INFORMATICS INSTITUTE OF TECHNOLOGY

In Collaboration with

UNIVERSITY OF WESTMINSTER

The University of Westminster, Coat of Arms

ATTRITIONPRO: AN EMPLOYEE ATTRITION PREDICTION SYSTEM USING DEEP LEARNING ENSEMBLE TECHNIQUES

A Dissertation by
Ms. Sara Yusri

Supervised by
Mr. Banuka Athuraliya

Submitted in partial fulfillment of the requirements for the BEng (Hons) in Software Engineering degree at the University of Westminster.

April 2024

DECLARATION

I hereby certify that this project report and all the artifacts associated with it are my own work, and it has not been submitted before nor is currently being submitted for any degree program.

Student Name: Sara Yusri
Registration Number: w1830237 - 20200540

Signature:

Date: 06-04-2024

ABSTRACT

Employee attrition is a problem that affects many organizations and companies. Many companies are actively strategizing and investing in employee retention strategies and experts in the field in order to retain their valuable employees. This project proposes an employee attrition prediction system, AttritionPro, that is able to utilize deep learning models applied in ensemble techniques to produce reliable and accurate predictions. The purpose of the system is to serve as a preemptive measure for dealing with employee attrition before it happens and allow managements to develop retention plans and make tactical decisions using forecasts of employee resignations.

The proposed functionalities of AttritionPro will allow HR departments to preemptively forecast employee resignations, the attrition risk level of an employee and generate a breakdown of the features contributing to that employee's attrition. This project tackles several aspects of the problem domain of Employee Attrition and contributes valuable research and insights into the problem and research domain. Ensemble methods such as stacking, voting, and simple averaging are used to combine various deep learning methods, including convolutional neural networks (CNN) and feedforward neural networks (FNN) and Wide and Deep models to achieve the best results. This study demonstrates the effectiveness of deep learning in identifying risk factors and recommending retention programs through evaluation and analysis. The results of the study indicate that Stacking gives the best accuracy and performance of these models. These findings contribute to the advancement of HR analytics and talent management practices, providing insights for organizations looking to reduce employee turnover.

Keywords: Deep Learning, Ensemble Technique, Employee Attrition, Retention Plan

Subject Descriptors:

- Computing methodologies -> Machine learning -> Machine learning algorithms -> Ensemble methods
- Social and professional topics -> Professional topics -> Management of computing and information systems -> Project and people management

PUBLICATIONS

1. **AttritionPro: An Employee Attrition Prediction System using Deep Learning Ensemble Techniques**
   Conference: EuCNC & 6G Summit 2024
   Status: Submitted, In-Review
   Venue: Antwerp, Belgium

2. **A Study on Deep Learning Models and Ensemble Techniques in Employee Attrition Prediction**
   Conference: ICoABCD 2024
   Status: Submitted, In-Review
   Venue: Indonesia

3. **A Study on Deep Learning Models and Ensemble Techniques in Employee Attrition Prediction**
   Conference: ICoABCD 2024
   Status: Submitted, In-Review
   Venue: Indonesia

ACKNOWLEDGEMENT

There are several people that I am thankful for, from the bottom of my heart, for their support and guidance throughout my final year research project. I would like to express my sincere gratitude to everyone who contributed to the completion of my research.

First and foremost, I would like to thank Mr. Banuka Athuraliya for his valuable guidance, unwavering support and constructive feedback as my supervisor throughout this research project. His wisdom, patience, and courage were instrumental in completing my project. I thank him for his mentorship and guidance, and for always being there to provide us with any help or advice whenever we needed it. I would like to thank the evaluators of my project for the time and effort they put into giving this project technical and domain evaluations which were very useful for this project.

I would also like to thank my colleagues and friends for their collaboration, support, and discussions throughout the year. Their friendship and support were invaluable in overcoming challenges and staying strong and motivated. I am also extremely thankful to my family for their love, support and understanding. Their support is the foundation of my education, and I am grateful for the trust they have placed in me.

I am also very grateful towards all the lecturers and management at the Informatics Institute of Technology for their support and guidance throughout the course of my degree.

Finally, I would like to thank everyone who contributed in many ways to the completion of this study. This work would not be possible without your support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AI	Artificial Intelligence
API	Application Programming Interface
AUC	Area Under Graph
CNN	Convolutional Neural Network
DL	Deep Learning
FNN	Feedforward Neural Network
GUI	Graphical User Interface
IoT	Internet of Things
ML	Machine Learning
OOAD	Object-Oriented Analysis and Design
RNN	Recurrent Neural Network
SDLC	Software Development Life Cycle

CHAPTER 1: INTRODUCTION

1.1 CHAPTER OVERVIEW

This chapter aims to provide an introduction to the domain of employee attrition in an organizational setting, the concerns it presents and establish the research aims and objectives of the study in relation to attempting to solve this problem. It will be an exploration into the severity of the problem, the implications it could pose, and ways in which potential solutions to the problem could be explored.

1.2 PROBLEM DOMAIN

1.2.1 The Challenge of Employee Attrition

Employee Attrition, which refers to a reduction in numbers generally associated with resignations, retirement, or death (Merriam-Webster, 2023), occurs when the inflow of employees into the company is unable to keep up with the outflow of workers. According to researchers, employee attrition is a pervasive phenomenon that occurs among all organizations (Srivastava, D. K., & Tiwari, P. K., 2020). Employee attrition poses a significant challenge to businesses in a wide variety of ways. The several negative implications include loss of expertise and knowledge, workflow disruptions, costly re-hiring processes, a decline in company morale, and customer dissatisfaction. With the COVID-19 pandemic that lasted heavily over the last few years, the term Great Resignation, otherwise known as the Big Quit was coined due to mass resignations that were being observed during this time period (Formica and Sfodera, 2022).

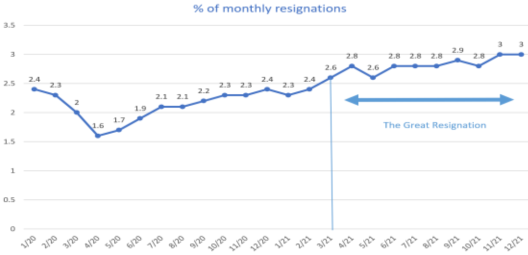


Figure 1.1 Percentage of Monthly Resignations (Self-Composed)

The figure above is a self-composed representation of the variation in the percentage of monthly resignations according to data from the US Bureau of Labor Statistics (BLS, 2021). Although it was initially decidedly attributed to the pandemic, it was later pointed out by some researchers that the uptick in resignations wasn't a phenomenon solely limited to the pandemic era. Instead, they claim that it was observed that Average Monthly Resignations were rising steadily year by year since even before the pandemic was a widespread contributor. (Fuller & Kerr, 2022).

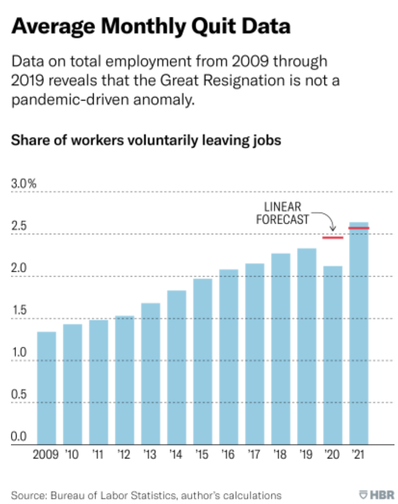


Figure 1.2 Average Monthly Quit Data (Fuller & Kerr, 2022)

Although the matter is still in dispute among various experts, the fact of the matter stands and that is the fact that resignations are a major risk factor for companies and it has been a matter of significant concern for organizations, especially during recent times (S. George, K. A. Lakshmi and K. T. Thomas, 2022). Therefore, there is a need in the corporate domain for a solution to this problem to be explored and innovated.

1.2.2 Factors Driving Employee Attrition

After analysis of the causes of the increasing levels of departures of employees from their workplaces, researchers have identified "a deep, prolonged, and widespread workforce dissatisfaction" (Formica, S. and Sfodera, F., 2022). According to De Smet, A., Dowling, B., Hancock, B., and Schaninger, B., (2022), the most common causes for the expedited resignations are lack of performance recognition, being treated with inadequate respect at the workplace, toxic work environment, or lack of sense of accomplishment both professionally and personally. These factors, along with others such as inadequate opportunities to learn and grow, poor work-life balance, lack of connection to the company's vision and mission, and a sense of being under appreciated or valued all contribute to the general dissatisfaction of employees (De Smet et al., 2022).

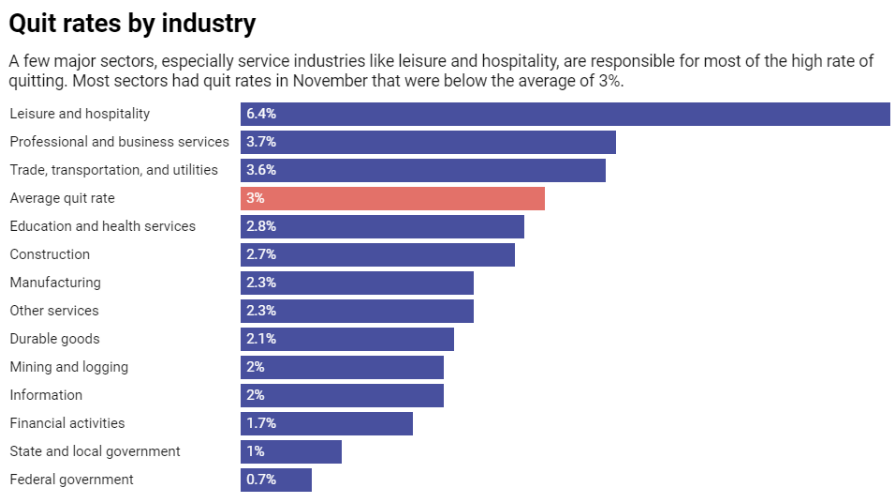


Figure 1.3 Quit Rates by Industry (Zagorsky J., 2022)

1.2.3 Prediction Systems

Given the complex nature and importance of employee attrition, organizations need to be able to pre-emptively detect whether or not their employees are at risk of resigning from the company. With the rise in popularity of Machine Learning and Artificial Intelligence systems, Prediction Systems are a prominent character present in nearly every domain that there is.

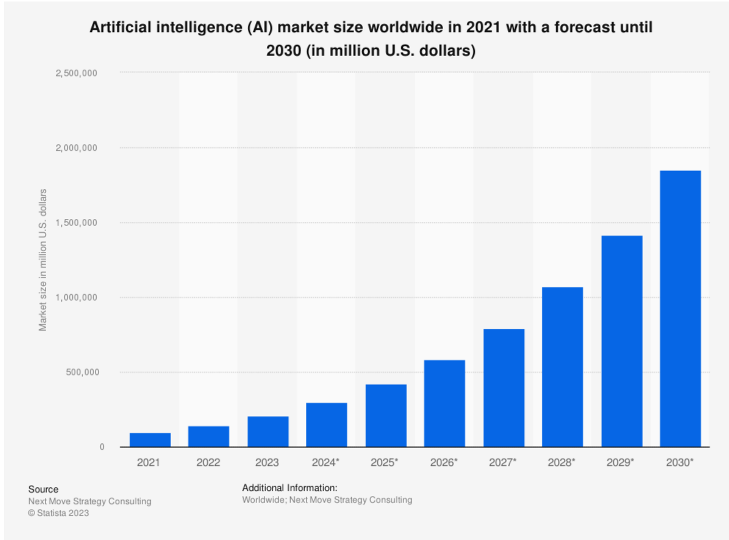


Figure 1.4 Market Size of Artificial Intelligence (Next Move Strategy Consulting, 2023)

Prediction systems provide a method of early identification of risks and in doing that, provide a preventive measure for retaining employees. According to S. George, K. A. Lakshmi, and K. T. Thomas (2022), being able to forecast employee attrition based on factors such as job satisfaction, work-life balance, and others, will provide significant help for organizations to take the proper courses of action to improve the workplace environment.

1.3 PROBLEM DEFINITION

The problem that the author is attempting to address is the growing implications of employee attrition as it affects talent retention measures, the sustainability of organizations, and in a broader context, socioeconomic progress (Chung, D. et al., 2023). The problem of Employee Attrition is a critical issue in many organizations since it has the possibility to result in large expenditures for recruiting and training new employees and also with the added cost due to reduced productivity and having a high turnover rate of employees contributes to negative workplace morale. (Das et al., 2022). Prediction systems can help in identifying trends and correlations in selected employee factors that contribute to resignation by utilizing methods of data exploration and visualization (Jain et al., 2019). Most of these prediction systems use Machine learning (ML) to learn and comprehend patterns in historical data to forecast future outcomes (Alsheref, Fattoh, and Ead, 2022). Ensemble models increase accuracy and performance by merging the results forecasted by several models (Alsheref, Fattoh and Ead, 2022). Employee attrition is a big concern for many firms, and advanced predictive technologies can give vital information to help reduce this issue as they may detect patterns and forecast future attrition concerns, allowing businesses to address possible difficulties and enhance staff retention.

1.3.1 Problem Statement

The need for a high-accuracy scalable employee attrition prediction system is evident in order to help companies and organizations in dealing with the implications of talent loss and optimizing employee retention strategies.

1.4 RESEARCH MOTIVATION

The research was motivated by the significant impact of employee churn on companies and the economy as a whole. Having witnessed the increasing challenge of conserving skilled workers in the midst of a growing brain drain and workforce emigration, the author is motivated to contribute to a solution that enables businesses to prevent attrition and cultivate a stable, skilled workforce.

1.5 EXISTING WORK

Citation	Brief Description	Limitations/ Future Work	Contribution
Employee Attrition Prediction			
(Raza, et al., 2022)	This study used 4 advanced ML algorithms: Extra Trees Classifier (ETC), Logistic Regression (LR), Support Vector Machine (SVM), Decision Tree Classifier (DTC). It also used Synthetic Minority Oversampling Technique (SMOTE) to balance the dataset.	The study did not use DL models and ensemble techniques which has the potential to improve the models' accuracy. Also, as this study used an oversampling method to balance data, it could cause overfitting which might not translate well to unseen data.	The study conducted a comparison of the four ML models used based on accuracy score value in order to determine the optimal performance fit evaluation approach. It also adjusted the suggested ETC approach to attain the maximum accuracy ratings when compared

			to other ML techniques and research.
(Gamage, 2022)	Utilizes retention, strategic planning, and ensemble methods using XGBoost Classifier, LightGBM Classifier, K-Nearest Neighbor, Logistic Regression, etc.	The author states that deep learning models have shown promising results in past studies and could be researched as a potential future contribution to the research domain.	Provides a framework for future research on employee attrition prediction. By leveraging various ML algorithms, the system contributed to the development of more accurate and reliable prediction models.
(Chakraborty et al., 2021)	Logistic Regression, Linear Discriminant Analysis, Ridge Classification, Lasso Classification, Decision Tree, Random Forest	Does not assess the influence of the variables across genders, educational backgrounds, skill, and performance levels. An in-depth analysis can be conducted whose results can be used to generalize India's sales industry. "In future, some **deep learning** algorithm will be used for getting better performance."	Random Forest showed the best accuracy out of all the techniques used. Naïve Bayes showed the least accuracy.
Ensemble Technique			
(Alsheref, Fattoh and Ead, 2022)	Random Forest, Gradient Boosting, and an Ensemble Model were applied.	The authors noted that no single model was perfect for every business setting, implying DL algorithms in an ensemble configuration may improve performance.	The models obtained high accuracy. However, the study did not investigate into the application of deep learning methods.
(Jain et al., 2019)	Utilized ML techniques Decision Trees (DT) and Random Forests (RF) to predict attrition. A	The researchers noted that the models might have a bias towards the majority class. This could provide a	The models demonstrated generally high accuracy with RF outperforming DT.

	univariate and bivariate data analysis was done alongside data visualizations to understand correlations between variables.	potential aspect for improvement using DL methods and Ensemble techniques that might handle class imbalance more effectively.	
(Muslim and Dasril, 2021)	Stacking ensemble model consisting of a basic model of KNN, decision tree, SVM, Random Forest, meta-learner LightGBM	While XGBoost-based feature significance is used for feature selection, the study does not investigate alternative feature selection strategies (filter-based or hybrid strategies that mix filter and embedding methods, etc.)	The stacking model achieved an accuracy rate of 97%, surpassing the individual basic models, including K-nearest neighbor, Decision tree, SVM, and Random Forest.

Table 1.1: Analysis of Existing Work (Self-Composed)

1.6 RESEARCH GAP

While researchers have indicated that Deep Learning (DL) models hold the potential to surpass the performance of traditional Machine Learning (ML) models in the domain of employee attrition prediction (Chakraborty et al., 2021), similar assertions have been made regarding ensemble methods incorporating DL models (Gamage, 2023). This presents an exciting and challenging problem area with significant implications for the field of employee attrition prediction. Current employee attrition prediction systems predominantly rely on the established paradigm of Machine Learning (ML) models and ensemble methods grounded in conventional ML algorithms. However, the utilization of Deep Learning (DL) models and the exploration of ensemble techniques within the context of employee attrition prediction represent uncharted territories with the potential to deliver groundbreaking results.

Conventional machine learning algorithms are suggested as having hindered performance when it comes to high-dimensional data. (Janiesch, C., Zschech, P., & Heinrich, K., 2021). Furthermore, the performance of ML algorithms is highly dependent on feature engineering and could overlook complicated and intricate patterns in the dataset. (Janiesch, C., Zschech, P., & Heinrich, K., 2021). Although earlier studies have looked at employee attrition prediction, there is still a gap in making thorough and reliable forecasts that consider the complexities and intricacies involved with high-dimensional data. Bridging this research gap is critical for providing businesses with improved tools that give accurate and practical predictions for effective human resource management.

1.7 CONTRIBUTION TO THE BODY OF KNOWLEDGE

Although researchers have suggested DL models to be more suitable and capable of outperforming ML models in this domain (Chakraborty et al., 2021) and the same has been said for ensemble methods using DL models (Gamage, 2023), current employee attrition prediction systems focus on machine learning models and ensemble methods that utilize machine learning algorithms. The use of Deep Learning models and methods of implementing an ensemble technique have yet to be explored with regard to the domain of employee attrition prediction, even though it has been suggested that they could outperform most existing systems.

Ensemble methods combining ML and DL algorithms or only DL algorithms have not been investigated thoroughly. The author aims to investigate into Ensemble Methods that use DL Models to gauge the validity of these claims and to assess whether the employee attrition prediction domain could benefit from utilizing Deep Learning algorithms in the Ensemble Technique.

1.7.1 Contribution to Problem Domain

Application of Deep Learning Ensemble Methods in Employee Attrition Prediction

This research contributes significantly to the problem domain of employee attrition. It introduces the innovative application of Deep Learning (DL) ensemble methods, which have the potential to substantially enhance the accuracy of attrition prediction. Conventional ML models have faced challenges, especially when dealing with high-dimensional data (Janiesch et al., 2021). By adopting DL models and exploring ensemble techniques, this study addresses the shortcomings of existing models and harnesses the power of DL for more precise predictions. In the context of predicting employee turnover, the application of Deep Learning ensemble techniques is a novel approach. Previous research predominantly relied on conventional machine learning models, and the introduction of Deep Learning methods represents a significant knowledge addition.

1.7.2 Contribution to Research Domain

Advancing the Role of Deep Learning in Ensemble Techniques

The proposed solution seeks to make significant contributions to the technology and domain elements of the employee attrition prediction problem by presenting an extensive ensemble of deep learning algorithms. A reliable and accurate prediction framework may be designed by combining cutting-edge approaches such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and other potential deep learning architectures. This technical improvement is intended to improve prediction capability and provide fresh insights. Within the research domain, this study contributes by advancing the understanding of Deep Learning's potential within ensemble techniques. By exploring the combination of Deep Learning and ensemble methods, this research attempts to offer insights into how these advanced techniques can collaboratively improve prediction. The study delves into hyperparameter tuning, feature engineering, and other strategies to enhance model precision. This broader understanding of Deep Learning's role in ensembles not only benefits attrition prediction but also extends its applicability to diverse ML challenges.

1.8 RESEARCH CHALLENGE

The integration of Deep Learning Ensemble techniques inside the Human Resources (HR) sector is the fundamental challenge in bridging the research gap and delivering the specified contribution. While Deep Learning has proven to be effective in a variety of sectors, its application in HR, notably for attrition prediction, has received little attention. Most contemporary HR systems rely on traditional machine learning models (Janiesch et al., 2021), and moving to Deep Learning ensembles necessitates a paradigm shift. Dealing with the complexities of HR data is another important difficulty. Deep Learning models perform well with high-dimensional data (Chakraborty et al., 2021), but collecting significant features and organizing the data for optimal model performance is difficult. The difficulty here is to navigate through feature engineering approaches and data pretreatment procedures to fully realize Deep Learning's potential for forecasting attrition.

1.8.1 Research Questions

The following are the research questions that will lead this study:

RQ1: How can Deep Learning ensemble methods be effectively integrated into the field of Human Resources (HR) for accurate employee attrition prediction?
RQ2: What holistic methods, feature engineering, and data preprocessing techniques can optimize the performance of Deep Learning ensemble models for employee attrition prediction in the HR context?

1.9 RESEARCH AIM

The primary aim of this study is to design, develop, and evaluate an advanced ensemble prediction model that incorporates several deep learning algorithms to effectively predict employee turnover in enterprises.

This model aims to improve the accuracy of prediction and interpretability of employee attrition predictions by leveraging the strengths of both traditional machine learning algorithms such as XGBoost, Random Forest, and Logistic Regression, as well as deep learning techniques such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). The study hopes to contribute to the current body of knowledge by presenting a holistic solution for tackling the complicated domain of employee attrition, allowing companies to effectively utilize strategies for retention and enhance workforce management procedures.
Traditional ML algorithms such as XGBoost, Random Forest, and Logistic Regression have demonstrated their effectiveness in various predictive tasks and are known for their interpretability, making them invaluable in providing actionable insights to organizations. On the other hand, deep learning techniques, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have shown remarkable promise in handling complex data patterns and sequences, offering the potential for more accurate predictions in the realm of employee attrition.

1.10 RESEARCH OBJECTIVES

Research Objectives	Description	Learning Outcome	Research Question

Problem Definition	To identify the contributing factors behind employee attrition RO1: To determine the factors that affect employee turnover. RO2: To verify previous limitations in employee attrition and DL and ensemble domain research in order to find and address gaps.	LO1, LO3, LO4	RQ1
Literature Review	To examine current literature for relevant insights and critically evaluate earlier research on the topic. RO3: To identify trends, gaps, and emerging topics in the domain of employee attrition prediction and synthesize data from earlier studies. RO4: To assess the performance of various techniques and approaches used in prior research initiatives, with a particular emphasis on their relevance to the proposed attrition prediction model.	LO1, LO4, LO8	RQ2
Requirement Elicitation	RO6: To collect and analyze relevant factors influencing employee attrition, including historical employee data, workplace conditions, and external variables. RO7: To explore existing literature and models in the field of attrition prediction to identify key features, variables, and methodologies that have proven effective.	LO2, LO3	RQ1, RQ2
Research Design	To design an architecture and a framework capable of addressing the concerns indicated in the solution. RO8: To design the high-level architecture for the proposed solution involving the DL and Ensemble methods. RO9: To develop an early prototype and prediction model to better assess the proposed solution.	LO1, LO2, LO5, LO8	RQ2
Implementation	To develop a system capable of tackling the issues mentioned. RO10: To apply the DL models to more sample data and select the best technique.	LO1, LO2, LO5, LO7	RQ2

Testing - Quantitative	To rigorously evaluate the developed system and ML models using quantitative methods and appropriate datasets to measure performance. RO11: To conduct model assessment, performance testing, and integration execution for the ensemble model.	LO1, LO5, LO6, LO8	RQ2
Evaluation - Qualitative	To assess the effectiveness of the solution and gather qualitative insights from relevant stakeholders and experts. RO12: To engage in qualitative evaluation methods to collect feedback and insights from users and experts in the attrition domain, and technological domain and analyze the data obtained in order to improve on them and enhance its usability and effectiveness.	LO1, LO5, LO6, LO8	RQ3
Documentation	RO13: To keep a detailed research record that includes methodology, findings about DL and Ensemble technique, and obstacles, track ethical, legal, or professional difficulties, and how they were resolved, and write a thorough document that summarizes skills, plans for the project, and critical reviews of the final ensemble model.	LO6, LO7, LO8	RQ1, RQ2

Table 1.2: Research Objectives (Self-Composed)

1.11 FEATURES OF THE PROTOTYPE

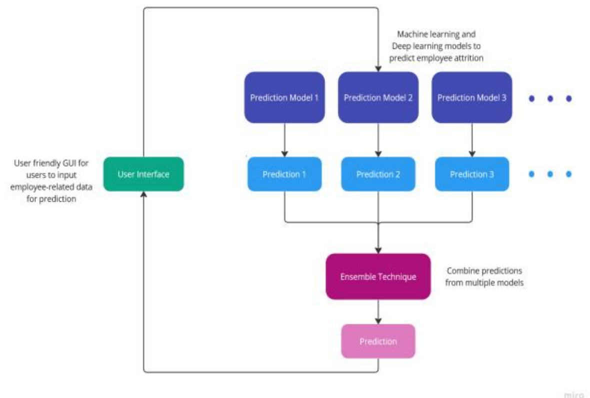


Figure 1.5 Prototype Feature Diagram (Self-Composed)

CHAPTER 2: LITERATURE REVIEW

2.1 CHAPTER OVERVIEW

This chapter will cover the problem domain, go over existing work that has already been conducted in the domain and critically evaluate and review the technological aspects pertaining to all stages of the project.

2.2 CONCEPT GRAPH

The Concept Graph showing all the project stages, and the technologies and techniques involved is given in **Appendix A: Concept Graph**.

2.3 PROBLEM DOMAIN

2.3.1 Employee Attrition

Employee attrition, often referred to as employee turnover, is a significant concern for many organizations. It represents the voluntary or involuntary reduction of workforce and can result in significant expenditures for organizations, such as recruiting, training, and productivity reductions. Employee turnover can also have an influence on team morale and result in increased turnover. Predicting employee attrition is thus an important issue for human resource departments and management (Bhartiya et al., 2019).

Employee attrition may be caused by a number of variables, including job satisfaction, salary, work-life balance, and role compatibility. Understanding these parameters and their connections to attrition is critical for constructing successful attrition prediction algorithms (Raza et al., 2022).

2.3.2 Machine Learning & Deep Learning in Attrition Prediction

Due to its capacity to handle large amounts of data and complicated relations between variables, machine learning has been widely employed for attrition prediction. ML models learn from prior data to build a model that can predict future outcomes reliably. These algorithms may be used to discover trends in employee behavior and conditions that suggest a possibility of workers leaving the organization (Bhartiya et al., 2019).

Deep learning, a subset of machine learning (ML), has gained favor in recent years for attrition prediction. DL models, especially neural networks, can learn and extract characteristics from raw data automatically. This trait makes them particularly effective for attrition prediction, since they may reveal deep patterns and linkages in data that typical ML algorithms may not see or access (Pokkuluri and Devi, 2023).

2.3.3 Ensemble Techniques in Attrition Prediction

To boost prediction accuracy, ensemble approaches mix numerous ML models. These strategies capitalize on individual models' strengths while mitigating their faults, making for a more resilient and accurate prediction model. (Pokkuluri and Devi, 2023).

Using ensemble approaches to predict attrition can enhance prediction accuracy by lowering the chance of overfitting and increasing the generalizability of the model. As a result, ensemble approaches represent a viable option for improving the efficacy of attrition prediction models.

2.3.4 Deep Learning Algorithms and Ensemble Techniques in Attrition Prediction

The use of deep learning algorithms to forecast staff turnover is gaining popularity. Deep learning models can understand complicated patterns and detect non-linear correlations in data, making them excellent for forecasting staff turnover (Bhartiya et al., 2019).

Ensemble approaches, which integrate numerous models to get a final forecast, have also been used to enhance attrition prediction accuracy. Ensemble approaches can produce more reliable and accurate predictions by utilizing the strengths of diverse models and adjusting for their deficiencies (Pokkuluri and Devi, 2023).

2.3.5 Performance Evaluation of Attrition Prediction Models

2.4 EXISTING WORK

Employee attrition prediction is a significant concern for many organizations. With the rise of machine learning techniques, researchers have begun to explore its potential in predicting employee turnover (Srivastava and Nair, 2018)

2.4.1 Traditional Machine Learning Techniques in Employee Attrition Prediction

To forecast employee attrition, traditional machine learning algorithms have been frequently deployed. Among these approaches are logistic regression, decision trees, support vector machines, and random forest (Alduayj & Rajpoot, 2018).

Logistic Regression (LR) is a statistical approach for modeling the likelihood of a certain occurrence (in this example, an employee leaving the firm) by fitting data to a logistic curve (Najafi-Zangeneh et al., 2021). Ponnuru et al. found logistic regression to be a reliable strategy for predicting employee attrition (Najafi-Zangeneh et al., 2021).

Another frequent approach used in attrition prediction is decision trees (DT). They function by dividing data into branches depending on certain requirements, allowing for easier visualization and comprehension of the model (Jain, Jain, and Pamula, 2020). Mohbey employed decision trees to predict employee attrition in a research study. (Najafi-Zangeneh et al., 2021).

Support Vector Machines (SVM) are also used to forecast attrition. SVMs determine the hyperplane that best separates various classes in a dataset, making them an effective tool for binary classification issues (Raza et al., 2022). Multiple research projects have compared SVM to other machine learning models, demonstrating that it is a reliable tool for forecasting employee turnover (Raza et al., 2022).

Random Forest (RF), an ensemble learning approach that builds several decision trees and combines their outputs, has also been used to predict employee turnover (Jain, Jain, and Pamula, 2020).

These standard machine learning algorithms, despite their success, have limits, particularly when working with complicated and non-linear data patterns (Jain, Jain, and Pamula, 2020). Logistic regression, for example, implies a linear relationship between the input variables and the log chances of the output, which may not be true in many real-world cases (Najafi-Zangeneh et al., 2021). Decision trees have a high propensity to overfit or underfit data, resulting in poor generalization performance (Jain, Jain, and Pamula, 2020). Large datasets and multi-class classification issues might be difficult for support vector machines (Raza et al., 2022). While random forests are frequently more accurate than individual decision trees, they can be computationally costly and difficult to comprehend (Jain, Jain, and Pamula, 2020).

Traditional machine learning algorithms have been widely used and have shown promise in forecasting employee loss, but they are not without limitations. Future study should focus on other strategies, resolving their shortcomings and improving their effectiveness (Alduayj, and Rajpoot, 2018).

2.4.2 Ensemble Learning Techniques in Employee Attrition Prediction

Ensemble learning algorithms have demonstrated potential in forecasting employee turnover. Multiple models are combined in these strategies to increase forecast accuracy. 2022) (Shamim, Khan, and Javed). They have been used to improve individual model performance by minimizing bias and variation, as well as to mix several types of models, possibly harnessing the benefits of each. 2022) (Shamim, Khan, and Javed).

Alsheref, Fattoh, and Ead, 2022 developed an ensemble model based on probabilities from many models. The model employed majority voting, which means that each model predicts for all test cases, and the final output prediction is the one with the most votes.

Furthermore, the Stacked Ensemble Method (Gowdru et al., 2023), a technique in which the predictions of numerous classifiers are utilized as input to another model (commonly referred to as the meta-classifier), has been used to forecast employee attrition. Based on the predictions of the different models, the meta-classifier is trained to generate the final prediction (Gowdru et al., 2023). This technique frequently outperforms individual models (Gowdru et al., 2023).

Ensemble learning approaches, despite their usefulness, have limits. They frequently necessitate significant processing resources and are less interpretable than individual models. (Shamim, Khan, and Javed, 2022). Future research should continue to explore these and other ensemble techniques, addressing their limitations, and enhancing their performance (Gowdru, et al., 2023).

2.4.3 Deep Learning Techniques in Employee Attrition Prediction

Due to their capacity to model complicated, non-linear connections in data, deep learning approaches are gaining favor in the field of employee attrition prediction. (Gurler, Pak, and Gungor, 2023). Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformer-based networks are examples of approaches that can learn numerous layers of representations, each of which transforms the input data into a more abstract and composite representation. Gurler, Pak, and Gungor (Gurler, Pak, and Gungor, 2023).

A Transformer-based neural network was used to predict employee attrition in a study by Wenhui Li. (Li W., 2023). As a computational approach for detecting staff turnover, the Transformer-based network was distinguished by contextual embeddings adjusting to tubular data. (Li W., 2023). In terms of prediction efficiency, the experimental findings showed that this model outperformed other state-of-the-art models. This study also discovered that deep learning in general, and Transformer-based networks in particular, are promising for dealing with tabular and unbalanced data. (Li W., 2023).

Another study looked into the usage of Recurrent Neural Networks (RNNs) to predict employee turnover. (Raza et al., 2022). RNNs, especially those with Long Short-Term Memory (LSTM) units, are well suited to sequential data because they can capture temporal dependencies. (Raza et al., 2022). The use of RNNs in predicting employee attrition shows the promise of deep learning techniques in this arena. (Raza et al., 2022).

Deep learning algorithms, despite their promising outcomes, present a number of obstacles. They require massive amounts of data as well as computing resources. (Gurler, Pak, and Gungor, 2023). Furthermore, deep learning models are also referred to as black boxes due to their intricate, non-linear structure. (Raza et al., 2022).

2.4.4 Conclusion

Traditional machine learning approaches, ensemble learning techniques, and deep learning techniques have all been used to predict employee attrition. Each of these techniques has its own set of strengths and limitations, and their use in predicting employee attrition has shown encouraging results.

Because of their simplicity and interpretability, traditional machine learning techniques such as logistic regression, decision trees, support vector machines, and random forest have been frequently employed. 2020 (Jain, Jain, and Pamula). However, they frequently struggle with complicated and non-linear data patterns. (Gurler, Pak, and Gungor, 2023).

Ensemble learning approaches, which integrate many models to increase prediction accuracy, have been found to minimize bias and variance, hence improving individual model performance. (Gurler, Pak, and Gungor, 2023). However, they frequently need significant processing resources and are less interpretable than individual models. (Ganthi, et al., 2022).

Deep learning approaches like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformer-based networks may learn numerous layers of representations, each of which transforms the input data into a more abstract and composite representation. (Fallucchi, et al., 2020). Deep learning methods, despite their promising outcomes, face various problems, including the need for enormous amounts of data and computer resources, as well as their complicated, non-linear character, which makes them difficult to comprehend. (Fallucchi, et al., 2020).

While these approaches have made significant progress in forecasting staff loss, there are still numerous hurdles to overcome. Future research should focus on exploring novel machine learning, deep learning, and ensemble techniques, addressing their limitations, and enhancing their performance.

2.5 TECHNOLOGICAL REVIEW

The literature Survey results and details can be found in **Appendix B: Literature Survey**. The in-depth breakdown and analysis of the technologies and methods used in existing work can be found in **Appendix C: Technological Review** under the different stages and phases of the project.

2.5.1 Dataset Selection

The selection of datasets is an important part of developing and executing prediction models for employee turnover. Several studies in literature have used several datasets, each with its own set of features. The IBM HR Analytics Employee Attrition & Performance dataset is a popular choice because of its extensive range of 35 attributes, which includes basic HR factors such as age, education, and gender. This dataset contains a wealth of information that may be used to investigate issues associated with employee attrition. However, one major shortcoming is that it may not capture industry-specific subtleties due to its uniformity. Another often used dataset is the IBM HR Employee Attrition dataset, which was generated by IBM data scientists and comprises 1470 records and has been used to investigate the factors of employee attrition. The benefit of this dataset is that it was created by industry specialists, providing applicability to real-world applications; nonetheless, its memory utilization may pose limits in some computing systems.

The unbalanced nature of datasets, as observed by Karande and Shyamala (2019), offers issues for model training and generalizability. Datasets containing designed knowledge, such as the one from IBM Watson Analytics (Chakraborty et al., 2021), on the other hand, give selected features but may lose granularity in the process.

It's important to note the variety of dataset sources, which range from private datasets developed by companies like IBM to open-source datasets available on platforms like Kaggle. While commercial datasets provide specificity and dependability, open-source datasets benefit the larger scientific community but may lack key unique insights. Furthermore, statistics from real-time reporting (Wu, 2022) add a temporal dimension, boosting comprehension of changing trends in employee attrition.

The selection of datasets for predicting employee attrition entails compromises between specificity, generalizability, and technical factors. The range of dataset formats and sources among the examined papers illustrates the dynamic nature of research in this domain, stressing the importance of carefully considering dataset features depending on each study's individual aims and context.

2.5.2 Data Preprocessing

Data cleaning, which includes operations such as managing duplicate entries and correcting errors, is an important first step in guaranteeing the dataset's dependability (Jain, 2017). Data scaling and other feature scaling strategies help to homogenize numerical features and reduce dominance difficulties during training of models (Alshiddy, M.S., & Aljaber, B., 2023; Qutub, A., et al., 2021). The difficulties of categorical representation of data are addressed by feature encoding and engineering to ensure that the models are able to successfully interpret non-numerical features (Raza et al., 2022; Chakraborty et al., 2021; Qutub, A., et al., 2021).

Techniques for finding and preserving the most significant characteristics, such as information gain-based feature selection and eliminating redundant variables, help to model efficiency (Alshiddy, M.S., & Aljaber, B., 2023; Jain, D. 2017). Approaches such as correlation analysis ensure solid management of missing data with minimum information loss (Alshiddy, M.S., & Aljaber, B., 2023; Alsheref, Fattoh, and Ead, 2022). While normalization of data contributes to uniform numerical characteristics, it comes with a risk of possible noise amplification (Alshiddy, M.S., & Aljaber, B., 2023; Qutub, A., et al., 2021).

2.5.3 Feature Selection and Engineering

Mutual Information and Information Gain-Based techniques, for example, function irrespective of specific models, analyzing the significance of characteristics to the goal variable. These computationally efficient approaches, used by Raza et al. (2022) and Qutub et al. (2021), may ignore subtle inter-feature interactions. Wrapper approaches, such as those investigated by Chakraborty et al. (2021) and Alsheref, Fattoh, and Ead (2022), adjust feature selection to the complexities of the algorithms, but at a greater cost of computation and with the risk of overfitting.

Embedded approaches, such include L1 Regularization (Lasso) and Recursive Feature Elimination (RFE), incorporate feature selection into the model-building workflow in a smooth manner.

However, as Chakraborty et al. (2021) indicate in the literature, these strategies can be computationally demanding and may not uniformly perform across all models.

Random Forest Feature Importance, an ensemble-based technique, has gained popularity because of its reliability, ability to handle non-linearity, and provision of useful feature importance scores (Chakraborty et al., 2021; Alshiddy & Aljaber, 2023; Qutub et al., 2021). Another approach, Principal Component Analysis (PCA) (Alsheref, Fattoh, and Ead, 2022; Qutub et al., 2021), proves its effectiveness in dimensionality reduction, albeit its application particular are challenging.

As illustrated by Srivastava and Eachempati (2021), feature selection supplemented by expert knowledge provides a subjective aspect but can influence the selection process based on domain knowledge. Raza et al. (2022) and Jain (2017), both use data correlation analysis and redundancy reduction to minimize dimensionality by deleting duplicate features and dealing with multicollinearity. Kim et al. (2022) proposes that using time-series information is especially useful for jobs requiring temporal characteristics.

Feature engineering comprises a number of approaches, such as quantitative value conversion, one-hot encoding and data scaling, each of which serves a distinct role in preparing data for modeling. Artelt and Gregoriades (2023) focus on choosing and creating factors known to effect employee attrition, whereas Alshiddy and Aljaber (2023) stress the need of balancing skewed datasets. The use of label encoding and satisfaction calculation by Sharma et al. (2022) and Wu (2022), demonstrates the variety of feature engineering techniques, each with its own set of benefits and problems. However, it is critical to recognize that the efficacy of these strategies varies depending on the properties of the data and the ML model being trained.

2.5.4 Algorithm Selection Process

Algorithm selection is an important part of developing good predictive models, and academics have used a variety of ways to pick the best algorithm for certain tasks such as employee attrition prediction. Because of its simplicity and interpretability, LR is generally selected when the relationship between variables and the output is linear (Al-Darraji et al., 2021). Support Vector Machines (SVM) excel at processing high-dimensional and non-linear data, but finding proper kernels and hyperparameters is difficult (Raza et al., 2022). Random Forest, a robust ensemble approach, works well with high-dimensional data, although deep trees can cause overfitting (Chakraborty et al., 2021; Alsheref, Fattoh, & Ead, 2022).

Decision trees are interpretable and perform well with categorical data, but they are susceptible to overfitting (Chakraborty et al., 2021). K-Nearest Neighbors (KNN) is simple yet sensitive to neighbor selection, and it can be operationally costly for big datasets (Qutub et al., 2021). Gradient Boosting Classifier, an ensemble approach, enhances accuracy but is computationally costly and susceptible to overfitting (Raza et al., 2022). Latent Dirichlet Allocation, intended for text data, finds hidden subjects efficiently but is limited to text-based applications (Gupta et al., 2021).

Extra Tree Classifier is resistant to overfitting and has a low computational cost, although it may not be as reliable as other approaches (Alshiddy & Aljaber, 2023). Gradient Boosting (XGBoost) is well-known for its great predictive accuracy and efficiency, but it necessitates careful hyperparameter adjustment (Alsheref, Fattoh, & Ead, 2022). Artificial Neural Networks (ANN) are good for complicated non-linear interactions, but they require a lot of data and computer power (Alsheref, Fattoh, & Ead, 2022). Recurrent Neural Networks (RNNs) are useful for sequential and time series data, although deep RNNs may encounter vanishing/exploding gradient difficulties (Kim et al., 2022).

Genetic Algorithms (GA) provide excellent optimization for ensemble weighting and feature selection, but they can be computationally demanding and need parameter fine-tuning (Kim et al., 2022). Deep learning technique Bi-LSTM is good for sequential data but requires a large amount of data and resources (Al-Darraji et al., 2021). The Voting Classifier (VC) Ensemble Method integrates many models to enhance accuracy, but it adds computing complexity and necessitates hyperparameter adjustment (Chung et al., 2023). The C4.5 Classifier is an interpretable decision tree model that is effective but prone to overfitting (Chakraborty et al., 2021). CAT Boost, which is intended for categorical features, is efficient but memory-intensive and requires hyperparameter adjustment (Sharma et al., 2022). CART, a decision tree model, is interpretable and can handle categorical and numerical data, although it can overfit with deep trees (Chakraborty et al., 2021).

The unique properties of the dataset, the requirements, and the compromise between interpretability and accuracy in prediction should all be considered while choosing an algorithm.

2.5.5 Algorithm Training Process

Common approaches for algorithm training include the Train-Test Split method, in which the dataset is separated into training and testing sets, provides a rapid performance estimate but potentially lacks robustness due to a single random split (Sharma et al., 2022; Wu, 2022). A more robust option, cross-validation, includes numerous splits to better capture data variability and offer a more trustworthy assessment of model performance (Qutub et al., 2021; Chung et al., 2023).

Ensemble learning approaches, which combine predictions from many models to improve performance as a whole, have grown in popularity. Autotuning approaches improve the model's capacity to capture complicated interactions by optimizing hyperparameters (Alsheref, Fattoh, and Ead, 2022). This approach is refined further by ensemble weight optimization, which dynamically adjusts weights for each model in the ensemble to reach optimal outcomes (Kim et al., 2022). While these strategies improve predictive skills, they add computing complexity and may need careful tweaking.

Some research has taken novel techniques, such as adding genetic algorithms into the train-test-split procedure, which allows for real-time prediction updates and ongoing model development based on developing datasets (Wu, 2022). Others investigated the advantages of supervised learning using several classifiers, combining the capabilities of various methods to develop more reliable prediction models (Qutub et al., 2021; Karande and Shyamala, 2019).

Each approach has its own set of advantages and disadvantages. Although the Train-Test Split and Cross-validation procedures are simple, they may lack robustness. Ensemble learning and autotuning improve prediction power while increasing processing requirements. The use of genetic algorithms for real-time updates adds agility but necessitates a large amount of data. Supervised learning with many classifiers yields reliable predictions but needs careful tweaking. The algorithm training technique used is determined by criteria such as dataset qualities, processing resources, and the ideal level of model robustness.

2.5.6 Ensemble Technique

Due to its capacity to improve prediction performance by mixing many models, ensemble approaches have received a lot of interest in the domain of machine learning. To address issues in forecasting employee turnover, researchers have investigated several ensemble approaches such as Stacking and Voting. Alshiddy and Aljaber's (2023) stacking and voting ensemble approaches combine predictions from various base models to generate a more reliable final prediction. The intrinsic complexity of ensemble approaches, on the other hand, may result in increasing processing needs.

A study published by Alsheref, Fattoh, and Ead, 2022 explores the use of ensemble models to predict employee turnover. Overall, the study found that the ensemble model was effective in identifying employee factors and improving the model's predictive accuracy. In particular, majority voting models was highlighted due to their simplicity and efficiency in combining predictions from multiple models to obtain more robust predictions. The study shows that the ensemble model achieves approximately **85%** accuracy on test data, which is more than the accuracy achieved by the other models. (Alsheref, Fattoh, and Ead, 2022).

In a study by Chung et al., 2023, IBM HR data was used to build a stacking-based ensemble model, and this was compared to several other models. The researchers concluded that the ensemble model showed the "highest performance". (Chung et al., 2023).

Model	Accuracy	Precision	Recall	F-score	AUC
SVM	0.912	0.967	0.853	0.906	0.912
LR	0.921	0.952	0.889	0.920	0.921
ANN	0.923	0.936	0.908	0.922	0.923
XGB	0.925	0.968	0.879	0.921	0.925
RF	0.947	0.980	0.914	0.946	0.947
ESM3	0.952	0.966	0.934	0.950	0.951
ESM2	0.968	0.998	0.934	0.966	0.967
ESM1	**0.976**	**0.998**	**0.951**	**0.975**	**0.975**

Figure 2.1: Test Results from the Study (Chung et al., 2023)

2.5.7 Hyperparameter Tuning

Researchers have used several strategies to improve hyperparameters in the domain of predicting employee attrition, which is a vital step in improving the effectiveness of the machine learning models. Raza et al. (2022) used randomized grid search, which includes methodically sampling hyperparameter configurations, allowing for fast analysis of the hyperparameter space. This strategy strikes a compromise between resource utilization and performance enhancement, while its effectiveness is strongly dependent on the search space selected.

While these approaches provide promising results, the process of hyperparameter tuning is critical, and researchers have used techniques such as randomized grid search, and autotuning to maximize model performance. Notably, as emphasized in research by Raza et al. (2022), Chakraborty et al. (2021), and Alsheref, Fattoh, and Ead (2022), the choice of hyperparameter tuning strategy is dependent on the unique aims and features of the datasets.

Notably, several research, such as Alsheref, Fattoh, and Ead (2022), used autotuning approaches, which automate the process of determining optimal hyperparameter combinations. While autotuning is convenient, it lacks transparency and interpretability when compared to hand tuning. In addition, grid search approaches, such as the one utilized by Qadir, Noreen, and Shah (2021), systematically examine hyperparameter configurations but can be operationally costly, particularly in high-dimensional areas.

2.5.8 Evaluation Metrics

Metrics such as accuracy, precision, recall, and F1 score are frequently used in various research, providing a comprehensive assessment of accuracy, precision, and the compromise between precision and memory. For example, Raza et al. (2022) used these measures to assess the performance of their Extra Trees Classifier (ETC) model, which achieved an impressive 93% accuracy on unknown data. Gupta et al. (2021) emphasized the significance of accuracy, F1 score, precision, and recall in comparing and selecting the best model for forecasting employee attrition.

To completely examine model performance, Alshiddy and Aljaber (2023) used a larger set of indicators, including area under the curve (AUC). Similarly, Alsheref, Fattoh, and Ead (2022) evaluated models using the misclassification rate, accuracy, and F1 score, offering a more comprehensive view of model performance. This variation in measurements assists in capturing various elements of model behavior. The difficulty resides, however, in the interpretability of these measures, particularly for non-experts, as well as the possible bias induced by picking metrics based on specific issue features.

While some research, such as Qutub et al. (2021) and Yahia, Hlel, and Colomo-Palacios (2021), highlight the relevance of AUC-ROC for unbalanced datasets, others, such as Srivastava and Eachempati (2021), emphasize metrics such as RMSE and R-squared to evaluate model fit.

It is critical to evaluate the performance of the prediction algorithms in order to verify their efficacy and dependability. Various parameters, including accuracy, precision, recall, and the area under the Receiver Operating Characteristic (ROC) curve, are used in this process. The ratio of correct projections made by the model is referred to as accuracy, the fraction of genuine positive forecasts is referred to as precision, and the proportion of true positive predictions is referred to as recall. The ROC curve is a graphical depiction of the true positive rate of the model vs the false positive rate (Raza et al., 2022).

In addition to these indicators, confusion matrix analysis may be utilized to evaluate the attrition prediction model's performance. A confusion matrix is a table that illustrates how many right and wrong predictions the model produced for each class. These performance evaluation tools aid in identifying the model's strengths and limitations, directing future developments.

2.5.9 Benchmarking

Benchmarking is a crucial step in determining the effectiveness of algorithms for forecasting employee attrition. In the literature, many benchmarking approaches have been explored, each giving unique insights on model performance. Raza et al. (2022) achieved 93% accuracy while comparing their ETC approach to current models such as Decision Tree, Logistic Regression, Random Forest, and Support Vector Machines. Using a baseline model, Chakraborty et al. (2021) discovered that Random Forest outperformed others in terms of accuracy measurements. Benchmarking was utilized by Gupta et al. (2021) to assess the accuracy and F1 score of Logistic Regression, KNN, and Random Forest models. Alshiddy and Aljaber (2023) compared their proposed model to Naive Bayes, SVM, and Random Forest using AUC as the key benchmarking measure. Alsheref, Fattoh, and Ead (2022) conducted extensive benchmarking using several algorithms, measuring accuracy, F1 score, cumulative lift, and Gini coefficient.

