Assessment Report

on

“Employee Attrition Prediction”

Submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

SESSION 2024-2025

in

CSE(AI)

By

Name : Pratham Mishra

Roll Number : 202401100300180

Section : C

“Under the supervision of”

“Mr. Mayank Lakhotia”

KIET Group of Institutions, Ghaziabad

May, 2025

Introduction

As organizations strive to maintain a competitive edge in today's rapidly changing business environment, employee retention has emerged as a critical factor. High attrition rates can lead to a loss of experienced talent, disruption in operations, and increased hiring costs. Therefore, predicting employee attrition has become a priority for HR departments in large organizations.

This project leverages the IBM HR Analytics Employee Attrition dataset to develop a machine learning model capable of predicting whether an employee is likely to leave the organization. By using classification techniques, the goal is to identify patterns and key features influencing attrition, enabling companies to take proactive measures to retain talent.

Problem Statement

To develop a supervised machine learning model that accurately predicts whether an employee will leave the organization based on historical HR data. The classification outcome will help human resource departments in reducing attrition and improving employee engagement and satisfaction.

Objectives

- To preprocess and analyze the IBM HR dataset effectively.
- To build a classification model to predict employee attrition.
- To evaluate the model using accuracy, precision, recall, F1-score, and confusion matrix.
- To visualize and interpret feature importance for key HR attributes contributing to attrition.

Methodology

Data Collection:

The dataset used in this project is the IBM HR Analytics Employee Attrition dataset, publicly available on Kaggle. It includes various employee-related features such as job role, work environment, performance ratings, and satisfaction levels.

Data Preprocessing:

- **Handling Missing Values**: Verified and handled any missing values using appropriate imputation strategies.
- **Categorical Encoding**: Converted categorical variables (e.g., Gender, MaritalStatus, JobRole) into numerical format using one-hot encoding.
- **Feature Scaling**: Standardized continuous variables using `StandardScaler` to ensure uniform scale.
- **Train-Test Split**: Split the dataset into training (80%) and testing (20%) sets using `train_test_split`.

Model Building:

- Implemented various classification models including:
    - **Logistic Regression**
    - **Random Forest Classifier**
    - **Support Vector Machine (SVM)**
- Selected the best model based on evaluation metrics.

Model Evaluation:

- Used metrics such as **accuracy**, **precision**, **recall**, **F1-score**.
- Visualized **confusion matrix** using Seaborn heatmap.
- Extracted and visualized **feature importance** (especially from tree-based models) to understand which features most impact attrition.

Data Preprocessing

The dataset was carefully cleaned and transformed to ensure model efficiency and accuracy:

- All numerical missing values were replaced with the mean.
- Categorical variables were one-hot encoded (e.g., for Department, EducationField, JobRole).
- The entire dataset was normalized using StandardScaler.
- Stratified split ensured balanced distribution of the attrition class across train and test sets.

Model Implementation

Three classification models were tested. Logistic Regression was simple and interpretable. Random Forest showed high performance due to its ensemble nature and capability to handle both linear and non-linear relationships. SVM was also experimented with but had longer training times.

The Random Forest classifier was finalized due to its high accuracy and balanced precision-recall tradeoff. It also provided insight into feature importance.

Evaluation Metrics

The models were evaluated using the following metrics:

- Accuracy: Overall correctness of the model's predictions.
- Precision: Proportion of true positive attrition predictions among all predicted positives.
- Recall: Proportion of actual attrition cases that were correctly predicted.
- F1 Score: Harmonic mean of precision and recall.
- Confusion Matrix: Visualized using a Seaborn heatmap to identify false positives and false negatives.

Results and Analysis

- The Random Forest model demonstrated superior performance over logistic regression and SVM in terms of accuracy and recall.
- The confusion matrix showed a relatively low number of false negatives, indicating strong performance in identifying likely attrition cases.
- Feature importance analysis revealed that Job Satisfaction, OverTime, YearsAtCompany, WorkLifeBalance, and EnvironmentSatisfaction were among the top predictors of attrition.
- Employees with frequent overtime, low satisfaction scores, or limited years at the company were more likely to leave.

