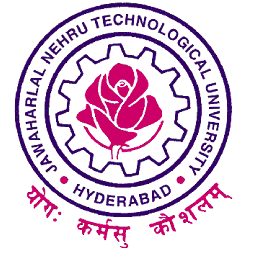
**DETECTION AND PREVENTION OF WORKFLOW ATTRITION**

**An Industry Oriented Mini Project Report**

***Submitted to***



# Jawaharlal Nehru Technological University, Hyderabad

*In partial fulfillment of the requirements for the*

*award of the degree of*

**BACHELOR OF TECHNOLOGY**

**in**

**ARTIFICIAL INTELLIGENCE & MACHINE LEARNING**

By

**K. KOUSHIK (20VE1A6690)**

**M. SANKALP (20VE1A6691)**

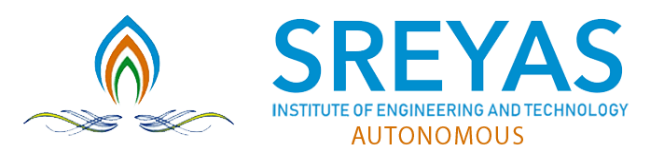
**M. HITHISHINI (20VE1A6692)**

**M. AKSHAYA (20VE1A6693)**

**Under the Guidance of**

**Dr. K. MADAN MOHAN**

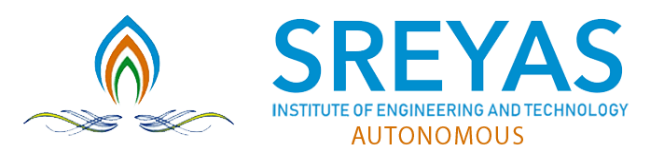
**Assistant Professor**



**DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

Approved by AICTE, New Delhi | Affiliated to JNTUH, Hyderabad | Accredited by NAAC “A” Grade & NBA| Hyderabad | PIN: 500068

(2020-2024)



**DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

Approved by AICTE, New Delhi | Affiliated to JNTUH, Hyderabad | Accredited by NAAC “A” Grade & NBA| Hyderabad | PIN: 500068

**Certificate**

This is to certify that the Industry Oriented Mini Project Report on ***“DETECTION AND PREVENTION OF WORKFLOW ATTRITION***” submitted by **K. Koushik, M. Sankalp, M. Hithishini, M. Akshaya** bearing Hall Ticket No’s.**20VE1A6690, 20VE1A6691, 20VE1A6692, 20VE1A6693** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Artificial Intelligence & Machine Learning** from Jawaharlal Nehru Technological University, Kukatpally, Hyderabad for the academic year 2023-24 is a record of bonafide work carried out by him / her under our guidance and Supervision.

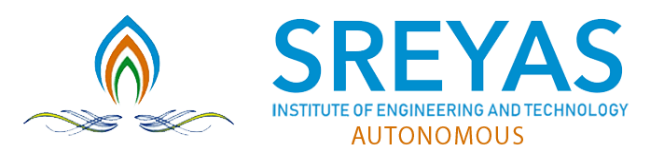
**Internal Guide Head of the Department**

**Dr. K. Madan Mohan Dr. A. Swathi**

**Asst. Professor**

**Mini-Project Coordinator Signature of the External Examiner**

**Mr. S. Prudhvi Raj**



**DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

Approved by AICTE, New Delhi | Affiliated to JNTUH, Hyderabad | Accredited by NAAC “A” Grade & NBA| Hyderabad | PIN: 500068

**DECLARATION**

We, **K. Koushik, M. Sankalp, M. Hithishini, M. Akshaya**, bearing Roll No’s **20VE1A6690, 20VE1A6691, 20VE1A6692, 20VE1A6693** hereby declare that the Project titled "***DETECTION AND PREVENTION OF WORKFLOW ATTRITION***” done by us under the guidance of **Dr. K. Madan Mohan**, which is submitted in the partial fulfillment of the requirement for the award of the B.Tech degree in **Artificial Intelligence & Machine Learning** at **Sreyas Institute of Engineering & Technology** for Jawaharlal Nehru Technological University, Hyderabad is our original work.

**K. KOUSHIK 20VE1A6690**

**M. SANKALP 20VE1A6691**

**M. HITHISHINI 20VE1A6692**

**M. AKSHAYA 20VE1A6693**

**ACKNOWLEDGEMENT**

The successful completion of any task would be incomplete without mention of the people who made it possible through their guidance and encouragement crowns all the efforts with success.

We take this opportunity to acknowledge with thanks and a deep sense of gratitude to **Dr. K. Madan Mohan**, **Assistant Professor**, for her constant encouragement and valuable guidance during the project work.

A Special vote of Thanks to **Dr. A. SWATHI (Head of the Department, AIML) and Mr. S. PRUDHVIRAJ (Seminar Co-Ordinator)** has been a source of Continuous motivation and support. They had taken time and effort to guide and correct me all through the span of this work.

We owe everything to the **Department Faculty, Principal** and the **Management** who made my term at Sreyas Institute of Engineering and Technology a stepping stone for my career. I treasure every moment I have spent in college.

Last but not the least, my heartiest gratitude to my parents and friends for their continuous encouragement and blessings. Without their support, this work would not have been possible.

**K. KOUSHIK 20VE1A6690**

**M. SANKALP 20VE1A6691**

**M. HITHISHINI 20VE1A6692**

**M. AKSHAYA 20VE1A6693**

# CHAPTERS INDEX

# CHAPTER 1. INTRODUCTION………………………………………………………1

## Problem Statement………………………………………………………….………..3

## **CHAPTER 2. LITERATURE SURVEY……………………..………..……………….5**

## 2.1 Existing System……...…………..................................................................................7

## 2.2 Proposed System….......................................................................................................9

## **CHAPTER 3. SYSTEM DESIGN….………………………………………………….11**

## 3.1 Importance of Design………………………………………………………………...11

## 3.2 UML Diagrams………………………………………………………………………14

## 3.2.1 Use Case Diagram

## 3.2.2 Sequence Diagram

## 3.2.3 Activity Diagram

## 3.2.4 System Architecture

## 3.3 Functional Requirements…………………………………………………………….21

## **CHAPTER 4. IMPLEMENTATION………………………………………………….22**

4.1 Module Description………………………………………………………………….22

4.2 Module Components…………………………………………………………………23

4.2.1 Dataset

4.2.2 Data Analysis and Visualization

4.2.3 Web Interface

4.3 Sample Code………………………………………………………………………....27

4.3.1 Long Short-Term Memory

4.3.2 Single Perceptron Layer model

4.3.3 Sequential Feed-Forward Neural Networks

4.3.4 Ensemble Methods (Bagging, Boosting, Staking)

4.3.5 Web Development (Back-End)

**CHAPTER 5. TESTING………………………………………………………….……35**

5.1 Importance of Testing………………………………………………………………..35

5.2 Types of Testing………………………………………………………………...…...36

5.2.1 Model Testing

5.2.2 Web Interface Testing

5.2.3 End-to-End Testing

**CHAPTER 6. RESULTS………………………………………………………………40**

**CHAPTER 7. CONCLUSION AND FUTURE SCOPE……………………………..47**

**CHAPTER 8. REFERENCES…………………………………………………………49**

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Fig. No** | **Name of Figures** | **Page No** |
| **Fig.1** | **Use-Case Diagram for the Attrition detection and prevention process** | **15** |
| **Fig.2** | **Sequence Diagram for the Attrition detection and prevention process** | **16** |
| **Fig.3** | **Activity Diagram for the Attrition detection and prevention process** | **18** |
| **Fig.4** | **System Architecture for the Attrition detection and prevention process** | **19** |
| **Fig.5** | **Modules** | **23** |
| **Fig.6** | **Comparison of Attrition with No Attrition from the dataset** | **25** |
| **Fig.7** | **Comparison of number of employees and their work mode** | **25** |
| **Fig.8** | **Graph showing how marital status effects Attrition** | **26** |
| **Fig. 9** | **Graph showing how Work mode effects Attrition** | **26** |
| **Fig. 10** | **Graph showing how Job satisfaction effects Attrition** | **26** |
| **Fig. 11** | **Graph showing how gender effects Attrition** | **26** |
| **Fig. 12** | **LSTM Algorithm Output** | **40** |
| **Fig. 13** | **Feed Forward Neural Network Output** | **41** |
| **Fig. 14** | **Single Layer Perceptron Model Output** | **41** |
| **Fig. 15** | **Ensemble Methods Output** | **42** |
| **Fig. 16** | **16 of the Features contributing to the attrition and the Attrition distribution within every feature** | **43** |
| **Fig. 17** | **6 more Features contributing to the attrition and the Attrition distribution within every feature** | **44** |
| **Fig. 18** | **16 of the important Features contributing to the attrition.** | **44** |
| **Fig. 19** | **User Interface of web page** | **45** |
| **Fig. 20** | **Output Showing Attrition with Accuracy** | **46** |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table. No** | **Name of Table** | **Page No** |
| **Table. 1** | **Added features and their types** | **24** |
| **Table. 2** | **Table of already existing features and their types** | **24** |
| **Table. 3** | **Table showing the comparison multiple models and their accuracies** | **42** |

**ABSTRACT**

Any corporation understands the importance of the workforce in attaining and maintaining a competitive advantage. Workflow attrition rates should be recognized as an interfering element in a business's growth.  Making decisions can play an important role in administration and may indicate the most vital component in the planning process. Attrition is a well-known issue that necessitates sound management decisions in order to retain highly qualified staff. In order to reduce workflow attrition, organizations today have a strong business interest in understanding the factors that contribute to this occurrence. There are several factors leading to the attrition. Predicting employee attrition and determining the key contributors to attrition are thus important organizational goals in order to optimize their human resource strategy. Excitingly, Artificial Intelligence (AI), Machine Learning, and Deep Learning have been actively used, in forecasting attrition probabilities in advance using an automated technique. The convergence of cutting-edge technology and smart human resource management is set to reshape the workforce retention landscape. Organizations may proactively manage attrition concerns by leveraging the power of AI, ML, and deep learning, making educated decisions that optimize their human resource and contribute to long-term competitive advantage. The pursuit of improved attrition forecast accuracy indicates a commitment to ongoing improvement and adaptability in an ever-changing business environment. The goal of this research is to utilize machine and deep learning models and compare them to bring out the highest possible accuracy. We aspire to use Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN) Algorithms to reach accuracy up to 94%, compared to the previous high of 92%.

**Keywords:** Workflow Attrition, Predictive model, Deep Learning and Neural Networks, Dataset enhancement

**CHAPTER 1**

**INTRODUCTION**

Workflow attrition in a company refers to the departure of staff by typical processes like retirement and resignation, clients passing away from old age, or layoffs brought on by a shift in the organization's target demographics. An organization's high rate of staff attrition is a serious problem because it has a significant influence on them. When employees leave a company, they take priceless tacit information behind them, frequently giving the company a competitive edge. Staff turnover disrupts the company's throughput since it lacks a creative workforce. The company bears the cost of hiring and training new workers, as well as business disruption, as a result of employee turnover. On the other side, higher retention results in lower hiring and training costs as well as the gradual addition of more seasoned employees to the workforce. The key to building and sustaining a collaborative workplace is to draw in and keep individuals who work together. The forecast of employee attrition before it occurs can help to reduce its influence or the management to halt it. According to the research, happy and motivated employees are more creative, innovative, and achieve considerably more.

The human resources (HR) division can assist in developing such an atmosphere by searching through employees' database records. Management can enhance decisions and lower staff turnover by analyzing these data. Understanding and reducing workflow attrition is critical for organizational resilience in the complex world of labor dynamics. Workflow attrition encompasses a wide range of personnel departures, from traditional reasons such as retirement and resignation to external variables such as client attrition due to aging or strategy shifts. High attrition rates have far-reaching consequences since departing employees bring invaluable tacit information, which often contributes to a company's competitive advantage. The departure of competent workers disrupts workflow and incurs significant expenses in terms of recruiting, training, and business continuity. The key is to create a collaborative workplace in which people are not only recruited but also kept. Predicting staff attrition becomes critical, allowing firms to mitigate its effects or react effective

To begin, we combined deep learning approaches with preprocessing procedures to improve prediction of staff attrition. Second, we examined dataset features to identify critical aspects and supplemented the dataset with modern factors such as work style, feedback, hiring processes, and workplace culture. Third, for realistic results, we ran studies on both balanced and imbalanced datasets. To improve employee attrition prediction, we first integrated the deep learning technique with some preprocessing operations. Second, dataset features are examined to determine their relationship to one another and to find the most essential aspects. We even enhanced the dataset by adding latest aspects that might affect this attrition like work mode, no feedback or recognition, bad hiring process and toxic work culture to name a few. We analyzed the interconnection Third, in order to obtain realistic results, we ran our model on overbalanced and imbalanced datasets. Thus, we are proposing deep and machine learning techniques for predicting employee attrition.

The algorithms of the application are exploring Ensemble models of Deep learning with techniques like Bagging, boosting and stacking to overcome the overfitting of data. CNN algorithm we explore is 1-D Convolutions, RNN algorithms are Bidirectional RNN and LSTM, where in LSTM (Long-Short Term Memory) we use the classic cell states and gates. Finally, our holistic approach to staff attrition prediction, which employs advanced machine and deep learning algorithms, strives to provide employers with proactive talent retention solutions. The combination of HTML and CSS for the front-end and Python Flask for the back-end guarantees a seamless and interactive experience for users attempting to decipher the complexities of labor dynamics. The user-friendly web interface includes input fields for each factor in the dataset, allowing users to enter values that are relevant to their organization's environment.

Following submission, the Python Flask-powered backend interacts easily with the deep learning algorithms. These algorithms, trained on large datasets, process user inputs quickly to estimate whether attrition is likely (Yes) or not (No). The interface also shows users the correctness of the forecast, which provides vital information into the model's evaluation dependability. This web interface is an effective educational tool, providing users with a practical way to understand the impact of numerous workplace characteristics on attrition rates.

This tool becomes a helpful instrument in establishing a proactive and educated approach to talent management as firms traverse the ever-changing world of human resources.

Our proposed model employs ensemble models of deep learning, incorporating techniques such as Bagging, Boosting, and Stacking to mitigate data overfitting. In the realm of CNN algorithms, we explored 1-D Convolutions, while RNN algorithms featured Bidirectional RNN and LSTM, leveraging classic cell states and gates in LSTM.

* 1. **Problem Statement**

The aim of this project is to develop a predictive model using deep learning modules that accurately anticipates employee attrition within any given organization by the company background in work front.

* By analyzing historical data and identifying patterns, the model will help HR professionals and management to make informed decisions, implement effective retention strategies, and ultimately reduce the negative impact of attrition on the organization.
* Our recent surge in employee turnover within the software development team is raising concerns about workload, career advancement opportunities, and work-life balance. We need to investigate these factors and implement targeted solutions to prevent further losses.

Recently there is an increase attention to HR, as worker skills and quality represent factors of growth and a tangible modest benefit for businesses. Once proved its mettle in marketing and sales, analytics and AI is likewise becoming central to employee related choices within HR management. Organizational development mostly depends on staff retention. Dropping employees normally influences the self-esteem of the company and contracting with new employees is extra costly than retaining present ones. Thus, we are proposing deep and machine learning techniques for predicting employee attrition.

**Dataset:** The project will utilize a comprehensive dataset containing information about employees, their attributes, performance metrics, job-related factors, and reasons for leaving (if available). The dataset will be divided into a training set for model development and a test set for evaluation.