Ensemble models outperformed Random Forest, Gradient Boosting, and an ensemble approach, according to Srivastava and Eachempati (2021). Jain (2017) investigated many approaches and determined that adaptive boosting was the most effective in terms of benchmarking accuracy. Kim et al. (2022) evaluated models using RMSE and Clinical Accuracy Grid-Error Grid Analysis, providing specific metrics. In terms of accuracy, precision, recall, and F1-score, Al-Darraji et al. (2021) compared their methodology to state-of-the-art techniques. Qadir, Noreen, and Shah (2021) compared Bi-LSTM against MLP and Nave Bayes, confirming its superiority across numerous parameters. Yahia, Hlel, and Colomo-Palacios (2021) used Kaggle and IBM HR Analytics datasets for external validation. Artelt and Gregoriades (2023) customized advice to several departments using logistic regression, random forest, and XGBoost.

Sharma et al. (2022) made no particular mention of benchmarking. Although Wu (2022) did not specify benchmarking methodologies, Mehta and Modi (2021) examined Random Forest and Gradient Boosting with a focus on accuracy. Chung et al. (2023) used accuracy, precision, recall, F-score, and AUC to compare several models. According to Karande and Shyamala (2019), ensemble learning outperforms SVM, logistic regression, and random forest in terms of accuracy, precision, and specificity. Although the approaches and measurements used in these benchmarking methodologies differ, they all lead to a deeper knowledge of algorithm performance.

2.6 CHAPTER SUMMARY

In conclusion, the literature review chapter has analyzed the usage of standard machine learning, ensemble learning, and deep learning approaches in predicting employee attrition. Traditional machine learning approaches have been widely used because of their ease of use and interpretability, but they frequently struggle with complicated and non-linear data patterns. Ensemble learning approaches, which combine many models to improve prediction accuracy, have shown promise in decreasing bias and variation, but they frequently necessitate significant processing resources and can be less interpretable than individual models. (Ganthi, et al., 2022). Deep learning techniques such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformer-based networks have shown promise in learning multiple layers of representations, but they also present challenges such as the need for large amounts of data, computational resources, and their complex, non-linear nature, which makes them difficult to interpret. (Fallucchi, et al., 2020).

CHAPTER 3: METHODOLOGY

3.1 CHAPTER OVERVIEW

This chapter will cover the various methodology aspects chosen to conduct the research project. The author will establish the research methodology, development methodology, and project management methodology and identify all aspects of the project with regard to requirements and risks involved.

3.2 RESEARCH METHODOLOGY

Layer	Description	Justification
Research Philosophy	Due to the project's quantitative aspect, **positivism** was chosen as the research philosophy for this project. The emphasis of this research technique is on measurable data, statistical analysis, and empirical observations.	The project's goal is to create machine learning models that can anticipate certain outcomes by working with structured data and numerical measures. As a result, positivist philosophy is ideally suited to guiding research methods in this situation.
Research Approach	For this project, the **deductive** research technique was chosen because it matches the goal of leveraging current machine learning theories and algorithms to forecast employee turnover. Existing ideas and proven ML frameworks are used as a starting point in this method.	The study assesses the application of these theories and approaches to the specific problem of staff attrition prediction, with the goal of determining whether they can be used effectively for this purpose.
Methodological Choice	This research study employs the **mono-method** approach, concentrating solely on quantitative data gathering and analysis approaches. Statistical approaches and ML algorithms will be utilized to collect and evaluate numerical data associated with employee attrition.	The mono-method choice is appropriate for this project due to its alignment with the predominantly quantitative nature of the research. This approach is selected to maintain a consistent and structured methodology throughout the research process.
Research Strategy	A **survey**-based research technique was adopted for this project. This method	A survey-based strategy is ideally suited to the quantitative character of

	incorporates creating and deploying a structured questionnaire to obtain quantitative data from employees from various demographics and job responsibilities. The survey's goal is to determine their chance of attrition and collect appropriate information for study.	this research study and its purpose of forecasting staff attrition. This technique is consistent with the positivist viewpoint, and by using surveys, the project may collect data on a variety of issues that may impact attrition.
Time Horizon	The time horizon chosen for this research topic is **cross-sectional**, implying that data will be collected at a particular moment in time. It refers to obtaining data from workers and organizational records within a certain duration in this context.	The cross-sectional time horizon is suited for this research study since it provides a picture of the employee attrition situation at a certain point in time. This method is especially effective for examining current attrition levels, identifying probable contributing causes, and constructing forecast models based on existing data.

Table 3.1: Research Methodologies (Self-Composed)

3.3 DEVELOPMENT METHODOLOGY

3.3.1 Life Cycle Model

The project will be built iteratively and incrementally using an iterative and incremental Software Development Life Cycle (SDLC) approach. By building on prior iterations, this technique enables for continual refining and enhancement of the prediction system. It allows us to gather input, adjust to changes, and deal with any issues that occur during the implementation process. The SDLC's iterative approach ensures that the project progresses in accordance with the project's goals and needs.

3.3.2 Requirement Elicitation Methodology

This research project's demand elicitation technique includes a thorough examination of current literature and resources on staff attrition prediction. It entails identifying essential characteristics, circumstances, and signs related to employee attrition. Academic papers, reports, surveys, and related databases are the key data sources for requirement elicitation. The project seeks to lay a strong basis for identifying the requirements required for the creation of successful attrition prediction models by using existing knowledge and information gained from surveys.

3.3.3 Design Methodology

The project's design process will be based on the concepts of Structured systems analysis and design method (SSADM). SSADM was chosen for its potential to allow the structuring of software, its reusability, and extension. Because of the complexity of the project, a design solution that enables the smooth integration of several ML and deep learning algorithms is required. The emphasis on clarity and structure in SSADM will help to create a reliable and consistent prediction system. SSADM also places heavy emphasis on the data of the system and makes it simpler to comprehend the connections between data and processes of the system.

3.3.4 Programming Paradigm

The programming paradigm chosen will also reflect the design methodology in that the author uses SSADM for its potential to allow the structuring of software, its reusability, and extension. Smooth integration of several algorithms is required for this project, therefore making it an important point of consideration.

3.3.5 Evaluation Methodology

The developed prediction system will be evaluated through extensive testing and performance evaluation. To guarantee the accuracy and efficacy of the system's predictions, many assessment criteria will be used:

- **Accuracy**: The ratio of accurately predicted events over the total number of events.
- **Precision**: The proportion of actual positive forecasts to total projected positive cases, used to assess positive accuracy of predictions.
- **Recall (Sensitivity)**: The proportion of genuine positive predictions to total real positive occurrences, which measures the system's capacity to recognize all positive cases.
- **F1 Score**: A balanced assessment metric based on the harmonic mean of accuracy and recall, particularly useful for unbalanced datasets.
- **AUC-ROC (Area Under Curve)**: The area under the curve (AUC) of the Receiver Operating Characteristic (ROC) curve, which provides an overall assessment of the model's discriminating ability.

These evaluation criteria will help us to make educated decisions regarding the prediction model's efficacy by providing a full review of its performance. An iterative SDLC, OOAD principles, and strong assessment procedures will all help to ensure the successful creation of an accurate and dependable staff attrition prediction system.

3.4 SOLUTION METHODOLOGY

The solution will be created through a structured method that includes the following major stages:

3.4.1 Data Collection and Preparation

Both the amount of data and the quality of data are crucial factors in the accuracy of attrition prediction. High-quality data that correctly represents the workforce of the business is required for training successful ML, DL, and Ensemble Models. This entails gathering information on a wide range of employee characteristics and experiences, such as job satisfaction, remuneration, work-life balance, and job role fit (Raza et al., 2022).

However, data collecting has its own set of difficulties. For example, due to privacy considerations and the sensitivity of the topic, acquiring accurate and complete statistics on employee experiences and views might be challenging. Furthermore, the data may be skewed, with more records of workers who stayed than those who departed, which may bias the prediction model toward projecting that employees will not resign. (Alsheref, Fattoh, and Ead, 2022). In such cases, data needs to be preprocessed and engineered beforehand, boosting the accuracy of the model prediction and minimizing its complexity (Raza et al., 2022).

During this first step, detailed information on employees and their attrition statuses will be used. There are many datasets available that are open source. The author will have to check if they are suitable and if so, they can be used for the model training and testing stages. If they are not suitable, the author will have to create their own dataset by collecting real-world employee attrition data from organizations. This data will have to be subjected to extensive pre-preparation in order to handle missing values, and outliers, and assure consistency. Cleansing and transforming data will establish the foundation for accurate modeling.

3.4.2 Feature Engineering

Feature engineering is critical to the success of any predictive model. To extract relevant characteristics from the dataset, advanced approaches will be used. These designed features will capture detailed correlations between variables, improving the model's capacity to detect attrition trends. Some features might need to be left out; some might need to be processed before being input to the models. These need to be identified and handled accordingly.

3.4.3 Algorithm Selection

Several algorithms need to be researched and studied to identify which specific models to implement in this system. The base models that the author has chosen are FNN, Wide and Deep Model, and CNN.

Justification for FNN:

The FNN method was chosen because of its simplicity, effectiveness, and wide applicability to various types of classification problems. FNNs have been widely used in practice and are known to perform well on many types of datasets. Nandal, et al., 2024 in their study of employee attrition prediction identified FNN as being "the most successful technique" out of the several they implemented.

Model	Accuracy	Precision	F1-Score	Recall
Gradient Boosting	0.894	0.765	0.400	0.271
AdaBoost	0.883	0.619	0.377	0.271
Random Forest	0.872	1.00	0.41	0.021
Stacking	0.899	0.789	0.448	0.312
XG-Boost	0.886	0.650	0.382	0.271

Model	**Accuracy**	**Precision**	**F1-Score**	**Recall**
FNN	0.975	1.00	0.9126	0.8393
CNN	0.942	0.8148	0.8000	0.7857

Figure 3.1: Test Results of Nandal et al.'s study. (Nandal, et al., 2024)

Justification for Wide and Deep:

Ali Raza et al. (2022) investigated various machine learning methods for predicting worker turnover and highlighted the importance of selecting a model that suits the specific properties. In this study, the performance of wide and deep models was compared with traditional machine learning models such as gradient boosting, gradient boosting, random forests, and extremely randomized trees. The results show that Wide and Deep models can capture complex patterns and relationships in data, leading to better prediction of employee churn. This study concluded that combining Wide and Deep models with good data preparation techniques such as one-hot coding and normalization can improve accuracy.

The Wide and Deep model was able to achieve achieved a test AUC/ROC value of 0.75, showing a very high level of accuracy. For an imbalanced-class dataset, the AUC/PR value was 0.38, outperforming other models that achieved AUC/PR values of at most 0.37. These metrics show that this model is very effective at identifying at-risk employees and helping build retention strategies. (Ali Raza et al., 2022).

Justification for CNN:

Convolutional neural networks (CNN) were chosen as a base classifier because they are capable of handling high data and complex patterns that often arise in employee attrition domains. CNNs are particularly effective at extracting spatial hierarchies and patterns in data; This makes them ideal for tasks such as predicting employee turnover. CNNs have been successfully used to predict employee turnover in customer service; some with over 90% accuracy. (Rashid & Jabar, 2016). This success can be attributed to CNN's ability to learn hierarchical representation of data, which is especially useful for capturing subtle patterns and relationships that drive employee attrition.

Classifier	Correctly classified instances	Incorrectly classified instances	Kappa statistic	Mean absolute error	Root mean squared error	Relative absolute error, %	Root relative squared error, %
FRNN	973.3998	26.6002	0.9468	0.0913	0.2082	18.2564	41.6434
DT	978.4557	21.5444	0.9569	0.0287	0.1436	5.7455	28.7205
NB	888.7000	111.3000	0.7774	0.1320	0.3120	26.3899	62.4054
CNN	961.0827	38.9173	0.9222	0.0901	0.1820	18.0100	36.3968

Figure 3.2: Test Results from Rashid and Jabar's study (Rashid & Jabar, 2016).

3.4.4 Model Training and Validation

Using the prepared data from the previous sections, the selected models will be rigorously trained. Techniques such as cross-validation to ensure robustness and generalization will be included. The goal of this step is to provide the models with the capacity to generate accurate predictions based on the patterns seen in the training data.

3.4.5 Hyperparameter Tuning

It is critical to fine-tune model hyperparameters in order to improve their performance. Parameter configurations will need to be investigated systematically in order to find a balance between bias and variance, resulting in models with higher predictive ability. Auto-hyperparameter tuning tools such as RandomSearch can be used to implement automated hyperparameter tuning for the base models and some of the ensemble models too.

3.4.6 Ensemble approaches

Several ensemble approaches will be investigated in order to tap into the collective knowledge of various models. The techniques chosen are stacking, voting and simple average to merge individual predictions, improving prediction accuracy and dependability overall.

Justification for Stacking:

Compared with other ensembling methods, it is necessary to choose the stacking method for because this method can combine the power of several models to improve prediction accuracy. Stacked models have proven to be particularly useful in projects where the relationship between features and result is complex and non-linear.

A study published by G. D. Devi and S. Kamalakkannan, 2022 shows the effectiveness of stacking in predicting employee attrition. The results show that the author's proposed stacking model outperforms the other models in terms of accuracy, precision, recall, and F1 score. This suggests that the stacking method can be used to increase accuracy.

Model Description	Accuracy	Precision	Recall	Error rate
KNN	0.721	1.00	0.563	0.279
Random Forest	0.946	1.00	0.915	0.054
Naïve Bayes	0.973	0.958	1.00	0.027
Proposed WAM with ensemble LR	0.982	0.973	1.00	0.018

Figure 3.3: Test Results from the Study (Devi and Kamalakkannan, 2022).

Justification for Voting:

Choosing the voting ensemble method for predicting employee attrition over other ensembling methods is due to its simplicity and effectiveness in improving prediction accuracy. The voting ensemble method, specifically the majority voting ensemble model, works by having each model in the ensemble predict for all test instances, and the final output prediction is the one receiving the majority of the votes. This approach is particularly beneficial in tasks where the prediction is not as critical as the confidence in the prediction, as it provides a consensus among the models, potentially reducing the impact of outliers or models that are prone to overfitting.

The Work	Algorithm	Accuracy	Precision	Recall	F1-Measure	AUC
[6]	Voting Classifier	79.25%	N/A	N/A	12.22%	83.83%
[7]	LightGBM	85.3%	N/A %	N/A %	N/A %	N/A %
[8]	SVM	86.77 %	N/A %	N/A %	N/A %	N/A %
[9]	Extra Trees	93%	93%	93%	93%	N/A%
[10]	Bagging	83.74%	83.70%	83.70%	77.50%	N/A
[11]	LR	81%	43%	82%	56%	N/A
[12]	RF	85.12%	N/A%	N/A%	N/A%	80.84%
[13]	Voting Classifier	93%	N/A	N/A	58%	N/A
[13]	Stacking	88%	N/A	N/A	50%	N/A
[14]	NB	82.5%	38.6%	54.1%	44.6%	N/A%
[15]	CLARA[1]	65%	N/A%	N/A%	N/A%	N/A%
This Work	The proposed model	94.5255%	94.5%	94.5%	94.5%	98.5%

Figure 3.4: Test Results from the Study (Alshiddy and Aljaber, 2023).

3.5 PROJECT MANAGEMENT METHODOLOGY

3.5.1 Project Scope

In-Scope

The objective of this project is to develop an advanced ensemble prediction system for employee attrition utilizing a variety of machine learning and deep learning methods. The model will be trained and tested on a large dataset that includes key variables about the employee. The scope covers the investigation, selection, and implementation of machine learning and deep learning algorithms, as well as hyperparameter tuning to improve the performance of the model. To simplify user interactions with the prediction system, the implementation will include the creation of an intuitive graphical user interface (GUI). The project also includes documentation as well as an evaluation of the model's performance indicators, such as accuracy, precision, recall, and F1 Score.

Out-Scope

The prediction system will not be integrated with any live HR databases or systems as part of this project. The model's performance will be evaluated using historical data, and its implementation in a real-time setting is not included in the scope. Furthermore, the project does not include any specific retention techniques or interventions based on the model's predictions. The project's focus is on developing a prediction tool to help companies make educated decisions about employee attrition, and it does not include the implementation of such decisions or solutions.

3.5.2 GANTT Chart

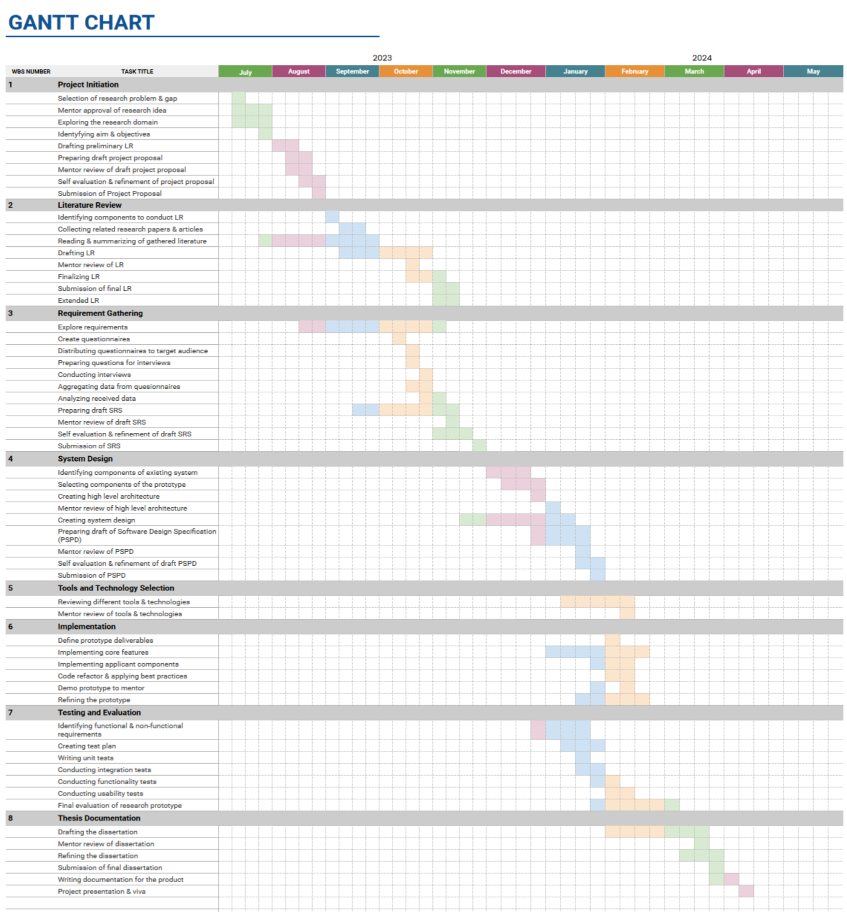


Figure 3.5 GANTT Chart (Self-Composed)

3.5.3 Deliverables, Milestones, and Dates of Deliverables

Deliverable	Date
Tentative Idea and finalized supervisor	31st Jul 2023
Project Initiation Document	28th Aug 2023
Final Project Proposal Submission	2nd Oct 2023
Literature Review Submission	30th Oct 2023
Software Requirement Specification (SRS) Submission	27th Nov 2023
Proof of Concept Submission	18th Dec 2023
Project Specification Design and Prototype Submission	29th Jan 2024
Minimum Viable Product Submission	4th Mar 2024
Thesis Submission	1st Apr 2024

Table 3.2: Deliverables and Milestones (Self-Composed)

3.6 RESOURCE REQUIREMENTS

To guarantee the effective creation, testing, and assessment of the employee attrition prediction system, the appropriate hardware, software tools, expertise, and a relevant dataset must be available.

3.6.1 Hardware Requirements

The following hardware resources will be required for the development and execution of the staff attrition prediction system:

- **Processor:** For intense tasks and managing development environments, a powerful CPU (e.g., Core i5 8th gen, Ryzen 5, or M1) is required. Primarily, an Intel CPU will be utilized.
- **Memory (RAM):** 8GB or more RAM is required for training models and handling large amounts of data.
- **Storage:** Around 30GB of disk space is required to store code and project data.

3.6.2 Software Requirements

The following software tools and platforms are required for the effective development of the system:

- **Operating System:** The project is platform-independent and can operate on a variety of OS's, including Windows, Linux, and macOS. However, because of its accessibility, convenience of setting up essential tools, and author familiarity, **Windows** will be the dominant development environment.
- **Programming Languages: Python** will be the main programming language used to create and implement the prediction models. Its simplicity, comprehensive ML libraries, and vast developer community make it a solid pick. **JavaScript** may be used to create server-side components, notably those that handle communication between the user and the model.
- **Machine Learning Libraries: TensorFlow** will be used for its comprehensive DL support in Python. It integrates well with Keras, making high-level API creation easier. **Scikit-Learn** may be used to supplement DL approaches for typical ML jobs.
- **Web Framework: Flask** will be the preferred choice for constructing the RESTful API that allows communication between the client and the prediction model. For extra feature needs, Node.js and Golang are alternate possibilities.
- **Front-End Framework:** The client-side interface will be built using **Angular.js** The author prefers Angular because of its performance, large developer community, and familiarity. If Angular proves insufficient, Vue.js and Svelte are viable alternatives.
- **Integrated Development Environment (IDE):** Because of its versatility, lightweight nature, and wide plugin support, **VSCode** will be the main development environment. If particular Python environment needs arise, PyCharm may serve as an additional choice for Python development.
- **Development Environment:** Due to its adaptability and local execution, **Jupyter Notebook** will be utilized as the primary development environment for constructing and testing the forecasting model. If GPU resources are necessary for model training, Google Colab will be a backup choice.
- **Reference Management:** Because of its user-friendly design and the author's desire, **Zotero** will be utilized to handle references and research items.

- **Document Creation:** Microsoft Office will be used in combination with the Google Suite to create research reports and professional documents as needed.
- **Cloud Storage: Google Drive** will be the primary cloud storage solution for storing up research documents and files.
- **Design Tools: Figma** may be used for design and prototyping, **Draw.io** to create flowcharts and diagrams, and **Canva** may be used to prepare presentation slides.

3.6.3 Skills Requirements

The following skills will be required for the project's successful completion:

- Strong understanding and practical expertise with machine learning methods, particularly those used for classification tasks.
- Knowledge of ensemble learning techniques and their applications in improving forecasting algorithms.
- Ability to identify and determine appropriate features for improved accuracy.
- Ability to document the project, including model information, data preprocessing steps, and results.

3.6.4 Data Requirements

The employee attrition prediction system's development and assessment will depend on a large dataset encompassing employee-related factors such as socioeconomic factors, job satisfaction, performance measures, and past attrition data. To guarantee reliable model training and assessment, the dataset should be fair and well-processed. The dataset to be utilized in the project is the IBM HR Analytics Employee Attrition & Performance dataset found on Kaggle consisting of 35 attributes of employee attrition.

3.7 RISK MANAGEMENT

Risk Item	Severity*	Frequency*	Mitigation Plan
Inadequate Data Quality	4	3	For optimal data quality, use extensive data cleaning/validation methods.
Model Overfitting	3	4	Monitor model performance on a regular basis, use suitable regularization

			approaches and verify results using out-of-sample data.
Limited Availability of Data	4	3	Investigate ways to obtain additional relevant data. Data augmentation strategies and cooperating with other data sources may be considered.
Technical Challenges	3	3	Establish a well-documented procedure for development and monitoring changes with version control. Update libraries on a regular basis and respond to technical issues as soon as practical.

*On a scale of 1 to 5 with 1 being the minimum and 5 being the maximum

Table 3.3: Deliverables and Milestones (Self-Composed)

CHAPTER 4: SOFTWARE REQUIREMENT SPECIFICATION

4.1 CHAPTER OVERVIEW

This chapter will cover the requirement elicitation process and findings, conduct a stakeholder analysis, and establish a context diagram and a use case diagram for the system. Priorities will be assigned to all the requirements identified based on the findings discovered during the requirement elicitation process.

4.2 RICH PICTURE DIAGRAM

The Rich Picture Diagram showing the stakeholders, structure, geographical localities, people, entities, information and data involved in the system is given below.

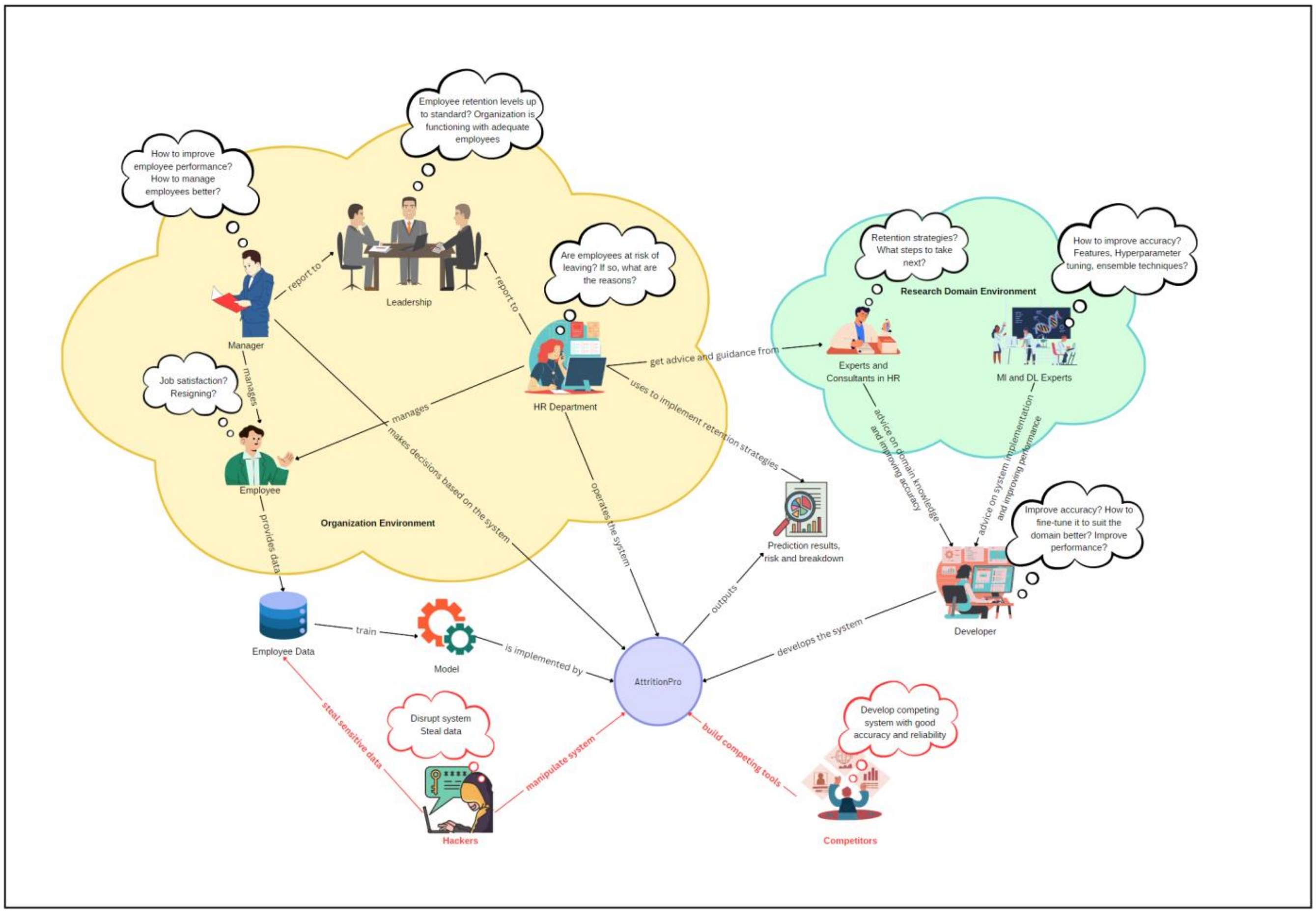


Figure 4.1 Rich Picture Diagram (Self-Composed)

4.3 STAKEHOLDER ANALYSIS

4.3.1 Stakeholder Onion Model

The stakeholder onion model below provides a graphical representation of the many stakeholders involved in the system, establishing several categories of stakeholders (internal, external primary and secondary) and also, their influence on the project.

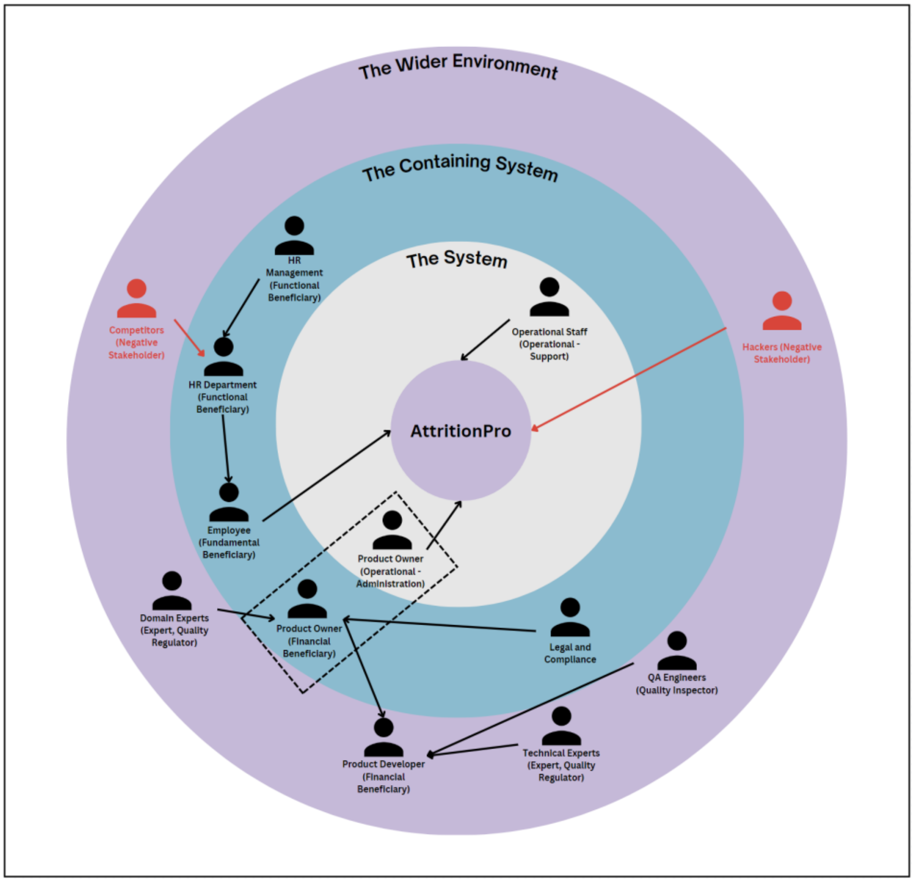


Figure 4.2 Stakeholder Onion Model (Self-Composed)

4.3.2 Stakeholder Description

The stakeholders of AttritionPro encompass a wide range of people, organizations, and entities, each with their own varying levels and aspects of influence. The end users of the system are the HR managers.

Stakeholder	Role	Description
Product Owner	Financial Beneficiary, Operational - Administration	Handles higher-level decisions regarding the system and manages the general operation of the prediction system.
Operational Staff	Operational – Support	Maintains and handles AttritionPro. Passes on results from the system to the HR Department and managers. Oversee the system's implementation and operation.
HR Management	Functional Beneficiary	Use the forecasts to plan strategically for their company and to manage their personnel efficiently. Direct and develop strategic HR planning and proactively handle attrition concerns using the predictions of the system.
HR Department		
Employee	Fundamental Beneficiary	Employees provide the input data for the predictions. The focus of the predictive system, reaping the benefits of enhanced retention strategies and increased employment stability.
Domain Experts	Expert, Quality Regulator	Provide advice and contribute important perspectives and suggest guidance on implementing industry best practices.
Product Developer	Financial Beneficiary	Product developer develops the attrition prediction system that uses DL Ensemble methods
Technical Experts	Expert, Quality Regulator	Provide technical advice and suggest guidance on implementing domain best practices. Guarantee the impact and smooth integration of the entire system.
QA Engineers	Quality Inspector	Tests and evaluates the performance of the system and ensures that the accuracy, reliability

		and functionality of the system is up to standard.
Hackers	Negative Stakeholders	May attempt to gain access to sensitive information with the intention of stealing and misusing private data, may attempt to alter and disrupt system operations.
Competitors		May attempt to provide an equivalent service to the user in order to create a rival system or reverse-engineer the current system.

Table 4.1: Stakeholder Description (Self-Composed)

4.4 REQUIREMENT ELICITATION METHODS

The requirement elicitation methods employed are Literature Review (LR), Surveys, and Prototyping.

4.4.1 Literature Review (LR)

The Literature Review will involve detailed research into existing work in the domain and academic publications relevant to the project domain. This methodology is useful in that it helps to comprehend the scope of existing solutions, their shortcomings and the research gaps that require research attention. The literature review will also provide information on ensemble techniques, deep learning algorithms and the techniques that were used in applying them into this domain. This information would be beneficial in choosing the right variables, parameters, and other fine tunings for this project. (Lim, Henriksson and Zdravkovic, 2021).

4.4.2 Surveys

The survey methodology will be used to collect the opinions and ideas of employees, HR personnel and leadership of companies, in order to better optimize the system to suit their requirements. The surveys present a special chance to find factors that are important for the organizations where the model will be used, but may not have been taken into account in the literature.

4.4.3 Prototyping

In order to gather further requirements, a reduced model of the system was produced. This technique is especially helpful since it generates early feedback and visualizes how the finished product would function before it is fully implemented. It was beneficial in order to define project goals and scope. The research objectives and the author's capabilities were analyzed and evaluated. The process flow and features, functional and non-functional were also further defined in this step. More details about the implementation of the ML Ensembles can be found in the **Appendix B: Software Requirement Specification**.

4.5 DISCUSSION OF FINDINGS

The findings obtained from the various requirement elicitation methods are discussed and analyzed below.

4.5.1 Literature Review

Finding	**Citation**
To forecast employee attrition, deep learning techniques, ensemble learning approaches, and traditional machine learning methodologies have all been used. Each of these methods has advantages and disadvantages of its own, and the results of using them to forecast staff turnover are promising.	(Yedida, Rahul, et al., 2018).
Conventional machine learning methods including logistic regression, decision trees, RF, and SVMs have been widely used because of their ease of use and interpretability. 2020 (Pampula, Jain, and Jain). They usually have trouble handling complex and non-linear data patterns, though. Gurler, Gungor, and Pak (2023).	(Jain, Jain, and Pamula). (Gurler, Pak, and Gungor, 2023).
It has been discovered that ensemble learning techniques, which combine numerous models to boost prediction accuracy, reduce bias and variance, enhancing the performance of individual models. Gurler, Gungor, and Pak (2023). They are less interpretable than individual models and often require large processing resources. [Ganthi and others, 2022).	(Gurler, Pak, and Gungor, 2023). (Ganthi, et al., 2022).

Multiple layers of representations, each of which converts the input data into a more abstract and composite representation, may be learned via deep learning techniques like CNNs, RNNs, and Transformer-based networks. (Et al., Fallucchi, 2020). Notwithstanding their encouraging results, deep learning techniques have a number of drawbacks, such as the requirement for massive amounts of data and computational power in addition to their complex, non-linear nature, which makes them challenging to understand.	(Fallucchi, et al., 2020).
Even though these methods have made great strides in staff loss predictions, many obstacles remain to be addressed. Subsequent investigations ought to concentrate on investigating innovative machine learning, deep learning, and ensemble methodologies, mitigating their constraints, and optimizing their efficacy.	(Gim and Im, 2023)

Table 4.2: Discussion of Findings - Literature Review (Self-Composed)

4.5.2 Survey

The findings and results of the survey conducted are discussed and analyzed in the **Appendix A: Survey**.

4.5.3 Prototyping

Criteria	**Findings**

To monitor performance of regular ML techniques	The ML classification algorithms, AdaBoost Classifier, Bagging Classifier, Gradient Boosting Classifier, K Nearest Neighbors Classifier, Random Forest Classifier and Multi-Layer Perceptron Classifiers were implemented to assess their performances and accuracies.
	
	`print(model_and_score)`
	
	`{'bagging classifier': '97.08454810495627%', 'KNN classifier': '86.88046647230321%', 'Random Fo rest calssifier': '100.0%', 'Adaboost classifier': '85.71428571428571%', 'Gradientboot classifi er': '90.0874635568513%', 'MLP': '84.25655976676384%'}`

To identify data loading and processing methods

Data Preprocessing and Pipelining

```
X_train=project_data.drop(columns=["Attrition"])
y_train=project_data["Attrition"]

X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.3)
```

```
print('Train dataset shape:',X_train.shape)
print('Test dataset shape', y_train.shape)
```

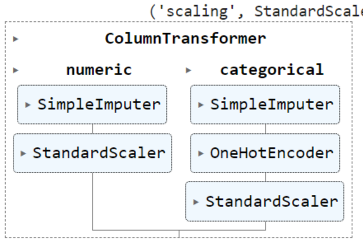
```
Train dataset shape: (1029, 34)
Test dataset shape (1029,)
```

```
numeric_columns = X_train.select_dtypes(exclude='object').columns
print(numeric_columns)
print('*'*100)
categorical_columns = X_train.select_dtypes(include='object').columns
print(categorical_columns)
```

```
Index(['Age', 'DailyRate', 'DistanceFromHome', 'Education', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobSatisfaction', 'MonthlyIncome',
       'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
       'PerformanceRating', 'RelationshipSatisfaction', 'StandardHours',
       'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
       'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')
****************************************************************************************************
Index(['BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole',
       'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
```

```
numeric_features = Pipeline([
    ('handlingmissingvalues',SimpleImputer(strategy='median')),
    ('scaling',StandardScaler(with_mean=True))
])

print(numeric_features)
print('*'*100)

categorical_features = Pipeline([
    ('handlingmissingvalues',SimpleImputer(strategy='most_frequent')),
    ('encoding', OneHotEncoder()),
    ('scaling', StandardScaler(with_mean=False))
])

print(categorical_features)

processing = ColumnTransformer([
    ('numeric', numeric_features, numeric_columns),
    ('categorical', categorical_features, categorical_columns)
])

processing
```

```
Pipeline(steps=[('handlingmissingvalues', SimpleImputer(strategy='median')),
                ('scaling', StandardScaler())])
****************************************************************************************************
Pipeline(steps=[('handlingmissingvalues',
                 SimpleImputer(strategy='most_frequent')),
                ('encoding', OneHotEncoder()),
                ('scaling', StandardScaler(with_mean=False))])
```



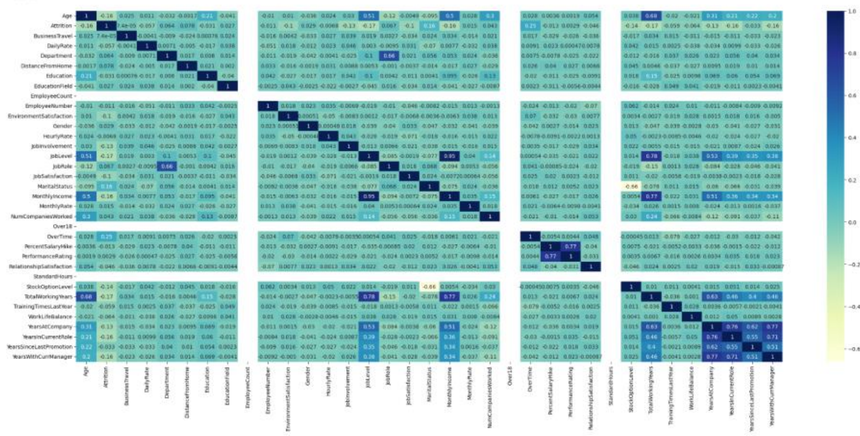
To identify effect of different features	**Correlation Analysis**
	`df1 = project_data.copy()`
	
	`encoder = LabelEncoder()`
	`for column in categorical_columns:`
	`    df1[column] = encoder.fit_transform(df1[column])`
	
	`plt.figure(figsize=(30,12))`
	`corr = df1.corr()`
	`sns.heatmap(corr, annot=True, cmap="YlGnBu")`
	
	<Axes: >
	

Figure 4.3: Discussion of Findings - Prototyping (Self-Composed)

Results of Prototyping:

```
for index, tup in enumerate(trained_models):
    prepare_classification_report(tup[0], tup[1])
    print("\n")
```

```
bagging classifier Report :
              precision    recall  f1-score   support

          No       0.88      0.96      0.91       379
         Yes       0.39      0.18      0.24        62

    accuracy                           0.85       441
   macro avg       0.63      0.57      0.58       441
weighted avg       0.81      0.85      0.82       441
```

```
Adaboost classifier Report :
              precision    recall  f1-score   support

          No       0.88      0.98      0.93       379
         Yes       0.67      0.19      0.30        62

    accuracy                           0.87       441
   macro avg       0.77      0.59      0.62       441
weighted avg       0.85      0.87      0.84       441
```

```
KNN classifier Report :
              precision    recall  f1-score   support

          No       0.88      0.95      0.92       379
         Yes       0.42      0.21      0.28        62

    accuracy                           0.85       441
   macro avg       0.65      0.58      0.60       441
weighted avg       0.82      0.85      0.83       441
```

```
Gradientboot classifier Report :
              precision    recall  f1-score   support

          No       0.88      0.97      0.92       379
         Yes       0.54      0.23      0.32        62

    accuracy                           0.86       441
   macro avg       0.71      0.60      0.62       441
weighted avg       0.84      0.86      0.84       441
```

```
Random Forest calssifier Report :
              precision    recall  f1-score   support

          No       0.89      0.97      0.93       379
         Yes       0.57      0.26      0.36        62

    accuracy                           0.87       441
   macro avg       0.73      0.61      0.64       441
weighted avg       0.84      0.87      0.85       441
```

```
MLP Report :
              precision    recall  f1-score   support

          No       0.87      0.99      0.92       379
         Yes       0.55      0.10      0.16        62

    accuracy                           0.86       441
   macro avg       0.71      0.54      0.54       441
weighted avg       0.82      0.86      0.82       441
```

Figure 4.4: Results of Prototyping

4.6 SUMMARY OF FINDINGS

To summarize the findings obtained from the Literature Review, Interviews, Surveys and Brainstorming are summarized and tabulated below.

No.	Finding	Literature Review	Survey	Prototyping
1	The problem domain defined is a critical issue in that it negatively affects organizations heavily when valued employees leave the organization.	✓		
2	The prediction system proposed would benefit organizations, HR departments, managers/leadership, and employees.	✓	✓	
3	The factors most contributing to employee resignations are work environment satisfaction, monthly income, salary promotions, working hours and job involvement.	✓	✓	✓
4	Generating the risk of attrition, on top of the binary prediction would be extremely beneficial, allowing management to focus their priority on valuable high-risk employees.		✓	
5	The ability to produce a feature-wise breakdown of the various features affecting an employee's resignation risk and to what extent, each of these features contribute would be useful in developing retention strategies and to target the 'Why?' of the problem. (E.g.: Why exactly are employees leaving?).	✓		✓
6	Developing an intuitive, user-friendly GUI for users to conveniently use and operate the system would increase the value and functionality of the general system.	✓		✓
7	Various deep learning ensemble techniques are yet to be developed in the research domain and are predicted by the research community to perform well.	✓		

Table 4.3: Summary of Findings (Self-Composed)

4.7 CONTEXT DIAGRAM

The context diagram showing the interactions between the various entities involved in the system.

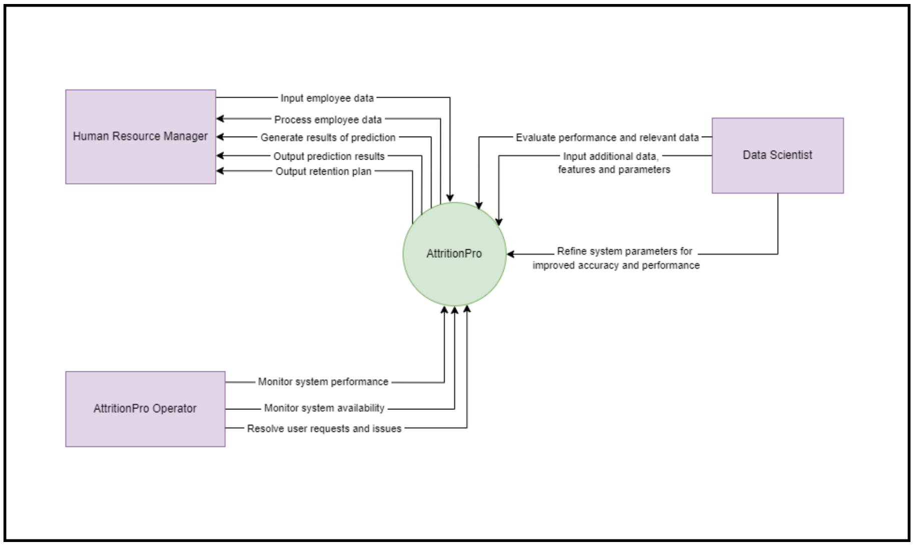


Figure 4.3 Context Diagram (Self-Composed)

4.8 USE CASE DIAGRAM

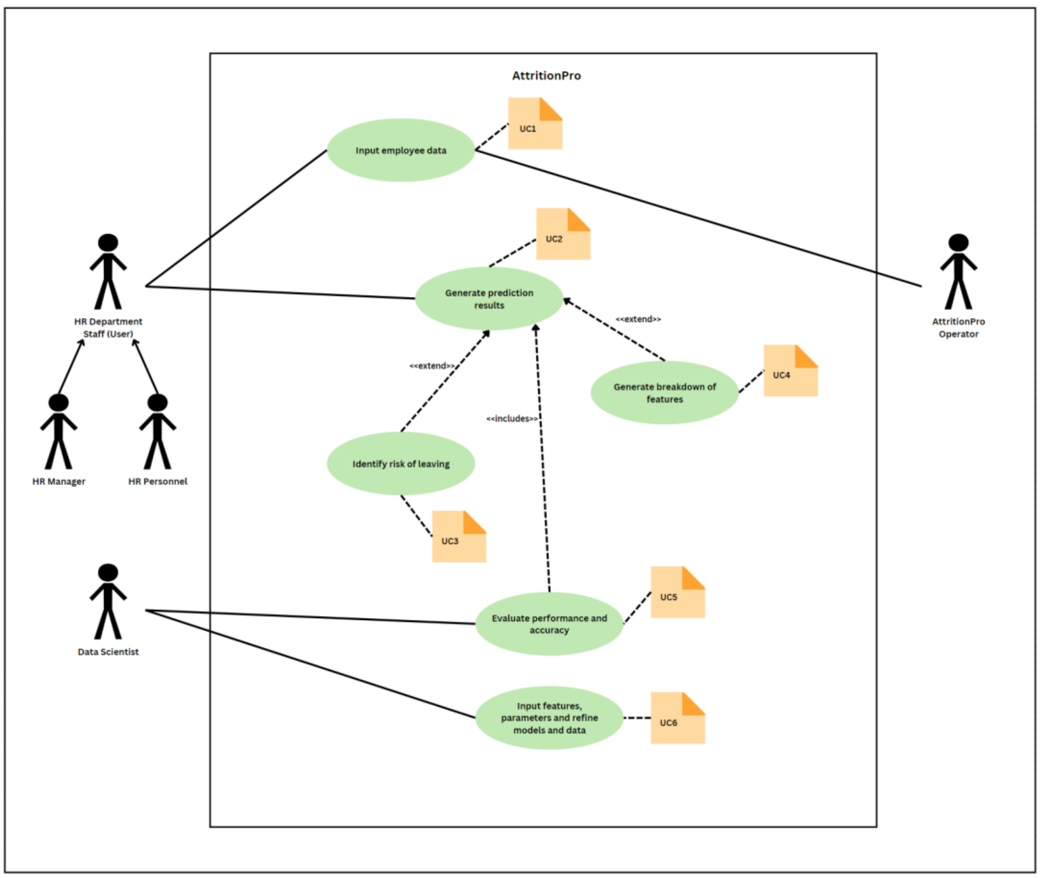


Figure 4.5: Use Case Diagram (Self-Composed)

4.9 USE CASE SPECIFICATION

The Use Cases presented in the diagram above are discussed in further detail in the following tables. The rest of the use case specifications are attached in **Appendix B: Use Case Specification**.

4.10 REQUIREMENTS WITH PRIORITIZATION

4.10.1 Functional Requirements

The functional requirements of the system are given below, with their priorities according to the MoSCoW principle.

ID	Requirement	Priority	Use Case
FR1	Users must be able to input a certain employee's details to make a prediction of their attrition.	Must	UC1
FR2	Prediction results should be generated for the input data, showing the attrition of that employee.	Must	UC2
FR3	The results of the prediction should be displayed to the user in a user-friendly, intuitive way.	Should	UC2
FR4	The risk of attrition for an employee should be generated from the data input by the user.	Should	UC3
FR5	The user should be able to view the generated risk of attrition in a presentable manner.	Should	UC3
FR6	The prediction history could be saved, and the most recent predictions could be shown to the user on the GUI.	Could	UC4
FR7	The feature-wise breakdown of the factors most affecting an employee's risk of attrition would be displayed to the user.	Would	UC4
FR8	The system would have an intuitive GUI for the user to interact with the system.	Would	NA

Table 4.4: Functional Requirements (Self-composed)

4.10.2 Non-Functional Requirements

ID	Requirement	Description	Priority
NFR1	Performance	The performance of the system, in terms of speed, efficiency and prediction, should be consistent and adequate.	Desirable
NFR2	Security	Employee details and other confidential and sensitive information need to be protected appropriately.	Desirable
NFR3	Scalability	The system should be able to scale to function at a larger scale and capable of handling large amounts of data and parameters.	Desirable
NFR4	Quality of Output	The system should be able to make reliable and accurate predictions consistently.	Important
NFR5	Usability	The usability of the system has to be at a level where it is easy to use, convenient and straightforward. It should be simple and easy to understand and beginner friendly.	Important

Table 4.5: Non-Functional Requirements (Self-composed)

4.11 CHAPTER SUMMARY

This chapter covered the specification of the stakeholders, requirements, context and use cases of the system and established the priorities and definition of the prediction system and the wider environment.

CHAPTER 5: SOCIAL, LEGAL, ETHICAL AND PROFESSIONAL ISSUES

5.1 CHAPTER OVERVIEW

This chapter will go over all of the social, legal, ethical and professional issues that this project might bring up. The social issues section will explore the impact that this project might have on the wider environment surrounding it and society in general, specifically identifying any negative or harmful effects that it might have. In a similar way, any issues that develop during the conducting of this project will be analyzed and discussed in this chapter.

