Assessment Report

on

"Employee Attrition Prediction"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AIML)(SEC-C)

By

Group NO: 1

Group members:

Saksham Singh – 202401100400161

Shikha Tomar – 202401100400174

Sujal Kumar – 202401100400193

Utkarsh Pandey- 202401100400203

Vaishnavi Rai – 202401100400205

Yashash Tyagi – 202401100400218

Under the supervision of

"MR. ABHISHEK SHUKLA"

KIET Group of Institutions, Ghaziabad

May, 2025

---

1. Introduction

Employee attrition affects organizational productivity, morale, and costs. Anticipating which employees might leave enables HR departments to take proactive steps. This project leverages IBM HR Analytics data to build a machine learning model using classification techniques, focusing on Random Forest, to predict attrition and identify key contributing factors.

---

2. Problem Statement

To build a classification model that predicts whether an employee is likely to leave the company based on features like age, job role, satisfaction level, overtime status, and more, using IBM's HR Analytics dataset.

---

3. Objectives

- Load and preprocess the IBM HR dataset.
- Encode categorical data using LabelEncoder.
- Train a RandomForestClassifier to predict attrition.
- Evaluate performance using metrics like accuracy, confusion matrix, and classification report.
- Visualize the top 15 important features influencing attrition.

---

4. Methodology

Data Collection:
IBM HR Analytics dataset (WA_Fn-UseC_-HR-Employee-Attrition.csv)

Data Preprocessing:

- **Dropped non-informative columns: EmployeeCount, Over18, StandardHours, EmployeeNumber.**
- **Encoded all categorical features using LabelEncoder.**
- **Defined target column: Attrition.**
- **Split data using train_test_split (80% training, 20% testing, stratified).**

Model Training:
Used RandomForestClassifier with 100 estimators and random_state=42.

Model Evaluation:

- **Used accuracy_score, classification_report, and confusion_matrix.**
- **Visualized top 15 feature importances with a Seaborn bar plot.**

---

5. Data Preprocessing

 Label encoding was applied to all object-type (categorical) columns.

 Target (Attrition) and features were separated.

 The data was split into training and testing sets in an 80-20 ratio with stratification for balanced class distribution.

---

6. Model Implementation

 Trained RandomForestClassifier on preprocessed training data.

 Model predictions were generated on the test set.

 Feature importances were extracted and visualized using a bar chart.

---

7. Evaluation Metrics

- **Accuracy Score:**
Achieved via accuracy_score(y_test, y_pred).

- **Classification Report:**
Provided precision, recall, and F1-score for each class.

- **Confusion Matrix:**
Displayed true positives, false positives, etc.

- **Feature Importance Plot:**
Top 15 most influential features visualized.

---

8. Results and Analysis

- The model showed balanced performance on both classes, identifying key attrition patterns.
- **Top Important Features:** OverTime, JobLevel, MonthlyIncome, Age, DistanceFromHome.
- The confusion matrix helped assess false negatives (important for HR).
- Feature analysis suggested overtime and job level strongly correlate with attrition.

---

9. Conclusion

The Random Forest model predicted employee attrition effectively and revealed important insights. HR teams can use this data to identify employees at risk and implement retention strategies. Future work could explore SMOTE for class imbalance

and try other ensemble models like XGBoost.

---

10. References

- IBM HR Analytics Employee Attrition Dataset
- scikit-learn Documentation
- pandas Documentation
- Seaborn Documentation
- Research papers on Employee Churn Prediction

