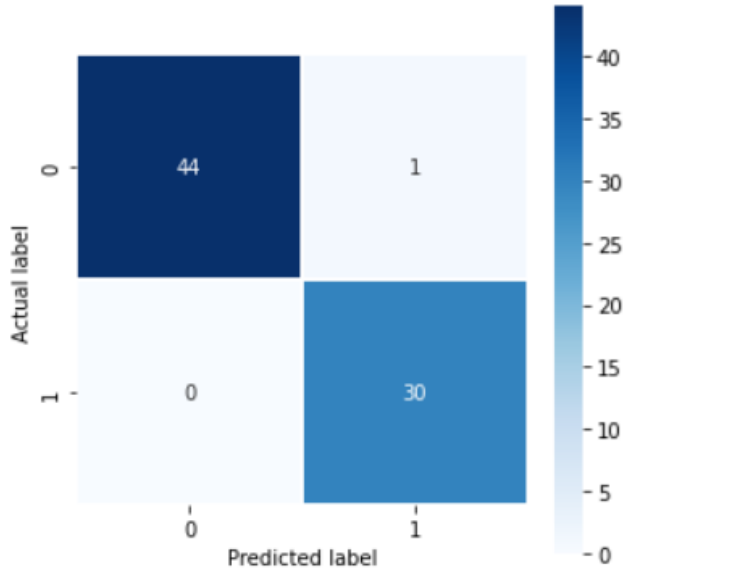

Figure 4.21: NB's model confusion matrix of Dataset2

The classification report has shown the precision, the predicted model contributed significantly to correctly predicting a positive risk outcome, which is 97%. Recall demonstrated that the model completely predicted the outcome 100% for each risk factor. The f1-score 98% is nearly equal to 100%. In the test dataset, risk belonged to each class, with 45 having low risk and 30 having a high risk effect by the support value is shown in Figure 4.22.

	precision	recall	f1-score	support
0	1.00	0.98	0.99	45
1	0.97	1.00	0.98	30
accuracy			0.99	75
macro avg	0.98	0.99	0.99	75
weighted avg	0.99	0.99	0.99	75

Figure 4.22: NB's model classification report of Dataset2

4.1.4 RF

In Dataset1, classifier RandomForestClassifier was applied using "gini" criterion, 4 "n estimators", random state 1, and "n jobs" -1 which provided the test accuracy of 97.6% and train accuracy of 99.5%. GridSearchCV level of 10 was applied and attained the test accuracy of 94% and training accuracy of 96.73%.

Previously, features importance was checked to see which feature reflects the most in order to make optimum predictions. Numerous tests were conducted by giving different values of estimators. Moreover, different values against "max depth" and "min sample leaf" are tested. Below Figure 4.23 has shown that estimators 4 and 5 gave the best testing and training accuracy values with 4 "max depth" and 5 "min samples leaf" are used for minimum number of samples required to be at a leaf node. On estimator 2 and 3, when testing and training accuracy is reduced the testing and training error increased. On the other side, estimator 4 and 5 has presented an increased in testing and training accuracy while reduction in testing and training error.

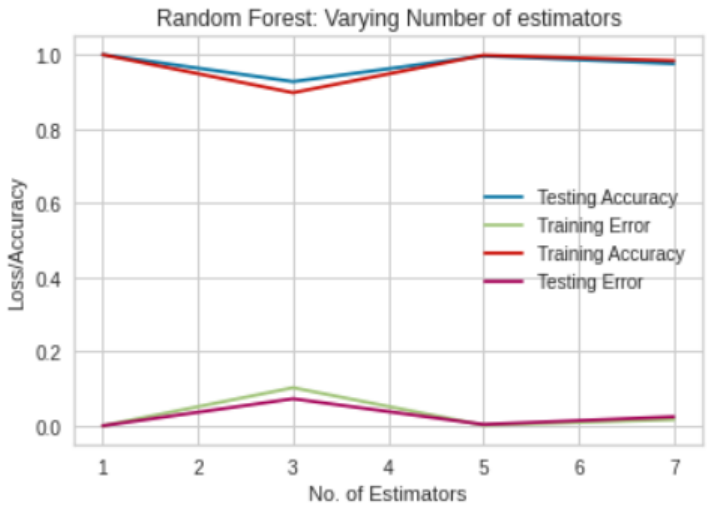

Figure 4.23: RF model's loss and accuracy plot against no. of estimators on the train and test Dataset1

The forest created for optimal results with "estimator" 4, "max depth" 4, and "min sample leaf" 5 is shown in Figure 4.24 below.

RF confusion matrix using heatmap is illustrated in the Figure 4.25 has shown 282 correct predictions 7 incorrect model's predictions.

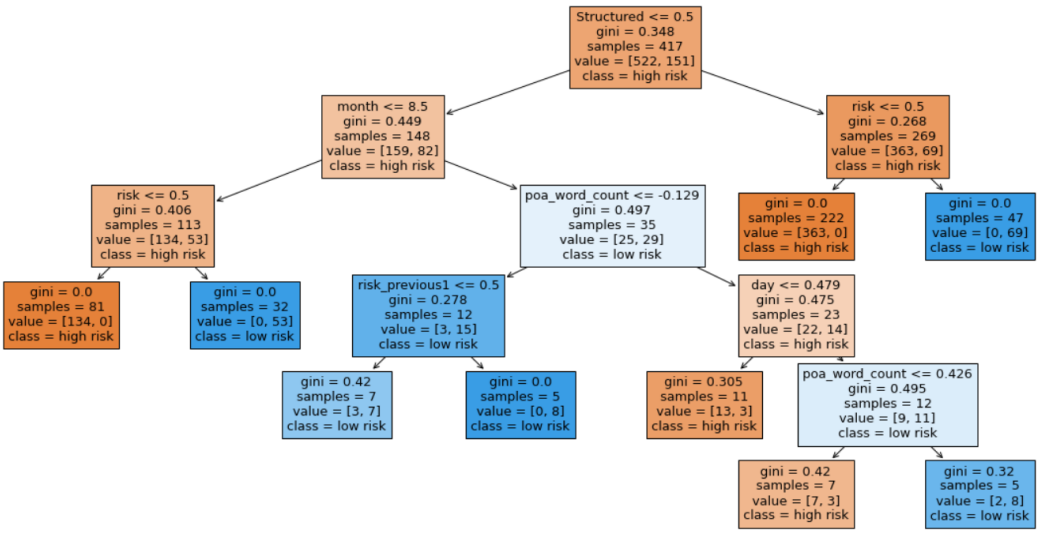

Figure 4.24: RF model's forest with optimal results "estimator" 4, "max depth" 4, and "min sample leaf" 5 in Dataset1

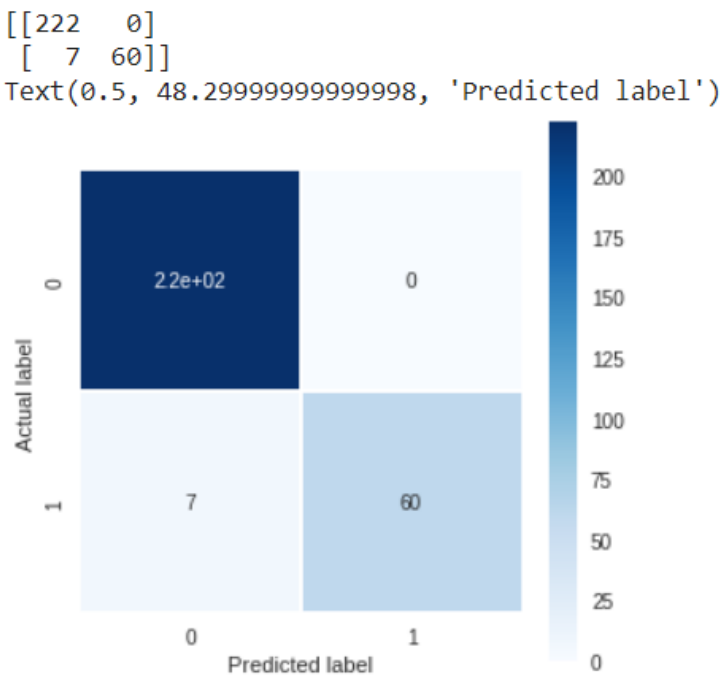

Figure 4.25: RF model's confusion matrix of Dataset1

In classification report, precision has shown that the predicted model had a significant influence to accurately predicting a favorable risk outcome i.e. 100%. Recall revealed that the model properly predicted the result for 90% of the risk factors. The f1-score is 94%. The support value demonstrated an instance of how risk belonged to each class in the test dataset, with 222 having low risk and 67 having high risk effect. The report is

shown in Figure 4.26.

	precision	recall	f1-score	support
0	0.97	1.00	0.98	222
1	1.00	0.90	0.94	67
accuracy			0.98	289
macro avg	0.98	0.95	0.96	289
weighted avg	0.98	0.98	0.98	289

Figure 4.26: RF model's classification report of Dataset1

Features importance is demonstrated in Figure 4.27 based on the results extracted from above scenario. Class "risk" has shown drastic consequences in the project risk report, next "report count" presented the number of reports available for the projects, and "project risk likelihood" indicated the risk occurrence probability.