Code

```
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.preprocessing import LabelEncoder, StandardScaler

from sklearn.linear_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score


# Step 1: Load the dataset

file_path = '/content/WA_Fn-UseC_-HR-Employee-Attrition.csv'

df = pd.read_csv(file_path)

print("Dataset loaded successfully!")

print(df.head())


# Step 2: Encode target variable first to avoid errors in correlation

df['Attrition'] = df['Attrition'].apply(lambda x: 1 if x == 'Yes' else 0)


# Step 3: Exploratory Data Analysis (EDA)

print("\nDataset info:")

print(df.info())


print("\nChecking for missing values:")

print(df.isnull().sum())


print("\nTarget variable distribution:")

print(df['Attrition'].value_counts())
```

```
# Visualize target variable distribution
sns.countplot(x='Attrition', data=df)
plt.title('Attrition Distribution (0=No, 1=Yes)')
plt.show()

# Step 4: Correlation Heatmap (only numeric columns)
plt.figure(figsize=(16,12))
numeric_df = df.select_dtypes(include=[np.number])  # Select numeric only to avoid error
corr = numeric_df.corr()
sns.heatmap(corr, annot=False, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Feature Correlation Heatmap')
plt.show()

# Step 5: Data Preprocessing
categorical_cols = df.select_dtypes(include=['object']).columns.tolist()
print(f"\nCategorical columns to encode: {categorical_cols}")

# Encode categorical columns using LabelEncoder
le = LabelEncoder()
for col in categorical_cols:
    df[col] = le.fit_transform(df[col])
```

```
# Drop irrelevant columns
X = df.drop(['Attrition', 'EmployeeNumber'], axis=1)
y = df['Attrition']

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Step 6: Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,
                                                    random_state=42, stratify=y)

print(f"\nTraining samples: {X_train.shape[0]}, Testing samples:
{X_test.shape[0]}")

# Step 7: Logistic Regression
log_model = LogisticRegression(random_state=42)
log_model.fit(X_train, y_train)
y_pred_log = log_model.predict(X_test)

print("\nLogistic Regression Performance:")
```

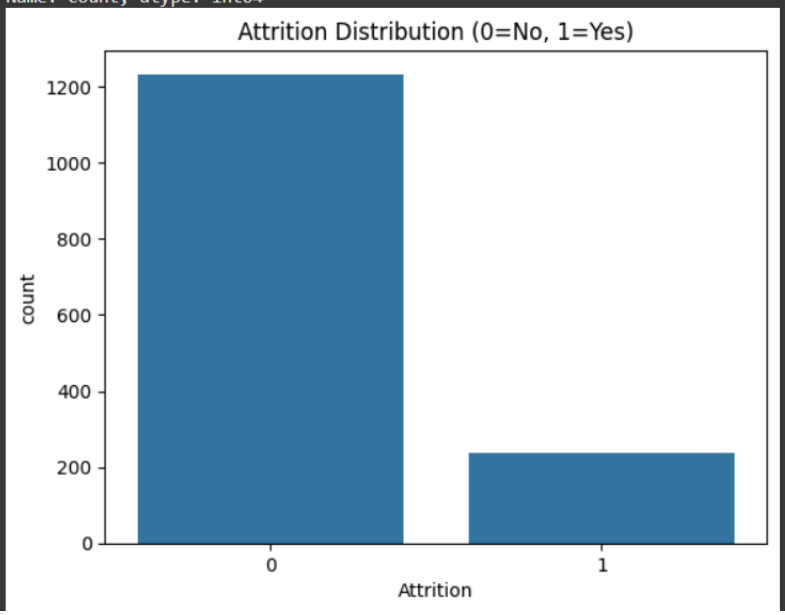
```
print(f"Accuracy: {accuracy_score(y_test, y_pred_log):.4f}")
print("Classification Report:\n", classification_report(y_test, y_pred_log))

# Step 8: Random Forest Classifier
rf_model = RandomForestClassifier(random_state=42, n_estimators=100)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)

print("\nRandom Forest Classifier Performance:")
print(f"Accuracy: {accuracy_score(y_test, y_pred_rf):.4f}")
print("Classification Report:\n", classification_report(y_test, y_pred_rf))

# Step 9: Feature Importance visualization
importances = rf_model.feature_importances_
features = X.columns

feat_importances = pd.Series(importances,
index=features).sort_values(ascending=False)

plt.figure(figsize=(12,8))
sns.barplot(x=feat_importances[:15], y=feat_importances.index[:15])
plt.title('Top 15 Feature Importances from Random Forest')
```