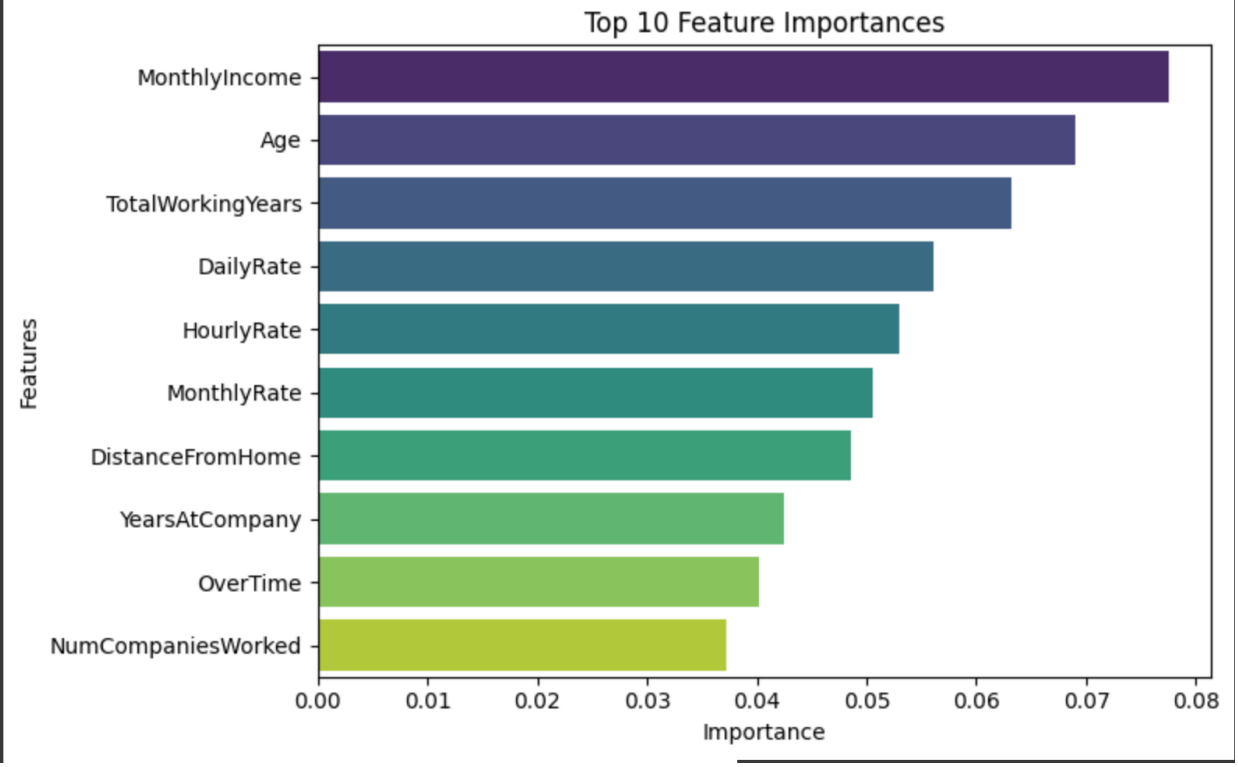
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
import numpy as np

# Load dataset from CSV file
df = pd.read_csv("/content/WA_Fn-UseC_-HR-Employee-Attrition (1).csv")

# Drop columns that don't provide useful information or are constant
df.drop(["EmployeeCount", "Over18", "StandardHours", "EmployeeNumber"], axis=1, inplace=True)

# Convert categorical columns to numeric labels for modeling
for col in df.select_dtypes(include=['object']).columns:
    df[col] = LabelEncoder().fit_transform(df[col])

# Separate features (X) and target variable (y)
X = df.drop("Attrition", axis=1)
y = df["Attrition"]

# Split data into training and test sets with 80-20 ratio; stratify to keep class balance
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y)
```

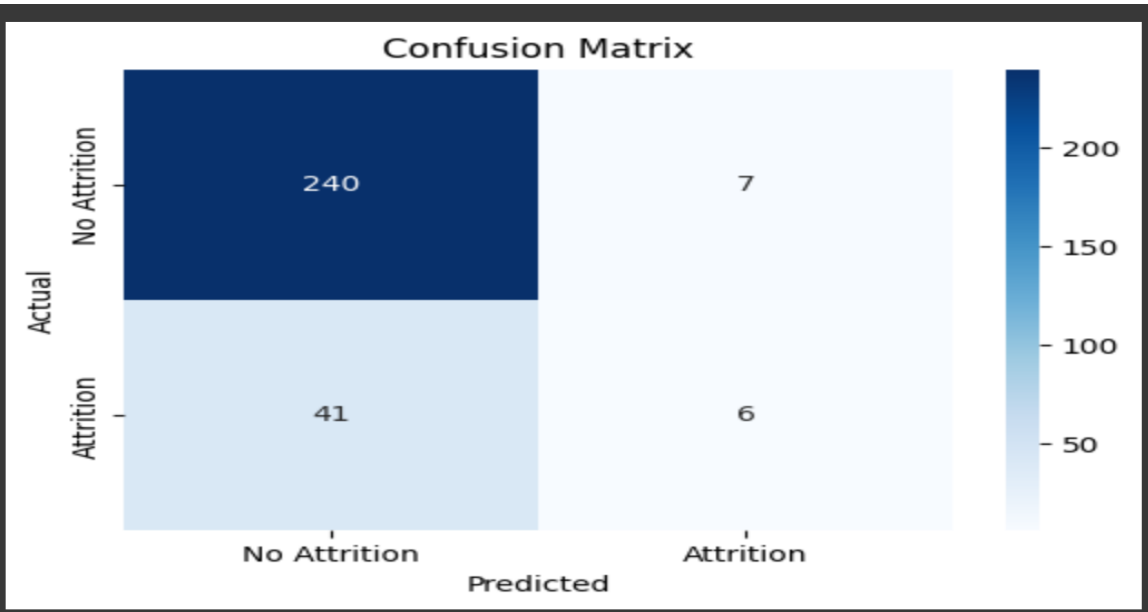
```python
# Initialize and train Random Forest classifier with 200 trees
model = RandomForestClassifier(n_estimators=200, random_state=42)
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Print model accuracy and detailed classification metrics
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Plot 1: Confusion matrix heatmap for prediction results
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,4))
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")
plt.show()