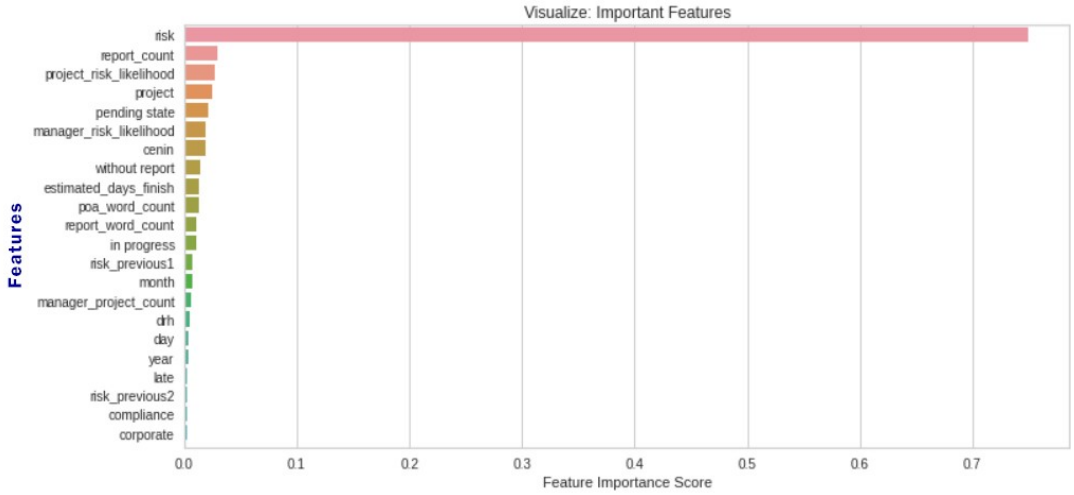

Figure 4.27: RF model's features importance attained after making experiments in Dataset1

In Dataset2, classifier RandomForestClassifier was applied using "gini" criterion, "estimators" 2, random state 2, and "n jobs" -1 which provided the test accuracy of 92.2% and train accuracy of 95.7%. GridSearchCV level was applied with cv level "10" and attained the test accuracy of 92% and training accuracy of 93.7%.

Earlier, features importance was checked for which various tests were conducted by giving different values of estimators. Moreover, different values against "max depth" and "min sample leaf" are also tested. Among them, estimator 2 gave the best testing and training accuracy values with "max depth" 4 and "min samples leaf "5 is shown in Figure 4.28, used for minimum number of samples required for a leaf node. On

estimator 2, when testing and training accuracy increased the testing and training error got decreased.

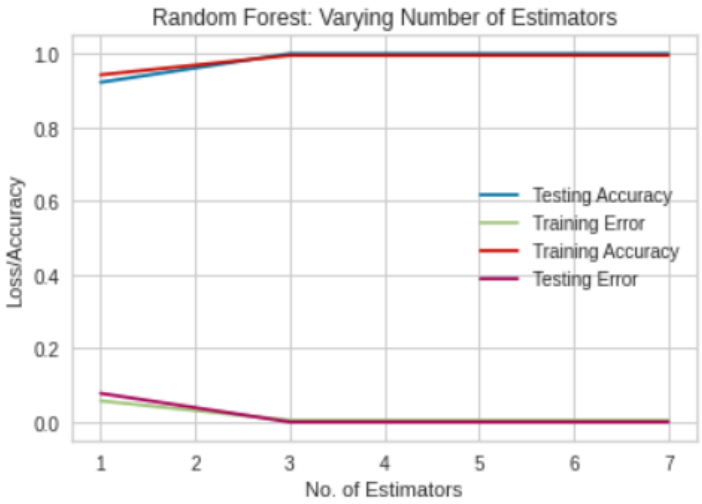

Figure 4.28: RF model's loss and accuracy plot against no. of estimators on the train and test Dataset2

The forest created for "estimator" 2, "max depth"4, and "min sample leaf" 5 is shown in Figure 4.29.

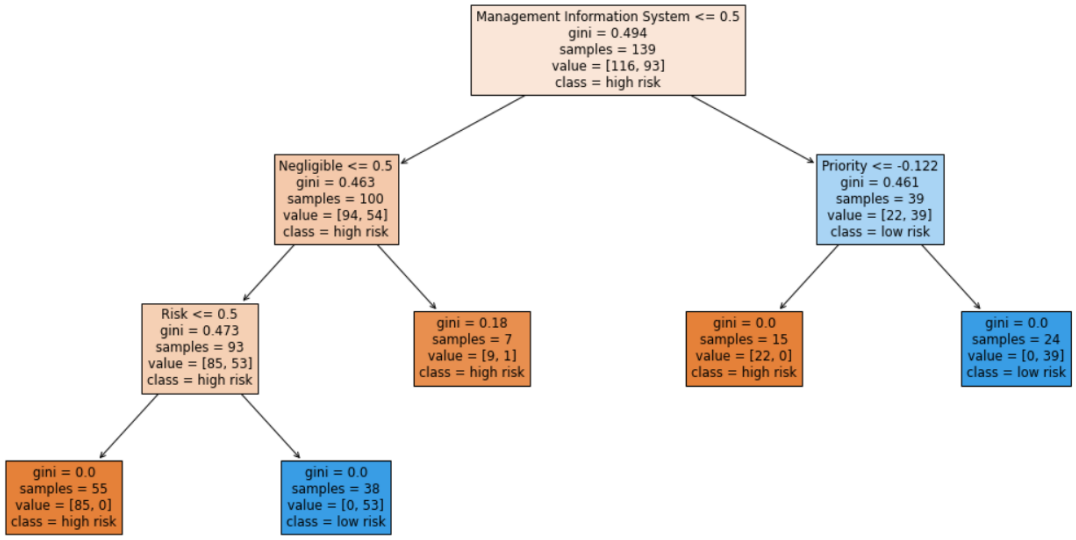

Figure 4.29: RF model's forest with optimal results "estimator" 2, "max depth" 4, and "min sample leaf" 5 in Dataset2

RF confusion matrix using heatmap is illustrated in the Figure 4.30 has shown 83 correct predictions and 7 incorrect predictions.

```
[[49  0]
 [ 7 34]]
Text(0.5, 48.29999999999998, 'Predicted label')
```

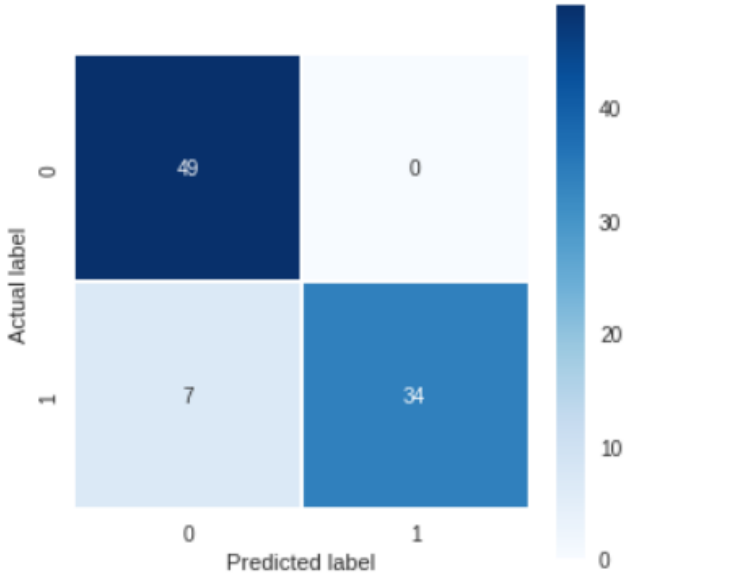

Figure 4.30: RF model's confusion matrix of Dataset2

In RF's classification report, precision presented the predicted model had a significant influence on accurately predicting a favorable risk outcome i.e. 100%. Recall revealed that the model properly predicted 83% of the risk factors. The f1-score is 91%. The support value demonstrated an instance of how risk belonged to each class in the test dataset, with 49 having low risk and 41 having high risk effect is shown in Figure 4.31.

	precision	recall	f1-score	support
0	0.88	1.00	0.93	49
1	1.00	0.83	0.91	41
accuracy			0.92	90
macro avg	0.94	0.91	0.92	90
weighted avg	0.93	0.92	0.92	90

Figure 4.31: RF model's classification report of Dataset2

Features importance is demonstrated in Figure 4.32 based on the results extracted from above criterion. Class "Risk" has shown a radical change, "Probability" has shown probability of risk occurrence, and "Medium" indicated the risk magnitude level.

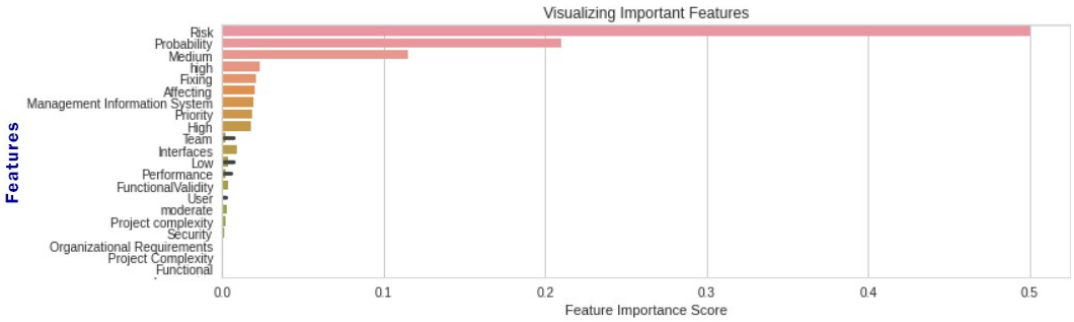

Figure 4.32: RF model's features importance attained after making experiments in Dataset2

4.1.5 K-NN

In Dataset1, from a range of 1-9 multiple values for "n neighbors (k)" are tested with values of "p" as 1 and 2. The train and test accuracy graph was plotted to check which value of "k" gave the best accuracy with "p" as 1.