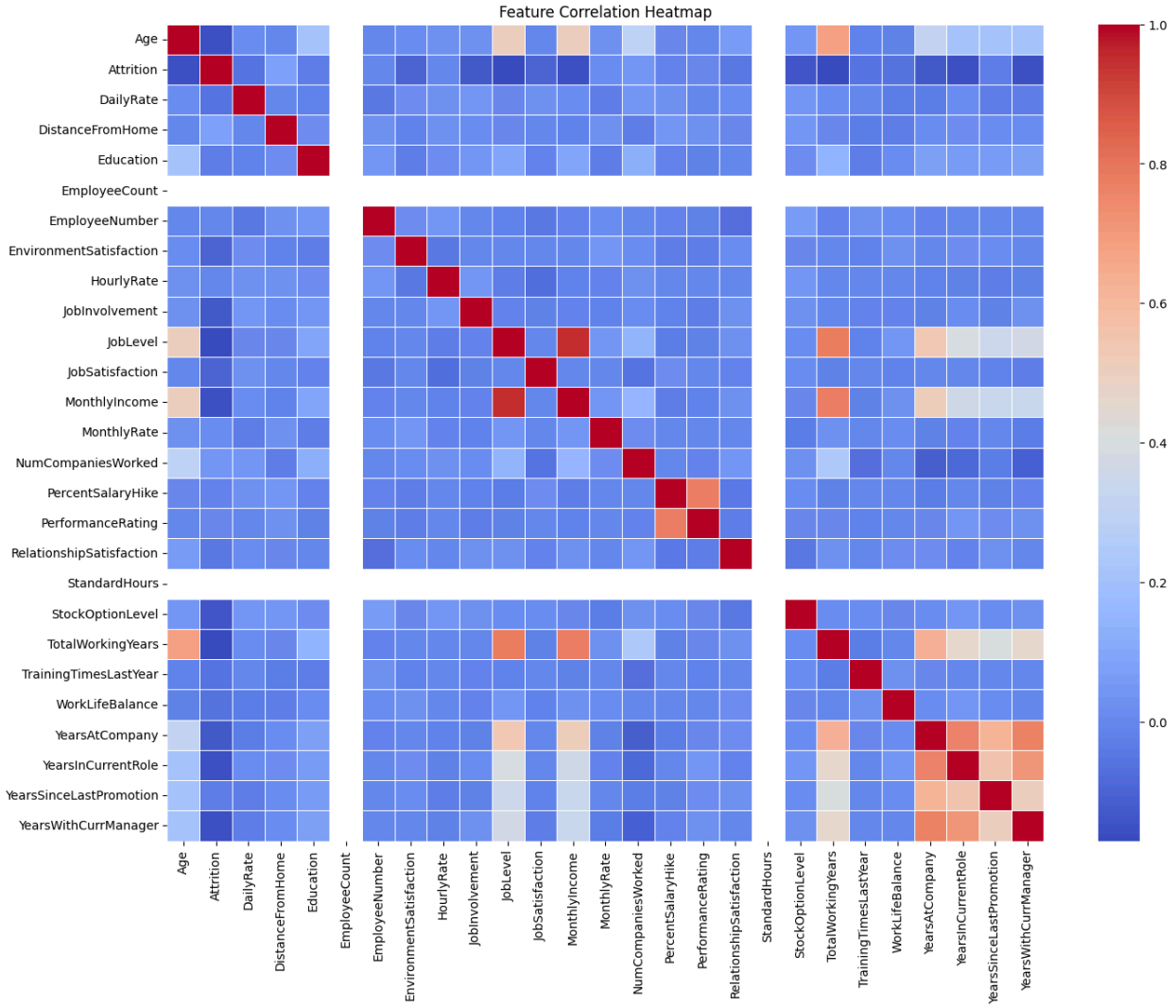
```
plt.xlabel('Importance Score')
plt.ylabel('Feature')
plt.show()

# Step 10: Confusion Matrix for Random Forest
cm = confusion_matrix(y_test, y_pred_rf)

plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=['No Attrition', 'Attrition'],
            yticklabels=['No Attrition', 'Attrition'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - Random Forest')
plt.show()
```