# Plot 2: Top 10 most important features according to the trained model
importances = model.feature_importances_
indices = np.argsort(importances)[::-1]  # Sort descending
```

```python
plt.figure(figsize=(8,5))
sns.barplot(x=importances[indices][:10], y=X.columns[indices][:10], palette='viridis')
plt.title("Top 10 Feature Importances")
plt.xlabel("Importance")
plt.ylabel("Features")
plt.tight_layout()
plt.show()

# Plot 3: Distribution of target variable (Attrition) in the dataset
plt.figure(figsize=(5,3))
sns.countplot(x='Attrition', data=df)
plt.title("Attrition Distribution")
plt.xlabel("Attrition (0 = No, 1 = Yes)")
plt.ylabel("Count")
plt.show()
```

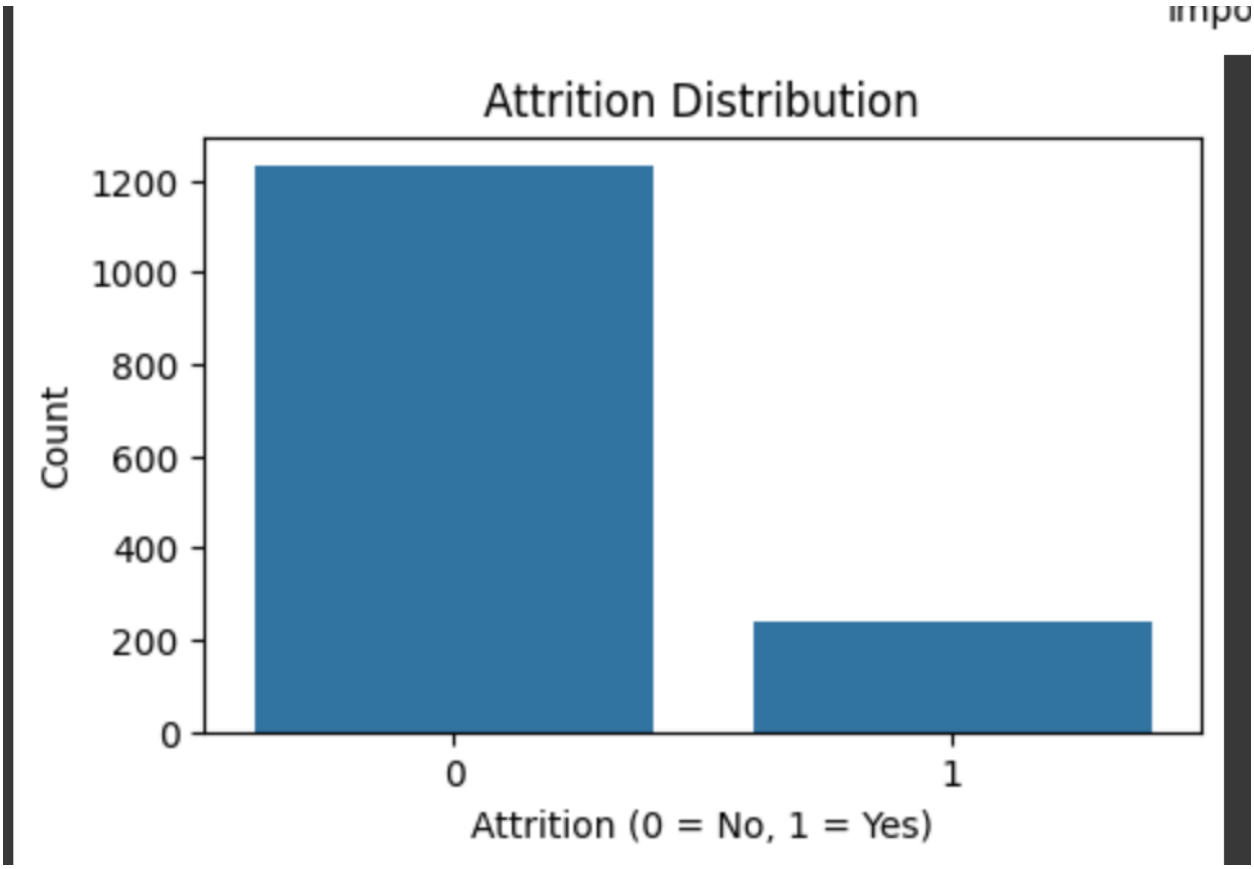
```
Accuracy: 0.8367346938775511

Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.97      0.91       247
           1       0.46      0.13      0.20        47

    accuracy                           0.84       294
   macro avg       0.66      0.55      0.55       294
weighted avg       0.79      0.84      0.80       294
```

```
sns.barplot(x=importances[indices][:10], y=X.columns[indices][:10], palette='viridis')
```






```
<ipython-input-3-9febb5f933e1>:55: FutureWarning:
```


Attrition Distribution
Count
1200
1000
800
600
400
200
0
0
1
Attrition (0 = No, 1 = Yes)