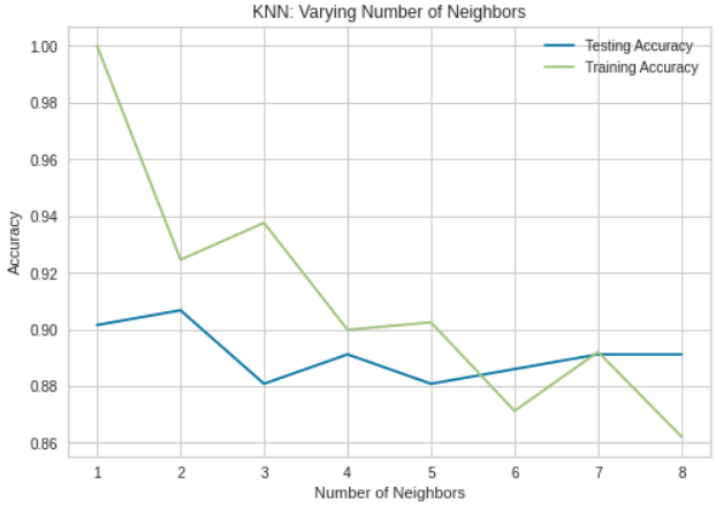

Figure 4.33: K-NN model's no. of neighbors vs accuracy plot on the train and test Dataset1 when value of k is 2 and p is 1 (Manhattan Distance)

The graph indicated that "k" with 2 and "p" with 1 gave the better test and train accuracy of 90.6% and 92.4% respectively. Here, main point was emphasized where both the test and train accuracy values got increased. At "k" with 3, train accuracy improved and test accuracy reduced so this point is ignored. In order to make more test to find the best accuracy, loss and accuracy graph was plotted with same parameters. This is shown in Figure 4.33.

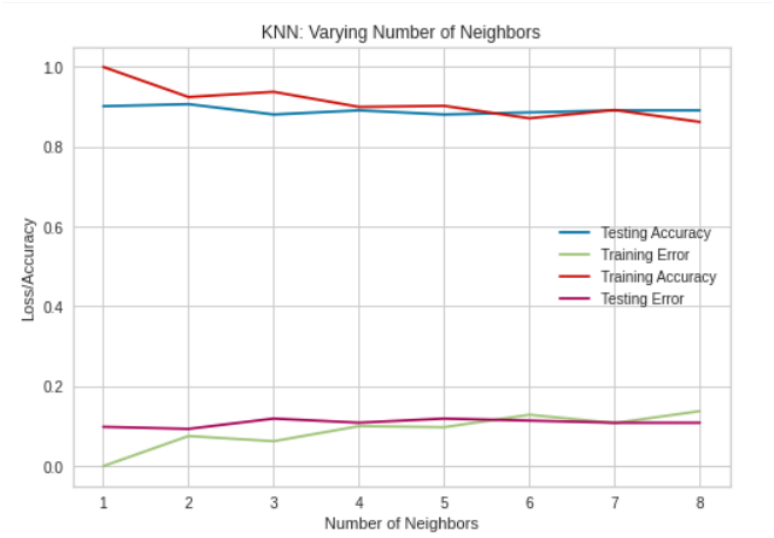

Figure 4.34: K-NN model's loss / accuracy vs no. of neighbors plot on the train and test Dataset1 when value of k is 2 and p is 1 (Manhattan Distance), attained optimal results

The graph indicated when testing accuracy is increased, testing error is reduced and when training accuracy is increased, training error decreased on "k" with 2 and "p" as 1 value. This is shown in Figure 4.34. Moreover, when testing accuracy is decreased, testing error is increase and when training accuracy is increased, training error decreased on "k" with 3 and "p" as 1 value.

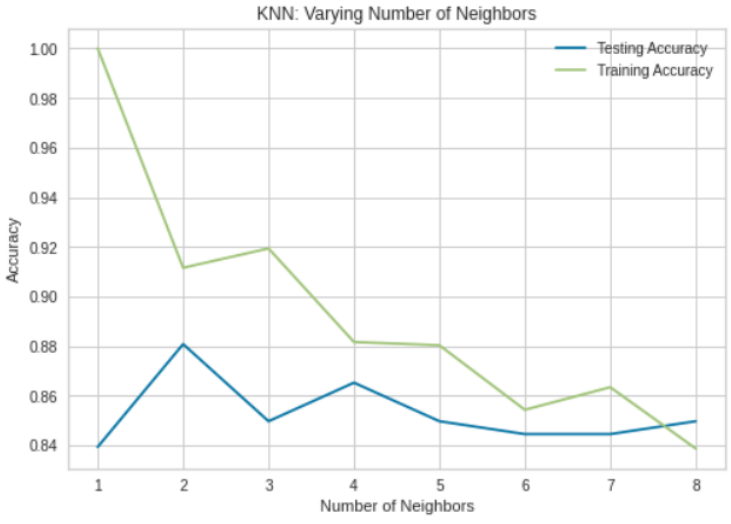

Figure 4.35: K-NN model's no. of neighbors vs accuracy plot on the train and test Dataset1 when value of k is 2 and p is 2 (Euclidean Distance)

The graph illustrated that at “k” with 2 and “p” with value 2 gave the test and train accuracy of 88% and 90% respectively. Again, that point was emphasized where both the test and train values got increased. At “k” with value 3, train accuracy improved and test accuracy reduced so this point is ignored. In order to make more test to find the best accuracy, loss and accuracy graph was plotted with same parameters is shown in Figure 4.35

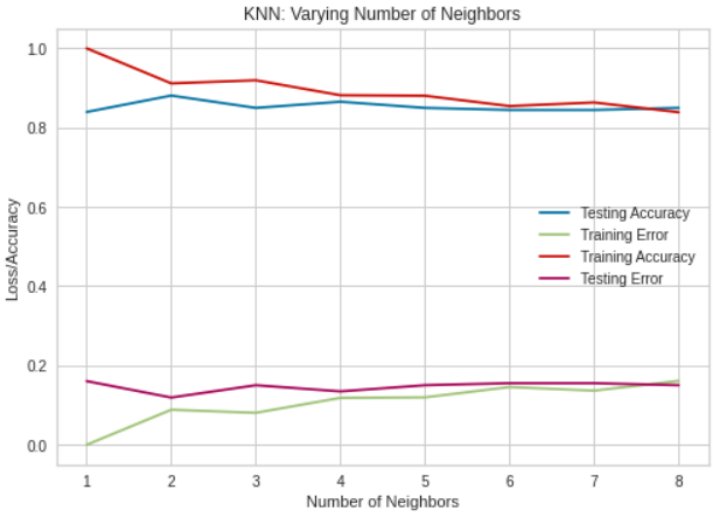

Figure 4.36: K-NN model’s loss / accuracy vs no. of neighbors plot on the train and test Dataset1 when value of k is 2 and p is 2 (Euclidean Distance)

The graph illustrated in Figure 4.36 has shown that when testing accuracy is increased, testing error is reduced and when training accuracy is increased, training error decreased on “k” with 2 and “p” with 2 value. Moreover, when testing accuracy is decreased, testing error is increase and when training accuracy is increased, training error decreased on “k” as 3 and “p” as 2.

From above all, best testing accuracy was 90.6% and training accuracy on which the model was trained has attained 92.4% accuracy on “k” as 2 and “p” as 1 is shown in Figure 4.33 and Figure 4.34. Moreover, cross-validation was applied with cv level “10” and attained testing accuracy of 83.9% on training accuracy of 84.9% respectively.

K-NN confusion matrix using heatmap is illustrated in the Figure 4.37 has shown 175 correct predictions and 18 incorrect predictions.

In K-NN’s classification report, precision has shown the accuracy of predicted model for which the risk outcome is predicted as 88%. Recall has shown that for 48% of the

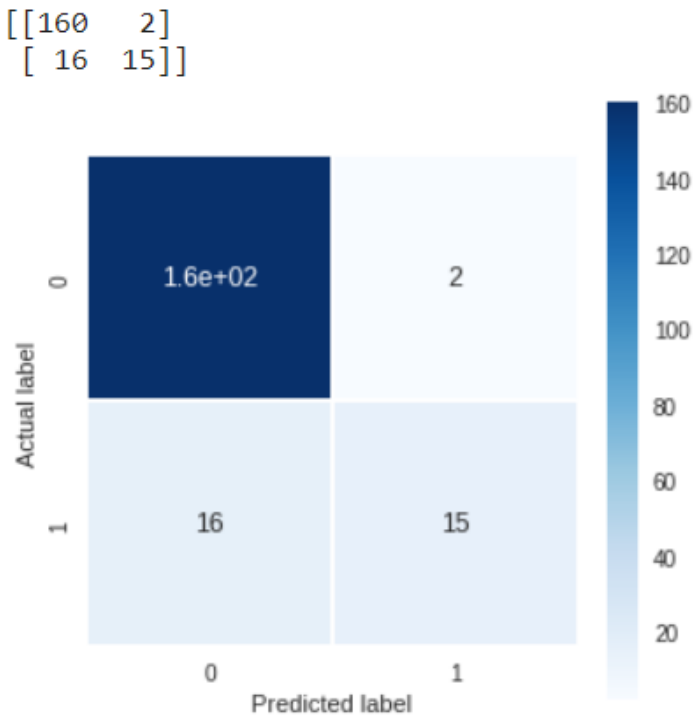

Figure 4.37: K-NN model's confusion matrix of Dataset1

risk factors, the model accurately predicted the outcome. 62% is the f1-score. With 162 having low risk and 31 having high risk effect, the support value illustrated how risk belonged to each class in the test dataset is shown in Figure 4.38

	precision	recall	f1-score	support
0	0.91	0.99	0.95	162
1	0.88	0.48	0.62	31
accuracy			0.91	193
macro avg	0.90	0.74	0.79	193
weighted avg	0.90	0.91	0.90	193

Figure 4.38: K-NN model's classification report of Dataset1

Similar to Dataset1, in Dataset2 multiple values for "n neighbors (k)" are examined from the range of 1 to 9, with values of "p" as 1 and 2. To determine which value of "k" delivers the best accuracy when "p" is 1, the train and test accuracy graph was plotted.