5.2 SLEP ISSUES AND MITIGATION

5.2.1 Social Issues

The use of employee attrition prediction systems raises various social issues, specifically due to their potential to worsen currently existing biases and discrimination within the datasets. This is a possibility because if biases are not carefully managed, they could lead to unfair treatment or discrimination against certain groups of people. This, in turn, could cause complications with inequalities, biases and discrimination in society.

On top of that, introducing systems like this could lead to employees getting displaced from their workplaces because automation and prediction might cause certain organizations to replace human workers to avoid problems that could occur due to human factors like resignations.

It is essential to make sure that these systems are developed and implemented in a transparent manner in order to properly address and mitigate these social concerns. This includes performing detailed impact assessments to determine other potential social hazards and include important stakeholders in the process to get feedback and applying thorough data preprocessing and cleaning to remove biases in the training data.

5.2.2 Legal Issues

Attrition prediction systems could bring up a few legal issues, including concerns about data privacy, intellectual property rights, and the need to comply with industry regulations. Intellectual property rights for the algorithms, models, and datasets are also important to consider to ensure proper legal compliance.

5.2.3 Ethical Issues

Ethical considerations are very important as these systems can significantly affect people's lives and societal well-being. A major ethical issue is the risk of algorithmic bias, where unfair or discriminatory results may be produced because of biases in training data.
Some important factors that need to be considered are transparency, accountability, and the potential for unintended consequences to society.

To combat these ethical challenges, organizations need to focus on ethical principles like fairness, transparency, and accountability throughout the process. This includes using techniques to detect and reduce bias, providing explanations for the decisions made by the algorithms, and setting up systems for detecting accountability.

5.2.4 Professional Issues

Personnel operating these systems need to have the right technical skills and knowledge to develop and use them effectively and with responsibility. Industry codes of conduct and ethical guidelines need to be followed strenuously. This helps make sure that they are developed and used with integrity and professionalism. To handle these, organizations need to invest in ongoing training and professional development for the people working. They should set clear guidelines and standards for professional behavior and promote a culture of ethical responsibility and accountability within the organization.

5.3 CHAPTER SUMMARY

There are several social, legal, ethical and professional issues when it comes to developing a system such as AttritionPro. These need to be appropriately mitigated and resolved to ensure that the system is fair and transparent and doesn't introduce any socio-ethical concerns.

CHAPTER 6: DESIGN

6.1 CHAPTER OVERVIEW

This chapter will delve into the expectations of the project design-wise and scope over what technologies the author plans to use and the architecture that the implementation will be using. The component diagram and the data flow structures will also be demonstrated.

6.2 DESIGN GOALS

Design Goal	Description
Performance	The system needs to be effective, dependable, and solid. Therefore, the system has to function consistently and adequately in terms of speed, efficiency, and prediction. This design goal would be the highest priority over the others due to it being an imperative requirement of the system.
Correctness	Correctness refers to the accuracy and validity of the results generated. This design goals establishes that the developed system has to generate reliable, genuine outputs.
Usability	The system's usability needs to be such that it is simple, convenient, and easy to use. It needs to be simple, understandable, and accessible to users new to it. The user interface has to be intuitive enough for users to easily use it.
Scalability	In addition to being able to handle enormous volumes of data and parameters, the system should be scalable to operate at a bigger scale. The system should be able to adjust and adapt to expansion.
Adaptability	Adaptability refers to the capability of the system to adjust and transformed to suit new functionalities, features, variables, data, and to keep up with evolving requirements over time.

Table 6.1: Design Goals (Self-Composed)

6.3 SYSTEM ARCHITECTURE DESIGN

6.3.1 Tiered Architecture Diagram

The architecture diagram for the implementation of AttritionPro is shown below. The author has chosen the Three-Tiered Architecture. This is due to the system needing a flexible and more isolated architecture and a Three-Tiered Architecture Diagram being the ideal choice in that case.

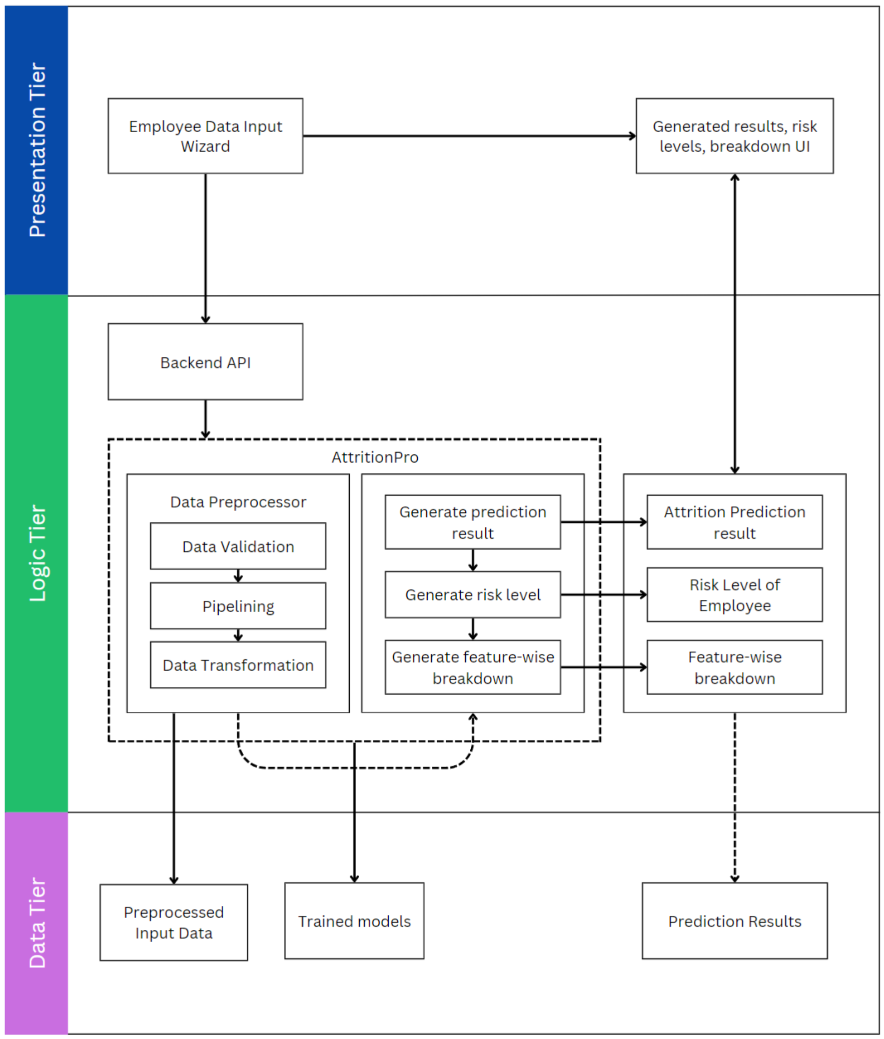


Figure 6.1: Tiered Architecture Diagram (Self-Composed)

6.3.2 Discussion of Tiers of the Architecture

Data Tier

1. Preprocessed Input Data: The data that is generated after all of the transformations and preprocessing, to be used in the prediction are represented in this diagram as the Preprocessed Input Data.

2. Trained models: These are the models and algorithms trained by the system for use in the predictions. These are deep learning neural networks and algorithms utilizing the deep learning algorithms in the ensemble technique.

3. Prediction Results: These are all of the generated outputs of the system, including the generated prediction result, risk of attrition, and the breakdown of the features contributing to attrition.

Logic Tier

1. Backend API: This represents the API interface that will communicate between the frontend user interface and the prediction system.

2. Data Preprocessor: The data from the user input needs to be preprocessed before being sent to the predictor. The format of the data, handling inaccurate and erroneous values and preparing the data to be input into the system will be done in this step of the Logic Tier.
   a. Data Validation: The user input data needs to be thoroughly checked and validated beforehand. Here, the system will check whether the employee data is in a useable, appropriate and valid format before proceeding to the next steps.
   b. Pipelining: The pipelining for the preprocessing is done here. In this step, the numeric and categorical data will be identified and the preprocessing steps for each will be defined.
   c. Data Transformation: The data will be transformed and modified to act as suitable inputs for the system. Missing and incorrect values will be dealt with, feature scaling would be done, and the data will be standardized accordingly.

3. AttritionPro Prediction System: The preprocessed data will be used to generate predictions using the DL ensemble models.
    a. Generate prediction result: The binary classification results of the employee data will generate the prediction result of attrition.
    b. Generate risk level: The risk of attrition will be generated to represent the chance of the employee resigning from the company.
    c. Generate feature-wise breakdown: The Feature-wise breakdown of the factors contributing to the employee’s attrition are generated in this step of the architecture.

Presentation Tier

1. Employee Data Input Wizard: The user interface that the user, HR personnel, would be using to input details about the employee such as job role, work environment, age, monthly income, job involvement, age, salary increment percentage, stock options, business trips and travel options, department and working hours. These features will be used for generating the predictions by the system.

2. Generated results, risk levels and breakdown UI: This is where all of the outputs of the system will be displayed to the user. This includes the generated result, attrition risk, and the breakdown of the features affecting attrition.

6.4 SYSTEM DESIGN

6.4.1 Choice of Design Paradigm

The project's design process will be based on the concepts of Structured systems analysis and design method (SSADM). SSADM was chosen for its potential to allow the structuring of software, its reusability, and extension. Because of the complexity of the project, a design solution that enables the smooth integration of several ML and deep learning algorithms is required. The emphasis on clarity and structure in SSADM will help to create a reliable and consistent prediction system. SSADM also places heavy emphasis on the data of the system and makes it simpler to comprehend the connections between data and processes of the system.

6.4.2 Data Flow Diagrams

The data flow diagrams below represent the various components involved in the project and the connections between them.

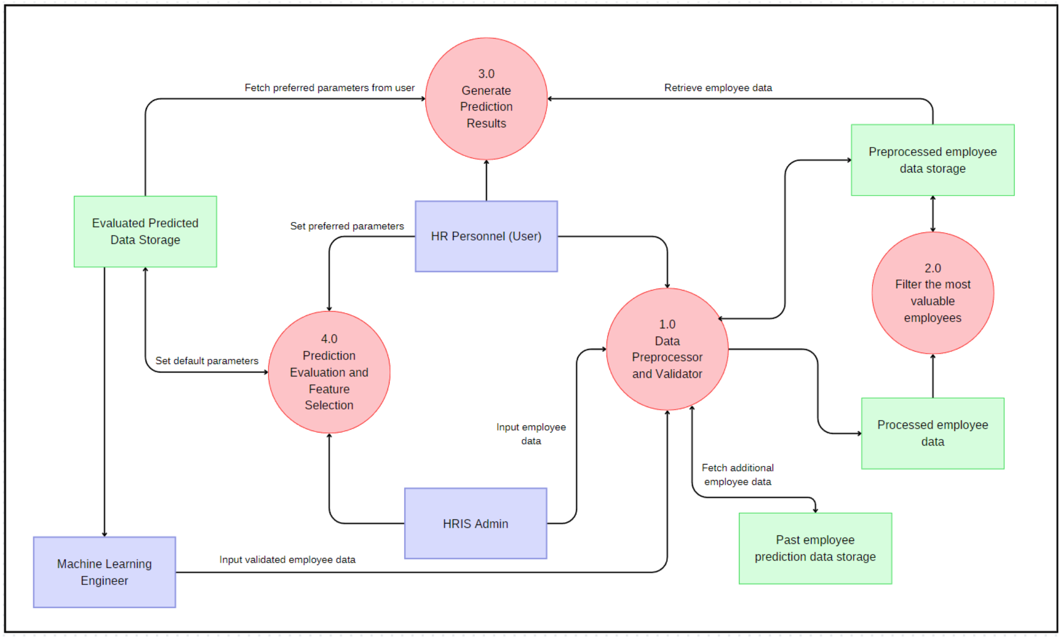


Figure 6.2: Data Flow Diagram - Level 01 (Self-Composed)

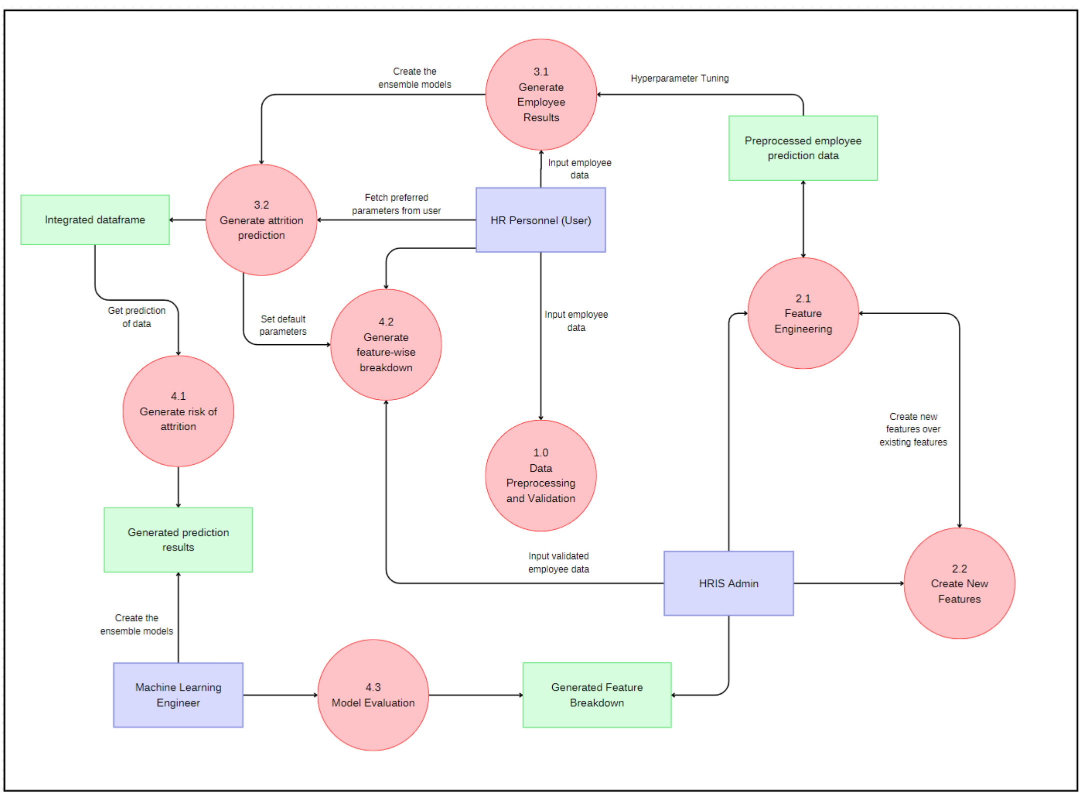


Figure 6.3: Data Flow Diagram - Level 02 (Self-Composed)

6.4.3 Algorithmic Design

The algorithm presented for AttritionPro is designed to enhance the predictive performance of 3 base deep learning models, i.e. a Feedforward Neural Network (FNN), a Wide & Deep model, and a Convolutional Neural Network (CNN), by combining their predictions. This ensemble method is inspired by the concept of a Super Learner, where meta-model is created and trained to learn the optimal combination of the individual predictions from base models.

The Super Learner algorithm is especially effective in situations where the base models have different strengths and weaknesses, and their combined predictions can lead to more accurate and robust predictions.

Base Classifiers:

1. FNN (Feedforward Neural Network)

The FNN method was chosen as one of the base classifiers due to its simplicity, effectiveness, and wide applicability to various types of classification problems. FNNs have been widely used in practice and are known to perform well on many types of datasets. Additionally, the flexibility of FNN architecture allows for easy experimentation with different network configurations and hyperparameters, making it a suitable choice for hyperparameter tuning experiments.

Design	
Input Layer	The input layer consists of neurons corresponding to the features of the input data. The number of neurons in this layer was determined by the input dimensionality.
Hidden Layers	The FNN contains two hidden layers, each followed by a dropout layer to prevent overfitting. The number of neurons in these hidden layers and the dropout rate are hyperparameters that are tuned during the hyperparameter optimization process.
Output Layer	The output layer consists of a single neuron with a sigmoid activation function. Since this is a binary classification problem, the sigmoid function outputs probabilities that the input belongs to the positive class.

Loss Function	The binary cross-entropy loss function is used, which is commonly used for binary classification problems.
Optimizer	The Adam optimizer is used to optimize the model's weights during training.
Hyperparameter Tuning	
Hyperparameters Tuned	• Number of units (neurons) in each hidden layer (units_1 and units_2) • Dropout rates for each dropout layer (dropout_1 and dropout_2) • Learning rate of the optimizer (learning_rate)
Search Space	The search space for each hyperparameter was defined using appropriate ranges and step sizes. For example, the number of units in each hidden layer was chosen from a range between 32 and 512 with a step size of 32.
Objective	The objective of the hyperparameter tuning process was to maximize the validation accuracy of the model.
Optimization Algorithm	The RandomSearch algorithm was used to randomly sample hyperparameter combinations from the search space and evaluate their performance on the validation data.
Training and Evaluation	The best model found during the hyperparameter tuning process was trained on the entire training data using the optimal hyperparameters. Its performance was then evaluated on the test data to obtain the final accuracy.

Table 6.2: Algorithmic Design of FNN (Self-Composed)

The choice of two hidden layers in the FNN architecture is a common practice in neural network design. Adding more layers can potentially capture more complex patterns in the data, but it also increases the risk of overfitting, especially with limited training data. The number of neurons in each hidden layer is chosen within a reasonable range to balance model complexity and generalization performance. Dropout layers are added after each hidden layer to regularize the model and prevent overfitting by randomly dropping a fraction of the neurons during training.

2. Wide and Deep

Wide and Deep architecture, launched by Google in 2016, provides the advantages of memory (wide area) and extension (deep) learning technologies. Through the combination of wide and deep layers, the model can learn both shallow, broad patterns and deep, complex relationships in data. This approach is especially useful for data structures that combine categorical and numerical features, because capturing both types of patterns can make predictions more efficient. In addition, Wide and Deep models have been shown to perform well in many real-world applications, making them a popular choice for a variety of classifications.

Design	
Input Layer	The model has two input layers, one for the wide component and one for the deep component. The two input layers have the same similarity in input dimensions.
Wide Component	The wide component consists of a single dense layer with a ReLU activation function. This component is responsible for memorizing feature interactions and is typically less deep but wider.
Deep Component	The deep component consists of two dense layers with ReLU activation functions. These layers capture complex patterns in the data and are responsible for generalization. The number of units in each deep layer is a hyperparameter set during the hyperparameter optimization.
Merged Layer	The outputs from the wide and deep components are concatenated into a single merged layer.
Output Layer	The merged layer is fed into the output layer, which has a single neuron with a sigmoid activation function, suitable for binary classification tasks.
Loss Function	The binary cross-entropy loss function is used, which is frequently used for binary classification problems.
Optimizer	The Adam optimizer is used to optimize and adjust the weight of the model during training.
Hyperparameter Tuning	
Hyperparameters Tuned	Number of units (neurons) in the wide layer (wide_units) and each deep layer (deep_units_1 and deep_units_2)

Search Space	The search space for each hyperparameter was defined using appropriate ranges and step sizes. For example, the number of units in the wide layer was chosen from a range between 32 and 256 with a step size of 32. The neurons for the first deep layer were selected from a range between 32 and 256 with a step size of 32 and for the second deep layer, range between 16 and 128 and a step size of 16.
Objective	The objective of the hyperparameter tuning process was to maximize the validation accuracy of the model.
Optimization Algorithm	The RandomSearch algorithm was used to randomly sample hyperparameter combinations from the search space and evaluate their performance on the validation data.
Training and Evaluation	The best model found during the hyperparameter tuning process was trained on the entire training data using the optimal hyperparameters. Its performance was then evaluated on the test data to obtain the final accuracy.

Table 6.3: Algorithmic Design of Wide and Deep (Self-Composed)

The Wide and Deep model consists of two types of layers: wide and deep.

Wide Layers: These are dense layers that perform simple, linear computations. They typically have a large number of units to handle the high-dimensional, sparse input features.

Deep Layers: These are dense layers that form a deep neural network. They capture complex, nonlinear patterns in the data through multiple layers of computation. The number of units and the depth of the network are hyperparameters that can be tuned during the optimization process.

3. CNN (Convolutional Neural Network)

CNNs are well-suited for tasks involving sequence data because they are capable of capturing spatial and temporal dependencies, respectively. CNNs automatically learn hierarchical patterns and features from input data, making them more efficient. The CNN-based classifier was chosen because it can effectively capture spatial dependencies in the data, and perform hyperparameter tuning to improve its performance by exploring various sets of layers and hyperparameters.

Design	
Input Layer	The model has an input layer consisting of Conv1D, which is suitable for processing one-dimensional sequence data, like time series or 1D signal data.
Convolution Layers	The model consists of one Conv1D layer with varying numbers of filters and kernel sizes. Filters determine the number of output filters in the convolution, while kernel sizes define the length of the 1D convolution window.
Pooling Layer	Following each Conv1D layer, a MaxPooling1D layer with a pool size of 2 is applied to reduce the dimensionality of the feature maps.
Flatten Layer	The Flatten layer converts the 2D feature maps into a 1D vector to prepare the data for input into the fully connected layers.
Fully Connected Layer	The model has two fully connected Dense layers with ReLU activation functions. The number of neurons in each layer is a hyperparameter tuned during the optimization process.
Dropout Layers	Dropout layers with a dropout rate of 0.2 are included after each Dense layer to prevent overfitting by randomly dropping a fraction of the input units.
Output Layer	The output layer consists of a single neuron with a sigmoid activation function, suitable for binary classification tasks.
Loss Function	Binary cross-entropy loss function is used to measure the difference between the predicted probabilities and the actual binary labels.
Optimizer	The Adam optimizer is utilized to optimize the model's weights during training. Adam is an adaptive learning rate optimization algorithm that combines the advantages of AdaGrad and RMSProp.
Hyperparameter Tuning	
Hyperparameters Tuned	Number of filters in the convolutional layers (filters) , and the size of the convolutional kernel (kernel_size).
Search Space	The search space for each hyperparameter was defined using appropriate ranges and step sizes. For example, the number of filters was chosen from

	a range between 16 and 64 with a step size of 16, and the kernel size was chosen from a range between 2 and 5.
Objective	The objective of the hyperparameter tuning process was to maximize the validation accuracy of the model.
Optimization Algorithm	The Hyperband algorithm was used for hyperparameter tuning, which employs a successive halving algorithm to efficiently allocate resources to promising hyperparameter configurations.
Training and Evaluation	The best model found during the hyperparameter tuning process was trained on the entire training data using the optimal hyperparameters. Its performance was then evaluated on the test data to obtain the final accuracy.

Table 6.4: Algorithmic Design of CNN (Self-Composed)

Ensemble Deep Learning Models:

The Simple Average, Voting, and Stacking methods are used to generate three distinct ensemble models. The ensemble model with the best accuracy is chosen to be the best performing super-learner model.

1. The Simple Average method:

The simple average ensemble method is a straightforward and easy technique to implement and understand that leverages the collective knowledge of multiple models to reduce overfitting. By doing this, it can improve predictive performance without requiring complex algorithms or training processes.

Prediction Aggregation	For each instance in the test dataset, predictions are obtained from each base model.
Averaging	The predictions from all base models are averaged across each instance. This averaging is performed element-wise if the base models are producing probabilities or can be rounded to obtain binary predictions if needed.

Final Prediction	Since this is a binary classification task, the averaged probabilities are rounded to the nearest integer (0 or 1) to obtain the final ensemble prediction.

Table 6.5: Algorithmic Design of Simple Average (Self-Composed)

2. Voting method:

By combining predictions from multiple base models trained on different subsets of data or using different algorithms, the voting ensemble method is able to capture a broad range of patterns in the data.

Prediction Aggregation	For each instance in the test dataset, predictions are obtained from each base model.
Voting	The predictions are rounded to 0 or 1, representing the class labels. The voting ensemble then determines the final prediction for each instance based on the majority vote of the 3 individual base models. For example, if the majority of base models predict class 1 for a particular instance, the ensemble prediction for that instance will be class 1.
Final Prediction	The final predictions obtained from the voting process are used as the ensemble's prediction for the test dataset.

Table 6.6: Algorithmic Design of Voting (Self-Composed)

3. Stacking method:

A prediction is produced for each of the base models. A meta-feature matrix is then produced by amalgamating the individual predictions together. The prediction of each base model for every data point is represented by a matrix called the meta-feature matrix. After that, a meta-model trains itself using this meta-feature matrix to get the final prediction.

Design	
Input Layer	The input to the stacking classifier consists of the predictions made by the base models on the training data.
Meta Model Architecture	The meta model is implemented as a feedforward neural network (FNN) with two hidden layers.

	- The first hidden layer has a variable number of units (neurons) and uses the ReLU activation function. - A dropout layer is applied after the first hidden layer to avoid overfitting, with the dropout rate as a hyperparameter that gets tuned during the optimization. - The second hidden layer also has a variable number of units and uses the ReLU activation function. - Another dropout layer is applied after the second hidden layer. - The output layer consists of a single neuron with a sigmoid activation function, to be suitable for binary classification tasks.
Loss Function	Binary cross-entropy loss function is used, which is commonly used for binary classification problems.
Optimizer	The Adam optimizer is used to optimize the model's weights during training.
Hyperparameter Tuning	
Hyperparameters Tuned	Number of units and dropout rates in the hidden layers, and the learning rate for the optimizer.
Search Space	The search space for each hyperparameter is defined using appropriate ranges and step sizes.
Objective	The objective of the hyperparameter tuning process was to maximize the validation accuracy of the model.
Optimization Algorithm	RandomSearch algorithm is used to randomly sample hyperparameter combinations from the search space and evaluate their performance on the validation data.
Training and Evaluation	The best model found during the hyperparameter tuning process is trained on the entire training data using the optimal hyperparameters.

Table 6.7: Algorithmic Design of Stacking (Self-Composed)

6.4.4 UI Design

The low fidelity UI design for the prototype is given below.

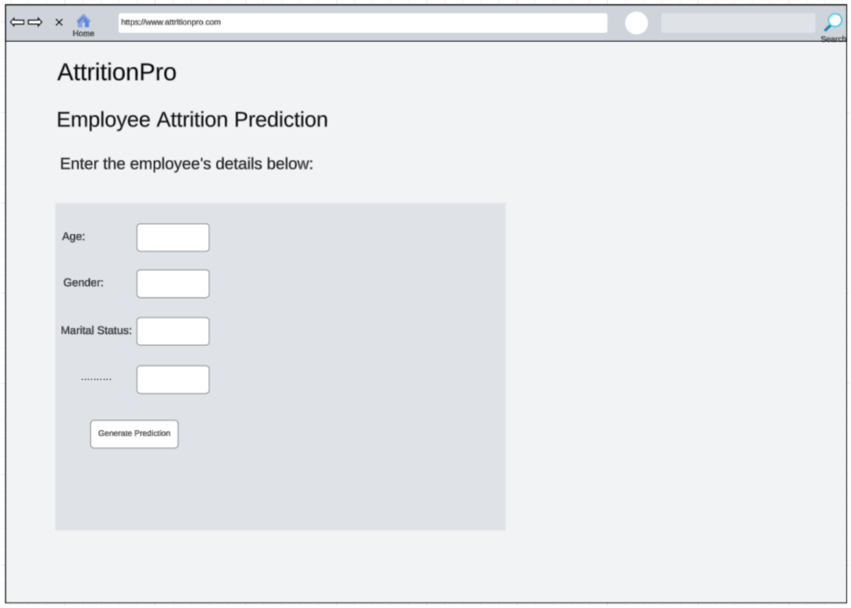
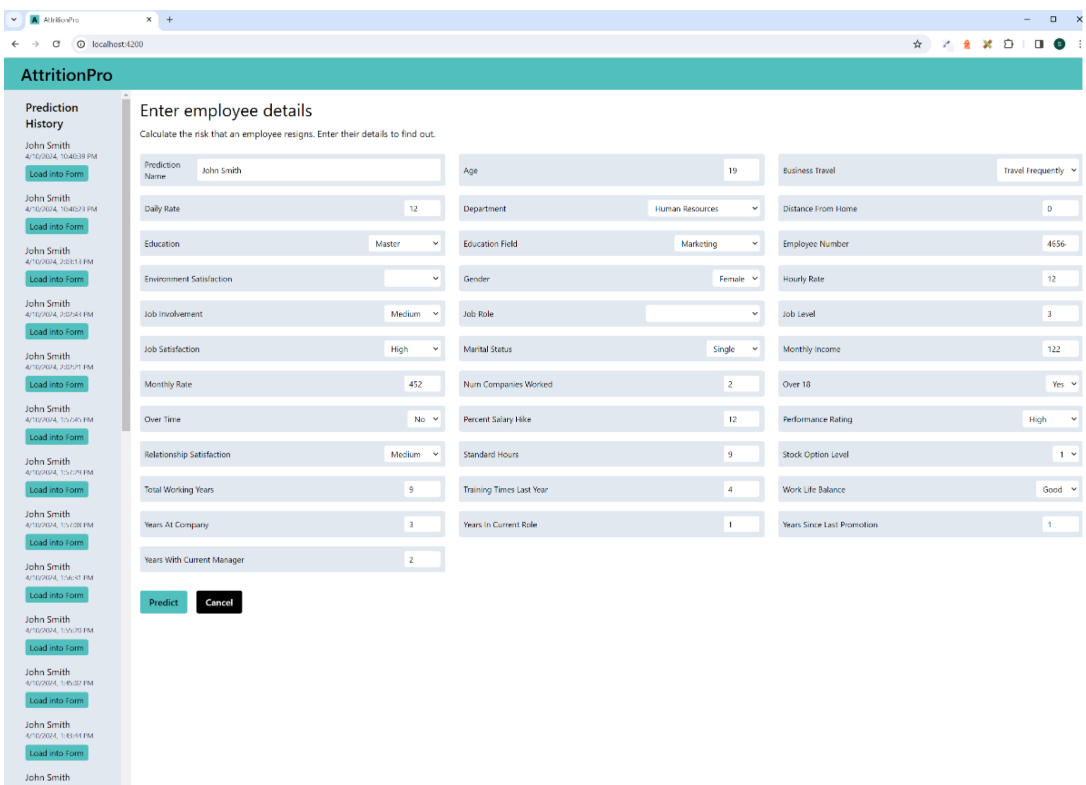
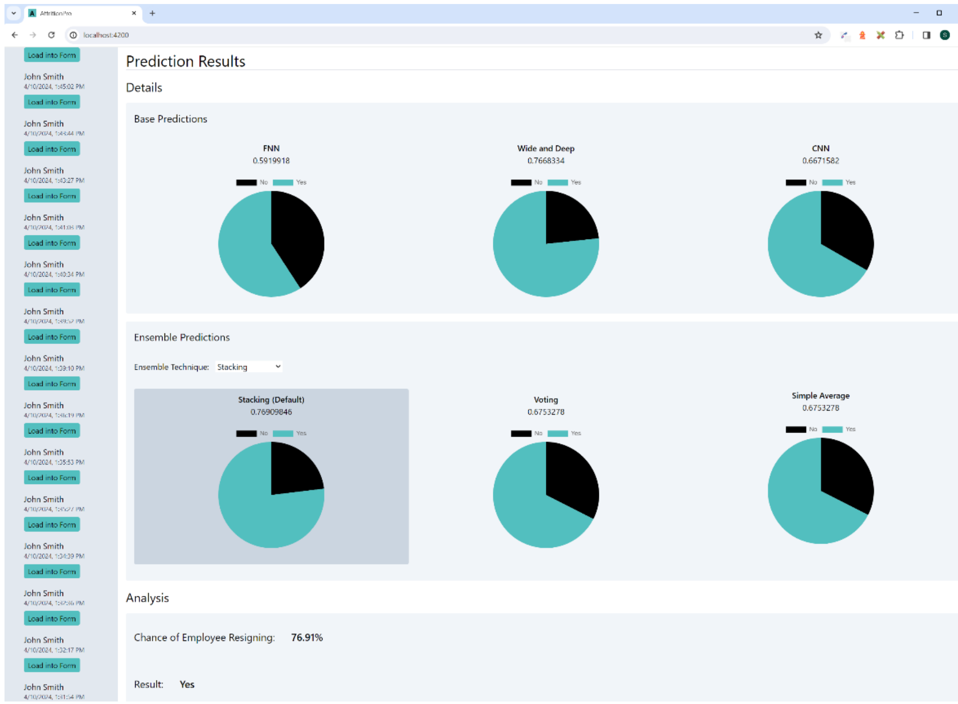


Figure 6.4: UI Design - Low Fidelity Wireframe (Self-Composed)





More screenshots of the actual design of the User Interface can be found in the **Appendix**.

6.4.5 System Process Flow Chart

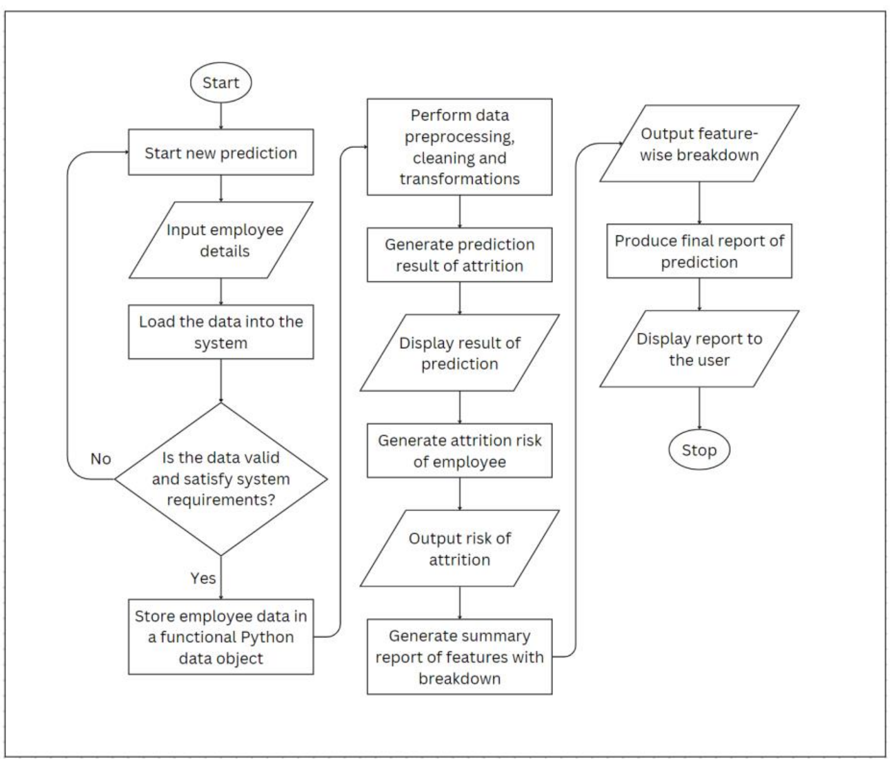


Figure 6.5: System Process Flow Chart (Self-Composed)

6.5 CHAPTER SUMMARY

The design chapter discussed the design goals that the project should meet and the system architecture, along with a discussion of the tiers involved. The algorithmic design of the system was also discussed and established along with several diagrams to identify the various Data flows and process flows involved.

CHAPTER 7: IMPLEMENTATION

7.1 CHAPTER OVERVIEW

This chapter will define the technologies selected for each aspect of the project with their justifications and discuss the implementation of the core functionality of the system. The steps involved and the code are examined in further detail.

7.2 TECHNOLOGY SELECTION

7.2.1 Technology Stack

The architecture of the system was divided into 3 layers: Data Tier, Logic Tier and finally, the Presentation Tier. The diagram below shows the technology selection for each of the layers of AttritionPro.

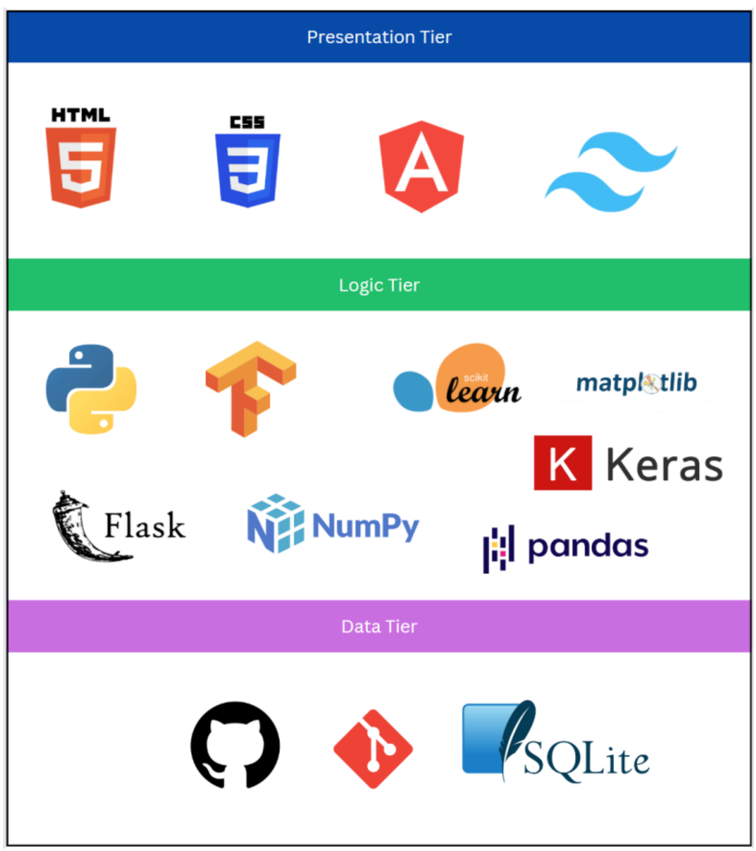


Figure 7.1: Technology Stack (Self-Composed)

7.2.2 Dataset Selection

The dataset selected is an open-source dataset created by IBM Scientists to represent various features of 1470 employees.

The data requirements of the system were established by referring to researchers' findings on factors that most affect employee attrition: age, education level, job satisfaction, monthly income, salary increment percentage, work life balance, standard hours in a working day, time since last promotion, relationship status, years at company.

The dataset chosen aligns with the data requirements of the project, due to it having the following columns:

Age, Attrition, BusinessTravel, DailyRate, Department, DistanceFromHome, Education, EducationField, EmployeeCount, EmployeeNumber, EnvironmentSatisfaction, Gender, HourlyRate, JobInvolvement, JobLevel, JobRole, JobSatisfaction, MaritalStatus, MonthlyIncome, MonthlyRate, NumCompaniesWorked, Over18, OverTime, PercentSalaryHike, PerformanceRating, RelationshipSatisfaction, StandardHours, StockOptionLevel, TotalWorkingYears, TrainingTimesLastYear, WorkLifeBalance, YearsAtCompany, YearsInCurrentRole, YearsSinceLastPromotion, YearsWithCurrManager

7.2.3 Development Frameworks

Development Framework	**Justification for Selection**
Flask	Flask was chosen as a Python web framework due to it's inherent lightweight and simple nature (Ghimire, 2020). As with the requirements of this project, Flask was identified as the better alternative over Django and FastAPI.
Angular	Angular was the selected front-end framework. It has high performance and optimized for developing dynamic web applications. Due to it being an opinionated framework, it provides more built-in functionalities and services, reducing the number of external libraries needed to be used.

Table 7.1: Selection of Development Frameworks (Self-Composed)

7.2.4 Programming Languages

Programming Language	Justification for Selection
Python	Python is a flexible language that can be applied to many different tasks, such as machine learning, data analysis, and web development. It offers several different libraries and frameworks, including NumPy for numerical operations, PyMC3 for probabilistic programming, and pandas for data analysis and statistics (Raschka, et al., 2020). These libraries are essential for ML algorithm implementation. Additionally, Python is widely utilized in many different industries and has a vibrant community. As a result, a multitude of tools, packages, and resources will be available, all of which will be helpful while developing.
JavaScript	It is excellent at building dynamic and interactive online apps and comes with a plethora of frameworks and modules that make front-end development easier, such as React, Angular, and Vue.js.

Table 7.2: Selection of Programming Languages (Self-Composed)

7.2.5 Libraries

Library	Justification for Selection
Pandas	Because the Pandas library is a potent Python data manipulation toolkit, it was used. It offers adaptable data structures for working with organized information. It provides operations and data structures for working with time series data and numerical tables. Large dataset handling and analysis are two areas in which it excels.
NumPy	NumPy provides support for large, multi-dimensional arrays and matrices and other data-type objects. It makes doing high-level mathematical functions on these data structures easier. (Oliphant, 2006)
Matplotlib	Matplotlib is a Python plotting library. It supports many types of interactive and non-interactive graphs. (Tosi, 2009).

Seaborn	Seaborn is a data visualization library based on matplotlib. It is used for drawing statistical graphics (Waskom, 2021). It also provides good integration with Pandas.
Pickle	A Python object structure can be serialized and de-serialized using the Pickle module. Stated differently, it transfers a Python object to a character stream and the other way around. It is utilized when you wish to save your program's state for subsequent loading.
Scikit-learn	Scikit-learn provides implementations of various classification, regression and clustering algorithms including SVMs, random forests, gradient boosting and k-means. (Pedregosa, et al., 2011).

Table 7.3: Selection of Libraries (Self-Composed)

7.2.6 IDE

IDE	Justification for Selection
Visual Studio Code	VS Code is lightweight but powerful and supports multiple programming languages and platforms, including Python, and offers useful extension features.
Google Colab	Google Colab can be used to write and execute Python code. The ML component of this project is conducted entirely through Colab.

Table 7.4: Selection of IDEs (Self-Composed)

7.2.7 Summary of Technology Selection

Component	Tools
Development Frameworks	Python, JavaScript
Programming Languages	Flask, Angular
Libraries	Pandas, NumPy, Matplotlib, Seaborn, Pickle, Scikit-learn
IDEs	Visual Studio Code, Google Colab
Version Control	GitHub, Git, Google Drive, OneDrive

Table 7.5: Summary of Technology Selection (Self-Composed)

7.3 IMPLEMENTATION OF CORE FUNCTIONALITY

7.3.1 Data Preprocessing and Pipelining

The numerical and categorical columns were identified and preprocessed using 2 different pipelines. The numeric features were processed using a SimpleImputer transformer to fill in missing values with each feature's median value. Afterwards, the features were scaled by removing the mean value and scaling each to unit variance using the StandardScaler.

Identification of categorical features is done by handling missing values, encoding them into integer and scaling their features. First, the SimpleImputer was used to deal with the missing values. To convert the categorical values into binary, the OneHotEncoder was used. After using StandardScaler on the data, the ColumnTransformer was used to apply the pipelines and preprocess the data.

Data Preprocessing and Pipelining

```
X_train=project_data.drop(columns=["Attrition"])
y_train=project_data["Attrition"]

X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.3)
```

```
print('Train dataset shape:',X_train.shape)
print('Test dataset shape', X_test.shape)

print('Train dataset rows: ', len(X_train))
print('Test dataset rows: ', len(X_test))
```

```
Train dataset shape: (1029, 34)
Test dataset shape (441, 34)
Train dataset rows:  1029
Test dataset rows:  441
```

```
numeric_columns = X_train.select_dtypes(exclude='object').columns
print(numeric_columns)
print('*'*100)
categorical_columns = X_train.select_dtypes(include='object').columns
print(categorical_columns)
```

```
Index(['Age', 'DailyRate', 'DistanceFromHome', 'Education', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobSatisfaction', 'MonthlyIncome',
       'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
       'PerformanceRating', 'RelationshipSatisfaction', 'StandardHours',
       'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
       'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')
****************************************************************************************************
Index(['BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole',
       'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
```

```
numeric_features = Pipeline([
    ('handlingmissingvalues',SimpleImputer(strategy='median')),
    ('scaling',StandardScaler(with_mean=True))
])

print(numeric_features)
print('*'*100)

categorical_features = Pipeline([
    ('handlingmissingvalues',SimpleImputer(strategy='most_frequent')),
    ('encoding', OneHotEncoder()),
    ('scaling', StandardScaler(with_mean=False))
])

print(categorical_features)

processing = ColumnTransformer([
    ('numeric', numeric_features, numeric_columns),
    ('categorical', categorical_features, categorical_columns)
])

processing
```

7.3.2 Creating the Base Models

The implementation of creating the three base models was done as follows. The algorithmic design, justification of the layers and explanation of the models can be found in **6.4.3 Algorithmic Design.**

```
def create_fnn(hp):
    model = Sequential()
    model.add(Dense(units=hp.Int('units_1', min_value=32, max_value=512, step=32),
                    activation='relu', input_dim=input_dim))
    model.add(Dropout(rate=hp.Float('dropout_1', min_value=0.0, max_value=0.5, step=0.1)))

    model.add(Dense(units=hp.Int('units_2', min_value=32, max_value=256, step=32),
                    activation='relu'))
    model.add(Dropout(rate=hp.Float('dropout_2', min_value=0.0, max_value=0.5, step=0.1)))

    model.add(Dense(1, activation='sigmoid'))

    model.compile(optimizer=Adam(learning_rate=hp.Choice('learning_rate', values=[1e-2, 1e-3, 1e-4])),
                  loss='binary_crossentropy',
                  metrics=['accuracy'])
    return model
```

```
def create_wide_and_deep_model(hp):
    input_dim = X_train.shape[1]

    wide_inputs = Input(shape=(input_dim,))
    deep_inputs = Input(shape=(input_dim,))

    wide_layer = Dense(units=hp.Int('wide_units', min_value=32, max_value=256, step=32),
                       activation='relu')(wide_inputs)

    deep_layer = Dense(units=hp.Int('deep_units_1', min_value=32, max_value=256, step=32),
                      activation='relu')(deep_inputs)
    deep_layer = Dense(units=hp.Int('deep_units_2', min_value=16, max_value=128, step=16),
                      activation='relu')(deep_layer)

    merged_layer = concatenate([wide_layer, deep_layer])

    output = Dense(1, activation='sigmoid')(merged_layer)

    model = Model(inputs=[wide_inputs, deep_inputs], outputs=output)

    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

    return model
```

```python
def create_cnn_model(hp):
    input_shape = (X_train_reshaped.shape[1], X_train_reshaped.shape[2])

    model = Sequential()

    model.add(Conv1D(filters=hp.Int('filters', min_value=16, max_value=64, step=16),
                     kernel_size=hp.Int('kernel_size', min_value=2, max_value=5, step=1),
                     padding='same',
                     activation='relu',
                     input_shape=input_shape))

    model.add(MaxPooling1D(pool_size=2))
    model.add(Flatten())

    model.add(Dense(128, activation='relu'))
    model.add(Dropout(0.2))

    model.add(Dense(64, activation='relu'))
    model.add(Dropout(0.2))

    model.add(Dense(1, activation='sigmoid'))

    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

    return model
```

7.3.3 Training the Base Models

Hyperparameter Tuning was performed for all of the models and the most optimal hyperparameters were identified. The base models were then trained according to these most optimal hyperparameters to get the best accuracy. Explanations and details about the hyperparameter tuning performed can be found at **6.4.3 Algorithmic Design.**

```
input_dim = X_train.shape[1]

fnn_tuner = kt.RandomSearch(
    create_fnn,
    objective='val_accuracy',
    max_trials=10,
    directory='my_dir',
    project_name='fnn_hyperparameter_tuning')

fnn_tuner.search(X_train, y_train, epochs=100, validation_data=(X_test, y_test))

best_fnn_model = fnn_tuner.get_best_models(num_models=1)[0]
best_fnn_hyperparameters = fnn_tuner.get_best_hyperparameters(num_trials=1)[0]

best_fnn_model.summary()

best_fnn_model.fit(X_train, y_train, epochs=100, batch_size=32, validation_data=(X_test, y_test))

best_fnn_accuracy = best_fnn_model.evaluate(X_test, y_test, verbose=0)[1]
print("Best FNN Model Accuracy:", best_fnn_accuracy)
print("Best Hyperparameters:", best_fnn_hyperparameters)
```

```
wd_tuner = RandomSearch(
    create_wide_and_deep_model,
    objective='val_accuracy',
    max_trials=10,
    directory='my_dir',
    project_name='wide_and_deep_hyperparameter_tuning'
)

wd_tuner.search_space_summary()
wd_tuner.search([X_train, X_train], y_train, epochs=100, batch_size=32, validation_data=([X_test, X_test], y_test))

best_wd_model = wd_tuner.get_best_models(num_models=1)[0]
best_wd_hyperparameters = wd_tuner.get_best_hyperparameters(num_trials=1)[0]

best_wd_model.fit([X_train, X_train], y_train, epochs=100, batch_size=32, validation_data=([X_test, X_test], y_test))

best_wide_and_deep_accuracy = best_wd_model.evaluate([X_test, X_test], y_test, verbose=0)[1]
print("Best Wide & Deep Model Accuracy:", best_wide_and_deep_accuracy)
```

```
cnn_tuner = kt.Hyperband(create_cnn_model,
                         objective='val_accuracy',
                         max_epochs=10,
                         factor=3,
                         directory='my_dir',
                         project_name='intro_to_kt')

cnn_tuner.search(X_train_reshaped, y_train, epochs=100, validation_data=(X_test_reshaped, y_test))

best_cnn_hyperparameters=cnn_tuner.get_best_hyperparameters(num_trials=1)[0]

best_cnn_model = cnn_tuner.hypermodel.build(best_cnn_hyperparameters)
cnn_history = best_cnn_model.fit(X_train_reshaped, y_train, epochs=100, validation_data=(X_test_reshaped, y_test))

best_cnn_accuracy = cnn_history.history['val_accuracy'][-1]
print("Best CNN Model Accuracy:", best_cnn_accuracy)
```

7.3.4 Implementing Ensemble Techniques

The predictions from the base models were combined to implement all 3 of the ensemble techniques: Stacking, Voting and Simple Averaging.

```
ensemble_methods = ['stacking', 'voting', 'simple_average']

for i, method in enumerate(ensemble_methods, 1):

    print(f"\n-------- Ensemble Method: {method} ----------")
    ensemble_predictions = ensemble_predict(models, X_test, X_train, method)
    ensemble_accuracy = accuracy_score(y_test, ensemble_predictions)
    if method == 'stacking':
        stacking_accuracy = ensemble_accuracy
    elif method == 'voting':
        voting_accuracy = ensemble_accuracy
    elif method == 'simple_average':
        simple_average_accuracy = ensemble_accuracy
```

```
def ensemble_predict(models, X_test, X_train, method):
    test_predictions = []
    train_predictions = []

    for model in models:
        if isinstance(model, Sequential):
            X_test_input = X_test
            X_train_input = X_train
        elif isinstance(model, Model):
            X_test_input = (X_test, X_test)
            X_train_input = (X_train, X_train)
        else:
            raise ValueError("Invalid model type provided.")

        test_predictions.append(model.predict(X_test_input))
        train_predictions.append(model.predict(X_train_input))

    if method == 'simple_average':
        test_predictions = np.array(test_predictions)
        return np.round(np.mean(test_predictions, axis=0)).astype(int)
    elif method == 'voting':
        test_predictions = np.array(test_predictions)
        class_votes = np.round(np.mean(test_predictions, axis=0)).astype(int)
        final_prediction = np.where(np.sum(class_votes, axis=1) > test_predictions.shape[0] / 2, 1, 0)
        return final_prediction
    elif method == 'stacking':
        meta_X_train = np.concatenate(train_predictions, axis=1)
        meta_X_test = np.concatenate(test_predictions, axis=1)

        best_hps = tune_metamodel(meta_X_train, y_train)
        meta_model = create_metamodel_fnn(best_hps)
        meta_model.fit(meta_X_train, y_train, epochs=100, batch_size=32, validation_split=0.2)
        model_name = 'Stacking'
        prepare_evaluation(meta_model, model_name, X_test_svd, y_test_encoded)
        df_ensemble_metrics = pd.DataFrame(ensemble_metrics).T
        print('df_ensemble_metrics')
        print(df_ensemble_metrics)
        print()
        meta_model.save("stacking_ensemble_model.h5")
        meta_predictions = meta_model.predict(meta_X_test)
        return np.round(meta_predictions).astype(int)
    else:
        raise ValueError("Invalid ensemble method provided.")
```

Hyperparameter Tuning was performed for the meta-model built for the stacking method. Explanations and details about the hyperparameter tuning performed can be found at **6.4.3 Algorithmic Design.**

```
def create_metamodel_fnn(hp):
    model = Sequential()
    model.add(Dense(units=hp.Int('units_1', min_value=32, max_value=512, step=32),
                    activation='relu',
                    input_dim=input_dim))
    model.add(Dropout(rate=hp.Float('dropout_1', min_value=0.2, max_value=0.5, step=0.1)))
    model.add(Dense(units=hp.Int('units_2', min_value=32, max_value=256, step=32),
                    activation='relu'))
    model.add(Dropout(rate=hp.Float('dropout_2', min_value=0.2, max_value=0.5, step=0.1)))
    model.add(Dense(1, activation='sigmoid'))

    model.compile(optimizer=Adam(learning_rate=hp.Choice('learning_rate', values=[1e-2, 1e-3, 1e-4])),
                  loss='binary_crossentropy',
                  metrics=['accuracy'])
    return model

def tune_metamodel(X_train, y_train):
    tuner = RandomSearch(
        create_metamodel_fnn,
        objective='val_accuracy',
        max_trials=10,
        executions_per_trial=3,
        directory='stacking_hyperparameter_tuning',
        project_name='stacking')

    tuner.search(X_train, y_train, epochs=100, validation_split=0.2)
    best_hps = tuner.get_best_hyperparameters(num_trials=1)[0]
    return best_hps
```

7.4 CHAPTER SUMMARY

This chapter covered the initial implementation of the core functional requirements of the project. The technologies used in every aspect of the implementation were discussed and justified. The implementation of the basic functionality of AttritionPro developed up until this point was discussed in the chapter. Improvements and refinements will be done for the upcoming MVP submission based on this initial development of the system.