**Goals:**

* **Create a Stable Predictive Model:** The project's major purpose is to create a powerful predictive model using deep learning modules. This model will use past data to predict staff attrition inside a business. The model seeks to capture subtle patterns and trends that contribute to employee turnover by applying advanced methodologies.
* **Use analytics and artificial intelligence in human resource management:** Recognizing the growing importance of analytics and artificial intelligence (AI) in a variety of corporate areas, the project intends to apply these technologies to human resource management. The project aims to improve the efficiency and efficacy of HR processes by leveraging analytics and AI, particularly in the context of employee-related decisions. This involves implementing data-driven talent retention strategies.
* **Improve Organizational Development by Retaining Employees:** The project's ultimate purpose is to contribute to organizational development by improving worker retention. Employee turnover can have a negative influence on morale and incur additional costs in recruiting and training new employees.

**CHAPTER 2**

**LITERATURE SURVEY**

Pradip Kumar Talapatra et al.[1] explored the attrition rate and the perspective of industries in India which have factors of their own. They concluded that benefits not meeting their needs is highest factor of attrition.

Fatemeh Mozaffari et al.[2] analyzed and came to a conclusion that by using the gradient boosting model ML model they could achieve their accuracy rate up to 89%.

Ali Raza et al.[3] utilized the Machine Learning Algorithms like Extra Trees Classifier (ETC), Support vector machine (SVM), Logistic Regression (LR), and Decision Tree Classifier (DTC), compared them in accuracy term and could achieve up to 90%. Mohammed A. Abu Rumman et al.[4] explored machine learning and deep learning algorithms , he came to the conclusion that all the ML Algorithms like KNN, Random Forests (RF), and SVM using various parameters.

Shobhanam Krishna et al.[5] explored Employee Attrition Analysis using Random Forest and he found out that the model built using Random Forest Classifier is transformed by SMOTE mechanism to improve target class imbalance. After SMOTE mechanism metrics of the training model are improved however, validation metrics are improved slightly specially sensitivity has very little impact.

Karthik Sekaran et al.[6] explored two powerful Explainable AI (XAI) models named Local Interpretable Model-Agnostic Explainer (LIME) and Shapley Additive eXplainer (SHAP) which unveiled logical insights from the data that could assist the management authorities in countermeasure the risk of employee attrition.

Md. Monir Ahammod Bin Atique et al.[7] used the method called CatBoost in which a state-of-the-art boosting method, CatBoost, and a feature engineering process have been applied for detecting and analyzing employee attrition which reveals the best recall rate of 0.89, with an accuracy of 0.8945.

Sumati Sidharth et al.[8] used Random Forest and the AdaBoost classifier to make predictions. Their paper also introduces the factors that influence employee attrition inside any organization and will provide a clear perspective to top management in making key

decisions regarding the retention of most of the workforce in the organization.

Hamamache Kheddouci et al.[9] explains the prediction systems are not generic and are specific to each company and a generic attrition model based on bipartite graph properties and machine learning algorithms is developed.

Sini Raj Pulari et al.[10] analyzes Employee Attrition Using Machine Learning and Deep Learning Techniques where the analysis of data have used deep learning methodologies and machine learning techniques to gather more valuable insights than using the traditional methods.

The researchers in [11] offered a 3 phase outline for predicting employee attrition. The 1st phase, utilized “max-out” technique for the selection of features to reduce the dataset. Logistic regression model was used, training and testing for the prediction, in 2nd phase. The validation model and assurance investigation was accomplished in the 3rd phase. Their accuracies were low, their system was very complex because of the pre and post processing.

There researchers in [12] employed random forest and classification trees for predicting worker attrition. They began with pre-processed the dataset by omitting the less influential variables by Pearson correlation. Nevertheless, the work demonstrated minor enhancement in accuracy in relation with the other ML techniques.

There researchers in [13] employed tree techniques for the prediction of attrition of employees. The used techniques contain light gradient boosted and random forests that achieved the highest accuracy. Private Dataset was used that consisted of 5,550 samples. Other works, such as in [14], employed private dataset also, this prevent us from making a comparing their work with the current study. According to the book “How the Naïve Bayes Classifier Works in Machine Learning” by [25]

In term of classification Naïve Bayes Algorithm is a fast, highly scalable algorithm that can be used for Binary and Multiclass classification. It is a simple algorithm that depends on doing a bunch of counts but can easily be used on small to large dataset. It is also a popular choice for text classification problems.

In the study conducted by Q. a. Al-Radaideh and E. Al Naqi, they used Naïve Bayes classification algorithm to predict a target class depending on in its calculations on probabilities, namely Bayesian theorem. It was also stated that because of this use, results from classifier are more accurate and effective, and more sensitive to new data added to the dataset.

Also, in another study conducted by[27]2 N. Deepa et al, Naive Bayes classifier are used for the prediction of COVID-19. The accuracy of NB got a 99%. Another study applying the Naïve Bayes is a research by A. [28] Yudhana et al,applying the soil nitrogen mapping with a smart prototype using the TCS3200 sensor combined with Naive Bayes algorithm and GIS. . A research by L. Dey et al, utilized Naïve bayes in sentiment analysis. From the result obtained it was seen that in case of movie reviews Naïve Bayes ‘got better results than K-NN.

Similarly, Firth et al. (2007) [29] state that 'work stresses' such as extra hours or job ambiguity have a direct psychological influence on individuals and can be a prelude to depression for their perseverance. Keeping track of overtime hours is also important. as well as the working relationships between managers and employees can contribute to a reduction in unhappiness with their position. Furthermore, it boosts motivation. Employees and lowers the likelihood of attrition.

**2.1 Existing System**

Employee attrition is a critical concern that has drawn the attention of researchers who have sought to unravel its intricacies from various perspectives. Some scholars have delved into the nuanced analysis of employee behavior, aiming to unearth the multifaceted causes influencing their decisions to either remain within a business or seek alternative employment opportunities. A noteworthy study in this domain employed advanced Machine Learning (ML) techniques to forecast employee attrition based on a wealth of data pertaining to the workforce. This encompassed the utilization of several ML methods, including K-Nearest Neighbors (KNN), Random Forests (RF), and Support Vector Machines (SVM), with a deliberate exploration of various parameters.

One key aspect of this investigation involved the consideration of three distinct forms of the dataset—unbalanced, under-sampled, and over-sampled. The researchers observed a significant uptick in accuracy when dealing with the over-sampled dataset. However, it was noted that the original dataset's accuracy remained suboptimal, revealing the challenges inherent in addressing imbalances in the data distribution. Notably, the study also ventured into the exploration of various machine learning algorithms, such as the Random Forest Classifier, Light Gradient Boosting Machine (LGBM) Classifier, and Multi-Layer Perceptron (MLP) Classifier. The findings from these explorations highlighted a reasonable level of accuracy derived from these algorithms, underscoring the potential of ML in predicting and understanding employee attrition patterns.

Moreover, the emphasis on employing different machine learning algorithms reflects a recognition of the complexity of the attrition prediction task. Various algorithms offer unique advantages and may be better suited to capturing different nuances within the data. The study prioritizes accuracy as a key metric, acknowledging its crucial role in preventing company losses associated with unexpected employee departures.

In a parallel study utilizing the same dataset, researchers undertook a comparative analysis of ML methods for predicting employee attrition. Naïve Bayes, decision tree, and K-Nearest Neighbors (KNN) were among the algorithms scrutinized. The researchers adopted a meticulous approach, employing cross-validation with a 10-fold strategy and allocating 65% of the dataset for training and 35% for testing. Despite their rigorous methodology, the study's accuracy was found to be comparatively lower than other studies, primarily attributed to the absence of a dedicated data pre-processing phase. This underscores the importance of data cleaning and feature engineering as crucial steps in the ML pipeline to enhance the predictive capabilities of models.

Further, additional research efforts, as exemplified in [14], have employed private datasets, making direct comparisons challenging. While these studies contribute valuable insights, the overarching aim is to continuously strive for enhanced accuracy and confidence in predicting employee attrition.

However, despite the promising outcomes achieved through ML methods, the discussion also sheds light on some challenges encountered in the process. The dichotomy between prediction and detection is acknowledged as a complex task, requiring nuanced approaches within the models. Additionally, the simplicity of the coding process is juxtaposed with the maintenance challenges posed by multiple codes for all attributes. Furthermore, the researchers note that the familiarity with libraries supporting these endeavors is not as widespread, hinting at the ongoing need for increased awareness and utilization of available ML libraries.

In conclusion, the multifaceted exploration of employee attrition showcases the dynamic nature of research in this domain.

The utilization of various ML algorithms and the critical evaluation of their efficacy highlight the complexity and challenges inherent in predicting employee attrition accurately. As the field continues to evolve, there is a continual pursuit of refining methodologies, enhancing accuracy, and overcoming challenges to provide actionable insights for organizations grappling with employee attrition issues. Prediction and detection are challenging tasks working individually in the models.

* Coding is simple but maintaining multiple codes for all the attributes.
* Libraries support was not that much familiar.

**2.2 Proposed System**

The proposed study represents a comprehensive approach to enhancing predictive accuracy through the integration of machine learning (ML), deep learning (DL), and rigorous data preprocessing techniques. The system incorporates three distinct DL algorithms and three ensemble DL methods, each contributing to a robust and accurate prediction model. Specifically, the Long-Short Term Memory (LSTM) algorithm, Sequential Feed-Forward Neural Network algorithm, and Single Perceptron Layer model form the core DL components. In addition, ensemble methods, namely Bagging, Boosting, and Stacking, are employed to further refine predictive capabilities.

To construct the LSTM model, a Keras sequential model is generated. This architecture comprises two LSTM layers, each with 64 units. The first LSTM layer generates sequences, while the second does not—a conventional setup for sequence-to-sequence models. The model is compiled using the Adam optimizer and binary cross-entropy loss function, a common choice for binary classification problems. During training, accuracy is defined as the performance metric to be monitored.

The training and testing data are reshaped to align with the expected input shape of the LSTM model, represented in a 3D format: [samples, time steps, features].

The achieved test accuracy is approximately 94%, indicating that the model correctly classified 94% of instances in the test dataset. This outstanding performance surpasses the scores obtained by the machine learning models employed in both the current study and the existing system. Notably, the dataset is balanced before training, validating, and testing all models, contributing to the improved accuracy of the proposed DL model.

The success of the proposed DL model can be attributed to the strategic application of deep learning techniques and the meticulous utilization of preprocessing methods, including feature selection. The most impactful features identified in the study include Stock Option Level, Monthly Income, Job Satisfaction, Job Involvement, and Total Working Years.

To make the model more accessible, a user-friendly web interface has been developed. This interface considers every factor in the dataset, providing input boxes for users to input values and understand the impact of various factors on the attrition rate. The web page is designed using HTML and CSS for the front-end, while the back-end is powered by Python Flask. Users can easily input values into the provided boxes, submit their inputs, and receive predictions regarding attrition, along with the associated accuracy achieved through DL algorithms.

In conclusion, this study not only answers fundamental questions regarding the implementation of a high-accuracy attrition detection model but also extends its reach through a user-friendly web interface. The integration of advanced DL algorithms, thoughtful preprocessing, and a strategic feature selection process has elevated the predictive capabilities of the model, showcasing the potential of cutting-edge technologies in addressing complex challenges in human resource management.

**CHAPTER 3**

**SYSTEM DESIGN**

**3.1 Importance of Design**

The proposed System has three Deep Learning algorithms and three ensemble DL methods. We used Long-Short term Memory algorithm, Sequential feed-forward neural network algorithm and Single Perceptron layer model. The ensemble methods are Bagging, Boosting and Stacking.

To construct the LSTM Model. A Keras sequential model is generated. There are two LSTM layers with 64 units each added. The first LSTM layer produces sequences, whereas the second does not. This is a common arrangement for sequence-to-sequence models. To prevent overfitting, a dropout layer with a dropout rate of 0.2 is introduced. For binary classification (Attrition prediction), a dense layer with one output unit and a sigmoid activation function is added. The Adam optimizer and binary cross-entropy loss function are used to compile the model. For binary classification problems, binary cross-entropy is often utilized. During training, the accuracy measure is also defined to be monitored. The training and testing data are reshaped to correspond to the LSTM model's expected input shape (3D shape: [samples, time steps, features]).

The output informs you about the performance of the LSTM model in terms of its ability to predict employee attrition on the provided test data. The model achieved a test accuracy of approximately 94%, meaning it correctly classified 94% of the instances in the test dataset.

For a feedforward neural network with successively stacked layers. The ReLU activation function is used in the input layer, which comprises 69 units (neurons). A hidden layer with 32 units and ReLU activation exists. The output layer contains one unit with a sigmoid activation function, which is appropriate for binary classification. The model is built using the Adam optimizer, a binary cross-entropy loss function (often used for binary classification), and accuracy as a training metric. The model is trained with 75 epochs and a batch size of 65 using the training data. A subset of the training data (30%) is utilized as validation data during training to check the model's performance on data that was not encountered during training. The ReLU (Rectified Linear Unit) activation function is a simple non-linear function commonly used in neural networks. It is defined as:

**ReLU(x) = max (0, x)**

*Equation 1: ReLU Equation where x is input to the function*

The inputs are routed via the ReLU (Rectified Linear Unit) activation function, which is:

output\_i = max (0, input i)

where input i represents the weighted sum of the inputs.

The ReLU activation function, like the input layer, is applied to the weighted sum of inputs for each unit in the hidden layer. For a single hidden layer unit:

Input: z = (w1 \* output\_1) + (w2 \* output\_2) +... + (w69 \* output\_69) + b, where w1, w2, ..., w69 are weights, output\_1, output\_2 ,..., output\_69 is input layer outputs, and b is the bias.The hidden layer's outputs serve as inputs to the output layer.

The sigmoid activation function is used, which compresses the output to the range [0, 1], which can be interpreted as a probability. For the single output layer unit:

Input: z = (w1 \* output\_hidden\_1) + (w2 \* output\_hidden\_2) +... + (w32 \* output\_hidden\_32) + b, where w1, w2,..., w32 are weights, output\_hidden\_1, output\_hidden\_2,..., output\_hidden\_32 are hidden layer outputs, and b is bias.

Output = 1 / (1 + exp(-z)), where exp denotes the exponential function.

The script prints the test loss and test accuracy after training. The accuracy shows how successfully the algorithm can classify employees into either 'Attrition' or 'No Attrition'. The greatest test accuracy is 93%, offering a more interpretable assessment of the model's performance.

The Perceptron class in scikit-learn is used to build a Perceptron model with the following parameters:

max\_iter=2000: The maximum number of iterations that the perceptron can perform before converging. This is set at 2000 in order to ensure adequate training.

random\_state=42: For reproducibility, a random seed is used.

The fit approach is used to train the Perceptron model using the training data.The trained Perceptron model is used to make predictions on the test data. y\_pred holds the projected values.Using scikit-learn's accuracy\_score, the script calculates and prints the accuracy of the Perceptron model on the test data. The output (Y) of a single perceptron in a neural network can be calculated as follows for input features X1, X2,..., Xn, weights W1, W2,..., Wn, and bias (threshold) B:

**Y = Σ (Xi \* Wi) + B**

*Equation 2: Single Perceptron layer output equation*

Each component of the equation represents the following:

Y: The perceptron's output.

Xi: The value of the input feature for feature i.

Wi denotes the weight associated with the input feature i.

B: A scalar value representing the bias term.

Σ: The summing operator that adds the products of the input characteristics and their weights.

