FACULTY OF COMPUTING
UNIVERSITI TEKNOLOGI MALAYSIA

MASTER OF SCIENCE (DATA SCIENCE)
SEMESTER 1 2024/2025
ADVANCED ANALYTICS FOR DATA SCIENCE (MCSD2213)

PROJECT GROUP

TITLE: Data-Driven Insights into Employee Attrition: Machine Learning Approaches for Workforce Stability

LECTURER:
DR. NOR HAIZAN MOHAMED RADZI

BY:

NAME	STUDENT ID
NURAMIRA SHAFINAZ BINTI ZULAILEE	MCS231031
SOLEHAH NAJIIHAH BINTI ABD JAMAL	MCS231035
UMMI FARIHAH BINTI ABD WAHID	MCS231032

SUBMIT:
23 JANUARY 2025

Table of Contents

List of Figures

List of Tables

1.0 Executive Summary

Employee attrition is a critical challenge for organizations that will impact operational efficiency, financial performance, and workforce morale. High turnover rates will increase the costs associated with recruitment, onboarding, and training and lead to the loss of institutional knowledge and disruptions to the team dynamics. In response to these challenges, data-driven approaches are increasingly being used to analyse and predict attrition, enabling proactive interventions to enhance employee retention.

This study leverages a comprehensive dataset containing 1,470 records and 35 attributes, covering demographic, organizational, and satisfaction-related variables. The key focus is the Attrition variable, which indicates whether an employee has left the organization. The dataset captures a wide range of factors influencing attrition, such as job satisfaction, work-life balance, career progression opportunities, and compensation levels.

By analysing this dataset, the study aims to identify the key factors influencing attrition which will uncover the demographic, organizational, and job satisfaction related to the variables that significantly impact an employee's decision to leave. Furthermore, it can develop predictive models with the use of advanced machine learning techniques to predict employees at risk of attrition, providing actionable insights for human resource management. with the analysis, it also can provide insights that enable organizations to design targeted interventions aimed at reducing turnover, improving employee satisfactions and optimizing workforce stability.

The result from this will have practical implications for organizations. By understanding attrition drivers, businesses can implement proactive strategies, such as improving work-life balance, offering competitive compensation, and fostering career development opportunities. Predictive models further enable HR managers to focus retention efforts on high-risk employees, ensuring the efficient allocation of resources.

In conclusion, this study underscores the value of leveraging data analytics to address employee attrition. By transforming raw data into actionable insights, organizations can reduce turnover rates, enhance employee engagement, and build a resilient workforce, ultimately contributing to long-term success and sustainability.

1.1 Introduction

Employee turnover is considered an important problem for organizations around the world as it will affect their performance and productivity. Apart from directly affecting business operations, high employee turnover increases the cost of attracting and training other employees most of whom leave in the first few months including costs attributed to fluctuating workforce. In addition, business organizations suffer from loss of expertise and experience which are problems in terms of organizational memory affecting organizational performance and team cohesiveness consequently poor worker morale among the retained staff.

In today's competitive business environment, retaining top talent has become increasingly challenging. Factors influencing attrition are often multifaceted, ranging from individual employee characteristics, such as job satisfaction and career development opportunities, to organizational dynamics, such as workplace culture and management practices. Understanding these factors is essential for organizations aiming to build a stable and engaged workforce. This study is based on a comprehensive dataset that captures various dimensions of employee profiles, including demographic information (e.g., age, gender, marital status), work-related characteristics (e.g., department, job role, years at the company), and satisfaction metrics (e.g., job satisfaction, work-life balance).

Organizations can uncover valuable insights into the underlying by analysing the dataset that causes employee turnover. For instance, trends might be revealing that employees in specific roles of departments are more likely to leave due to job dissatisfaction or limited career advancement opportunities. Similarly, factors like excessive workload, lack of work-life balance, or inadequate compensation might emerge as significant contributors to attrition. Predictive models built on such data can identify employees at high risk of attrition, enabling HR teams to intervene effectively. For instance, targeted retention strategies such as offering tailored training programs, improving workplace policies, or addressing employee grievances can be implemented to address the root causes of dissatisfaction.

To conclude, analysing the data related to employee attrition that is provided in this data set is not only a technical process but a vital business necessity. It draws the connection between quantitative and qualitative analytic specifications and tangible organizational applications to improve organizational stability and work productivity. The outcomes will enable organizations to improve on the management of workforce, and employee satisfaction

and develop a reliable workforce that delivers improved organizational performance in the long run.

1.2 Problem background

Employee attrition is a persistent and costly issue for organizations, directly impacting financial performance, operational efficiency, and overall workforce stability. High turnover rates lead to significant expenses related to recruitment, onboarding, and training, as well as indirect costs stemming from lost productivity, decreased morale, and the disruption of team dynamics. For many organizations, this challenge is exacerbated by the difficulty in identifying the root causes of attrition and implementing effective retention strategies.

Despite efforts to address employee attrition, many organizations struggle to predict and mitigate turnover effectively due to the complexity and multifaceted nature of the factors that influence it. One significant factor is job satisfaction, where dissatisfaction with role responsibilities, lack of recognition, or limited career advancement opportunities can prompt employees to seek alternatives. Additionally, poor work-life balance and excessive workloads often lead to burnout, further increasing the likelihood of attrition. Organizational factors, such as ineffective leadership, inadequate compensation, and insufficient training or development opportunities, also contribute significantly to turnover. Moreover, demographic and personal factors, including proximity to the workplace and individual aspirations, play an essential role in employees' decisions to leave. Addressing these diverse and interrelated issues requires a deeper understanding and data-driven strategies to effectively reduce attrition rates.

In conclusion, organizations collect vast amounts of data on their employees, but these data often remain underutilized, leaving managers without actionable insights. Predictive analytics, when applied to employee data, offers the potential to uncover patterns and relationships that traditional analysis may miss. By identifying employees at risk of leaving and understanding the key drivers of attrition, organizations can implement proactive measures to address underlying issues and improve retention. This will highlight the need for a predictive framework that not only identifies patterns of attrition but also provides actionable insights to inform data-driven decision-making.

1.3 Objectives

The objective of this project is:

a) To identify key factors influencing attrition to uncover the primary factors that contribute to employee turnover.
b) To develop a robust predictive model for employee attrition using Logistic Regression, XGBoost, and SVM models.
c) To compare the performance of Logistic Regression, XGBoost, and SVM models.

2.0 Dataset Description

This project will make use a dataset obtained from Kaggle that contains the performance and employee attrition, fictionally created by IBM data scientists. The dataset has a structure of 1470 rows and 35 columns. Generally, it has the employee personal details such as age, gender, relationship satisfaction, job role and the job-related information such as job role, performance rating, total working years etc.

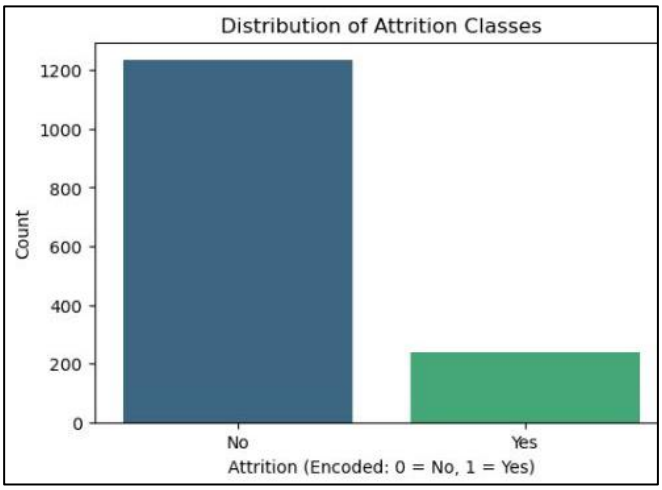
Table 1: Attributes and their description.

Attributes	Description
Age	The age of the employee
Attrition	Status attrition of the employee
BusinessTravel	Either the employee travel rarely or frequently
DailyRate	The daily rate working
Department	The department of the employee works in.
DistanceFromHome	The distance of office from home
Education	Education level of the employee
EducationField	Education field of the employee
EmployeeCount	The employee counts
EmployeeNumber	The employee numbers
EnvironmentSatisfaction	The level of environment satisfaction.
Gender	Gender of the employee
HourlyRate	The hourly rate of working.
JobInvolvement	The level involvement of the job
JobLevel	The job level of the employee
JobRole	The job role of the employee
JobSatisfaction	The level of job satisfaction
MaritalStatus	The marital status of the employee
MonthlyIncome	The monthly income of the employee
MonthlyRate	The monthly rate of the employee
NumCompaniesWorked	Number of companies he/she has worked
Over18	Either the employee is over 18 or not
OverTime	Either if the employee work overtime or not
PercentSalaryHike	The percentage of salary hike
PerformanceRating	The performance rating of the employee
RelationshipSatisfaction	The relationship satisfaction of the employee
StandardHours	Standard hours working
StockOptionLevel	The stock option level
TotalWorkingYears	Total working years he/she has

TrainingTimesLastYear	Total training times of last year
WorkLifeBalance	The level of work-life balance of the employee
YearsAtCompany	Total years the employee at the company
YearsInCurrentRole	Total years the employee in the current role
YearsSinceLastPromotion	Total years since last promotion
YearsWithCurrManager	Total years with current manager

2.1 Exploratory Data Analysis (EDA)

Table 1: List of Output Exploratory Data Analysis (EDA) I

A. General Employee Demographics	**Explanation**
 Figure 1: Distribution of Attrition Classes	In this figure, the data shows the number of employees who have either resigned from the organization or have not resign from the organization employees. More than 80 percent of the organization's employees have not experienced turnover, while about 20 percent have left the organization. This imbalance is a common problem seen in data sciences when dealing with predictive models and can often require that the minority class (attrition) be addressed using techniques such as oversampling or under sampling to provide a more accurate prediction and analysis. This graph is very helpful in understanding the dependent variable related to the attrition prediction task.

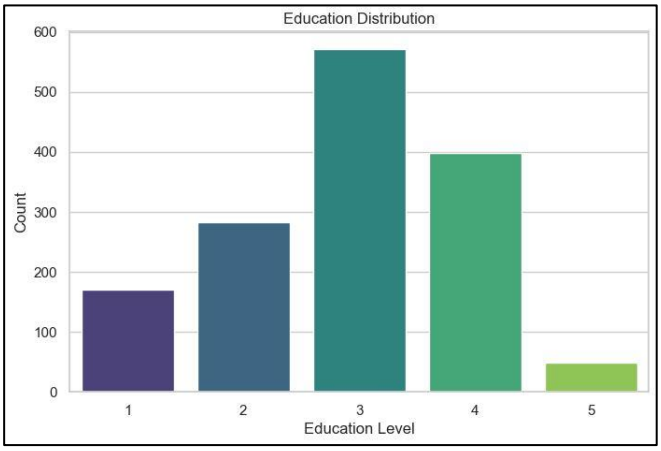


Figure 2: Education Distribution

The bar chart below indicates the number of the employees according to the level of education. The maximum employees are having the education level of 3, the second level most employees are having the education level of 4 but the minimum employees are having the education level of 5. This distribution also reveals the general educational profile in the employed labor market.

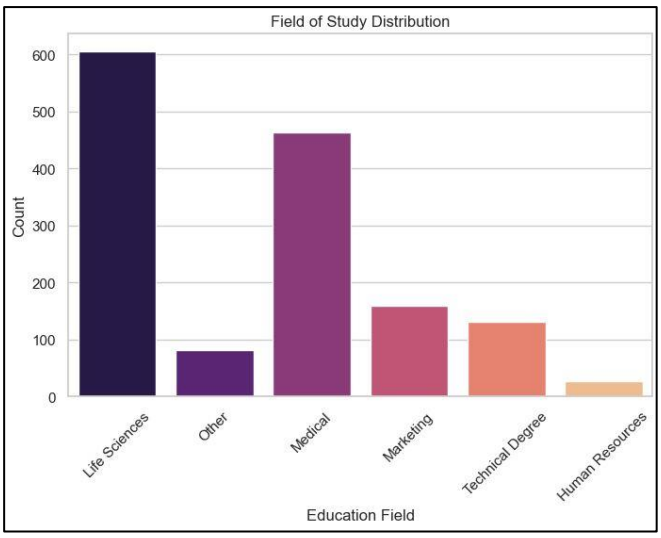


Figure 3: Field of Study Distribution

The following bar chart shows the number of employees according to their level of education. Medical category has the highest representation of study in Life Sciences with other categories having other fields Life sciences, Marketing, Technical Degrees, and Human Resources having the lowest. It will also supplement the distribution of education level to reflect workforce diversity.