CHAPTER 8: TESTING

8.1 CHAPTER OVERVIEW

In this chapter, the author discusses the established goals of testing, the criteria set and the test results, in terms of confusion matrices, AUC ROC curves, and other metrics such as F1 score, recall, precision and accuracy. The features and functionalities of the system will be tested, along with non-functional requirements.

8.2 OBJECTIVES AND GOALS OF TESTING

The testing goals and objectives for this project are as follows:

- Ensure that the base models and ensemble model of the system are performing as required.
- Evaluate whether the suggested algorithmic design and implementation improves performance and accuracy of the model.
- Benchmark the author's solution against other research and systems that have been implemented in the domain to gauge how it matches up to industry standards.
- Functional and Non-functional requirements of system should be fulfilled.
- The 'Must' have and 'Should' have functionalities of the system must be fulfilled according to the MoSCoW prioritization levels.
- Test the system's performance, accuracy, functionalities, and design to validate if all aspects of the developed system are functioning as proposed.

8.3 TESTING CRITERIA

The testing conducted will follow 2 specific criterions in order to fully ensure that the implemented system aligns with the proposed system.

1. Functional Quality – The author will be looking at the technical implementation of the system and test whether it aligns with and fully encapsulates the functionalities specified in the requirements.
2. Structural Quality – The structural integrity of the developed system, it's performance and other non-functional requirements will be tested and evaluated by the author.

8.4 MODEL TESTING

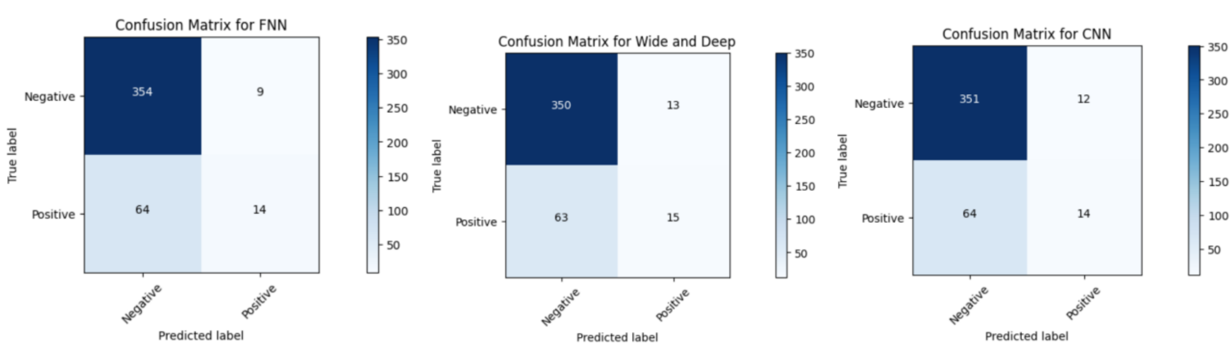
8.4.1 Confusion Matrix

The results of the system were analyzed by preparing confusion matrices and roc curves. The accuracies of the final ensemble models created are as follows.

Base Model Results:

	Accuracy	F1 Score	Precision	Recall	Specificity	Misclassification Rate
FNN	82.99%	0.271845	0.560000	0.179487	0.969697	0.170068
Wide and Deep	82.77%	0.283019	0.535714	0.192308	0.964187	0.172336
CNN	82.77%	0. 0.2830 19	0.535714	0.179487	0.966942	0.172336

Confusion Matrices for the base models:



Ensemble Model Results:

	Accuracy	F1 Score	Precision	Recall	Specificity	Misclassification Rate
Voting	82.99%	0.2653061	0.500000	0.179765	0.512396	0.1610
Simple Average	82.99%	0.285714	0.555556	0.192307	0.966942	0.1610

Stacking	**83.45%**	**0.365217**	**0.567568**	**0.269231**	**0.955923**	**0.165533**

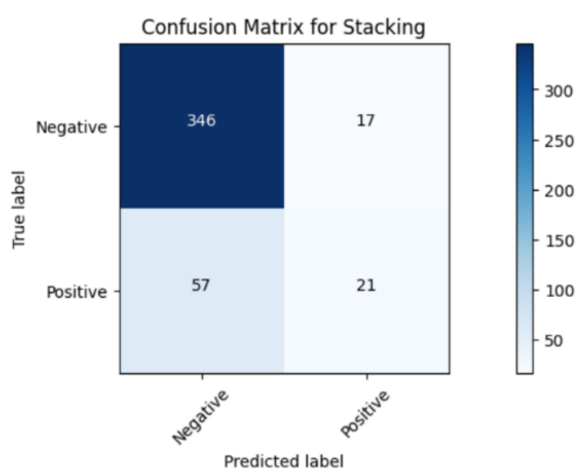
Among the different models, Stacking achieved the highest accuracy at 83.45%, and other metrics such as F1 score, accuracy and improvement are also quite effective compared to the other models. This shows that the Stacking ensemble method is quite effective in improving prediction performance on multiple models.

On the other hand, the Voting Method and the Simple Average Method also showed accuracies of 83.90% and 82.99%, respectively. However, compared to stacking, their performance on other metrics such as precision, recall, and misclassification rate are slightly different.

Overall, the results show that ensembling (especially Stacking) has good potential to improve the accuracy and efficiency of attrition prediction.

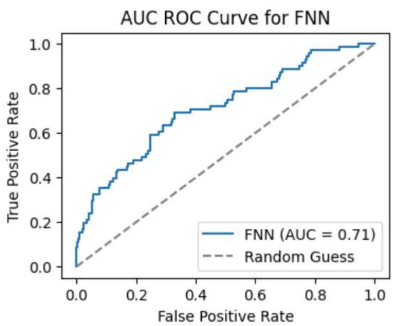
The Stacking method was chosen as the best performing ensemble technique out of the 3 methods. Therefore, the final developed system by default used the Stacking ensemble method.

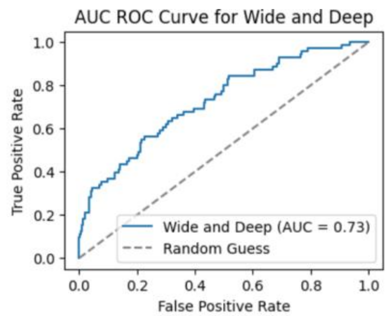
Confusion Matrix for the Stacking Model:

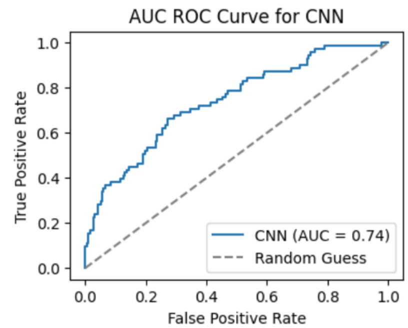


8.4.2 AUC/ROC Curve

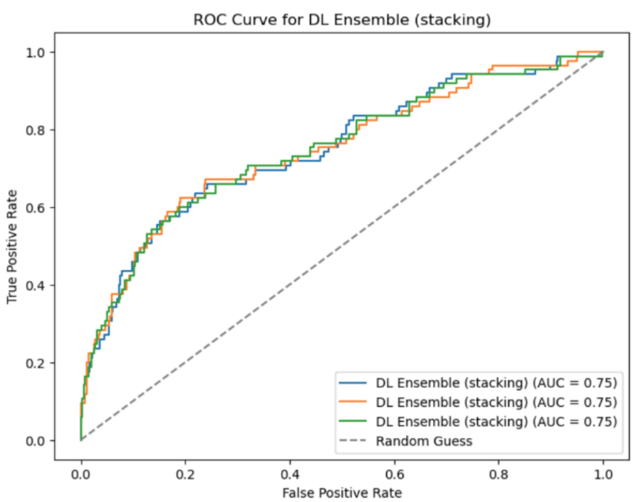
The AUC/ROC curves for the base models are given below. More metrics and test results can be found in **Appendix D: Testing**.







The AUC/ROC curve for the Stacking Model:



An AUC of 0.75 indicates that the model is on average 75% effective in distinguishing between positive and negative classes. This means that for two different data points the model has a 75% chance of splitting or ranking, and positive cases have a higher probability of prediction than negative.

8.5 BENCHMARKING

	Accuracy	F1 Score	Precision	Recall	Specificity	Misclassification Rate
Voting	82.99%	0.2653061	0.500000	0.179765	0.512396	0.1610
Simple Average	82.99%	0.285714	0.555556	0.192307	0.966942	0.1610
Stacking	**83.45%**	**0.365217**	**0.567568**	**0.269231**	**0.955923**	**0.165533**

The ensemble models developed here in this project match and in some cases, outperform other existing systems in terms of accuracy, F1 score, specificity and other metrics. A study by Raza et al., 2022 was able to achieve an accuracy of 80.5% for an ML approach, and similarly, a deep NN approach by Al-Darraji et al., 2021 achieved an accuracy of 81.5%. The ensemble models developed in this project were able to outperform both of these approaches.

Moreover, the F1 score, which is a harmonic mean of precision and recall, is a critical metric for balancing these two aspects. Our models achieve higher F1 scores compared to other systems. For example, the F1 scores of all of our models are higher than the F1 score of 0.25 reported by a study on employee attrition prediction using ML and ensemble methods. (Zolfani et al., 2021).

In terms of specificity, the AttritionPro models achieve higher specificity rates. This indicates a lower rate of false positives. The Stacking model's specificity rate of 0.955923 is significantly higher than the specificity rate of 0.92 reported by a study on an improved machine learning-based framework for employee attrition prediction. (Zolfani et al., 2021).

The high performance of the AttritionPro models, as evidenced by the statistical comparison above shows the overall effectiveness and suitability of ensemble techniques in predicting employee resignations. This justifies the choice of ensemble methods for predicting employee attrition, highlighting the scope of their potential for being valuable tools for HR analytics and building employee retention strategies.

8.6 FUNCTIONAL TESTING

ID	User Action	Expected Outcome	Actual Outcome	Test Result
FR1	User inputs a certain employee’s details to make a prediction of their attrition.	Employee details are retrieved and validated.	Employee details are retrieved and validated.	Pass
FR2	User clicks on Predict button.	Prediction results should be generated for the input	Prediction results should be generated for the	Pass

		data, showing the attrition of that employee.	input data, showing the attrition of that employee.	
FR3	User clicks on Predict button.	The results of the prediction should be displayed to the user in a user-friendly, intuitive way.	The results of the prediction should be displayed to the user in a user-friendly, intuitive way.	Pass
FR4	User views the generated results.	The risk of attrition for an employee should be generated from the data input by the user.	The risk of attrition for an employee should be generated from the data input by the user.	Pass
FR5	User views the generated results.	The user should be able to view the generated risk of attrition in a presentable manner.	The user should be able to view the generated risk of attrition in a presentable manner.	Pass
FR6	User views the prediction history	The prediction history could be saved, and the most recent predictions could be shown to the user on the GUI.	The prediction history is saved, and the most recent predictions is shown to the user on the GUI.	Pass
FR8	User interacts and engages with AttritionPro.	The system would have an intuitive GUI for the user to interact with the system.	The system would have an intuitive GUI for the user to interact with the system.	Pass

8.7 MODULE AND INTEGRATION TESTING

Module	**Input**	**Expected Outcome**	**Actual Outcome**	**Test Result**
Exploratory Data Analysis	IBM Employee Attrition dataset	Identify the different types of features, conduct correlation analysis and identify the factors that most affect attrition.	Identify the different types of features, conduct correlation analysis and identify the factors that most affect attrition.	Pass
Data Preprocessing	IBM Employee Attrition dataset	Cleaned and preprocessed data	Cleaned and preprocessed data	Pass
Feature Selection	Preprocessed data from previous module	Selected features for modeling	Selected features for modeling	Pass
Model Training	Selected features from previous module	Trained model with specified hyperparameters	Trained model with specified hyperparameters	Pass
Model Evaluation	Trained model, Test dataset	Evaluation metrics (accuracy, precision, recall)	Evaluation metrics (accuracy, precision, recall)	Pass
Hyperparameter Tuning	Selected features, Train set	Optimized hyperparameters	Optimized hyperparameters	Pass
Generating prediction using Super Learner	User input employee data	Generating prediction using Super Learner to get chance of employee resigning	Generating prediction using Super Learner to get chance of employee resigning	Pass
Generating risk level	User input employee data	Generating risk level using prediction from previous module	Generating risk level using prediction from previous module	Pass

8.8 NON-FUNCTIONAL TESTING

8.8.1 Accuracy Testing

Accuracy testing was conducted with all of the base classifiers individually and the super learner ensemble model as well. The results of the accuracy testing are discussed in **8.4 Model Testing.**

8.8.2 Performance Testing

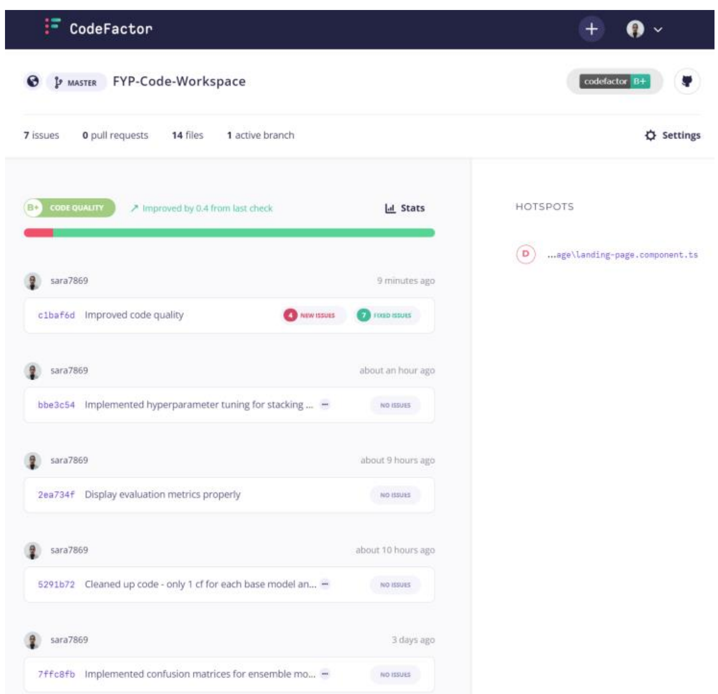
Results of Performance Testing can be found in the Appendix. Refer to the **Appendix D: Testing**.

8.8.3 Usability Testing

Results of Usability Testing can be found in the Appendix. Refer to the **Appendix D: Testing**.

8.8.4 Maintainability Testing

CodeFactor was used to test the maintainability of the code. The entire implementation of the system receives a B+ grading on CodeFactor for maintainability.



8.9 LIMITATIONS OF THE TESTING PROCESS

Although the experimental procedures performed in this study were designed to evaluate all procedures developed, there are still some limitations that must be acknowledged:

Limitations in data variability: The testing process depended on much of IBM's Employee Attrition dataset. Although this data are widely used in research on prediction, their applicability may be limited to certain organizations. Therefore, the generalizability of the test to other data sets or markets may be limited.

Considering model robustness: The testing method demonstrates that the model design (e.g. base classifiers and ensemble methods) is robust to different data and different information. However, the performance of these models may vary depending on different data or actual deployment conditions.

Metrics: Evaluation of model performance often focuses on indicators such as accuracy, precision, recall, and area under the ROC curve. While these metrics provide a good idea of the model's performance, they may not capture all aspects of the model's behavior, such as interpretation or bias.

Hyperparameter Tuning: Although hyperparameter tuning is done to improve the performance of the model, choosing the hyperparameter search space and tuning method will not search the entire solution space available. Therefore, better hyperparameter configurations might be identified through research.

Limitations in Non-Functional Testing: When performing tests such as performance and usability testing, the evaluation of these tests may not include all factors related to system performance and user experience.

Testing Environment: Testing is done in a controlled environment, which may not fully represent the complexity and uncertainty of a real deployment. Factors such as data drift, model degradation over time, or unexpected behavior under production load will not be clearly identified.

Limitations in functional testing: While functional testing covers basic user interaction and physical functionality, it may not include all possible outcomes for clients or cases. Some features or user functions may not be fully tested, leading to possible misses or undetected issues.

8.10 CHAPTER SUMMARY

The testing chapter of this study comprehensively evaluated the developed attrition prediction system through various testing methodologies. It included both model testing, focusing on the performance of base classifiers and ensemble techniques, and functional testing, ensuring the correct functioning of system features. Additionally, non-functional testing was conducted to assess aspects such as accuracy, performance, usability, and maintainability. The testing process provides better insight into the performance of the system, highlights strengths, limitations, and areas for improvement. By testing the method on multiple aspects, this study provides a robust and reliable solution for predicting employee turnover.

CHAPTER 9: EVALUATION

9.1 CHAPTER OVERVIEW

This chapter presents the evaluation process and methods used to evaluate the effectiveness and efficiency of the system. It describes the methodology used for evaluation, the selection process of evaluators, and the inherent limitations of the evaluation method. Combining various analysis results with recommendations from domain experts and end users, this chapter provides a comprehensive overview of the evaluation process used in this study.

9.2 EVALUATION METHODOLOGY AND APPROACH

The evaluation methodology used in this research project combines quantitative and qualitative methods to evaluate the implemented system. Quantitative evaluation is based on the results of the testing phase. These include the measurement parameters such as accuracy, precision, recall and F1 score. The quality assessment also depends on the feedback received from domain experts and end users. Expert and user comments and feedback were analyzed to gain insight into the system's performance and potential areas for improvement. In addition, a video showing the system's functionalities and capabilities was created and distributed to the evaluators as a demo.

User Manual: **User Manual.**

The video and the evaluation data can be accessed at the following link: **AttritionPro: Presentation.**

9.3 EVALUATION CRITERIA

Id	Criteria	Purpose
EC1	Problem Definition and Novelty	Evaluate the significance and novelty of the identified problem in the context of employee attrition prediction. Determine whether the research addresses a unique and substantial gap in the field.
EC2	Technical and Theoretical Review	Assess the quality and effectiveness of the literature review conducted for the study, focusing on both theoretical and

		technical aspects. Determine the comprehensiveness and relevance of the literature to the research topic.
EC3	Scope and Depth of Research	Determine the adequacy of the research scope, depth, and complexity in addressing the identified problem. Identify any constraints or limitations that may impact the accuracy and applicability of the research findings.
EC4	System Development and Technological Stack	Evaluate the suitability of the chosen technological stack for developing the attrition prediction system. Assess the functionality and effectiveness of the prototype in addressing the research objectives.
EC5	Data Utilization and Model Training	Assess the appropriateness of the data used for model training, including representativeness, preprocessing, and mitigation of biases. Determine the efficacy of the data in training accurate and reliable prediction models.
EC6	Performance Analysis and Results Evaluation	Evaluate the quantitative performance metrics of the prediction models to determine their effectiveness in predicting employee attrition. Assess the significance of the results in contributing to academic knowledge and practical applications.

Table 9.1: Evaluation Criteria (Self-Composed)

9.4 SELF-EVALUATION

Id	Criteria	Evaluation
EC1	Problem Definition and Novelty	The author believes that the problem identified is significant and novel in the context of research conducted in the domain. This research addresses a unique and major gap in the field by providing a new approach to attrition prediction.
EC2	Technical and Theoretical Review	In the author's opinion, the literature review conducted for the study is of high quality and effectively covers both theoretical and technical aspects related to employee attrition prediction. The review is comprehensive, relevant, and provides a solid foundation for the research topic.

EC3	Scope and Depth of Research	There's room for improving the scope, depth, and complexity of the research. More functionalities and features can be implemented. However, the author had to limit it to a few priority features due to the time constraints of the project. The research made significant contributions to the body of knowledge and research field.
EC4	System Development and Technological Stack	The chosen technological stack for developing AttritionPro is appropriate and well-suited to the research objectives. The prototype demonstrates functionality and effectiveness in addressing the research objectives, although there is room for further refinement.
EC5	Data Utilization and Model Training	The data used for the training of the models is suitable and appropriate. Suitable preprocessing steps are taken to mitigate biases and to prepare the data for optimal training of the model.
EC6	Performance Analysis and Results Evaluation	The quantitative performance metrics of the prediction models demonstrate their effectiveness in predicting employee attrition. The results obtained contribute significantly to both academic knowledge and practical applications in the field. However, there may be areas for improvement in the performance analysis and results evaluation process in the future.

Table 9.2: Self-Evaluation (Self-Composed)

9.5 SELECTION OF THE EVALUATORS

Evaluators of two distinct domains were selected for the evaluation of the system as follows.

CAT ID	**Category**
1	Expert Opinion: Experts in Machine Learning, Deep Learning and Data Science were selected as the first category of evaluators.

2	Focus Group Testing: The end users of the system: HR personnel, HR managers and management level evaluators were selected to evaluate the utilization of the project.

Table 9.3: Selection of Evaluators (Self-Composed)

9.6 EVALUATION RESULTS

9.6.1 Expert Opinion

Prof. Chandana Gamage **Professor of Computer Science, University of Moratuwa**	
Overall impression	"From a technical standpoint, the system displays a well-engineered architecture, The preprocessing techniques and feature selection methods have been used to optimize the model's performance. The use of ensemble learning techniques enhances the predictive accuracy and reliability."
Effectiveness of model	"The chosen models exhibit strong predictive capabilities, leveraging a combination of feature engineering, model tuning, and ensemble techniques to enhance performance. Their adaptability to diverse datasets ensures reliable predictions across various organizational contexts."
Complexity of system	"AttritionPro exhibits a high level of complexity in its architecture for a final year project. The integration of various ML, DL models, data preprocessing pipelines, and backend infrastructure adds complexity. The system's modular design and well-defined components help manage this complexity effectively."
Novelty of system	"The attrition prediction model and its underlying architecture demonstrate a commendable level of novelty, especially in their approach to feature engineering and model selection. By using a combination of traditional statistical methods and modern DL algorithms, the system achieves a nuanced understanding of employee attrition factors."
Contribution to research domain	"The project's contribution to the research domain is significant, as it introduces novel methodologies and approaches to employee attrition prediction. By using advanced DL algorithms and data-driven techniques,

	the project advances HR analytics. The findings not only add to the existing body of knowledge but also provide insights for organizations seeking to mitigate attrition risks and optimize workforce management.”
Ms. Thinayani Gamage **Engineer - Technology at EvonSys**	
Overall impression	“AttritionPro seems to be a very well-designed, well-engineered system that I believe provides a lot of contribution to the domain because of it's very functional and useful features.”
Effectiveness of model	“DL Ensemble Methods are very good at predicting employee attrition. I believe they have the potential to really improve the domain of employee attrition prediction. They have been implemented well and with appropriate hyperparameter tunings, and data preprocessing as well.”
Complexity of system	“In my opinion, I believe this is very complex and difficult project to pull off. The proposed solution was to build several deep learning models and combine them in various ensemble modelling techniques. In my final year project, I used a few ML models and tried to combine them into one ensemble model. So, I can tell that this project was very complicated to design and implement but the student has done a great job and managed to deliver even more than what she had initially proposed.”
Novelty of system	“The models are quite novel. I personally have never come across wide and deep models in this domain before. And the ensembling techniques and hyperparameter tuning are also done to a great degree. This project has tried out a completely new architecture and design and has managed to build a system with really good prediction performance.”
Contribution to research domain	“This project seems to have made a huge contribution to the domain for being a 1 year undergraduate project. Most existing work only covers 1 or 2 DL models at a time and even, they are implemented only as base models. She has conducted a lot of thorough research, identified novel approaches and designed and implemented them. She has even managed to optimize

	them and get them to outperform existing models in the forefront of this domain.”
Mr. Udara Nilupul **Machine Learning Engineer, Ascentic**	
Overall impression	“As an expert in the field, I am impressed by the system's utilization of advanced predictive modeling techniques to forecast employee turnover. Including interpretability measures adds transparency to the predictive outcomes generated.”
Effectiveness of model	“Selected machine learning models have been shown to be effective in predicting employee turnover. Their ability to identify complex patterns in data and produce accurate predictions contributes to the HR decision-making process.”
Complexity of system	“AttritionPro seems to show a high level of complexity because it uses multifaceted architecture and advanced modeling techniques. Various data processing pipelines and model training frameworks are used which needs attention to detail and effort to do.”
Novelty of system	“From a technical aspect, the model and its architecture shows a considerable level of novelty. Using state-of-the-art ML algorithms, coupled with advanced feature engineering strategies, sets the system apart. And, adding to that, the use of ensemble techniques and model stacking methods further improves the model's novelty.”
Contribution to research domain	“This project contributes a lot to the research domain because it is advancing our understanding of the topic of employee attrition prediction. The student has used diverse and various deep learning models and ensemble techniques. This adds knowledge to existing literature and shows new insights into predictive analytics in HR management. The findings from this project have the potential to influence future research directions.”
Ms. Tehara Fonseka **Graduate Research Student at Western University, Canada** **Former Machine Learning Engineer at Leather Broadcasting, Inc.**	

Overall impression	"The system demonstrates a well-engineered architecture, incorporating effective preprocessing techniques and ensemble learning methods. These could contribute greatly to enhanced predictive accuracy and reliability."
Effectiveness of model	"The models seem to exhibit strong performance in accurately predicting employee attrition, leveraging advanced feature engineering and ensemble techniques."
Complexity of system	"The student's system displays a moderate level of complexity in its architecture. Ensembling several complicated deep learning models into multiple ensembling techniques to get the most accurate one. The student has designed and built a complex and intricate algorithmic design and implementation which is very commendable."
Novelty of system	"The attrition prediction model and its underlying architecture demonstrate a substantial level of novelty, particularly in their approach to feature engineering and ensemble learning techniques. I, personally, am encountering a solution such as the one proposed for the first time."
Contribution to research domain	"I believe this project makes a notable contribution to the research domain by advancing our understanding of employee attrition dynamics. Its empirical insights and innovative methodologies offer valuable implications for both academia and industry."

More evaluation results can be found in **Appendix E: Evaluation**.

9.6.2 Focus Group Testing

Ms. Hashini Perera **Senior HR Executive, Grains 'N' Green**	
How **intuitive** and **easy-to-use** do you find AttritionPro?	"I found AttritionPro to be very user-friendly and intuitive as an HR professional. Even non-technical individuals can engage with it easily thanks to its user-friendly interface."

Any **challenges** or **difficulties** while interacting with the system or interpreting its results?	"I had very minimal difficulty in using the system, this had to do with deciphering some technical language in the results. But the prediction results and analysis were very useful and easy to understand and interpret."
How well does the project match the industry's **requirement**? Does the system's predictions resonate with the factors typically associated with employee attrition in your organization?	"The causes causing attrition in our organization are in good agreement with the predictions made by the system. It correctly pinpoints important trends and indicators, offering insightful information for HR decision-making."
How much do you think this project **contributes** to your domain in terms of value? How **valuable** do you perceive the system's predictions in terms of aiding HR decision-making and strategic planning?	"5/5 - For our HR department, the predictions produced by AttritionPro are quite helpful. They offer practical insights that support proactive talent management programs and strategic planning. I believe this is very valuable to our domain and that the student has made an excellent contribution to our field."
Mr. Kannan Periyasamy **HR Manager - System & Services, Daraz (Alibaba Group)**	
How **intuitive** and **easy-to-use** do you find AttritionPro?	"For HR experts, AttritionPro provides a very user-friendly and intuitive interface. Even those with little technological knowledge can easily use it thanks to its simple navigation and intuitive layout."
Any **challenges** or **difficulties** while interacting with the system or interpreting its results?	"There were no major issues that I ran across when using the system."
How well does the project match the industry's **requirement**? Does the system's predictions resonate with the factors typically associated with	"Yes, the factors influencing employee attrition in our company are well represented by the system's projections. The insights given are in good agreement with HR observations and internal data."

employee attrition in your organization?	
How much do you think this project **contributes** to your domain in terms of value? How **valuable** do you perceive the system's predictions in terms of aiding HR decision-making and strategic planning?	“5/5 - AttritionPro's forecasts are a useful source of information for HR decision-making. It could make it possible for us to create focused retention plans and proactively manage attrition threats.”
Mr. Thilina Chandrasekara **Human Resources Manager, Sheraton Kosgoda Turtle Beach Resort**	
How **intuitive** and **easy-to-use** do you find AttritionPro?	“AttritionPro is a very user-friendly and intuitive tool for HR professionals. The attrition prediction process is made easier by its well-organized features and user-friendly interface, making it suitable for users with different degrees of technical competence.”
Any **challenges** or **difficulties** while interacting with the system or interpreting its results?	“I had very few difficulties utilizing the system. It was very straightforward and to the point.”
How well does the project match the industry’s **requirement**? Does the system's predictions resonate with the factors typically associated with employee attrition in your organization?	“The factors linked to employee attrition in our firm are in line with the predictions made by the system. Our HR data and organizational observations are in good agreement with the patterns and indicators that have been found.”
How much do you think this project **contributes** to your domain in terms of value? How **valuable** do you perceive the system's predictions in terms of aiding HR decision-making and strategic planning?	“5/5 - Strategic planning and HR decision-making benefit greatly from the forecasts produced by AttritionPro. They provide useful information that guides proactive talent management techniques and focused interventions.”
Mr. Ishara Priyadarshana	

Senior Manager – HR, Fairway Holdings	
How **intuitive** and **easy-to-use** do you find AttritionPro?	"AttritionPro's user-friendly layout and simplified UI make it an ideal tool for HR professionals. Because of the system's efficient architecture, users may easily obtain and analyze a lot of predictive information."
Any **challenges** or **difficulties** while interacting with the system or interpreting its results?	"None at all. It was very easy and convenient to use."
How well does the project match the industry's **requirement**? Does the system's predictions resonate with the factors typically associated with employee attrition in your organization?	"The system satisfies the requirements of the problem perfectly. A lot of factors about the employees get considered and a lot of useful predictions and results are shown to us. The factors influencing employee attrition in our company closely match the predictions made by the system. The insights offered align with both our organizational observations and HR data."
How much do you think this project **contributes** to your domain in terms of value? How **valuable** do you perceive the system's predictions in terms of aiding HR decision-making and strategic planning?	"4/5 - AttritionPro's forecasts are a useful tool for directing HR decision-making procedures. They offer useful information that may be used to detect attrition risks and guide strategic workforce planning projects."

9.7 LIMITATIONS OF EVALUATION

Although the evaluation method used in this research project combines quantitative and qualitative methods, there are some limitations. One limitation is that focus group analysis cannot represent the diversity of end users who may interact with the system in a real environment. Additionally, the evaluation process may be subject to time and resource constraints that affect the depth and understanding of the analysis. These limitations should be taken into account when interpreting test results and drawing conclusions about performance and effectiveness.

9.8 EVALUATION ON FUNCTIONAL REQUIREMENTS

ID	Requirement	Priority	Evaluation
FR1	Users must be able to input a certain employee's details to make a prediction of their attrition.	Must	Implemented
FR2	Prediction results should be generated for the input data, showing the attrition of that employee.	Must	Implemented
FR3	The results of the prediction should be displayed to the user in a user-friendly, intuitive way.	Should	Implemented
FR4	The risk of attrition for an employee should be generated from the data input by the user.	Should	Implemented
FR5	The user should be able to view the generated risk of attrition in a presentable manner.	Should	Implemented
FR6	The prediction history could be saved, and the most recent predictions could be shown to the user on the GUI.	Could	Implemented
FR7	The feature-wise breakdown of the factors most affecting an employee's risk of attrition would be displayed to the user.	Would	Implemented
FR8	The system would have an intuitive GUI for the user to interact with the system.	Would	Implemented
Completion of Functional Requirements = $\frac{8}{8} * 100\% = 100\%$			

9.9 EVALUATION ON NON-FUNCTIONAL REQUIREMENTS

ID	Requirement	Description	Priority	Evaluation
NFR1	Performance	The performance of the system, in terms of speed, efficiency and prediction, should be consistent and adequate.	Desirable	Implemented

NFR2	Security	Employee details and other confidential and sensitive information need to be protected appropriately.	Desirable	Not Implemented
NFR3	Scalability	The system should be able to scale to function at a larger scale and capable of handling large amounts of data and parameters.	Desirable	Implemented
NFR4	Quality of Output	The system should be able to make reliable and accurate predictions consistently.	Important	Implemented
NFR5	Usability	The usability of the system has to be at a level where it is easy to use, convenient and straightforward. It should be simple and easy to understand and beginner friendly.	Important	Implemented
Completion of Non-Functional Requirements = $\frac{4}{5} * 100\% = 80\%$				

9.10 CHAPTER SUMMARY

In summary, evaluating the system is done using a balanced method that combines quantitative measurements and qualitative recommendations. The evaluation process includes many elements such as problem definition, process analysis, design, data use and performance analysis. The opinions of the experts and the results obtained from the evaluation team gave an idea about the strengths and limitations of the system. Although there are some limitations, the evaluation process provides important suggestions for the future development and improvement of the application, ultimately making it useful and effective in real-world situations.

CHAPTER 10: CONCLUSION

10.1 CHAPTER OVERVIEW

This chapter will take a look at the achievement of the project aims, goals and objectives, what skills were used (both existing and new), any deviations that the project may have taken from the initial project proposal and establish the improvements required when progressing forward into the more research in the domain.

10.2 ACHIEVEMENTS OF RESEARCH AIMS AND OBJECTIVES

10.2.1 Achievement of Aim

The primary aim of this study is to design, develop, and evaluate an advanced ensemble prediction model that incorporates several deep learning algorithms to effectively predict employee turnover in enterprises.

The research project successfully has accomplished the proposed research aim by designing a novel architecture, developing it into a functional system and then evaluating it according to industry standards to complete a fully functional employee attrition prediction system using multiple deep learning models in an ensembled super-learner.

10.2.2 Achievement of Objectives

Research Objectives	Learning Outcomes	Status
Problem Definition: To identify the contributing factors behind employee attrition. RO1: To determine the factors that affect employee turnover. RO2: To verify previous limitations in employee attrition and DL and ensemble domain research in order to find and address gaps.	LO1, LO3, LO4	Completed

Literature Review: To examine current literature for relevant insights and critically evaluate earlier research on the topic. RO3: To identify trends, gaps, and emerging topics in the domain of employee attrition prediction, and synthesize data from earlier studies. RO4: To assess the performance of various techniques and approaches used in prior research initiatives, with a particular emphasis on their relevance to the proposed attrition prediction model.	LO1, LO4, LO8	Completed
Requirement Elicitation: RO6: To collect and analyze relevant factors influencing employee attrition, including historical employee data, workplace conditions, and external variables. RO7: To explore existing literature and models in the field of attrition prediction to identify key features, variables, and methodologies that have proven effective.	LO2, LO3	Completed
Research Design: To design an architecture and a framework capable of addressing the concerns indicated in the solution. RO8: To design the high-level architecture for the proposed solution involving the DL and Ensemble methods. RO9: To develop an early prototype and prediction model to better assess the proposed solution.	LO1, LO2, LO5, LO8	Completed
Implementation: To develop a system capable of tackling the issues mentioned. RO10: To apply the DL models to more sample data and select the best technique.	LO1, LO2, LO5, LO7	Completed
Testing – Quantitative: To rigorously evaluate the developed system and ML models using quantitative methods and appropriate datasets to measure performance. RO11: To conduct model assessment, performance testing, and integration execution for the ensemble model.	LO1, LO5, LO6, LO8	Completed

Evaluation – Qualitative: To assess the effectiveness of the solution and gather qualitative insights from relevant stakeholders and experts. RO12: To engage in qualitative evaluation methods to collect feedback and insights from users and experts in the attrition domain, and technological domain and analyze the data obtained in order to improve on them and enhance its usability and effectiveness.	LO1, LO5, LO6, LO8	Completed
Documentation: RO13: To keep a detailed research record that includes methodology, findings about DL and Ensemble technique, and obstacles, track ethical, legal, or professional difficulties, and how they were resolved, and write a thorough document that summarizes skills, plans for the project, and critical reviews of the final ensemble model.	LO6, LO7, LO8	Completed

Table 10.1: Achievement of Objectives (Self-Composed)

10.3 UTILIZATION OF KNOWLEDGE FROM THE COURSE

Module	**Knowledge utilized**
Software Development Group Project	The author was able to utilize multiple skills and knowledge that they had gained from this project. Documentation skills, project management skills, experience building and training ML models to building a functional ML system were all utilized by the author during the course of this project.
Machine Learning	Building, training and understanding ML were skills gained during this module. This knowledge was directly transferrable to the author throughout the project.
Web Design and Development	The module introduced patterns and standards of web design and development. The various aspects of building a web application from its architecture to its look and feel were covered in the module.
Software Development	Fundamentals and basic concepts of Python were introduced in the module which made it convenient to use during model building and for the Python backend as well.

Table 10.2: Utilization of Knowledge from this Course (Self-Composed)

10.4 USE OF EXISTING SKILLS

- **Angular** – Angular was used as the frontend framework during implementation because the author had some experience and existing skills in Angular. This was convenient as it streamlined and fast-tracked development of the frontend.
- **Flask –** The author utilized previous skills gained and experience from using Flask in their Software Development Group Project. The author has used it before as a backend framework with API endpoints to communicate with the frontend and to generate predictions from a pre-trained model. This knowledge was useful during implementation of the backend and setting up communications between the UI, the database and the models.
- **Machine Learning** – The author was able to use previous skills gained and experience in ML that they had gained from their modules and also from experience using it during their Software Development Group Project.

10.5 USE OF NEW SKILLS

- **Deep Learning** – At the start of the project, the author had limited knowledge and experience working with Deep Learning models. The author had to learn the fundamentals of DL and their implementations.
- **Ensemble Techniques** – The author had no prior knowledge regarding ensemble techniques that were required to build a super learner model for prediction.
- **LATEX Documentation** – Documentation using formats such as LATEX were completely new to the author. Therefore, during the course of the research, the author learnt and applied these new skills during the documentation process.
- **Exploratory Data Analysis –** Univariate and Bivariate data analysis, along with Correlation heat maps were generated to understand patterns in the data.
- **Wide and Deep Models –** The author had no prior experience or understanding about Wide and Deep Models. Implementation, design and methodology of wide and deep models had to be learned and applied in this project.

10.6 ACHIEVEMENT OF LEARNING OUTCOMES

Research Objective	Learning Outcomes	Research Question	Status
Problem Definition	LO1, LO3, LO4	RQ1	Completed
Literature Review	LO1, LO4, LO8	RQ2	Completed
Requirement Elicitation	LO2, LO3	RQ1, RQ2	Completed
Research Design	LO1, LO2, LO5, LO8	RQ2	Completed
Implementation	LO1, LO2, LO5, LO7	RQ2	Completed
Testing – Quantitative	LO1, LO5, LO6, LO8	RQ2	Completed
Evaluation – Qualitative	LO1, LO5, LO6, LO8	RQ3	Completed
Documentation	LO6, LO7, LO8	RQ1, RQ2	Completed

Table 10.3: Achievement of Learning Outcomes (Self-Composed)

10.7 PROBLEMS AND CHALLENGES FACED

Problems and challenges faced during the research, design and implementation of this project can be found at **Appendix F: Conclusion**.

10.8 DEVIATIONS

Scope Related Deviations

In terms of scope related deviations, some new functionalities was introduced to the system – Attrition Risk. When an employee's details are entered and a result is generated, the system will identify the risk of that specific employee leaving the company. The risk of attrition will be classified into 3 classes – High Risk, Medium Risk, Low Risk. This will be helpful to identify the status of the employee in more detail and understand the seriousness of the context at hand.

Another new feature that was implemented was reloading saved predictions from recent predictions into the system to allow the user to make adjustments and test out different combinations for predictions. This allows users to make changes to the employee's details to test and identify various changes and improvements they could make and the effect it would have on the employee's risk of resignation.

Additionally, the author implemented allowing the user to choose the ensemble technique they want to make the final prediction with, out of three possible techniques. By default, the stacking method is used, and the user can change this to use either Voting or Simple Average, to get the rest of the results and analysis done using that technique.

Furthermore, the author was able to implement calculating the Top Contributing Factors for employee attrition and display it to the user, with their respective positive and negative correlation values. This information is useful to the user in that it shows the underlying factors that are affecting attrition, to what extent they affect and whether they have a positive or negative effect on the result.

The author was also able to create and provide a user manual for the system for users to refer to as a guide. This was provided to the evaluators during the evaluation process.

Overall, no deductions had to be made. The only changes that were made were additions to the functionality of the system.

10.9 LIMITATIONS OF THE RESEARCH

- Data Limitations – The dataset used is the IBM Employee Attrition dataset built by research scientists at IBM to simulate actual employee data. However, the dataset only consists of 1470 records with 35 attributes. This doesn't provide the most ideal abundance of data that a Deep Learner Ensemble algorithm would need for its optimal performance. With more data, there is a possibility that the proposed model architecture might perform better.
- Model Interpretability – The proposed architecture is not an easily understandable algorithm to an untrained user. To interpret and understand how the system works and its underlying processes, it would require a professional with knowledge about such algorithms.

10.10 FUTURE ENHANCEMENTS

The performance of the developed algorithms could be improved further in future works by optimizing the features and data preprocessing. Further, more combinations and amalgamations of deep learning algorithms and ensemble modelling methods can be developed to further assess the performance, accuracy, and suitability of deep learning ensemble models for employee attrition.

The future of attrition prediction seems promising. Advances in machine learning, deep learning, and ensemble methods continue to increase the accuracy and efficiency of prediction. Furthermore, greater investment in these systems is being driven by a growing realization of the value of staff retention. As a result, more advanced and effective attrition prediction models can be expected in the future (Pokkuluri and Devi, 2023).

Future attrition prediction research can concentrate on overcoming these challenges, establishing more effective prediction models, and investigating new machine learning and deep learning methodologies. Furthermore, additional study is required to comprehend the numerous elements that contribute to employee attrition and how they interact. This can aid in the development of more detailed and sophisticated attrition prediction models (Raza et al., 2022).

10.11 RESEARCH PAPER

The author was able to make conference paper submissions to EuCNC & 6G Summit 2024 and ICoABCD 2024 and to the journal IJAI 2024. An extract of the research paper can be found in the **Appendix F: Conclusion**.

10.12 ACHIEVEMENT OF CONTRIBUTION TO THE BODY OF KNOWLEDGE

10.12.1 Contribution to the Problem Domain

1. Application of Deep Learning Ensemble Methods in Employee Attrition Prediction

One significant contribution of this research project to the problem domain is about introducing deep learning with ensemble techniques into the domain of employee attrition prediction. This study improves the accuracy and reliability of the attrition prediction by using methods such as stacking, voting, and simple averaging. This advancement addresses the critical issue businesses face in identifying and retaining valuable employees. Using multiple integrated deep learning models provides a more powerful way to analyze complex data and extract valuable patterns related to employee turnover.

2. Improvement in Employee Attrition Prediction and Retention

Another important contribution is the use of deep learning to improve employee forecasting and retention strategies. Organizations can prevent turnover and hold onto valuable personnel by proactively implementing targeted retention programs based on realistic predictions of future attrition instances. Companies can save a lot of money by taking a proactive approach to employee retention, which also helps to create a more stable and effective team. As a result, the research advances the general efficacy of HR procedures in relation to organizational sustainability and talent management.

10.12.2 Contribution to the Research Domain

1. Advancing the Role of Deep Learning in Ensemble Techniques

This study investigates novel approaches for merging multiple models to improve prediction performance, hence advancing the role of deep learning in ensemble techniques. It illustrates how deep learning algorithms can enhance the precision and resilience of predictive models by incorporating deep neural networks into ensemble frameworks. CNNs and FNNs are two examples of individual models whose complementary strengths are leveraged by the ensemble approach to generate higher prediction power over standalone models. This work highlights the value of ensemble learning in model creation and broadens the range of deep learning applications in predictive analytics.

2. Integration of Ensemble Methods in HR Analytics

The use of ensemble methods into HR analytics provides a contribution to the field of study. The work presents a new design pattern for applying machine learning in HR decision-making processes by integrating ensemble approaches. By using ensemble approaches, HR personnel may be able to make use of the combined predictive power of the various models. This could lead to more precise and dependable employee turnover results in the future.

10.13 CONCLUDING REMARKS

In conclusion, this thesis explores the use of deep learning in attrition prediction to improve the accuracy and efficiency of the forecasting model in HR analytics. Through comprehensive examination of stacking, voting, and simple averaging methods, this research demonstrates the potential of ensemble learning to improve the predictive performance allowing proactive retention tactics to be put in place. Using the collective intelligence of different models, ensemble designs provide effective solutions to identify and resolve employee risk factors.

In addition, this research contributes to the advancement of both the problem domain and research domain. Fundamentally, the use of deep learning ensembles represents a significant step forward for the HR industry in improving employee churn prediction and management. Integrating advanced analytics into HR decision-making processes shows great potential to improve corporate sustainability and employee management strategies.

In the research domain, this research contributes to the development of deep learning and ensembles by exploring new integration models and methods. This research expands the scope of predictive models by advancing the role of deep learning in a learning system and reinforces the importance of ensemble methods in dealing with intricate data.

The implemented functionalities combined provide the end users of the system with answers to, not just a "What", but also a "How" and a "Why" to the questions regarding employee attrition. The users are not just shown the chance and risk of an employee resigning. They are provided with a system that provides insight into the underlying factors that affect the problem and a system that allows a user to make changes and fine tunings to the input data over time, by using existing results to gauge how the changes affect the result and to identify exactly which combinations of features will result in them being able to retain a specific employee. Thereby, ultimately providing an effective solution to the problem of employee attrition.

Overall, the results of this study demonstrate the potential of deep learning in human resources research and predictive modeling. As organizations continue to embrace data-driven decision-making processes, the insights and methods presented in this thesis provide valuable insight into improving management intelligence and repair organization. Going forward, more research and the use of ensemble technologies will be needed to improve the accuracy and effectiveness of predictive models in HR analytics and other areas.

REFERENCES

Al-Darraji, S., Honi, D. G., Fallucchi, F., Abdulsada, A. I., Giuliano, R., & Abdulmalik, H. A. (2021). Employee attrition prediction using deep neural networks. *Computers*, *10*(11), 141.

Alsheref, F. K., Fattoh, I. E., & M Ead, W. (2022). Automated Prediction of Employee Attrition Using Ensemble Model Based on Machine Learning Algorithms. *Computational intelligence and neuroscience*, *2022*, 7728668. https://doi.org/10.1155/2022/7728668.

Chakraborty, R., Mridha, K., Shaw, R. N., & Ghosh, A. (2021, September). Study and prediction analysis of the employee turnover using machine learning approaches. In 2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON) (pp. 1-6). IEEE.

Chung, D., Yun, J., Lee, J., & Jeon, Y. (2023). Predictive model of employee attrition based on stacking ensemble learning. *Expert Systems with Applications*, *215*, 119364.
Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*(3), 685-695.

Das, S., Chakraborty, S., Sajjan, G., Majumder, S., Dey, N., Tavares, J.M.R.S. (2023). Explainable AI for Predictive Analytics on Employee Attrition. In: Patel, K.K., Santosh, K.C., Patel, A., Ghosh, A. (eds) Soft Computing and Its Engineering Applications. icSoftComp 2022. Communications in Computer and Information Science, vol 1788. Springer, Cham. https://doi.org/10.1007/978-3-031-27609-5_12.