In the case of a single perceptron with input features Age, BusinessTravel, DailyRate,..., Bad Hiring Process, weights w1, w2, w3,..., wN, and bias b can be used to determine the output (Y) as follows:

Y = w1 \* Age + w2 \* BusinessTravel + w3 \* DailyRate +... + wN \* Unsatisfactory Hiring Process + b

In this equation, Y represents the perceptron's output, which is the weighted sum of the input features plus the bias.

The dataset's input features include Age, BusinessTravel, DailyRate,..., and Bad Hiring Process.

The weights associated with each input feature are denoted by w1, w2, w3,..., wN.

The bias term, b, is a scalar value.

This weighted total is computed by the perceptron, and the resulting value (Y) is utilized to create a binary classification judgment based on whether Y is positive or negative.

In this code, the best accuracy we could achieve in predicting if employees are going to leave their jobs (attrition) based on the input features is 89%.

A Bagging Classifier is created with the Random Forest classifier as the base estimator and 100 ensemble estimators.

The Bagging Classifier assembles a number of base classifiers, each learned on a different subset of the training data. The Random Forest classifier, which was utilized as the basic classifier, is well-known for its ability to handle complex data and forecast well. The model can prevent overfitting and increase generalization by integrating many Random Forest classifiers in a bagging ensemble. The fit approach is used to train the Bagging Classifier on the training data. This yields a Random Forest classifier ensemble. The trained Bagging Classifier is used to make predictions on the test data. The output of the script is the accuracy of the Bagging Classifier on the test data, which represents the proportion of correctly classified instances in the test dataset. The best accuracy is 88%.

To develop an effective classification Boosting model for forecasting employee attrition, the code makes use of XG Boost, a prominent gradient boosting algorithm. XG Boost is a powerful ensemble learning algorithm that can handle complex data and produce high predicted accuracy.

As the basis learner, an XG Boost classifier is setup with several hyperparameters. The following are important hyperparameters:

booster: Gradient boosting type (in this case, 'gbtree').

max\_depth (5): Maximum tree depth.

Learning\_rate: The boosting step size shrinking (0.2).

n\_estimators: The number of rounds of boosting (100).

subsample: The fraction of samples that were used for fitting (0.8).

colsample\_bytree: Feature fraction used for fitting (0.8).

Binary logistic categorization is the goal.

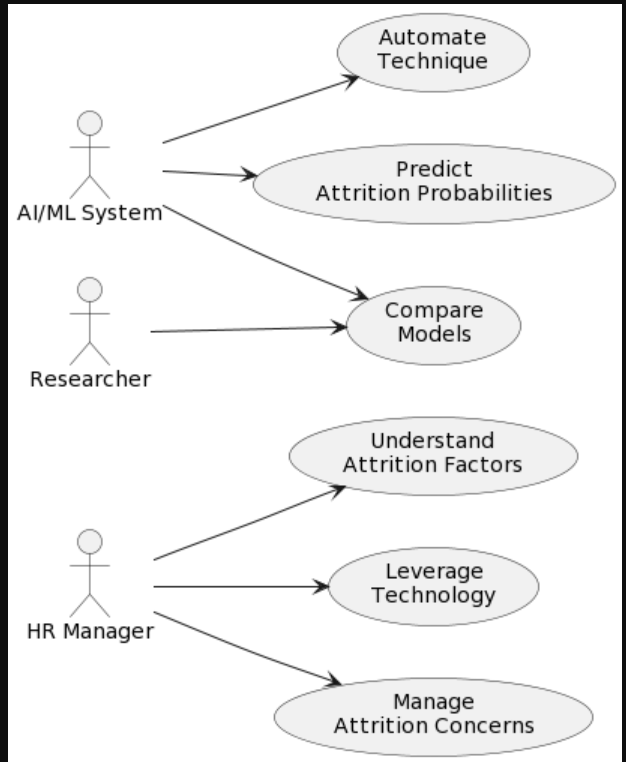
The maximum accuracy that could be achieved through this model is 87%.

The Stacking Classifier from scikit-learn is used to build a stacking model. The basis classifiers and their names are included in the estimator’s parameter. The meta-classifier (Logistic Regression) is specified by the final\_estimator argument. To boost prediction accuracy, the algorithm implements a stacking ensemble learning technique, which combines various basic classifiers (Random Forest and Gradient Boosting) with a meta-classifier (Logistic Regression). Stacking is a powerful strategy that takes advantage of the benefits of various algorithms. This stacking approach often leads to improved predictive accuracy compared to using individual classifiers. The best accuracy we could achieve here is 87%.

**3.2 UML Diagrams**

Using UML diagrams, such as use case diagrams, class diagrams, and sequence diagrams, makes it easier to express system components, interactions, and relationships. UML is a popular visual modeling language for creating, visualizing, specifying, and recording the behavior and structure of software and other systems. With the use of a set of graphical notations supplied by UML, software engineers, system architects, and designers can create visual models of systems. UML diagrams are the most commonly used approach for depicting various system components. UML's visual model makes it possible for stakeholders to discuss, evaluate, and comprehend various system components in a clear and simple manner.

**3.2.1 Use Case Diagram**

****

**Fig 1: Use-Case Diagram for the Attrition detection and prevention process**

System for AI/ML (AI\_ML): The AI/ML System represents automated systems that use artificial intelligence and machine learning.

Function in Use Cases: Automate Technique: The AI/ML System uses advanced algorithms to automate methods, most likely related to forecasting attrition probability.

Predict Attrition Probabilities: Using machine learning models, the AI/ML System is responsible for forecasting attrition probabilities.

Model Comparison: The AI/ML System compares various machine and deep learning models to improve accuracy.

Researcher: The Researcher is an external actor, either an individual or a team, who is involved in research activities.

**Function in Use Cases**: Model Comparison: The Researcher compares various machine and deep learning models, contributing to continuing improvement and flexibility in an ever-changing corporate environment.

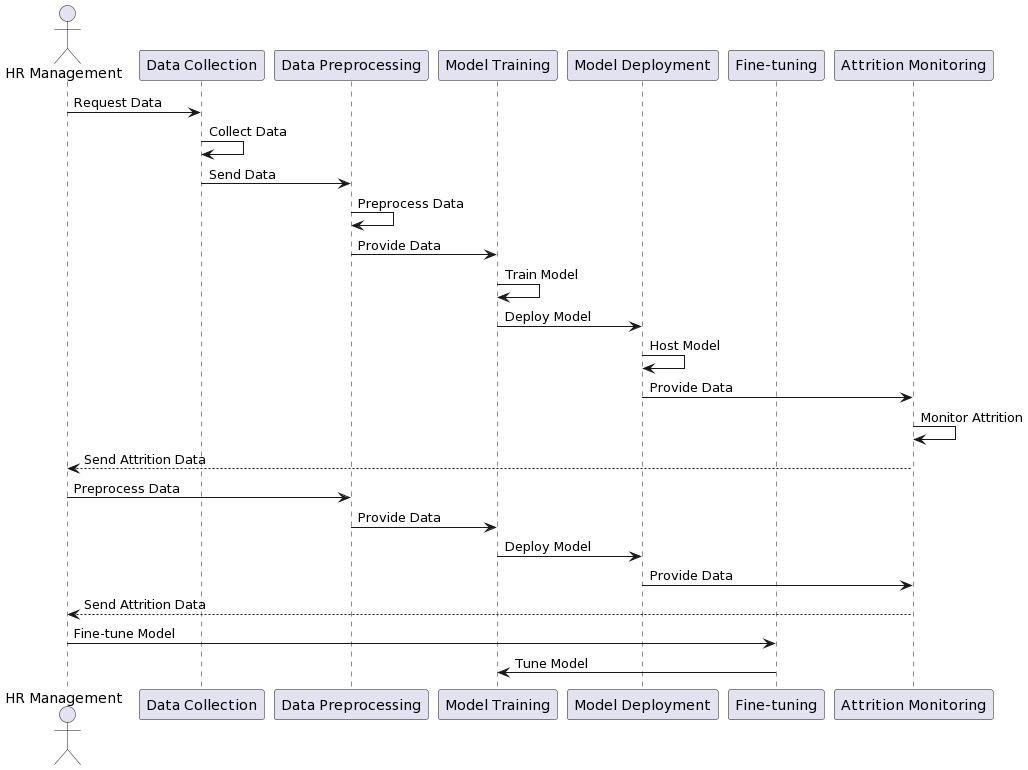
**Human Resources Manager (HRM**): Function: The HR Manager is a significant actor in the organization who is in charge of numerous human resource management activities.

**Function in Use Cases:** Understand the elements Contributing to Workforce Attrition: The HR Manager is responsible in determining the elements that lead to workforce attrition.

**Leverage Technology:** The HR Manager is responsible for utilizing technology for human resource management, including the usage of AI/ML technologies.

**Manage Attrition Concerns:** The HR Manager is actively involved in the organization's attrition management.

**3.2.2 Sequence Diagram**

****

**Fig 2: Sequence Diagram for the Attrition detection and prevention process**

**Recognize Attrition Factors**: The sequence begins with the HR Manager requesting information from the AI/ML System in order to better understand the elements that contribute to attrition.

The AI/ML System activates and analyzes data and patterns in order to provide insights regarding attrition factors. The AI/ML System then sends the results and insights back to the HR Manager.

**Utilize Technology:** The HR Manager makes a request to the AI/ML System to use technology for human resource management, which may include the adoption of AI/ML solutions.

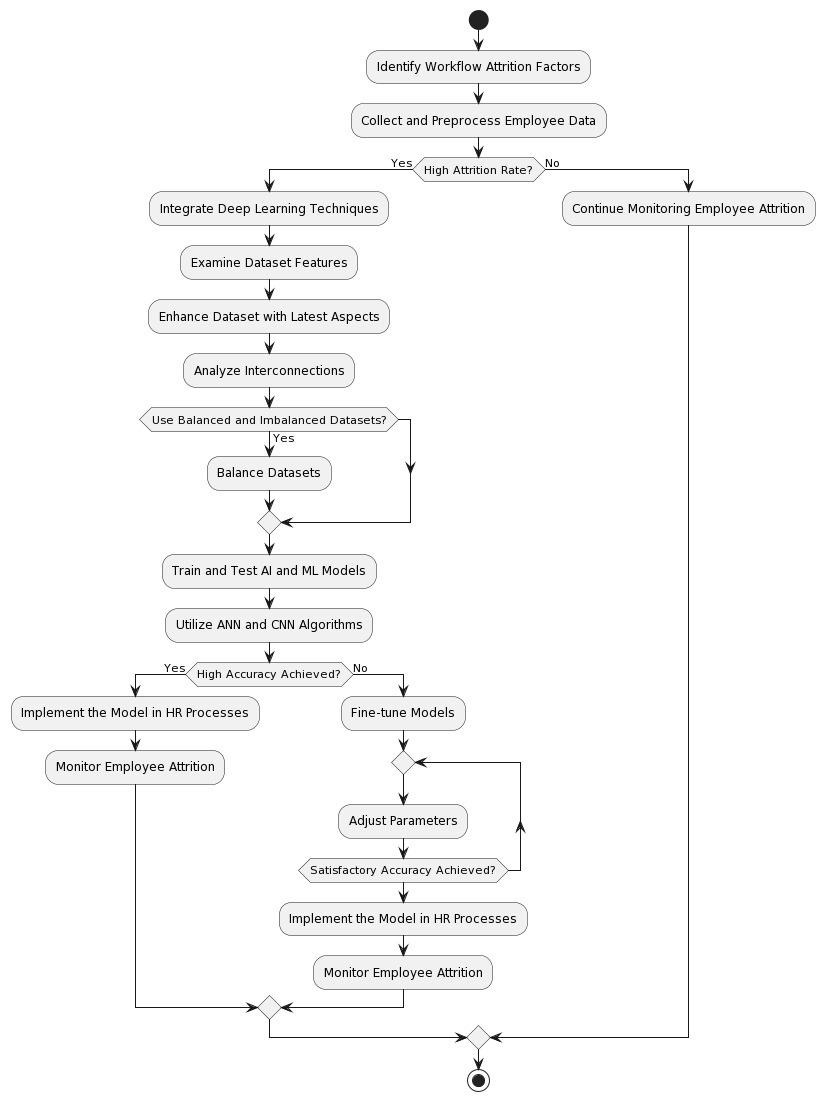
The AI/ML System initiates and performs the required activities. The graphic does not represent a specific response, but this stage might involve the AI/ML System sending updates or confirmation to the HR Manager.

**Manage Attrition Issues:** The HR Manager sends a request to the AI/ML System to handle attrition issues. The AI/ML System activates and offers the HR Manager with recommendations and insights. The AI/ML System then sends the results and insights back to the HR Manager.

**Model Comparison:** The Researcher activates and launches a request to the AI/ML System in order to compare several machine and deep learning models. The AI/ML System activates and does model comparison. The AI/ML System returns results and insights to the researcher.

**Insights and Results:** The AI/ML System sends results and insights to both the HR Manager and the Researcher throughout the sequence. These outcomes could include data analysis findings, updates to technology deployment, management advice, and model comparison details.

In summary, the HR Manager initiates requests for understanding attrition factors, using technology, and addressing attrition concerns.

**3.2.3 Activity Diagram**

**Fig 3: Activity Diagram for the Attrition detection and prevention process**

**Recognize Attrition Factors:** The procedure begins with the HR Manager initiating the action of understanding attrition factors. This entails the AI/ML System assessing data and trends.

**Utilize Technology**: The HR Manager initiates the use of technology for human resource management. The AI/ML System is in charge of implementing AI/ML solutions.

**Manage Attrition Issues**: The action of handling attrition issues is initiated by the HR Manager.

The AI/ML System addresses attrition by making recommendations and providing insights.

**Model Comparison:** The activity of comparing machine and deep learning models is initiated by the Researcher. The AI/ML System performs model comparison and offers the Researcher with results and insights. Round rectangles represent the activities, while arrows illustrate the flow of activities between them. The "(\*)" sign denotes the beginning and finish of the process.