The graph illustrated that the test and train accuracy were better at 96.67% and 98.7%, respectively, when "k" with value 7 and "p" as 1 as shown in Figure 4.39. Here, the place

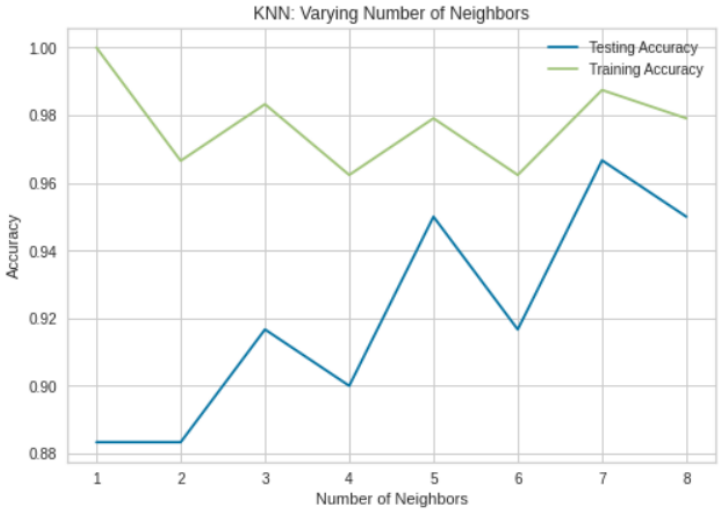

Figure 4.39: K-NN model's no. of neighbors vs accuracy plot on the train and test Dataset2 when value of k is 7 and p is 1 (Manhattan Distance)

where the test and train accuracy values improved was underlined. This argument is disregarded since at "k" with 3 and "k" with value 5, test accuracy decreased and train accuracy increased. Loss and accuracy graphs were plotted using the same settings in order to conduct further tests to determine the best accuracy.

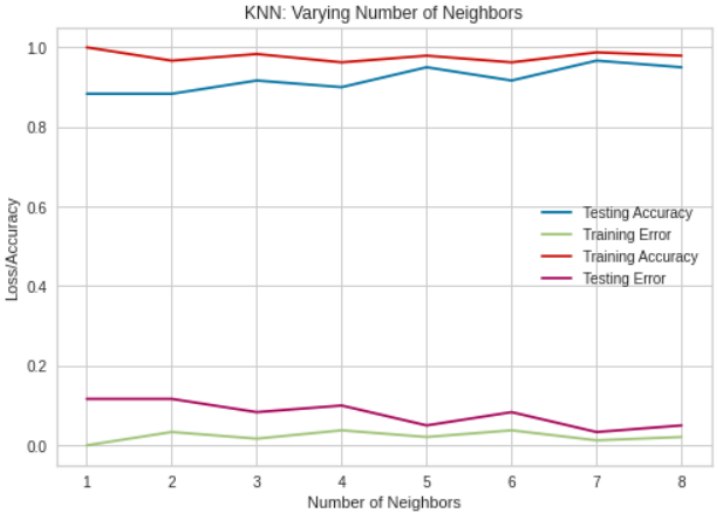

Figure 4.40: K-NN model's loss / accuracy vs no. of neighbors plot on the train and test Dataset2 when value of k is 7 and p is 1 (Manhattan Distance), attained optimal results

The graph indicated when testing accuracy is increased, testing error is reduced and when training accuracy is increased, training error decreased on "k" with 7 when "p"

with value 1. This is shown in Figure 4.40. Moreover, when testing accuracy is decreased, testing error is increased and when training accuracy is increased, training error decreased on "k" with 3 and "k" with 5 when p=1.

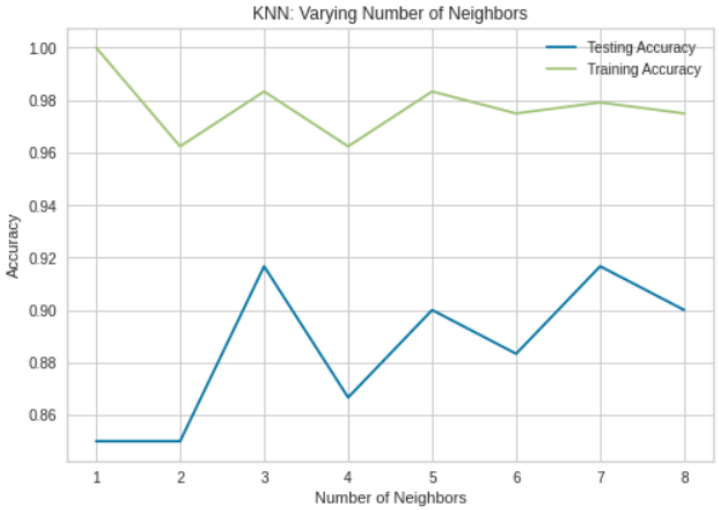

Figure 4.41: K-NN model's no. of neighbors vs accuracy plot on the train and test Dataset2 when value of k is 3 and p is 2 (Euclidean Distance)

The above graph illustrated that "k" with 3 and "p" with 2 gave the test and train accuracy of 92% and 98.2% respectively is shown in Figure 4.41. Again, that point was emphasized where both the test and train values got increased. At "k" with value 7, test and train both accuracies are also increased but these accuracies are below as compared to "k" with 3 value so this point is ignored. In order to make more test to find the best accuracy, loss and accuracy graph was plotted with same parameters.

The graph given below indicated, when testing accuracy is increased, testing error is decreased and when training accuracy is increased, training error decreased on "k" with 3 when "p" as value 2 is illustrated in Figure 4.42. Moreover, when testing accuracy is increased, testing error is reduced and when training accuracy is increased, training error decreased on "k" with 5 but after that training accuracy and training error has shown somewhat straight on "k" with 7 and "p" as value 2.

From above all, the testing accuracy was 96.67% and training accuracy on which the model was trained has attained 98.7% accuracy on "k" with 7 and "p" as 1 gave the optimal results. Moreover, cross-validation was applied using cv level "10" and presented test accuracy 95% and train accuracy 97% on same parameters of "k" and "p".

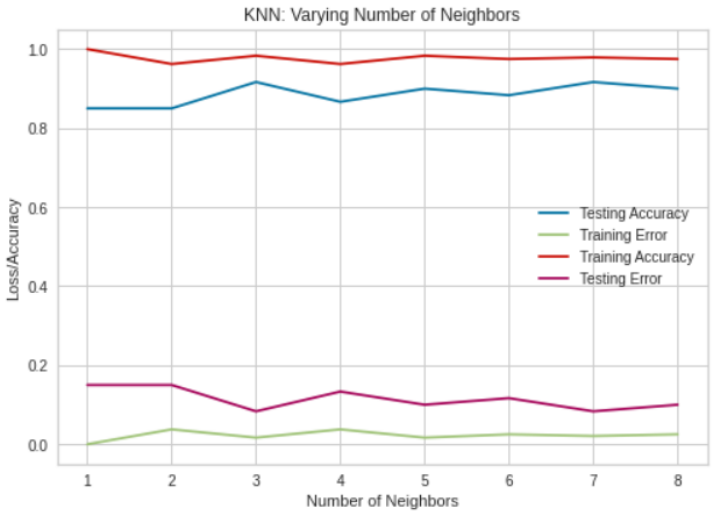

Figure 4.42: K-NN model's loss / accuracy vs no. of neighbors plot on the train and test Dataset2 when value of k is 7 and p is 2 (Euclidean Distance)

K-NN confusion matrix using heatmap is illustrated in the Figure 4.43 below has shown 58 correct predictions and 2 incorrect predictions.

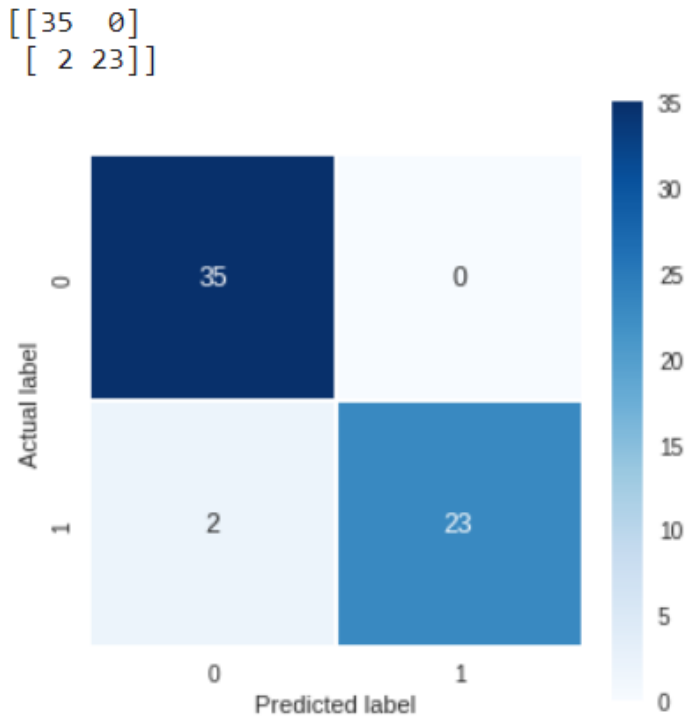

Figure 4.43: K-NN model's confusion matrix of Dataset2

In K-NN's classification report, precision has depicted that the predicted model had a significant influence on accurately predicting a favorable risk outcome i.e. 100%. Recall revealed that the model properly predicted the result for 92% of the risk factors. The f1-score is 96%. The support value demonstrated an instance of how risk belonged to each class in the test dataset, with 35 having low risk and 25 having high risk effect is shown in Figure 4.44.

	precision	recall	f1-score	support
0	0.95	1.00	0.97	35
1	1.00	0.92	0.96	25
accuracy			0.97	60
macro avg	0.97	0.96	0.97	60
weighted avg	0.97	0.97	0.97	60

Figure 4.44: K-NN model's classification report of Dataset2

4.2 Final Results and Findings

Based on the best tested and resulted parameters, both the test and train accuracies are plotted for which bar graphs against both Dataset1 and Datset2 with their accuracies are shown in Figure 4.45 and Figure 4.46 by keeping in view accuracies found in all applied ML techniques.

