

PREDICTION OF EMPLOYEE ATTRITION USING MACHINE LEARNING

*A project report submitted in the partial fulfilment of the requirements for the award
of the degree of*

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Submitted By

P. Bhavani Manjunadha	(20471A05M2)
G. Venkata Sai	(20471A05K6)
A. Narendra Reddy	(20471A05O0)

Under the esteemed guidance of

Mr.SK. CH.M. Subhani M.Tech
ASSISTANT PROFESSOR



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET
(AUTONOMOUS)**

Accredited by NAAC with A+ Grade and NBA under Cycle -1
NIRF rank in the band of 251-320 and an ISO 9001:2015

Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada
KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET-522601
2023-2024

**NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET
(AUTONOMOUS)**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE



This is to certify that the main project entitled “**Prediction of Employee Attrition Using Machine Learning**” is a bonafide Work done by “**P. Bhavani Manjunadha(20471A05M2), G. Venkata Sai(20471A05K6), A. Narendra Reddy (20471A05O0)**” in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in the Department of **COMPUTER SCIENCE AND ENGINEERING** during the academic year 2022- 2023.

PROJECT GUIDE

Mr.SK.CH. M. Subhani M.Tech.

PROJECT CO-ORDINATOR

Dr. M. Sireesha M.Tech., Ph.D.

HEAD OF THE DEPARTMENT

Dr. S. N. Tirumala Rao M.Tech., Ph.D.

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We wish to express our thanks to carious personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairperson sir **M. V. Koteswara Rao**, B.sc who took keen interest on us in every effort throughout this course. We owe out gratitude to our principal **Dr.M. Sreenivasa Kumar**, M.Tech., Ph.D(UK), MISTE, FIE(1) for his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude to **Dr. S. N. Tirumala Rao**, M.Tech., Ph.D. head of the department (HOD),computer science and engineering(CSE) department and our guide **Dr.M. Sireesha** M.Tech.,Ph.D of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks to **Dr. M. Sireesha** M.Tech., Ph.D. Coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff of department for their cooperation and encouragement during our B. Tech degree. we have no words to acknowledge the warm affection, constant inspiration and encouragement that we receive from our parents.

We affectionately acknowledge the encouragement received from our friends and those who involved in giving valuable suggestions and clarifying out doubts, which had really helped us in successfully completing our project.

By

P. Bhavani Manjunadha (20471A05M2)

G. Venkata Sai (20471A05K6)

A. Narendra Reddy (20471A05M2)

ABSTRACT

Attrition is the departure of employees from the organization for any reason (voluntary or involuntary), including resignation, termination, death or retirement. Attrition rate refers to the rate at which employees leave a company over a specified period of time. It is typically expressed as a percentage and is calculated by dividing the number of employees who leave the company (voluntarily or involuntarily) by the average number of employees during that period, and then multiplying by 100. Attrition can be caused by a variety of factors, including dissatisfaction with the job, a poor work environment, or a desire for a better salary or benefits. The attrition reports are seen by the human resources department and the manager of the department where the employee who left work worked. we have implemented various Machine Learning Algorithms to predict the attrition rate of an organization. We have followed various processes such as Dataset Collection, Data Preprocessing, Data Visualization, apply SMOTE on the data and training the model by using the various machine learning models like Random Forest, Decision Tree, Extra Tree Classifier, Logistic Regression, K-Nearest Neighbors, Gradient Boost and XG Boost Classifier algorithms. The results are evaluated using accuracy score and confusion matrix. XG Boost classifier algorithm giving the best accuracy i.e., 91.16% compared to the other evaluating models. This work will help organizations to better understand the attrition causes.

INDEX

S. No.	CONTENTS	PAGENO
I	LIST OF FIGURES	XV
1.	INTRODUCTION	1
	1.1. Introduction	1
	1.2. Existing System	2
	1.3. Proposed System	2
	1.4. System Requirements	3
	1.4.1 Hardware Requirements	3
	1.4.2 Software Requirements	3
2.	LITERATURE	4
	2.1 Literature Survey	4
3.	SYSTEM ANALYSIS	6
	3.1 Some Machine Learning Methods	6
	3.2 Applications of Machine Learning	7
	3.3 Importance of machine learning	7
	3.4 Implementation of machine learning using Python	8
	3.5 Scope of the project	9
4.	METHODOLOGY	10
	4.1 Data Set	10
	4.2 Data Preprocessing	12
	4.3 Implementation of Machine Learning Algorithm on Training Data	16
5.	IMPLEMENTATION CODE	22
	5.1 Backend	22
	5.2 Frontend	27
	5.3 Connection	29
6.	RESULT ANALYSIS	31
7.	OUTPUT SCREENS	28
8.	CONCLUSION AND FUTURE SCOPE	33
9.	BIBLIOGRAPHY	35

LIST OF FIGURES

S.NO.	FIG NO.	CONTENTS	PAGENO
1	3.1	Types of machine learning	5
2	4.1	Methodology	10
3	4.2	Employee dataset	11
4	4.3	Null values info	12
5	4.4	Correlation	13
6	4.5	Feature Importance	15
7	4.6	Imbalanced data	15
8	4.7	Balanced data	16
9	4.8	Random Forest	17
11	6.1	Result Analysis	26
12	6.2	Logistic Regression confusion matrix	27
13	6.3	Decision tree confusion matrix	27
14	6.4	Random Forest confusion matrix	27
15	7.1	Home Screen	28
16	7.2	After giving input	28
17	7.3	Evaluation of employee attrition for discontinuation	29
18	7.4	Evaluation of employee attrition for continuation	29

1. INTRODUCTION

1.1 INTRODUCTION

In today's world data is being created at an ever-increasing rate. The analysis of this stored data has proved to be beneficial in gaining insights and creating general awareness about any business or organization. Data analysis is the process of collecting, inspecting, cleansing, transforming, and modeling raw data with the aim of deriving valuable insights and retrieving relevant information to reach a conclusion for good decision making. Machine Learning is the process of using algorithms to train a machine to predict accurately using the existing data[1]. Employees are a crucial resource for any organization, and hence withdrawal of productive employees might affect an organization with respect to various aspects. Some of the consequences of Employee Attrition are: Investing in staffing and training new employees[3], increased burden on existing employees and a decline in the performance[14] of the organization. In this work, we intend to classify employees with respect to previous 'Attrition' patterns and other relevant attributes. The outcome of many research shows that the most valuable asset and important resource in organizations are their employees.

Now a day due to increased competition and improved requirement in employees' proficiency determines the attrition rate. The employee attrition is considered to be a serious issue for organizations. The cost of searching and training employees is very high. Organizations need to search, hire and train new employees. Loss of experienced workers especially high performers is difficult to manage and is negatively related to the success and performance of organizations. The study focuses on the variables that may lead to control the attrition rate of the employee. The problem of employee turnover has turn to eminence in organizations because of its pessimistic impacts on issues on work place self-esteem and efficiency. The organizations deal with this problem is by predicting the risk of attrition of employees using machine learning techniques thus giving organizations to take proactive action for retention.

1.2 EXISTING SYSTEM

Employee attrition is defined as employees leaving their organizations for unpredictable or uncontrollable reasons. Many terms make up attrition, the most common being termination, resignation, planned or voluntary retirement, structural changes, long-term illness, layoffs. In the traditional methods, all the activities inside an organization are carried out manually. The employee records are maintained manually and if any employee wants to leave an organization, they are maintained manually. All this requires a lot of human effort and time and they are prone to a lot of errors. There are many factors involved which may cause a potential employee to leave an organization.?

For example, an employee may leave an organization due to poor salary, lack of proper infrastructure, and many other factors. All these things can't be predicted manually. To overcome these shortcomings, an employee attrition model is designed by using various machine learning techniques. The model is trained properly by providing the proper data so that the model makes good predictions. The performance of the model is evaluated by calculating the proper accuracy score.

DISADVANTAGES

- Prediction accuracy is not good when compared to other methods.
- Less efficient and less robust when compared to other methods

1.3 PROPOSED SYSTEM

Initially the data is downloaded from Kaggle is pre-processed first so that we can extract important features like Monthly Income, Last Promotion Year, Salary Hike and etc. that are quite natural for employee attrition. Dependent variables or Predicted variable are the one that helps to get the factors that mostly dependent on employee related variables[7]. For example the employee ID or employee count has nothing to do with the attrition rate. Exploratory Data Analysis is an initial process of analysis, in which you can summarize characteristics of data to can predict who, and when an employee will terminate the service. The system builds a prediction model by using random forest technique. It is one of the ensembles learning technique which consists of several decision trees rather than a single decision tree for classification. The techniques perform dependent variable analysis and word formation vector to evaluate the employee churn. Hence, by improving employee assurance and providing a desirable working environment, we can certainly reduce this problem significantly

To deal with the problem, we developed automatic employee attrition prediction using machine learning techniques. We will train the machine with previous dataset. so, machine can analyse and understand the process. Then machine will check for employee to get attrition and give us result.

ADVANTAGES

- More robust and reliable when compared to all the previous methods.
- Prediction accuracy is good and it consumes less time in order to make predictions.
- Whole process will be automated, so human error will be avoided.
- Eligible applicant will be sanctioned loan without any delay.

1.4 SYSTEM REQUIREMENTS

1.4.1 HARDWARE REQUIREMENTS

- Processor : Intel Core i5
- Cache Memory : 4MB
- Hard Disk : 30GB or more
- RAM : 8GB

1.4.2 SOFTWARE REQUIREMENTS

- Coding Language : Python
- Python Distribution : Anaconda, Flask
- Browser : Any Latest Browser Like Chrome

2. LITERATURE

2.1 LITERATURE SURVEY

In this paper, modified approaches using various data mining techniques are collected to analyze the employee attrition rate at various levels. The study related to data mining for extracting the employee's attrition rate used in various models and the comprehensive literature review of various researcher's works are stated below;

Qasem A, A.Radaideh and Eman A Nagi, has applied data mining techniques to build a classification model to predict the performance of employees. They adopted CRISP-DM data mining methodology in their work. The Decision tree was the main data mining tool used to build the classification model, where several classification rules were generated. They validated the generated model; several experiments were conducted using real data collected from several companies. The model is intended to be used for predicting new applicants' performance.