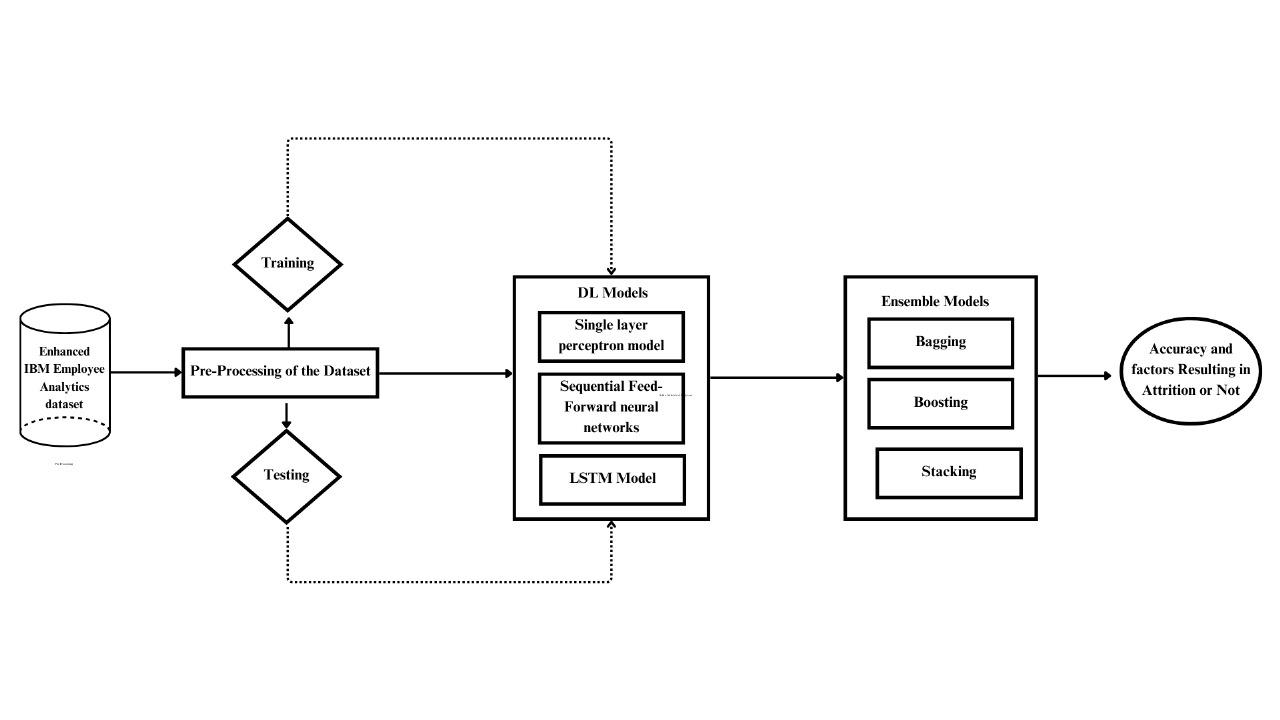
**3.2.4 System Architecture**

Dataset: Begin with a dataset that is appropriate for your situation. Ascertain that it has been thoroughly preprocessed, is clean, and has been divided into training and testing sets.

Tasks: Data exploration entails learning about the structure and features of your dataset.

Data cleaning entails dealing with missing numbers, outliers, and other data quality issues.

Data splitting is the process of dividing a dataset into training and testing sets.



**Fig 4: System Architecture for the Attrition detection and prevention process**

**Preprocessing**: Prepare the data for training by doing the appropriate transformations and feature engineering.

Tasks: Normalize or normalize numerical features via feature scaling.

Handling Unbalanced Data: Address any class imbalances that may exist.

Sequence preparation (for LSTM): Consider bending sequential data for LSTM input.

Training Schemes: Individual models are trained on preprocessed training data.

**Models**: LSTM (Long Short-Term Memory): This type of memory is best suited for sequential data, such as time series or natural language processing.

Single Perceptron Model: A single-layer neural network.

Feed-forward in a Sequential Order A simple neural network with numerous layers is modeled.

Model architectures must be defined. Models should be built using appropriate loss functions and optimizers. Models should be trained using the training dataset.

Models of Ensembles: To improve overall performance, combine the predictions of many base models.

**Methods of Ensemble**: Bagging (Bootstrap Aggregating): Independently train many models and integrate their predictions (e.g., Random Forest).

Boosting: Sequentially train models, providing more weight to misclassified examples.

Stacking: Using a meta-model, combine predictions from numerous models.

Use libraries such as scikit-learn to implement ensemble approaches.

For each ensemble technique, train and optimize hyperparameters.

**Model Evaluation**: Using the testing dataset, evaluate the performance of individual models and ensembles. Metric accuracy must be calculated. Check for overfitting by evaluating the models on both training and testing sets. Individual model and collective performance should be compared.

**Analysis and Results**: Analyze the findings to learn about the strengths and shortcomings of each model and ensemble. Examine the accuracy results for each model and ensemble. Recognize trends in erroneous classifications. Consider the importance of features in ensemble models.

Finally, and most importantly, optimization: Draw insights from the analysis and propose additional optimization efforts.

Tasks: Improve model performance by optimizing hyperparameters. Investigate further preprocessing steps or feature engineering. Consider fine-tuning ensemble approaches for improved outcomes. Starting with data preparation, training individual models, exploring ensemble approaches, and eventually reviewing the outcomes, this procedure gives a methodical approach to constructing and evaluating models. Adjustments can be performed based on the dataset's and issue domain's specific requirements and characteristics.

**3.3 Functional Requirements**

SOFTWARE REQUIREMENTS

* Operating System: Windows 10/11, Linux, IOS
* Programming Language: Python 3.7-3.11
* Library: Tensorflow2

HARDWARE REQUIREMENTS

* Processor: Intel i3 or higher
* RAM: 4GB RAM and higher

The project's software and hardware requirements are meticulously outlined to ensure optimal performance and compatibility. The system supports a variety of operating systems, including Windows 10/11, Linux, and iOS, providing customers with platform freedom. Python 3.7-3.11 is used to implement the project, which is a versatile and widely used programming language noted for its readability and vast libraries. The Tensorflow2 library is used specifically, emphasizing the importance of deep learning capabilities for the project. On the hardware front, a modest setup with an Intel i3 processor or above and a minimum of 4GB RAM is suggested. These hardware criteria create a balance between performance and usability, allowing the project to be accessible to a wide variety of users using conventional computing settings. Overall, the software and hardware requirements specified provide a firm foundation for the project's development and management.

**CHAPTER 4**

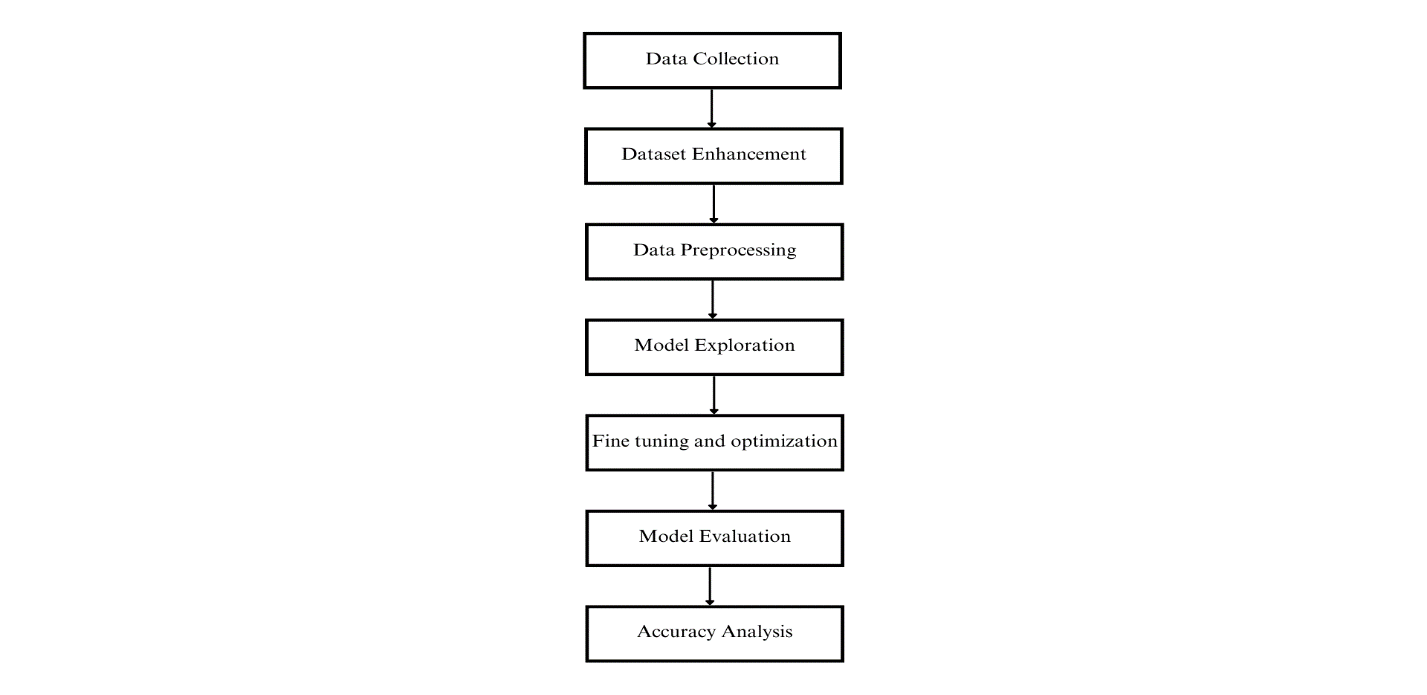
**IMPLEMENTATION**

**4.1 Module Description**

1. **Data Collection**: The initial Dataset ‘IBM HR Analytics Employee attrition and performance’ is taken from Kaggle.
2. **Dataset Enhancement**: The dataset initially has 35 features from 1,470 workers. We have enhanced the dataset by adding 4 more categorical features which are apt features for the present working conditions. We used pandas DataFrame methods to add and enhance the data.
3. **Data Preprocessing**: Pre-processing is an important step in machine and deep learning that significantly improves model performance. Pre-processing includes normalizing, cleaning, and categorizing data encoding. The dataset has been cleaned and standardized by using scalars. Some of the attributes found in this data set are categorical rather than numerical. Most ML and DL algorithms do not support categorical features right away. The original data includes multiple type categorical variables such as (Business Travel, Departments, Education Field, Gender, Job Role, Marital Status, and Over Time).This type of feature should be associated with numerical numbers. To convert them, "one hot encoding" and “Label Encoding” came into play. Missing values are dealt with ML imputation. The most crucial dataset attributes are found for Attrition analysis.
4. **Model Exploration**: Various Deep learning and Neural Network Models that fit to our criteria and algorithms are explored and put to test. We used 4 models:
   * Long-Short Term Memory Algorithm (LSTM)
   * Sequential Feed-Forward Neural Network
   * Single Layer Perceptron Model
   * Ensemble Methods (Bagging, Boosting and Stacking)
5. **Fine tuning and Optimization:** The process and algorithms are kept on trained and tested to improve the accuracy and other factors for each model. We performed Multiple Iterations of every algorithm until we reached the highest accuracy and optimized the code like removing data overfitting etc.
6. **Model Evaluation:** After multiple optimization rounds, The best model is found by evaluating the factors.

All the 4 models are evaluated by testing for test loss and accuracy and also cross-validation. Accuracy is seen in percentages

1. **Accuracy Analysis:** Accuracies of models are analyzed and a conclusion is made. The model that achieves the highest accuracy is concluded as the best model. Also by comparison we see how many models have the similar level of accuracy metric to them.



**Fig 5: Modules**

**4.2 Model components**

**4.2.1 Dataset**

The dataset used is the ‘IBM HR Analytics Employee attrition and performance’ retrieved from Kaggle. The number records in it are up to 1450.

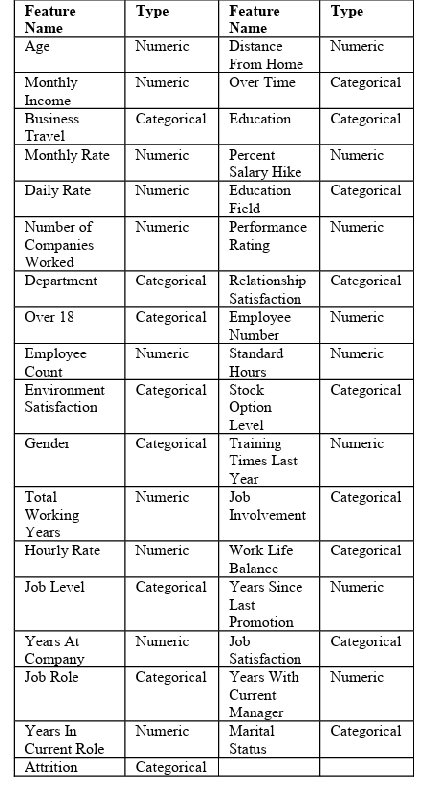
Our implied ML and DL models analyze the dataset to identify the most pertinent data points that improve accuracy and develop a projecting model in accordance with the following steps:

**Dataset Enhancement:** The dataset initially has35 features from 1,470 workers. We have enhanced the dataset by adding 4 more categorical features which are apt features for the present working conditions. We used pandas Data Frame methods to add and enhance the data.

**Data Preprocessing:** Pre-processing is an important step in machine and deep learning that significantly improves model performance. Pre-processing includes normalizing, cleaning, and categorizing data encoding.

**Preparing the collected Dataset:** The dataset has been cleaned and standardized by using scalars. Some of the attributes found in this data set are categorical rather than numerical. Most ML and DL algorithms do not support categorical features right away. The original data includes multiple type categorical variables such as (Business Travel, Departments, Education Field, Gender, Job Role, Marital Status, and Over Time). This type of feature should be associated with numerical numbers. To convert them, "one hot encoding" and “Label Encoding” came into play. missing values are dealt with ML imputation. The most crucial dataset attributes are found for Attrition analysis. Dividing the dataset: Divide the dataset into train, validate, and test. Train and test the proposed models by using train and validate data. Using the test data, test the proposed models.

|  |  |
| --- | --- |
| **Features** | **Type** |
| Work mode | Categorical |
| Toxic environment | Categorical |
| Appreciation | Categorical |
| Hiring process | Categorical |

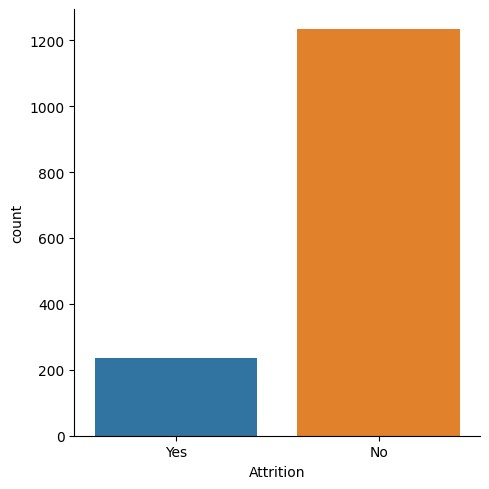
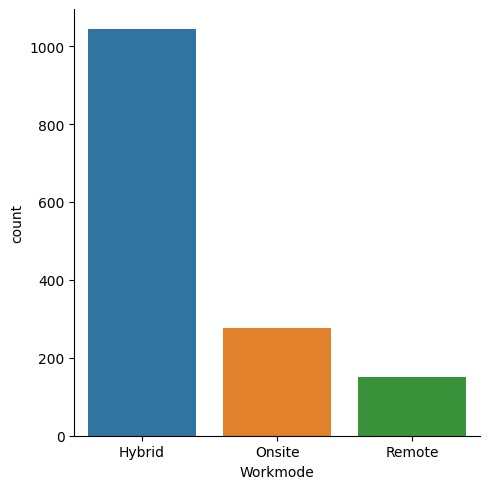
******

**Table 2: Table of already existing features and their types**

**Table 1: Added features and their types**

**4.2.2 Data Analysis and Visualization**

The data in the dataset, is analyzed before putting into the models by using the EDA method. EDA is the process of analyzing and visualizing data to obtain insights, uncover patterns, and comprehend the features of a dataset. To prepare the dataset for modeling, it involves tasks such as data visualization, data summary statistics, data cleansing, and data transformation. EDA assists data scientists and analysts in understanding the structure, distribution, and correlations between variables in their data. Python libraries like Pandas for data manipulation, Matplotlib and Seaborn for data visualization, and statistical approaches like mean, median, and standard deviation calculations are common EDA tools. From this process we could show the relation of Attrition to many factors in the dataset and also could show whether there is Attrition or not.