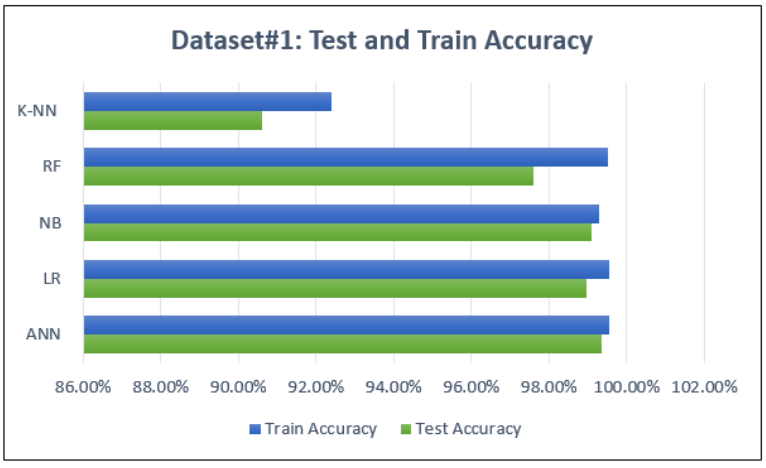

Figure 4.45: Final findings of test and train accuracies of each ML techniques in Dataset1

While comparing both the datasets, the most optimal results are provided by the ANN model i.e. test accuracy of 99.34% in Dataset1 and 99.2% in Dataset2. Second level best

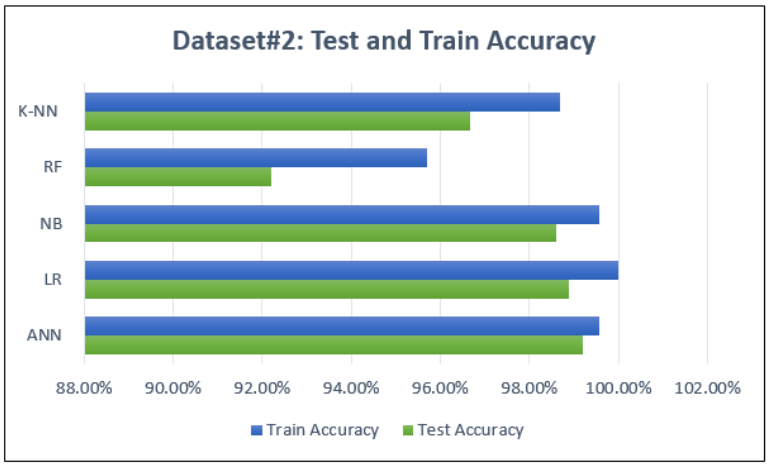

Figure 4.46: Final findings of test and train accuracies of each ML techniques in Dataset2

accuracy is provided by the NB model in Dataset1 which is 99.1% and LR in Dataset2 which is 98.88%. In addition, LR and NB has provided comparable results in both the datasets. However, K-NN with 90.6% in Dataset1 and RF with 92.2% in Dataset2 have showed decline as compared to other applied ML techniques. The findings of test and train accuracies attained in each ML technique in Dataset1 and Dataset2 are highlighted in Table 4.1.

Table 4.1: Final findings of test and train accuracies attained for each ML technique in Dataset1 and Dataset2

Benchmark Datasets	Inputs	Outputs	Algorithms	Evaluation Metric(s)
A machine learning model to predict project risk	20 risk attributes with 70 projects	Binary classification i.e. 1 - high risk, 0 - low risk	Test and Train Accuracy of ANN, LR, NB, RF, K-NN	ANN= 99.34%, 99.56% LR= 98.96%, 99.55% NB= 99.1%, 99.3% RF= 97.6%, 99.5% K-NN= 90.6%, 92.4%
Empirical Assessment of Machine Learning Techniques for Software Requirements Risk Prediction	14 risk attributes with 5 levels of risks identified and including 4 projects	Multi-class classification problem to a binary classification i.e. 3,4,5 - 1 (high risk), 1,2 - 0 (low risk)	Test and Train Accuracy of ANN, LR, NB, RF, K-NN	ANN= 99.2%, 99.58%, LR= 98.88%, 100% NB= 98.6%, 99.56% RF= 92.2%, 95.7% K-NN= 96.67%, 98.7%

In both Dataset1 and Dataset2, following parameters are used and found accuracies, precision, recall, and f1-score against each applied ML technique. Results found in ANN, LR, NB, RF, and K-NN models are shown in Table 4.2, 4.3, 4.4, 4.5 and 4.6 respectively.

Table 4.2: Types of parameters, test and train accuracies, precision, recall, and f1-score used for ANN model in Dataset1 and Dataset2

Dataset1	**Parameters**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
ANN	Test size=0.3 Train size=0.7 Units=32 (input layer) Units=12 (hidden layers) Classifier=Sequential Activation ='relu' (input and hidden layers) Activation='sigmoid' (ouput layer) Optimizer='adam' Random state=2 Batch size=40 Epochs=145 Dropout=0.2 Cv=10	Test=99.34% Train=99.56%	0.99 1.00	1.00 0.97	1.00 0.98
Dataset2	Parameters	Accuracy	Precision	Recall	F1-Score
ANN	Test size=0.2 Train size=0.8 Units=16 (input) Units=14 (hidden layers) Classifier= Sequential Activation='relu' (input and hidden layers) Activation='sigmoid' (output layer) Optimizer='adam' Random state=42 Batch size=32 Epochs=100 Dropout=0.2 LR= 1e-4 Cv=10	Test=99.2% Train=99.58%	0.99 1.00	1.00 0.97	1.00 0.98

Table 4.3: Types of parameters, test and train accuracies, precision, recall, and f1-score used for LR model in Dataset1 and Dataset2

Dataset1	**Parameters**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
LR	Test size=0.1 Train size=0.9 Classifier=LogisticRegression Penalty=L2 Solver= 'lbfgs' Random state=101 Cv=5	Test=98.96% Train=99.55%	0.99 1.00	1.00 0.96	0.99 0.98
Dataset2	Parameters	Accuracy	Precision	Recall	F1-Score
LR	Test size=0.3 Train size=0.7 Classifier=LogisticRegression Penalty=L2 Solver= 'lbfgs' Random state=42 Cv=5	Test=98.88% Train=100%	0.98 1.00	1.00 0.98	0.99 0.99

Table 4.4: Types of parameters, test and train accuracies, precision, recall, and f1-score used for NB model in Dataset1 and Dataset2

Dataset1	**Parameters**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
NB	Test size=0.25 Train size=0.75 Classifier=GaussianNB Random state=0 Cv=6	Test=99.1% Train=99.3%	1.00 0.95	0.99 1.00	0.99 0.98
Dataset2	Parameters	Accuracy	Precision	Recall	F1-Score
NB	Test size=0.25 Train size=0.75 Classifier=GaussianNB Random state=0 Cv=6	Test=98.6% Train=99.56%	1.00 0.97	0.98 1.00	0.99 0.98

Table 4.5: Types of parameters, test and train accuracies, precision, recall, and f1-score used for RF model in Dataset1 and Dataset2

Dataset1	**Parameters**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
RF	Test size=0.3 Train size=0.7 Classifier=RandomForestClassifier Random state=1 Stratify=y Criterion='gini' N_Estimators=4 N_jobs=-1 Min samples leaf=5 Max depth=4 Cv=10	Test=97.60% Train=99.50%	0.97 1.00	1.00 0.90	0.98 0.94
Dataset2	Parameters	Accuracy	Precision	Recall	F1-Score
RF	Test size=0.3 Train size=0.7 Classifier=RandomForestClassifier Random state=2 Stratify=y Criterion='gini' N_Estimators=2, 25 N_jobs=-1 Min samples leaf=5 Max depth=4 Cv=10	Test=92.2% Train=95.7%	0.88 1.00	1.00 0.83	0.93 0.91

Table 4.6: Types of parameters, test and train accuracies, precision, recall, and f1-score used for K-NN model in Dataset1 and Dataset2

Dataset1	**Parameters**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
K-NN	Test size=0.2 Train size=0.8 Classifier=KNeighborsClassifier Random state=0 N_neighbors=2 Metric= 'minkowski' P=1 (Manhattan Distance) Cv=10	Test=90.60% Train=92.40%	0.91 0.88	0.99 0.48	0.95 0.62
Dataset2	Parameters	Accuracy	Precision	Recall	F1-Score
K-NN	Test size=0.2 Train size=0.8 Classifier=KNeighborsClassifier Random state=0 N_neighbors=7 Metric= 'minkowski' P=1 (Manhattan Distance) Cv=10	Test=96.67% Train=98.7%	0.95 1.00	1.00 0.92	0.97 0.96

When comparing the resulted accuracies of previous research work with our work, most significant accuracies are found in test accuracies wherein all the applied ML techniques has beaten the results of the previous works in both the datasets, shown in graph given below 4.47. In Dataset1, all the applied ML techniques has attained accuracies above 90% which was around 70% in previous work. Similarly, all the applied ML algorithms has shown accuracies above 90% in Dataset2. In addition, two more ML techniques are applied in both Dataset1 and Dataset2 i.e. ANN and LR wherein ANN model has shown the best accuracy results overall. Even K-NN with lowest accuracy 90.60% in Dataset1 and RF with lowest accuracy 92.20% in Dataset2 has beaten results plus NB has achieved good results in both the Datasets as compared to previous works.