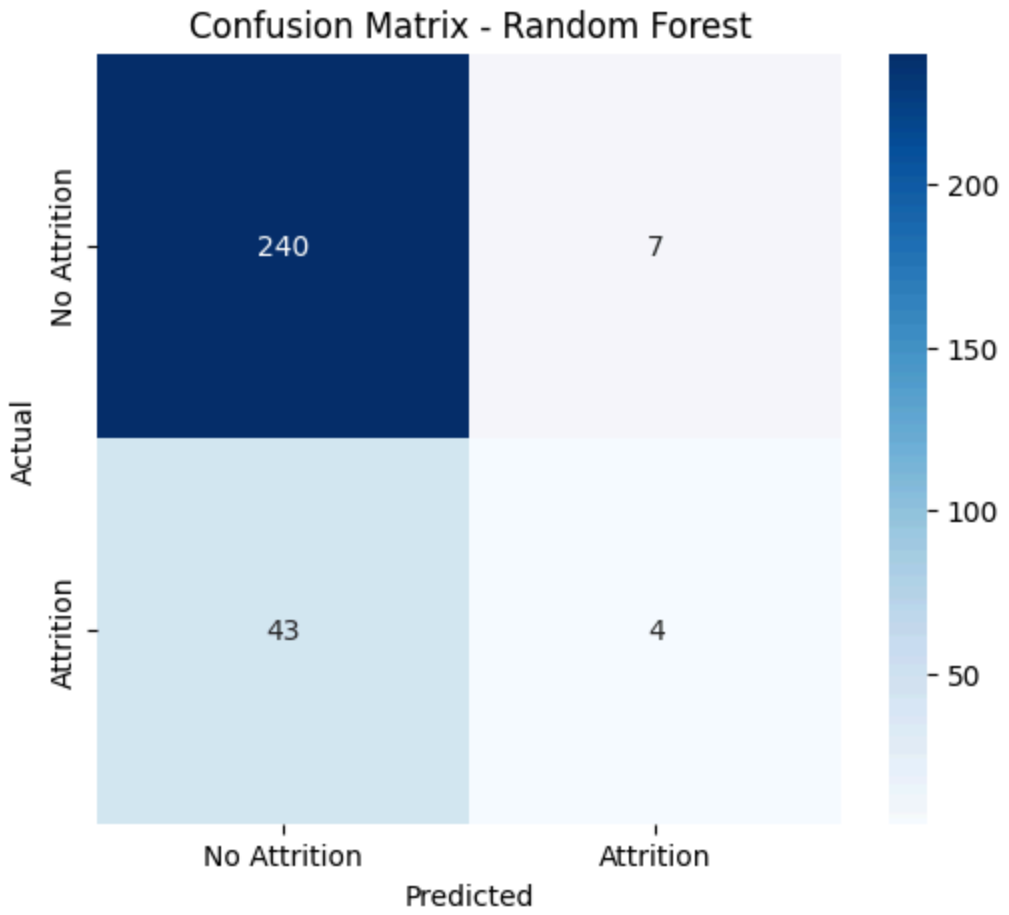
Result/Output

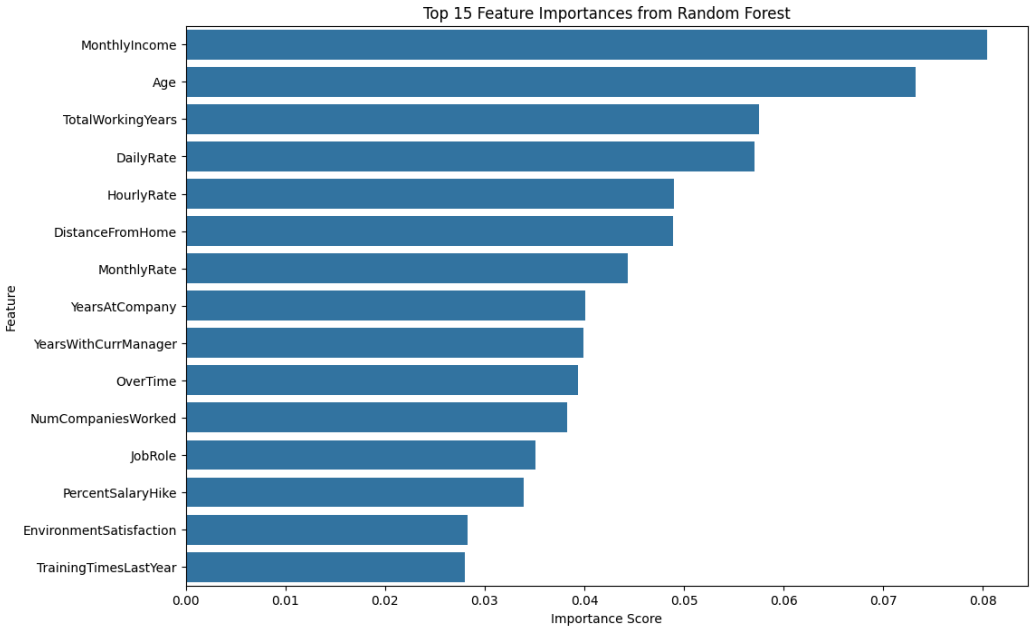
```
Dataset loaded successfully!
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
1   49        No  Travel_Frequently        279  Research & Development
2   37       Yes      Travel_Rarely       1373  Research & Development
3   33        No  Travel_Frequently       1392  Research & Development
4   27        No      Travel_Rarely        591  Research & Development

   DistanceFromHome  Education EducationField  EmployeeCount  EmployeeNumber  \
0                 1          2  Life Sciences              1               1
1                 8          1  Life Sciences              1               2
2                 2          2          Other              1               4
3                 3          4  Life Sciences              1               5
4                 2          1        Medical              1               7

   ...  RelationshipSatisfaction StandardHours  StockOptionLevel  \
0  ...                         1           80                 0
1  ...                         4           80                 1
2  ...                         2           80                 0
3  ...                         3           80                 0
4  ...                         4           80                 1

   TotalWorkingYears  TrainingTimesLastYear WorkLifeBalance  YearsAtCompany  \
0                  8                      0               1               6
1                 10                      3               3              10
2                  7                      3               3               0
3                  8                      3               3               8