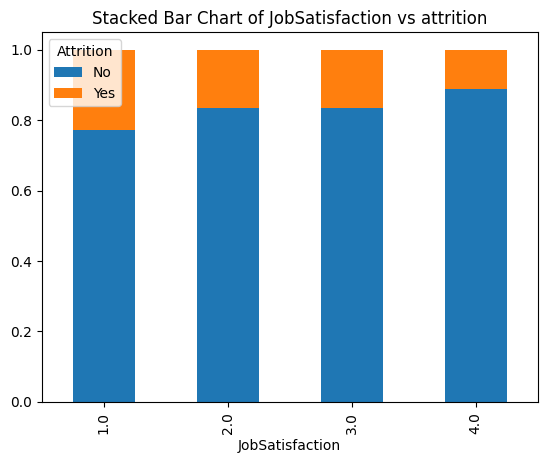
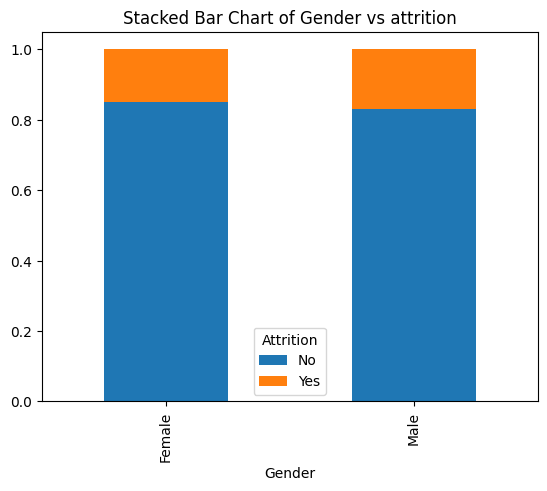


**Fig 6:** **Comparison of Attrition with No Attrition from the dataset**

**Fig 7:** **Comparison of number of employees and their work mode**

Typically, categorical data would be explored in EDA by showing the distribution of categories, computing summary statistics for each category, and investigating correlations between categorical variables and the target variable. We have plotted graphs for various categorical data.

By further analysis and more visualizations, we could draw the distribution of the features in the dataset, plotted the stacked bar plots to show how every feature and its distribution has an effect on Attrition.

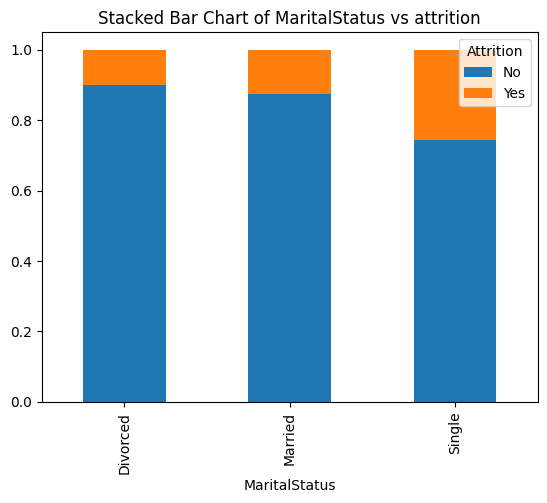
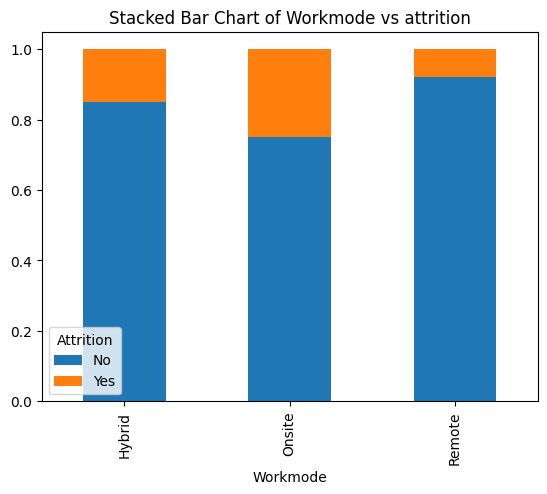


***Graph-4: Graph showing how Work mode effects Attrition***

**Fig 8****: Graph showing how marital status effects Attrition**

**Fig 9:** **Graph showing how Work mode effects Attrition**

Above are the graphs where, Graph-3 shows the relation of marital status and how its distribution has Attrition difference. We can see that single Employees are more likely to leave the company. Graph-4 shows that employees who are working on-site are more presumed to be resigning



**Fig 11:** **Graph showing how gender effects Attrition**

**Fig 10:** **Graph showing how Job satisfaction effects Attrition**

Similarly, Graph-5 captures the relation of Job satisfaction with Attrition and clearly employees with less satisfaction are more interested to find another company and leave the present company. Graph-6 shows that Gender is not a big factor as the results of Attrition vs No attrition are neutral and equal.

**4.2.3 Web Interface**

Our revolutionary web interface allows for a thorough examination of the factors impacting attrition rates. It incorporates input boxes matching to each dataset attribute, allowing users to enter values and comprehend the subtle implications on attrition. The interface incorporates HTML and CSS for a visually appealing front-end, while Python Flask runs the back-end, assuring robust functioning. Users can easily enter values, submit data, and receive predictions that determine whether attrition is likely (Yes or No) and the accuracy of these forecasts using powerful Deep Learning Algorithms. This user-centric platform enables HR professionals and management to make informed decisions while executing focused measures to mitigate the negative impact of attrition. The interface's dynamic nature allows for real-time insight of workforce dynamics, supporting a proactive approach to personnel management.

**4.3 Sample Code**

**4.3.1 Long-Short Term Memory**

**import pandas as pd**

**import numpy as np**

**import tensorflow as tf**

**from sklearn.preprocessing import LabelEncoder**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.preprocessing import StandardScaler**

**from sklearn import metrics**

**from sklearn.metrics import precision\_score, mean\_squared\_error, recall\_score, f1\_score**

**from keras.models import Sequential**

**from keras.layers import LSTM, Dense, Dropout**

**from keras.optimizers import Adam**

**import pickle**

**# Load dataset**

**data = pd.read\_csv('attrition.csv')**

**categorical\_column = ['Attrition', 'BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole', 'MaritalStatus', 'Over18', 'OverTime', 'Workmode', 'Appreciation', 'Toxic culture', 'Bad Hiring process']**

**encoder=LabelEncoder()**

**data[categorical\_column]=data[categorical\_column].apply(encoder.fit\_transform)**

**# Handling Missing Values**

**data.dropna(inplace=True)**

**X=data[['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','Workmode','Appreciation','Toxic culture','Bad Hiring process']]**

**y = data['Attrition']**

**# Split the dataset into training and testing sets**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)**

**# Standardize**

**scaler = StandardScaler()**

**X\_train = scaler.fit\_transform(X\_train)**

**X\_test = scaler.transform(X\_test)**

**#LSTM model**

**model = Sequential()**

**model.add(LSTM(64, input\_shape=(X\_train.shape[1], 1), return\_sequences=True))**

**model.add(LSTM(64, return\_sequences=False))**

**model.add(Dropout(0.2))**

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

**model.compile(optimizer=Adam(learning\_rate=0.002),loss='binary\_crossentropy', metrics=['accuracy'])**

**# Reshape data**

**X\_train = X\_train.reshape(X\_train.shape[0], X\_train.shape[1], 1)**

**X\_test = X\_test.reshape(X\_test.shape[0], X\_test.shape[1], 1)**

**# Train the model**

**model.fit(X\_train, y\_train, epochs=50, batch\_size=64, validation\_data=(X\_test, y\_test))**

**print(model.predict('age'))**

**4.3.2 Single perceptron Layer Model**

**import numpy as np**

**import pandas as pd**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.preprocessing import LabelEncoder**

**from sklearn.preprocessing import StandardScaler, MinMaxScaler**

**from sklearn.impute import SimpleImputer**

**from sklearn.linear\_model import Perceptron**

**from sklearn.metrics import accuracy\_score**

**data = pd.read\_csv('test.csv')**

**categorical\_column = ['Attrition', 'BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole', 'MaritalStatus', 'Over18', 'OverTime', 'Workmode', 'Appreciation', 'Toxic culture', 'Bad Hiring process']**

**encoder=LabelEncoder()**

**data[categorical\_column]=data[categorical\_column].apply(encoder.fit\_transform)**

**# Handling Missing Values**

**data.dropna(inplace=True)**

**X=data[['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','Workmode','Appreciation','Toxic culture','Bad Hiring process']]**

**y = data['Attrition']**

**imputer = SimpleImputer(strategy='mean')**

**X = imputer.fit\_transform(X)**

**# Split the dataset into training and testing sets**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=45)**

**# Create and train the Perceptron model**

**model = Perceptron(random\_state=42)**

**model.fit(X\_train, y\_train)**

**scaler = MinMaxScaler()**

**X\_train\_scaled = scaler.fit\_transform(X\_train)**

**X\_test\_scaled = scaler.transform(X\_test)**

**# Create and train the Perceptron model**

**model = Perceptron(max\_iter=2000, random\_state=42)**

**model.fit(X\_train, y\_train)**

**# Make predictions**

**y\_pred = model.predict(X\_test)**

**# Evaluate the model**

**accuracy = accuracy\_score(y\_test, y\_pred)**

**print(f'Accuracy: {accuracy:.2f}')**

**acc=int(round(accuracy,2)\*100)**

**print("The percentage of times this model shows the Attrition rate correctly is:")**

**print(acc,'%')**

**4.3.3 Sequential Feed-Forward Neural Networks**

**import numpy as np**

**import pandas as pd**

**from sklearn.model\_selection import train\_test\_split,cross\_val\_score, cross\_validate**

**from sklearn.preprocessing import StandardScaler**

**from sklearn.preprocessing import LabelEncoder**

**from sklearn.model\_selection import GridSearchCV**

**from sklearn.neural\_network import MLPClassifier**

**from keras.models import Sequential**

**from keras.layers import Dense**

**import time**

**random\_seed = 316**

**pd.set\_option('display.max\_columns', None)**

**data = pd.read\_csv('test.csv')**

**data["Attrition"]=data["Attrition"].map({"Yes":1 ,"No": 0})**

**categorical\_column = ['Attrition', 'BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole', 'MaritalStatus', 'Over18', 'OverTime', 'Workmode', 'Appreciation', 'Toxic culture', 'Bad Hiring process']**

**encoder=LabelEncoder()**

**data[categorical\_column]=data[categorical\_column].apply(encoder.fit\_transform)**

**X=data[['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','Workmode','Appreciation','Toxic culture','Bad Hiring process']]**

**y = data['Attrition']**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)**

**scaler = StandardScaler()**

**X\_train\_scaled = scaler.fit\_transform(X\_train)**

**X\_test\_scaled = scaler.transform(X\_test)**

**# Build the neural network model**

**model = Sequential()**

**model.add(Dense(units=69, activation='relu', input\_dim=X\_train\_scaled.shape[1]))**

**model.add(Dense(units=32, activation='relu'))**

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

**model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])**

**model.fit(X\_train\_scaled, y\_train, epochs=75, batch\_size=65, validation\_split=0.3)**

**# Evaluate the model**

**loss, accuracy = model.evaluate(X\_test\_scaled, y\_test)**

**print("Test Loss:", loss)**

**print("Test Accuracy:", accuracy)**

**acc = int(round(accuracy, 2) \* 100)**

**print("The percentage of times this model shows the Attrition rate correctly is:")**

**print(acc, '%')**

**4.3.4 Ensemble Methods (Bagging, Boosting, Stacking)**

**import numpy as np**

**import pandas as pd**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.preprocessing import LabelEncoder**

**from sklearn.impute import SimpleImputer**

**from sklearn.ensemble import BaggingClassifier, AdaBoostClassifier, RandomForestClassifier, StackingClassifier**

**from sklearn.tree import DecisionTreeClassifier**

**from sklearn.linear\_model import LogisticRegression**

**from sklearn.metrics import accuracy\_score**

**df = pd.read\_csv('attrition.csv')**

**#Encode**

**categorical\_column = ['Attrition', 'BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole', 'MaritalStatus', 'Over18', 'OverTime', 'Workmode', 'Appreciation', 'Toxic culture', 'Bad Hiring process']**

**encoder=LabelEncoder()**

**df[categorical\_column]=df[categorical\_column].apply(encoder.fit\_transform)**

**X=df[['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','Workmode', 'Appreciation', 'Toxic culture', 'Bad Hiring process']]**

**y = df['Attrition']**

**df.dropna(inplace=True)**

**#Imputation**

**imputer = SimpleImputer(strategy='mean')**

**X = imputer.fit\_transform(X)**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)**

**# Define base classifiers**

**base\_classifier1 = DecisionTreeClassifier(random\_state=45)**

**base\_classifier2 = RandomForestClassifier(n\_estimators=30, random\_state=45)**

**base\_classifier3 = LogisticRegression(solver='lbfgs', max\_iter=1000)**

**# Stacking Classifier**

**estimators = [('dt', base\_classifier1), ('rf', base\_classifier2), ('lr', base\_classifier3)]**

**stacking\_classifier = StackingClassifier(estimators=estimators, final\_estimator=LogisticRegression())**

**stacking\_classifier.fit(X\_train, y\_train)**

**stacking\_predictions = stacking\_classifier.predict(X\_test)**

**stacking\_accuracy = accuracy\_score(y\_test, stacking\_predictions)**

**print(f'Stacking Classifier Accuracy: {stacking\_accuracy:.2f}')**

**acc=int(round(stacking\_accuracy,2)\*100)**

**print("The percentage of times this model shows the Attrition rate correctly is:")**

**print(acc,'%')**

**# Bagging Classifier**

**bagging\_classifier = BaggingClassifier(base\_classifier1, n\_estimators=10, random\_state=42)**

**bagging\_classifier.fit(X\_train, y\_train)**

**bagging\_predictions = bagging\_classifier.predict(X\_test)**

**bagging\_accuracy = accuracy\_score(y\_test, bagging\_predictions)**

**print(f'Bagging Classifier Accuracy: {bagging\_accuracy:.2f}')**

**acc=int(round(bagging\_accuracy,2)\*100)**

**print("The percentage of times this model shows the Attrition rate correctly is:")**

**print(acc,'%')**

**# AdaBoost Classifier**

**adaboost\_classifier=AdaBoostClassifier(base\_classifier2,n\_estimators=20, random\_state=42)**

**adaboost\_classifier.fit(X\_train, y\_train)**

**adaboost\_predictions = adaboost\_classifier.predict(X\_test)**

**adaboost\_accuracy = accuracy\_score(y\_test, adaboost\_predictions)**

**print (f'AdaBoost Classifier Accuracy: {adaboost\_accuracy:.2f}')**

**acc=int(round(adaboost\_accuracy,2)\*100)**

**print ("The percentage of times this model shows the Attrition rate correctly is:")**

**print(acc,'%')**

**4.3.5 Web Development (Back-End)**

**from flask import Flask, render\_template, request**

**from model import predict\_data**

**app=Flask(\_\_name\_\_)**

**@app.route("/", methods=["GET", "POST"])**

**def index():**

**if request.method=="POST":**

**form\_data = request.form.to\_dict()**

**form\_data['Over18'] = 'Y'**

**for key in form\_data:**

**form\_data[key] = [form\_data[key]]**

**predictions, accuracy = predict\_data(form\_data)**

**return render\_template('result.html', attrition=predictions[0], accuracy=accuracy)**

**return render\_template('index.html')**

**if \_\_name\_\_ == "\_\_main\_\_":**

**app.run(debug=True, host='localhost', port=3000)**

**CHAPTER 5**

**TESTING**

**5.1 Importance of Testing**

The deep learning attrition model and its accompanying web interface must be thoroughly tested to ensure the system's overall dependability and effectiveness. The validation of input data in model testing ensures the algorithm's resistance against diverse scenarios, including extreme values and outliers. The model's adaptability to different scenarios and datasets is ensured by evaluating data kinds, scalability, and cross-validation. Accuracy metrics and robustness testing are used to determine the model's prediction capabilities as well as its capacity to generalize beyond the training data.

In the case of web interfaces, thorough testing is critical to user happiness. Input validation and usability checks ensure a consistent user experience, while performance testing evaluates the system's responsiveness and efficiency. Security procedures are critical for assuring sensitive data security and user authentication.

The deep learning attrition model and its accompanying web interface must be thoroughly tested to ensure the system's overall dependability and effectiveness. The validation of input data in model testing ensures the algorithm's resistance against diverse scenarios, including extreme values and outliers. The model's adaptability to different scenarios and datasets is ensured by evaluating data kinds, scalability, and cross-validation. Accuracy metrics and robustness testing are used to determine the model's prediction capabilities as well as its capacity to generalize beyond the training data.

Comprehensive error handling and compatibility testing across several browsers and devices handle potential user difficulties. End-to-end testing and deployment assessments check the entire workflow, confirming the model's integration into the web interface. In essence, this rigorous testing procedure not only finds and corrects any flaws while also instilling confidence in the attrition prediction system's dependability, security, and user-friendliness in real-world circumstances.

**5.2 Types of Testing**

Testing our deep learning attrition model and web interface entails assessing both the model's performance and the user interface's functionality. Some test cases selected for extensive testing are as follows:

**5.2.1 Model Testing**

**1. Validation of Input:**

The goal is to guarantee that the model handles multiple input scenarios correctly. Testing the model's resilience in real-world settings by validating it with usual input values, extreme values, and outliers, as well as verifying its ability to handle missing or null data.

**2. Data Types:**

The goal of this test is to ensure that the model is compatible with various data formats.

Testing the model with several data types for input features, such as integers, floats, and categorical variables, guarantees the model's flexibility and ability to handle different data representations.

**3. Scalability:**

The goal is to evaluate the model's performance on large datasets.

Testing the model's scalability with a large dataset assures that it will be efficient and responsive when dealing with enormous amounts of data.

**4. Model Precision:**

The goal is to validate the predicted performance of the model.

Using a separate dataset for evaluation, as well as metrics such as precision, recall, F1-score, and confusion matrix, provides a thorough insight of the model's accuracy and ability to handle multiple classes well.

**5. Model Robustness:**

The goal of this experiment is to see how well the model handles noise and disturbances.

Including noise in the input data helps evaluate the model's robustness and capacity to make

credible predictions in the presence of unpredictability.

**6. Cross-validation:**

The goal is to assess the model's generalization across different data splits.

Cross-validation ensures that the model's performance is constant across different subsets of data, lowering the danger of overfitting.

**5.2.2 Web Interface Testing**

**1. Input Validation:**

The goal is to ensure that the web interface accepts various user inputs correctly.

Testing with valid and invalid input values, as well as ensuring that the interface gives clear feedback for erroneous inputs, improves the user experience and prevents errors.

**2. Usability:**

The goal of this test is to determine how user-friendly the web interface is.

Testing for usability and responsiveness across several devices and browsers ensures that users have a positive and consistent experience.

**3. Integration Testing:**

The goal of this test is to ensure that the web interface and the Flask backend communicate seamlessly.

Ensuring excellent communication and response handling between the frontend and backend components ensures a dependable end-to-end system.

**4. Efficiency:**

The goal of this test is to determine how responsive the web interface is.

Testing reaction time and identifying potential latency issues aids in the maintenance of a smooth and efficient user experience, particularly during interactions with the model.

**5. Edge test cases:**

The goal of this test is to see how the interface handles uncommon scenarios.

Testing with edge cases, such as empty inputs, ensures that the interface performs

consistently and offers adequate feedback in unusual circumstances.

**6. Security:**

The goal is to validate the security measures that are in place.

It is critical to validate good security, particularly with regard to user authentication and sensitive data, in order to defend against potential vulnerabilities and illegal access.

**7. Handling Errors:**

The goal of this test is to see how the interface reacts to errors.

Testing error handling on both the front end and the back end ensures that users receive what they expect.

**8. Adaptability:**

The goal is to ensure interoperability across several environments.

Testing the online interface on a variety of operating systems and browsers guarantees broad compatibility, preventing usability difficulties for users with varying configurations.

**5.2.3 End-to-End Testing**

**1. Completion of Workflow:**

The goal of this test is to validate the whole user path from input to output.

End-to-end workflow testing guarantees that the user's interactions with the web interface and the following model predictions are in sync, resulting in a seamless user experience.

**2. Implementation:**

The goal is to detect problems in a production-like setting.

Testing the deployed system in a setting that simulates the real-world production environment aids in the discovery of any unexpected faults that may develop during actual operation.

**3. Stress Testing:**

The goal of this test is to assess system performance under large loads. The system's response to several users accessing the web interface at the same time is

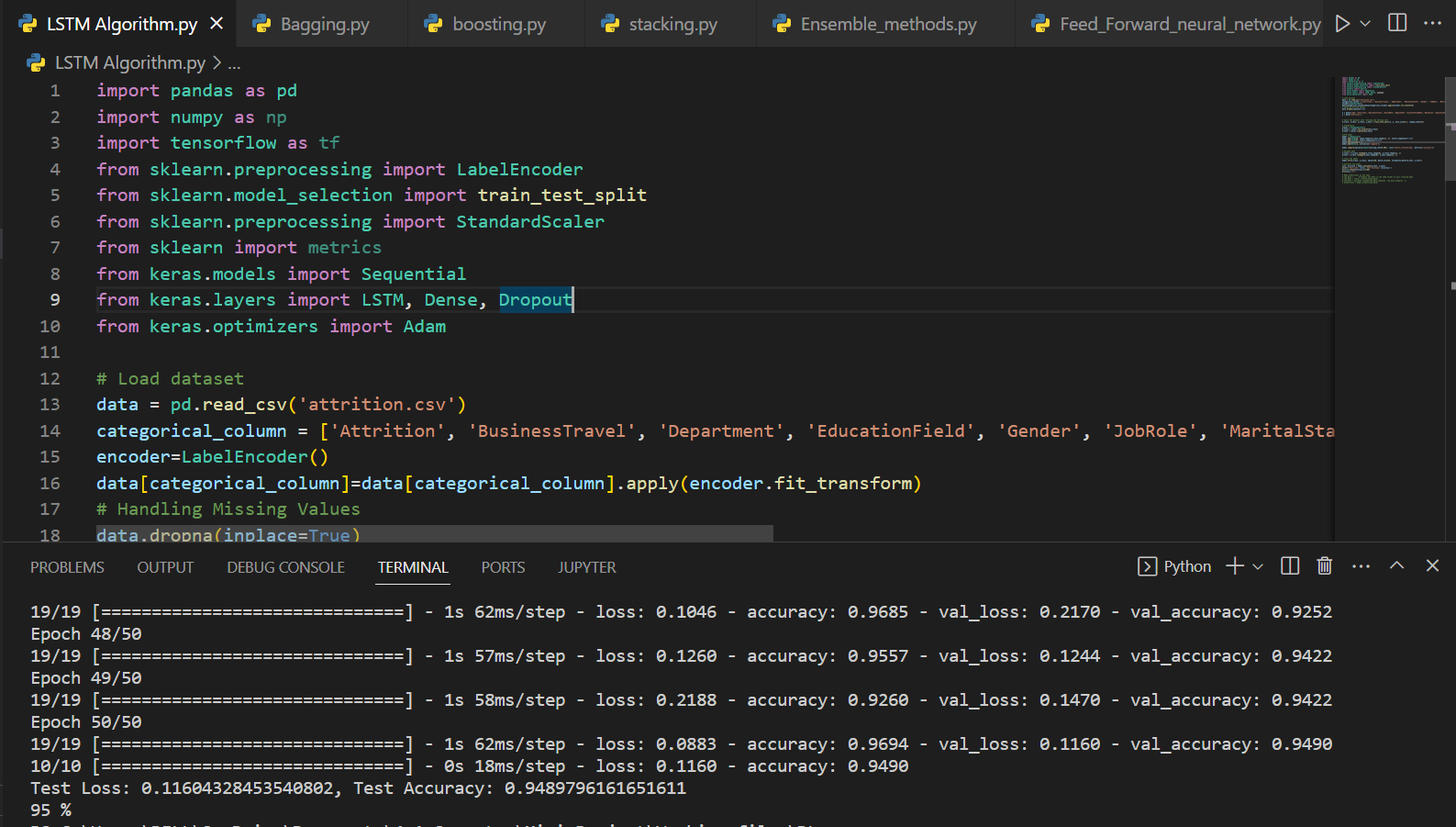
evaluated to ensure that it stays responsive and efficient even during peak usage.

Documenting these test cases and their conclusions is critical for the system's continued maintenance and enhancement.

Documenting these test cases and their outcomes is critical for the attrition prediction system's continued maintenance and improvement. Using automated testing technologies facilitates continuous integration and allows for efficient and effective regression testing. This extensive testing approach ensures the attrition prediction system's dependability, accuracy, and user satisfaction.

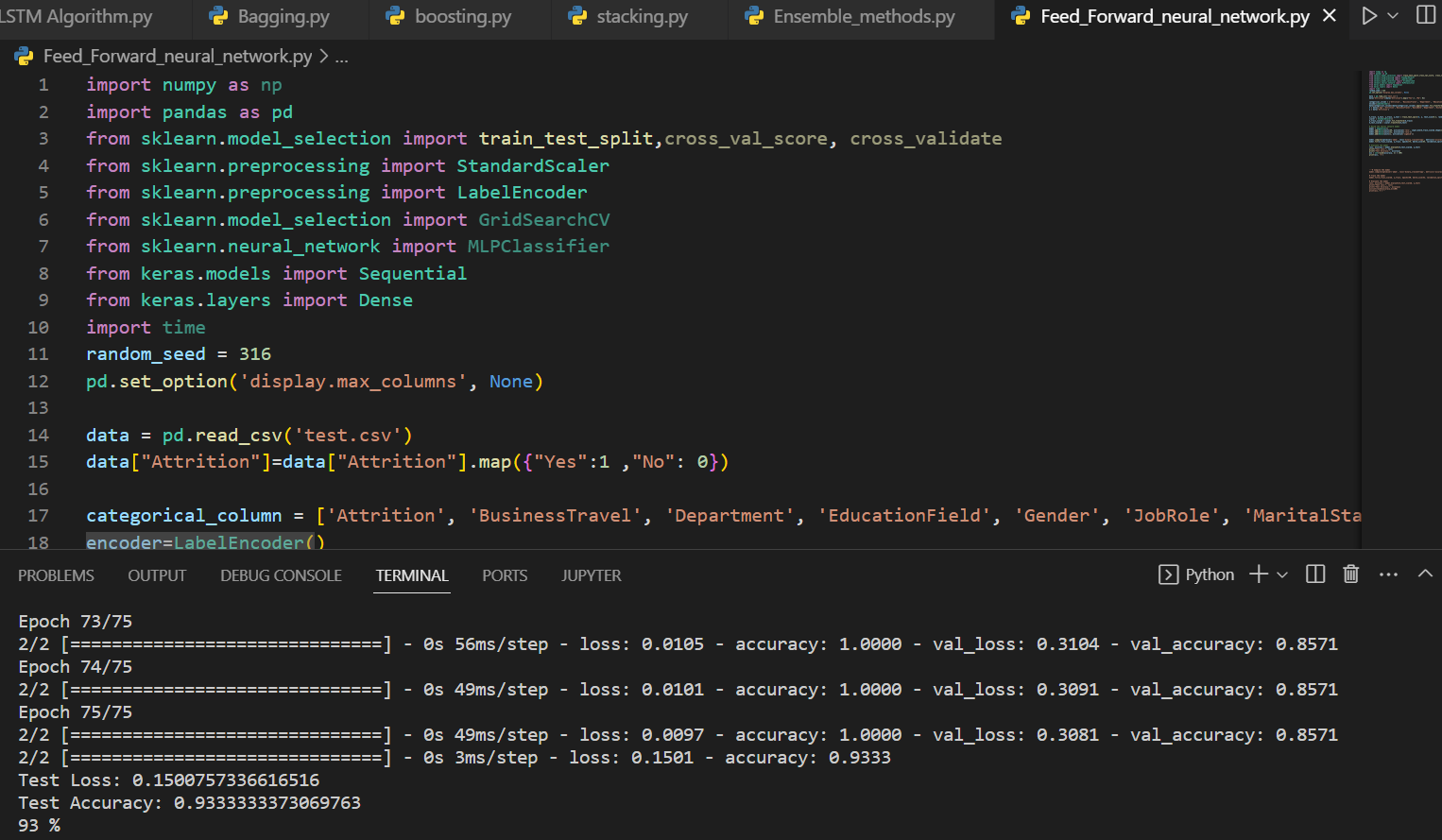
**CHAPTER 6**

**RESULTS**

After optimization, fine-tuning, and multiple iterations, we can finally observe the results provided by the algorithms. The goal is to find a model that can detect whether attrition is present or not, and the result is the percentage by which the proposed models correctly predict the presence of attrition. Here by the EDA methods, we found that the workflow Attrition is present by 16% and No Attrition is present by whopping 84%. The Attrition rates are hence analyzed by the models. We now see the models and their accuracies of if they can show this accurately:

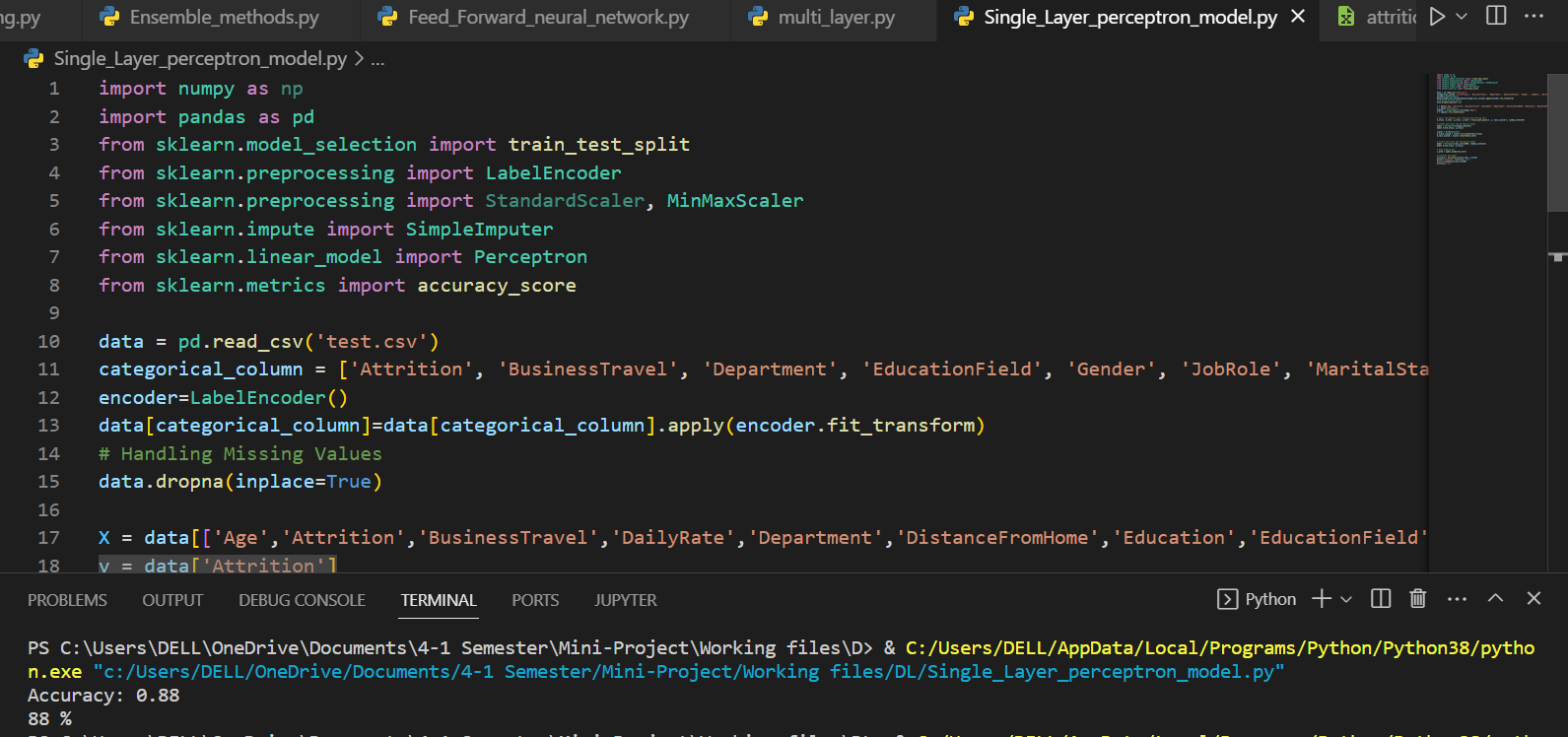
**Fig 12****: LSTM Algorithm Output**

Achieving a **95%** accuracy using LSTM Algorithm signifies a highly effective model for identifying potential turnover, facilitating proactive HR management.

****

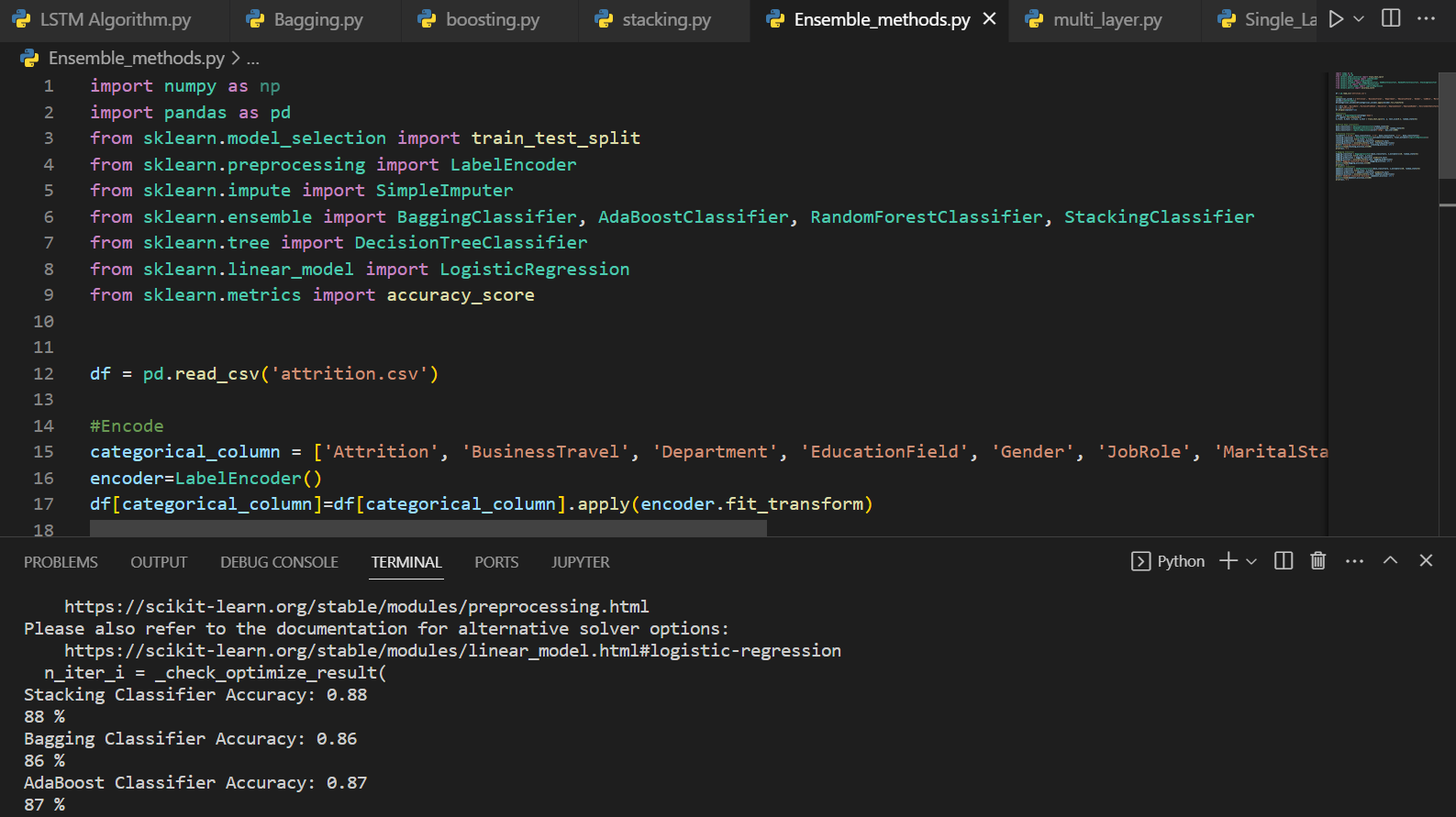
**Fig 13: Sequential Feedforward Neural Network Output**

Attaining a **93%** accuracy using a Sequential Feedforward Neural Network Algorithm demonstrates a robust model for identifying turnover risks, aiding proactive HR strategies



**Fig 14:** **Single Layer Perceptron Model Output**

Obtaining an **88%** accuracy using Single-Layer Perceptron Model indicates moderate success in identifying turnover trends but may benefit from more complex models for improved accuracy.



**Fig 15:** **Ensemble Methods Output**

Reaching an **87%** accuracy using Ensemble Model suggests a good predictive performance, leveraging multiple models to effectively identify turnover risks.

**Table 3: Table showing the comparison multiple models and their accuracies**

|  |  |
| --- | --- |
| **Deep Learning Model** | **Best Accuracy** |
| Long-Short Term Memory Algorithm | 95% |
| Sequential Feed-Forward Neural Network | 93% |
| Single-Layer perceptron model | 89% |
| Bagging | 88% |
| Boosting | 87% |
| Stacking | 87% |

Based on methods such as bagging, boosting, stacking, sequential feed forward, LSTM, and single layer perceptron. The LSTM with the highest accuracy of 95% demonstrates that it is the best at identifying sequential patterns in data. Sequential feed forward works well as well, with a 93% accuracy rate, and is a viable option for activities that do not always require sequential modeling.

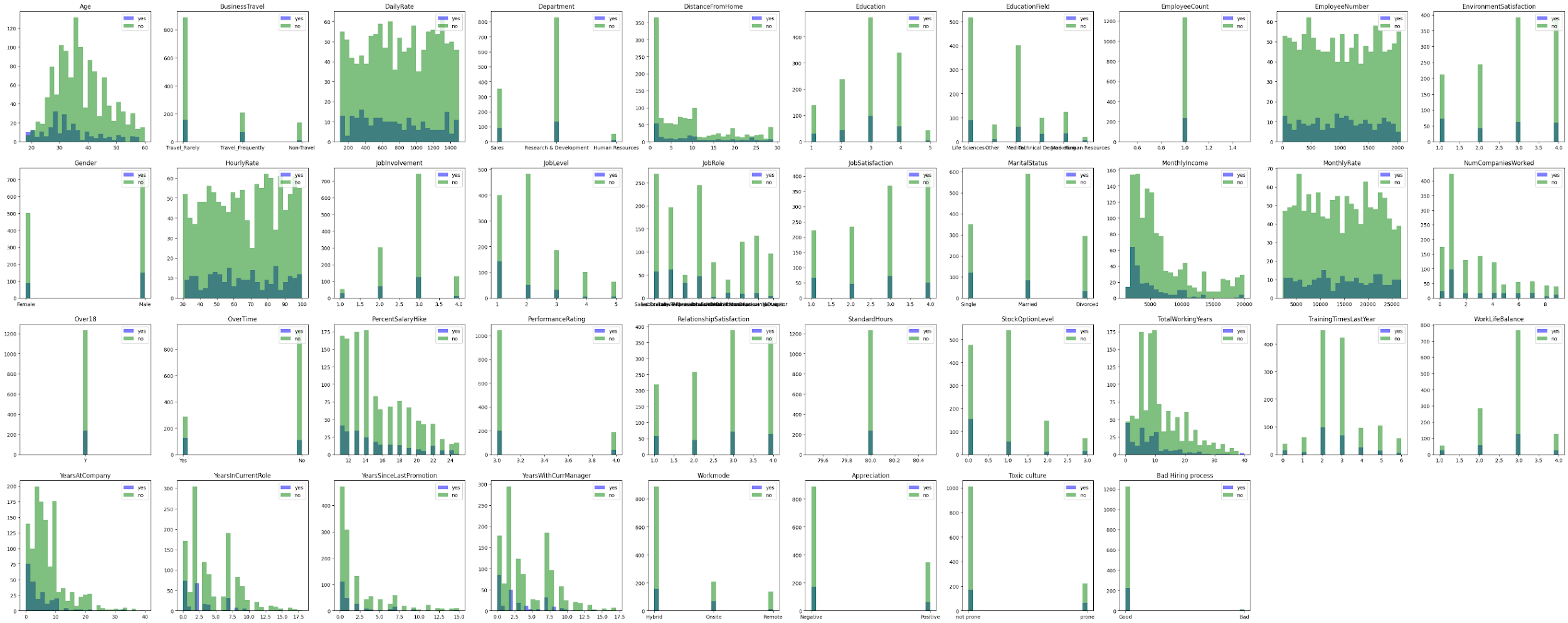
The Single Layer Perceptron model, with an accuracy of 89%, is a simpler model that may be useful for less demanding applications or when computational resources are limited. Both bagging and stacking appear to be equivalent in terms of accuracy, with 88%. Stacking combines the predictions of multiple ensemble methods, whereas

bagging is an ensemble technique that improves generality.

Boosting, with an accuracy rate of 87%, is slightly less accurate than the other algorithms. More optimization or a more complex ensemble technique would be advantageous.

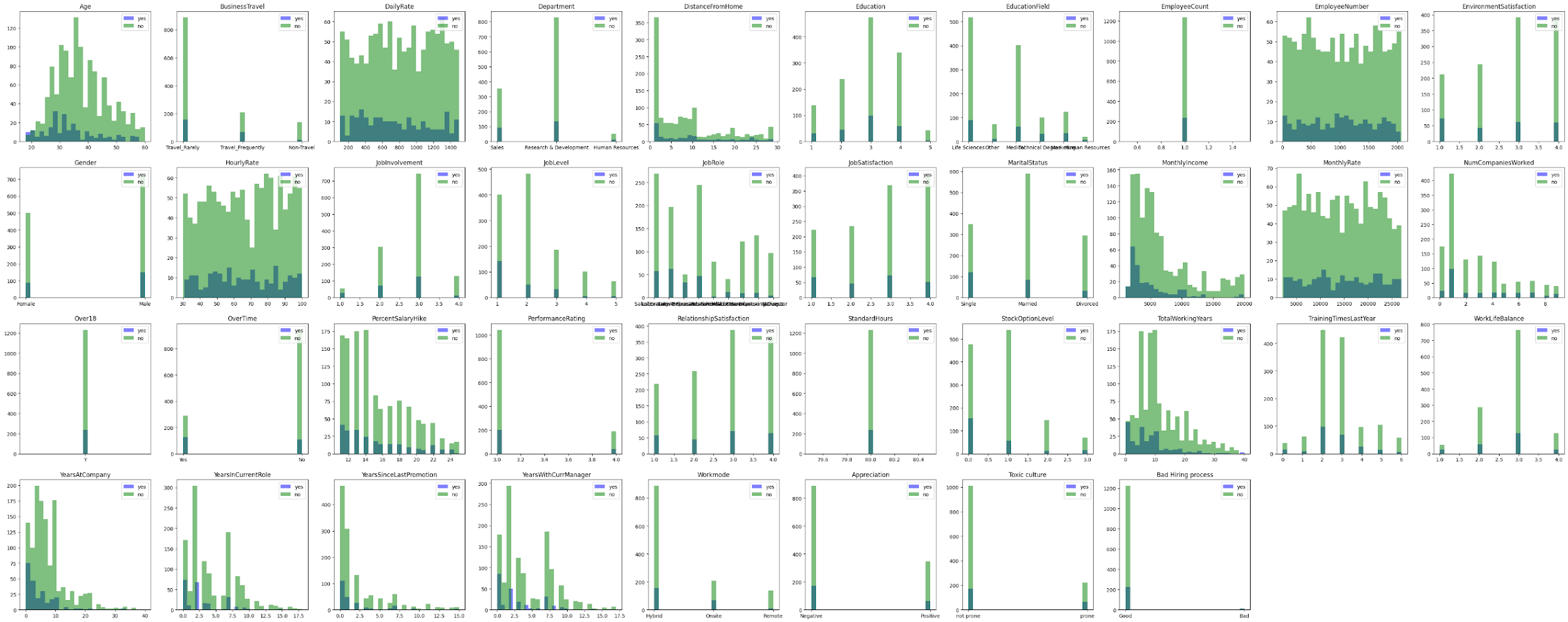
In the web interface when the inputs are given by the user, then the output is given as Yes or No for Attrition and the percentage of Accuracy with which the result is Correctly detected. The maximum Accuracy by which the Attrition is detected is up to 95%.

In the following graph images, we can see how every single factor, their distributions and

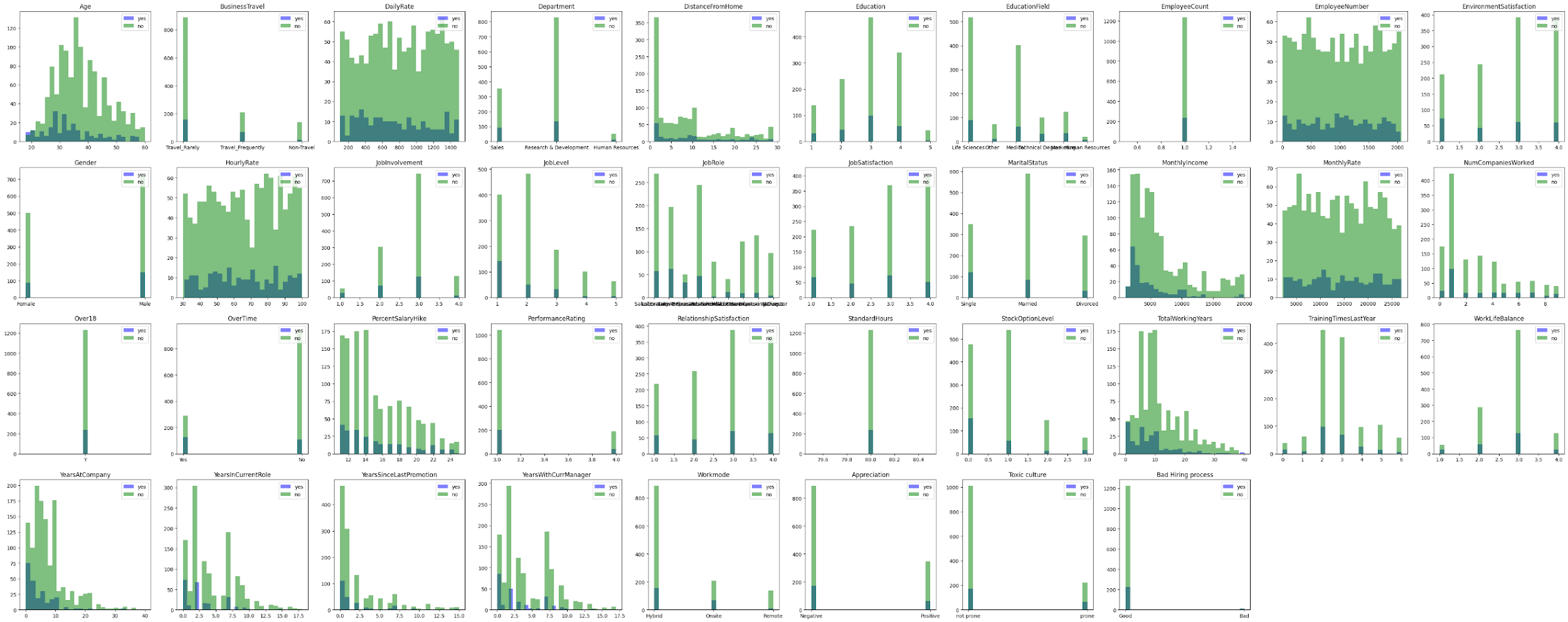
Attrition are Related. By this data we can determine Attrition or Not Attrition

**Fig-16****: 16 of the Features contributing to the attrition and the Attrition distribution within every feature.**

The above figure-10 shows the Attrition is not more likely on a level of huge excavation or mass resignation on few features. The most important features that have more negative effects than others are Over time, Percent salary hike and Years at company.



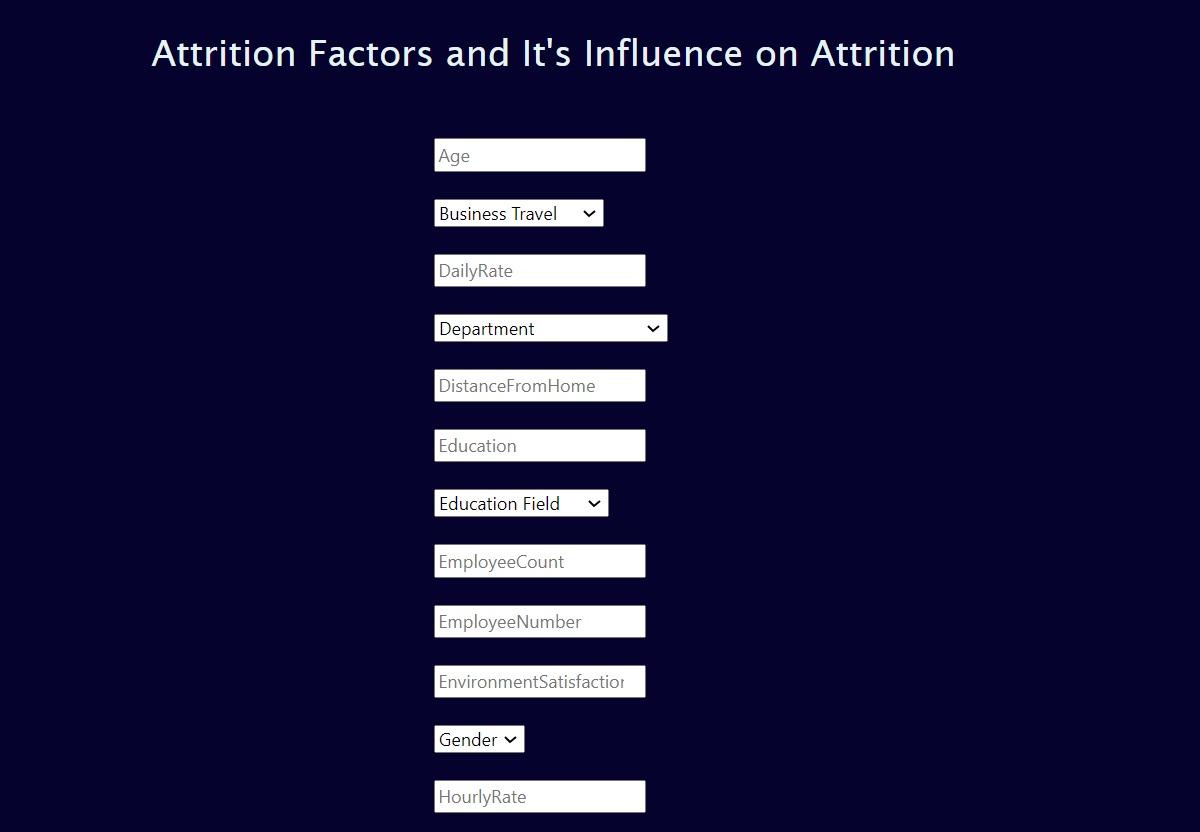
**Fig-17****: 6 more Features contributing to the attrition and the Attrition distribution within every feature.**



.

**Fig-18:** **16 of the important Features contributing to the attrition.**

Figure-18 Has the most important features that has a negative effect causing to Attrition namely Job role, Job Satisfaction, Monthly income, Toxic Culture, Education And stock option level. Following user input, the interface delivers outputs showing whether attrition is expected (Yes or No), as well as the accuracy % achieved by powerful Deep Learning Algorithms. This real-time feedback method provides meaningful data to HR professionals and management, allowing them to evaluate workforce opinion and make educated decisions. Impressively, The interface reaches a maximum detection accuracy of up to 95%, demonstrating its efficacy in predicting attrition.



**Fig 19:** **User Interface**

Our dynamic online interface has been painstakingly designed to include every individual element in the dataset, providing users with a complete exploration of their impact on attrition rates. This interface, which includes user-friendly input boxes, invites users to enter data in order to acquire a more nuanced knowledge of how various factors effect attrition. The interface offers a smooth user experience by combining HTML and CSS for an easy front-end and Python Flask for a powerful back-end.



**Fig 20:** **Output Showing Attrition With Accuracy**

We discovered that a small number of the 38 analyzed parameters have a negative link with attrition after thorough data analysis and visualization. The distribution patterns show that only a small number of employees inside the firm are prone to leaving. Notably, our investigation reveals a strong trend: employees with lower benefits across multiple criteria are more vulnerable to attrition than their counterparts with larger benefits in the same areas.

This thorough research provides a valuable lens through which HR and management may improve numerous elements and acquire a comprehensive grasp of the workforce's attitude toward the organization. After completing the exploratory data analysis (EDA) phase, we will concentrate on the development and experimentation of deep learning and ensemble models. These Based on feature and distribution patterns, models are developed to generate outputs indicating attrition or no attrition. Following that, we perform a full model comparison, ranking them based on their accuracy in identifying attrition rates and precision in discriminating 'Attrition' from 'Not Attrition'.

***Graph-7: Pie chart showing the percentage of Attrition that is caused***

**CHAPTER 7**

**CONCLUSION AND FUTURE SCOPE**

The primary objective of this study is to provide invaluable assistance to the human resources department by furnishing insights into the potential decisions of employees on the verge of termination, specifically those intending to leave the company voluntarily. Our proposed model not only anticipates the concealed risk of employee attrition but also gauges its accuracy in making correct predictions.

The dataset underwent a meticulous evaluation to identify the most influential attributes motivating employees to part ways with the organization. In the quest for the optimal model tailored to job requirements and dataset characteristics, the study found that Long-Short Term Memory (LSTM) demonstrates the most robust performance, albeit potentially challenging for less complex roles. Single Layer Perceptron models and sequential feed-forward models emerge as viable alternatives, offering a balanced trade-off between accuracy and computational complexity. Ensemble techniques, including stacking, boosting, and bagging, further enhance model resilience and performance, underscoring the importance of leveraging deep learning algorithms, coupled with judicious pre-processing and feature selection.

The study identifies critical attributes that significantly impact employee attrition, with Stock Option Level, Monthly Income, Job Satisfaction, Job Involvement, and Total Working Years standing out as the top contributors. The development of a user-centric interface allows individuals to input factors and comprehend the nuanced effects of different value ranges on the attrition rate. Impressively, our model achieves a remarkable accuracy rate of up to 95%, showcasing its prowess in accurately detecting attrition patterns.

Notably, this study comprehensively addresses key questions surrounding the implementation of a highly accurate staff attrition detection model. It emphasizes the need for a careful balance between model accuracy and complexity, considering the intricacies of data quality and the specific challenges being addressed. The trade-off is essential to ensure that the chosen model aligns with the unique demands of the organizational context, contributing meaningfully to human resource management strategies. In conclusion, the study's findings underscore the significance of adopting advanced models, optimizing feature selection, and tailoring solutions to address the multifaceted nature of employee attrition in diverse organizational settings.

The future scope includes continuous refinement and adaptation of the staff attrition detection model through the integration of real-time data feeds, the exploration of enhanced feature engineering techniques, and the encouragement of interdisciplinary collaboration to investigate psychological and social influences on employee decisions. Personalized staff retention methods, seamless integration with human resource systems, and research into explainable AI techniques for increased transparency could all be advancements in the future. Furthermore, the model's adaptability across industries will be a significant focus, assuring its relevance and efficacy in a variety of organizational scenarios.

**CHAPTER 8**

**REFERENCES**

1. VSRD journal employee attrition and strategic retention challenges in indian manufacturing industries : a case study 1pradip kumar talapatra\*, 2saket rungta and 3a.jagadeesh
2. Employee attrition prediction in a pharmaceutical company using both machine learning approach and qualitative data Fatemeh Mozaffari, Marzieh Rahimi, Hamidreza Yazdani, Babak Sohrabi
3. Predicting Employee Attrition Using Machine Learning Approaches by Ali Raza1ORCID,Kashif Munir ,Mubarak Almutairi,Faizan Younas 1 andMian Muhammad Sadiq Fareed 1
4. predicting employee attrition and performance using deep learning samer m. arqawi1 , mohammed a. abu rumman2, eman akef zitawi3
5. HR Analytics: Employee Attrition Analysis using Random Forest Shobhanam Krishna and Sumati Sidharth\*
6. Interpreting the Factors of Employee Attrition using Explainable AI by Karthik Sekaran; Shanmugam S
7. Employee Attrition Analysis Using CatBoost Md. Monir Ahammod Bin Atique, Md. Nesarul Hoque & Md. Jamal Uddin
8. Analyzing Employee Attrition Using Machine Learning: the New AI Approach Shobhanam Krishna; Sumati Sidharth
9. A New Approach for Employee Attrition Prediction Lydia Douaidi & Hamamache Kheddouci
10. A Comparative Study of Employee Attrition Analysis Using Machine Learning and Deep Learning Techniques Sini Raj Pulari, A. Punitha, S. Raja Varshni Meenachi & Shriram Vasudevan
11. Saleh, et al. Brain Tumor Classification Using Deep Learning, 2020 International Conference on Assistive and Rehabilitation Technologies (iCareTech). IEEE, 2020.
12. Setiawan, S. Suprihanto, A. Nugraha, J. Hutahaean, HR analytics: Employee attrition analysis using logistic regression. In IOP Conference Series: Materials Science and Engineering; IOP Publishing: Bandung, Indonesia, 2020; vol. 830, pp. 32-41, 2020.
13. S.Taylor, N. El-Rayes, M. Smith. An Explicative and Predictive Study of Employee Attrition Using Tree-based Models. In Proceedings of the 53rd Hawaii International Conference on System Sciences, Hawaii, HI, USA, pp.7–10 January 2020.
14. R. Yedida, R. Reddy, R. Vahi, R. Jana, A. GV, D. Kulkarni. Employe Attrition Prediction. arXiv 2018, arXiv:1806.10480.
15. Abu Naser, S.S. “Evaluating the effectiveness of the CPP-Tutor, an intelligent tutoring system for students learning to program in C++” Journal of Applied Sciences Research, vol. 5, no. 1, pp. 109-114, 2009.
16. Albatish, I.M., et al. Modeling and controlling smart traffic light system using a rule based system. Proceedings - 2019 International Conference on Promising Electronic Technologies, ICPET 2019, pp. 55–60, 2019, 8925318
17. Elzamly, A., et al. “A new conceptual framework modelling for cloud computing risk management in banking organizations” International Journal of Grid and Distributed Computing, vol. 9, no. 9, pp. 137–154, 2016.
18. Buhisi, N. I., et al. “Dynamic programming as a tool of decision supporting” Journal of Applied Sciences Research, vol. 5, no. 6, pp. 671-676, 2009.
19. Naser, S. S. A. “Developing an intelligent tutoring system for students learning to program in C++” Information Technology Journal, vol. 7, no. 7, pp. 1051-1060, 2008.
20. Fawzy A., et al. “Mechanical Reconfigurable Microstrip Antenna” International Journal of Microwave and Optical Technology, vol. 11, no. 3, pp.153-160, 2016.
21. Elzamly, A., et al. “A new conceptual framework modelling for cloud computing risk management in banking organizations” International Journal of Grid and Distributed Computing, vol. 9, no. 9, pp. 137–154, 2016.
22. Buhisi, N. I., et al. “Dynamic programming as a tool of decision supporting” Journal of Applied Sciences Research, vol. 5, no. 6, pp. 671-676, 2009.
23. Naser, S. S. A. “Developing an intelligent tutoring system for students learning to program in C++” Information Technology Journal, vol. 7, no. 7, pp. 1051-1060, 2008.
24. Fawzy A., et al. “Mechanical Reconfigurable Microstrip Antenna” International Journal of Microwave and Optical Technology, vol. 11, no. 3, pp.153-160, 2016.
25. B. K. Bhardwaj and S. Pal, “Data Mining: A prediction for performance improvement using classification”
26. ] Q. a. Al-Radaideh and E. Al Naqi, “Using Data Mining Techniques to Build a Classification Model for Predicting Employees Performance,” Int. J. Adv. Comput. Sci. Appl., vol. 3, no. 2, pp. 144–151, 2012.
27. N. Deepa, J. S. Priya, and T. Devi, “Towards applying Internet of Things and Machine Learning for the Risk Prediction of COVID-19 in pandemic situation using Naive Bayes Classifier for improving Accuracy,” Mater. TODAY Proc., 2022, doi: 10.1016/j.matpr.2022.03.345.
28. A. Yudhana, D. Sulistyo, and I. Mufandi, “GIS-based and Naïve Bayes for nitrogen soil mapping in Lendah, Indonesia,” Sens. Bio-Sensing Res., vol. 33, p. 100435, 2021, doi: 10.1016/j.sbsr.2021.100435.
29. Studies, 4, 85-91. doi: 10.4236/jhrss.2016.42010. [3] Firth, Lucy & Mellor, David & Moore, Kathleen (Kate) & Loquet, Claude. (2004). How Can Managers Reduce Employee Intention to Quit?. Journal of Managerial Psychology. 19. 170-187. 10.1108/02683940410526127.
30. Gloor, P. A., Fronzetti Colladon, A., Grippa, F., & Giacomelli, G. (2017). Forecasting Managerial Turnover through E-Mail Based Social Network Analysis. Computers in Human Behavior, 71, 343-352. http://dx.doi.org/10.1016/j.chb.2017.02.017