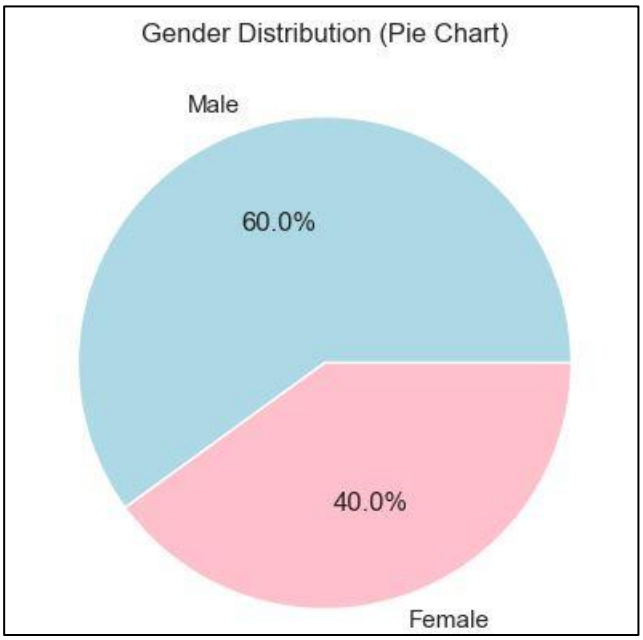


Figure 4: Gender Distribution

It shall be recalled that the workforce gender distribution is depicted in the pie chart below and the male gender made up 60% and the female gender 40%. Such disparity could be due to gender representation patterns in particular sectors/positions within an organization and that affects diversity management plans.

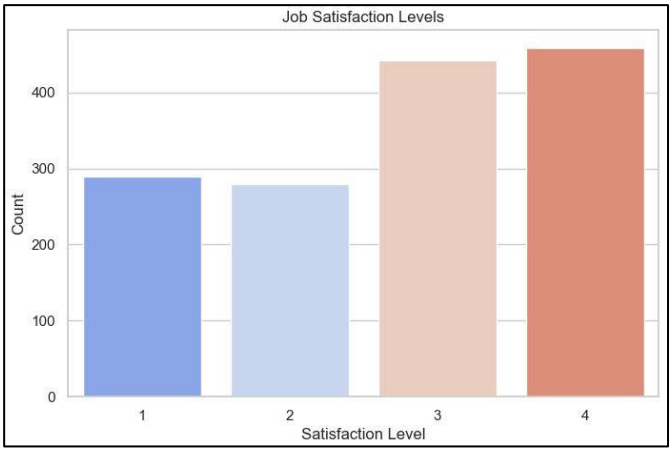


Figure 5: Job Satisfaction Levels

The following bar chart shows the number of employees that are satisfied, neutral and dissatisfied with their job. As with environment satisfaction, most of the respondents classify themselves as level 3 or level 4 in satisfaction, which indicates overall job satisfaction. Satisfaction level 1 and 2 represent a smaller but significant part of the workforce to indicate dissatisfaction areas.

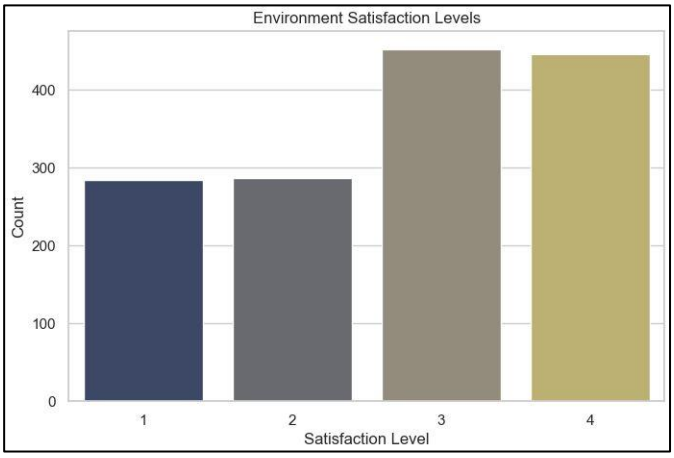


Figure 6: Environment Satisfaction Levels

The bar chart above shows the bar chart of the employee satisfaction level in terms of the work environment in a scale of 1 to 4. Employee ratings of the environment satisfaction level average to level 3 or 4 suggesting most employees have positive attitudes. Nonetheless, it is evident that a large number of employees are still at level 1 or 2 concerning satisfaction, which might be of concern for the organization.

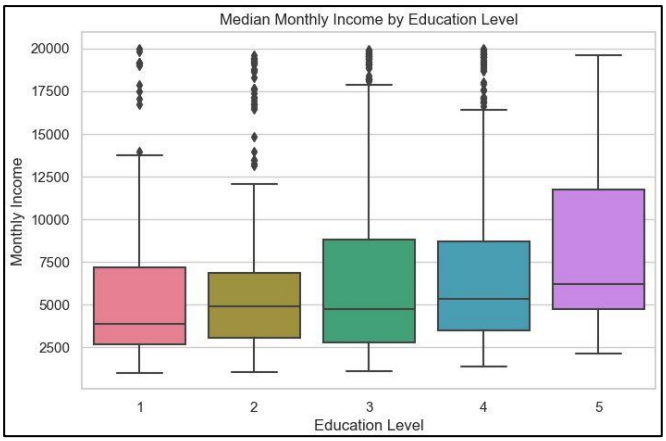
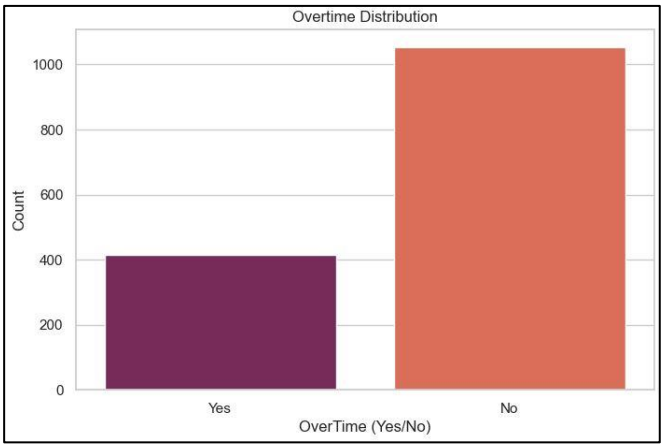


Figure 7: Median Monthly Income by Education Level

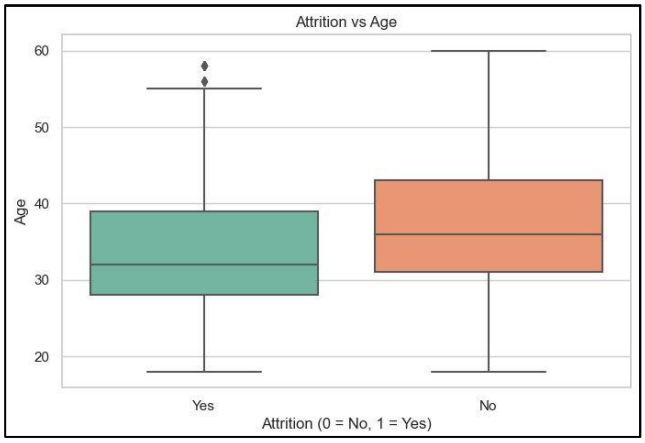
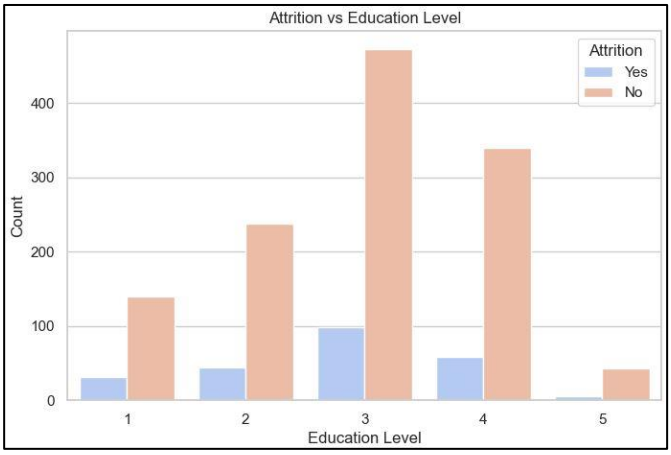
This box plot illustrates the median monthly income across different levels of education. The results show that the median income tends to increase with the education level: level 5 has the highest median and the biggest spread. This has the implication of positively linking educational attainment with the income that a person is likely to earn.



This type of bar chart shows the number of employees who works overtime and the number of those who does not. It is also clear from the figure that more than two-thirds of the employees do not work overtime. This fact is crucial for effective analysis of work-life balance and its impact on the employee turnover.

Figure 8: Overtime Distribution	

Table 2: List of Output Exploratory Data Analysis (EDA) II

B. Specific Employee Demographics Relation with Attrition	**Explanation**
 Figure 9: Attrition vs Age	This box plot refers to the differences in age among the employees who remained with the organization (attrition = No) and those who left (attrition = Yes). For the current employees, the mean age is lower than those of the former employees, which shows that the employees who left are relatively younger than those who remain in the company. Secondly, based on the interquartile range of ages for attrition = Yes, the distribution of ages for employees who leave has less variation in its variance. This calls for age as a possible predictor of the rate of employee turnover.
 Figure 10: Attrition vs Education Level	The bar chart looks at the attrition rates by education level. Employees with mid-level education (Level 3) have the highest count of attribute attrition = No, meaning they are not leaving the organization. But attrition rates are somewhat stable by education which may suggest that education cannot be the main predictor of turnover.

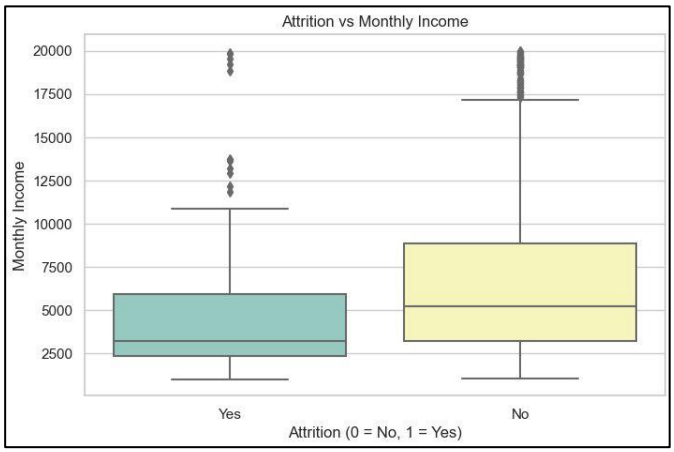


Figure 11: Attrition vs Monthly Income

This box plot depicts attrition by the level of monthly income. Those employees who leave the organization have comparatively lower monthly income; the median income for the attrition = Yes is less than the median for the attrition = No. Moreover, the range of incomes are even more restricted among the employees who have left. This finding underlines the fact of fair compensation as a crucial factor that helps to prevent turnover.