Fallucchi, F., Coladangelo, M., Giuliano, R., & William De Luca, E. (2020). Predicting employee attrition using machine learning techniques. *Computers*, *9*(4), 86.

Fuller & Kerr. (2022). “The Great Resignation Didn’t Start with the Pandemic”. Available from https://hbr.org/2022/03/the-great-resignation-didnt-start-with-the-pandemic. [Accessed 16 September 2023].

Gamage, T. (2023). EA-Sys - An Ensemble Approach Using Super Learner for Employee Attrition Prediction & Retention System. *Asia Conference on Algorithms, Machine Learning, and Information Technology (AMLIT).* Nanjing, China. 28 July 2023.

Ghimire, D. (2020). Comparative study on Python web frameworks: Flask and Django.

Heidari, M., Zad, S., & Rafatirad, S. (2021, April). Ensemble of supervised and unsupervised learning models to predict a profitable business decision. In *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)* (pp. 1-6). IEEE.

Jain, N., Tomar, A. and Jana, P.K. (2020). A Novel Scheme for Employee Churn Problem Using Multi-attribute Decision Making Approach and Machine Learning. Journal of Intelligent Information Systems. Available from https://doi.org/10.1007/s10844-020-00614-9 [Accessed 20 November 2020].

Jain, P.K., Jain, M. & Pamula, R. (2020). Explaining and predicting employees’ attrition: a machine learning approach. *SN Appl. Sci.* **2**, 757. https://doi.org/10.1007/s42452-020-2519-4.

Jain, R., & Nayyar, A. (2018, November). Predicting employee attrition using XGBoost machine learning approach. In *2018 international conference on system modeling & advancement in research trends (smart)* (pp. 113-120). IEEE.

Marvin, G., Jackson, M., and Alam, Md.G.R. (2021). A Machine Learning Approach for Employee Retention Prediction. 2021 IEEE Region 10 Symposium (TENSYMP). Available from https://doi.org/10.1109/tensymp52854.2021.9550921 [Accessed 29 November 2022].

Muslim, M. A., & Dasril, Y. (2021). Company bankruptcy prediction framework based on the most influential features using XGBoost and stacking ensemble learning. International Journal of Electrical and Computer Engineering (IJECE), 11(6), 5549-5557.

Naz, K. et al. (2022). Predictive Modeling of Employee Churn Analysis for IoT-Enabled Software Industry. Applied Sciences, 12 (20), 10495. Available from https://doi.org/10.3390/app122010495 [Accessed 26 November 2022].

Next Move Strategy Consulting. (2023). Artificial intelligence (AI) market size worldwide in 2021 with a forecast until 2030 (in million U.S. dollars). Statista. Statista Inc.. Accessed: October 06, 2023. https://www.statista.com/statistics/1365145/artificial-intelligence-market-size/.

Patro, A.C. et al. (2021). A Novel Approach to Improve Employee Retention Using Machine Learning. IEEE Xplore. Available from https://doi.org/10.1109/CSNT51715.2021.9509601 [Accessed 19 December 2022].

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel M., Prettenhofer P., Weiss R and Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, *12*, 2825-2830.

S. George, K. A. Lakshmi and K. T. Thomas. (2022). "Predicting Employee Attrition Using Machine Learning Algorithms," *2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, Greater Noida, India, 2022, pp. 700-705, Doi: 10.1109/ICAC3N56670.2022.10074131.

Tüylü, A. N. A., & Eroglu, E. (2022). The prediction of product return rates with ensemble machine learning algorithms. *Journal of Engineering Research*.

Raschka, Sebastian, Joshua Patterson, and Corey Nolet. (2020). "Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence" *Information* 11, no. 4: 193. https://doi.org/10.3390/info11040193.

Raza, Ali, Kashif Munir, Mubarak Almutairi, Faizan Younas, and Mian Muhammad Sadiq Fareed. (2022). "Predicting Employee Attrition Using Machine Learning Approaches" *Applied Sciences* 12, no. 13: 6424. https://doi.org/10.3390/app12136424.

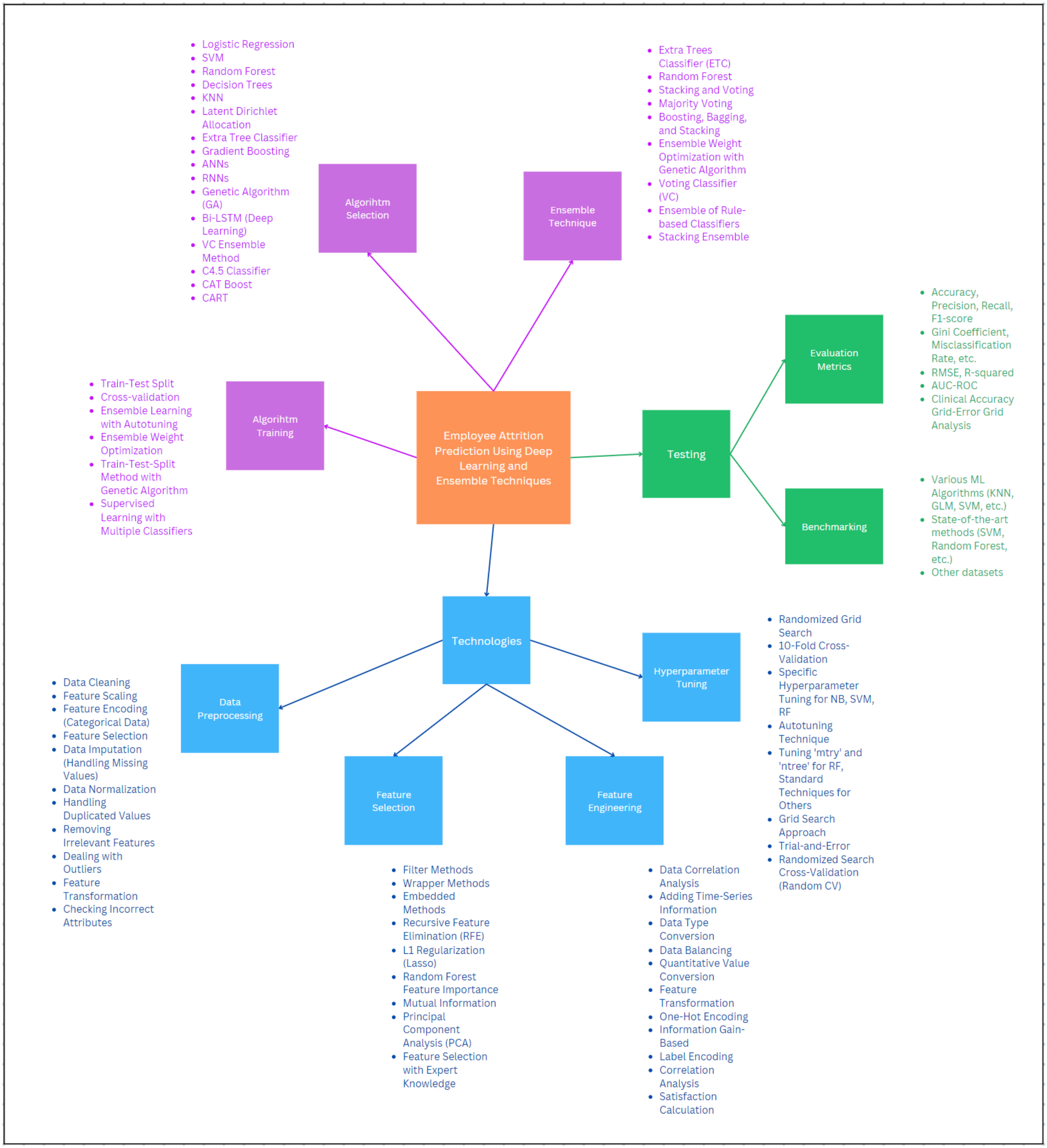
Waskom, M. L. (2021). Seaborn: statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021.

Zagorsky J. (2022). “Are we really facing a resignation crisis?”. Available from https://www.weforum.org/agenda/2022/01/great-resignation-crisis-quit-rates-perspective/. [Accessed 16 September 2023]

Zagorsky, J. L. (2024). How Did the Great Resignation Impact Government Jobs?. In *Pivoting Government through Digital Transformation* (pp. 57-71). Auerbach Publications.

APPENDIX A: LITERATURE REVIEW

CONCEPT GRAPH



LITERATURE SURVEY

For a more legible view of the Literature Survey, refer to this **Link**.

Citation	Summary	Technology used										Limitations	Contributions
		Dataset	Data Preprocessing	Feature Selection	Feature Engineering	Algorithm Selection Process	Algorithm Training	Ensemble Technique	Hyperparameter Tuning	Evaluation Metrics	Benchmarking		
(Raza et al., 2022)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Kim, D., Choi, D., Kang, A.R., Woo, J., Han, Y., Chun, S.W., & Kim, J. (2022). Intelligent Ensemble Deep Learning System for Blood Glucose Prediction Using Genetic Algorithms. Complex., 2022, 7902418:1-7902418:10.	This study presents an intelligent ensemble deep learning system for blood glucose prediction using genetic algorithms. The proposed model uses a combination of multiple algorithms and a genetic algorithm to optimize the weight ensemble and improve the accuracy and robustness of the predictions. The study found that this approach outperformed single-algorithm models and could be further improved with more strictly recorded data. The paper also provides an overview of previous studies and discusses the potential implications of this research for diabetes patients.	The dataset used in this study consists of blood glucose readings, carbohydrate intake, and insulin administration information collected from diabetic patients. The data was collected remotely through web pages and databases, and then preprocessed to fit the neural networks. The study used two different datasets: one with continuous glucose monitoring (CGM) readings only, and another with CGM readings and exogenous factors (CHO intake and insulin administration time information). The performance of the proposed model was evaluated using root mean square error (RMSE) and compared to other models using the same datasets.	The study performed several preprocessing tasks to clean and transform the data into a suitable form for fitting the neural networks. The data used in this study were collected from real-life conditions and may have contained outliers due to errors in wireless communication devices. Therefore, the primary purpose of preprocessing was to clean the data and remove any outliers. Additionally, the exogenous factors recorded directly by patients were likely to be inconsistent because they were directly inputted by patients, requiring appropriate preprocessing tasks. The study also converted the data to the lowest dimension to unify the insulin and carbohydrate information that occurs for each patient. Finally, the time-series data (i.e., the time information of insulin and carbohydrate intake) were added to the past blood glucose values via preprocessing.	The study did not explicitly mention any feature selection process. However, the study used two different datasets: one with continuous glucose monitoring (CGM) readings only, and another with CGM readings and exogenous factors (CHO intake and insulin administration time information). The study then used two different types of models: univariate models that used only past blood glucose values as input and multivariate models that used past blood glucose values in addition to CHO intake and insulin administration time information. Therefore, it can be inferred that the study used all available features in the dataset and did not perform any feature selection.	The study did not explicitly mention any feature engineering process. However, the study added the time-series data (i.e., the time information of insulin and carbohydrate intake) to the past blood glucose values via preprocessing. This can be considered a form of feature engineering, as it created new features by combining existing features. Additionally, the study used two different types of models: univariate models that used only past blood glucose values as input and multivariate models that used past blood glucose values in addition to CHO intake and insulin administration time information. This can also be considered a form of feature engineering, as it created new features by combining blood glucose values with other relevant information.	The study used five different recurrent neural network (RNN)-based algorithms to predict future blood glucose values at 15, 30, and 60 min prediction horizons (PHs) from time-series data and exogenous factors. The five RNN algorithms used in the study were Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Simple Recurrent Unit (SRU), Recurrent Highway Network (RHN), and Multiplicative Integration Convolutional Neural Network (MICNN). The study then used an ensemble weight optimization technique using a genetic algorithm (GA) to derive robust models based on each factoring condition. The GA was used to optimize T2DM ensemble variable weights to prevent hypoglycemic and hyperglycemic events. Therefore, it can be inferred that the study used a combination of different RNN-based algorithms and an ensemble weight optimization technique to improve the performance of the models.	The authors split the input data according to the lookback period for the number of previous values to be used and the time step (i.e., sequence size). All samples were set up with sliding windows with a step size of one. The input data were then fed into five different recurrent neural network (RNN)-based algorithms, including Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Simple Recurrent Unit (SRU), Recurrent Highway Network (RHN), and Multiplicative Integration Convolutional Neural Network (MICNN), to yield the predicted blood glucose values. Each algorithm is a slight variant, as described in Table 1, based on the hyperparameters. All hyperparameters are determined experimentally through trial-and-error for good performance. The predicted blood glucose values from the five RNN algorithms were then combined using an ensemble weight optimization technique based on a genetic algorithm (GA) to derive robust models based on each factoring condition. The GA was used to optimize T2DM ensemble variable weights to prevent hypoglycemic and hyperglycemic events. Therefore, the algorithm training process used in the study involved feeding the input data into five different RNN-based algorithms, optimizing the hyperparameters of each algorithm, and combining the predicted values using an ensemble weight optimization technique based on a genetic algorithm.	The authors used an ensemble weight optimization technique based on a genetic algorithm (GA) to derive robust models based on each factoring condition. The GA was used to optimize T2DM ensemble variable weights to prevent hypoglycemic and hyperglycemic events. The ensemble technique involved combining the predicted blood glucose values from the different recurrent neural network (RNN)-based algorithms, including Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Simple Recurrent Unit (SRU), Recurrent Highway Network (RHN), and Multiplicative Integration Convolutional Neural Network (MICNN), using the optimized ensemble weights. The ensemble weights were optimized using a genetic algorithm, which is a metaheuristic optimization algorithm inspired by the process of natural selection. The genetic algorithm was used to find the optimal combination of weights for each algorithm to minimize the prediction error. Therefore, the ensemble technique used in the study involved combining the predicted values from the different RNN-based algorithms using optimized ensemble weights obtained through a genetic algorithm.	In the intelligent ensemble deep learning system, hyperparameter tuning was performed for the five RNN-based algorithms used in the prediction models. The hyperparameters were determined experimentally through trial-and-error to achieve optimal performance. The specific hyperparameters that were tuned for each algorithm are not mentioned in the provided context.	Two evaluation metrics were used to assess the performance of the proposed models: RMSE (Root Mean Square Error) and CG-EGA (Clinical Accuracy Grid-Error Grid Analysis). RMSE is a commonly used evaluation indicator in regression problems and was used to compare the performance of different models. CG-EGA is a method that quantifies the clinical accuracy of measured and predicted blood glucose (BG) values. It divides a scatterplot of reference glucose values and predicted glucose values into five regions (A to E), with Zone A indicating the most clinically significant results. These metrics were used to evaluate the predictive accuracy and clinical relevance of the models.	The benchmark used to compare the performance of the proposed models in this project was the root mean square error (RMSE). The objective function of the models was to minimize the RMSE between the predicted blood glucose (BG) values and the actual BG values. The lower the RMSE, the better the performance of the models.	The study also applies machine learning models, specifically recurrent neural networks (RNNs), for BG prediction. However, there may be other advanced machine learning techniques or algorithms that could be explored for improved prediction accuracy. Future work in this area could involve exploring the use of other advanced machine learning techniques, such as deep learning models or ensemble methods, to further enhance the accuracy of BG prediction.	The research contribution of the study towards the employee attrition prediction domain in terms of technology is the application of deep learning algorithms, specifically recurrent neural network (RNN) and RNN-based algorithms such as long short-term memory (LSTM), stacked LSTM, bidirectional LSTM, and gated recurrent unit. These algorithms are optimized for sequence data, making them suitable for analyzing time-series data such as historical employee data. The study also utilizes a genetic algorithm (GA) to optimize the weights of the prediction models through ensemble techniques. This combination of deep learning algorithms and GA optimization enhances the accuracy and performance of the models.
[illegible]	The study focuses on predicting employee attrition using machine learning techniques. The researchers gathered employee data from the IBM dataset and performed preprocessing to prepare the data for analysis. They then analyzed the dataset to identify the most influential features that contribute to employee attrition. The study compared their proposed deep learning model with other state-of-the-art methods and found that their model outperformed the competitors in terms of accuracy and f1-score. [illegible]	The IBM HR dataset used in this study contains 35 features for 1470 employees. The dataset includes various information such as age, monthly income, business travel, daily rate, education level, job satisfaction, marital status, and attrition (whether the employee left the company or stayed). The dataset was provided by IBM Analytics.	The dataset underwent several preprocessing operations. Firstly, some features that were identical for all employees, such as EmployeeCount, Over18, and StandardHours, were omitted. The EmployeeNumber feature was also removed as it was unrelated to the classification problem. Additionally, categorical data encoding and rescaling were performed on the dataset.	The feature selection techniques used in this study were not explicitly mentioned in the provided context.	The most important features that influence employee attrition were identified. Additionally, the dataset was balanced to address its imbalanced nature. The specific feature engineering techniques used in this study are not mentioned in the provided context.	[illegible]	In this study, the algorithm was trained using a deep learning model. The dataset was divided into two parts: a training set and a test set. The training set, which accounted [illegible]	[illegible]	[illegible]	[illegible]	[illegible]	The limitations of this study in terms of technology include: 1. Dataset comparability: The study used its own dataset, which makes it difficult to directly compare the results with other studies that used different datasets. This limits the generalizability of the findings. 2. Hyperparameter selection: The study employed a grid search approach to tune the hyperparameters of the prediction model. However, the selection of hyperparameters such as the number of hidden layers, number of neurons, and activation functions can still be further optimized to improve the model's performance. 3. Imbalanced dataset: The dataset used in the [illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	The feature selection technique used in this study is Principal Component Analysis [illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Mehta, V., & Modi, S. (2021, December). Employee Attrition System Using Tree Based Ensemble Method. In 2021 2nd International Conference on Communication, Computing and Industry 4.0 (C2I4) (pp. 1-4). IEEE.	The study explains the issue of employee attrition and proposes a solution using Machine Learning Algorithms. It reviews the problem of employee attrition using the tree-based Ensemble Machine Learning Model and identifies statistically significant factors that connect to an employee's decision to leave. The study aims to help business organizations mitigate the loss of talented employees and improve overall performance. The authors use the 'IBM HR Analytics Employee Attrition Performance' dataset, which consists of 1470 observations and 35 unique features. They also discuss the challenges of data collection and the importance of data preprocessing in the performance of the algorithm or model. The study emphasizes the use of tree-based ensemble techniques, including the random forest and gradient boost, to improve the accuracy of employee attrition prediction.	The dataset used in the study is the 'IBM HR Analytics Employee Attrition Performance' dataset. It consists of 1470 different observations and 35 unique features. The dataset was created by IBM and was collected by asking employees of IBM to respond to a questionnaire. The features in the dataset are used to understand the factors that connect to an employee's decision to leave the organization. The dataset plays a crucial role in training and evaluating the tree-based Ensemble Machine Learning Model for predicting and preventing employee attrition in business organizations.	The authors of the study performed several preprocessing steps on the dataset to ensure its quality and integrity. They started by reviewing the data and conducting a quality check to identify any missing values. To handle missing values, they used a null call to determine whether any feature had a missing value or not, and then filled any null characters with random data to ensure that the dataset was complete and free of missing values. Additionally, the authors transformed categorical data into dummy variables. This involved converting categorical variables into binary indicators, where 0 denoted absence and 1 denoted presence. This process created a two-dimensional binary matrix, with each column representing a different category. Furthermore, the authors removed duplicate characteristics and rows from the dataset to ensure that no data was replicated, thus making the dataset unique. These preprocessing steps were crucial in preparing the dataset for the application of the tree-based Ensemble Machine Learning Model, as they ensured that the data was clean, complete, and in a suitable format for analysis and modeling.	The study emphasizes the importance of feature selection in the model's accuracy. It mentions that the accuracy of the model depends on the features included in the model. To visualize the distribution of the data to the features of the dataset, the authors used a Kernel Density Estimate (KDE) plot to get the exact density of the dataset allocated in the features with each other. Additionally, they created a correlation matrix to understand how the characteristics are connected. While the study provides insights into the visualization and understanding of the features, it does not explicitly mention a specific feature selection technique or process. Therefore, the details of the feature selection approach used in the study are not explicitly provided in the paper.	The paper does not contain specific details about the feature engineering techniques used in the study.	Two tree-based ensemble techniques, namely Random Forest and Gradient Boosting were used to improve the accuracy of the employee attrition prediction. They selected these algorithms based on their ability to handle high-dimensional data, handle missing values, and provide accurate predictions. The authors also used a feature selection approach with the Random Forest algorithm to identify the most important features that contribute to employee attrition. The study does not provide further details on the algorithm selection process.	The authors employed a train-test-split method to estimate the performance of the algorithm. This method involves partitioning the dataset into training and testing data, which is accomplished with the help of the scikit-learn library. After partitioning the data, the authors trained the Random Forest and Gradient Boosting models individually and performed the algorithm used in each model to determine the accuracy of the future models. The authors used scikit-learn for performing both the models, and in the case of Random Forest, they determined the set of parameters that they fed into their Random Forest classifier.	The authors employed two tree-based ensemble techniques, namely Random Forest and Gradient Boosting. In Random Forest, the authors used a feature selection approach to identify the most important features that contribute to employee attrition. The authors also determined the set of parameters that they fed into their Random Forest classifier. In Gradient Boosting, the authors added predictors in a sequential order, and the gradient of descent aided the gradient booster in identifying and correcting errors in learners' predictions. The authors evaluated the tree-based ensemble to get the best results from the existing tree methods. Overall, the authors used ensemble techniques to improve the accuracy of the employee attrition prediction and identify the most important features that contribute to employee attrition.	The authors used a grid search approach to tune the hyperparameters of the Random Forest model. They defined a range of values for each hyperparameter and then trained the model with all possible combinations of hyperparameters. The authors then selected the combination of hyperparameters that resulted in the best performance of the model. The hyperparameters that were tuned include the number of trees in the forest, the maximum depth of the trees, and the minimum number of samples required to split an internal node. The authors did not mention any hyperparameter tuning for the Gradient Boosting model.	Several metrics were used to evaluate the performance of the Random Forest and Gradient Boosting models. The metrics used included accuracy, precision, recall, and F1 score. The authors also used a confusion matrix to visualize the performance of the models. The confusion matrix helped to identify the number of true positives, true negatives, false positives, and false negatives. The authors used these metrics to compare the performance of the Random Forest and Gradient Boosting models. They found that the Gradient Boosting model performed better than the Random Forest model in terms of accuracy and recall. The authors also used cross-validation to validate the performance of the models. They used a 10-fold cross-validation approach to estimate the performance of the models on unseen data. Overall, the authors used a variety of evaluation metrics to assess the performance of the models and ensure that they were accurate and reliable.	The authors did not mention any specific benchmarking approach in their study. However, they compared the performance of their Random Forest and Gradient Boosting models with each other and evaluated their performance using various metrics such as accuracy, precision, recall, and F1 score. They also compared their results with previous studies that used different machine learning algorithms to predict employee attrition. The authors found that their Gradient Boosting model outperformed the Random Forest model and achieved a higher accuracy rate than previous studies. Therefore, the authors concluded that their proposed tree-based ensemble method is a reliable and accurate approach for predicting employee attrition.	First, the authors noted that the dataset used in this study was limited to a medium-sized dataset with only 35 features. They suggested that future studies could include more important features that directly affect employee turnover to increase the robustness of the model. Second, the authors mentioned that the model is restricted to only supervised machine learning, which takes a lot of computational time and requires new features to be added every time as an input, which takes a lot of human effort. They suggested that future studies could explore unsupervised machine learning techniques to overcome this limitation. Third, the authors suggested that future studies could explore the use of other ensemble methods such as Adaboost classifier to further improve the accuracy of the model. Finally, the authors suggested that future studies could deploy most data visualization between parameters and attrition to get a better idea of the factors that influence the most. Overall, the authors concluded that their proposed tree-based ensemble method is a promising approach for predicting employee attrition, but there is still room for improvement and further research in this area.	This study proposes a tree-based ensemble method for predicting employee attrition that outperforms previous studies that used different machine learning algorithms. The authors used two approaches, Random Forest and Gradient Boosting, and found that the Gradient Boosting classifier provides more accuracy in comparison to Random Forest. They also used several evaluation metrics to assess the performance of the models and ensure that they were accurate and reliable.
Chung, D., Yun, J., Lee, J., & Jeon, Y. (2023). Predictive model of employee attrition based on stacking ensemble learning. Expert Systems with Applications, 215, 119364.	The study focuses on predicting employee attrition using machine learning techniques. The researchers compare the performance of different models, including SVM, logistic regression, artificial neural networks (ANN), and XGBoost. They find that an ensemble model, combining random forest and ANN as base models with logistic regression as the meta learner, achieves the highest performance. The ensemble model outperforms individual models and shows excellent accuracy in predicting attrition. The study suggests that the ensemble approach can overcome the limitations of single models and create synergy through the combination of multiple models. Overall, the study highlights the effectiveness of machine learning in predicting employee attrition and the advantages of using ensemble models.	The dataset used in this study is the "IBM HR Analytics Employee Attrition & Performance data" published in Kaggle's Competitions and created by IBM data scientists. It consists of 1470 records and 31 variables, including 1 dependent variable (employee attrition) and 30 independent variables. The independent variables are categorized into personal attributes variables, job-related variables, and career-related variables. The dataset contains information on factors related to employee attrition, such as distance from home by job role and attrition, and average monthly income by education and attrition.	In this study, the dataset used for employee attrition prediction underwent several preprocessing steps. These steps were taken to ensure the accuracy and quality of the data used for training the machine learning models. First, exploratory data analysis was conducted to understand the structure and characteristics of the data. This helped in identifying any inconsistencies or missing values in the dataset. Next, certain variables were excluded from the model building process. The variables 'Standard Time', 'Over 18', and 'Employee Count' had the same observed values and were not informative for the prediction task. Additionally, the 'Employee Number' variable, which was an identification number, was also excluded. Categorical variables were encoded using one-hot encoding and label encoding techniques. One-hot encoding was used for categorical variables with three or more categories, mapping them to numerical values while maintaining their independence. For categorical variables with two categories, numerical mapping was done using 0 and 1. Numerical variables were normalized using the estimated mean and standard deviation of each variable. This normalization step was important because some machine learning models are sensitive to the magnitude of relative values between features. By performing these preprocessing steps, the dataset was prepared for training the machine learning models to predict employee attrition.	The feature selection techniques used in this study were not explicitly mentioned in the paper.	The feature engineering techniques used in this study were not explicitly mentioned in the paper.	In this study, the algorithm for the predictive model was selected through a comparison of five single algorithms and an ensemble model. The single algorithms used were Logistic Regression, Random Forest, XGBoost, SVM, and ANN. The ensemble model was constructed using a combination of these individual models. The performance of each model was evaluated in terms of accuracy, precision, recall, F-score, and AUC. Based on the comparison of these metrics, the best-performing model was selected as the final model for predicting employee attrition.	In this study, the algorithm was trained using the prepared dataset. The dataset was divided into three subsets: training, validation, and testing. The training data was used to build the machine learning model. The validation data was used to estimate the performance of the model when applied to new data. Finally, the test data was used to evaluate the best performing model. The performance of the models was compared in terms of accuracy, precision, recall, F-score, and AUC, and the best model was selected as the final model.	In this study, the ensemble technique was used to predict employee attrition. The researchers combined multiple machine learning models, including MLP, RF, LR, SMO, LWL, KStar, and KNN, using a stacking ensemble approach. The process involved training each individual model using the training data. Then, the predictions from these models were used as input variables for a meta learner, which was Logistic Regression in this case. The meta-learner was trained to make the final predictions for employee attrition. By combining the predictions of multiple models through the ensemble technique, the researchers were able to achieve higher performance and accuracy in predicting attrition compared to using a single model.	In this study, the hyperparameters of the machine learning models were tuned to optimize their performance. The specific hyperparameters for each model are listed in Table 2 of the study. The models used in the study were Logistic Regression, Random Forest, XGBoost, Support Vector Machine (SVM), and Artificial Neural Network (ANN). The hyperparameters that were tuned include penalty, C, tol, max_iter for Logistic Regression; max_depth, n_estimator, criterion, min_sample_split, min_samples_leaf for Random Forest; max_depth, eta, objective, early_stoppings, num_rounds for XGBoost; C, multi_class, max_iter, tol for SVM; and activation function, number of hidden layers, learning rate, epochs, max iteration, batch size for ANN.	In this study, the performance of the predictive models for employee attrition was evaluated using various metrics. These metrics include accuracy, precision, recall, F-score, and AUC (Area Under the Curve). By comparing the performance of each model in terms of these metrics, the best model was selected as the final model.	The benchmark used to evaluate the performance of the predictive models for employee attrition in this project was the comparison of prediction performance in terms of accuracy, precision, recall, F-score, and AUC of each model. The best performing model was selected as the final model based on these metrics.	The limitations of this study include: 1. Limited dataset: The study relies on a specific dataset from IBM analytics HR data, which may not fully represent the diversity of employee attrition patterns in other organizations or industries. Using a more diverse dataset could enhance the generalizability of the findings. 2. Model selection: The study focuses on ensemble machine learning algorithms, specifically Random Forest and Artificial Neural Network, combined with Logistic Regression as the meta learner. While these models have shown good performance, there may be other advanced machine learning techniques or algorithms that could be explored for better prediction accuracy. 3. Feature selection: The study utilizes a set of predefined features for predicting employee attrition. However, there may be additional relevant features that were not considered in this study. Exploring and incorporating new features could improve the predictive power of the models. 4. Interpretability: Ensemble models can be complex and difficult to interpret. Understanding the underlying factors and variables that contribute to employee attrition may be challenging with these models. Incorporating interpretability techniques or using simpler models could provide more transparent insights. Future directions for this study in terms of technology could include: Utilizing deep learning models: Deep learning models, such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs), could be explored to capture complex patterns and relationships in the data. These models have shown promising results in various prediction tasks.	The research contribution of the study is the proposal of a new ensemble-based prediction model. The study utilizes machine learning techniques such as Logistic Regression, Random Forest, XGBoost, SVM, and Artificial Neural Network to construct predictive models. Additionally, the study introduces a stacking-based ensemble model that combines the strengths of different algorithms to achieve higher performance in predicting employee attrition. This approach enhances the accuracy and effectiveness of attrition prediction using advanced technology.
Karande, S., & Shyamala, L. (2019). Prediction of employee turnover using ensemble learning. In Ambient Communications and Computer Systems: RACCCS-2018 (pp. 319-327). Springer Singapore.	This study focuses on the prediction of employee turnover using ensemble learning. Employee turnover is a significant problem in various industries, and predicting whether an employee will leave or stay can help organizations take necessary steps to control retention. The study proposes the use of ensemble learning, which combines multiple classification models and assigns weights based on their individual predicted accuracy. The ensemble model calculates a weighted average of the probabilities from each classifier to classify an employee. The study also discusses the use of under sampling to address imbalanced datasets and improve the sensitivity and specificity of the ensemble model. The results show that the ensemble model outperforms individual classifiers in terms of accuracy, precision, and specificity. Overall, the study highlights the importance of predicting employee turnover and demonstrates the effectiveness of ensemble learning in addressing this issue.	The dataset used in this study is the employee turnover dataset from IBM Data, which is used for a case study of HR Analytics. The dataset contains details of employee turnover and includes attributes such as job satisfaction, overtime, salary, distance from home, marital status, and employee's perception of fairness. The class attribute in the dataset represents employee attrition, with a value of 0 indicating that the employee did not leave and a value of 1 indicating that the employee left the company. The dataset is imbalanced, with a higher percentage of instances having the class variable "no" (employee will not leave) compared to instances with the class variable "yes" (employee will leave).	The preprocessing steps performed on the dataset used in this study were not mentioned in the paper.	The feature selection techniques used in this study are not explicitly mentioned in the paper.	The feature engineering techniques used in this study are not mentioned in the paper.	The process for selecting the algorithm used in this study involved applying diverse machine learning algorithms on the dataset to predict employee turnover. The algorithms used in this study included Support Vector Machines (SVM), decision trees, and ensemble learning models. The performance of each algorithm was evaluated, and the best performing algorithm was selected based on accuracy. The ensemble learning model, which is a combination of three weak learners, was also used to improve the prediction results. By using weighted average, the ensemble model provided a more reliable and accurate prediction of employee turnover.	The process used to train the algorithms in this study involved applying diverse machine learning algorithms on the dataset to predict employee turnover. The algorithms used included Support Vector Machines (SVM), Adaboost, Bagging, and Random Forest. Each algorithm was trained on the dataset and generated individual classifications. Weighted averages were then calculated from the probabilities of the individual classifications, and the final classification was determined based on the weighted average. This ensemble learning approach aimed to improve the performance and reliability of the model.	In this study, the ensemble technique was used to predict employee turnover. The researchers combined multiple classification algorithms, including Support Vector Machine, Logistic Regression, and Random Forest, to create an ensemble model. Each classification model was assigned a weight based on its individual predicted accuracy. The ensemble model then calculated the weighted average of the probabilities predicted by each classifier. Based on this weighted average, an employee could be classified as either likely to leave or likely to stay. By using the ensemble learning approach, the researchers aimed to improve the accuracy of the predictions and provide organizations with valuable insights to control employee retention.	In this study, the hyperparameter tuning was not explicitly mentioned. The focus of the study was on the implementation of an ensemble learning model for predicting employee turnover using various classifiers and techniques. The specific hyperparameters and their tuning were not discussed in the provided context.	The evaluation metrics used to assess the performance of the ensemble learning model in predicting employee turnover were accuracy, precision, sensitivity, and specificity. These metrics were calculated based on the confusion matrix, which compares the predicted class labels with the actual class labels. The accuracy is the ratio of the total number of correct predictions to the total number of predictions. Precision is the ratio of the predicted positive cases that are correct. Sensitivity is the ratio of the positive cases that are correctly identified. Specificity is the ratio of the negative cases that are correctly identified.	The benchmark used to evaluate the performance of the ensemble learning model in predicting employee turnover was a set of performance metrics. These metrics included accuracy, precision, sensitivity, and specificity. The results of the ensemble model were compared to those of other models such as support vector machine, logistic regression, and random forest. The ensemble model outperformed the other models in terms of accuracy, precision, and specificity. However, its sensitivity was slightly lower than that of the random forest model. Overall, the ensemble model showed better prediction accuracy compared to the other models.	The limitations of this study in terms of technology are not explicitly mentioned in the provided context. However, based on the information given, it can be inferred that the study focuses on the application of ensemble learning models and machine learning algorithms for predicting employee turnover. The scope of the study is to explore the use of these techniques to improve the accuracy and reliability of employee churn prediction. It does not specifically mention any limitations related to the technology used, but it is important to note that the effectiveness of the models and algorithms may depend on the quality and availability of the data, as well as the computational resources and software tools used for implementation.	The research contribution of the study is the application of ensemble learning as an advancement of traditional algorithms. The study proposes the use of ensemble machine learning algorithms to evaluate and decide the features that play a crucial role in predicting the risk of employee turnover. By combining multiple weak learners and using weighted average, the study aims to provide an improved system for predicting employee churn and giving HR professionals a heads-up to plan strategies to overcome turnover. This approach of using ensemble learning is a novel contribution to the field of employee attrition prediction.

TECHNOLOGICAL REVIEW

Dataset Selection

Dataset		
Citation of where to find the dataset	Description	Citations of work that have used that dataset
IBM HR Employee Attrition dataset	Benchmark dataset by IBM data scientists, 1470 instances, 35 features, commonly found in real-world employee databases.	Raza et al., 2022 Alshiddy & Aljaber, 2023 Qutub et al., 2021 Jain, 2017 Al-Darraji et al., 2021 Artelt & Gregoriades, 2023 Chung et al., 2023 Mehta & Modi, 2021
IBM Watson Analytics dataset	Open-source, 32 features, 1470 jobs, engineered knowledge. Balanced attrition categories.	Chakraborty et al., 2021
Collection of correlated data	1470 employees, numerical and categorical data, attributes include age, hourly rate, job involvement, etc. Prepared for analysis, visualization, and utilization using data cleaning and normalization.	Gupta et al., 2021
Real dataset from SAS library	Real dataset from SAS library, 35 variables/columns, 1.5k rows. Details of the dataset source or name not provided.	Alsheref, Fattoh, & Ead, 2022
Employee records dataset	From a mid-sized FMCG company, attributes include satisfaction, appraisal rating, projects, turnover status. Validated on a different FMCG employee dataset.	Srivastava & Eachempati, 2021
Blood glucose dataset	Collected from diabetic patients, two versions: CGM readings only and CGM readings with exogenous factors. Used for neural networks, evaluated using RMSE.	Kim et al., 2022
HR_comma_sep.csv from Kaggle	15,000 samples, 10 features: satisfaction, evaluation, project count, average monthly hours, etc. Converted non-numerical values to numerical.	Qadir et al., 2021
Kaggle HR Analytics and IBM HR Analytics datasets	Kaggle: 15,000 samples, IBM: 1,470 samples. Kaggle dataset has 9 features, IBM dataset has 34 features, including 11 selected features for the proposed model.	Yahia et al., 2021
Weekly reports dataset	Weekly reports of 97 employees for 6 months, 2172 reports, labeled as positive or negative. Used for Transformer Encoder.	Wu, 2022
Employee turnover dataset from IBM Data	Attributes include job satisfaction, overtime, salary, distance from home, marital status, perception of fairness. Class variable "0" for employees who stayed and "1" for those who left.	Karande & Shyamala, 2019

Data Preprocessing

Preprocessing			
Name	Pros	Cons	Citations
Data Cleaning	Removes noisy data, such as duplicates and inconsistencies, which can lead to more accurate predictions.	Data cleaning can be time-consuming, especially for large datasets.	(Raza, et al., 2022) (Gupta, et al., 2021)
	Enhances data quality by filling in missing values or handling outliers.	Decisions about how to handle missing values or outliers may introduce bias if not carefully considered.	(Jain, 2017) (Al-Darraji, 2021) (Kim, et al., 2022)
Feature Scaling	Scales features to a similar range, preventing some features from dominating others during model training.		(Gupta, et al., 2021)
	Can improve the convergence of optimization algorithms used in deep learning.	In some cases, feature scaling might not be necessary, and it could slightly increase computational overhead.	(Qadir, Noreen, and Shah, 2021)
Feature Encoding (Categorical Data):	Converts categorical variables into a numerical format that can be used by machine learning algorithms.	May introduce additional features and increase the dimensionality of the dataset (one-hot encoding).	(Alsheref, Fattoh and Ead, 2022) (Yahia, Hlel and Colomo-Palacios, 2021) (Artelt and Gregoriades, 2023)
	Allows models to work with a wider range of data types.	Depending on the encoding method, it can lead to a higher computational cost.	(Qadir, Noreen, and Shah, 2021) (Sharma et al., 2022) (Chung et al., 2023) (Qutub et al., 2021)
Feature Selection	Reduces the dimensionality of the dataset, which can lead to faster training times and improved model interpretability.	Feature selection may result in information loss if important features are mistakenly removed.	
	Focuses the model on the most relevant features, potentially leading to better predictive performance.	Selecting the right features can be challenging and may require domain knowledge.	(Alsheref, Fattoh and Ead, 2022)
Data Imputation (Handling Missing Values)	Fills in missing data, allowing you to use more complete datasets for training.	The imputation method used can introduce bias if not done carefully.	(Alshiddy, M.S., & Aljaber, B., 2023) (Qutub et al., 2021)
	Prevents models from discarding records with missing values, which can be valuable.	Imputed data may not fully capture the actual underlying distribution of missing values.	(Jain, 2017) (Karande and Shyamala, 2019)
Data Normalization	Scales features to have a common statistical distribution, which can improve convergence during training.	May not be necessary for all deep learning models or datasets.	(Alshiddy, M.S., & Aljaber, B., 2023) (Srivastava and Eachempati, 2021)
	Often used when applying deep learning models to standardize input data.	In some cases, normalization can lead to numerical instability.	(Kim, et al., 2022)
Handling Duplicated Values	Ensures data integrity: Removing duplicated values helps maintain the accuracy and consistency of the dataset.	Potential data loss: In some cases, removing duplicates might result in the loss of important information.	
	Reduces the potential for bias: Duplicates can skew the analysis and lead to incorrect results; eliminating them helps in achieving unbiased results.	Subjective decisions: Determining what constitutes a duplicate can sometimes be subjective and might require domain knowledge.	(Alshiddy, M.S., & Aljaber, B., 2023)
	Enhances the efficiency of algorithms: Removing duplicates can lead to faster computation and more efficient model training.	May not be suitable for all datasets: In some cases, duplicates might be intentional or convey meaningful information, making their removal inappropriate.	(Artelt and Gregoriades, 2023)

	Reduces noise: Removing irrelevant features can lead to a cleaner dataset, reducing the noise in the data and improving model performance.	Information loss: There's a risk of discarding potentially useful information, and in some cases, it might not be clear which features are truly irrelevant.	
	Faster computation: Fewer features mean quicker model training and prediction.	Requires domain knowledge: Identifying irrelevant features often relies on domain expertise, and automated methods may not always make the right choices.	
Removing Irrelevant Features	Improved interpretability: Simplifying the dataset can make it easier to understand and interpret the model's results.	Impact on model accuracy: In some cases, removing features might have unintended consequences and negatively impact the model's performance.	(Alshiddy, M.S., & Aljaber, B., 2023)
	Improved data quality: Addressing outliers can lead to cleaner and more reliable data.	Subjective decisions: Deciding how to handle outliers can be subjective and may depend on the specific context.	
	Enhanced model performance: Outliers can distort models; their removal or transformation can improve model accuracy.	Data loss: Removing outliers can result in the loss of potentially valuable information or signal.	
Dealing with Outliers	More robust analysis: Outlier handling can result in more robust and resilient statistical analyses.	Impact on generalization: Extreme outliers might be rare but can be important for modeling the tails of a distribution, which might affect the model's generalization ability.	(Jain, 2017)
	Adapts data for modeling: Transformation techniques can make the data more suitable for machine learning algorithms.	Complex process: Feature transformation can be complex and might require a deep understanding of the data and domain knowledge.	
	Can reveal underlying patterns: Transformations can bring out hidden patterns and relationships in the data.	Overfitting risk: Some transformations might lead to overfitting if not applied judiciously.	
Feature Transformation	Enhanced model performance: Transformation can improve the performance of certain models, especially linear models.	May not always improve results: Not all datasets benefit from feature transformation, and in some cases, it may not lead to better model performance.	(Srivastava and Eachempati, 2021)
	Enhances data quality: Identifying and correcting incorrect attributes helps maintain the quality and integrity of the dataset.	Manual inspection: Correcting incorrect attributes often requires manual inspection and data cleansing efforts.	(Jain, 2017)
	Improved model performance: Ensuring data accuracy can lead to more accurate model predictions.	Time-consuming: The process of checking and rectifying incorrect attributes can be time-intensive.	
Checking Incorrect Attributes	Increases trust in results: Accurate data instills confidence in the model's outputs and conclusions.	Potential data loss: Overzealous correction can lead to the loss of legitimate data or introduce new errors if not done carefully.	(Mehta and Modi, 2021)

Feature Selection

Feature Selection			
Name	Pros	Cons	Citations of work
Filter Methods	Fast and computationally efficient. Independent of the machine learning model. Can provide insights into the relationships between features and the target variable.	May not consider feature interactions. May overlook features that are relevant only in combination with other features.	(Raza, et al., 2022) (Alshiddy, M.S., & Aljaber, B., 2023) (Qutub, A., et al., 2021)
Wrapper Methods	Tailored to the specific machine learning model, optimizing feature selection for model performance. Can handle feature interactions. Provides a robust approach for identifying the best subset of features.	Computationally expensive, especially for deep learning models and large datasets. May lead to overfitting if not used carefully.	(Chakraborty et al., 2021) (Alsheref, Fattoh and Ead, 2022)
Embedded Methods	Feature selection is integrated into the model training process. Efficient and can select relevant features while training the model. Effective for deep learning models with built-in regularization techniques.	Limited to the specific algorithm's feature selection capabilities. May not be as flexible as standalone methods in terms of feature selection strategies.	(Al-Darraji, 2021)
Recursive Feature Elimination (RFE)	Iteratively removes the least important features, resulting in an ordered feature ranking. Works well with models that provide feature importances. Allows you to specify the desired number of features to keep.	Can be computationally expensive for large datasets and deep learning models. The choice of the number of features to keep can impact the final subset.	(Yahia, Hlel and Colomo-Palacios, 2021)
L1 Regularization (Lasso)	Encourages sparse feature selection by pushing some feature coefficients to zero. Suitable for linear models and can be effective in reducing feature dimensionality. May work well for high-dimensional data.	Less effective for non-linear models or when the relationships between features and the target are not strictly linear. The choice of regularization strength (lambda) can be a hyperparameter to tune.	(Chakraborty et al., 2021)
Random Forest Feature Importance	Provides feature importances based on the decrease in impurity when a feature is used for splitting in decision trees. Fast and interpretable for ensemble models like random forests. Can handle non-linear relationships between features and the target.	May not be as effective for deep learning models, as they have different mechanisms for determining feature importance. Feature importance scores can be biased in the presence of correlated features.	(Yahia, Hlel and Colomo-Palacios, 2021)
Mutual Information	Measures the dependence between a feature and the target, considering non-linear relationships. Suitable for various machine learning models. Can handle feature interactions.	Can be computationally intensive for high-dimensional data. May require binning of continuous features.	(Raza, et al., 2022)
Principal Component Analysis (PCA)	Linear dimensionality reduction technique that creates new uncorrelated features (principal components). Can reduce feature dimensionality while preserving as much variance as possible.	PCA does not consider the target variable when selecting features, which may result in loss of predictive power. Works well when feature interpretation is not critical but might be less interpretable.	(Alsheref, Fattoh and Ead, 2022) (Qutub, A., et al., 2021)

Feature Selection with Expert Knowledge	Domain Expertise: Expert knowledge can provide valuable insights into the domain-specific relevance of features, helping to select the most important ones based on the problem at hand.	Subjectivity: Expert-driven feature selection is subjective and may introduce bias based on the expert's assumptions and perspectives, potentially leading to a biased or limited feature set.	(Srivastava and Eachempati, 2021)
		Lack of Data-Driven Validation: The approach may lack rigorous data-driven validation, making it challenging to demonstrate the effectiveness of the selected features objectively.	
	Customization: Experts can tailor feature selection to the specific needs of the problem, ensuring that the chosen features align with the objectives of the analysis.	Potential for Overlooking Features: Depending solely on expert knowledge may lead to the oversight of features that might have predictive value but are not apparent to the expert.	

Feature Engineering

Feature Selection			
Name	Pros	Cons	Citations
	Identifies highly correlated and potentially redundant features, helping to reduce dimensionality and noise in the dataset.	May lead to information loss if important but weakly correlated features are removed.	
	Provides insights into feature relationships, which can be valuable for data interpretation and model performance.	Subjective in nature, as setting correlation thresholds involves human judgment.	
Data Correlation Analysis	Supports feature selection by highlighting the most relevant features based on their correlation with the target variable.	Limited to linear relationships and may not capture complex feature interactions.	(Raza, et al., 2022)
	Incorporates temporal information, which can be valuable in modeling time-dependent patterns.	May not be suitable for all datasets, especially those with non-time-related problems.	
		Increases the dimensionality of the data, potentially leading to computational challenges.	
Adding Time-Series Information	Enhances the dataset with additional context, allowing models to capture time-based trends and seasonality.	Requires domain expertise to determine the relevant time-series features.	(Kim, D., et al. 2022)
	Reduces dimensionality and multicollinearity, which can improve model performance and interpretation.	May discard potentially useful information if redundant features are incorrectly identified.	
	Eliminates potentially irrelevant or highly correlated features, simplifying the dataset.	The decision to drop variables is subjective and may require domain knowledge.	
Dropping Redundant Variables	Speeds up computation by working with a smaller feature set.	Can introduce bias if not done carefully, as feature relevance can vary across different tasks.	(Jain, D. 2017)
	Converts categorical variables into numerical format, making them suitable for many machine learning algorithms.	Increases dimensionality, especially with one-hot encoding, which may lead to issues with high cardinality.	
	Expands the range of models that can be applied to the dataset.	May introduce multicollinearity, particularly when using one-hot encoding.	
Data Type Conversion	Can improve the performance of models by handling categorical data effectively.	Requires careful consideration of the encoding method and possible trade-offs.	(Qutub, A., et al. (2021)
	Addresses class imbalance issues in the dataset, leading to better model training and predictions.	May not be necessary for balanced datasets and could introduce noise if overused.	
	Reduces the risk of the model being biased towards the majority class.	Imbalanced datasets may require specialized techniques beyond simple data balancing.	
Data Balancing	Enhances the model's ability to identify patterns in the minority class.	The choice of balancing method should be made carefully based on the specific dataset and problem.	(Alshiddy, M.S., & Aljaber, B., 2023)
	Converts non-numerical categorical variables into numerical format, making them usable in machine learning models.	Assumes linearity of the converted values, which may not be suitable for all features.	(Chakraborty et al., 2021)
	Enables models to work with a wider range of features and data types.	May not capture complex relationships in the original categorical data.	
Quantitative Value Conversion	Simplifies data preprocessing by providing consistent data types.	One-hot encoding can significantly increase dimensionality in the dataset.	(Gupta, et al., 2021)

	Adapts data for modeling, making it more suitable for machine learning algorithms.	Feature transformation can be complex and may require a deep understanding of the data and domain knowledge.	
	Can reveal hidden patterns and relationships in the data.	Some transformations might lead to overfitting if not applied judiciously.	
Feature Transformation	Improves the performance of certain models, especially linear ones.	Not all datasets benefit from feature transformation, and it may not always improve model performance.	(Alsheref, Fattoh and Ead, 2022)
	Converts categorical variables into a format that can be easily used by machine learning models.	Increases dimensionality significantly, especially with high-cardinality categorical variables.	
	Preserves all categories and information within categorical features.	May introduce multicollinearity, as each new binary column is correlated with others.	
One-Hot Encoding	Widely supported by many machine learning libraries.	Can be computationally expensive when applied to large datasets.	(Raza, et al., 2022)
	Selects the most relevant features based on their information gain with the target variable.	Can be computationally intensive when working with large datasets.	
	Provides an objective and data-driven method for feature selection.	The selection criteria can be subjective, as the choice of threshold for information gain requires judgment.	
Information Gain-Based	Reduces dimensionality and improves model efficiency.	May not work well for all models, and performance can vary based on the dataset.	(Alshiddy, M.S., & Aljaber, B., 2023)
	Incorporates domain expertise into the feature engineering process, leading to more meaningful and contextually relevant features.	Subjective and dependent on the availability of domain experts.	
	Enhances the interpretability of features and model results.	Expert-driven feature engineering can introduce bias and may not always align with the data-driven results.	
Feature Engineering with Expert Knowledge	Customizes feature engineering to the specific needs of the problem.	Potential data loss if overzealous feature engineering removes useful information or introduces errors.	(Srivastava and Eachempati, 2021)
	Puts all features on the same scale, preventing some features from dominating others.	Scaling may not be necessary for all datasets and models, especially those robust to feature scales.	(Alshiddy, M.S., & Aljaber, B., 2023)
	Improves model training and performance, particularly for algorithms sensitive to feature scales.	May amplify noise in the data, particularly if the dataset contains outliers.	
Data Scaling	Reduces the risk of numerical overflow problems.	The choice of scaling method should align with the characteristics of the data.	(Qutub, A., et al., 2021)
	Relevance: Ensures that only the most relevant features are retained, potentially improving model interpretability and efficiency.	Subjectivity: The selection process may be subjective, relying on domain expertise or assumptions about feature importance.	
	Dimensionality Reduction: Reduces the complexity of the dataset by focusing on a subset of critical features.	Information Loss: May result in the exclusion of potentially valuable features, leading to information loss.	
Specific Features Selection	Improved Model Performance: Can lead to better model performance by excluding irrelevant or redundant information.	Model Dependency: Effectiveness can vary depending on the machine learning algorithm used.	(Artelt and Gregoriades, 2023)

Label Encoding	Simplicity: Label encoding is a straightforward method, easy to implement across various types of categorical data.	Assumption of Ordinality: Imposes an ordinal relationship on categorical data that might not inherently possess such a relationship.	(Sharma et al., 2022)
	Memory Efficiency: Consumes less memory compared to one-hot encoding, which can be beneficial for large datasets.	Misleading Model: For algorithms that interpret numerical values as ordinal, label encoding might introduce unintended relationships.	
	Preservation of Order: Retains the ordinal relationship between categories, which can be important for certain algorithms.	Limited Applicability: Not suitable for categorical variables without a clear ordinal relationship.	
Correlation Analysis	Feature Selection: Identifies and removes highly correlated features, reducing redundancy in the dataset.	Causation vs. Correlation: Correlation does not imply causation; high correlation does not necessarily mean one variable causes the other.	(Sharma et al., 2022)
	Improved Model Interpretability: Aids in understanding the relationships between different variables in the dataset.	Selective to Linear Relationships: Primarily captures linear relationships; may not detect nonlinear associations between variables.	
	Enhanced Model Performance: Removing highly correlated features can prevent multicollinearity, which can improve certain models.	Threshold Dependency: Setting correlation thresholds is somewhat arbitrary and may impact results.	
Satisfaction Calculation	Subjective Measure: Reflects the emotional tendencies of employees, providing a nuanced view of satisfaction.	Subjectivity: The calculation of satisfaction may be subjective and influenced by the interpretation of emotional tendencies.	(Wu, 2022)
	Individualized: Can be personalized, considering individual differences in the interpretation of satisfaction.	Data Availability: Requires data on emotional tendencies, which may not always be readily available or measurable.	
	Integration with Models: Can be seamlessly integrated into models as a feature, potentially improving predictive accuracy.	Interpretability: The interpretation of calculated satisfaction scores might vary, making it challenging to standardize.	