4                  6                      3               3               2

   YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                   4                        0                     5
1                   7                        1                     7
2                   0                        0                     0
```

```
Target variable distribution:
Attrition
0    1233
1     237
Name: count, dtype: int64
```




Feature Correlation Heatmap
Age
Attrition
DailyRate
DistanceFromHome
Education
EmployeeCount
EmployeeNumber
EnvironmentSatisfaction
HourlyRate
JobInvolvement
JobLevel
JobSatisfaction
MonthlyIncome
MonthlyRate
NumCompaniesWorked
PercentSalaryHike
PerformanceRating
RelationshipSatisfaction
StandardHours
StockOptionLevel
TotalWorkingYears
TrainingTimesLastYear
WorkLifeBalance
YearsAtCompany
YearsInCurrentRole
YearsSinceLastPromotion
YearsWithCurrManager
1.0
0.8
0.6
0.4
0.2
0.0


Confusion Matrix - Random Forest
240
7
43
4
No Attrition
Attrition
Actual
No Attrition
Attrition
Predicted
200
150
100
50


Top 15 Feature Importances from Random Forest
MonthlyIncome
Age
TotalWorkingYears
DailyRate
HourlyRate
DistanceFromHome
MonthlyRate
YearsAtCompany
YearsWithCurrManager
OverTime
NumCompaniesWorked
JobRole
PercentSalaryHike
EnvironmentSatisfaction
TrainingTimesLastYear
Feature
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Importance Score

References

scikit-learn documentation – https://scikit-learn.org/

pandas documentation – https://pandas.pydata.org/

Seaborn visualization – https://seaborn.pydata.org/

Kaggle

Research papers on employee attrition and workforce analytics