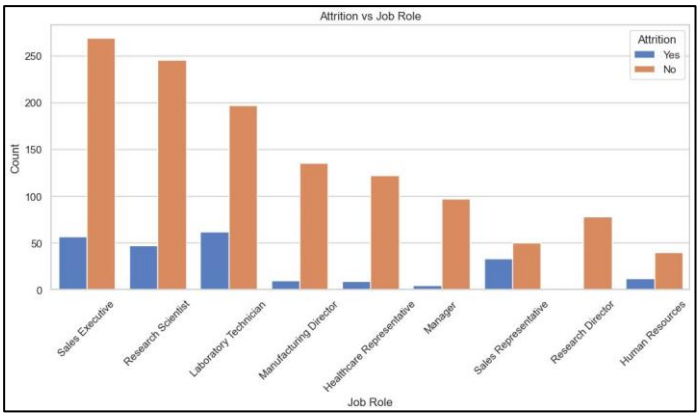
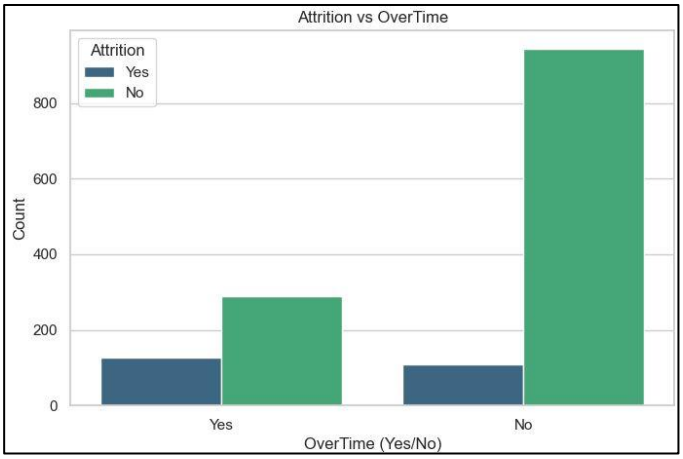
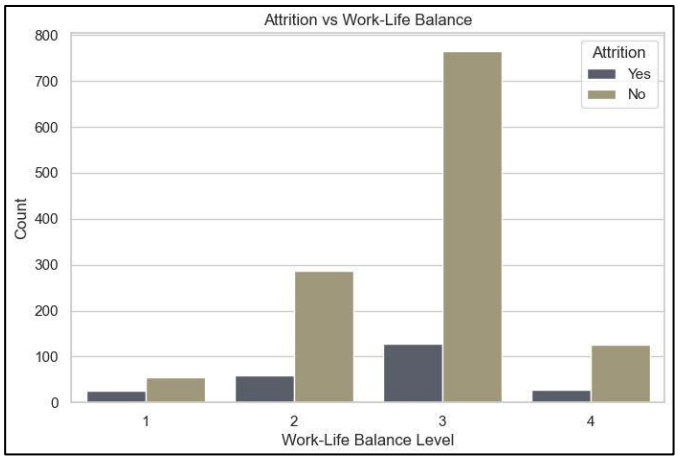
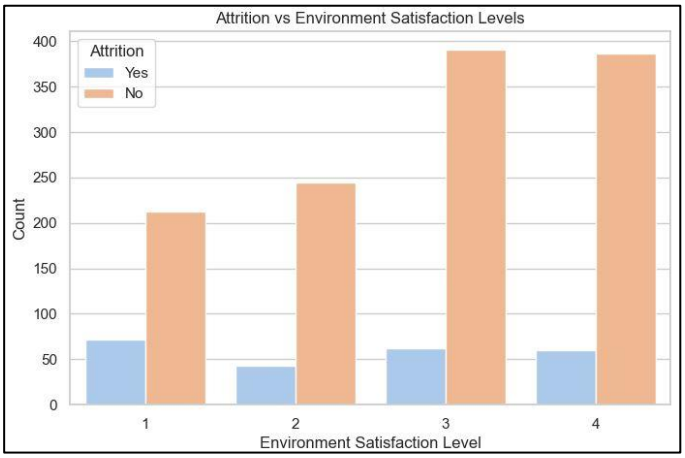


Figure 12: Attrition vs Job Role

This figure compares attrition by the position of the job in the organization. In the current dataset, the sales executives and research scientists have the highest attrition rates signified by the higher count of attrition = Yes. On the other hand, positions like human resource, research director and manager depict a trend of low turnover rates. This implies that job des, stress or job satisfaction may be some of the correlates to attrition if certain job roles are more stressful or have lesser job satisfaction.



This chart shows the attrition rate for the employees who had worked overtime compared to the one who did not. So it can be concluded that number of Employees changing the organization = Yes is higher for those who work on overtime basis. On the other hand, workers who do not work in overtime have a poor attrition rate when compared to the work's. This implies that overtime could have negative effects on the workers' satisfaction levels and

Figure 13: Attrition vs Overtime	therefore lead to increased turnover levels.
 Figure 14: Attrition vs Work-Life Balance	The following bar chart shows the trend of attrition based on the work-life balance of employees. It is evident that the organizations having higher work-life balance levels (Level 3 and 4) have lesser number of attrition = No. On the other hand, the employees that endured level 1 work-life balance have a higher likelihood of attrition. This goes a long way in supporting the fact that organizations need to embrace work-life balance in order to curb turnover.
 Figure 15:Attrition vs Environment Satisfaction Levels	This figure aims to compare attrition with environment satisfaction levels of employees. The chart above reveal that the percentage of EMPLOYEE ATTRITION = No rises as the environmental satisfaction level rises. For example, Level 4 of the environment satisfaction is associated with the lowest level of attrition, and Level 1, the highest levels of attrition. Therefore, this study implies that a favourable organisational climate has a positive outcome in keeping employees.

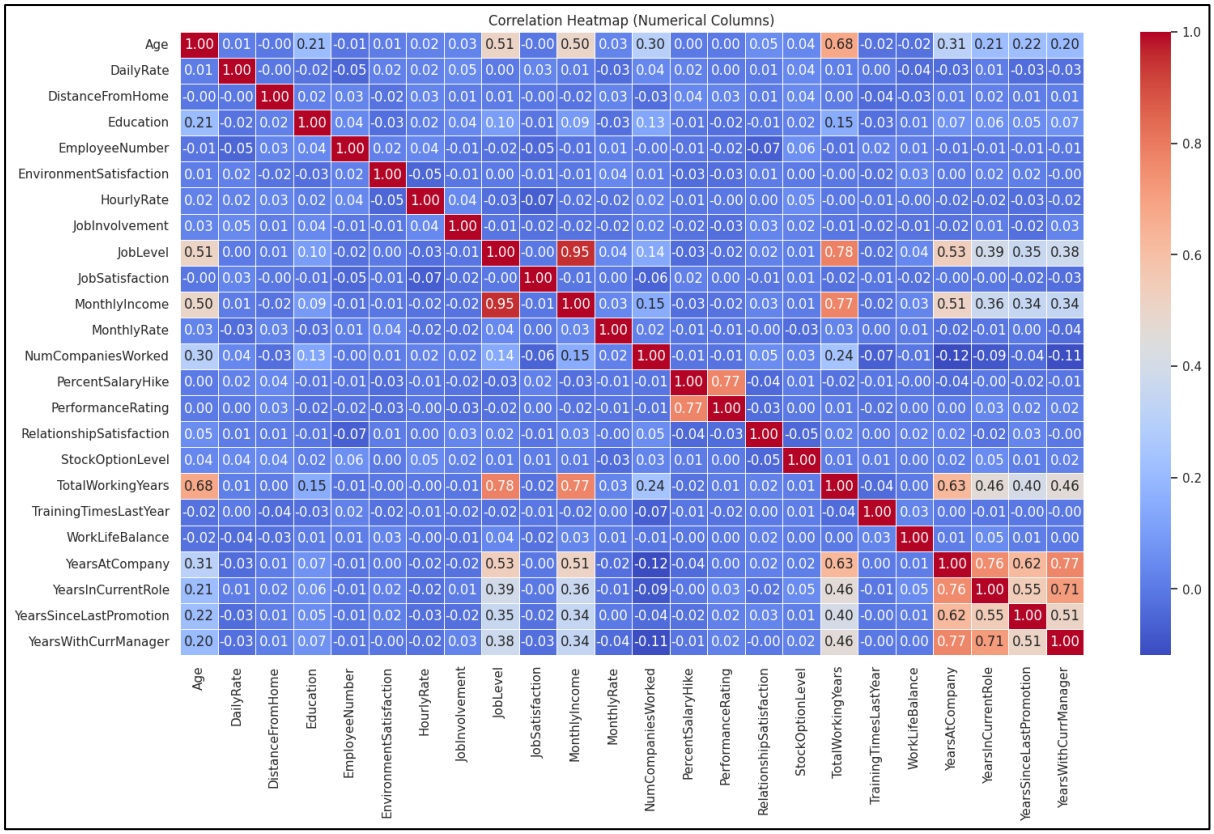


Figure 16: Correlation Heatmap of The Dataset

The correlation heatmap in figure 16 shows the inter-correlation among all the numerical variables contained in the data set. Values closer to 1 are red and depicted as positive and strong while values closer to -1 are blue and are considered strong and negative. This means that by increasing one's job levels, for instance in `MonthlyIncome', the results will be an increase in monthly income. Moreover, as the results pointed out, both `YearsAtCompany` and `YearsInCurrentRole` are positively related, suggesting that the longer the period of an employee's time at the company, the longer they have been in their current position. At the same time, the majority of variables reveal low levels of interdependence, which is supported by the large use of light blue tones. This heatmap is useful for showing associations between features and can be helpful in the feature selection and feature interpretation stages of model construction.

3.0 Methodology

3.1 Overview

This research focuses on utilising the supervised machine learning methods, such as XGBoost model, Logistic Regression model, and Support Vector Machine (SVM) model to predict the employee attrition based on various features in the dataset. Besides that, the dataset contains demographic, word-related, and satisfaction-level attributes that are preprocesses to handle the missing values, encode categorical variables, and standardized numerical features. To reduce class imbalance, SMOTE is used to perform on the dataset while splitting the data into train and test data. The performance of the models is measured by accuracy, precision, recall, F1-score, and AUC-ROC for the purpose of prediction. This project methodology approach seeks to establish the ability and reliability of the machine learning models in the analysis of the data on the employee's attrition, and the issues related to the analysis of imbalanced data set.

I. Problem Formulation
II. Data Collection
III. Data Pre-processing
IV. Modelling
V. Performance Validation and Evaluation

The details of the research framework for this study are shown in the Figure below.

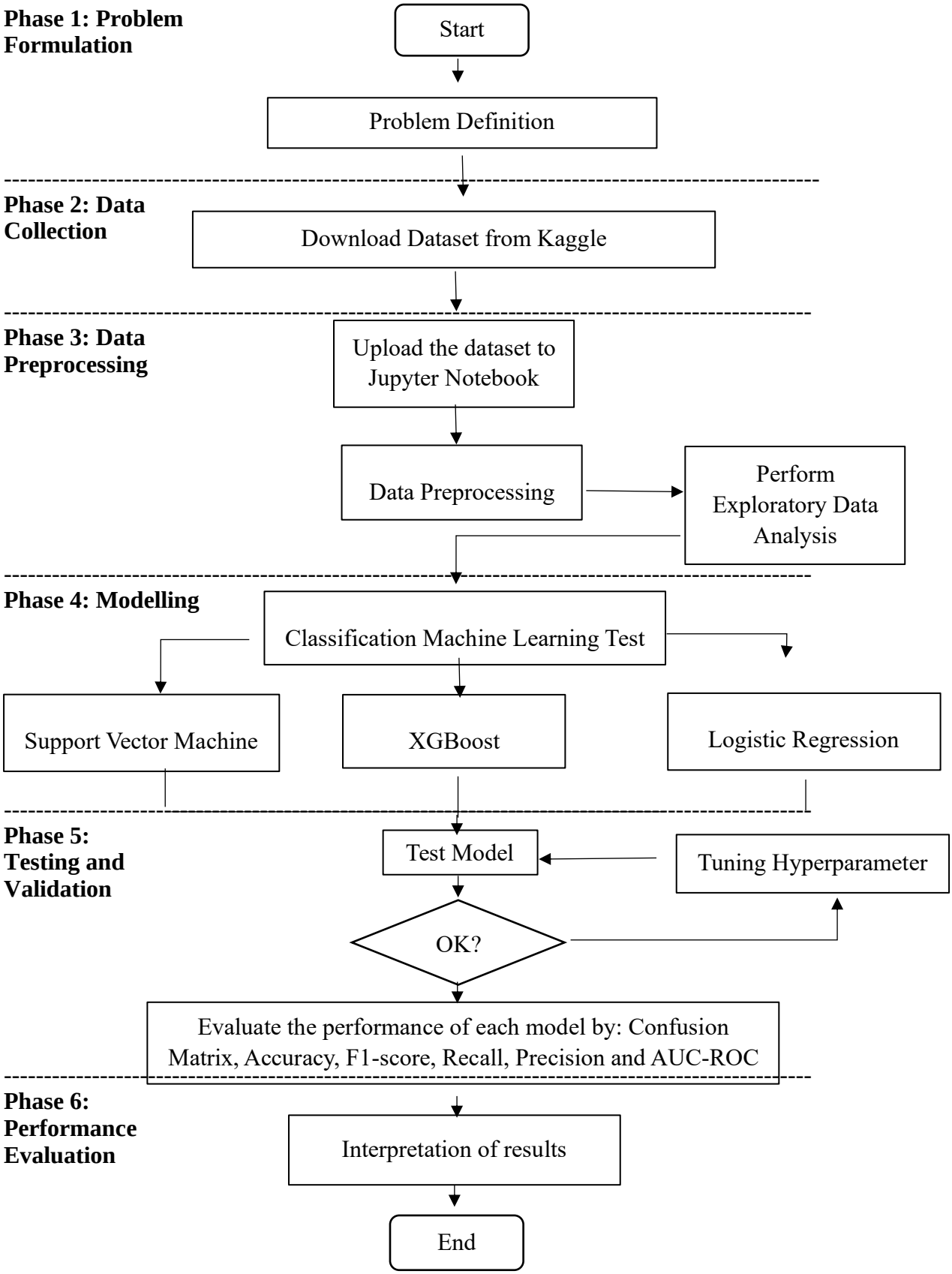


Figure 17: Research Framework Methodology

3.2 Data Preprocessing

```
# Display basic information about the dataset
print("Dataset Info:")
employee_df.info()
```

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       1470 non-null   int64
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64
 6   Education                 1470 non-null   int64
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64
 9   EmployeeNumber            1470 non-null   int64
 10  EnvironmentSatisfaction   1470 non-null   int64
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64
 13  JobInvolvement            1470 non-null   int64
 14  JobLevel                  1470 non-null   int64
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64
 19  MonthlyRate               1470 non-null   int64
 20  NumCompaniesWorked        1470 non-null   int64
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64
 24  PerformanceRating         1470 non-null   int64
 25  RelationshipSatisfaction  1470 non-null   int64
 26  StandardHours             1470 non-null   int64
 27  StockOptionLevel          1470 non-null   int64
 28  TotalWorkingYears         1470 non-null   int64
 29  TrainingTimesLastYear     1470 non-null   int64
 30  WorkLifeBalance           1470 non-null   int64
 31  YearsAtCompany            1470 non-null   int64
 32  YearsInCurrentRole        1470 non-null   int64
 33  YearsSinceLastPromotion   1470 non-null   int64
 34  YearsWithCurrManager      1470 non-null   int64
dtypes: int64(26), object(9)
```

Figure 18: Dataset Information Summary

The general overview of the dataset is shown in Figure 18. The dataset has 1470 records and 35 features, in which several features are categorical, and the others are numerical. Every column contains non-null data, which is shown by the non-null counts equal to the total entries. The first check helps to determine that there are no missing values in the dataset, and it also gives information about the data types, such as 26 continuous variables and 9 discrete variables. It is useful knowledge for deciding on the preprocessing of data that is necessary for machine learning modelling.

```
# 1. Handle Missing Values

# Check for missing values
print("Missing Values Per Column:")
print(employee_df.isnull().sum())
```

```
Missing Values Per Column:
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
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

Figure 19: Missing Values Per Column

As shown in the Figure 19 above, the missing value analysis was performed for all the columns of the given dataset. All the columns have zero missing values, which means that we do not need to impute any data in the dataset. This is helpful in data preprocessing, as there is no need for other methods like dealing with missing values since they are dealt with here hence makes the next step of feature selection, encoding and normalization to be easily tackled.

```
# 2. Drop Irrelevant or Redundant Columns

# Identify columns with unique or constant values
print("\nUnique Value Count Per Column:")
employee_df.nunique()
```

```
Unique Value Count Per Column:
Age                         43
Attrition                    2
BusinessTravel               3
DailyRate                  886
Department                   3
DistanceFromHome            29
Education                    5
EducationField               6
EmployeeCount                1
EmployeeNumber            1470
EnvironmentSatisfaction      4
Gender                       2
HourlyRate                  71
JobInvolvement               4
JobLevel                     5
JobRole                      9
JobSatisfaction              4
MaritalStatus                3
MonthlyIncome             1349
MonthlyRate               1427
NumCompaniesWorked          10
Over18                       1
OverTime                     2
PercentSalaryHike           15
PerformanceRating            2
RelationshipSatisfaction     4
StandardHours                1
StockOptionLevel             4
TotalWorkingYears           40
TrainingTimesLastYear        7
WorkLifeBalance              4
YearsAtCompany              37
YearsInCurrentRole          19
YearsSinceLastPromotion     16
YearsWithCurrManager        18
dtype: int64
```

Figure 20: Unique Value Count Per Column

In figure 20, there is a breakdown of the distribution of unique values for each column in the dataset. In this case, the examination shows that some of the necessary columns, including EmployeeCount, Over18, and StandardHours, have only one unique value, which may not be useful in predictive analysis. On the other hand, we have columns such as MonthlyIncome, where the number of unique values is again relatively high, representing continuous data and EmployeeNumber which is identifying data. This analysis is important for filtering out the unnecessary columns that needs to be eliminated, thereby bringing the features used into the model into perspective.

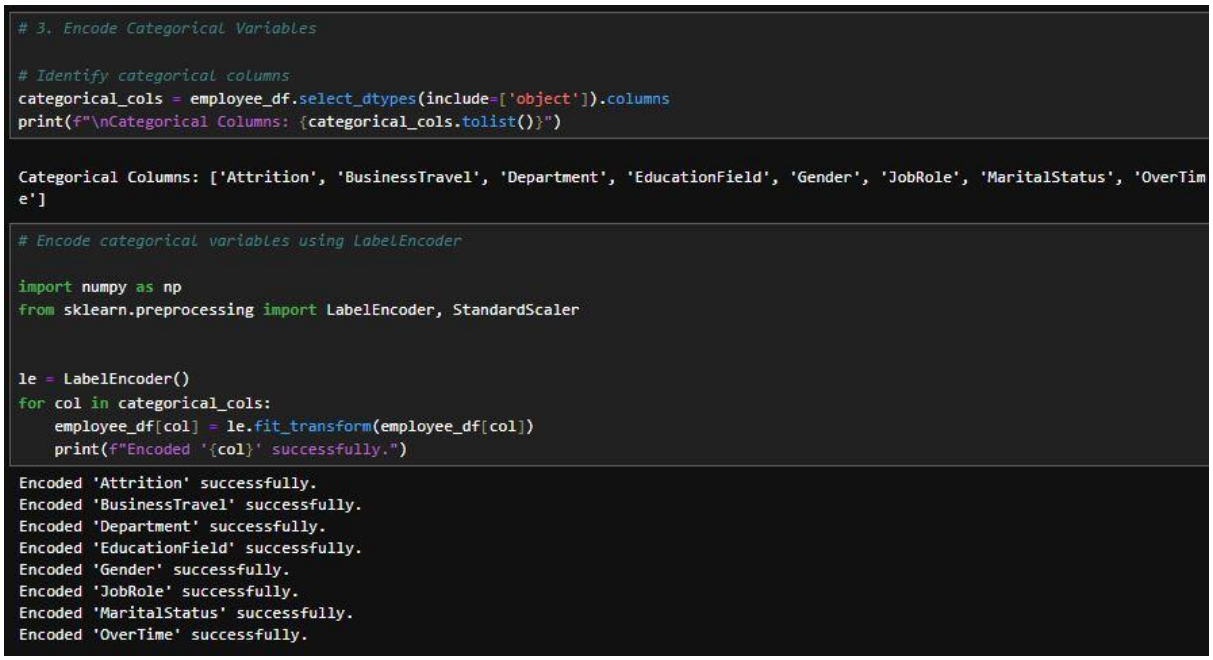
```
# Since unique columns that are irrelevant are:'EmployeeCount', 'Over18', 'StandardHours
# Drop columns that have constant values or are irrelevant
columns_to_drop = ['EmployeeCount', 'Over18', 'StandardHours']
employee_df.drop(columns=columns_to_drop, axis=1, inplace=True)
print(f"\nColumns after dropping irrelevant ones: {employee_df.columns.tolist()}")
```

```
Columns after dropping irrelevant ones: ['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department', 'DistanceFromHome', 'Educa
tion', 'EducationField', 'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate', 'JobInvolvement', 'JobLevel', 'JobRol
e', 'JobSatisfaction', 'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked', 'OverTime', 'PercentSalaryHike', 'Perf
ormanceRating', 'RelationshipSatisfaction', 'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance', 'Ye
arsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion', 'YearsWithCurrManager']
```

```
# Now we will split the data for 2 purpose, one for EDA (before encoded and normalize) and the other one is for Supervised Machine L
raw_employee_df = employee_df.copy()
```

Figure 21: Columns After Dropping Irrelevant Features

Figure 21 shows the columns of the dataset after feature selection where there were noisy features such as EmployeeCount, Over18, and StandardHours which have constant values. This step helps in filtering out unwanted noise from the data and concentrates on the features which show variations. The resulting dataset only includes the testing fields that can predict the employee attrition, which makes the following modeling steps run more efficiently and accurately.



```
# 3. Encode Categorical Variables

# Identify categorical columns
categorical_cols = employee_df.select_dtypes(include=['object']).columns
print(f"\nCategorical Columns: {categorical_cols.tolist()}")
```

```
Categorical Columns: ['Attrition', 'BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole', 'MaritalStatus', 'OverTim
e']
```

```
# Encode categorical variables using LabelEncoder

import numpy as np
from sklearn.preprocessing import LabelEncoder, StandardScaler


le = LabelEncoder()
for col in categorical_cols:
    employee_df[col] = le.fit_transform(employee_df[col])
    print(f"Encoded '{col}' successfully.")
```

```
Encoded 'Attrition' successfully.
Encoded 'BusinessTravel' successfully.
Encoded 'Department' successfully.
Encoded 'EducationField' successfully.
Encoded 'Gender' successfully.
Encoded 'JobRole' successfully.
Encoded 'MaritalStatus' successfully.
Encoded 'OverTime' successfully.
```

Figure 22: Encoded Categorical Variables

Figure 22 below displays the categorical variable that went through the LabelEncoder method. The variables BusinessTravel, Department, and Gender were encoded to ensure they became acceptable inputs for the machine learning models. This step helps in ensuring that the

categorical data is presented in a format that can be understood by the model while at the same time preserving the feature information.

```
# 4. Normalize/Standardize Numerical Features

# Identify numerical columns
numerical_cols = employee_df.select_dtypes(include=['int64', 'float64']).columns

# Exclude the target variable 'Attrition' from normalization if it is numerical
if 'Attrition' in numerical_cols:
    numerical_cols = numerical_cols.drop('Attrition')
print(f"\nNumerical Columns (to be standardized): {numerical_cols.tolist()}")

# Standardize the numerical columns
scaler = StandardScaler()
employee_df[numerical_cols] = scaler.fit_transform(employee_df[numerical_cols])
```

```
Numerical Columns (to be standardized): ['Age', 'DailyRate', 'DistanceFromHome', 'Education', 'EmployeeNumber', 'EnvironmentSatisfac
tion', 'HourlyRate', 'JobInvolvement', 'JobLevel', 'JobSatisfaction', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked', 'Percent
SalaryHike', 'PerformanceRating', 'RelationshipSatisfaction', 'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear', 'Wor
kLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion', 'YearsWithCurrManager']
```

Figure 23: Identified Numerical Columns for Standardization

In figure 23, the ID of the selection of numerical columns that were standardized exclude the target variable Attrition. The scale of the features is standardized to make the numerical columns comparable to each other and avoid the influence of the large numerical features. This process is important for enhancing algorithm performance and achieving better convergence, especially in cases where algorithms are highly dependent on feature scaling.

```
# 5. Verify Preprocessed Data
print("Preprocessed Dataset Head:")
employee_df.head()

# Display summary statistics for numerical columns after scaling
print("Summary Statistics After Preprocessing:")
employee_df.describe()
```

```
Preprocessed Dataset Head:
Summary Statistics After Preprocessing:
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField
count	1.470000e+03	1470.000000	1470.000000	1.470000e+03	1470.000000	1.470000e+03	1.470000e+03	1470.000000
mean	-3.504377e-17	0.161224	1.607483	5.075305e-17	1.260544	4.350262e-17	7.008755e-17	2.247619
std	1.000340e+00	0.367863	0.665455	1.000340e+00	0.527792	1.000340e+00	1.000340e+00	1.331369
min	-2.072192e+00	0.000000	0.000000	-1.736576e+00	0.000000	-1.010909e+00	-1.868426e+00	0.000000
25%	-7.581700e-01	0.000000	1.000000	-8.366616e-01	1.000000	-8.875151e-01	-8.916883e-01	1.000000
50%	-1.011589e-01	0.000000	2.000000	-1.204135e-03	1.000000	-2.705440e-01	8.504925e-02	2.000000
75%	6.653541e-01	0.000000	2.000000	8.788772e-01	2.000000	5.932157e-01	1.061787e+00	3.000000
max	2.526886e+00	1.000000	2.000000	1.726730e+00	2.000000	2.444129e+00	2.038524e+00	5.000000

8 rows × 32 columns

Figure 24: Summary Statistics After Preprocessing

The summary statistics of the numerical features after standardization are given in the figure 8. Both the mean values are quite small and are close to zero, and the standard deviations are around one, which validates that the standardization process was correctly performed. This makes all the features numerical so that no feature is dominant because of its large scale as compared to other features. This step is also crucial as it helps to enhance the stability as well as the accuracy of the generated models.

3.3 Model Development and Analysis

3.3.1 XGBoost Model

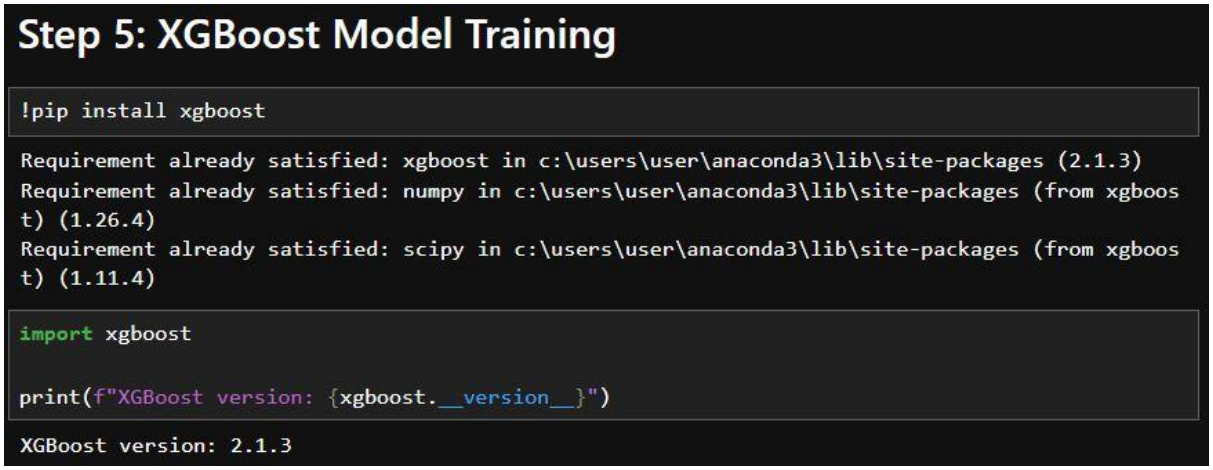


Figure 25: Installation and Verification of XGBoost Library

This figure proves that the successful installation of the XGBoost library version is 2.1.3. The verification also confirms that the library is well compatible with the Python environment and is ready for use in the model training and evaluation.

```python
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

# Step 4: Initialize and Train XGBoost Classifier
xgb_model = XGBClassifier(
    objective = 'binary:logistic',  # Binary classification
    eval_metric = 'logloss',        # Evaluation metric
    use_label_encoder = False,      # Avoid unnecessary warning
    random_state = 42
)

# Train the model
xgb_model.fit(X_train_resampled, y_train_resampled)

# Step 5: Make Predictions for Training and Testing Data
y_train_pred = xgb_model.predict(X_train_resampled)
y_train_pred_prob = xgb_model.predict_proba(X_train_resampled)[:, 1]

y_test_pred = xgb_model.predict(X_test)
y_test_pred_prob = xgb_model.predict_proba(X_test)[:, 1]

# Step 6: Evaluate the Model on Training Data
print("\nTraining Data Evaluation:")
print("Accuracy Score (Training):", accuracy_score(y_train_resampled, y_train_pred))
print("\nClassification Report (Training):")
print(classification_report(y_train_resampled, y_train_pred))
print("\nConfusion Matrix (Training):")
print(confusion_matrix(y_train_resampled, y_train_pred))

train_auc_score = roc_auc_score(y_train_resampled, y_train_pred_prob)
print(f"\nAUC-ROC Score (Training): {train_auc_score:.4f}")
print("================================================================================")
```

Figure 26: XGBoost Model Initialization, Training, and Predictions

The full code above shows how the XGBoost model is created, trained and how predictions are made using the model. It defines the objective function as binary logistic regression and the evaluation metric as log loss. The proposed model is applied to a resampled training data since the classes are imbalanced, and the predictions are made for both training and testing datasets to assess performance.

```
# Step 7: Evaluate the Model on Testing Data
print("\nTesting Data Evaluation:")
print("Accuracy Score (Testing):", accuracy_score(y_test, y_test_pred))
print("\nClassification Report (Testing):")
print(classification_report(y_test, y_test_pred))
print("\nConfusion Matrix (Testing):")
print(confusion_matrix(y_test, y_test_pred))

test_auc_score = roc_auc_score(y_test, y_test_pred_prob)
print(f"\nAUC-ROC Score (Testing): {test_auc_score:.4f}")

# Step 8: Plot AUC-ROC Curve for Training and Testing Data
# Training Data ROC Curve
fpr_train, tpr_train, _ = roc_curve(y_train_resampled, y_train_pred_prob)
plt.figure(figsize = (12, 6))
plt.plot(fpr_train, tpr_train, label = f'Training AUC = {train_auc_score:.4f}', color = 'blue')

# Testing Data ROC Curve
fpr_test, tpr_test, _ = roc_curve(y_test, y_test_pred_prob)
plt.plot(fpr_test, tpr_test, label = f'Testing AUC = {test_auc_score:.4f}', color = 'green')

# Random Guess Line
plt.plot([0, 1], [0, 1], color = 'red', linestyle = '--', label = 'Random Guess')

# Customize the Plot
plt.xlabel('False Positive Rate (FPR)')
plt.ylabel('True Positive Rate (TPR)')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc = 'lower right')
plt.grid()
plt.show()
```

Figure 27: Training and Testing Evaluation Metrics for XGBoost

The code above shows the evaluation metric of XGBoost model. They compute training and testing metrics of accuracy, precision, recall, F-measure, and AUC-ROC. Furthermore, it creates a confusion matrix to determine misclassification. The code also plots the ROC curve which enables a comparison of the model's ability to distinguish between classes for different datasets, providing for the best results, a thorough assessment of the model.

```
Training Data Evaluation:
Accuracy Score (Training): 1.0

Classification Report (Training):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       863
           1       1.00      1.00      1.00       863

    accuracy                           1.00      1726
   macro avg       1.00      1.00      1.00      1726
weighted avg       1.00      1.00      1.00      1726


Confusion Matrix (Training):
[[863   0]
 [  0 863]]

AUC-ROC Score (Training): 1.0000
================================================================================

Testing Data Evaluation:
Accuracy Score (Testing): 0.8344671201814059

Classification Report (Testing):
              precision    recall  f1-score   support

           0       0.87      0.94      0.91       370
           1       0.48      0.28      0.35        71

    accuracy                           0.83       441
   macro avg       0.67      0.61      0.63       441
weighted avg       0.81      0.83      0.82       441


Confusion Matrix (Testing):
[[348  22]
 [ 51  20]]

AUC-ROC Score (Testing): 0.6914
```

Figure 28: Output of The Evaluating the XGBoost Model

This figure shows the evaluation of XGBoost model on training set and testing set. The results obtained for the training dataset are perfect in terms of accuracy, precision, recall, and F1-Score for both the models, and the confusion matrix also showed no misclassification. However, the accuracy of the testing dataset reduced to 83.4 % with poor recall of the minority class at 28 % and F1 score at 35%. These metrics indicate that the model has a high training accuracy and poor capability for the minority class prediction on the testing data.

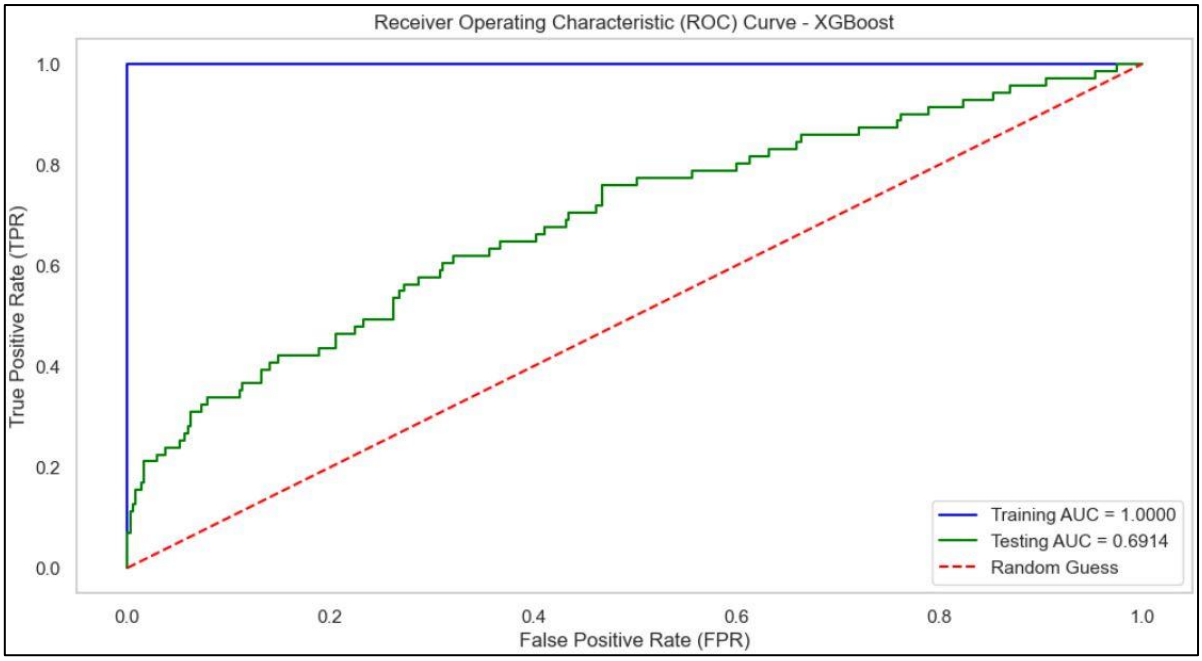


Figure 29: Output of Receiver Operating Characteristic (ROC) Curve for XGBoost Model

This figure also explains the Receiver Operating Characteristic (ROC) curve of the XGBoost model. The AUC score of the training data set was also 1.0 showing that the model is perfectly discriminating between the classes. However, the testing dataset has a Mean Per Class = 0.6914 which describes a moderate ability to classify between the positive and negative classes. The difference in the two is the training AUC and testing AUC, where the model may have performed a very comprehensive training but is poor in testing or in the unseen data sets.

3.3.2 Logistic Regression Model

Step 6: Logistic Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

# Step 1: Initialize and Train Logistic Regression Model
logistic_model = LogisticRegression(random_state=42, max_iter=1000)
logistic_model.fit(X_train_resampled, y_train_resampled)

# Step 2: Make Predictions for Training and Testing Data
y_train_pred = logistic_model.predict(X_train_resampled)
y_train_pred_prob = logistic_model.predict_proba(X_train_resampled)[:, 1]

y_test_pred = logistic_model.predict(X_test)
y_test_pred_prob = logistic_model.predict_proba(X_test)[:, 1]

# Step 3: Evaluate Logistic Regression Model on Training Data
print("\nLogistic Regression - Training Data Evaluation:")
print("Accuracy Score (Training):", accuracy_score(y_train_resampled, y_train_pred))
print("\nClassification Report (Training):")
print(classification_report(y_train_resampled, y_train_pred))
print("\nConfusion Matrix (Training):")
print(confusion_matrix(y_train_resampled, y_train_pred))

train_auc_score = roc_auc_score(y_train_resampled, y_train_pred_prob)
print(f"\nAUC-ROC Score (Training): {train_auc_score:.4f}")
print("================================================================================")
```

Figure 30: Logistic Regression Model Initialization, Training, and Predictions

This figure also shows how the logistic regression model is launched, trained using the resampled data, and how outcomes for the training and testing data are predicted. This code also limits the number of iterations to 1000 and depend on the predicted probabilities for the evaluation metrics such as AUC-ROC.

```
# Step 4: Evaluate Logistic Regression Model on Testing Data
print("\nLogistic Regression - Testing Data Evaluation:")
print("Accuracy Score (Testing):", accuracy_score(y_test, y_test_pred))
print("\nClassification Report (Testing):")
print(classification_report(y_test, y_test_pred))
print("\nConfusion Matrix (Testing):")
print(confusion_matrix(y_test, y_test_pred))

test_auc_score = roc_auc_score(y_test, y_test_pred_prob)
print(f"\nAUC-ROC Score (Testing): {test_auc_score:.4f}")

# Step 5: Plot AUC-ROC Curve for Logistic Regression
# Training Data ROC Curve
fpr_train, tpr_train, _ = roc_curve(y_train_resampled, y_train_pred_prob)
plt.figure(figsize = (12, 6))
plt.plot(fpr_train, tpr_train, label = f'Training AUC = {train_auc_score:.4f}', color = 'blue')

# Testing Data ROC Curve
fpr_test, tpr_test, _ = roc_curve(y_test, y_test_pred_prob)
plt.plot(fpr_test, tpr_test, label = f'Testing AUC = {test_auc_score:.4f}', color = 'green')

# Random Guess Line
plt.plot([0, 1], [0, 1], color = 'red', linestyle = '--', label = 'Random Guess')

# Customize the Plot
plt.xlabel('False Positive Rate (FPR)')
plt.ylabel('True Positive Rate (TPR)')
plt.title('Receiver Operating Characteristic (ROC) Curve - Logistic Regression')
plt.legend(loc = 'lower right')
plt.grid()
plt.show()
```

Figure 31: Training and Testing Evaluation Metrics for Logistic Regression Model

This figure above shows the script coding used in Python programming language to plot the ROC for the Logistic Regression model. The code determines the False Positive Rate (FPR) and True Positive Rate (TPR) for both training and testing the set datasets and plot an ROC curve of the model against a baseline random guess. For better performance comparison the training and testing AUC scores are also added to the visualization. Besides that, these are also intertwined with the evaluation methodology, where the technical parameters like accuracy, precision, recall, F1-score, and AUC-ROC and the confusion matrices belong to. Besides, the code creates an ROC curve to help understand the model's discrimination capability between the positive and negative class.

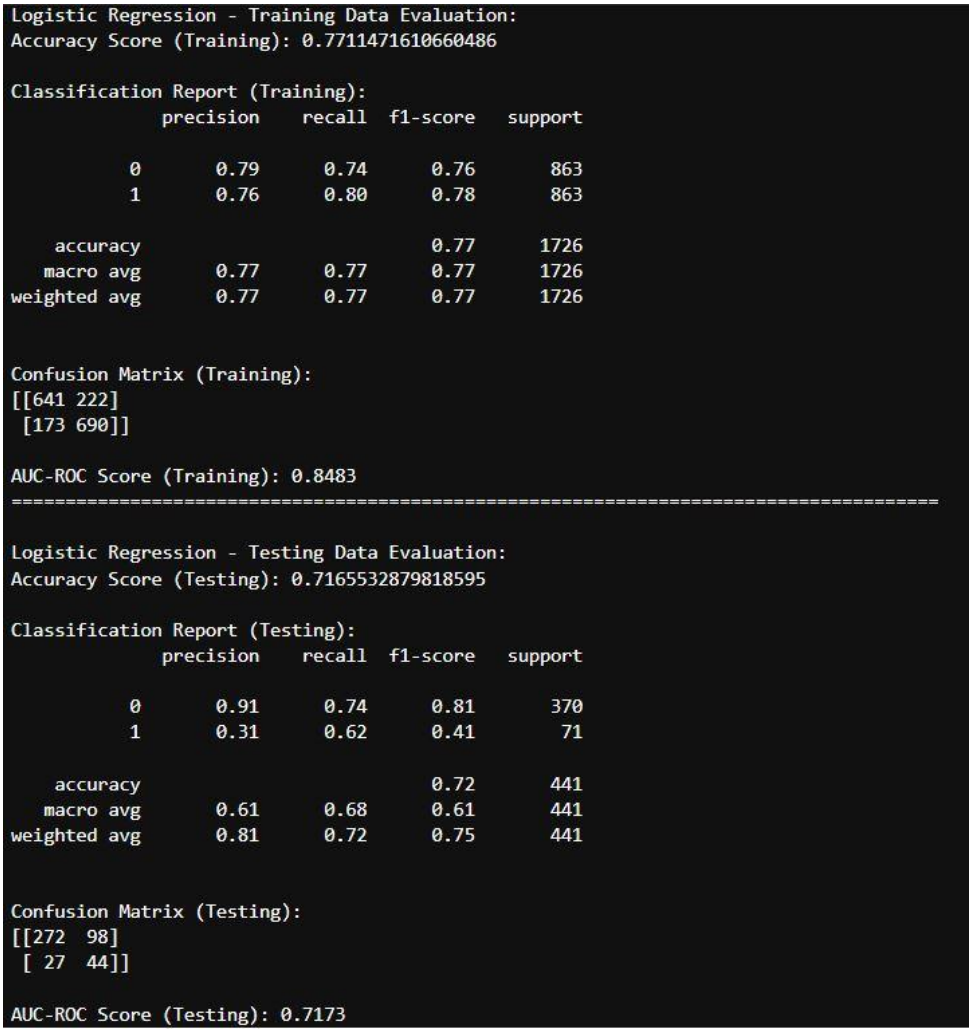


Figure 32: Output of The Evaluating the Logistic Regression Model

This figure above also shows the evaluation indices of the Logistic Regression model when it was trained on the training set and tested on the test set. The training dataset had an accuracy of 77.1 percent and had equal balanced recall and F1 scores with the score of 0.77. As for the testing dataset, the accuracy has decreased to 71.6%, while the recall and F1-score (the macro-average), made 0.61 and 0.75 correspondingly. The results presented in the confusion matrix show that even through the model generally identified the majority class, it poorly identified the minority class since its recall values are relatively low for that class.

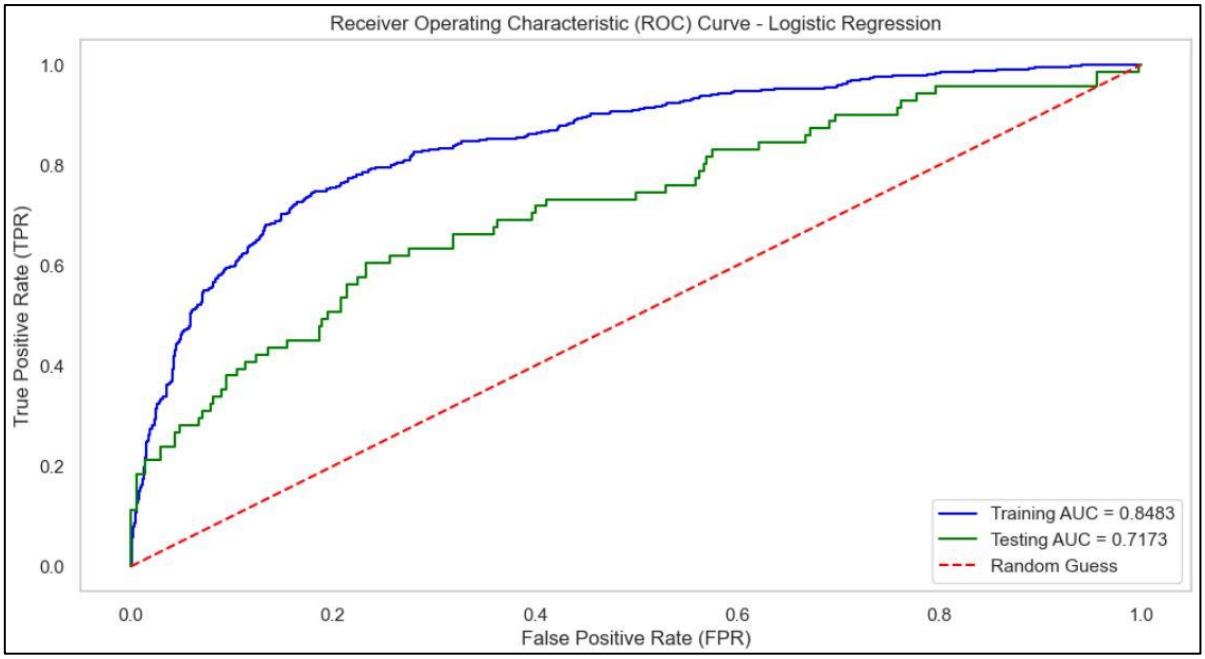


Figure 33: Output of Receiver Operating Characteristic (ROC) Curve for Logistic Regression Model

The following is the ROC curve for the model we have used, the Logistic Regression. Evaluation of the training dataset correspondingly yielded an average AUC of 0.8483, which showed that it had a good capability for class discrimination. Nevertheless, the model's performance is outperformed when tested against the actual testing dataset with an AUC score of only 0.7173. The difference between the training and the testing AUC suggests possible overfitting since the model can generalize on new data.

3.3.3 Support Vector Machine (SVM) Model

Step 6: SVM

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

# Step 1: Initialize and Train SVM Model
# Using the probability parameter for AUC-ROC evaluation
svm_model = SVC(probability = True, kernel = 'linear', random_state = 42)
svm_model.fit(X_train_resampled, y_train_resampled)

# Step 2: Make Predictions for Training and Testing Data
y_train_pred = svm_model.predict(X_train_resampled)
y_train_pred_prob = svm_model.predict_proba(X_train_resampled)[:, 1]

y_test_pred = svm_model.predict(X_test)
y_test_pred_prob = svm_model.predict_proba(X_test)[:, 1]

# Step 3: Evaluate SVM Model on Training Data
print("\nSVM - Training Data Evaluation:")
print("Accuracy Score (Training):", accuracy_score(y_train_resampled, y_train_pred))
print("\nClassification Report (Training):")
print(classification_report(y_train_resampled, y_train_pred))
print("\nConfusion Matrix (Training):")
print(confusion_matrix(y_train_resampled, y_train_pred))

train_auc_score = roc_auc_score(y_train_resampled, y_train_pred_prob)
print(f"\nAUC-ROC Score (Training): {train_auc_score:.4f}")
print("================================================================================")
```

Figure 34: SVM Model Initialization, Training, and Predictions

This figure also shows the training and predictions of the SVM model initialization steps to the algorithm. The model adopts the linear kernel with the probability=True parameter to make AUC- ROC evaluation possible. The predictions are made for both the training and testing datasets and the probabilities which have been predicted are used in calculating the ROC curve and every other performance measure.

```python
# Step 4: Evaluate SVM Model on Testing Data
print("\nSVM - Testing Data Evaluation:")
print("Accuracy Score (Testing):", accuracy_score(y_test, y_test_pred))
print("\nClassification Report (Testing):")
print(classification_report(y_test, y_test_pred))
print("\nConfusion Matrix (Testing):")
print(confusion_matrix(y_test, y_test_pred))

test_auc_score = roc_auc_score(y_test, y_test_pred_prob)
print(f"\nAUC-ROC Score (Testing): {test_auc_score:.4f}")

# Step 5: Plot AUC-ROC Curve for SVM
# Training Data ROC Curve
fpr_train, tpr_train, _ = roc_curve(y_train_resampled, y_train_pred_prob)
plt.figure(figsize = (12, 6))
plt.plot(fpr_train, tpr_train, label = f'Training AUC = {train_auc_score:.4f}', color = 'blue')

# Testing Data ROC Curve
fpr_test, tpr_test, _ = roc_curve(y_test, y_test_pred_prob)
plt.plot(fpr_test, tpr_test, label = f'Testing AUC = {test_auc_score:.4f}', color = 'green')

# Random Guess Line
plt.plot([0, 1], [0, 1], color = 'red', linestyle = '--', label = 'Random Guess')

# Customize the Plot
plt.xlabel('False Positive Rate (FPR)')
plt.ylabel('True Positive Rate (TPR)')
plt.title('Receiver Operating Characteristic (ROC) Curve - SVM')
plt.legend(loc = 'lower right')
plt.grid()
plt.show()
```

Figure 35: Training and Testing Evaluation Metrics for SVM Model

The following figure 35 showing the Python code for assessment of the SVM model both on training and testing data sets. It estimates the accuracy, precision, recall, F1-score and the area under receiving operating characteristic (AUC-ROC) curves. Furthermore, we create confusion matrices in order to address misclassification patterns. The results section of this evaluation offers a detailed assessment of the performance of the model on the two datasets.

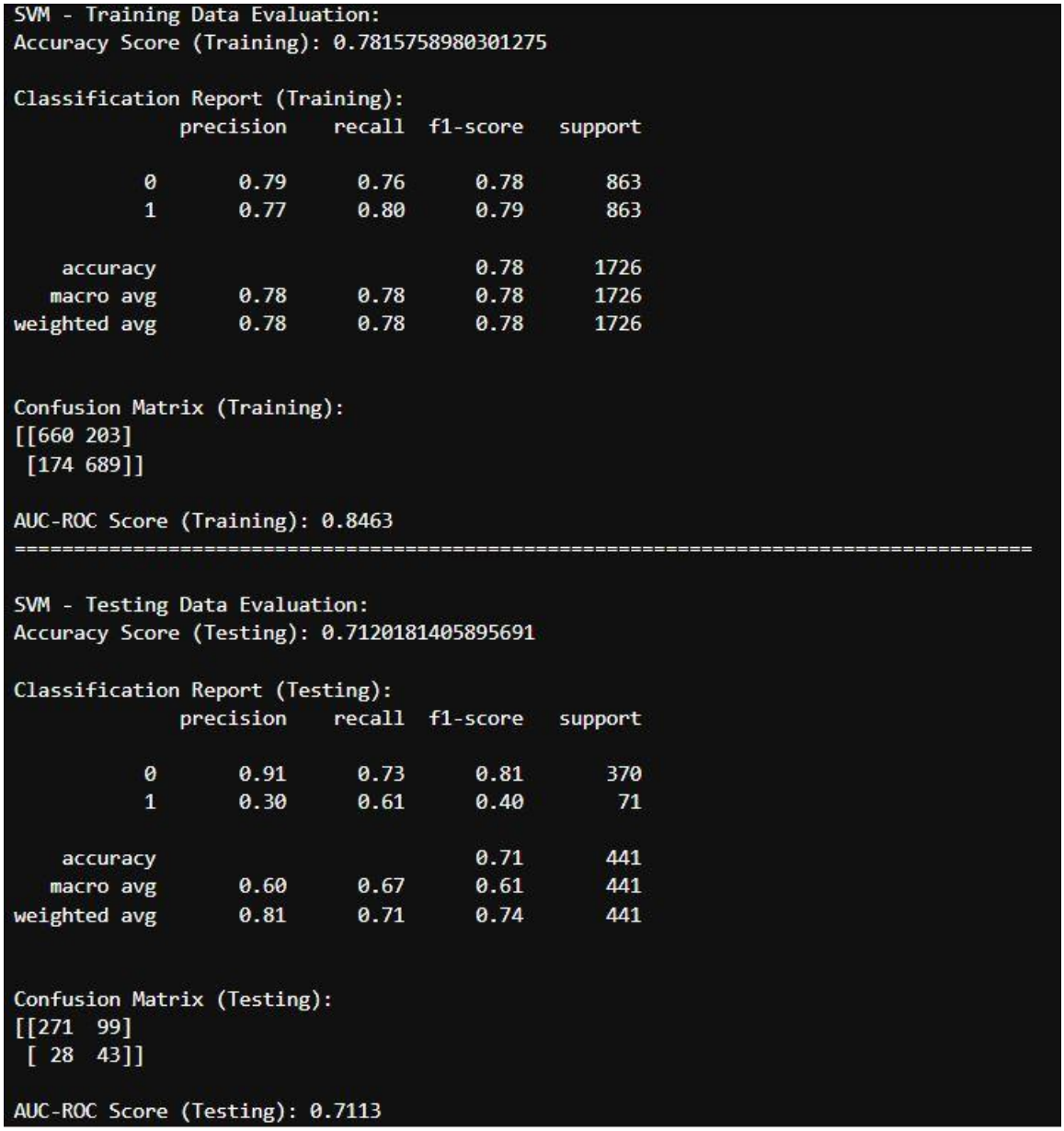


Figure 36: Output of The Evaluating the SVM Model

This figure exhibits the performance evaluation criteria of the SVM model on the training and testing data set. Training accuracy is 78.1 % and the balanced recall with F1 score are 0.78. The accuracy of the testing dataset declined to 71.2 % with recall and F1-scores of 0.61 and 0.74 (macro averages). The confusion matrices show that the model performed poorly with the minority class by having lower recall and precision values for the same class.

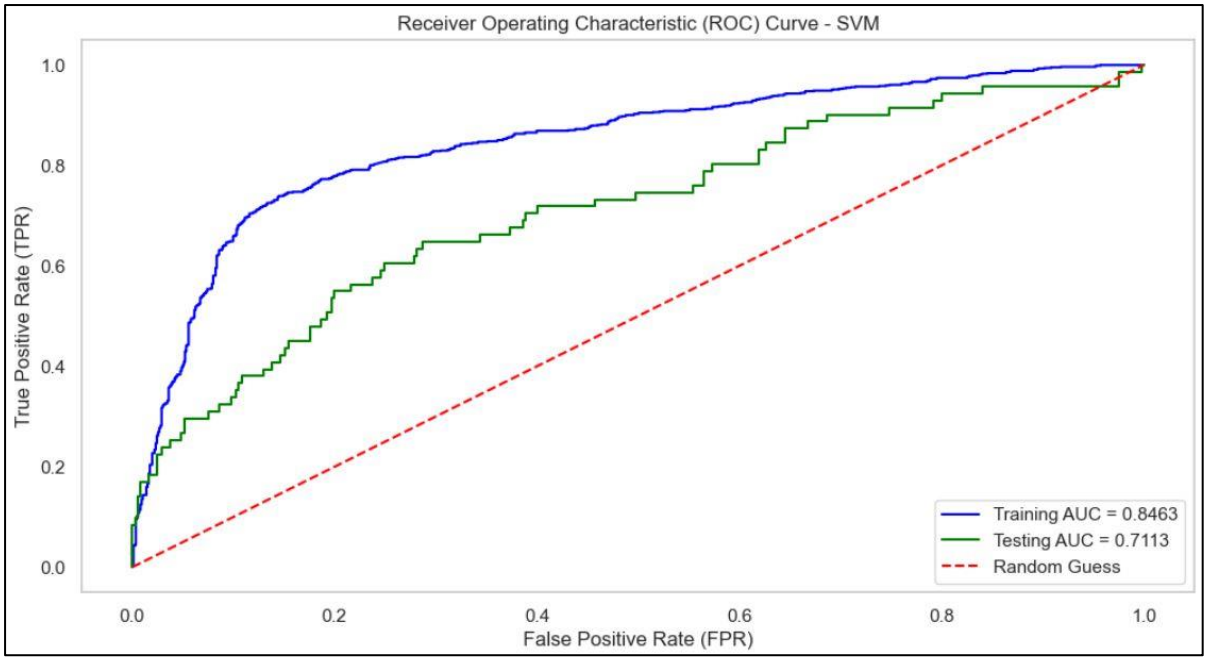


Figure 37: Output of Receiver Operating Characteristic (ROC) Curve for SVM Model

This figure 37 is the figure of the Receiver Operating Characteristic (ROC) curve of the SVM Model. For the training data set the classification result was evaluated using the AUC score and the value of 0.8463 shows that the classification was good. However, the testing data set has a low of 0.7113 AUC score although the model has a moderate performance on the unseen data. The difference in between training and testing AUC scores reveals certain difficulties with generalization, but the model's results remain acceptable for both datasets.

Table 3: Summary of Output of Three Models

Model	**Dataset**	**Accuracy**	**Recall (Macro Avg)**	**F1-Score (Macro Avg)**	**AUC-ROC Score**
XGBoost	Training	1.000	1.000	1.000	1.000
	Testing	0.834	0.610	0.630	0.6914
	Training	0.771	0.770	0.770	0.8483

Logistic Regression	Testing	0.716	0.610	0.720	0.7073
SVM	Training	0.782	0.780	0.780	0.8463
	Testing	0.712	0.610	0.740	0.7113

According to the results obtained from the comparison of the model performances, it will be realized that each algorithm had a different level of success in predicting the employee attrition. When applied to the training dataset the XGBoost model was able to achieve an accuracy, recall, F1 -score and AUC-ROC score of 1.0 which indicated complete overfitting of the model to the training data. However, the test performance declined to 0.6914 AUC-ROC and moderate, which raises questions about the model's ability to generalize on unseen data. Logistic Regression was also good, the AUC-ROC for the training data set was 0.8483 while that of the testing data set was 0.7173. For the testing data, Logistic Regression seems to have better generalization, but the model is still not well suited for the minority class. Likewise, the performance of the SVM model was also trained with the AUC-ROC score of 0.8463 and obtained the testing AUC-ROC score of 0.7113 thereby describing similar pattern as that of the Logistic Regression. The recall and F1-scores of the models presented moderate values for the majority class, but poor values for the minority which evidence shortcomings for dealing with class imbalance, despite resampling. In general, Logistic Regression and SVM showed a relatively equal comparison to XGBoost model but were more appropriate for the datasets that require minimum overfitting and higher generalization. However, more enhancements and some strategies like hyperparameters optimization or ensemble methods could enhance the models for the better performance on the imbalanced data sets.

3.4 Best Performing Model and Analysis

From the previous three models, the best model would be Logistic Regression because due to its balance between performance, simplicity, and interpretability, aligning with both predictive and business objectives in the context of employee attrition prediction. To optimize the predictive performance of the Logistic Regression model and address the high-dimensional

nature of the dataset, Principal Component Analysis (PCA) and clustering techniques were employed for feature selection. PCA was first applied to reduce the dimensionality of the dataset while retaining 95% of the variance. Subsequently, K-Means clustering was performed on the PCA-transformed data to group similar features into five distinct clusters. From each cluster, a representative feature was selected, resulting in a final subset of five features which are Age, BusinessTravel, DailyRate, HourlyRate, and OverTime.

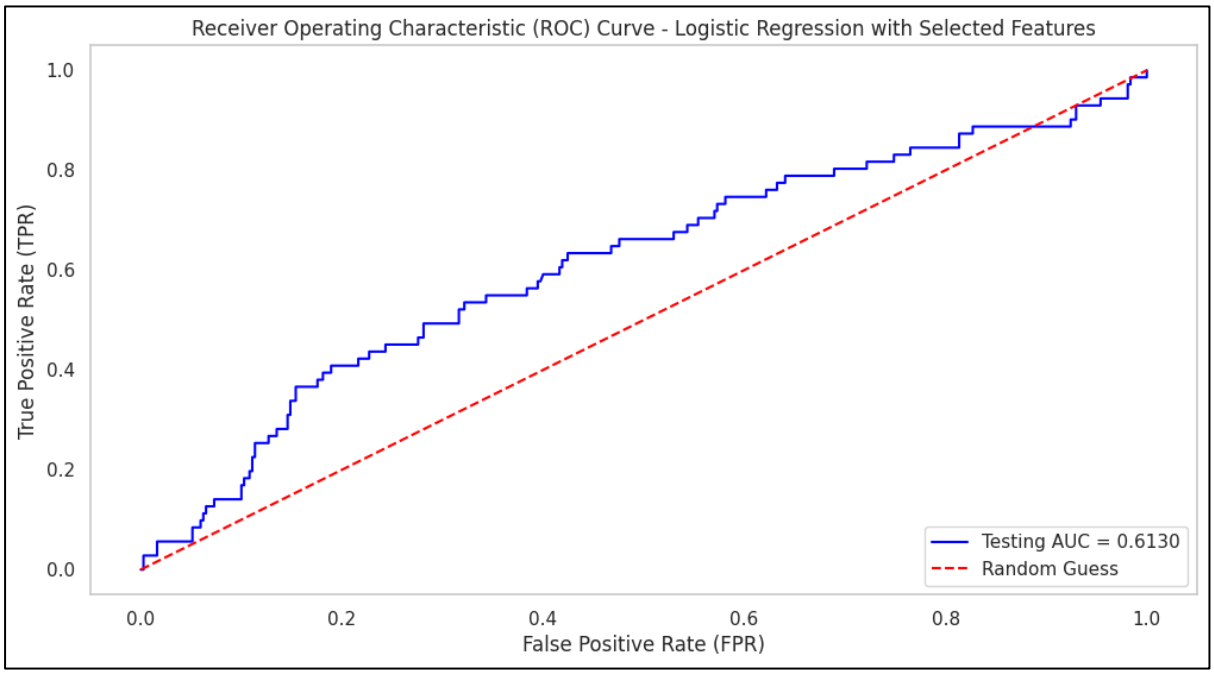


Figure 38: (ROC) Curve - Logistic Regression with Selected Features

Figure 38 shows the logistic regression model was re-trained using only selected features. The model was evaluated using the testing dataset with the performance metrics: Accuracy is 0.8367, F1-Score is 0.4555, and AUC-ROC is 0.6130. The Receiver Operating Characteristic (ROC) curve was plotted to visualize the model's ability to discriminate between positive and negative classes. The AUC-ROC score of 0.6130 indicates a moderate capability for class discrimination. This approach highlights the effectiveness of dimensionality reduction and clustering in simplifying the dataset while maintaining sufficient predictive power. However, the slight reduction in F1-Score suggests that the model's ability to balance precision and recall could be improved further with additional refinements, such as hyperparameter tuning or ensemble methods.

4.0 Model Validation and Optimization

After choosing the best features using PCA, cross-validation is done to ensures the model is able to generalize well to unseen data. Cross-validation splits the training data into 10 folds (chosen for this project) and each fold is used for testing while the rest is for training data. Logistic regression has multiple parameters that can be tuned to get the best performance which are 'penalty', 'solver', 'max_iter' and 'C' which is the inverse of regularization. Auto hyperparameter tuning such as GridSearch is a good tool by working through multiple combinations of parameter and find the one with the best performance. Figure below shows the range of parameters specifies for GridSearch.

```
param_grid = [{'penalty':['l1','l2','elasticnet','none'],
    'C' : [ 1, 2, 3, 4, 5],
    'solver': ['lbfgs','newton-cg','liblinear','sag','saga'],
    'max_iter'  : [100,1000,2500,5000] }]

clf = GridSearchCV(logistic_model, param_grid = param_grid, cv = 10, verbose=True, n_jobs=-1)
clf
```

Figure 39: Parameters range for GridSearch.

```
LogisticRegression
LogisticRegression(C=1, penalty='l1', random_state=42, solver='liblinear')
```

Figure 40: The best parameter combination.

Figure above shows the best parameters combination found by using GridSearch. The data then is trained using the combinations and tested on test data. The new results obtained is summarise in table below.

Table 4: Model result after validation.

Accuracy	0.8367

F1-Score	0.4556
AUC-ROC	0.6135

4.1 Results and Discussion

In this project, three models are used to train and predict the status of attrition of employees and their performance are compared using accuracy score, recall, f1-score and AUC-ROC graph. From Table 3, Logistic Regression has been proven to be the best model compared to others. To further improve the model performance, feature selection using PCA and clustering that address the high dimensionality were applied to select a subset of five representative features such as (Age, BusinessTravel, DailyRate, HourlyRate, OverTime). The model is re-trained with these features that resulted in a testing accuracy of 0.8367, F1-Score of 0.4555, and AUC-ROC of 0.6130. While the reduced feature set simplified the model and improve efficiency, the slight reduction in F1-Score indicate that the challenges in balancing the precision and recall for the minority class.

Table 5: Summary result before and after validation.

Cross validation	Accuracy	F1-Score	AUC-ROC Score
Before	0.8367	0.4555	0.6130
After	0.8367	0.4555	0.6135

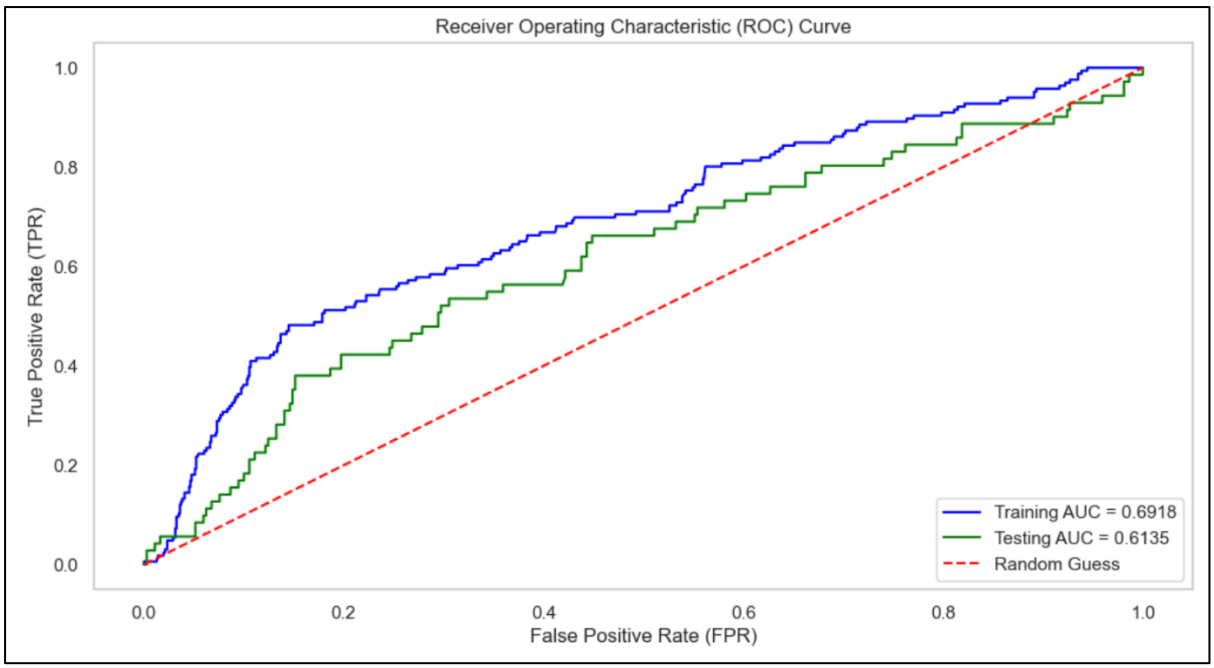


Figure 41: AUC-ROC graph post-optimization.

Table 5 shows the results of model performance before and after doing cross validation and hyperparameter tuning using the selected five features. The model very slightly improves in AUC-ROC score while the rest has the same value with the ones before validation.

To conclude, the results shows that logistic regression offers a balance trade-off between performance, simplicity, and interpretability. It generalizes better to unseen the data compared to the XGBoost and SVM, making it more suitable for real-world application. However, lower recall and F1-Scores for the minority class across all models highlight the need for further improvements. Additionally, the application of PCA and clustering for feature selection proved effective in simplifying the dataset without significantly compromising predictive power, underscoring the value of dimensionality reduction in machine learning workflows. Cross validation can further enhance the model's ability to generalize to unseen data.

4.2 Challenges and Decisions

This project aimed to develop a robust predictive model for employee attrition using Logistic Regression, XGBoost, and SVM while addressing key challenges in data preparation, model selection, and feature optimization. Below are the key challenges for this project:

1. **Encoding Categorical Variables:** It can be difficult to convert categorical data into numerical formats that machine learning algorithms can use, particularly when working with ordinal or high cardinality features during data preparation process. This is because inaccurate encoding techniques might result from misinterpreting the nature of categorical variables (nominal vs. ordinal), which may reduce the models' prediction ability.

2. **Unbalanced datasets:** Such as skewed target class distributions (e.g., in attrition prediction where the number of employees who depart may be significantly fewer than those who stay), are difficult to identify and address in EDA. Such imbalances might result in biassed models that favour the majority class, necessitating measures such as oversampling, under sampling, or complex algorithms to address the problem.

3. **Choosing parameters**: When doing the model validation, it is difficult to find the suitable range of values for parameter. So, a careful consideration needs to be done by putting an initial range of values and continues smaller the values scope based on the specific value chosen from the initial range.

4. **Dimensionality challenge**: The project also faced a dimensionality challenge because of the dataset contained numerous features which were highly correlated or redundant. This necessitated the use of dimensionality reduction techniques like PCA, followed by clustering, to select a subset of representative features. By reducing the feature space, the computational efficiency of the models improved, and the focus was directed toward key predictors of attrition, such as Age, OverTime, and BusinessTravel.

5. **Model Selection:** The Logistic Regression was chosen as the preferred model due to its balanced performance, interpretability, and simplicity compared to XGBoost and SVM. While XGBoost demonstrated strong training performance, it failed to generalize well, and SVM required significantly more computational resources. Logistic Regression, on the other hand, achieved consistent metrics across training and testing datasets and provided clear, interpretable results that aligned with the project's goals of generating actionable insights.

In conclusion, it became the evident that the employee attrition prediction requires a balance between prediction performance and interpretability. Logistic regression has emerged

as a reliable model for providing insights into the key factors influencing attrition such as work-life balance, overtime, and job satisfaction. The use of advanced preprocessing techniques like PCA and clustering ensured that the model remained efficient and interpretable. These findings emphasize the importance of thoughtful preprocessing, model selection, and evaluation metrics in predictive analytics, providing a foundation for more targeted HR strategies and interventions.

5.0 Conclusions

In conclusion, all three models, SVM, XGBoost and Logistic Regression has been explored and used for training the dataset to predict the status attrition of employees. Those models are then compared based on their performance using appropriate measuring metrics such as accuracy, recall, f1-score and AUC-ROC score. Logistic Regression has been found to be the best model. To further improve the performance, a feature selection using PCA and k-means clustering is being done to reduce the dimensionality of data. Lastly, model validation such as cross-validation and hyperparameter tuning using GridSearch is being done to ensure that the model can generalize well on unseen data. It does improve the performance of the model but only with a very small increases in AUC-ROC score.

REFERENCES

Gupta, R., Chand, A. S., Solanki, S., Gautam, T., & Garg, N. (2024). A Comparative Analysis of Logistic Regression and Support Vector Machine for Employee Churn Prediction. *Int. Conf. Sustain. Expert Syst., ICSES - Proc.*, 1821–1827. Scopus. https://doi.org/10.1109/ICSES63445.2024.10762955

Nawaz, M. S., Nawaz, M. Z., Fournier-Viger, P., & Luna, J. M. (2024). Analysis and classification of employee attrition and absenteeism in industry: A sequential pattern mining-based methodology. *Computers in Industry*, *159–160*, 104106. https://doi.org/10.1016/j.compind.2024.104106

Polisetti, S., Bhargavi, M., Chitneni, S., Eluri, S., Kattamuri, N., & Renugadevi, R. (2024). Stacking Models for Employee Attrition Prediction: Leveraging Logistic Regression and Random Forest. *Int. Conf. I-SMAC (IoT Soc., Mob., Anal. Cloud), I-SMAC - Proc.*, 863–867. Scopus. https://doi.org/10.1109/I-SMAC61858.2024.10714670

Qiao, F. (2023, February 16). Logistic Regression Model Tuning with scikit-learn — Part 1. *Medium*. https://towardsdatascience.com/logistic-regression-model-tuning-with-scikit-learn-part-1-425142e01af5