Algorithm Selection Process

Algorithm Selection			
Algorithm Name	**Pros**	**Cons**	**Citations**
Logistic Regression	Simple and interpretable.	Limited in capturing complex non-linear relationships.	(Raza, et al., 2022)
	Suitable for binary classification.		(Al-Darraji, et al., 2021)
	Provides probabilities of class membership.	May not perform well when features are not independent.	(Chakraborty et al., 2021)
	Works well when the relationship between features and the target variable is linear.		(Gupta, et al., 2021)
SVM	Effective for high-dimensional data and non-linear problems.	Choice of kernel and hyperparameters can be challenging.	(Raza, et al., 2022)
	Works well with clear margin of separation.		
	Can handle large feature spaces.	Doesn't provide probabilities by default.	(Al-Darraji, 2021)
Random Forest	Robust and handles non-linearity well.	Tends to overfit with deep trees.	(Raza, et al., 2022)
			(Chakraborty et al., 2021)
	Suitable for high-dimensional data.		(Alsheref, Fattoh and Ead, 2022)
		Can be computationally expensive for large datasets.	(Gupta, et al., 2021)
	Provides feature importance scores.		(Alshiddy and Aljaber, 2023)
			(Qutub et al., 2021)
	Resistant to overfitting.		(Jain, 2017)
Decision Trees	Simple, interpretable, and can handle categorical data.	Prone to overfitting, especially with deep trees.	(Chakraborty et al., 2021)
			(Alshiddy and Aljaber, 2023)
			(Qutub et al., 2021)
	Suitable for both classification and regression tasks.	Sensitive to small changes in data, which can lead to different tree structures.	(Alsheref, Fattoh and Ead, 2022)
			(Jain, 2017)
K-Nearest Neighbors (KNN)	Intuitive and simple to implement.	Sensitive to the choice of k (number of neighbors).	(Chakraborty et al., 2021)
	Works well for small to moderately sized datasets.	Computationally expensive for large datasets.	(Qutub et al., 2021)
	No training phase, making it easy to adapt to new data.	Doesn't handle imbalanced data well.	(Alsheref, Fattoh and Ead, 2022)
			(Jain, 2017)
Gradient Boosting Classifier	Effective ensemble method that improves model accuracy.	Can be computationally expensive and sensitive to overfitting.	(Raza, et al., 2022)
	Resistant to overfitting.	Requires tuning of hyperparameters.	
	Handles non-linear relationships well.		
Latent Dirichlet Allocation	Effective for topic modeling in text data.	Specific to text data and topic modeling tasks.	(Gupta, et al., 2021)
	Identifies hidden topics within text documents.	Requires domain knowledge to interpret topics.	
Extra Tree Classifier	Robust against overfitting and noise in data.	May not be as accurate as other methods.	(Alshiddy and Aljaber, 2023)
	Efficient and computationally inexpensive.	Limited feature selection capabilities.	

	Excellent predictive performance and efficient implementation.	Requires tuning of hyperparameters.	
	Handles missing data well.		
Gradient Boosting (XGBoost)	Supports parallel processing for faster training.	Can be memory-intensive for large datasets.	(Alsheref, Fattoh and Ead, 2022)
	Suitable for complex non-linear relationships and deep learning.	Requires large amounts of data and computational resources.	
	Effective for tasks like image recognition and natural language processing.	Sensitive to the choice of architecture and hyperparameters.	
Artificial Neural Networks	Can learn hierarchical features.	Can be challenging to interpret.	(Alsheref, Fattoh and Ead, 2022)
	Effective for sequential data and time series.	Sensitive to vanishing/exploding gradient problems in deep RNNs.	
	Can capture temporal dependencies.		
Recurrent Neural Networks (RNNs)	Suitable for tasks like natural language processing and speech recognition.	Limited in capturing long-term dependencies.	(Kim, et al., 2022)
	Can optimize ensemble weights for model combination.	May be computationally intensive for large datasets.	
	Effective for feature selection and hyperparameter tuning.		
Genetic Algorithm (GA)	Handles high-dimensional search spaces.	Requires fine-tuning of parameters.	(Kim, et al., 2022)
	Sequential Data Processing: Effective for sequential data and time series, making it suitable for tasks like natural language processing (NLP) and sentiment analysis.	Computational Resources: Requires substantial amounts of data and computational resources.	(Qadir, Noreen, and Shah, 2021)
	Capturing Long-Term Dependencies: Capable of capturing long-term dependencies in data sequences.	Hyperparameter Sensitivity: Sensitive to the choice of architecture and hyperparameters.	
Bi-LSTM (Deep Learning)	Hierarchy of Features: Can learn hierarchical features in data.	Interpretability: Challenging to interpret due to the complex structure.	(Al-Darraji, et al., 2021)
	Improved Accuracy: Effective ensemble method that can significantly enhance model accuracy.	Computational Complexity: Can be computationally expensive, especially with a large number of diverse models.	
	Resistant to Overfitting: Generally resistant to overfitting when compared to individual models.	Hyperparameter Tuning: Requires tuning of hyperparameters for both individual models and the ensemble.	
VC Ensemble Method	Diversity of Models: Combines predictions from multiple models, each contributing its unique perspective.	Interpretability: Interpretability might be compromised due to the combination of various models.	(Yahia, Hlel and Colomo-Palacios, 2021)

	Interpretability: Decision trees are simple and interpretable, aiding in understanding the model's decision-making process.	Overfitting: Prone to overfitting, especially with deep trees.	
	Handles Categorical Data: Can handle both numerical and categorical data.	Sensitive to Data Changes: Small changes in data can lead to different tree structures.	
C4.5 Classifier	Suitable for Both Tasks: Applicable for both classification and regression tasks.	May Not Capture Complex Relationships: May struggle to capture complex non-linear relationships.	(Qadir, Noreen, and Shah, 2021)
	Handles Categorical Features: Specifically designed to handle categorical features efficiently.	Hyperparameter Tuning: Requires tuning of hyperparameters.	(Yahia, Hlel and Colomo-Palacios, 2021)
	Efficient Implementation: Known for its efficient implementation and high predictive performance.		
CAT Boost	Parallel Processing: Supports parallel processing for faster training.	Memory-Intensive: Can be memory-intensive, especially for large datasets.	(Artelt and Gregoriades, 2023)
	Interpretability: Decision trees are simple and interpretable.	Overfitting: Prone to overfitting, especially with deep trees.	
	Handles Categorical Data: Can handle both numerical and categorical data.	Sensitive to Data Changes: Small changes in data can lead to different tree structures.	
CART (Classification and Regression Trees)	Feature Importance: Provides feature importance scores.	May Not Capture Complex Relationships: May struggle to capture complex non-linear relationships.	(Sharma et al., 2022)

Algorithm Training Process

Algorithm Training Process			
Method	Pros	Cons	Citations
Train-Test Split	Simple and easy to implement.	May not capture the variability in data due to a single random split.	(Wu, 2022)
	Provides a quick estimate of model performance.	Sensitivity to the specific data split.	(Mehta and Modi, 2021) (Chung et al., 2023)
Cross-validation	Provides a more robust estimate of model performance by averaging over multiple data splits.	Can be computationally intensive, especially with a large number of folds.	(Qadir, Noreen, and Shah, 2021)
	Reduces the impact of data variability.	May still be sensitive to the randomness in data splitting.	(Yahia, Hlel and Colomo-Palacios, 2021)
Ensemble Learning with Autotuning	Optimizes hyperparameters, enhancing model performance.	Computational complexity may increase, especially with large datasets.	(Alsheref, Fattoh & Ead, 2022)
	Can capture complex relationships in data through ensemble learning.	Requires careful tuning of autotuning parameters.	
Ensemble Weight Optimization	Effective in combining predictions from multiple models.	Requires careful hyperparameter tuning for optimal results.	(Kim et al., 2022)
	Enhances overall model performance through optimized ensemble weights.	Computational complexity may increase with the number of models.	
Train-Test-Split Method with Genetic Algorithm	Allows for real-time prediction updates.	Requires substantial amounts of data for effective genetic algorithm optimization.	(Wu, 2022)
	Incorporates genetic algorithms for efficient optimization.	Continuous model improvement may demand real-time computational resources.	
Supervised Learning with Multiple Classifiers	Utilizes the strengths of different classifiers.	Ensemble models may require careful tuning of individual classifiers.	(Qutub, A., et al., 2021)
	Can provide more robust predictions through ensemble methods.	Computational complexity may increase, especially with a large number of classifiers.	

Ensemble Technique

Ensemble Technique			
Ensemble Technique	Pros	Cons	Citations
Extra Trees Classifier (ETC)	Handles high-dimensional datasets effectively. Less prone to overfitting compared to other decision tree-based algorithms.	Increased computational complexity.	(Raza, et al., 2022)
Random Forest	Robust to overfitting. Handles high-dimensional datasets well.	Can be computationally expensive, especially with a large number of trees.	(Chakraborty et al., 2021) (Gupta, et al., 2021) (Jain, 2017)
Stacking and Voting	Improved accuracy through meta-level predictions.	Increased complexity in model interpretation.	(Alshiddy, M.S., & Aljaber, B., 2023)
Majority Voting	Simple implementation, effective in some cases	May not perform well in cases of highly imbalanced data	(Alsheref, Fattoh and Ead, 2022)
Random Forest and Gradient Boosting Ensemble	Improved accuracy through combining strengths	Increased computational demands	(Srivastava and Eachempati, 2021)
Decision Tree + Random Forest + Gradient Boosting Ensemble	Improved accuracy through combining strengths	May introduce complexity in model interpretation	(Qutub et al., 2021)
Boosting, Bagging, and Stacking	Improved accuracy through combining diverse models	Requires careful tuning	(Jain, 2017)
Ensemble Weight Optimization with Genetic Algorithm	Derives robust models through optimized variable weights	Increased computational complexity	(Kim, et al., 2022)
Voting Classifier (VC)	Improves accuracy through majority voting	Performance depends on the diversity of base classifiers	(Yahia, Hlel and Colomo-Palacios, 2021)
Logistic Regression, Random Forest, and XGBoost Ensemble	Combines different algorithms for robust predictions	May require fine-tuning of hyperparameters	(Artelt and Gregoriades, 2023)
Ensemble of Rule-based Classifiers	Improves performance through combining diverse models	Interpretability may be compromised	(Sharma et al., 2022)
Stacking Ensemble	Achieves higher performance compared to single models	Increased complexity in model interpretation	(Chung et al., 2023)
Support Vector Machine, Logistic Regression, and Random Forest Ensemble	Improved accuracy through combining different classifiers	Performance may vary based on the diversity of classifiers	(Karande and Shyamala, 2019)

Hyperparameter Tuning

Hyperparameter Tuning			
Hyperparameter Tuning	Pros	Cons	Citations
Randomized Grid Search	Efficient tuning results Good accuracy scores	Increased computational complexity	(Raza, et al., 2022)
10-Fold Cross-Validation	Robust optimization Helps prevent overfitting	Requires more computational resources	(Chakraborty et al., 2021)
Specific Hyperparameter Tuning for NB, SVM, RF	Tailored optimization for each algorithm Improved algorithm performance	Manual tuning may be time-consuming	(Alshiddy, M.S., & Aljaber, B., 2023)
Autotuning Technique	Automated hyperparameter selection Finds optimal combination of hyperparameters	May not capture domain-specific nuances	(Alsheref, Fattoh and Ead, 2022)
Tuning 'mtry' and 'ntree' for RF, Standard Techniques for Others	- Fine-tuned parameters for RF - Standard tuning techniques for other models	- Lack of detailed information on other model tuning	(Srivastava and Eachempati, 2021)
Grid Search Approach	- Systematic exploration of hyperparameter space - Comprehensive tuning	- Computationally expensive for large parameter spaces	(Qutub, A., et al., 2021)
Trial-and-Error	- Empirical determination of optimal parameters	- Time-consuming and resource-intensive	(Kim, et al., 2022)
Grid Search Approach	- Comprehensive search for optimal hyperparameters - Well-suited for small parameter spaces - Identifies optimal configuration	- Computationally expensive for large parameter spaces	(Al-Darraji et al., 2021) (Yahia, Hlel and Colomo-Palacios, 2021) (Qadir, Noreen, and Shah, 2021)
Randomized Search Cross-Validation (Random CV)	- Efficient search for optimal hyperparameters - Reduces search space through random sampling	- May not explore the entire hyperparameter space	(Sharma et al., 2022)
Grid Search Approach for RF, No Details for GB	- Systematic exploration for optimal RF hyperparameters - Lack of details for GB tuning	- Computational cost for large parameter spaces	(Mehta and Modi, 2021)

Evaluation Metrics

Evaluation Metrics			
Evaluation Metrics	Pros	Cons	Citations
Accuracy, Precision, Recall, F1-score	Comprehensive evaluation of model performance, considering aspects like correctness, precision, recall, and balance between precision and recall.	Limited information for specific types of errors.	(Raza, et al., 2022) (Chakraborty et al., 2021) (Gupta, et al., 2021) (Alshiddy and Aljaber, 2023) (Alsheref, Fattoh and Ead, 2022) (Srivastava and Eachempati, 2021) (Qutub et al., 2021)
Gini Coefficient, Misclassification Rate, etc.	Diverse metrics assessing discrimination, overall error, effectiveness, and balance between precision and recall.	Interpretability may be challenging, especially for less common metrics.	(Alsheref, Fattoh and Ead, 2022)
RMSE, R-squared	Quantifies the difference between predicted and actual values, assessing model fit to data.	Limited interpretability for non-experts.	(Srivastava and Eachempati, 2021)
Area Under the ROC Curve (AUC-ROC)	Evaluates models on the trade-off between true positive and false positive rates.	May not be suitable for imbalanced datasets.	(Qutub et al., 2021) (Yahia, Hlel and Colomo-Palacios, 2021)
Clinical Accuracy Grid-Error Grid Analysis	Measures predictive accuracy and clinical relevance, particularly for healthcare applications.	Requires domain-specific understanding.	(Kim, et al., 2022)
Gini Coefficient, Misclassification Rate, etc.	Assesses discrimination, overall error, and effectiveness for model selection.	Metrics may require specific domain knowledge for interpretation.	(Alsheref, Fattoh and Ead, 2022)
RMSE, ROC Curve	Measures classification performance and probability prediction assessment.	Limited interpretability for non-experts.	(Sharma et al., 2022)
Accuracy	Primary metric for assessing model correctness.	May not capture nuances in model performance.	(Wu, 2022)
Accuracy, Precision, Recall, F1 Score, AUC	Comprehensive evaluation metrics to provide a holistic view of model performance.	Choice of hyperparameters may influence results.	(Mehta and Modi, 2021)
Accuracy, Precision, Recall, F1 Score, AUC	Comprehensive metrics for assessing model performance and selecting the best model.	Results may vary based on the algorithm and data characteristics.	(Chung et al., 2023)
Accuracy, Precision, Sensitivity, Specificity	Standard metrics for model evaluation using confusion matrix-based calculations.	May not capture specific nuances in performance.	(Karande and Shyamala, 2019)

Benchmarking

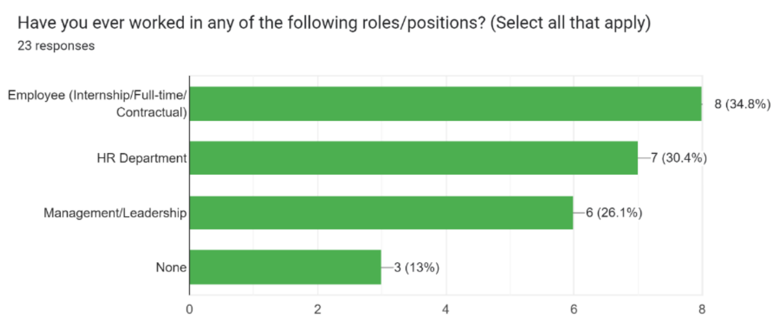
Benchmarking			
Benchmarking	Pros	Cons	Citations
ETC vs. Decision Tree, Logistic Regression, etc.	Outperformed other models in terms of accuracy (93%)	Limited insight into specific cons of benchmarked models	(Raza et al., 2022)
Random Forest vs. Baseline Model	Demonstrated superiority over baseline model (99% accuracy)	Lack of detailed explanation of the benchmarking process	(Chakraborty et al., 2021)
Logistic Regression vs. KNN vs. Random Forest	Comparative assessment aiding model selection (Logistic Regression selected)	Focused on a limited set of models	(Gupta et al., 2021)
Proposed Model vs. Naive Bayes, SVM, RF	Superiority demonstrated through statistical analysis (higher AUC)	Benchmarking limited to specific algorithms	(Alshiddy & Aljaber, 2023)
Multiple Algorithms (SVM, Naïve Bayes, etc.)	Comprehensive benchmarking involving multiple steps	Lack of specific details about the individual models' performance	(Alsheref, Fattoh, and Ead, 2022)
Random Forest, Gradient Boosting, Ensemble	Ensemble approach outperformed individual models (accuracy improvement)	Limited information on specific benchmarking metrics	(Srivastava and Eachempati, 2021)
No direct comparison with other studies/models	No specific benchmarking details provided	Lack of comparative insights with other studies	(Qutub et al., 2021)
Various ML Algorithms (KNN, GLM, SVM, etc.)	Comparative analysis for model selection (Adaptive boosting identified)	Detailed evaluation metrics not presented	(Jain, D. 2017)
RMSE and Clinical Accuracy Grid-Error Grid	Specific metrics for evaluating model performance	Limited information on other potential benchmarking metrics	(Kim et al., 2022)
State-of-the-art methods (SVM, Random Forest, etc.)	Comparison against established models in terms of accuracy, precision, recall	Limited information on specific benchmarking process	(Al-Darraji et al., 2021)
Comparison with MLP and Naïve Bayes	Superiority demonstrated over other models (Bi-LSTM outperformed)	Benchmarking focused on specific models	(Qadir, Noreen, and Shah, 2021)
Kaggle and IBM HR Analytics datasets	External validity assessment through benchmark datasets	Lack of detailed benchmarking process description	(Yahia, Hlel and Colomo-Palacios, 2021)
Logistic Regression, Random Forest, XGBoost	Multiple classifiers assessed; department-specific insights	Specific features used in analysis mentioned	(Artelt and Gregoriades, 2023)
Random Forest vs. Gradient Boosting	Comparison between models for performance evaluation	Lack of information on specific benchmarking metrics	(Mehta and Modi, 2021)
Multiple models (accuracy, precision, recall, etc.)	Selection of the best model based on performance metrics	Benchmarking process details not explicitly provided	(Chung et al., 2023)
Ensemble Learning vs. SVM, Logistic Regression, etc.	Outperformed other models in terms of accuracy, precision, specificity	Lower sensitivity compared to Random Forest	(Karande and Shyamala, 2019)

APPENDIX B: SOFTWARE REQUIREMENT SPECIFICATION

SURVEY

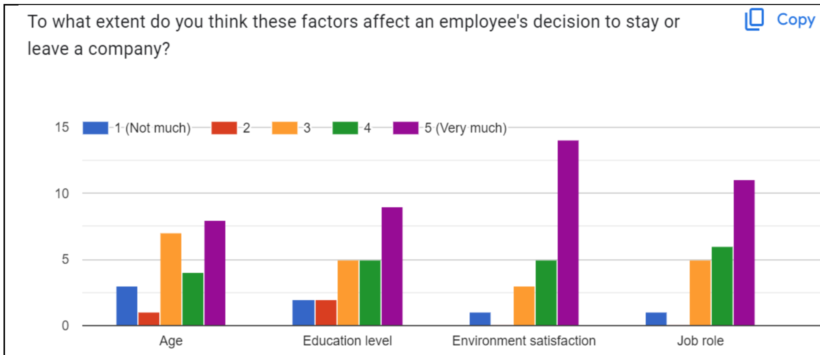
Link to the Google Form:
https://docs.google.com/forms/d/e/1FAIpQLSf_NOyD3SxpNXSUg6IVVG0lmhs4b6r3qCof2jiFksQSdUuB1w/viewform

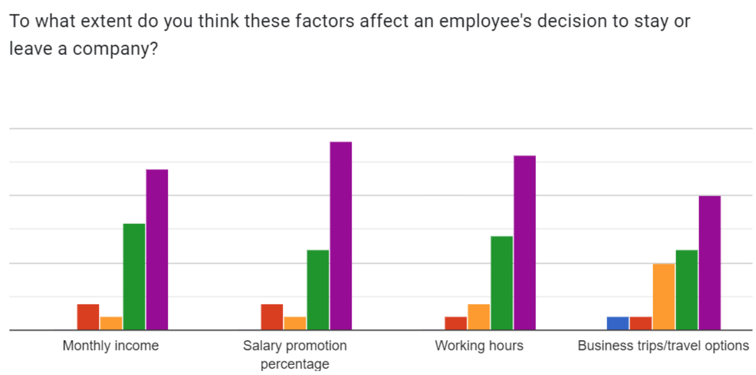
Question	**Have you ever worked in any of the following roles/positions?**
Aim of Question	To understand the context and background of the participant.
Findings and Conclusions	

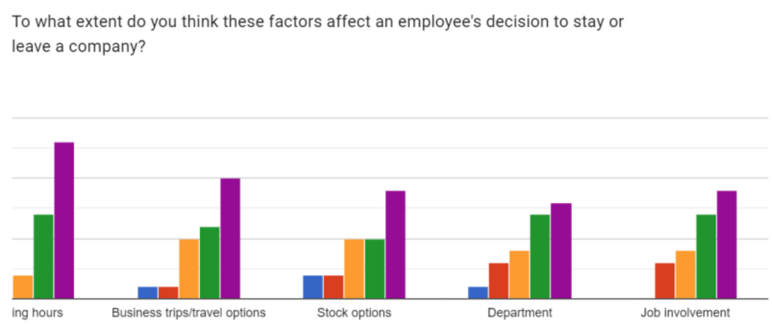


Most of the participants had experience working as an employee, as HR personnel or the leadership of a company. Therefore, it can be established that these individuals have some experience in the industry and its working conditions and the context behind the domain either first-hand or otherwise. 34.8% had employee experience, either in the form of an internship, a full-time position or a contractual position. 30.4% had experience working in the HR department of a company. 26.1% had experience working in management or leadership of an organization and 13% had no prior work experience.

Question	**To what extent do you think these factors affect an employee's decision to stay or leave a company?**
Aim of Question	To gauge the public's current perception of attrition and the factors they believe contribute to it.
Findings and Conclusions	

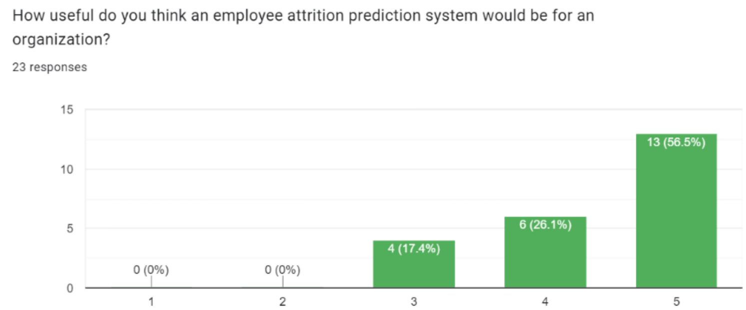




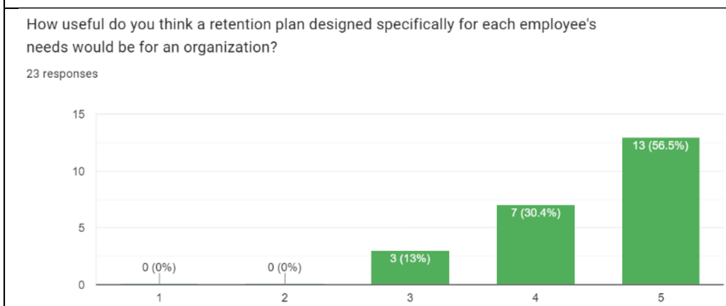


This question was intended to identify the features that, according to public sentiment, mattered the most to an employee's decision to stay or leave a company. Some of the features, such as monthly income, working hours and salary promotion percentage, were relatively homogenous, where the participants responded, almost unanimously, towards being higher contributors of attrition. Opinions were divided on features such as age. Most other features were balanced and align with a common consensus of a neutral to positive opinion.

Question	**How useful do you think an employee attrition prediction system would be for an organization?**
Aim of Question	To identify the opinions of the participants regarding how useful the system might be for an organization.
Findings and Conclusions	

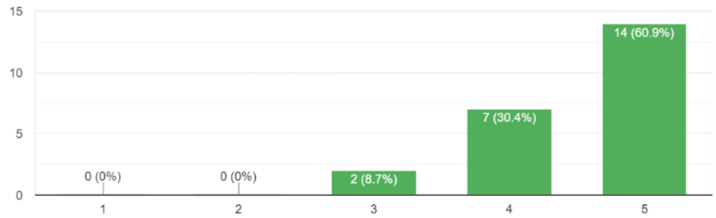


The results for this question aligned homogenously towards positive. The majority of the respondents (56.5%) have strongly rated the statement to be true.

Question	**How useful do you think a retention plan designed specifically for each employee's needs for an organization?**
Aim of Question	To identify the opinions of the participants regarding how useful creating a customized retention plan based on the distinct combination of features affecting each employee might be for an organization.
Findings and Conclusions	
 The responses to this question were also largely positive meaning that the public opinion believes that such a feature would be of decent use to a company.	
Question	**How much do you think employees would benefit from having organizations work to improve their work environments, work life balance, promotions, job satisfaction, and overall employee experience?**
Aim of Question	To identify the opinions of the respondents regarding how beneficial it would be, for employees, to have companies working to improve their retention levels and retention strategies.
Findings and Conclusions	

How much do you think employees would benefit from having organizations work to improve their work environments, work life balance, promotions, job satisfaction, and overall employee experience?

23 responses



The opinion of the participants identifies that workers would benefit from their organizations actively working to improve their work environments, work life balance, promotions, job satisfaction and other factors contributing to attrition.

PROTOTYPE IMPLEMENTATION

Data Preprocessing and Pipelining

```
X_train=project_data.drop(columns=["Attrition"])
y_train=project_data["Attrition"]

X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.3)
```

```
print('Train dataset shape:',X_train.shape)
print('Test dataset shape', y_train.shape)
```

```
Train dataset shape: (1029, 34)
Test dataset shape (1029,)
```

```
numeric_columns = X_train.select_dtypes(exclude='object').columns
print(numeric_columns)
print('*'*100)
categorical_columns = X_train.select_dtypes(include='object').columns
print(categorical_columns)
```

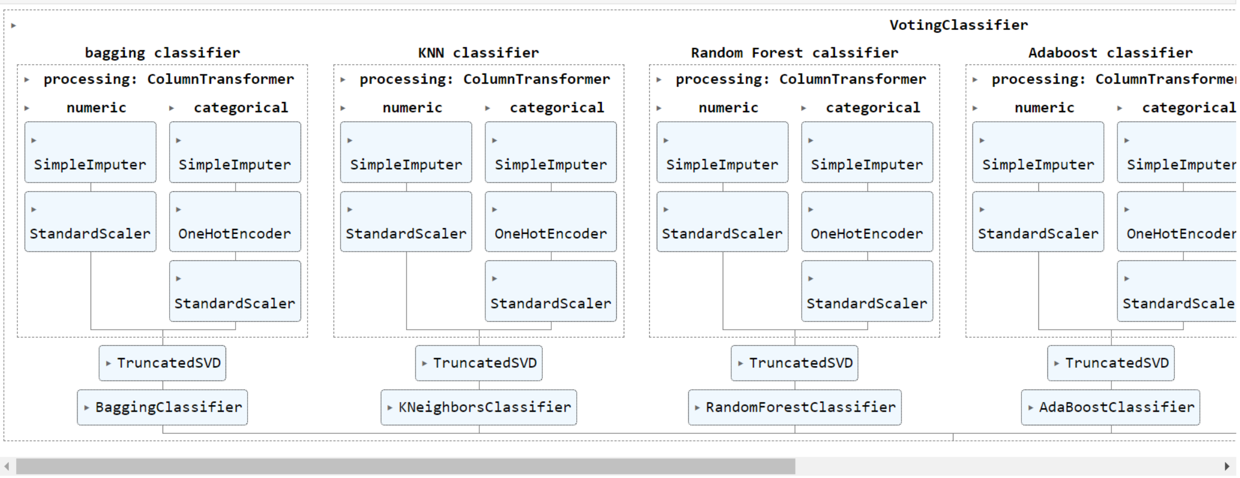
```
Index(['Age', 'DailyRate', 'DistanceFromHome', 'Education', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobSatisfaction', 'MonthlyIncome',
       'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
       'PerformanceRating', 'RelationshipSatisfaction', 'StandardHours',
       'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
       'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')
****************************************************************************************************
Index(['BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole',
       'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
```

```
numeric_features = Pipeline([
    ('handlingmissingvalues',SimpleImputer(strategy='median')),
    ('scaling',StandardScaler(with_mean=True))
])

print(numeric_features)
print('*'*100)

categorical_features = Pipeline([
    ('handlingmissingvalues',SimpleImputer(strategy='most_frequent')),
    ('encoding', OneHotEncoder()),
    ('scaling', StandardScaler(with_mean=False))
])

print(categorical_features)

processing = ColumnTransformer([
    ('numeric', numeric_features, numeric_columns),
    ('categorical', categorical_features, categorical_columns)
])

processing
```

Model Preparation

```
algorithms = [('bagging classifier', BaggingClassifier()),
              ('KNN classifier', KNeighborsClassifier()),
              ('Random Forest calssifier', RandomForestClassifier()),
              ('Adaboost classifier', AdaBoostClassifier()),
              ('Gradientboot classifier',GradientBoostingClassifier()),
              ('MLP', MLPClassifier())
             ]

trained_models = []
model_and_score = {}

for index, tup in enumerate(algorithms):
    model = prepare_model(tup[0],tup[1])
    model_and_score[tup[0]] = str(model.score(X_train,y_train)*100)+"%"
    trained_models.append((tup[0],model))

# Transform the training and testing data using the preprocessing steps
X_train_transformed = model.named_steps['processing'].transform(X_train)
X_test_transformed = model.named_steps['processing'].transform(X_test)

# Perform dimensionality reduction using TruncatedSVD
X_train_svd = model.named_steps['pca'].transform(X_train_transformed)
X_test_svd = model.named_steps['pca'].transform(X_test_transformed)
```

```
# Voting method

y_train = y_train.replace({'Yes': 1, 'No': 0})
y_test = np.where(y_test == 'Yes', 1, 0)

# Create a voting classifier with the trained models
voting_classifier = VotingClassifier(estimators=trained_models, voting='hard')

# Fit the voting classifier on the training data
voting_classifier.fit(X_train, y_train)
```




```
y_pred = voting_classifier.predict(X_test)
```

```
# Evaluate the voting classifier on the test data
voting_accuracy = voting_classifier.score(X_test, y_test)

print("Voting Classifier Accuracy:", voting_accuracy)
```

```
Voting Classifier Accuracy: 0.8435374149659864
```

```python
from sklearn.metrics import accuracy_score

# Collect predictions from all trained models
predictions = [model.predict(X_test) for _, model in trained_models]

# Convert string predictions to numeric values
numeric_predictions = [np.where(pred == 'Yes', 1, 0) for pred in predictions]

# Take the average of numeric predictions
average_predictions = np.mean(numeric_predictions, axis=0)

# Convert average predictions to binary labels (0 or 1)
average_predictions = np.where(average_predictions >= 0.5, 1, 0)

# Calculate accuracy of the simple average ensemble
average_accuracy = accuracy_score(y_test, average_predictions)

print("Simple Average Classifier Accuracy:", average_accuracy)

# Append the simple average ensemble to the trained_models list
trained_models.append(('Simple Average Classifier', average_predictions))

# Update the model_and_score dictionary with the simple average ensemble accuracy
model_and_score['Simple Average Classifier'] = f"{average_accuracy * 100:.2f}%"

with open('ML_simple_average_classifier.pkl', 'wb') as file:
    pickle.dump(average_predictions, file)
```

Model Evaluation

```python
print(model_and_score)
```

```
{'bagging classifier': '97.57045675413022%', 'KNN classifier': '87.07482993197279%', 'Random Forest calssifier': '100.0%', 'Adaboost classifier': '87.172
01166180757%', 'Gradientboot classifier': '90.67055393586006%', 'MLP': '85.51992225461613%', 'Voting Classifier': '84.35%', 'Simple Average Ensemble': '8
4.58%', 'Stacking Classifier': '84.58%'}
```

USE CASE SPECIFICATION

Use Case	Input employee data
UC Identifier	UC1
Description	Submit precise and organized employee details into the predictor so that it can be utilized for the prediction of employee attrition.
Primary Actor	HR Department Staff
Supporting Actors (if present)	AttritionPro Operator
Stakeholders and Interests	HR Department Staff, AttritionPro Operator
Pre-Conditions	• The employee data must be in the appropriate format, with all fields and information required, and the system must be active and usable. • The system needs to be able to handle and store employee data.

Post-Conditions	• The input employee data must be correct and consistent with the necessary format. • The data must have been effectively stored in the database. • The user must receive confirmation from the system that the data has been successfully stored. • The newly added employee data must be reflected in the system's database, and the input employee data must be retrieved and accessible for use in the employee retention prediction process in the future.

Use Case	Generate prediction results
UC Identifier	UC2
Description	To determine which employees are most likely to leave the company, analyze employee data that has been input.
Primary Actor	HR Department Staff
Supporting Actors (if present)	Data Scientist
Pre-Conditions	• The system's database has to have accurate and current employee data input into it. • The prediction procedure needs to be run through the system's configuration and setup. • The required models and algorithms have to be set up and operating properly. • To create the forecasts, the user needs to have the required access and permissions. • With the data at its disposal, the system needs to be able to generate forecasts that are useful and accurate to the user.
Post-Conditions	• It is necessary to produce meaningful and accurate predictions on employee attrition.

	• The forecasts need to be kept for accessibility and usage in the future. • The user must receive feedback from the system confirming that the forecasts were correctly generated. • For end users to use the predictions, they need to be able to access and retrieve them. • The recently created predictions need to be reflected in the database of the system.

APPENDIX C: IMPLEMENTATION

Data Preprocessing and Pipelining

```
X_train=project_data.drop(columns=["Attrition"])
y_train=project_data["Attrition"]

X_train, X_test, y_train, y_test = train_test_split(X_train, y_train, test_size=0.3)
```

```
print('Train dataset shape:',X_train.shape)
print('Test dataset shape', X_test.shape)

print('Train dataset rows: ', len(X_train))
print('Test dataset rows: ', len(X_test))
```

```
Train dataset shape: (1029, 34)
Test dataset shape (441, 34)
Train dataset rows:  1029
Test dataset rows:  441
```

```
numeric_columns = X_train.select_dtypes(exclude='object').columns
print(numeric_columns)
print('*'*100)
categorical_columns = X_train.select_dtypes(include='object').columns
print(categorical_columns)
```

```
Index(['Age', 'DailyRate', 'DistanceFromHome', 'Education', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobSatisfaction', 'MonthlyIncome',
       'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
       'PerformanceRating', 'RelationshipSatisfaction', 'StandardHours',
       'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
       'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')
****************************************************************************************************
Index(['BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole',
       'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
```

```
numeric_features = Pipeline([
    ('handlingmissingvalues',SimpleImputer(strategy='median')),
    ('scaling',StandardScaler(with_mean=True))
])

print(numeric_features)
print('*'*100)

categorical_features = Pipeline([
    ('handlingmissingvalues',SimpleImputer(strategy='most_frequent')),
    ('encoding', OneHotEncoder()),
    ('scaling', StandardScaler(with_mean=False))
])

print(categorical_features)

processing = ColumnTransformer([
    ('numeric', numeric_features, numeric_columns),
    ('categorical', categorical_features, categorical_columns)
])

processing
```

```python
def create_fnn(hp):
    model = Sequential()
    model.add(Dense(units=hp.Int('units_1', min_value=32, max_value=512, step=32),
                    activation='relu', input_dim=input_dim))
    model.add(Dropout(rate=hp.Float('dropout_1', min_value=0.0, max_value=0.5, step=0.1)))

    model.add(Dense(units=hp.Int('units_2', min_value=32, max_value=256, step=32),
                    activation='relu'))
    model.add(Dropout(rate=hp.Float('dropout_2', min_value=0.0, max_value=0.5, step=0.1)))

    model.add(Dense(1, activation='sigmoid'))

    model.compile(optimizer=Adam(learning_rate=hp.Choice('learning_rate', values=[1e-2, 1e-3, 1e-4])),
                  loss='binary_crossentropy',
                  metrics=['accuracy'])
    return model
```

```python
def create_wide_and_deep_model(hp):
    input_dim = X_train.shape[1]

    wide_inputs = Input(shape=(input_dim,))
    deep_inputs = Input(shape=(input_dim,))

    wide_layer = Dense(units=hp.Int('wide_units', min_value=32, max_value=256, step=32),
                       activation='relu')(wide_inputs)

    deep_layer = Dense(units=hp.Int('deep_units_1', min_value=32, max_value=256, step=32),
                      activation='relu')(deep_inputs)
    deep_layer = Dense(units=hp.Int('deep_units_2', min_value=16, max_value=128, step=16),
                      activation='relu')(deep_layer)

    merged_layer = concatenate([wide_layer, deep_layer])

    output = Dense(1, activation='sigmoid')(merged_layer)

    model = Model(inputs=[wide_inputs, deep_inputs], outputs=output)

    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

    return model
```

```python
def create_cnn_model(hp):
    input_shape = (X_train_reshaped.shape[1], X_train_reshaped.shape[2])

    model = Sequential()

    model.add(Conv1D(filters=hp.Int('filters', min_value=16, max_value=64, step=16),
                     kernel_size=hp.Int('kernel_size', min_value=2, max_value=5, step=1),
                     padding='same',
                     activation='relu',
                     input_shape=input_shape))

    model.add(MaxPooling1D(pool_size=2))
    model.add(Flatten())

    model.add(Dense(128, activation='relu'))
    model.add(Dropout(0.2))

    model.add(Dense(64, activation='relu'))
    model.add(Dropout(0.2))

    model.add(Dense(1, activation='sigmoid'))

    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

    return model
```

```python
input_dim = X_train.shape[1]

fnn_tuner = kt.RandomSearch(
    create_fnn,
    objective='val_accuracy',
    max_trials=10,
    directory='my_dir',
    project_name='fnn_hyperparameter_tuning')

fnn_tuner.search(X_train, y_train, epochs=100, validation_data=(X_test, y_test))

best_fnn_model = fnn_tuner.get_best_models(num_models=1)[0]
best_fnn_hyperparameters = fnn_tuner.get_best_hyperparameters(num_trials=1)[0]

best_fnn_model.summary()

best_fnn_model.fit(X_train, y_train, epochs=100, batch_size=32, validation_data=(X_test, y_test))

best_fnn_accuracy = best_fnn_model.evaluate(X_test, y_test, verbose=0)[1]
print("Best FNN Model Accuracy:", best_fnn_accuracy)
print("Best Hyperparameters:", best_fnn_hyperparameters)
```

```
wd_tuner = RandomSearch(
    create_wide_and_deep_model,
    objective='val_accuracy',
    max_trials=10,
    directory='my_dir',
    project_name='wide_and_deep_hyperparameter_tuning'
)

wd_tuner.search_space_summary()
wd_tuner.search([X_train, X_train], y_train, epochs=100, batch_size=32, validation_data=([X_test, X_test], y_test))

best_wd_model = wd_tuner.get_best_models(num_models=1)[0]
best_wd_hyperparameters = wd_tuner.get_best_hyperparameters(num_trials=1)[0]

best_wd_model.fit([X_train, X_train], y_train, epochs=100, batch_size=32, validation_data=([X_test, X_test], y_test))

best_wide_and_deep_accuracy = best_wd_model.evaluate([X_test, X_test], y_test, verbose=0)[1]
print("Best Wide & Deep Model Accuracy:", best_wide_and_deep_accuracy)
```

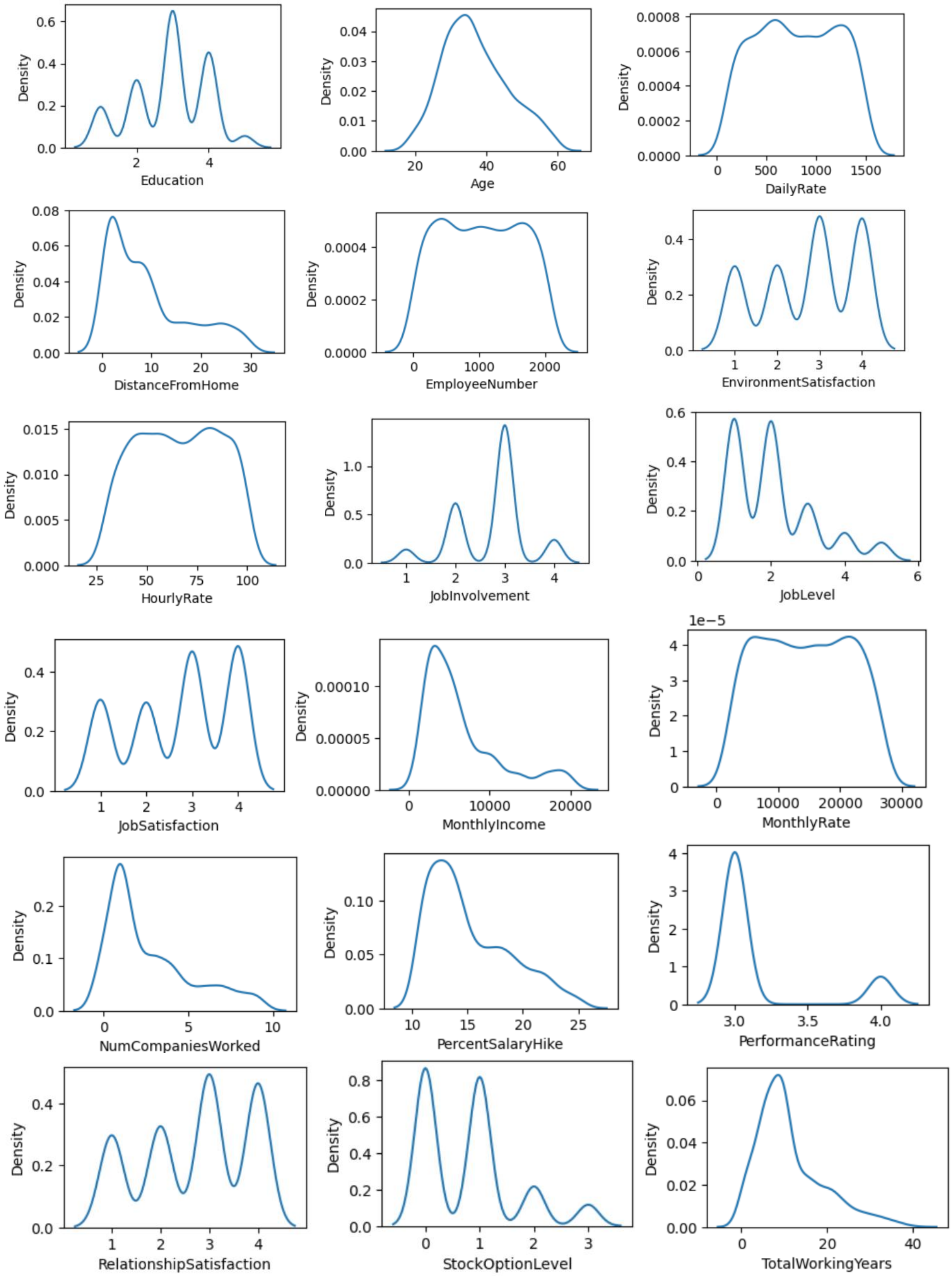
```
cnn_tuner = kt.Hyperband(create_cnn_model,
                     objective='val_accuracy',
                     max_epochs=10,
                     factor=3,
                     directory='my_dir',
                     project_name='intro_to_kt')

cnn_tuner.search(X_train_reshaped, y_train, epochs=100, validation_data=(X_test_reshaped, y_test))

best_cnn_hyperparameters=cnn_tuner.get_best_hyperparameters(num_trials=1)[0]

best_cnn_model = cnn_tuner.hypermodel.build(best_cnn_hyperparameters)
cnn_history = best_cnn_model.fit(X_train_reshaped, y_train, epochs=100, validation_data=(X_test_reshaped, y_test))

best_cnn_accuracy = cnn_history.history['val_accuracy'][-1]
print("Best CNN Model Accuracy:", best_cnn_accuracy)
```

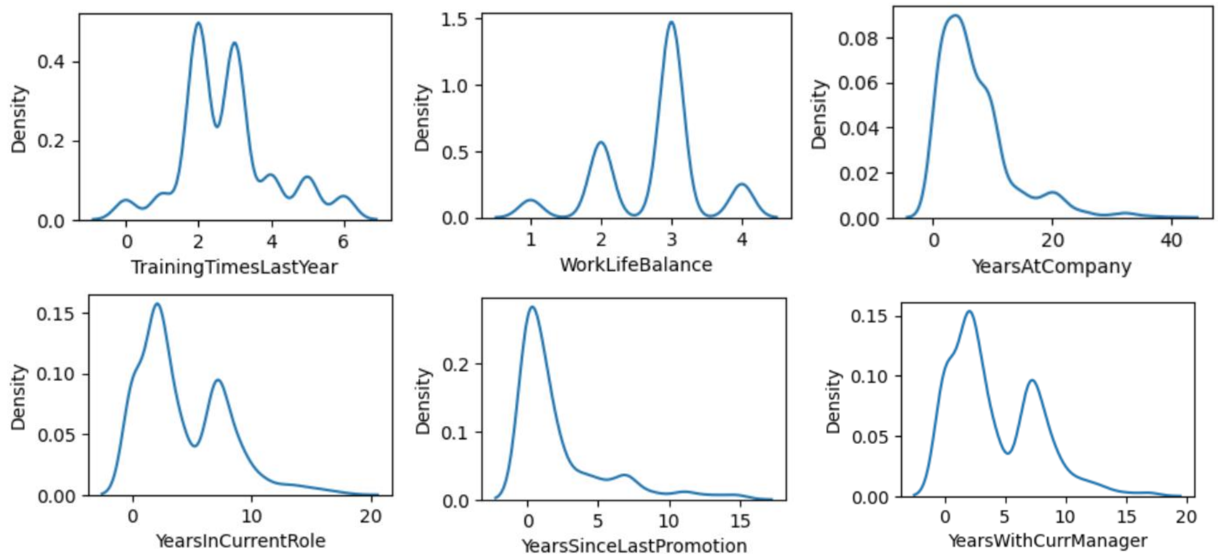
```
ensemble_methods = ['stacking', 'voting', 'simple_average']

for i, method in enumerate(ensemble_methods, 1):

    print(f"\n-------- Ensemble Method: {method} ----------")
    ensemble_predictions = ensemble_predict(models, X_test, X_train, method)
    ensemble_accuracy = accuracy_score(y_test, ensemble_predictions)
    if method == 'stacking':
        stacking_accuracy = ensemble_accuracy
    elif method == 'voting':
        voting_accuracy = ensemble_accuracy
    elif method == 'simple_average':
        simple_average_accuracy = ensemble_accuracy
```