Amir Mohammad Esmayeeli Sikaroudi, Rouzbeh Ghousi and Ali Esmayeeli Sikaroudi et al, implemented knowledge discovery steps on real data of a manufacturing plant. They chew over many characteristics of employees such as age, technical skills and work experience. They used to find out importance of data features is measured by Pearson Chi-Square test.

John M. Kirimi and Christopher Moturi et al, proposed a prediction model for employee performance forecasting that enables the human resource professionals to refocus on human capability criteria and thereby enhance the performance appraisal process of its human capital.

Rohit Punnoose and Pankaj Ajit et al, explored the application of Extreme Gradient Boosting (XGBoost) technique which is more robust because of its regularization formulation. Data from the HRIS of a global retailer is used to compare XGBoost against six historically used supervised classifiers and demonstrate its significantly higher accuracy for predicting employee turnover.

Kagmar et al. (2006) argues that some employee turnover can be beneficial for an organization but that turnover in most cases are very costly and can disrupt the workflow. As an example how costly employee turnover could be we turn our eyes towards the U.S. fast food industry. This is an industry which is exposed to high employee turnover, and only retraining costs of new employees per year is as high as \$4.3 billion (Kagmar et al, 2006). The first step of being able to reduce

employee turnover is to understand why the phenomenon is happening in the first place, and that is extremely important. This is because, knowing what causes the problem, gives organizations the chance to take action.

Batty Dorance Jeen (2014), mentioned that many organizations have the concern of employee turnover. It is highly destructive to both the organization as well as the employees. The research was conducted for the retail industry in Bangalore. Despite the incentives, motivational techniques and old practices of HRM there is still high attrition rate so study show cases ways to reduce the intentions of employee leaving the organization. This is performed by distributing questionnaires to retail outlets in Bangalore. The study concluded that turnover intention has influence on attrition factors such as QWL, career growth, working hours, personal/family reasons, and relation with internal co – worker, welfare, working condition, and salary.

Venkata Naga Manjula, Ruchita Ramani, Swati John (2013), articulated that the objective of the study is to understand the growth of ITES sector in India. The study in the BPO Industry is to understand and gauge the attrition rate, its intensity and make a causal analysis, to design strategy to stabilise the sector by suggesting mitigating the attrition.

Shivani Mishra, Deepa Mishra (2013), articulated that the study carried out is for shipping industry of Kutch, Gujarat. The purpose of the study is to analyse the turnover and the commitment to identify several domains of organization, human resource practices and other like employee characteristic and environmental factors, which may have a positive or negative impact on employees' intention to stay with an organization.

Vibha Gupta (2013), stated that in recent years the turnover is high in BPO sectors either by absenteeism or employees absconding without any prior notice. Turnover rates for permanent Agents/Executives were 15.6% in 2009 and 35% in 2012. Department of Human Resources which also tracks attrition of temporary employees measured the turnover rate for temporary employees to be 77% in 2012. Therefore study is focused on recruitment and retention challenges that the IT/BPO industry currently faces and to examine ways to reduce high turnover rates among first year Employees in the leading Domestic Call Center based in Indore.

3. SYSTEM ANALYSIS

3.1 SOME MACHINE LEARNING METHODS

Machine learning uses two techniques: supervised learning, which trains a model on known input and output data to predict future outputs, and unsupervised learning, which uses hidden patterns or internal structures in the input data.

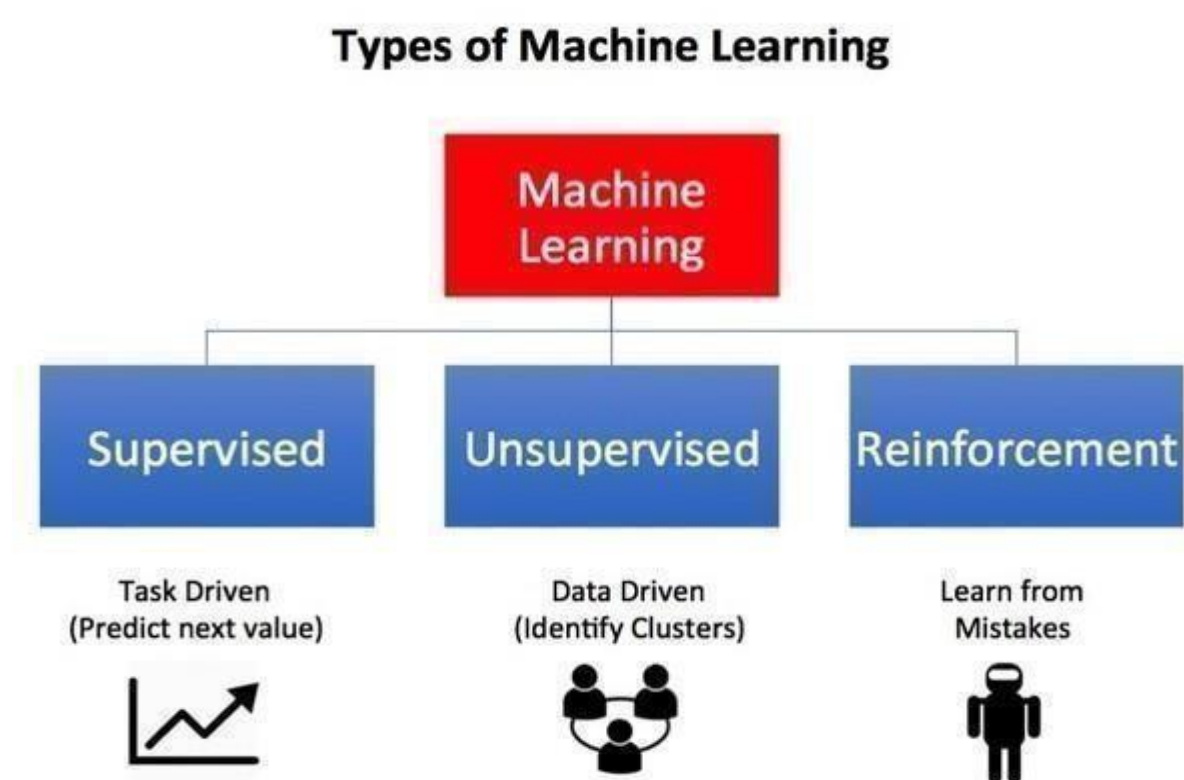


Figure: 3.1 Types of Machine Learning

Machine learning algorithms are often categorized as supervised and unsupervised.

Supervised machine learning algorithms:

Supervised machine learning creates a model that makes predictions based on evidence in the presence of uncertainty. A supervised learning algorithm takes a known set of input data and known responses to the data (**output**) and trains a model to generate reasonable predictions for the response to the new data. Use supervised learning if you have known data for the output you are trying to estimate. Supervised learning uses classification and regression techniques to develop machine learning models.

Unsupervised machine learning algorithms:

Detects hidden patterns or internal structures in unsupervised learning data. It is used to eliminate datasets containing input data without labeled responses. Clustering is a common unsupervised learning technique. It is used for exploratory data analysis to find hidden patterns and clusters in the data. Applications for cluster analysis include gene sequence analysis, market research, and commodity identification.

Reinforcement machine learning algorithms:

Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behavior or path it should take in a specific situation. Reinforcement learning differs from supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience.

3.2 APPLICATIONS OF MACHINE LEARNING

1. Virtual Personal Assistants
2. Predictions while Commuting
3. Videos Surveillance
4. Social Media Services
5. Email Spam and Malware Filtering
6. Online Customer Support
7. Search Engine Result Refining
8. Product Recommendations
9. Online Fraud Detection

3.3 IMPORTANCE OF MACHINE LEARNING IN EMPLOYEE ATTRITION

The use of machine learning classification models to predict whether an employee is likely to quit could greatly increase the human resource department's ability to intervene on time and possibly provide a remedy to the situation to prevent attrition.

3.4 IMPLEMENTATION OF MACHINE LEARNING USING PYTHON

Python is a popular programming language. It was created in 1991 by Guido van Rossum. It is used for:

- 1.web development (server-side),
- 2.software development,
- 3.mathematics,
- 4.system scripting.

The most recent major version of Python is Python 3. However, Python 2, although not being updated with anything other than security updates, is still quite popular. It is possible to write Python in an Integrated Development Environment, such as Thonny, PyCharm, NetBeans or Eclipse, Anaconda which are particularly useful when managing larger collections of Python files. Python was designed for its readability. Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.

Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose. In the older days, people used to perform Machine Learning tasks manually by coding all the algorithms and mathematical and statistical formula. This made the process time consuming, tedious and inefficient. But in the modern days, it is become very much easy and efficient compared to the olden days by various python libraries, frameworks, and modules. Today, Python is one of the most popular programming languages for this task and it has replaced many languages in the industry, one of the reasons is its vast collection of libraries.

Python libraries that used in Machine Learning are:

- 1.Numpy
- 2.Scipy
- 3.Scikit-learn
- 4.Pandas
- 5.Matplotlib

NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High-end libraries like TensorFlow uses NumPy internally for manipulation of Tensors.

SciPy is a very popular library among Machine Learning enthusiasts as it contains different modules for optimization, linear algebra, integration and statistics. There is a difference between the SciPy library and the SciPy stack. The SciPy is one of the core packages that make up the SciPy stack. SciPy is also very useful for image manipulation.

Skikit-learn is one of the most popular Machine Learning libraries for classical Machine Learning algorithms. It is built on top of two basic Python libraries, NumPy and SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms. Scikit learn can also be used for data-mining and data-analysis, which makes it a great tool who is starting out with Machine Learning.

Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the dataset must be prepared before training. In this case, Pandas comes handy as it was developed specifically for data extraction and preparation. It provides high-level data structures and wide variety tools for data analysis. It provides many inbuilt methods for groping, combining and filtering data.

Matpoltlib is a very popular Python library for data visualization. Like Pandas, it is not directly related to Machine Learning. It particularly comes in handy when a programmer wants to visualize the patterns in the data. It is a 2D plotting library used for creating 2D graphs and plots. A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc. It provides various kinds of graphs and plots for data visualization, histogram, error charts, bar chats, etc.

3.5 SCOPE OF THE PROJECT

The scope of this system is to maintain employee details in datasets, train the model using the large quantity of data present in datasets and predict whether the employee will continue or not in the organization.

4. METHODOLOGY

There are various steps involved in a machine learning project. The standard steps that you've to follow for a machine learning project. For any project, first, we have to collect the data according to our needs. The next step is to clean the data like removing values, removing outliers, handling imbalanced datasets, changing categorical variables to numerical values, etc.

After that training of a model, use various machine learning and deep learning algorithms. Next, is model evaluation using different metrics like recall, f1 score, accuracy, etc.

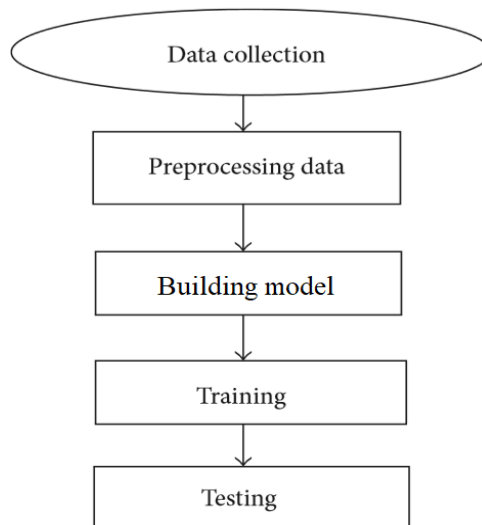


Figure: 4.1 Methodology

4.1 DATA SET

Dataset is a group of data. Most commonly a dataset agrees to the contents of a single database, where every column of the table signifies a particular variable, and each row agrees to a member of the dataset. For our project we take employee statistics from IBM which contains 1470 records and 35 fields including categorical and numeric features. Each record in the employee dataset signifies a single employee information and each field in the record signifies a feature of that particular employee[13].

Some of the attributes are:

Age- Numeric Discrete

Attrition-Categorical

Business Travel- Categorical

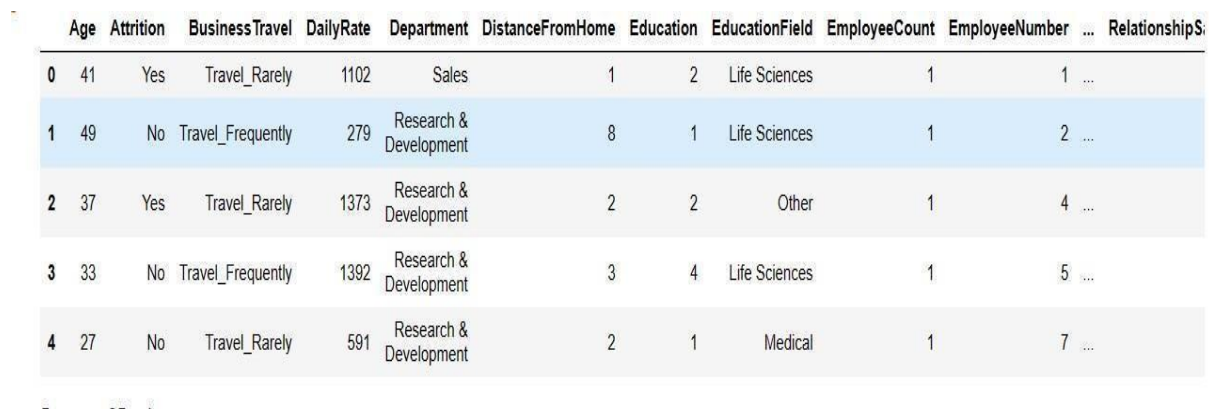
Daily Rate- Numeric Discrete

Department- Categorical

DistanceFromHome- Numeric
 Education- Categorical
 Education Field- Categorical
 Employee Count- Numeric Discrete
 Employee Number- Numeric Discrete
 Environment Satisfaction- Categorical
 Gender- Categorical
 Hourly Rate- Numeric Discrete
 Job Involvement- Categorical
 JobLevel-Categorical
 JobRole-Categorical

Data Set Link:

https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attrition-dataset?select=WA_Fn-UseC_-HR-Employee-Attrition.csv



	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipS
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...	

Figure:4.2 Employee dataset attribute names and information

The above figure is the part of employee dataset . There are 1470 rows and 35 attributes in dataset. 'Attrition' is the target variable to predict and values for it are 'YES' and 'No'.

4.2 DATA PREPROCESSING

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task. A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

4.2.1 Verifying NULL values

An easy way to check for missing values is to use the method *isnull*. We will get a data frame with true (1) and false (0) values, so we will sum the values and we can see in which column we have missing values.

```
df=pd.read_csv('employee.csv')
df.isna().sum()
```

```
Out[5]: Age                0
Attrition                 0
BusinessTravel            0
DailyRate                0
Department               0
DistanceFromHome         0
Education                0
EducationField            0
EmployeeCount            0
EmployeeNumber           0
EnvironmentSatisfaction  0
Gender                   0
HourlyRate               0
JobInvolvement           0
JobLevel                 0
JobRole                  0
JobSatisfaction          0
MaritalStatus            0
MonthlyIncome            0
MonthlyRate              0
NumCompaniesWorked       0
Over18                   0
```

Figure 4.3 Null values info

4.2.2 Correlation of attributes

Correlation is the statistical measure of the relationship between two variables. There are different types of correlation coefficients like Pearson coefficient (linear) and Spearman coefficient (non-linear) which capture different degrees of probabilistic dependence but not necessarily causation. We can find dependency between two attributes p and q using Correlation coefficient method using the formula. $r_{p,q} = \frac{\sum (p_i - \bar{p})(q_i - \bar{q})}{n\sigma_p\sigma_q} = \frac{\sum (p_i q_i) - n\bar{p}\bar{q}}{n\sigma_p\sigma_q}$ n is the total number of patterns, p_i and q_i are respective values of p and q attributes in patterns i, \bar{p} and \bar{q} are respective mean values of p and q attributes, σ_p , σ_q are respective standard deviations values of p and q attributes. Generally, $-1 \leq r_{p,q} \leq +1$. If $r_{p,q} < 0$, then p and q are negatively correlated. If $r_{p,q} = 0$, then p and q are independent attributes and there is no correlation between them. If $r_{p,q} > 0$, then p and q are positively correlated. We can drop the attributes that are having correlation coefficient value as 0 as it indicates that the variables are independent with respect to the prediction attribute.

	Age	DailyRate	DistanceFromHome	Education	EnvironmentSatisfaction	HourlyRate	JobInvolvement	JobLevel	JobSatisfaction
Age	1.000000	0.010661	-0.001686	0.208034	0.010146	0.024287	0.029820	0.509604	-0.004892
DailyRate	0.010661	1.000000	-0.004985	-0.016806	0.018355	0.023381	0.046135	0.002966	0.030571
DistanceFromHome	-0.001686	-0.004985	1.000000	0.021042	-0.016075	0.031131	0.008783	0.005303	-0.003669
Education	0.208034	-0.016806	0.021042	1.000000	-0.027128	0.016775	0.042438	0.101589	-0.011296
EnvironmentSatisfaction	0.010146	0.018355	-0.016075	-0.027128	1.000000	-0.049857	-0.008278	0.001212	-0.006784
HourlyRate	0.024287	0.023381	0.031131	0.016775	-0.049857	1.000000	0.042861	-0.027853	-0.071335
JobInvolvement	0.029820	0.046135	0.008783	0.042438	-0.008278	0.042861	1.000000	-0.012630	-0.021476
JobLevel	0.509604	0.002966	0.005303	0.101589	0.001212	-0.027853	-0.012630	1.000000	-0.001944
JobSatisfaction	-0.004892	0.030571	-0.003669	-0.011296	-0.006784	-0.071335	-0.021476	-0.001944	1.000000
MonthlyIncome	0.497855	0.007707	-0.017014	0.094961	-0.006259	-0.015794	-0.015271	0.950300	-0.007157
MonthlyRate	0.028051	-0.032182	0.027473	-0.026084	0.037600	-0.015297	-0.016322	0.039563	0.000644
NumCompaniesWorked	0.299635	0.038153	-0.029251	0.126317	0.012594	0.022157	0.015012	0.142501	-0.055699
PercentSalaryHike	0.003634	0.022704	0.040235	-0.011111	-0.031701	-0.009062	-0.017205	-0.034730	0.020002
PerformanceRating	0.001904	0.000473	0.027110	-0.024539	-0.029548	-0.002172	-0.029071	-0.021222	0.002297
RelationshipSatisfaction	0.053535	0.007846	0.006557	-0.009118	0.007665	0.001330	0.034297	0.021642	-0.012454
StockOptionLevel	0.037510	0.042143	0.044872	0.018422	0.003432	0.050263	0.021523	0.013984	0.010690
TotalWorkingYears	0.680381	0.014515	0.004628	0.148280	-0.002693	-0.002334	-0.005533	0.782208	-0.020185
TrainingTimesLastYear	-0.019621	0.002453	-0.036942	-0.025100	-0.019359	-0.008548	-0.015338	-0.018191	-0.005779
WorkLifeBalance	-0.021490	-0.037848	-0.026556	0.009819	0.027627	-0.004607	-0.014617	0.037818	-0.019459
YearsAtCompany	0.311309	-0.034055	0.009508	0.069114	0.001458	-0.019582	-0.021355	0.534739	-0.003803
YearsInCurrentRole	0.212901	0.009932	0.018845	0.060236	0.018007	-0.024106	0.008717	0.389447	-0.002305
YearsSinceLastPromotion	0.216513	-0.033229	0.010029	0.054254	0.016194	-0.026716	-0.024184	0.353885	-0.018214
YearsWithCurrManager	0.202089	-0.026363	0.014406	0.069065	-0.004999	-0.020123	0.025976	0.375281	-0.027656

Figure:4.4 Correlation

The data that we have had a large number of attributes, but we have used some major attributes in finding out the turnover rate. We have found out many interesting relationships from figure 3.4.1 among these attributes that led us to our goal of finding the turnover rate and in which year the turnover rate touched its peak. In our data, we have shown a correlation

between attributes such as how many years an employee spent in a company, how many years an employee spent in a company with current manager and how many years spent in the company since the last promotion. We have also shown the correlation between the level of job or service an employee is doing and monthly income of the employee. We have also considered the relation between the attributes like percent hike and the performance rating of an employee. We have also found out the correlation between attributes such as number of years spent by an employee under the current manager, the level of the job and percentage of hike in salary. So, we have used a number of attributes and correlations among them to find out the turnover rate of a company in a certain period of time.

4.2.3 Feature Selection

Feature Selection is considered as the most crucial theory in the fields of machine learning which has a significant amount of impact on the actual performance of the model you are building. These features can be simply used to coach your model and have an enormous influence on the performance. Trivial and unrelated features can have a negative impact on the performance of the model. Feature selection and Data cleaning should be the first and most significant step of your model designing. Feature Selection is the process where you automatically or manually select those features on the basis of some various techniques like Univariate Selection Feature Importance Correlation Matrix which contribute most to your dependent variable or output variable in which you are interested in. After analysing the dataset manually we came to a conclusion that these features Employee Count, Employee Number, Over18 have no direct impact on our output variable Attrition. Therefore, these features have been completely neglected before applying any feature selection methods[8].

Feature Importance

Feature importance gives you a score for each feature of your data, the higher the score more important or relevant is the feature towards your output variable. Feature importance is an inbuilt class that comes with Tree Based Classifiers, we will be using Extra Tree Classifier for extracting the top features for the dataset.

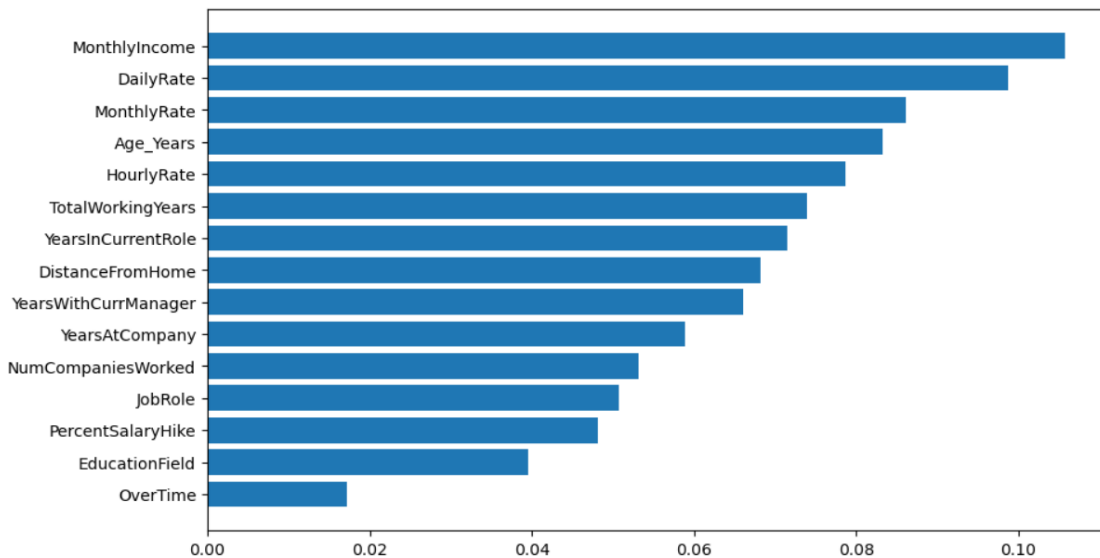


Figure 4.5 Feature Importance

The diagram above represents the feature importance of each feature of our dataset with the help of this feature importance method we could analyse that the features like Monthly income, Age , Daily rate , Hourly rate etc are some of the significant attributes . Along with that we came to conclusion that the features like Business travel Gender, Department, Performance rating are having least impact on our output variable Attrition. Therefore we can neglect these features beforehand.

4.2.4 Imbalanced Dataset

In the dataset 90% records are labelled with class YES and remaining 10% records are labelled with class NO. This type of datasets are called as imbalanced datasets and can have adverse effect on the performance of the model it makes the model biased towards majority class of output variable. Therefore handling imbalanced dataset becomes a necessary task for this type of problem statement.

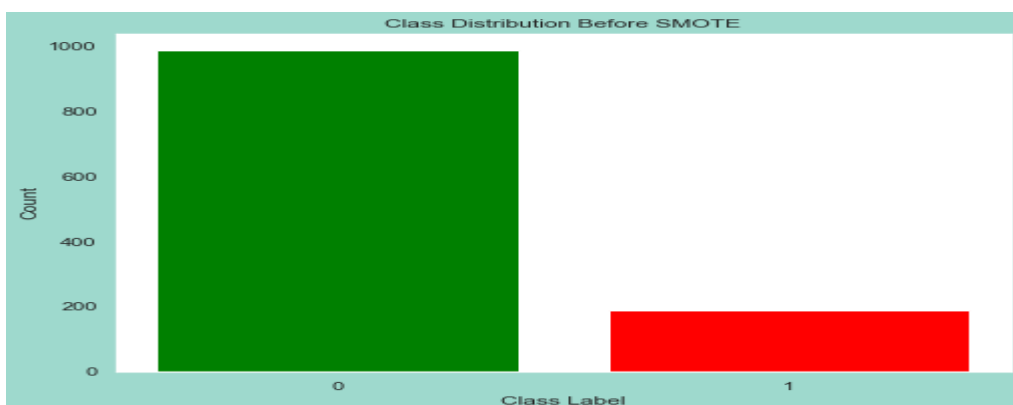


Figure 4.6 Imbalanced data

Following are some of the methods to handle imbalanceness of dataset

- 1.Random Under Sampling
2. Random Over Sampling
- 3.Custer Based Over Sampling
- 4.SMOTE

For our dataset we are using over SMOTE(Synthetic Minority Over-Sampling Technique) method to handle the imbalanceness of the dataset. Before over sampling 1233 records were labelled with class NO and only 237 records were labelled with class YES. After performing over sampling we similaried the number of records of both classes to 1233 records as shown in the diagram below

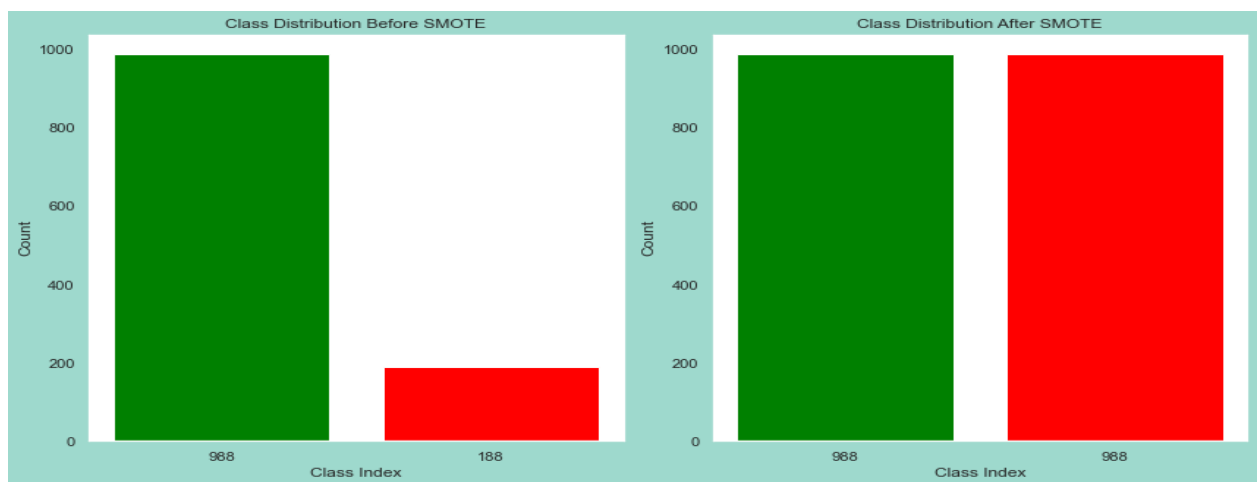


Figure 4.7 Balanced data

4.3 IMPLEMENTATION OF MACHINE LEARNING ALGORITHM ON TRAINING DATA

Data was gathered from Kaggle, one of the most providers of data sources for the purpose of learning, and hence the data is collected from the Kaggle, which had two sets of details, one of which was for the preparation and the supplementary tests. The dataset for training is the model in which datasets are further divided into datasets was used to train the model train and the minor dataset. For the measuring of the value of attrition, many regression models are applied during this study. The dataset is split into 2 sections.

One half for model training and also the other part for model analysis or testing. During this study, the info set is separated into two-part the first half is termed coaching knowledge and also the second called take a look at data, training data makes up for eighty percent of the whole data used, and the rest for test data. all of those models are trained with the training data part and so evaluated with the test data. The accuracy is checked with the assistance of f1 score.

In this process we use Decision Tree Algorithm and Random Forest Algorithm.

Random Forest Algorithm:

The random forest algorithm improves the flexibility and decision-making capacity of individual trees. It is another machine learning algorithm incorporating the ensemble learning theorem as its foundation, combining results from various decision trees to optimize training. In some use cases of loan and credit risk prediction, some features are more important than the rest or, more specifically, some features whose removal would improve the overall performance. Since we know the fundamentals of decision trees and how they choose features based on information gain, random forests would incorporate these benefits to give superior performance[9].

It uses a tree-like graph to show the possible consequences. If you input a training dataset with targets and features into the decision tree, it will formulate some set of rules. These Rules can be used to perform predictions. There are two stages in Random Forest algorithm, one is random forest creation, and the other is to make a prediction from the random forest classifier created in the first stage.

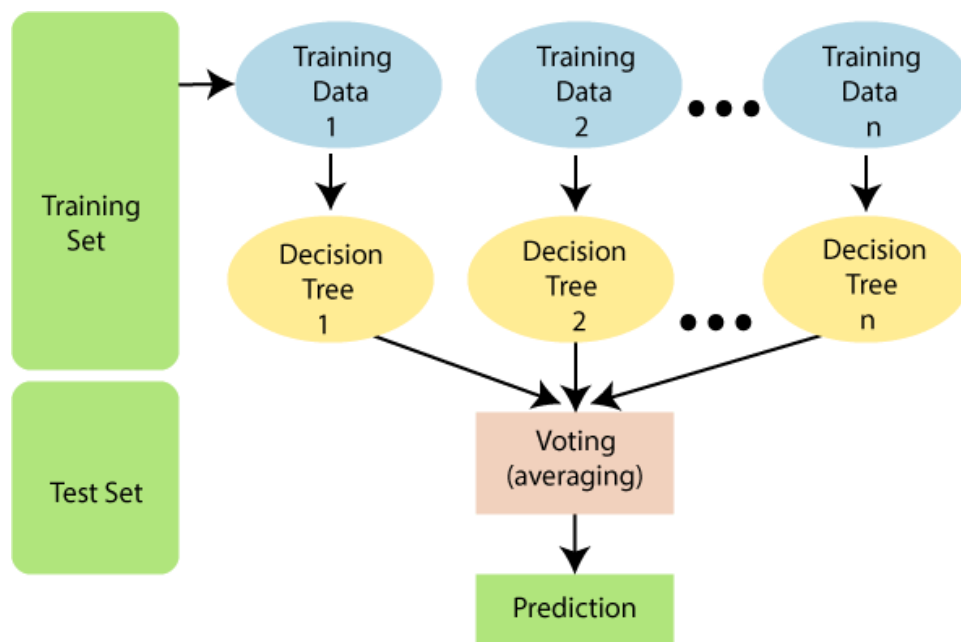


Figure 4.8: Random Forest Algorithm

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Decision Tree Algorithm

A decision tree is a type of supervised machine learning used to categorize or make predictions based on how a previous set of questions were answered. The model is a form of supervised learning, meaning that the model is trained and tested on a set of data that contains the desired categorization. In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node[2][3].

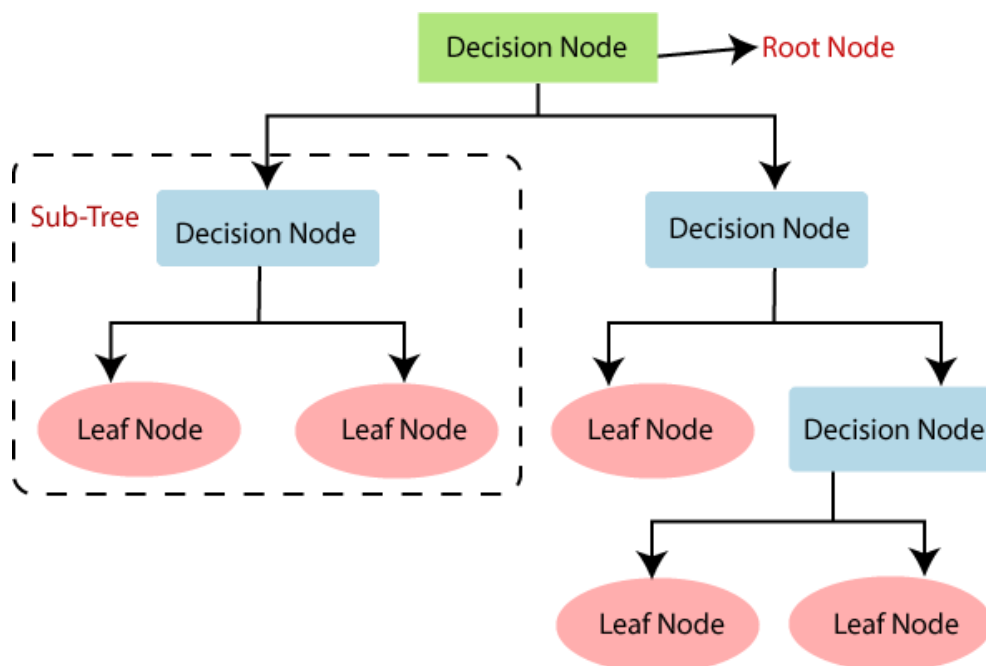


Figure 4.9: Decision tree classifier

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).

Step-3: Divide the S into subsets that contains possible values for the best attributes.

Step-4: Generate the decision tree node, which contains the best attribute.

Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Logistic Regression

Logistic regression is a statistical method used to predict the outcome of a dependent variable based on previous observations. It's a type of regression analysis and is a commonly used algorithm for solving binary classification problems[5][6].

Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems[14].

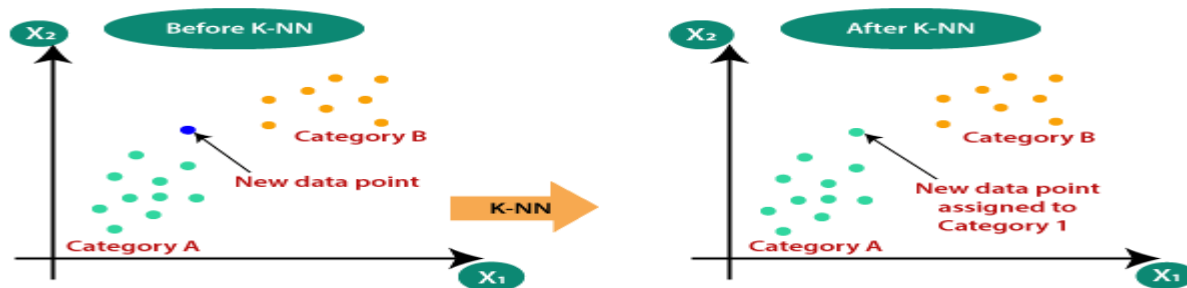
In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets[14].

K-Nearest Neighbor(KNN) Algorithm

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K-NN algorithm.



The K-NN working can be explained on the basis of the below algorithm:

Step-1: Select the number K of the neighbors

Step-2: Calculate the Euclidean distance of K number of neighbors

Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

Extra Tree Classifier:

Extremely Randomized Trees Classifier(Extra Trees Classifier) is a type of ensemble learning technique which aggregates the results of multiple de-correlated decision trees collected in a “forest” to output it’s classification result. In concept, it is very similar to a Random Forest Classifier and only differs from it in the manner of construction of the decision trees in the forest. Each Decision Tree in the Extra Trees Forest is constructed from the original training sample. Then, at each test node, Each tree is provided with a random sample of k features from the feature-set from which each decision tree must select the best feature to split the data based on some mathematical criteria (typically the Gini Index). This random sample of features leads to the creation of multiple de-correlated decision trees.

Gradient Boosting Algorithm:

Gradient boosting is a machine learning ensemble technique that combines the predictions of multiple weak learners, typically decision trees, sequentially. It aims to improve overall predictive performance by optimizing the model's weights based on the errors of previous iterations, gradually reducing prediction errors and enhancing the model's accuracy.

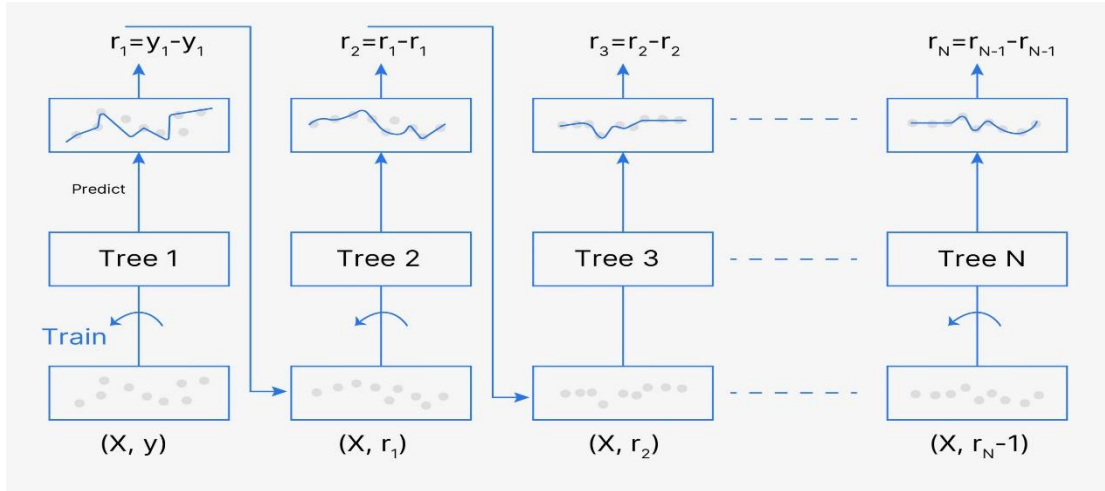


Fig. Gradient Boosting Tree for Regression

XG Boost Classifier:

XGBoost, or eXtreme Gradient Boosting, is a machine learning algorithm under ensemble learning. It is trendy for supervised learning tasks, such as regression and classification. XGBoost builds a predictive model by combining the predictions of multiple individual models, often decision trees, in an iterative manner. The algorithm works by sequentially adding weak learners to the ensemble, with each new learner focusing on correcting the errors made by the existing ones. It uses a gradient descent optimization technique to minimize a predefined loss function during training.

Key features of XGBoost Algorithm include its ability to handle complex relationships in data, regularization techniques to prevent overfitting and incorporation of parallel processing for efficient computation. XGBoost is widely used in various domains due to its high predictive performance and versatility across different datasets



Fig. Flow of XG Boost Classifier

5. IMPLEMENTATION CODE

5.1 BACKEND

Employee Prediction .py

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.datasets import make_classification
from sklearn import metrics
from plotly.subplots import make_subplots
import plotly.graph_objects as go
from plotly.offline import iplot
from warnings import filterwarnings

filterwarnings("ignore")

data = pd.read_csv('WA_Fn-UseC_-HR-Employee-Attrition.csv')

df.describe()

#check for NaN values
df.isnull().sum()

# univariate analysis of categorical data:
sns.set(rc={"axes.facecolor":"white", "figure.facecolor": "#9ed9cd"})
sns.set_palette("pastel")
for i, col in enumerate(cat):

    fig, axes = plt.subplots(1,2,figsize=(10,5))

    # count of col (countplot)

    ax=sns.countplot(data=df, x=col, ax=axes[0])
    activities = [var for var in df[col].value_counts().sort_index().index]
    ax.set_xticklabels(activities,rotation=90)
    for container in axes[0].containers:
        axes[0].bar_label(container)

    #count of col (pie chart)

    index = df[col].value_counts().index
    size = df[col].value_counts().values
    explode = (0.05, 0.05)

    axes[1].pie(size, labels=index,autopct='% 1.1f%%', pctdistance=0.85)

    # Inner circle
    centre_circle = plt.Circle((0,0),0.70,fc='white')
    fig = plt.gcf()
    fig.gca().add_artist(centre_circle)
    plt.suptitle(col,backgroundcolor='black',color='white',fontsize=15)
```

```

plt.show()

plt.figure(figsize = (15,25))
for idx, i in enumerate(num):
    plt.subplot(12, 2, idx + 1)
    sns.boxplot(x = i, data = df)
    plt.title(i,backgroundcolor='black',color='white',fontsize=15)
    plt.xlabel(i, size = 12)
plt.tight_layout()
plt.show()

import numpy as np
import matplotlib.pyplot as plt

# Assuming smote_target is a numpy array
unique_values_smote, counts_smote = np.unique(smote_target, return_counts=True)

plt.figure(figsize=(12, 6))

# Plotting class distribution before SMOTE
plt.subplot(1, 2, 1)
plt.bar(range(len(counts)), counts, color=['green', 'red'])
plt.xlabel('Class Index')
plt.ylabel('Count')
plt.title('Class Distribution Before SMOTE')
plt.xticks(range(len(counts)), counts) # Set the xticks to display only the counts

# Plotting class distribution after SMOTE
plt.subplot(1, 2, 2)
plt.bar(range(len(counts_smote)), counts_smote, color=['green', 'red'])
plt.xlabel('Class Index')
plt.ylabel('Count')
plt.title('Class Distribution After SMOTE')
plt.xticks(range(len(counts_smote)), counts_smote) # Set the xticks to display only the counts

plt.tight_layout()
plt.show()

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_state = 4)
print ('Train set:', X_train.shape, y_train.shape)
print ('Test set:', X_test.shape, y_test.shape)

Training_Accuracy_L=[]
Test_Accuracy_L=[]
Sensitivity_L=[]
Specificity_L=[]
F1Score_L=[]
Precision_L=[]
Negative_Predictive_Value_L=[]
False_Negative_Rate_L=[]
False_Positive_Rate_L=[]
False_Discovery_Rate_L=[]
False_Omission_Rate_L=[]
cv_accuracy_L=[]

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

```

```

import seaborn as sns
import warnings
import os
import scipy

from scipy import stats
from scipy.stats import pearsonr
from scipy.stats import ttest_ind
from sklearn.metrics import make_scorer, roc_auc_score, confusion_matrix, accuracy_score,
roc_curve, classification_report
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import cross_val_score, KFold

import math

def rounder(n):
    try:
        return math.ceil(n * 1000) / 1000
    except:
        return n

def fun(model,name,num_folds):
    test_pred = model.predict(X_test)
    train_pred = model.predict(X_train)

    train_acc=rounder(accuracy_score(y_train,train_pred)*100)
    test_acc=rounder(accuracy_score(y_test,test_pred)*100)

    Training_Accuracy_L.append(train_acc)
    Test_Accuracy_L.append(test_acc)

    print("\nTraining Accuracy:", train_acc)
    print("\nTesting Accuracy:",test_acc)

    print(classification_report(y_test,test_pred))
    test_conf_matrix = confusion_matrix(y_test,test_pred)
    plt.figure(figsize=(4, 4))
    sns.heatmap(test_conf_matrix, annot=True, fmt='g', cmap='Greens', cbar=False)
    t=name+' Confusion Matrix - Test Set'
    plt.title(t)
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.show()

    tn, fp,fn,tp = test_conf_matrix.ravel()

```

```

Sensitivity=rounder((tp) / (tp + fn))
Sensitivity_L.append(Sensitivity)

Specificity=rounder((tn) / (tn + fp))
Specificity_L.append(Specificity)

F1Score=rounder( (2 * tp) / (2 * tp+ fp + fn))
F1Score_L.append(F1Score)

Precision=rounder((tp) / (tp +fp))
Precision_L.append(Precision)

Negative_Predictive_Value=rounder((tn) / (tn + fn))
Negative_Predictive_Value_L.append(Negative_Predictive_Value)

False_Negative_Rate=rounder((fn) / (fn + tp))
False_Negative_Rate_L.append(False_Negative_Rate)

False_Positive_Rate=rounder((fp) / (fp + tn))
False_Positive_Rate_L.append(False_Positive_Rate)

False_Discovery_Rate=rounder((fp) / (fp + tp))
False_Discovery_Rate_L.append(False_Discovery_Rate)

False_Omission_Rate=rounder((fn) / (fn+ tn))
False_Omission_Rate_L.append(False_Omission_Rate)

print('Sensitivity:', Sensitivity)
print('Specificity:', Specificity)
print('F1 Score:', F1Score)
print('Precision:', Precision)
print('Negative Predictive Value:', Negative_Predictive_Value)
print('False Negative Rate:', False_Negative_Rate)
print('False Positive Rate:', False_Positive_Rate)
print('False Discovery Rate:', False_Discovery_Rate)
print('False Omission Rate:', False_Omission_Rate)

test_probabilities =model.predict_proba(X_test)[: , 1]
auc_score = roc_auc_score(y_test,test_probabilities)

fpr, tpr, thresholds = roc_curve(y_test, test_probabilities)
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label="+name+'(AUC = {:.2f})'.format(auc_score))
plt.plot([0, 1], [0, 1], linestyle='--', color='gray', label='Random Guess')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve - '+name)
plt.legend()
plt.show()

kf = KFold(n_splits=num_folds, shuffle=True,random_state=42)
cv_scores = cross_val_score(model, X_train, y_train, cv=kf, scoring='accuracy')
print(f"\n{num_folds}-Fold Cross-Validation Scores:")
print(cv_scores)

print(f"\nCross-Validation Accuracy Score: {max(cv_scores) * 100:.2f}%")
cv_accuracy_L.append(max(cv_scores)*100)

```

```

import xgboost as xgb
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import accuracy_score

# Define the parameters grid for XGBoost
params_xgb = {
    'n_estimators': [100, 200, 300, 400, 500],
    'max_depth': [1, 2, 3, 4, 5],
    'subsample': [0.5, 1],
    'colsample_bytree': [0.5, 1],
}

# Initialize RandomizedSearchCV for XGBoost
xgb_classifier = xgb.XGBClassifier()
xgb_random_search = RandomizedSearchCV(xgb_classifier, params_xgb, cv=20)

# Perform RandomizedSearchCV
xgb_random_search.fit(X_train, y_train)

# Get the best estimator from RandomizedSearchCV
best_xgb_classifier = xgb_random_search.best_estimator_

# Train the best XGBoost classifier
best_xgb_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_pred_xgb = best_xgb_classifier.predict(X_test)

# Calculate accuracy
xgb_acc = accuracy_score(y_test, y_pred_xgb)
print('XGBoost accuracy: {:.2f}%'.format(xgb_acc * 100))

# You can define your function fun to display evaluation metrics
fun(best_xgb_classifier, 'XGBoost Classifier', 3)

import matplotlib.pyplot as plt
colors = ['aqua', 'skyblue', 'orange', 'green', 'red', 'purple', 'pink']
plt.figure(figsize=(10, 6))
plt.barh(compare['Model'], compare['Accuracy'], color=colors)
plt.xlabel('Accuracy (%)')
plt.title('Model Comparison - Accuracy')
plt.xlim(0, 100)
for index, value in enumerate(compare['Accuracy']):
    plt.text(value, index, f'{value:.2f}', va='center', fontsize=10)

plt.show()

```


5.2 FRONTEND

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Employee Attrition</title>
  <!-- Include any required CSS files -->

  <style>
    body{
      background-image: url('https://www.employeeecycle.com/wp-
content/uploads/2022/11/PeopleAnalytics-403x224.jpg');
      background-repeat: no-repeat;
      background-size: cover;
      text-decoration: solid;

    }
    h1{
      margin-left:20%;
      color:black;
    }
    label{
      color:black;
      font-size: medium;
    }
    button{
      width:80px;
      height:40px;
      background-color: green;
      font-size: medium;
      margin-bottom: 1%;
    }
  </style>
</head>
<body>
  <header>
    <h1>Employee Attrition Predictor</h1><br><br>
  </header>
  <main>
    <form action="/employeePredict" id="form1" method="POST">
      <table>
        <th>
```

```
<div style="float:left;margin-left: 200px;text-align:center;">
<!-- Include input fields to gather student information -->
<label for="DailyRate">Daily Rate</label>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~
<input type="number" min="0" max="1499" name="DailyRate" required><br><br>

<label for="dfh">Distance From Home</label>
<input type="number" id="dfh" min="0" max="28" name="dfh" required><br><br>

<label for="EducationField">Education Field</label>
<select name="EducationField" class="first" required="required">
    <option value="1">Life Sciences</option>
    <option value="3">Medical</option>
    <option value="4">other</option>
    <option value="5">Technical Degree</option>
    <option value="2">Marketing</option>
    <option value="0">Human Resources </option>
</select><br><br>

<label for="HourlyRate">Hourly Rate</label>
<input type="number" id="HourlyRate" min="0" max="100" name="HourlyRate"
required></input><br><br>

<label for="JobRole">Job Role</label>

<select name="JobRole" class="first" required="required">
    <option value="7">Sales Executive</option>
    <option value="6">Research Scientist</option>
    <option value="2">Laboratory Technician</option>
    <option value="0">Healthcare Representative</option>
    <option value="4">Manufacturing Director</option>
    <option value="1">Human Resources</option>
    <option value="3">Manager</option>
    <option value="5">Research Director</option>
    <option value="8">Sales Representative</option>
</select><br><br>

<label for="MonthlyIncome">Monthly Income</label>
<input type="number" id="MonthlyIncome" min="0" max="5993"
name="MonthlyIncome" required><br><br>
<label for="MonthlyRate">Monthly Rate</label>
<input type="number" id="MonthlyRate" min="0" max="20000" name="MonthlyRate"
required><br><br>


<label for="ncw">Number of companies Worked</label>
<input type="number" id="ncw" name="ncw" min="0" max="18" required><br><br>
</div>
</th>
<th>
```

```

        <option value="0">No</option>
<option value="1">Yes</option>
</select><br><br>
    <label for="psh">Percent salary Hike</label>
    <input type="number" id="psh" min="0" max="14" name="psh" required><br><br>
    <label for="twy">Total Working Years</label>
    <input type="number" id="twy" min="0" max="40" name="twy" required><br><br>
    <label for="yac">Years At Company</label><br>
    <input type="number" id="yac" min="0" max="40" name="yac" required><br><br>
    <label for="ycr">Years In Current Role</label><br>
    <input type="number" id="ycr" min="0" max="18" name="ycr" required><br><br>
    <label for="ycm">Years With Current Manager</label><br>
    <input type="number" id="ycm" min="0" max="17" name="ycm" required><br><br>
    <label for="age">Age</label>
    <input type="number" id="age" min="18" max="60" name="age" required><br><br><br>

    <button type="submit" value="Predict">Submit</button>
</div>
</th>
</table>
<p>{{res}}</p>
</form>
</main>

</body>
</html>

```

5.3 CONNECTION

App.py

```

from flask import Flask, render_template, request
import pickle
import numpy as np
app = Flask(__name__)
model = pickle.load(open('Employee_attrition.pkl', 'rb'))

@app.route("/")
def Home():
    return render_template('index.html')

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

def employeePredict():

    if request.method=='POST':

        print(" ..... ")
        age= int(request.form['age'])
        dfh= int(request.form['dfh'])
        mi=int(request.form['MonthlyIncome'])

```

```

hr= int(request.form['HourlyRate'])
mr =int( request.form['MonthlyRate'])
dr =int( request.form['DailyRate'])
ycm = int(request.form['ycm'])
ycr = int(request.form['ycr'])
yac=int(request.form['yac'])
psh=int(request.form['psh'])
ncw=int(request.form['ncw'])
twy=int(request.form['twy'])
ov=(request.form['OverTime'])
if ov=="1":
    v=1
else:
    v=0
ef=(request.form['EducationField'])
ef=int(ef)
jr=(request.form['JobRole'])
jr=int(jr)
x=[dr,dfh,ef,hr,jr,mi,mr,ncw,v,psh,twy,yac,ycr,ycm,age]
print(x)
result=model.predict([[dr,dfh,ef,hr,jr,mi,mr,ncw,v,psh,twy,yac,ycr,ycm,age]])[0]
print(result)
if result==1:
    return render_template('index.html',res="Employee discontinued.")
else:
    return render_template('index.html',res="employee continued")
if __name__=="__main__":
    app.run(debug=True)

```

6. RESULT ANALYSIS

Employee attrition prediction problem, statement comes under the classification type of machine learning. To solve such a classification problem there are multiple choices available. Comparative analysis is a study of choosing the best algorithm for the problem statement in this section we illustrate the results of models based on their accuracy, precision, recall, and F1 score. Here we are comparing between multiple classification algorithms like Random forest(RFA) , Decision tree ,Logistic regression(LR), Extra Tree Classifier, K-Nearest Neighbor, Gradient Boost Classifier, XG Boost Classifier over evaluation metrics like Accuracy, Precision, Recall, and F1 score.

This phase evaluated the qualities of the adopted models. The results of the decisions made in the prediction phase were collected, for each algorithm, in the relative “confusion matrix”. This is a matrix where the values predicted by the classifier are shown in the columns and the real values of each instance of the test-set are shown in rows. To proceed with the performance evaluation, we used the confusion matrix to derive a series of fundamental metrics to quantitatively express the efficiency of each algorithm

By comparative analysis of below 7 algorithms we came to a firm conclusion that Random Forest Algorithm has the best accuracy. Therefore XG Boost Classifier is used for model building for our system.

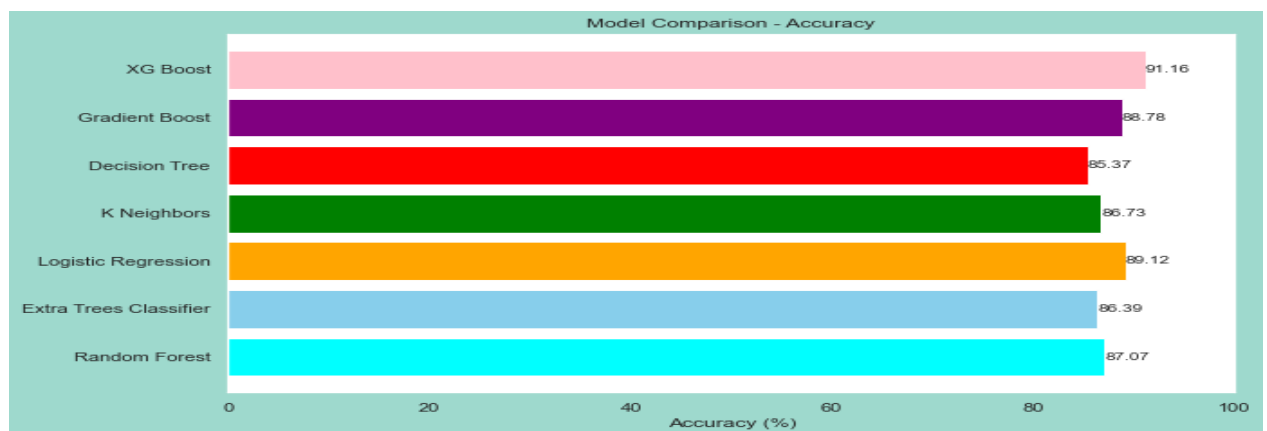


Figure:6.1 Result Analysis

Performance Evaluation:

All trained models were evaluated by measuring their accuracy, precision, recall and F1 score which are described below :

- Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision = $\frac{TP}{TP+FP}$
- Recall = $\frac{TP}{TP+FN}$
- F1 Score = $2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$

CONFUSION MATRIX

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known[10][11].

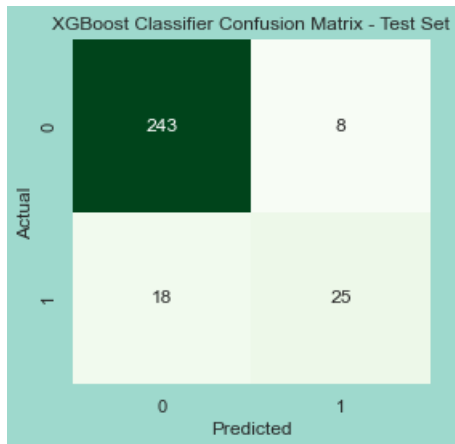


Fig. Confusion Matrix for XG Boost

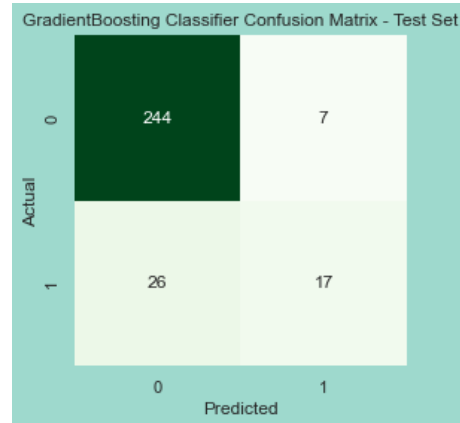


Fig. Confusion Matrix for Gradient Boost

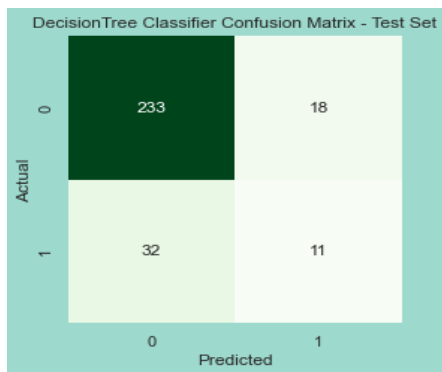


Fig. Confusion Matrix for Decision Tree

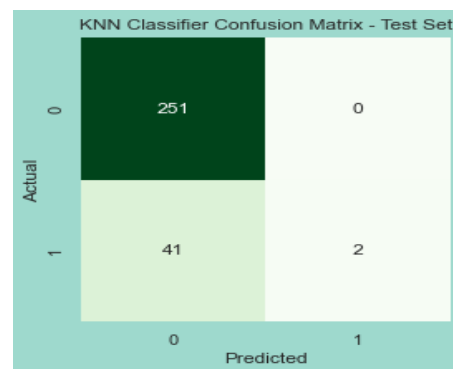


Fig. Confusion Matrix for KNN

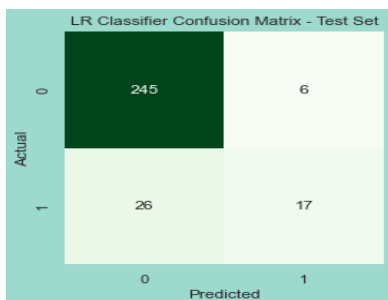


Fig. Confusion Matrix for LR

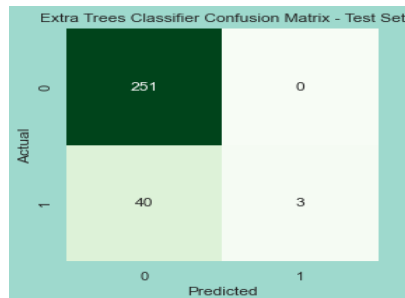


Fig. Confusion Matrix for Extra Trees

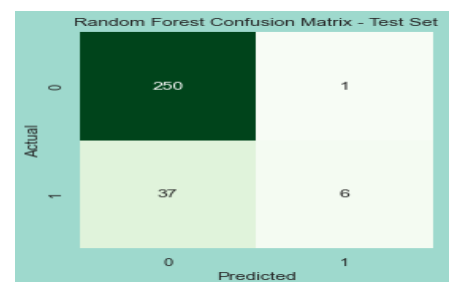
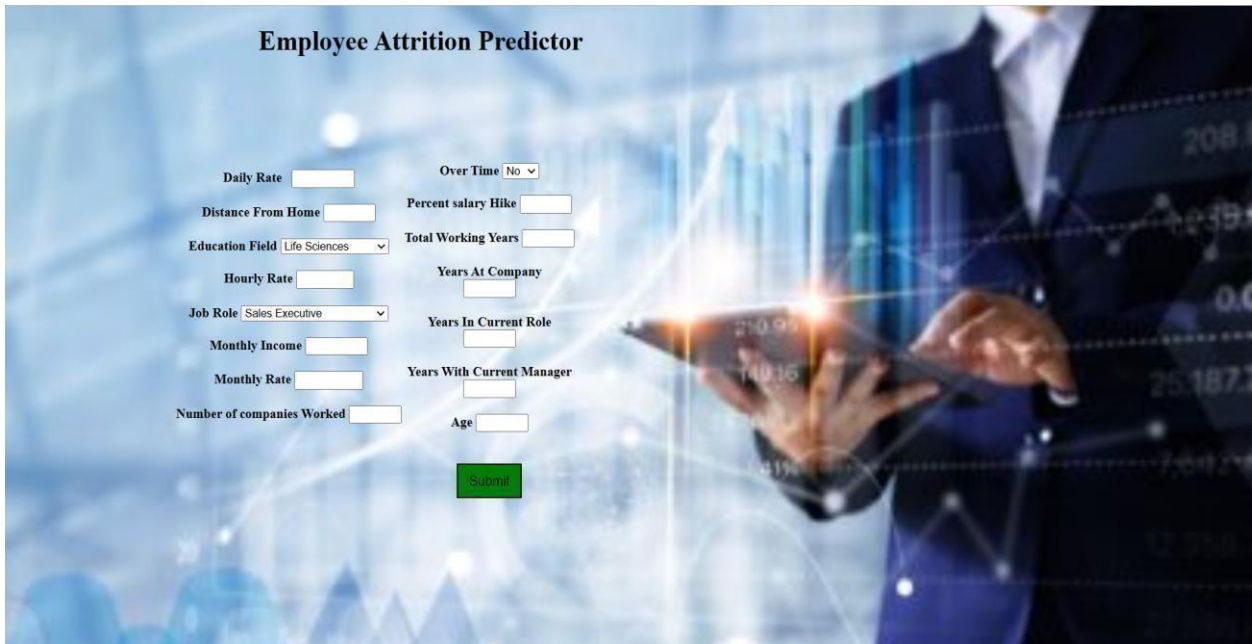


Fig. Confusion Matrix for Random Forest

7. OUTPUT SCREENS



The image shows the home screen of the 'Employee Attrition Predictor' application. The background features a blurred image of a person in a suit holding a tablet, with overlaid data visualizations like bar charts and line graphs. The form is titled 'Employee Attrition Predictor' and contains the following input fields:

Field	Value
Daily Rate	
Distance From Home	
Education Field	Life Sciences
Hourly Rate	
Job Role	Sales Executive
Monthly Income	
Monthly Rate	
Number of companies Worked	
Over Time	No
Percent salary Hike	
Total Working Years	
Years At Company	
Years In Current Role	
Years With Current Manager	
Age	

A green 'Submit' button is located at the bottom center of the form.

Figure:7.1 Home Screen

Figure 7.1 is the home screen of our project where we have to give values for the attributes in the above image.



The image shows the 'Employee Attrition Predictor' application after input has been provided. The background and title are the same as in Figure 7.1. The input fields now contain the following values:

Field	Value
Daily Rate	624
Distance From Home	0
Education Field	Life Sciences
Hourly Rate	64
Job Role	Sales Executive
Monthly Income	809
Monthly Rate	999
Number of companies Worked	8
Over Time	Yes
Percent salary Hike	0
Total Working Years	8
Years At Company	6
Years In Current Role	4
Years With Current Manager	5
Age	23

The green 'Submit' button remains at the bottom center.

Figure:7.2 After giving input

Figure 7.2, this is screen appears after giving input to fields in home screen

Employee Attrition Predictor

Daily Rate Over Time

 Distance From Home Percent salary Hike

 Education Field Total Working Years

 Hourly Rate Years At Company

 Job Role Years In Current Role

 Monthly Income Years With Current Manager

 Monthly Rate

 Number of companies Worked Age

Employee discontinued.

Figure:7.3 Evaluation of employee attrition for discontinuation

Figure 7.3 is the prediction of employee attrition. The value of result is ‘1’ which is labeled as ‘YES’ for attrition variable.

Employee Attrition Predictor

Daily Rate Over Time

 Distance From Home Percent salary Hike

 Education Field Total Working Years

 Hourly Rate Years At Company

 Job Role Years In Current Role

 Monthly Income Years With Current Manager

 Monthly Rate

 Number of companies Worked Age

employee continued

Figure:7.4 Evaluation of employee for continuation

Figure 7.4 is the prediction of employee attrition. The value of result is ‘0’ which is labeled as ‘No’ for attrition variable.

8. CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

After assessing the execution of four classification models, a significant finding was that if feature reduction for prediction is appropriately conducted, the accuracy rate of the classification models always be better compared to classification with feature selection. In particular, the XG Boost classifier with feature reduction achieved an accuracy score of 91.1%, while the Logistic Regression achieved 89%. XG Boost Classifier model giving best classification for True positives and True negatives data. The methods described in the paper for analyzing and categorizing data can form a basis for improving data-driven decision-making processes. These techniques can unlock new insights from data and help organizations improve their operations. Implementation of these methods can also contribute to a positive work culture and improve an organization's reputation in their respective industry.

8.2 FUTURE SCOPE

To develop more accuracy using machine learning algorithms and advanced techniques. The work can extend and improved for the automation of employee attrition prediction by using advanced techniques.

9. BIBLIOGRAPHY

1. <https://ieeexplore.ieee.org/document/9033784>
2. Srivastava, Devesh Kumar, and Priyanka Nair. "Employee attrition analysis using predictivetechniques." International Conference on Information and Communication Technology for Intelligent Systems. Springer, Cham, 2017.
3. S. S. Gavankar and S. D. Sawarkar, "Eager decision tree," 2017 2nd International Conference forConvergence in Technology (I2CT), Mumbai, 2017, pp. 837-840.
4. Safavian, S.R. Landgrebe. D, "A survey of decision tree classifier methodology", IEEETransactions on Systems, Man, And Cybernetics, Vol. 21, No. 3, May-June 1991.
5. Shmilovici A. (2009) Support Vector Machines. In: Maimon O., Rokach L. (eds) Data Mining andKnowledge DiscoveryHandbook. Springer, Boston.
6. Setiawan, I., et al. "HR analytics: Employee attrition analysisusing logistic regression." IOP Conference Series: MaterialsScience and Engineering. Vol. 830. No. 3. IOP Publishing, 2020
7. Schober, Patrick MD, PhD, MMedStat*; Vetter, Thomas R. MD, MPH†. Logistic Regression inMedical Research. Anesthesia & Analgesia 132(2):p 365-366, February 2021.| DOI: 10.1213/ANE.0000000000005247
8. Jayalekshmi J, Tessy Mathew, "Facial Expression Recognition and Emotion Classification System for Sentiment Analysis", 2017 5 Authorized licensed use limited to: University College London. Downloaded on May 23,2020 at 00:07:22 UTC from IEEE Xplore. Restrictions apply. International Conference on Networks & Advances in Computational Technologies (NetACT) |20-22 July 2017| Trivandrum.
9. Isabelle Guyon, Andre Elisseeff, "An Introduction to Variableand Feature Selection", Journal ofMachine Learning Research 3 (2003) 1157-1182.
10. Ilan Reinstein, "Random Forest(r), Explained", kdnuggets.com,October 2017[Online].Available:<https://www.kdnuggets.com/2017/10/randomforests-explained.html>
11. http://scikitlearn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
12. http://scikitlearn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html
13. Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer, "SMOTE: Synthetic Minority Over- sampling Technique", Journal of Artificial Intelligence Research 16(2002), 321 – 357
14. Pavan Subhash, "IBM HR Analytics Employee Attrition & Performance", [www.kaggle.com,2016\[Online\].Available:https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset](https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset)
15. Sperandei S. Understanding logistic regression analysis. Biochem Med (Zagreb). 2014 Feb15;24(1):12-8. doi: 10.11613/BM.2014.003. PMID: 24627710; PMCID: PMC3936971.