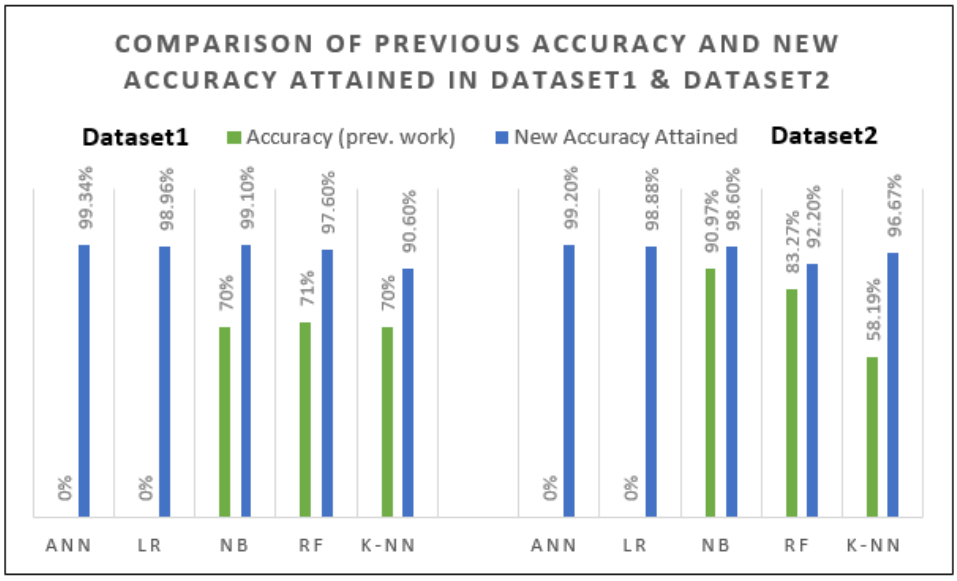

Figure 4.47: Comparison of Previous and New Research Work in Dataset1 and Dataset2

It is determined that the proposed methodology approach has increased accuracy in detecting the high risk factors while reducing development efforts in PM based applications to meet the project's deadline. The outcomes attained in this work, specially ANN model resulted in leading the development strategies and highlighted main elements pertaining to assess, manage, control and reduce the risk constraints in application's development.

4.3 Summary

By properly analyzing the datasets, after data preprocessing and feature engineering different optimized level parameters are applied by following the hyperparameter techniques in order to attain the best accuracy results. By comparing the test accuracy in both the datasets, ANN has gained a competing edge in both the datasets as compared to all other ML techniques applied whereas NB has attained the accuracy of second level. In order to visualize the outcomes, confusion matrix is created using heatmap and classification reports are generated to check the performance against each ML technique.

Chapter 5

Conclusion and Future Directions

5.1 Conclusion

A growing number of researchers are working and contributing on the dynamic field of risk prediction in different PM applications. We have conducted a comprehensive overview of ML algorithms for PM based applications to determine the risk assessment in project timeline. This study intends to investigate ML methods using risk-related datasets. Five ML techniques were investigated for risk prediction, and a thorough comparison of these techniques is performed for the proposed model.

The datasets are used to build ML models and trained to predict the risk levels based on different attributes of Dataset1 and Dataset2 such as project risk liklihood, risk previous, priority, estimated days finished, report count, status, dimension, probability, and the magnitude to determine how risk factor can effect the project timeline management and whether the impact of risk is high or low. Different parameters are plotted and tested graphically to predict and find best model accuracies. In this research, five distinct ML techniques are contrasted and evaluated for reducing error rates and increasing accuracy of the proposed model. Among all these techniques while comparing the benchmarked datasets, ANN in Dataset1 and Dataset2 has attained greater test accuracy i.e. 99.34% and 99.2%, as compared to other ML techniques. LR and NB in both datasets has provided comparable results while RF in Dataset1 has improved in test accuracy i.e. 97.6% as compared to Dataset2. Furthermore, K-NN in Dataset2 gave better results as compared to K-NN in Dataset1 and executed 96.67% of accuracy.

When comparing the ML techniques utilized in this work to earlier research works, both the datasets has depicted good outputs in which ANN outperformed and attained the best accuracy. Moreover, by adopting data preprocessing, feature selection, and hyperparameter tuning in five ML approaches, both Dataset1 and Dataset2 accuracies has improved in terms of test and train, as seen in Figure 4.47 and Table 4.1 when comparing the findings of this research.

5.2 Future Research Directions

The thorough findings of this study can serve as a guideline for other research works. Regarding improvements in risk prediction can be made by adopting any other techniques or any framework. There are also some directions for the future stated because the proposed methodology might use different parameters and pre-processing methods in order to reduce the errors and enhance the accuracy of model/s.

It may also bring new challenges in the risk prediction process to assess its effect on the project timeline and it may addressed with practical solutions in diverse PM applications. Finally, we believe that our study on ML-based applications serves as a technical reference for experts, professionals and decision-makers in business and academia, as well as for future research and PM based solutions.

References

[1] AA Goryachev, AV Goryachev, AV Monakhov, and NE Novakova. Project management system in computer aided design. In *2016 XIX IEEE International Conference on Soft Computing and Measurements (SCM)*, pages 221–223. IEEE, 2016.

[2] Milind Padalkar and Saji Gopinath. Are complexity and uncertainty distinct concepts in project management? a taxonomical examination from literature. *International Journal of Project Management*, 34(4):688–700, 2016.

[3] Celia Desmond. Project management tools-beyond the basics. *IEEE Engineering Management Review*, 45(3):25–26, 2017.

[4] Muhammad Shah Jahan, Muhammad Talha Riaz, Khawaja Sarmad Arif, and Muhammad Abbas. Software project management and its tools in practice in it industry of pakistan. In *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–6. IEEE, 2019.

[5] Roger S Pressman. *Software engineering: a practitioner's approach*. Palgrave macmillan, 2005.

[6] Franck Marle. An assistance to project risk management based on complex systems theory and agile project management. *Complexity*, 2020, 2020.

[7] Eng Seng Chia. Risk assessment framework for project management. In *2006 IEEE International Engineering Management Conference*, pages 376–379. IEEE, 2006.

[8] Muhammad Naeem Ahmed Khan, Aamir Mehmood Mirza, and Imran Saleem. Software risk analysis with the use of classification techniques: A review. *Engineering, Technology & Applied Science Research*, 10(3):5678–5682, 2020.

[9] Mohammed Najah Mahdi, Mohd Hazli Mohamed Zabil, Abdul Rahim Ahmad, Roslan Ismail, Yunus Yusoff, Lim Kok Cheng, Muhammad Sufyian Bin Mohd Azmi, Hayder Natiq, and Hushalini Happala Naidu. Software project management using machine learning technique—a review. *Applied Sciences*, 11(11), 2021. ISSN 2076-3417. URL https://www.mdpi.com/2076-3417/11/11/5183.

[10] Vlad-Sebastian Ionescu. An approach to software development effort estimation using machine learning. In *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 197–203. IEEE, 2017.

[11] Benjamin Schreck, Shankar Mallapur, Sarvesh Damle, Nitin John James, Sanjeev Vohra, Rajendra Prasad, and Kalyan Veeramachaneni. Augmenting software project managers with predictions from machine learning. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2004–2011. IEEE, 2018.

[12] Tabish Hanif Shaikh, Faisal Latif Khan, NoorAli Alimuddin Shaikh, Haidarali Nadir Shah, and Zainab Pirani. Survey of web-based project management system. In *2018 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pages 438–441. IEEE, 2018.

[13] Suriyan Jomthanachai, Wai-Peng Wong, and Chee-Peng Lim. An application of data envelopment analysis and machine learning approach to risk management. *Ieee Access*, 9:85978–85994, 2021.

[14] Maria Ilaria Lunesu, Roberto Tonelli, Lodovica Marchesi, and Michele Marchesi. Assessing the risk of software development in agile methodologies using simulation. *IEEE Access*, 9:134240–134258, 2021.

[15] Vladimir G Psoyants, Alexandr I Taganov, Aleksandr N Kolesenkov, and Irina V Bodrova. Risk management technology of software project sustainability in fuzzy conditions. In *2019 8th Mediterranean Conference on Embedded Computing (MECO)*, pages 1–4. IEEE, 2019.

[16] Anié Bermudez Peña, Gilberto Fernando Castro, Diana María Lúpez Alvarez, Inelda Anabelle Martillo Alcivar, Giselle Lorena Núñez, Danny Saavedra Cevallos, and Jorge Luis Zambrano Santa. Method for project execution control based on soft computing and machine learning. In *2019 XLV Latin American Computing Conference (CLEI)*, pages 1–7. IEEE, 2019.

[17] Gio Wiederhold and John McCarthy. Arthur samuel: Pioneer in machine learning. *IBM Journal of Research and Development*, 36(3):329–331, 1992.

[18] Susmita Ray. A quick review of machine learning algorithms. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pages 35–39, 2019. doi: 10.1109/COMITCon.2019.8862451.

[19] Iqbal H Sarker. Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, 2(3):1–21, 2021.

[20] Amanpreet Singh, Narina Thakur, and Aakanksha Sharma. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1310–1315, 2016.

[21] Raffaele Pugliese, Stefano Regondi, and Riccardo Marini. Machine learning-based approach: global trends, research directions, and regulatory standpoints. *Data Science and Management*, 4:19–29, 2021. ISSN 2666-7649. doi: https://doi.org/10.1016/j.dsm.2021.12.002. URL https://www.sciencedirect.com/science/article/pii/S2666764921000485.

[22] Rishabh Choudhary and Hemant Kumar Gianey. Comprehensive review on supervised machine learning algorithms. In *2017 International Conference on Machine Learning and Data Science (MLDS)*, pages 37–43, 2017. doi: 10.1109/MLDS.2017.11.

[23] Mengyu Huang. Theory and implementation of linear regression. In *2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL)*, pages 210–217, 2020. doi: 10.1109/CVIDL51233.2020.00-99.

[24] Xiaonan Zou, Yong Hu, Zhewen Tian, and Kaiyuan Shen. Logistic regression model optimization and case analysis. In *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, pages 135–139, 2019. doi: 10.1109/ICCSNT47585.2019.8962457.

[25] Zan Yang and Dan Li. Application of logistic regression with filter in data classification. In *2019 Chinese Control Conference (CCC)*, pages 3755–3759, 2019. doi: 10.23919/ChiCC.2019.8865281.

[26] Zhaolong Gao, Mengvi P. Gatpandan, and Paulino H. Gatpandan. Classification decision tree algorithm in predicting students' course preference. In *2021 2nd International Symposium on Computer Engineering and Intelligent Communications (ISCEIC)*, pages 93–97, 2021. doi: 10.1109/ISCEIC53685.2021.00026.

[27] Jing Huang, Jinzhi Zhou, and Linwen Zheng. Support vector machine classification algorithm based on relief-f feature weighting. In *2020 International Conference on Computer Engineering and Application (ICCEA)*, pages 547–553, 2020. doi: 10.1109/ICCEA50009.2020.00121.

[28] Jakub Nalepa and Michal Kawulok. Selecting training sets for support vector machines: a review. *Artificial Intelligence Review*, 52(2):857–900, 2019.

[29] Tannu Chauhan, Surbhi Rawat, Samrath Malik, and Pushpa Singh. Supervised and unsupervised machine learning based review on diabetes care. In *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, volume 1, pages 581–585, 2021. doi: 10.1109/ICACCS51430.2021.9442021.

[30] Md. Golam Rabiul Alam, Sajjad Hussain, Md. Mofaqkhayrul Islam Mim, and Md Tarikul Islam. Telecom customer behavior analysis using naive bayes classifier. In *2021 IEEE 4th International Conference on Computer and Communication Engineering Technology (CCET)*, pages 308–312, 2021. doi: 10.1109/CCET52649.2021.9544169.

[31] Feng-Jen Yang. An implementation of naive bayes classifier. In *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 301–306, 2018. doi: 10.1109/CSCI46756.2018.00065.

[32] Jahanzaib Javid, Muhammad Ali Mughal, and Mustansir Karim. Using knn algorithm for classification of distribution transformers health index. In *2021 International Conference on Innovative Computing (ICIC)*, pages 1–6, 2021. doi: 10.1109/ICIC53490.2021.9693013.

[33] Yunsheng Song, Xiaohan Kong, and Chao Zhang. A large-scale-nearest neighbor classification algorithm based on neighbor relationship preservation. *Wireless Communications and Mobile Computing*, 2022, 2022.

[34] Pengbo Wang, Yongqiang Zhang, and Wenting Jiang. Application of k-nearest neighbor (knn) algorithm for human action recognition. In *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, volume 4, pages 492–496, 2021. doi: 10.1109/IMCEC51613.2021.9482165.

[35] Syed Kumayl Raza Moosavi, Muhammad Hamza Zafar, Malik Naveed Akhter, Shahzaib Farooq Hadi, Noman Mujeeb Khan, and Filippo Sanfilippo. A novel artificial neural network (ann) using the mayfly algorithm for classification. In *2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2)*, pages 1–6, 2021. doi: 10.1109/ICoDT252288.2021.9441473.

[36] T. H. F Harumy, M. Zarlis, S. Effendi, and M. S Lidya. Prediction using a neural network algorithm approach (a review). In *2021 International Conference on Software Engineering Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)*, pages 325–330, 2021. doi: 10.1109/ICSECS52883.2021.00066.

[37] Omar Al-Salman, Jamila Mustafina, and Gailan Shahoodh. A systematic review of artificial neural networks in medical science and applications. In *2020 13th International Conference on Developments in eSystems Engineering (DeSE)*, pages 279–282, 2020. doi: 10.1109/DeSE51703.2020.9450245.

[38] Miel Verkerken, Laurens D'hooge, Tim Wauters, Bruno Volckaert, and Filip De Turck. Unsupervised machine learning techniques for network intrusion detection on modern data. In *2020 4th Cyber Security in Networking Conference (CSNet)*, pages 1–8, 2020. doi: 10.1109/CSNet50428.2020.9265461.

[39] Jelili Oyelade, Itunuoluwa Isewon, Olufunke Oladipupo, Onyeka Emebo, Zacchaeus Omogbadegun, Olufemi Aromolaran, Efosa Uwoghiren, Damilare Olaniyan, and Obembe Olawole. Data clustering: Algorithms and its applications. In *2019 19th International Conference on Computational Science and Its Applications (ICCSA)*, pages 71–81, 2019. doi: 10.1109/ICCSA.2019.000-1.

[40] Patrick Schneider and Fatos Xhafa. Chapter 8 - machine learning: Ml for ehealth systems. In Patrick Schneider and Fatos Xhafa, editors, *Anomaly Detection and Complex Event Processing over IoT Data Streams*, pages 149–191.

Academic Press, 2022. ISBN 978-0-12-823818-9. doi: https://doi.org/10.1016/B978-0-12-823818-9.00019-5. URL https://www.sciencedirect.com/science/article/pii/B9780128238189000195.

[41] Stephanie Kay Ashenden, Aleksandra Bartosik, Paul-Michael Agapow, and Elizaveta Semenova. Chapter 2 - introduction to artificial intelligence and machine learning. In Stephanie Kay Ashenden, editor, *The Era of Artificial Intelligence, Machine Learning, and Data Science in the Pharmaceutical Industry*, pages 15–26. Academic Press, 2021. ISBN 978-0-12-820045-2. doi: https://doi.org/10.1016/B978-0-12-820045-2.00003-9. URL https://www.sciencedirect.com/science/article/pii/B9780128200452000039.

[42] Giampaolo Casolla, Salvatore Cuomo, Vincenzo Schiano di Cola, and Francesco Piccialli. Exploring unsupervised learning techniques for the internet of things. *IEEE Transactions on Industrial Informatics*, 16(4):2621–2628, 2020. doi: 10.1109/TII.2019.2941142.

[43] Peter Wittek. 5 - unsupervised learning. In Peter Wittek, editor, *Quantum Machine Learning*, pages 57–62. Academic Press, Boston, 2014. ISBN 978-0-12-800953-6. doi: https://doi.org/10.1016/B978-0-12-800953-6.00005-0. URL https://www.sciencedirect.com/science/article/pii/B9780128009536000050.

[44] Vasileios C. Pezoulas, Themis P. Exarchos, and Dimitrios I. Fotiadis. Chapter 7 - machine learning and data analytics. In Vasileios C. Pezoulas, Themis P. Exarchos, and Dimitrios I. Fotiadis, editors, *Medical Data Sharing, Harmonization and Analytics*, pages 227–309. Academic Press, 2020. ISBN 978-0-12-816507-2. doi: https://doi.org/10.1016/B978-0-12-816507-2.00007-4. URL https://www.sciencedirect.com/science/article/pii/B9780128165072000074.

[45] Jorge Garza-Ulloa. Chapter 6 - application of mathematical models in biomechatronics: artificial intelligence and time-frequency analysis. In Jorge Garza-Ulloa, editor, *Applied Biomechatronics using Mathematical Models*, pages 373–524. Academic Press, 2018. ISBN 978-0-12-812594-6. doi: https://doi.org/10.1016/B978-0-12-812594-6.00006-8. URL https://www.sciencedirect.com/science/article/pii/B9780128125946000068.

[46] Pawel Cichosz. *Hierarchical clustering*, pages 349–372. 2015. doi: 10.1002/9781118950951.ch13.

[47] Javier Diaz-Rozo, Concha Bielza, and Pedro Larrañaga. Clustering of data streams with dynamic gaussian mixture models: An iot application in industrial processes. *IEEE Internet of Things Journal*, 5(5):3533–3547, 2018. doi: 10.1109/JIOT.2018.2840129.

[48] Angela Serra and Roberto Tagliaferri. Unsupervised learning: Clustering. In Shoba Ranganathan, Michael Gribskov, Kenta Nakai, and Christian Schönbach, editors, *Encyclopedia of Bioinformatics and Computational Biology*, pages 350–357. Academic Press, Oxford, 2019. ISBN 978-0-12-811432-2. doi: https://doi.org/10.1016/B978-0-12-809633-8.20487-1. URL https://www.sciencedirect.com/science/article/pii/B9780128096338204871.

[49] Abdulhamit Subasi. Chapter 3 - machine learning techniques. In Abdulhamit Subasi, editor, *Practical Machine Learning for Data Analysis Using Python*, pages 91–202. Academic Press, 2020. ISBN 978-0-12-821379-7. doi: https://doi.org/10.1016/B978-0-12-821379-7.00003-5. URL https://www.sciencedirect.com/science/article/pii/B9780128213797000035.

[50] Sudhakar Singh, Rakhi Garg, and Pramod Kumar Mishra. Performance analysis of apriori algorithm with different data structures on hadoop cluster. *Materials Processing & Manufacturing eJournal*, 2015.

[51] Ibrahim Obeidat, Wafa' Eleisah, and Kinda Magableh. Dimensionality reduction and supervised learning for intrusion detection. In *2022 8th International Conference on Information Management (ICIM)*, pages 86–91, 2022. doi: 10.1109/ICIM56520.2022.00023.

[52] Afnan M. Alhassan and Wan Mohd Nazmee Wan Zainon. Review of feature selection, dimensionality reduction and classification for chronic disease diagnosis. *IEEE Access*, 9:87310–87317, 2021. doi: 10.1109/ACCESS.2021.3088613.

[53] S. Velliangiri, S. Alagumuthukrishnan, and S Iwin Thankumar joseph. A review of dimensionality reduction techniques for efficient computation. *Procedia Computer Science*, 165:104–111, 2019. ISSN 1877-0509. doi: https://doi.org/10.1016/

j.procs.2020.01.079. URL https://www.sciencedirect.com/science/article/pii/S1877050920300879. 2nd International Conference on Recent Trends in Advanced Computing ICRTAC -DISRUP - TIV INNOVATION , 2019 November 11-12, 2019.

[54] Nico Migenda, Ralf Möller, and Wolfram Schenck. Adaptive dimensionality reduction for neural network-based online principal component analysis. *PloS one*, 16 (3):e0248896, 2021.

[55] Steven C. S. Ng. Principal component analysis to reduce dimension on digital image. *Procedia Computer Science*, 111:113–119, 2017.

[56] Yousef Jaradat, Mohammad Masoud, Ismael Jannoud, Ahmad Manasrah, and Mohammad Alia. A tutorial on singular value decomposition with applications on image compression and dimensionality reduction. In *2021 International Conference on Information Technology (ICIT)*, pages 769–772, 2021. doi: 10.1109/ICIT52682.2021.9491732.

[57] Brozolal Mondol, Romana Rahman Ema, Tajul Islam, and Md.Tanvir Islam. Singular value decomposition and detection of color histogram of an image by using wavelet based symmetry reflection. In *2019 2nd International Conference on Innovation in Engineering and Technology (ICIET)*, pages 1–5, 2019. doi: 10.1109/ICIET48527.2019.9290708.

[58] Quentin Fournier and Daniel Aloise. Empirical comparison between autoencoders and traditional dimensionality reduction methods. In *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 211–214, 2019. doi: 10.1109/AIKE.2019.00044.

[59] Reyhan Kevser Keser and Behçet Ugur Töreyin. Autoencoder based dimensionality reduction of feature vectors for object recognition. In *2019 15th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, pages 577–584, 2019. doi: 10.1109/SITIS.2019.00097.

[60] Rupesh Babu Shrestha, Mahsa Razavi, and P.W.C Prasad. An unsupervised machine learning technique for recommendation systems. In *2020 5th International Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA)*, pages 1–9, 2020. doi: 10.1109/CITISIA50690.2020.9371817.

[61] Gustavo Betarte, Álvaro Pardo, and Rodrigo Martínez. Web application attacks detection using machine learning techniques. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1065–1072. IEEE, 2018.

[62] Shashank Mouli Satapathy and Santanu Kumar Rath. Effort estimation of web-based applications using machine learning techniques. In *2016 International conference on advances in computing, communications and informatics (ICACCI)*, pages 973–979. IEEE, 2016.

[63] Kamil Židek, Ján Pitel', and Alexander Hošovskỳ. Machine learning algorithms implementation into embedded systems with web application user interface. In *2017 IEEE 21st International Conference on Intelligent Engineering Systems (INES)*, pages 000077–000082. IEEE, 2017.

[64] Muhammad Noman, Muhammad Iqbal, and Amir Manzoor. A survey on detection and prevention of web vulnerabilities. *International Journal of Advanced Computer Science and Applications*, 11(6), 2020.

[65] Przemyslaw Pospieszny, Beata Czarnacka-Chrobot, and Andrzej Kobylinski. An effective approach for software project effort and duration estimation with machine learning algorithms. *Journal of Systems and Software*, 137:184–196, 2018.

[66] Mahesh Babu Pasupuleti. The application of machine learning techniques in software project management-an examination. *ABC Journal of Advanced Research*, 7 (2):113–122, 2018.

[67] Martin Lindovsky and Christoffer Brasjö. Machine learning project management-a study of project requirements and processes in early adoption. Master's thesis, 2019.

[68] Murat Pasa Uysal. Machine learning and data science project management from an agile perspective: Methods and challenges. In *Contemporary Challenges for Agile Project Management*, pages 73–88. IGI Global, 2022.

[69] Wang Wei and Muhammad Ehsan Rana. Software project schedule management using machine learning & data mining. *International Journal of Scientific & Technology Research*, 8(9):1385–1389, 2019.

[70] Fumihiro Kumeno. Sofware engneering challenges for machine learning applications: A literature review. *Intelligent Decision Technologies*, 13(4):463–476, 2019.

[71] Nathalie Esmée Janssen. A machine learning proposal for predicting the success rate of it-projects based on project metrics before initiation. B.S. thesis, University of Twente, 2019.

[72] Anié Bermudez Peña, Gilberto Fernando Castro, Diana María Lúpez Alvarez, Inelda Anabelle Martillo Alcivar, Giselle Lorena Núñez, Danny Saavedra Cevallos, and Jorge Luis Zambrano Santa. Method for project execution control based on soft computing and machine learning. In *2019 XLV Latin American Computing Conference (CLEI)*, pages 1–7, 2019. doi: 10.1109/CLEI47609.2019.235097.

[73] Augusto Hayashida Marchinares and Igor Aguilar-Alonso. Project portfolio management studies based on machine learning and critical success factors. In *2020 IEEE International Conference on Progress in Informatics and Computing (PIC)*, pages 369–374, 2020. doi: 10.1109/PIC50277.2020.9350787.

[74] TQD Pham, T Le-Hong, and XV Tran. Efficient estimation and optimization of building costs using machine learning. *International Journal of Construction Management*, pages 1–13, 2021.

[75] Senerath Mudalige Don Alexis Chinthaka Jayatilake and Gamage Upeksha Ganegoda. Involvement of machine learning tools in healthcare decision making. *Journal of Healthcare Engineering*, 2021, 2021.

[76] Jeevith Hegde and Børge Rokseth. Applications of machine learning methods for engineering risk assessment–a review. *Safety science*, 122:104492, 2020.

[77] Zain Shaukat Shaukat, Rashid Naseem, and Muhammad Zubair. A dataset for software requirements risk prediction. In *2018 IEEE International Conference on Computational Science and Engineering (CSE)*, pages 112–118, 2018. doi: 10.1109/CSE.2018.00022.

[78] R Naseem, Z Shaukat, M Irfan, MA Shah, A Ahmad, F Muhammad, A Glowacz, L Dunai, JA Antonino-Daviu, and A Sulaiman. Empirical assessment of machine learning techniques for software requirements risk prediction. electronics 2021, 10, 168, 2021.

[79] Mohammad Mahmood Otoom. Abmj: An ensemble model for risk prediction in software requirements. *IJCSNS*, 22(3):710, 2022.

[80] Mohammad Ahmad Ibraigheeth and Syed Abdullah Fadzli. Software project failures prediction using logistic regression modeling. In *2020 2nd International Conference on Computer and Information Sciences (ICCIS)*, pages 1–5. IEEE, 2020.

[81] André Oliveira Sousa. Assessing risks in software projects through machine learning approaches. 2021.

[82] M Hanefi Calp and M Ali Akcayol. A novel model for risk estimation in software projects using artificial neural network. In *The International Conference on Artificial Intelligence and Applied Mathematics in Engineering*, pages 295–319. Springer, 2019.

[83] Yong Hu, Jiaxing Huang, Juhua Chen, Mei Liu, and Kang Xie. Software project risk management modeling with neural network and support vector machine approaches. In *Third International Conference on Natural Computation (ICNC 2007)*, volume 3, pages 358–362, 2007. doi: 10.1109/ICNC.2007.672.

[84] Wen-Ming Han. Discriminating risky software project using neural networks. *Computer Standards & Interfaces*, 40:15–22, 2015.

[85] Osamu Mizuno, Takanari Hamasaki, Yasunari Takagi, and Tohru Kikuno. An empirical evaluation of predicting runaway software projects using bayesian classification. In *International Conference on Product Focused Software Process Improvement*, pages 263–273. Springer, 2004.

[86] Chandan Kumar and Dilip Kumar Yadav. A probabilistic software risk assessment and estimation model for software projects. *Procedia Computer Science*, 54:353–361, 2015.

[87] Yong Hu, Xiangzhou Zhang, EWT Ngai, Ruichu Cai, and Mei Liu. Software project risk analysis using bayesian networks with causality constraints. *Decision Support Systems*, 56:439–449, 2013.

[88] Mohammad Hossein Zavvar, Ali Yavari, Seyedeh Mozhdeh Mirhassannia, Masoume Nehi, Amangaldi Yanpi, and Mohammad Hossein Zavvar. Classification of risk in

software development projects using support vector machine. *Journal of Telecommunication, Electronic and Computer Engineering*, 9:1–5, 2017.

[89] Yong Hu, Xiangzhou Zhang, Xin Sun, Mei Liu, and Jianfeng Du. An intelligent model for software project risk prediction. In *2009 International Conference on Information Management, Innovation Management and Industrial Engineering*, volume 1, pages 629–632. IEEE, 2009.

[90] Thitima Christiansen, Pongpisit Wuttidittachotti, Somchai Prakancharoen, and S Arj-ong Vallipakorn. Prediction of risk factors of software development project by using multiple logistic regression. *ARPN Journal of Engineering and Applied Sciences*, 10(3):1324–1331, 2015.

[91] Thefonseca. A machine learning model to predict project risk, 2016. URL https://github.com/thefonseca/project-risk.

[92] Sofien Gharbi, Hamza Bellakhdar, and Salah Eddine Mrabet. Project based learning in business intelligence with intervention of companies. In *2015 IEEE Global Engineering Education Conference (EDUCON)*, pages 384–387. IEEE, 2015.

[93] Aksha Iyer, Sara Bali, Ishita Kumar, Prathamesh Churi, and Kamal Mistry. Presentation abstraction control architecture pattern in business intelligence. In *International Conference on Advances in Computing and Data Sciences*, pages 666–679. Springer, 2019.

[94] Bhavana Ramesh and Akash Ramakrishna. Unified business intelligence ecosystem: a project management approach to address business intelligence challenges. In *2018 Portland International Conference on Management of Engineering and Technology (PICMET)*, pages 1–10. IEEE, 2018.

[95] Minsang D. Tamang, Vinod Kumar Shukla, Shaista Anwar, and Ritu Punhani. Improving business intelligence through machine learning algorithms. *2021 2nd International Conference on Intelligent Engineering and Management (ICIEM)*, pages 63–68, 2021.

[96] Oswaldo Moscoso-Zea, Jorge Castro, Joel Paredes-Gualtor, and Sergio Luján-Mora. A hybrid infrastructure of enterprise architecture and business intelligence & analytics for knowledge management in education. *IEEE access*, 7:38778–38788, 2019.

[97] Pariwat Ongsulee, Veena Chotchaung, Eak Bamrungsi, and Thanaporn Rodcheewit. Big data, predictive analytics and machine learning. In *2018 16th international conference on ICT and knowledge engineering (ICT&KE)*, pages 1–6. IEEE, 2018.