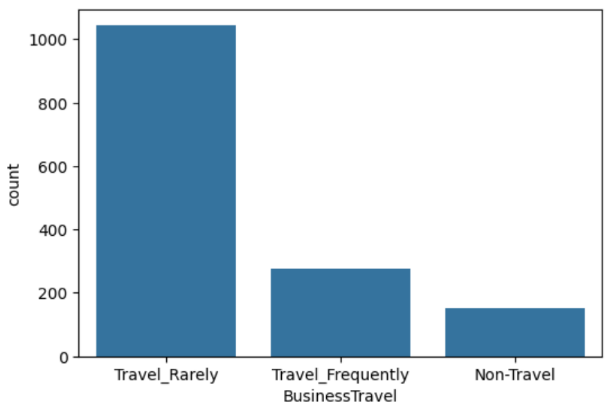
```python
def ensemble_predict(models, X_test, X_train, method):
    test_predictions = []
    train_predictions = []

    for model in models:
        if isinstance(model, Sequential):
            X_test_input = X_test
            X_train_input = X_train
        elif isinstance(model, Model):
            X_test_input = (X_test, X_test)
            X_train_input = (X_train, X_train)
        else:
            raise ValueError("Invalid model type provided.")

        test_predictions.append(model.predict(X_test_input))
        train_predictions.append(model.predict(X_train_input))

    if method == 'simple_average':
        test_predictions = np.array(test_predictions)
        return np.round(np.mean(test_predictions, axis=0)).astype(int)
    elif method == 'voting':
        test_predictions = np.array(test_predictions)
        class_votes = np.round(np.mean(test_predictions, axis=0)).astype(int)
        final_prediction = np.where(np.sum(class_votes, axis=1) > test_predictions.shape[0] / 2, 1, 0)
        return final_prediction
    elif method == 'stacking':
        meta_X_train = np.concatenate(train_predictions, axis=1)
        meta_X_test = np.concatenate(test_predictions, axis=1)

        best_hps = tune_metamodel(meta_X_train, y_train)
        meta_model = create_metamodel_fnn(best_hps)
        meta_model.fit(meta_X_train, y_train, epochs=100, batch_size=32, validation_split=0.2)
        model_name = 'Stacking'
        prepare_evaluation(meta_model, model_name, X_test_svd, y_test_encoded)
        df_ensemble_metrics = pd.DataFrame(ensemble_metrics).T
        print('df_ensemble_metrics')
        print(df_ensemble_metrics)
        print()
        meta_model.save("stacking_ensemble_model.h5")
        meta_predictions = meta_model.predict(meta_X_test)
        return np.round(meta_predictions).astype(int)
    else:
        raise ValueError("Invalid ensemble method provided.")
```

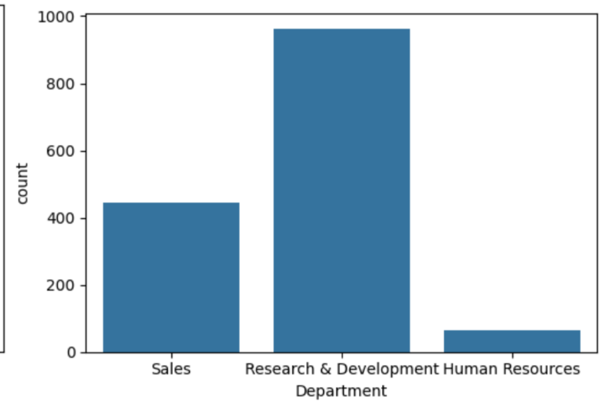
Exploratory Data Analysis

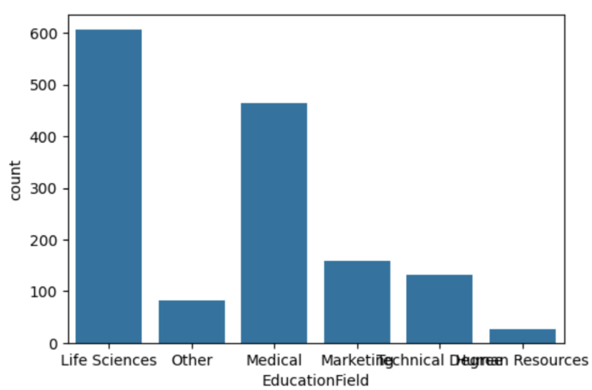
Univariate Data Analysis

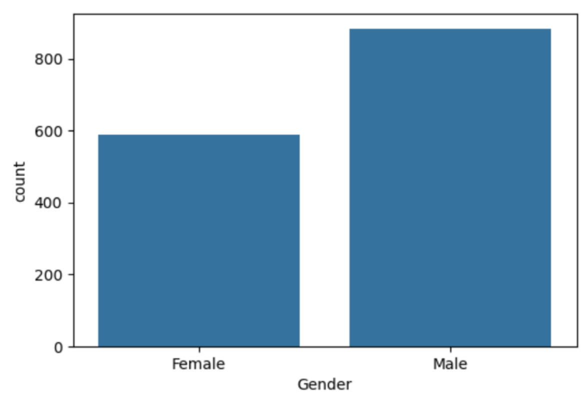

Density
Education
Age
DailyRate
DistanceFromHome
EmployeeNumber
EnvironmentSatisfaction
HourlyRate
JobInvolvement
JobLevel
JobSatisfaction
MonthlyIncome
1e−5
MonthlyRate
NumCompaniesWorked
PercentSalaryHike
PerformanceRating
RelationshipSatisfaction
StockOptionLevel
TotalWorkingYears

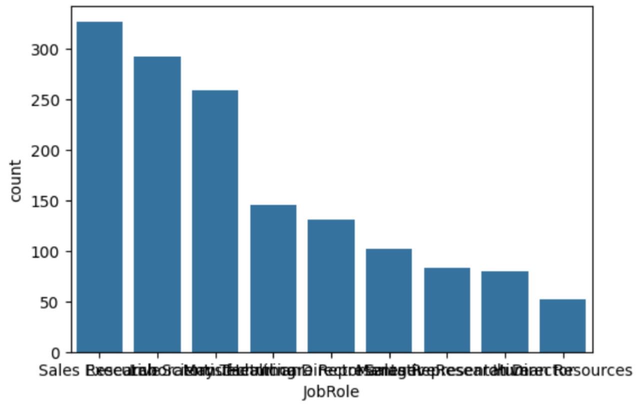

Density
TrainingTimesLastYear
WorkLifeBalance
YearsAtCompany
YearsInCurrentRole
YearsSinceLastPromotion
YearsWithCurrManager

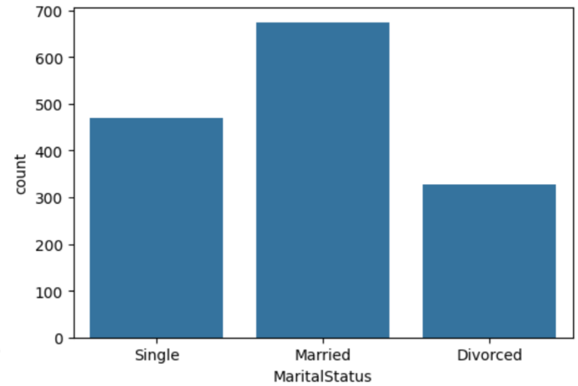

count
Travel_Rarely
Travel_Frequently
Non-Travel
BusinessTravel

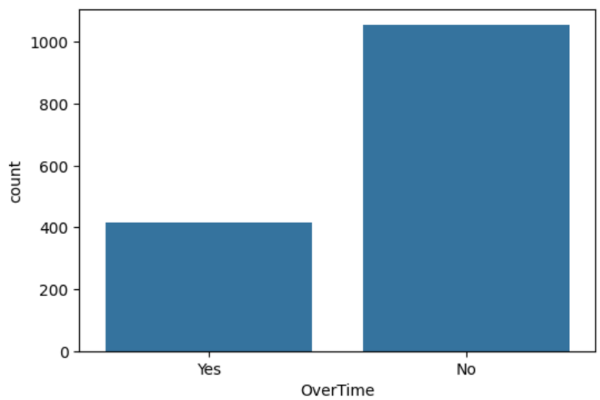

count
Sales
Research & Development
Human Resources
Department


count
Life Sciences
Other
Medical
Marketing
Technical Degree
Human Resources
EducationField

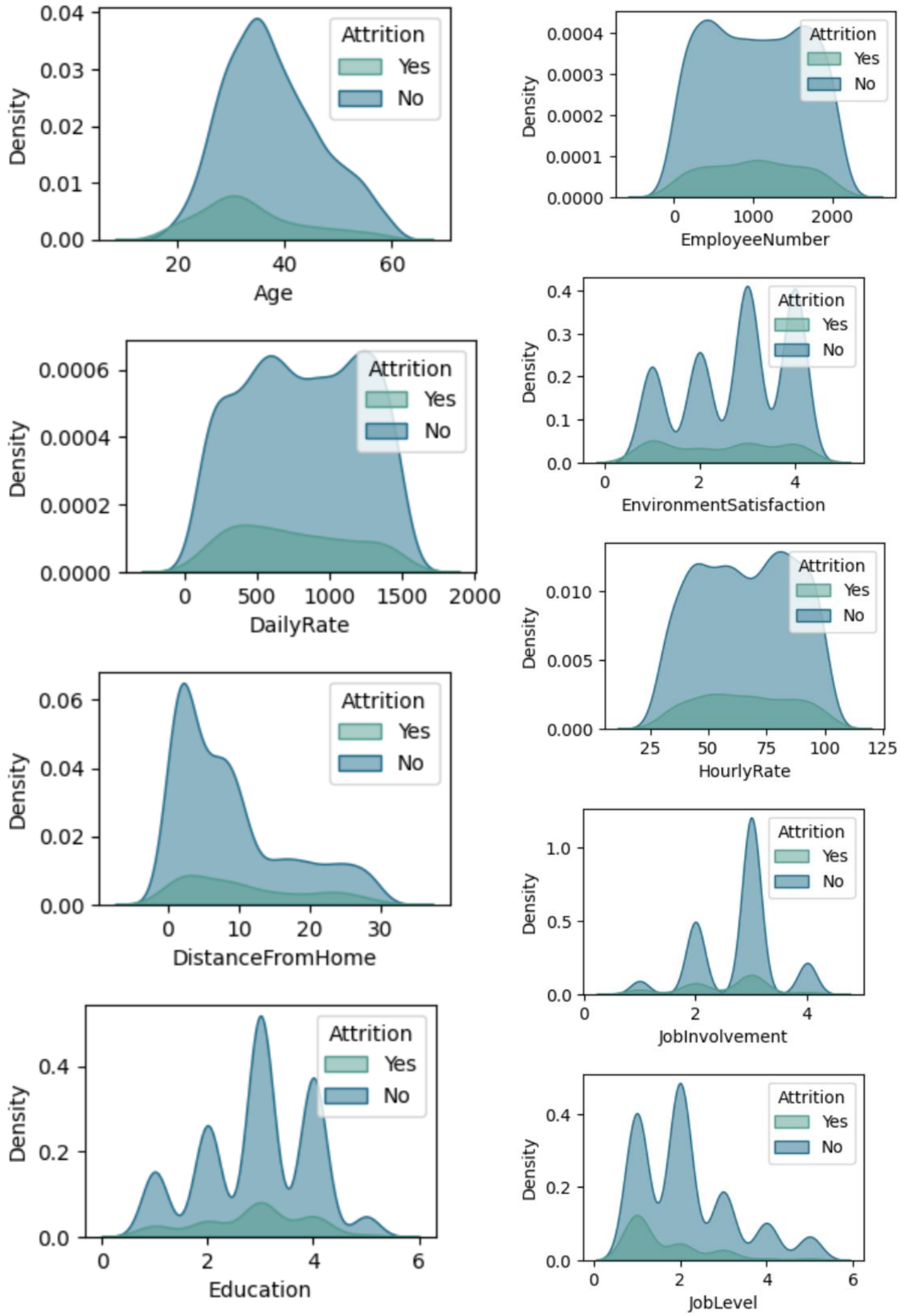

count
Female
Male
Gender

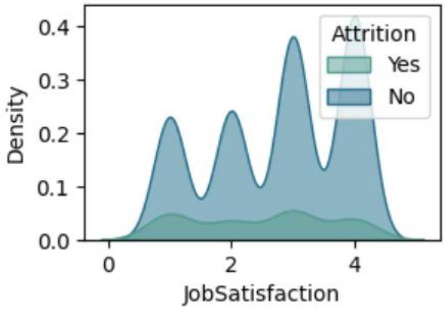

count
0
50
100
150
200
250
300
JobRole

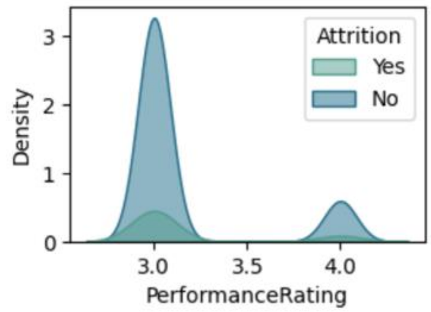

count
0
100
200
300
400
500
600
700
Single
Married
Divorced
MaritalStatus

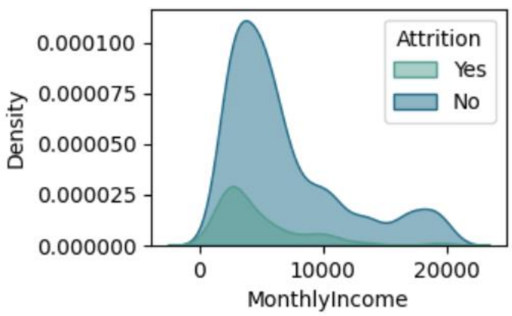

count
0
200
400
600
800
1000
Yes
No
OverTime

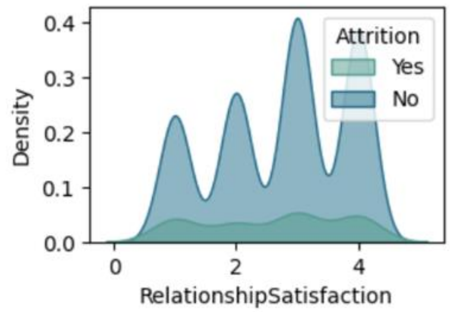
Bivariate Data Analysis

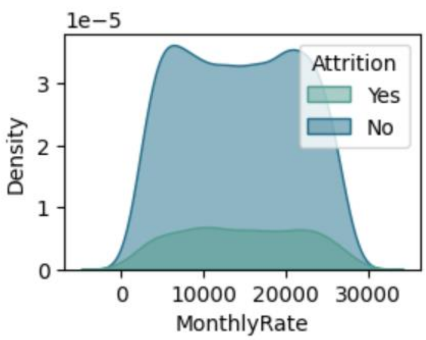


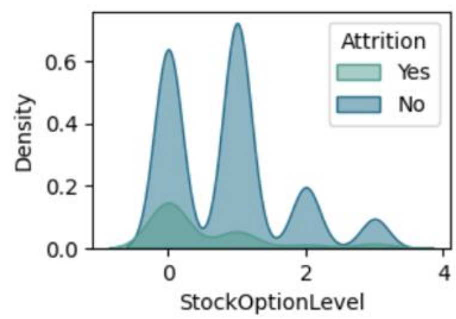

Attrition
Yes
No
Density
0.0 0.1 0.2 0.3 0.4
0 2 4
JobSatisfaction

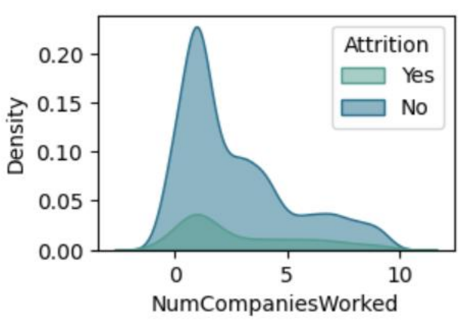

Attrition
Yes
No
Density
0 1 2 3
3.0 3.5 4.0
PerformanceRating

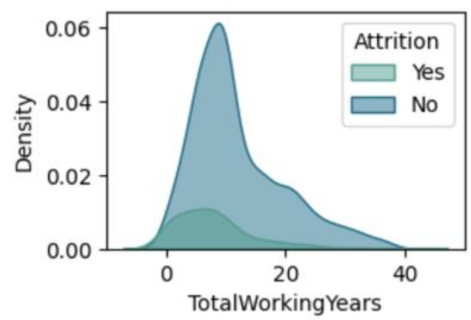

Attrition
Yes
No
Density
0.000000 0.000025 0.000050 0.000075 0.000100
0 10000 20000
MonthlyIncome


Attrition
Yes
No
Density
0.0 0.1 0.2 0.3 0.4
0 2 4
RelationshipSatisfaction

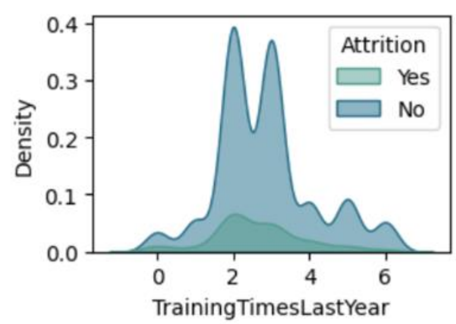

1e−5
Attrition
Yes
No
Density
0 1 2 3
0 10000 20000 30000
MonthlyRate

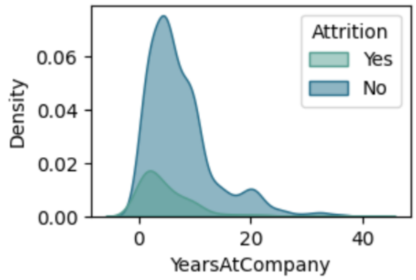

Attrition
Yes
No
Density
0.0 0.2 0.4 0.6
0 2 4
StockOptionLevel

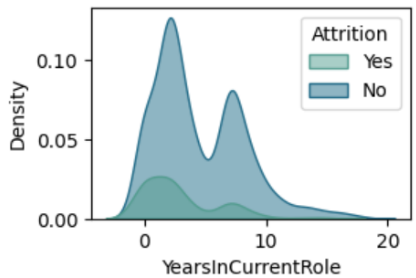

Attrition
Yes
No
Density
0.00 0.05 0.10 0.15 0.20
0 5 10
NumCompaniesWorked

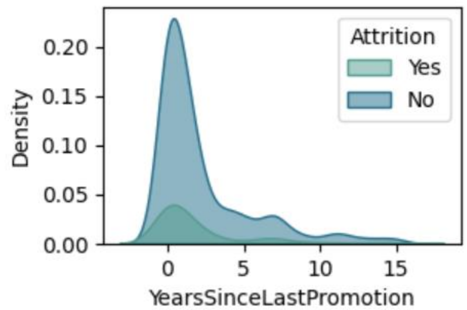

Attrition
Yes
No
Density
0.00 0.02 0.04 0.06
0 20 40
TotalWorkingYears

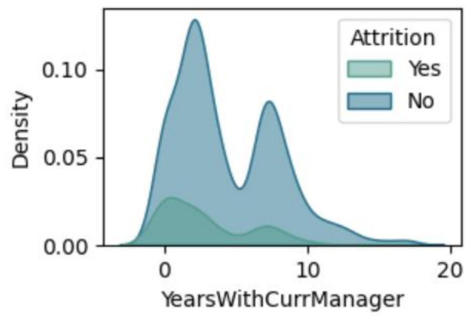

Attrition
Yes
No
Density
0.000 0.025 0.050 0.075 0.100
10 15 20 25
PercentSalaryHike


Attrition
Yes
No
Density
0.0 0.1 0.2 0.3 0.4
0 2 4 6
TrainingTimesLastYear

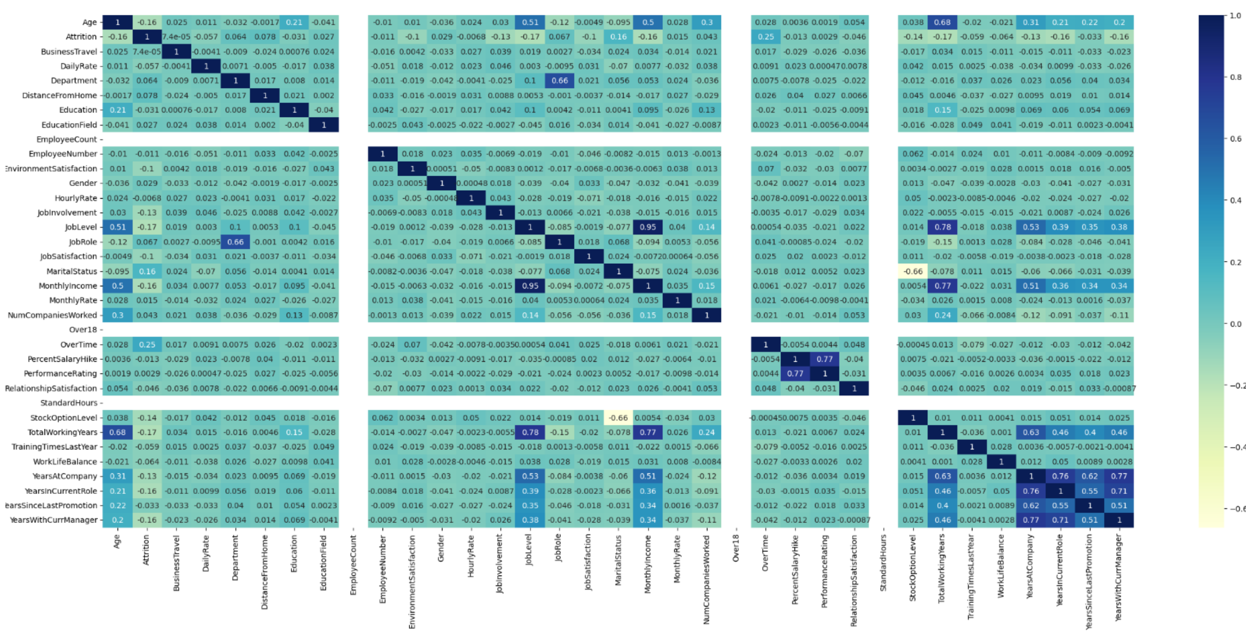




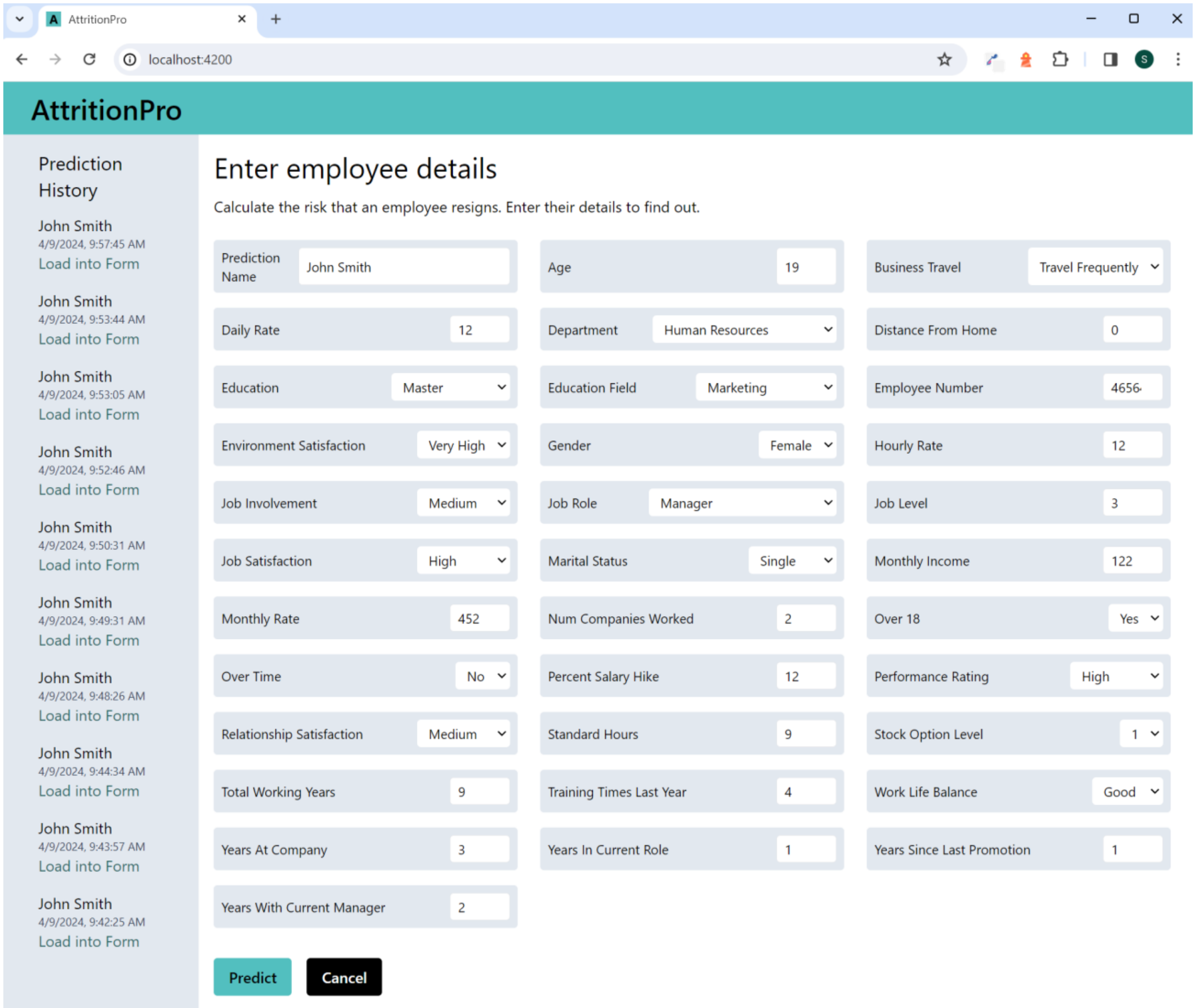


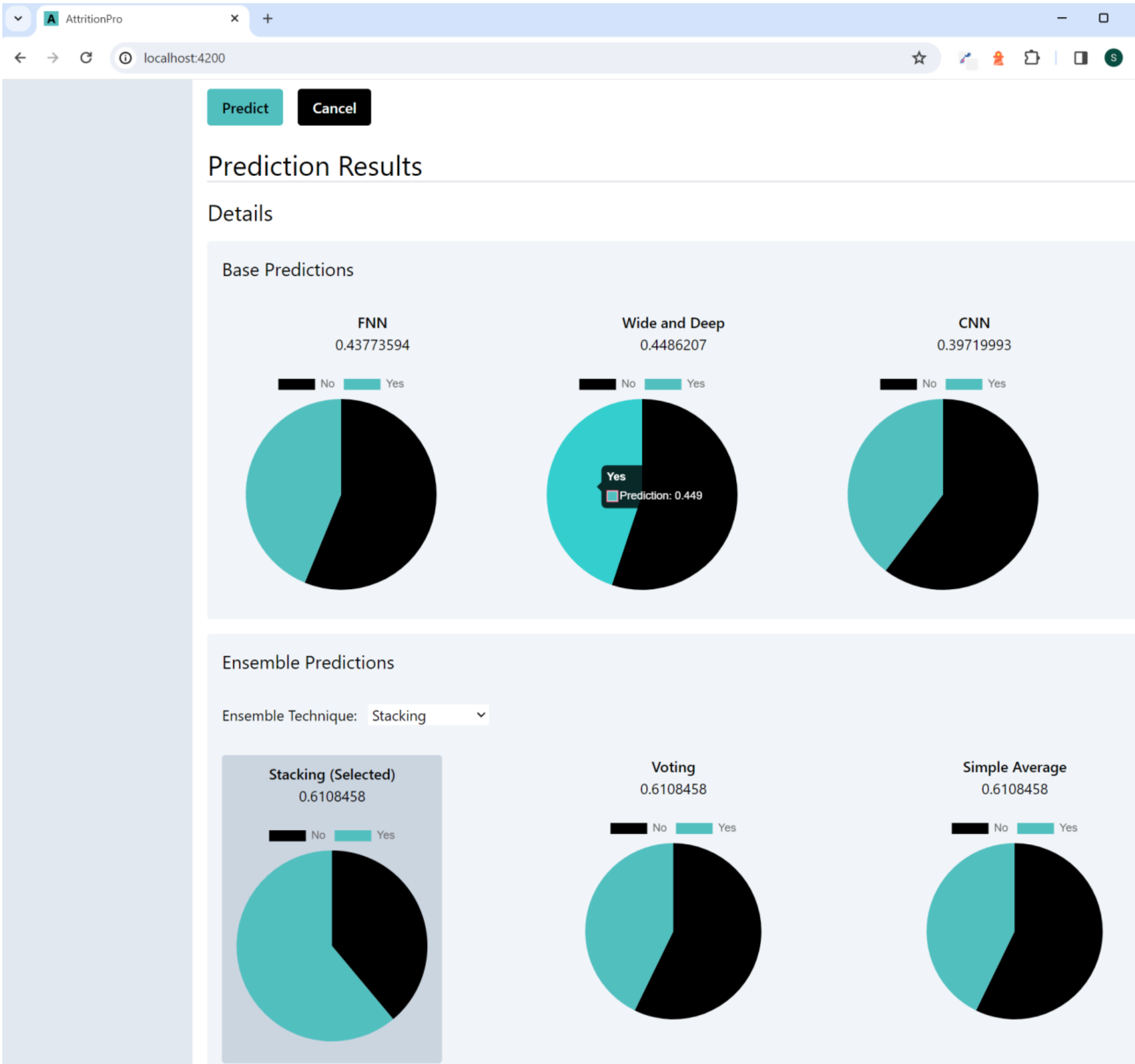


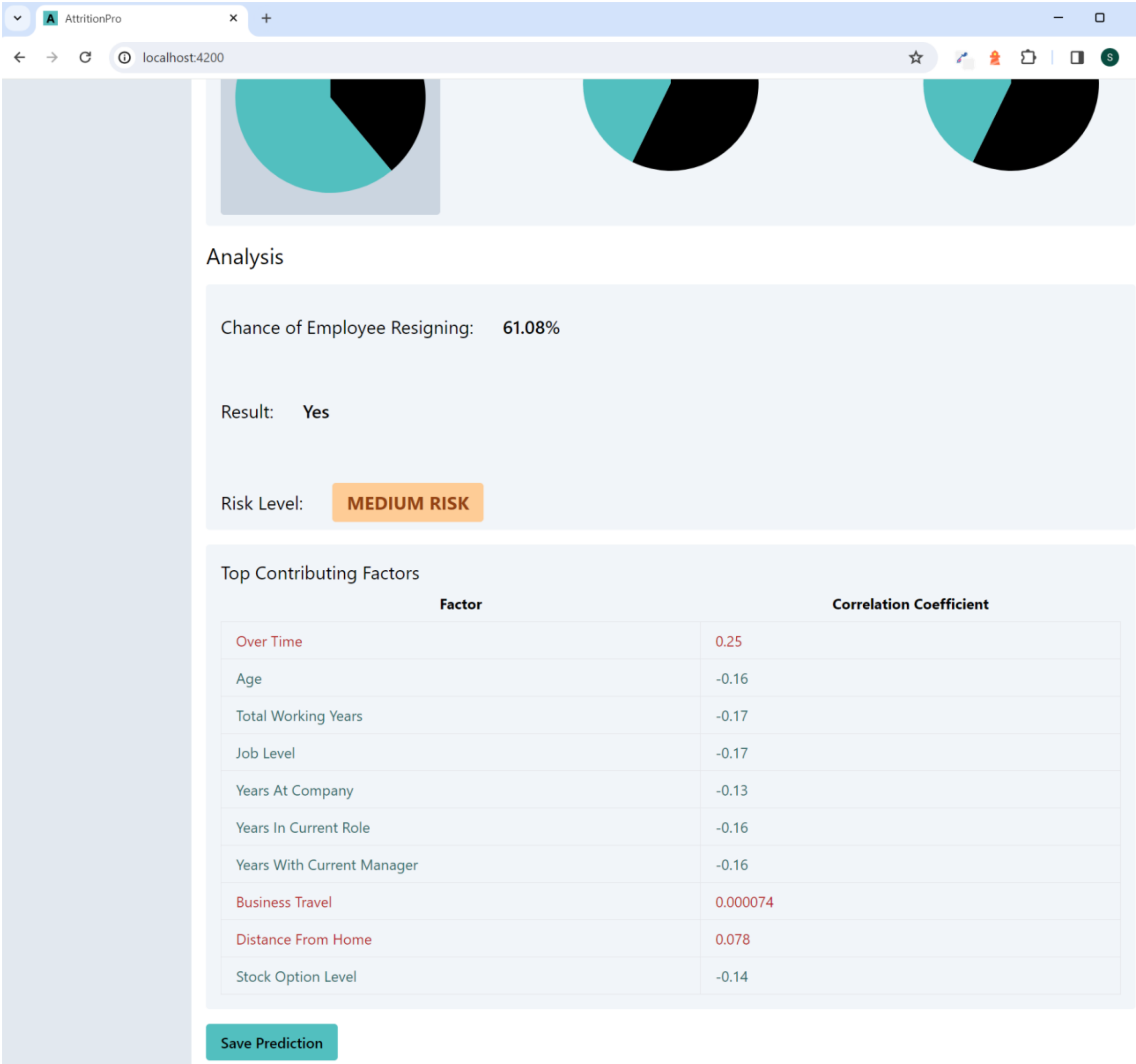
Correlation Analysis



User Interface




AttritionPro
localhost:4200
Predict
Cancel
Prediction Results
Details
Base Predictions
FNN
0.43773594
Wide and Deep
0.4486207
CNN
0.39719993
No
Yes
Yes
Prediction: 0.449
Ensemble Predictions
Ensemble Technique:
Stacking
Stacking (Selected)
0.6108458
Voting
0.6108458
Simple Average
0.6108458


AttritionPro
localhost:4200
Analysis
Chance of Employee Resigning: 61.08%
Result: Yes
Risk Level: MEDIUM RISK
Top Contributing Factors
Factor
Correlation Coefficient
Over Time
0.25
Age
-0.16
Total Working Years
-0.17
Job Level
-0.17
Years At Company
-0.13
Years In Current Role
-0.16
Years With Current Manager
-0.16
Business Travel
0.000074
Distance From Home
0.078
Stock Option Level
-0.14
Save Prediction

APPENDIX D: TESTING

Model Evaluation

The accuracies of the final ensemble models created are as follows.

Model Evaluation

```
print(model_and_score)
{'DL Ensemble (simple_average)': '82.54%', 'DL Ensemble (voting)': '82.31%', 'DL Ensemble (stacking)': '80.73%'}
```

A classification report was created to evaluate the model using input data. For all of the 3 ensemble models, a prediction was generated and converted into binary classes to represent Yes or No and the risk of attrition was calculated for each.

```
# Prepare Classification Report

# Transform input data using preprocessing steps
input_data = [input_data_df1, input_data_df2, input_data_df3, input_data_df4]

input_data_transformed = [model.named_steps['processing'].transform(data) for data in input_data]
input_data_svd = [model.named_steps['pca'].transform(data_transformed) for data_transformed in input_data_transformed]

# Define a function to convert probabilities to Yes/No labels
def convert_to_yes_no(predictions):
    return ["Yes" if pred > 0.5 else "No" for pred in predictions]

# Make predictions using the deep learning ensemble models
ensemble_predictions = {
    'simple_avg': [np.mean([model.predict(svd) for model in dl_ensemble_models_simple_avg], axis=0) for svd in input_data_svd],
    'voting': [np.mean([model.predict(svd) for model in dl_ensemble_models_voting], axis=0) for svd in input_data_svd],
    'stacking': [np.mean([model.predict(svd) for model in dl_ensemble_models_stacking], axis=0) for svd in input_data_svd]
}

# Convert ensemble predictions to Yes/No labels
ensemble_labels = {
    'simple_avg': [convert_to_yes_no(predictions) for predictions in ensemble_predictions['simple_avg']],
    'voting': [convert_to_yes_no(predictions) for predictions in ensemble_predictions['voting']],
    'stacking': [convert_to_yes_no(predictions) for predictions in ensemble_predictions['stacking']]
}

# Display the predictions
for i, data in enumerate(input_data):
    print(f"\nPredictions for input_data{i+1}:")
    print("Simple Average Ensemble:", ensemble_labels['simple_avg'][i], " Chance of Leaving: ", ensemble_predictions['simple_avg'][i])
    print("Voting Ensemble:", ensemble_labels['voting'][i], " Chance of Leaving: ", ensemble_predictions['voting'][i])
    print("Stacking Ensemble:", ensemble_labels['stacking'][i], " Chance of Leaving: ", ensemble_predictions['stacking'][i])
```

Classification Report:

```
Predictions for input_data1:
Simple Average Ensemble: ['No']  Chance of Leaving:  [[0.17318392]]
Voting Ensemble: ['No']  Chance of Leaving:  [[0.15544562]]
Stacking Ensemble: ['No']  Chance of Leaving:  [[0.18439801]]

Predictions for input_data2:
Simple Average Ensemble: ['No']  Chance of Leaving:  [[0.10108397]]
Voting Ensemble: ['No']  Chance of Leaving:  [[0.09914609]]
Stacking Ensemble: ['No']  Chance of Leaving:  [[0.10909622]]

Predictions for input_data3:
Simple Average Ensemble: ['No']  Chance of Leaving:  [[0.15436111]]
Voting Ensemble: ['No']  Chance of Leaving:  [[0.14808263]]
Stacking Ensemble: ['No']  Chance of Leaving:  [[0.17785585]]

Predictions for input_data4:
Simple Average Ensemble: ['No']  Chance of Leaving:  [[0.32005778]]
Voting Ensemble: ['No']  Chance of Leaving:  [[0.3027821]]
Stacking Ensemble: ['No']  Chance of Leaving:  [[0.3438516]]
```

The accuracies of the Base Models

```
Accuracy for FNN: 84.35%
Accuracy for Wide and Deep: 85.26%
Accuracy for CNN: 84.81%
```

```
Accuracy for FNN: 82.99%
Accuracy for Wide and Deep: 83.67%
Accuracy for CNN: 83.45%
```

The accuracies of the 3 ensemble techniques:

```
Accuracy for Stacking: 85.26%
Accuracy for Voting: 83.90%
Accuracy for Simple Average: 83.90%
```

```
Accuracy for Stacking: 84.13%
Accuracy for Voting: 82.31%
Accuracy for Simple Average: 83.67%
```

Other metrics for the base models:

	F1 Score	Precision	Recall	Support	Specificity
FNN	0.279570	0.650000	0.178082	73.0	0.980978
Wide and Deep	0.306122	0.600000	0.205479	73.0	0.972826
CNN	0.214286	0.818182	0.123288	73.0	0.994565

	Misclassification Rate
FNN	0.140590
Wide and Deep	0.145125
CNN	0.145125

	Accuracy	F1 Score	Precision	Recall	Support	Specificity
FNN	0.829932	0.271845	0.560000	0.179487	78.0	0.969697
Wide and Deep	0.836735	0.280000	0.636364	0.179487	78.0	0.977961
CNN	0.834467	0.215054	0.666667	0.128205	78.0	0.986226

	Misclassification Rate
FNN	0.170068
Wide and Deep	0.163265
CNN	0.165533

Other metrics for the stacking model:

	F1 Score	Precision	Recall	Specificity	Support
Stacking	0.0	0.0	0.0	1.0	73.0

	F1 Score	Misclassification Rate
Stacking	0.0	0.151927

	Accuracy	F1 Score	Misclassification Rate	Precision	Recall
Stacking	0.648526	0.104046	0.351474	0.09	0.123288

	Specificity	Support
Stacking	0.752717	73.0

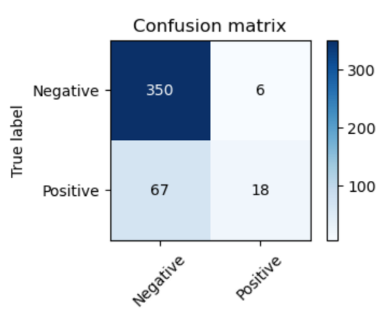
Model Evaluation

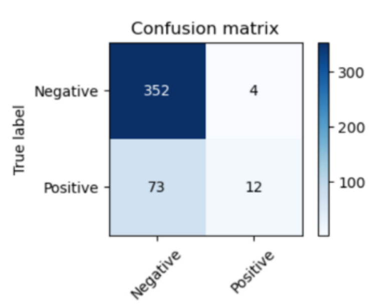
```
print(model_and_score)
{'DL Ensemble (simple_average)': '82.54%', 'DL Ensemble (voting)': '82.31%', 'DL Ensemble (stacking)': '80.73%'}
```

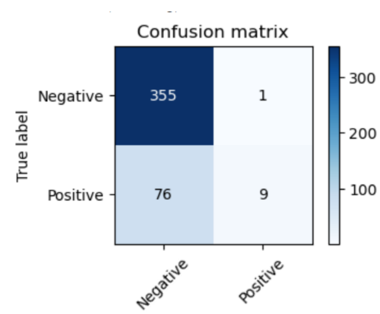
Classification Report:

```
Predictions for input_data1:
Simple Average Ensemble: ['No']  Chance of Leaving:  [[0.17318392]]
Voting Ensemble: ['No']  Chance of Leaving:  [[0.15544562]]
Stacking Ensemble: ['No']  Chance of Leaving:  [[0.18439801]]

Predictions for input_data2:
Simple Average Ensemble: ['No']  Chance of Leaving:  [[0.10108397]]
Voting Ensemble: ['No']  Chance of Leaving:  [[0.09914609]]
Stacking Ensemble: ['No']  Chance of Leaving:  [[0.10909622]]

Predictions for input_data3:
Simple Average Ensemble: ['No']  Chance of Leaving:  [[0.15436111]]
Voting Ensemble: ['No']  Chance of Leaving:  [[0.14808263]]
Stacking Ensemble: ['No']  Chance of Leaving:  [[0.17785585]]

Predictions for input_data4:
Simple Average Ensemble: ['No']  Chance of Leaving:  [[0.32005778]]
Voting Ensemble: ['No']  Chance of Leaving:  [[0.3027821]]
Stacking Ensemble: ['No']  Chance of Leaving:  [[0.3438516]]
```

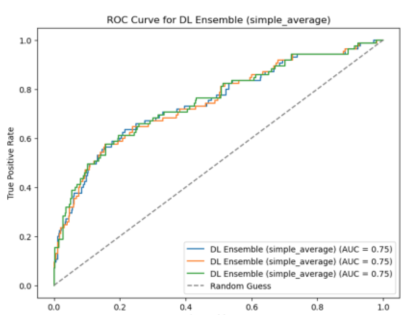
Confusion Matrix for the Simple Average Method, Voting Method and Stacking Method:

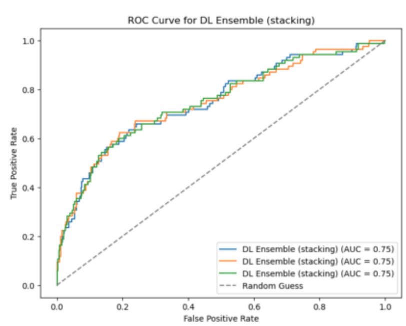


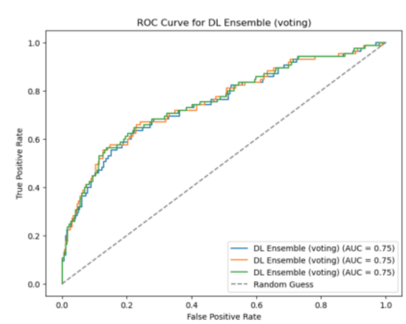




ROC Curves for the Simple Average Method, Voting Method and Stacking Method:

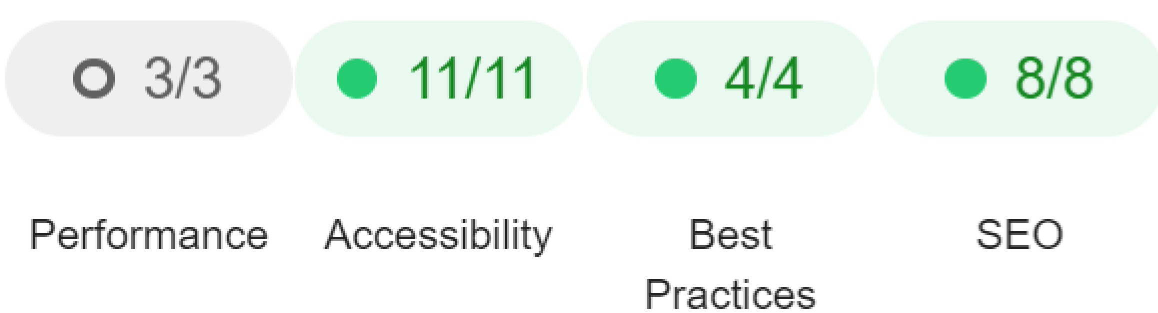






Performance Testing

Google Lighthouse



3/3

Performance

DIAGNOSTICS

Avoids an excessive DOM size — 400 elements

More information about the performance of your application. These numbers don't directly affect the Performance score.

PASSED AUDITS (3) Show

11/11

Accessibility

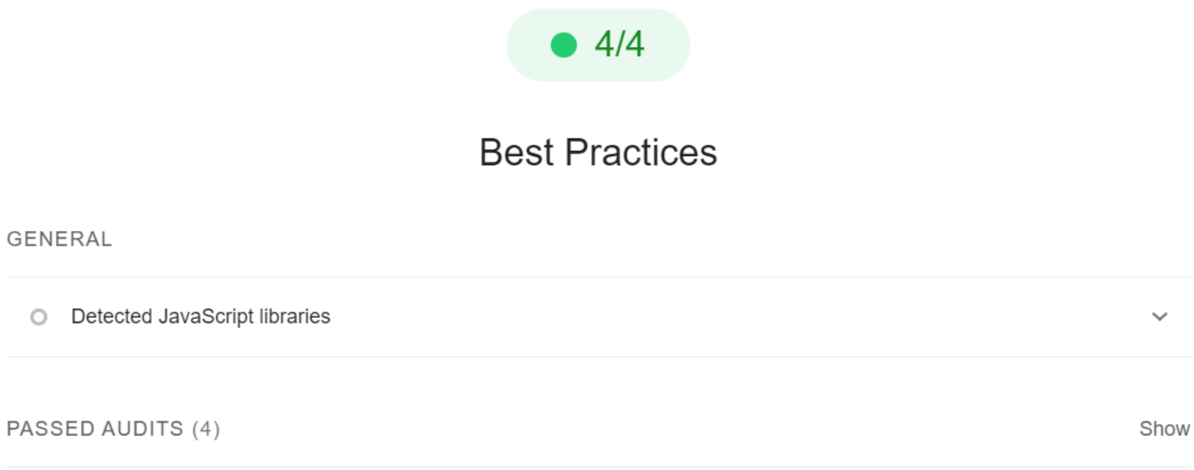
These checks highlight opportunities to improve the accessibility of your web app. Automatic detection can only detect a subset of issues and does not guarantee the accessibility of your web app, so manual testing is also encouraged.

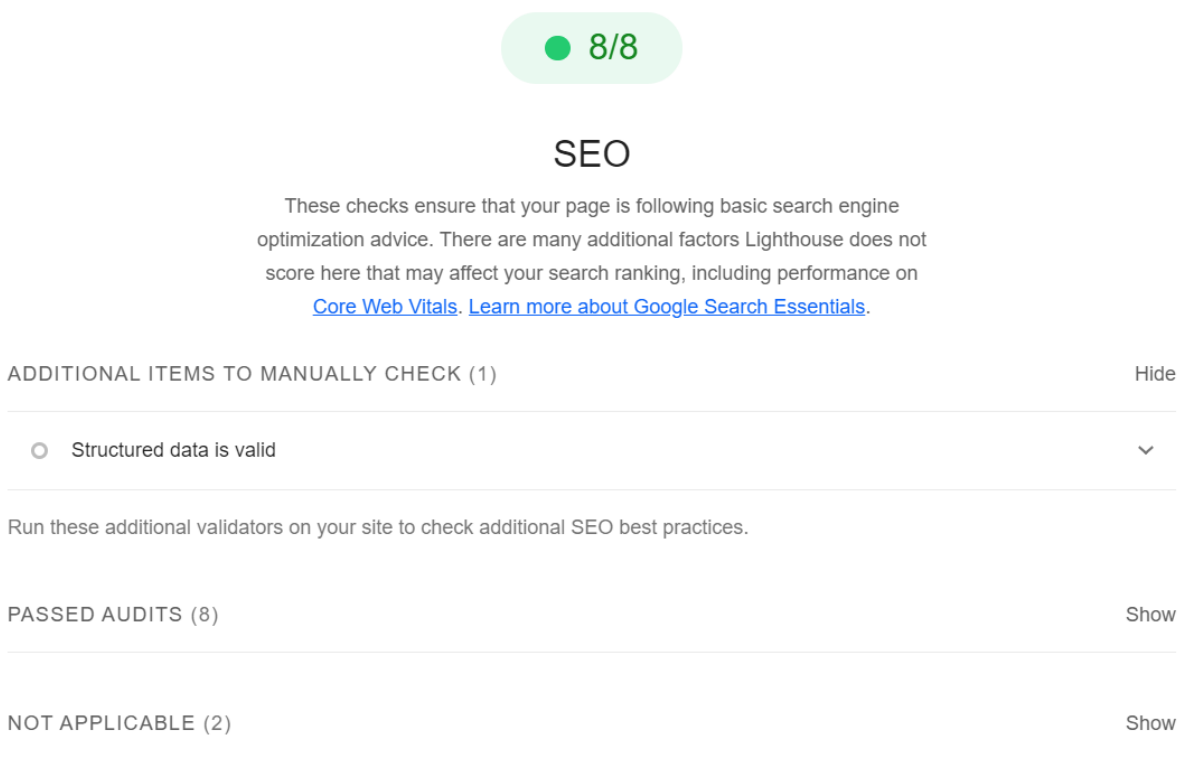
ADDITIONAL ITEMS TO MANUALLY CHECK (10) Show

These items address areas which an automated testing tool cannot cover. Learn more in our guide on conducting an accessibility review.

PASSED AUDITS (12) Show

NOT APPLICABLE (47) Show

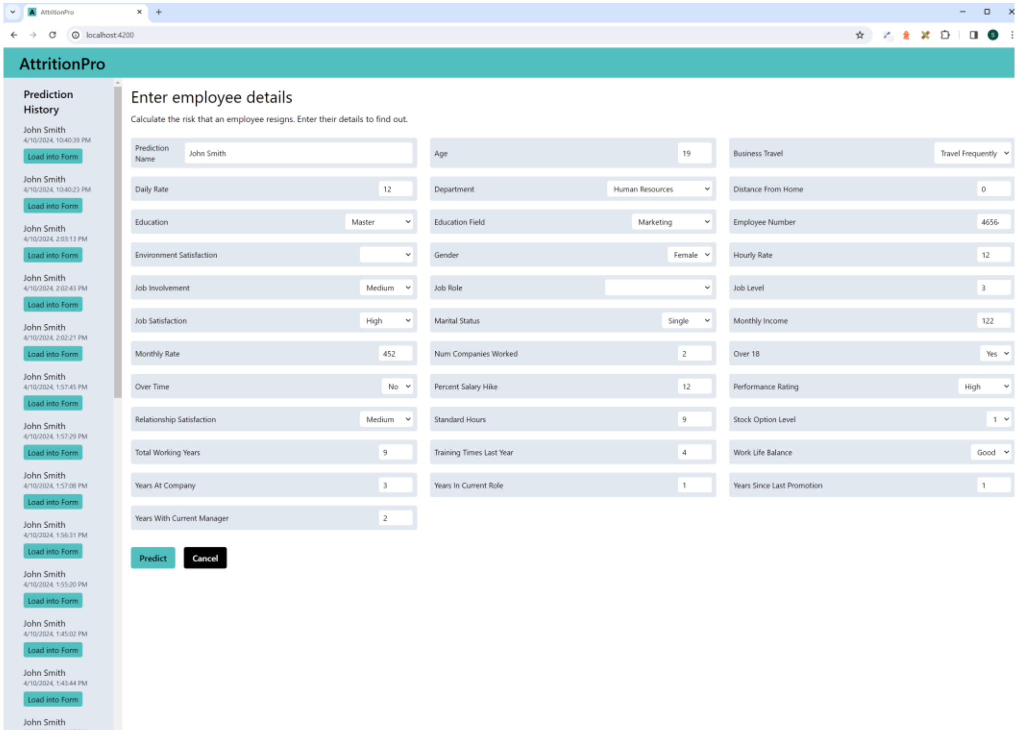


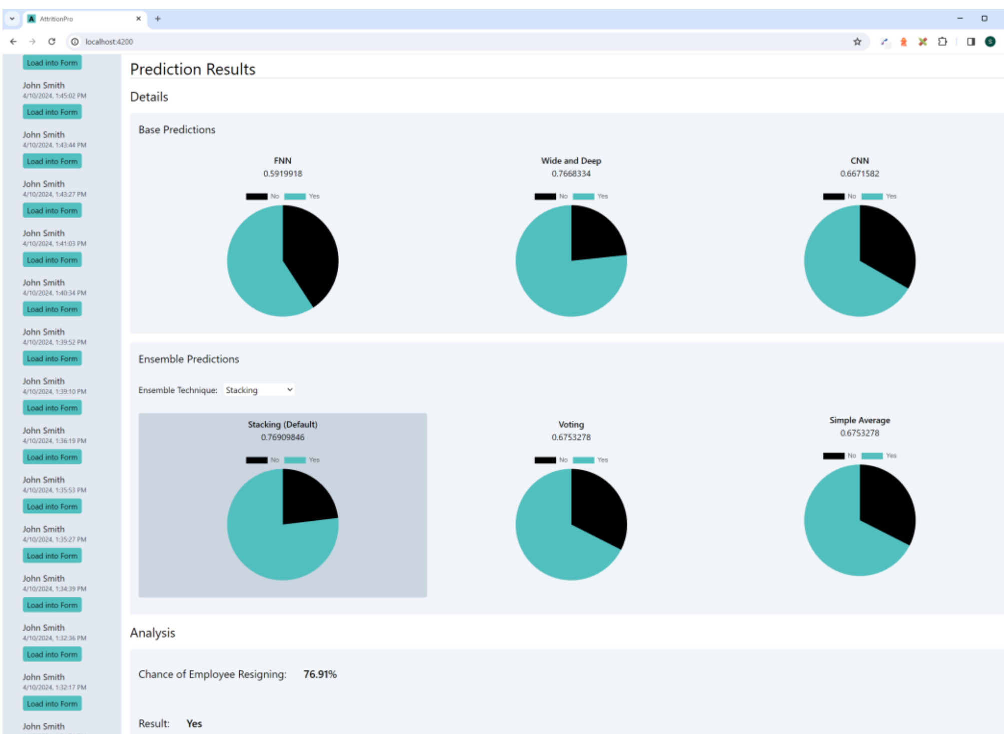


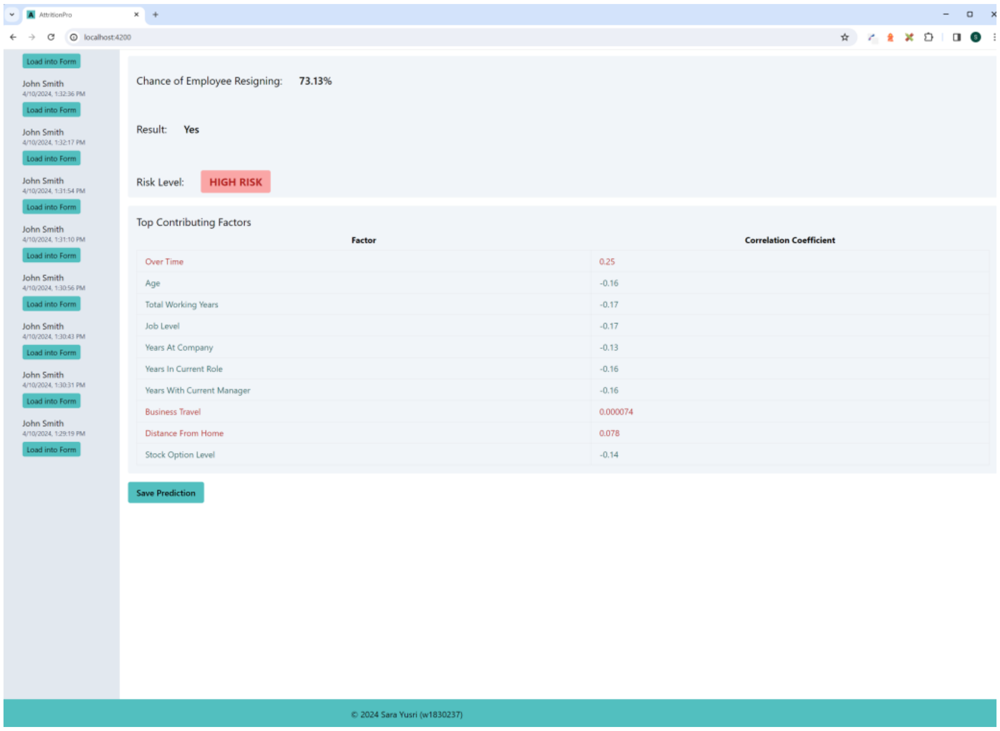
Compatibility Testing

Screenshots from Google Chrome, Mozilla Firefox and Microsoft Edge can be found below.

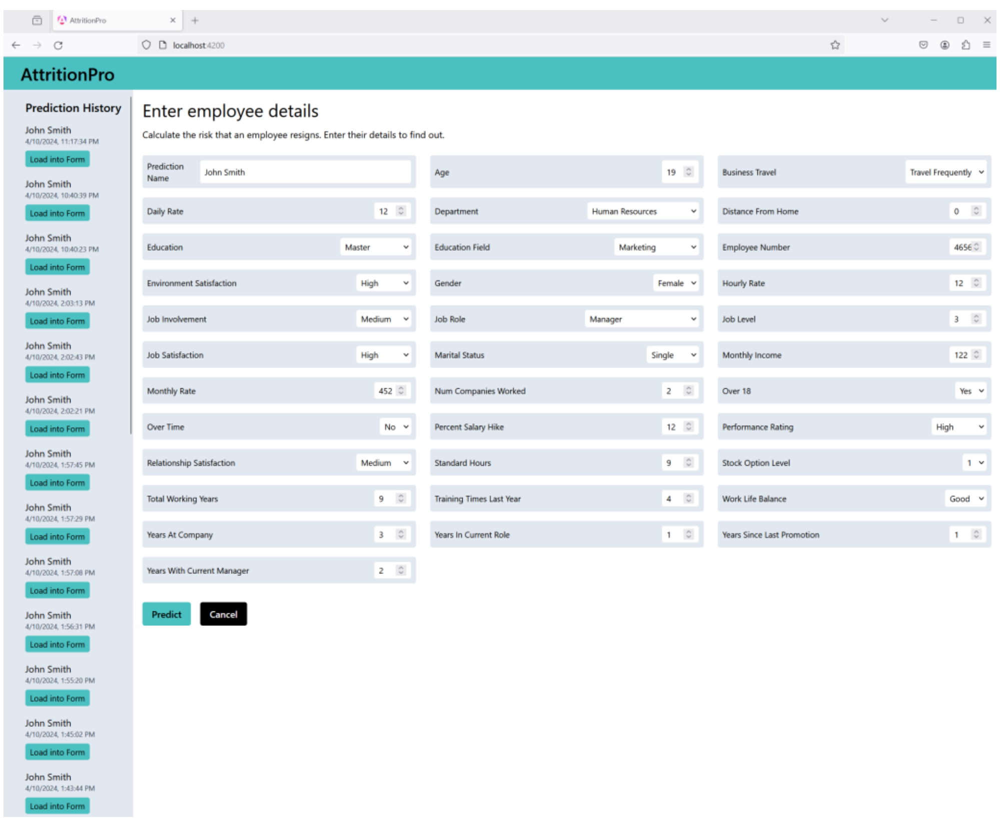
Google Chrome

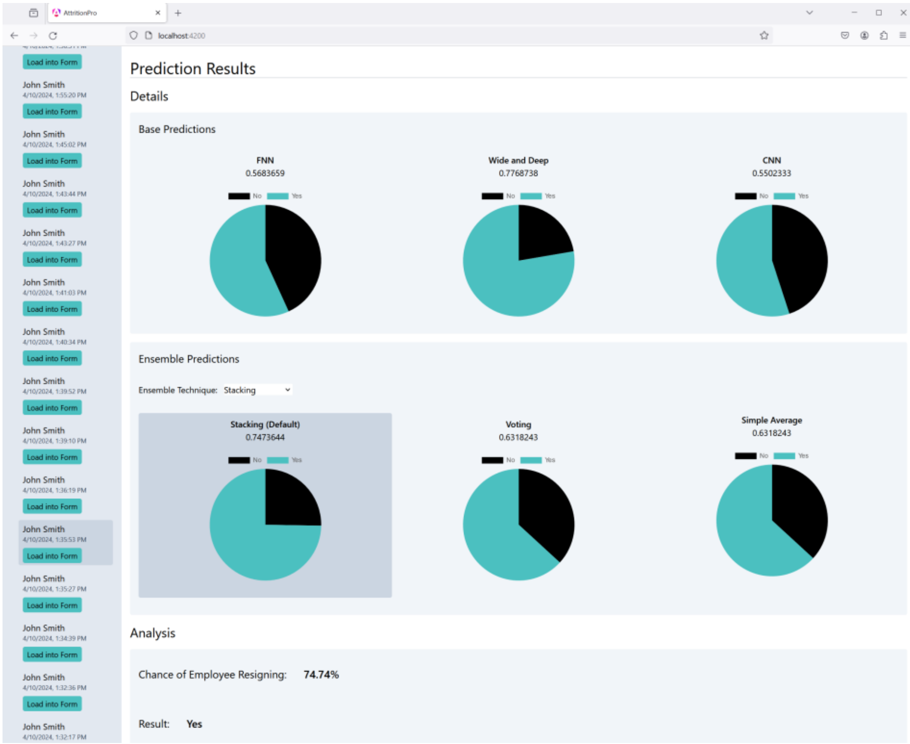

AttritionPro
localhost:4200
AttritionPro
Prediction History
John Smith
Load into Form
Enter employee details
Calculate the risk that an employee resigns. Enter their details to find out.
Prediction Name John Smith
Age 19
Business Travel Travel Frequently
Daily Rate 12
Department Human Resources
Distance From Home 0
Education Master
Education Field Marketing
Employee Number 4656
Environment Satisfaction
Gender Female
Hourly Rate 12
Job Involvement Medium
Job Role
Job Level 3
Job Satisfaction High
Marital Status Single
Monthly Income 122
Monthly Rate 452
Num Companies Worked 2
Over 18 Yes
Over Time No
Percent Salary Hike 12
Performance Rating High
Relationship Satisfaction Medium
Standard Hours 9
Stock Option Level 1
Total Working Years 9
Training Times Last Year 4
Work Life Balance Good
Years At Company 3
Years In Current Role 1
Years Since Last Promotion 1
Years With Current Manager 2
Predict
Cancel

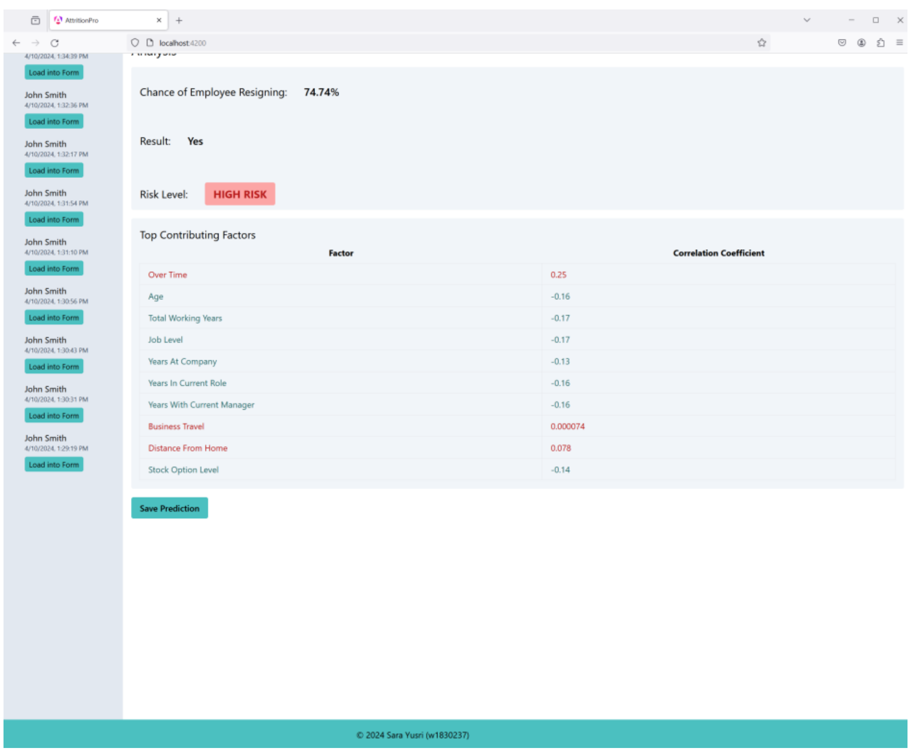

AttritionPro
localhost:4200
Load into Form
John Smith
Prediction Results
Details
Base Predictions
FNN
0.5919918
Wide and Deep
0.7668334
CNN
0.6671582
No
Yes
Ensemble Predictions
Ensemble Technique: Stacking
Stacking (Default)
0.76909846
Voting
0.6753278
Simple Average
0.6753278
Analysis
Chance of Employee Resigning: 76.91%
Result: Yes



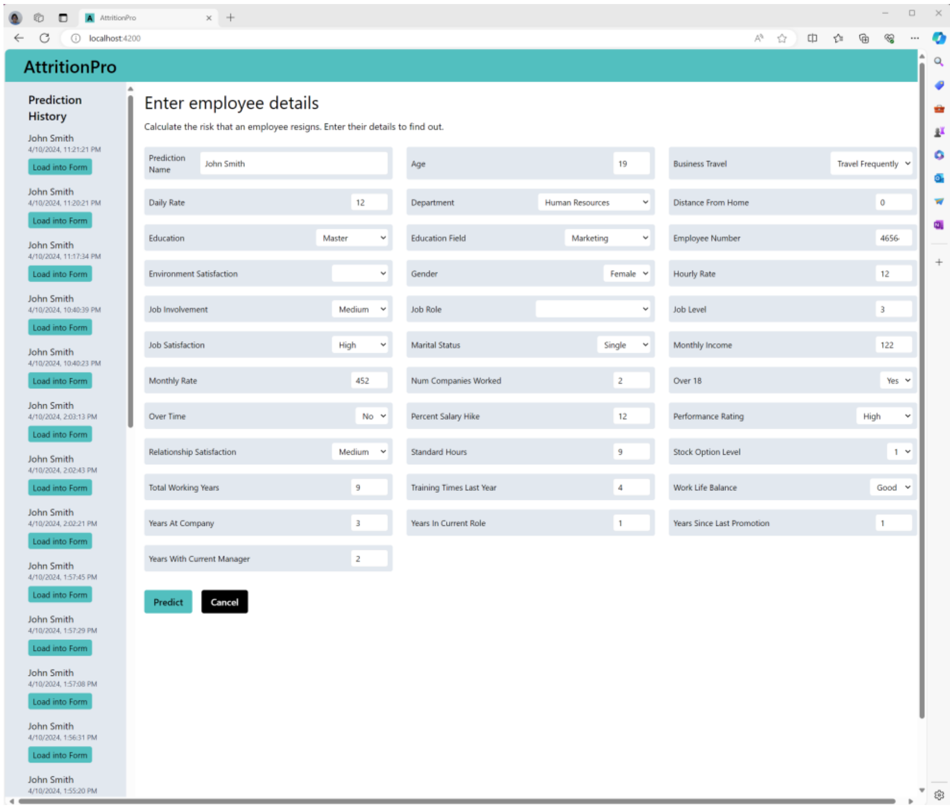
Mozilla Firefox

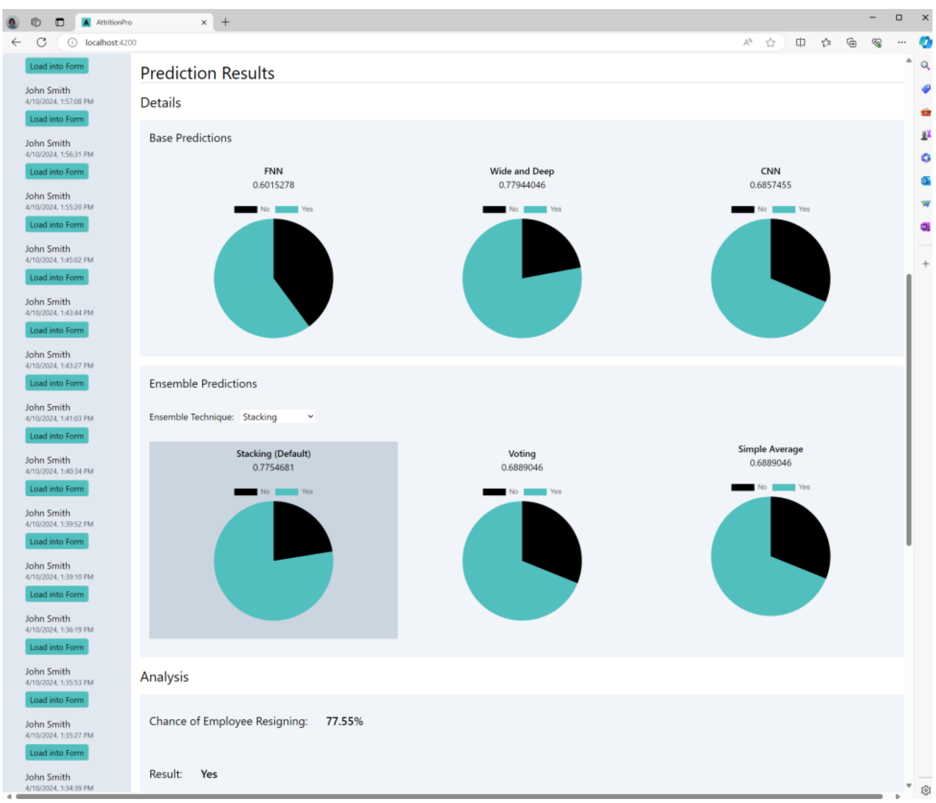


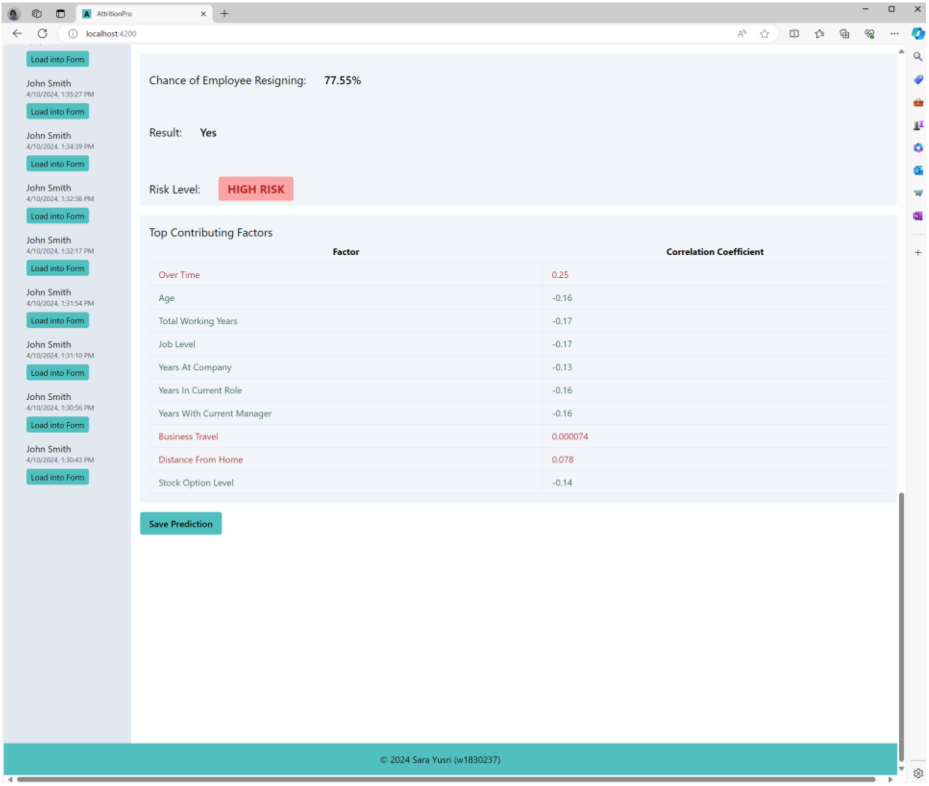




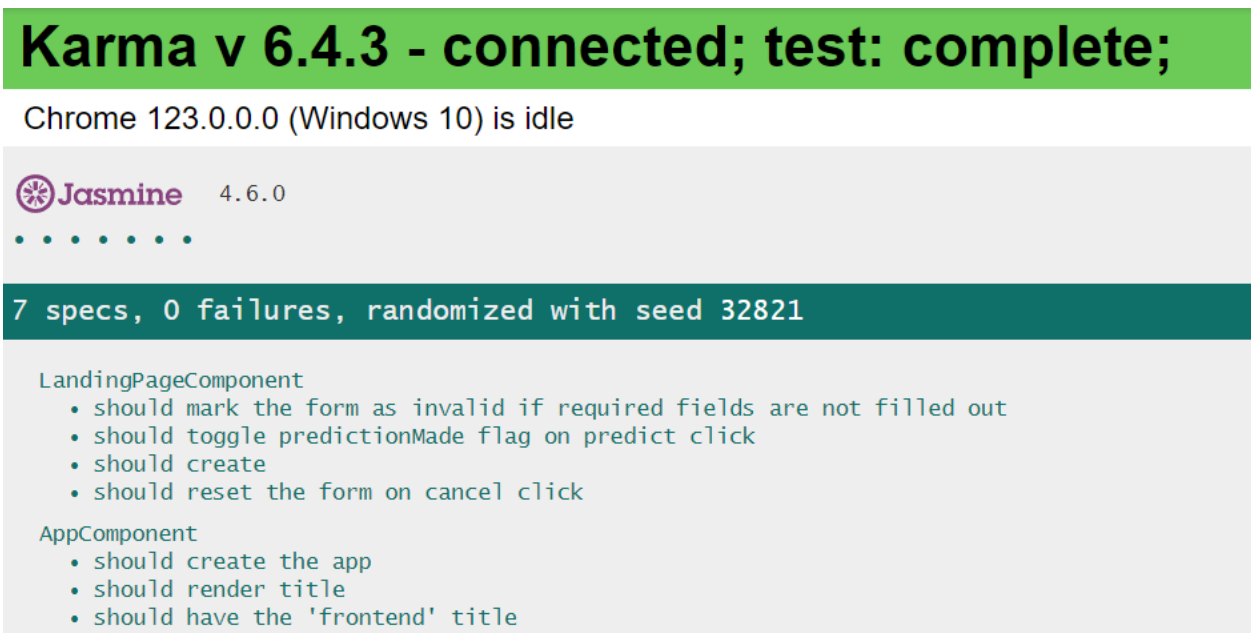
Microsoft Edge


AttritionPro
localhost:4200
AttritionPro
Prediction History
John Smith
4/10/2024, 11:21:21 PM
Load into Form
Enter employee details
Calculate the risk that an employee resigns. Enter their details to find out.
Prediction Name
John Smith
Age
19
Business Travel
Travel Frequently
Daily Rate
12
Department
Human Resources
Distance From Home
0
Education
Master
Education Field
Marketing
Employee Number
4656
Environment Satisfaction
Gender
Female
Hourly Rate
12
Job Involvement
Medium
Job Role
Job Level
3
Job Satisfaction
High
Marital Status
Single
Monthly Income
122
Monthly Rate
452
Num Companies Worked
2
Over 18
Yes
Over Time
No
Percent Salary Hike
12
Performance Rating
High
Relationship Satisfaction
Medium
Standard Hours
9
Stock Option Level
1
Total Working Years
9
Training Times Last Year
4
Work Life Balance
Good
Years At Company
3
Years In Current Role
1
Years Since Last Promotion
1
Years With Current Manager
2
Predict
Cancel


AttritionPro
localhost:4200
Load into Form
John Smith
Prediction Results
Details
Base Predictions
FNN
0.6015278
Wide and Deep
0.77944046
CNN
0.6857455
No
Yes
Ensemble Predictions
Ensemble Technique:
Stacking
Stacking (Default)
0.7754681
Voting
0.6889046
Simple Average
0.6889046
Analysis
Chance of Employee Resigning:
77.55%
Result:
Yes



Unit Testing



APPENDIX E: EVALUATION

Expert Opinion Evaluations

Evaluator	Based on your expertise, what are your overall impressions of the attrition prediction system?	How effective do you think the chosen machine learning models are in accurately predicting employee attrition?	How would you rate the system's usability and user-friendliness from a technical perspective?	Are there any specific areas of improvement or refinement that you would suggest for enhancing the system's performance?	Based on your expertise, how would you assess the complexity of AttritionPro and underlying architecture?	How would you assess the novelty of the attrition prediction model and its underlying architecture?	How much do you think this project contributes to the research domain and body of knowledge?
Mr. Udara Nilupul, Machine Learning Engineer at Ascentic	As an expert in the field, I am impressed by the system's utilization of advanced predictive modeling techniques to forecast employee turnover. Including interpretability measures adds transparency to the predictive outcomes generated.	Selected machine learning models have been shown to be effective in predicting employee turnover. Their ability to identify complex patterns in data and produce accurate predictions contributes to the HR decision-making process.	The system demonstrates a high level of usability and user-friendliness. The intuitive interface and well-organized navigation facilitate seamless interaction with the system, allowing users to efficiently access and utilize its functionalities.	Finetuning the feature selection process might help optimize model performance	AttritionPro seems to show a high level of complexity because it uses multifaceted architecture and advanced modeling techniques. Various data processing pipelines and model training frameworks are used which needs attention to detail and effort to do.	From a technical aspect, the model and its architecture shows a considerable level of novelty. Using state-of-the-art ML algorithms, coupled with advanced feature engineering strategies, sets the system apart. And, adding to that, the use of ensemble techniques and model stacking methods further improves the model's novelty.	This project contributes a lot to the research domain because it is advancing our understanding of the topic of employee attrition prediction. The student has used diverse and various deep learning models and ensemble techniques. This adds knowledge to existing literature and shows new insights into predictive analytics in HR management. The findings from this project have the potential to influence future research directions.
Mr. Hiran Hasanka, Machine Learning Engineer at OREL IT	From a technical standpoint, the system displays a well-engineered architecture. The preprocessing techniques and feature selection methods have been used to optimize the model's performance. The use of ensemble learning techniques enhances the predictive accuracy and reliability.	The selected models demonstrate performance metrics including performance, accuracy, recall, and F1 scores. These metrics demonstrate the model's ability to leverage nuances in employee turnover dynamics.	Overall, the system receives high marks for its technical usability and user-friendliness. It adheres to industry best practices and has an intuitive look and feel.	Improving model interpretability is essential for gaining deeper insights into the factors driving resignations. Techniques such as SHAP (SHapley Additive exPlanations) values or LIME (Local Interpretable Model-agnostic Explanations) can be used to provide more transparent and interpretable explanations of predictions.	AttritionPro exhibits a high level of complexity in its architecture for a final year project. The integration of various ML, DL models, data preprocessing pipelines, and backend infrastructure adds complexity. The system's modular design and well-defined components help manage this complexity effectively.	In terms of novelty, the model implemented in AttritionPro showcases innovative approaches in feature engineering and ensemble learning techniques. The utilization of deep learning architectures with ensemble methods adds a unique element to the predictive capabilities. The system's modular architecture allows for seamless integration of new algorithms, enhancing its novelty.	This project makes a substantial addition to the body of knowledge by pushing the boundaries of prediction in employee attrition domain. By researching and implementing innovative and novel techniques such as ensemble learning and deep learning architectures, the project expands the toolkit available to researchers and practitioners in HR analytics.
Prof. Chandana Gamage, Professor - Computer Science at University of Moratuwa	The system exhibits a well-engineered architecture, incorporating preprocessing techniques and feature selection methods to optimize model performance. The utilization of ensemble learning techniques enhances predictive accuracy and reliability.	The chosen models exhibit strong predictive capabilities, leveraging a combination of feature engineering, model tuning, and ensemble techniques to enhance performance. Their adaptability to diverse datasets ensures reliable predictions across various organizational contexts.	From a technical standpoint, the system demonstrates a high level of usability and user-friendliness. The intuitive interface and well-organized GUI allow seamless interaction with the system, so users can efficiently access and utilize its functionalities.	From a technical perspective, implementing more advanced optimization algorithms and regularization techniques could help mitigate overfitting and improve the generalization ability of the models. Additionally, fine-tuning hyperparameters through more extensive grid search or Bayesian optimization could lead to better model performance.	Considering the technical intricacies involved, AttritionPro can be regarded as moderately complex in its architecture. The system's utilization of diverse data, feature engineering methods, and model ensembles adds depth to its design. Clear documentation and well-defined interfaces contribute to managing this complexity effectively.	The attrition prediction model and its underlying architecture demonstrate a commendable level of novelty, especially in their approach to feature engineering and model selection. By using a combination of traditional statistical methods and modern DL algorithms, the system achieves a nuanced understanding of employee attrition factors.	The project's contribution to the research domain is significant, as it introduces novel methodologies and approaches to employee attrition prediction. By using advanced DL algorithms and data-driven techniques, the project advances HR analytics. The findings not only add to the existing body of knowledge but also provide insights for organizations seeking to mitigate attrition risks and optimize workforce management.
Ms. Thinayani Gamage, Engineer - Technology at EvonSys	AttritionPro seems to be a very well-designed, well-engineered system that I believe provides a lot of contribution to the domain becuase of it's very functional and useful features.	DL Ensemble Methods are very good at predicting employee attrition. I believe they have the potential to really improve the domain of employee attrition prediction. They have been implemented well and with appropriate hyperparameter tunings, and data preprocessing as well.	It has a very pleasant and intuitive UI. I think it's very direct and straightforward to use. Plus, the student has created a user manual for the system too which is very commendable. The system shows a lot of very valuable information to the user.	None that I can think of. The student has implemented many features that are very useful to the HR industry.	In my opinion, I believe this is very complex and difficult project to pull off. The proposed solution was to build several deep learning models and combine them in various ensemble modelling techniques. In my final year project, I used a few ML models and tried to combine them into one ensemble model. So, I can tell that this project was very complicated to design and implement but the student has done a great job and managed to deliver even more than what she had initially proposed.	The models are quite novel. I personally have never come across wide and deep models in this domain before. And the ensembling techniques and hyperparameter tuning are also done to a great degree. This project has tried out a completely new architecture and design and has managed to build a system with really good prediction performance.	This project seems to have made a huge contribution to the domain for being a 1 year undergraduate project. Most existing work only covers 1 or 2 DL models at a time and even, they are implemented only as base models. She has conducted a lot of thorough research, identifed novel approaches and designed and implemented them. She has even managed to optimize them and get them to outperform existing models in the forefront of this domain.
Ms. Aloka Abeysirigunawardana, Senior Data Scientist at Dialog Axiata	AttritionPro seems to present a good solution to the complex problem of employee turnover.	The chosen models represent a very effective collection of models for targeting the complex challenge of employee attrition. They are very effective in accurately identifying at-risk employees in organizations.	The usability looks great and the system is also very user friendly.	Improving the scalability and efficiency of the system's data processing pipeline could be good, especially when we're dealing with large datasets or real-time data streams. This might involve optimizing the data preprocessing steps, or having to use cloud-based solutions.	In my opinion, AttritionPro shows a high level of complexity due to its comprehensive and thorough architecture and sophisticated algorithms. The system's use of advanced machine learning techniques, such as deep learning and ensemble modeling, requires careful management of multiple components. Additionally, ensuring scalability and robustness further contributes to the system's complexity.	The system is using cutting-edge techniques and methodologies. It's using ensemble learning, including stacking and voting mechanisms, showcases a forward-thinking approach to predictive modeling.	This project makes a notable contribution to the research domain by addressing a critical need in HR analytics: employee attrition prediction. The project offers valuable insights into attrition drivers and factors. By shedding light on these, the project adds depth to the existing literature and opens future research avenues for further exploration.
Ms. Azfa Razzaq, Data Scientist at Effective Solutions Pvt Ltd	The attrition prediction system showcases a robust implementation of predictive modeling techniques. The integration of ensemble methods enhances predictive accuracy, making it a valuable tool for HR decision-making.	The chosen machine learning models exhibit commendable performance in accurately identifying potential attrition risks. Their adaptability to diverse datasets ensures reliable predictions across various organizational contexts.	From a technical perspective, the system's usability is commendable. Its intuitive interface and seamless navigation contribute to a positive user experience.	To further enhance system performance, exploring advanced hyperparameter optimization techniques could yield improvements in predictive accuracy and generalization	AttritionPro demonstrates a moderate level of complexity, given its architecture and integration of ensemble techniques.	The attrition prediction model and its underlying architecture demonstrate a notable level of novelty, particularly in their approach to feature engineering and ensemble learning.	This project significantly contributes to the research domain by advancing our understanding of employee attrition dynamics. Its innovative methodologies and empirical insights offer valuable implications for both academia and industry.
Ms. Dashini Moksha, Data Analyst at CUBE	AttritionPro exhibits a robust architecture, with advanced predictive modeling techniques.	DL and ensemble models demonstrate strong predictive capabilities, effectively identifying patterns indicative of employee attrition. Their performance metrics validate their suitability for practical deployment.	The system's usability from a technical perspective is commendable, featuring an intuitive interface and efficient navigation. Users can interact with the system seamlessly, accessing its functionalities with ease.	Enhancing model interpretability could provide deeper insights into attrition drivers and improve stakeholder trust.	AttritionPro's architecture exhibits a high level of complexity, requiring careful management of various components.	The attrition prediction model and its underlying architecture demonstrate a commendable level of novelty, particularly in their utilization of ensemble learning and advanced feature engineering techniques.	This project contributes significantly to the research domain by advancing our understanding of employee attrition dynamics. Its empirical insights and innovative methodologies offer valuable implications for research.

Ms. Tehara Fonseka, Former Machine Learning Engineer at Leather Broadcasting, Inc, Research Student at Western University, Canada	The system demonstrates a well-engineered architecture, incorporating effective preprocessing techniques and ensemble learning methods. These could contribute greatly to enhanced predictive accuracy and reliability.	The models seem to exhibit strong performance in accurately predicting employee attrition, leveraging advanced feature engineering and ensemble techniques.	The system's usability and user-friendliness from a technical perspective are commendable, featuring an intuitive interface and overall good design. It shows the entire process and is therefore, transparent and the visualizations are also very useful.	To further enhance system performance, I recommend optimizing the hyperparameters more and refining the feature selection methods. This might lead to improved predictive accuracy and generalization.	The student's system displays a moderate level of complexity in its architecture. Ensembling several complicated deep learning models into multiple ensembling techniques to get the most accurate one. The student has designed and built a complex and intricate algorithmic design and implementation which is very commendable.	The attrition prediction model and its underlying architecture demonstrate a substantial level of novelty, particularly in their approach to feature engineering and ensemble learning techniques. I, personally, am encountering a solution such as the one proposed for the first time.	I believe this project makes a notable contribution to the research domain by advancing our understanding of employee attrition dynamics. Its empirical insights and innovative methodologies offer valuable implications for both academia and industry.
Ms. Thamali Wijewardhana, Lead Machine Learning Engineer at Synopsys Inc	The attrition prediction system exhibits a robust and well-structured architecture, integrating various preprocessing techniques and ensemble learning methodologies. These elements are integrated effectively, suggesting potential for heightened predictive accuracy and reliability.	The selected machine learning models demonstrate commendable performance in accurately forecasting employee attrition. Many advanced feature engineering and ensemble techniques, these models exhibit strong predictive capabilities.	From a technical standpoint, the system showcases commendable usability and user-friendliness. Its intuitive interface, coupled with transparent process flow and informative visualizations, contributes to an overall positive user experience.	To further enhance the system's performance, I recommend conducting a thorough optimization of hyperparameters and refining feature selection methodologies. These refinements have the potential to elevate predictive accuracy and enhance model generalization.	AttritionPro demonstrates a large level of complexity in its architectural design. Deep learning models and sophisticated ensemble techniques, the system showcases a commendable degree of algorithmic sophistication and implementation intricacy.	The attrition prediction model and its underlying architecture exhibit a notable degree of novelty, particularly in their innovative approach to feature engineering and ensemble learning methodologies. The proposed solution introduces novel insights and methodologies, contributing to the advancement of the research domain.	This project constitutes a significant contribution to the research domain, offering valuable insights into the dynamics of employee attrition. Its empirical findings and methodological innovations hold the potential to enrich both academic discourse and practical applications within the industry. The functionalities and the implementation are also very well executed.
Ms. Sandani Sesanika, Associate ML Engineer at SenzMate	The attrition prediction system that the student has built demonstrates a promising architecture, integrating diverse preprocessing techniques and machine learning models. While there is room for improvement, particularly in model selection and performance optimization, the system shows potential for effective attrition prediction.	The chosen machine learning models show good effectiveness in predicting employee attrition. While they exhibit reasonable performance, there is scope for refinement in feature engineering and model tuning to enhance predictive accuracy.	From a technical standpoint, the system's usability and user-friendliness are adequate. There are areas where the interface could be simplified, and the process flow could be made more intuitive to enhance user experience.	To enhance system performance, I recommend focusing on optimizing feature selection techniques and fine-tuning model hyperparameters. Additionally, exploring alternative machine learning algorithms could provide valuable insights into improving prediction accuracy.	AttritionPro exhibits a moderate level of complexity in its architectural design. While the underlying algorithms and methodologies are well-implemented, there is potential for streamlining the architecture to improve scalability and efficiency.	The attrition prediction model and its underlying architecture demonstrate moderate novelty. While they leverage established techniques in feature engineering and machine learning, there is scope for innovation in exploring novel data sources and incorporating advanced modeling approaches.	This project makes a good contribution to the research domain by offering insights into employee attrition dynamics. While its empirical findings provide valuable insights, further research and refinement can be made to provide a great impact on the body of knowledge in this field.
Mr. Subhath Abeysekara, Software Architect at University of Moratuwa	The architecture of the attrition prediction system is strong and well-designed, combining advanced preprocessing methods and machine learning algorithms. Its all-encompassing strategy raises the possibility of precise forecasting and insightful understanding of the dynamics of employee attrition.	The selected machine learning models show good predictive power for employee attrition. These models, which make use of sophisticated feature engineering and ensemble learning techniques, provide excellent predicted accuracy and dependability.	Technically speaking, the system is quite user-friendly and easy to operate. Its user-friendly interface improves usability and makes interacting with the system easy. It also has a clear process flow and helpful visuals.	I suggest experimenting with different machine learning algorithms and investigating new feature engineering techniques to further improve system performance. Optimizing prediction accuracy and generalization could also be achieved by adjusting model hyperparameters.	AttritionPro's architectural design demonstrates a significant degree of complexity. The system has a high level of algorithmic sophistication and implementation complexity, incorporating complex deep learning models and sophisticated ensemble approaches.	Both the underlying architecture and the attrition prediction model demonstrate a high degree of innovation. Their novel use of ensemble learning and feature engineering approaches advances the state-of-the-art in employee attrition prediction.	This investigation advances our understanding of the dynamics of employee attrition, which is a significant contribution to the research field. The body of knowledge in this discipline is enhanced by its methodological breakthroughs and empirical insights, which have significant consequences for academia and industry.
Mr. Kalinga Fernando, AI Engineer at Analog Inference	With a variety of preprocessing methods and machine learning models, the attrition prediction system exhibits a thoughtful architecture. Although it is promising, there are certain areas where it may be improved in terms of prediction.	The machine learning models that were selected show a moderate level of efficacy in forecasting employee attrition. Although they function rather well, they could be more accurate in their predictions with time with more feature engineering and optimization of models.	From a technical standpoint, the system's usability and user-friendliness are satisfactory. However, there are opportunities to streamline the interface and improve the clarity of the process flow to enhance the user experience.	I recommend experimenting with various other machine learning and DL algorithms and investigating other feature engineering strategies to improve system performance. Optimizing model hyperparameters may also result in increased prediction accuracy.	AttritionPro demonstrates a moderate level of complexity in its architectural design. While the underlying algorithms are well-implemented, there is potential for simplification and optimization to improve scalability and efficiency.	The attrition prediction model and its underlying architecture show moderate novelty. While they incorporate established techniques, there is scope for innovation in exploring new data sources and incorporating advanced modeling approaches.	This project makes a valuable contribution to the research domain by providing insights into employee attrition dynamics. Its findings and methodologies offer valuable implications for both academic research and practical applications in the field, enriching the body of knowledge in this area.

Focus Group Evaluations

Evaluator	As a professional in the HR industry, how intuitive and easy-to-use do you find AttritionPro?	Have you encountered any challenges or difficulties while interacting with the system or interpreting its results?	Does the system's predictions resonate with the factors typically associated with employee attrition and resignation in your organization?	How valuable do you perceive the system's predictions in terms of aiding HR decision-making and strategic planning? (1 - 5 Scale)	Are there any additional features or functionalities that you believe would enhance the system's utility and practicality in an HR context?	Additional Notes or Remarks
Ms. Hashini Perera, Senior HR Executive at Grains 'N' Green	I found AttritionPro to be very user-friendly and intuitive as an HR professional. Even non-technical individuals can engage with it easily thanks to its user-friendly interface.	I had very minimal difficulty in using the system, this had to do with deciphering some technical language in the results. But the prediction results and analysis were very useful and easy to understand and interpret. Still, these were easily fixed with a little more direction.	The causes causing attrition in our organization are in good agreement with the predictions made by the system. It correctly pinpoints important trends and indicators, offering insightful information for HR decision-making.	5	Additional features such as real-time monitoring of attrition risk levels and customizable dashboards for HR analytics would further enhance the system's utility and practicality in our HR context.	For our HR department, the predictions produced by AttritionPro are quite helpful. They offer practical insights that support proactive talent management programs and strategic planning.
Mr. Kannan Periyasamy, HR Manager - System & Services at Daraz (Alibaba Group)	For HR experts, AttritionPro provides a very user-friendly and intuitive interface. Even those with little technological knowledge can easily use it thanks to its simple navigation and intuitive layout.	There were no major issues that I ran across when using the system.	Yes, the factors influencing employee attrition in our company are well represented by the system's projections. The insights given are in good agreement with HR observations and internal data.	5	The system's usefulness and efficacy in our HR environment would be further increased with the addition of features like automated alerts for attrition risk and individualized recommendations for employee retention actions.	
Mr. Thilina Chandrasekara, Human Resources Manager at Sheraton Kosgoda Turtle Beach Resort	AttritionPro is a very user-friendly and intuitive tool for HR professionals. The attrition prediction process is made easier by its well-organized features and user-friendly interface, making it suitable for users with different degrees of technical competence.	I had very few difficulties utilizing the system. It was very straightforward and to the point.	The factors linked to employee attrition at our firm are in line with the predictions made by the system. Our HR data and organizational observations are in good agreement with the patterns and indicators that have been found.	5	The system's usefulness and practicality in our HR environment would be further increased by adding more capabilities like configurable reporting options and interactive visualization tools for investigating attrition trends.	Strategic planning and HR decision-making benefit greatly from the forecasts produced by AttritionPro. They provide useful information that guides proactive talent management techniques and focused interventions.

APPENDIX F: CONCLUSION

Problem / Challenge	**Solution**
Availability and Quality of Data	Obtaining extensive and high-quality worker data can be challenging. (Raza et al., 2022).
Model Overfitting and Generalization	Overfitting occurs when a model learns to memorize the training data instead of capturing underlying patterns. To address this, techniques such as regularization, cross-validation, and early stopping can be employed to prevent overfitting and improve model generalization. Additionally, ensemble methods like stacking and bagging can help mitigate the effects of overfitting by combining multiple models (Raza et al., 2022).
Class Imbalance	Class imbalance refers to unequal distribution of classes in the dataset, which can skew model predictions. To tackle this issue, techniques such as oversampling, undersampling, and Synthetic Minority Over-sampling Technique (SMOTE) can be used to balance the classes. Moreover, performance metrics like F1 score and Area Under the Receiver Operating Characteristic (ROC) Curve (AUC-ROC) are more suitable for evaluating imbalanced datasets (Raza et al., 2022).

AttritionPro: An Employee Attrition Prediction System using Deep Learning Ensemble Techniques

Sara Yusri
Department of Computing
Informatics Institute of Technology
No.57, Ramakrishna Road, Colombo 6, Sri Lanka
sarayusri17@gmail.com

Banuka Athuraliya
Department of Computing
Informatics Institute of Technology
No.57, Ramakrishna Road, Colombo 6, Sri Lanka
banu@androdollar.com

***Abstract*—Employee attrition is a problem that affects many organizations and companies. Many companies are actively strategizing and investing into employee retention strategies and experts in the field in order to retain their valuable employees. This project proposes an employee attrition prediction system, AttritionPro, that is able to utilize deep learning models applied in ensemble techniques to produce reliable and accurate predictions. The purpose of the system is to serve as a preemptive measure for dealing with employee attrition before it happens and allow managements to develop retention plans and make tactical decisions using forecasts of employee resignations.**

The proposed functionalities of AttritionPro will allow HR departments to preemptively forecast employee resignations, the attrition risk level of an employee and generate a breakdown of the features contributing to that employee's attrition. This project tackles several aspects of the problem domain of Employee Attrition and contributes valuable research and insights into the problem and research domain. Ensemble methods such as stacking, voting, and simple averaging are used to combine various deep learning methods, including convolutional neural networks (CNN) and feedforward neural networks (FNN) and Wide and Deep models to achieve the best results. This study demonstrates the effectiveness of deep learning in identifying risk factors and recommending retention programs through evaluation and analysis. The results of the study indicate that Stacking gives the best accuracy and performance of these models. These findings contribute to the advancement of HR analytics and talent management practices, providing insights for organizations looking to reduce employee turnover.

***Index Terms*—Deep Learning, Ensemble Technique, Employee Attrition, Retention Plan, HR Management**

I. Introduction

Employee attrition, commonly known as turnover, is a pervasive issue faced by organizations across industries. It refers to the phenomenon of employees leaving their jobs voluntarily or involuntarily, and it has significant implications for organizational performance, productivity, and profitability. High turnover rates can disrupt operations, reduce team cohesion, and lead to a loss of institutional knowledge and expertise. Moreover, the costs associated with recruiting, hiring, and training new employees can be substantial, further exacerbating the impact of turnover on an organization's bottom line.

In recent years, the advent of advanced analytics and machine learning has provided organizations with new tools and techniques for addressing the challenges of employee attrition. By leveraging historical data on employee demographics, job characteristics, performance metrics, and other relevant variables, organizations can develop predictive models to forecast the likelihood of employee turnover. These models can help identify at-risk employees and enable organizations to implement targeted retention strategies to mitigate the negative impacts of turnover.

In this paper, we present AttritionPro, an employee attrition prediction system that utilizes deep learning ensemble techniques to forecast employee turnover. AttritionPro combines the predictions of multiple deep learning models, including Feedforward Neural Networks (FNN), Wide and Deep models, and Convolutional Neural Networks (CNN), to improve prediction accuracy. By leveraging the strengths of each individual model and aggregating their predictions, AttritionPro provides organizations with a comprehensive and reliable tool for predicting employee turnover and implementing proactive retention strategies.

II. Methodology

The development of AttritionPro involved several key steps, including data collection, preprocessing, model selection, and ensemble techniques.

A. Data Collection

The dataset selected is an open-source dataset created by IBM Scientists to represent various features of 1470 employees. The data requirements of the system were established by referring to researchers' findings on factors that most affect employee attrition: age, education level, job satisfaction, monthly income, salary increment percentage, work life balance, standard hours in a working day, time since last promotion, relationship status, years at company.

The dataset chosen aligns with the data requirements of the project, due to it having the following columns: