An assessment report

on

“Predict employees attrition”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name :YASHIKA TYAGI(202401100300286)

Under the supervision of

“Mr Abhisekh shukla”

KIET Group of Institutions, Ghaziabad

INTRODUCTION

Employee attrition, or employee turnover, is the process of employees leaving an organization. High attrition rates can have negative impacts, such as increased recruitment costs, loss of skilled workers, and disruption to team performance. As a result, being able to predict which employees are at risk of leaving can help companies take proactive steps to retain talent.

In this project, we aim to develop a machine learning classification model to predict whether an employee is likely to leave the company. We use the **IBM HR Analytics Employee Attrition Dataset**, which contains various attributes related to employee behavior, satisfaction, and job conditions — such as job role, salary, work-life balance, years at the company, and overtime status.

By training a **Random Forest Classifier**, we analyze patterns in the data to make accurate predictions about attrition. This can assist HR departments in identifying at-risk employees and implementing strategies to improve retention, reduce turnover costs, and enhance employee satisfaction.

METHODOLOGY

1. **Data Collection**: Load the employee attrition dataset using `pandas`.
2. **Data Preprocessing**:
    - Drop irrelevant columns (`EmployeeCount`, `EmployeeNumber`, `Over18`, `StandardHours`).
    - Encode categorical variables (e.g., `Attrition`) using `LabelEncoder`.
3. **Feature and Label Preparation**:
    - Separate features (`X`) and target variable (`y`).
4. **Train-Test Split**:
    - Split the dataset into training (80%) and testing (20%) sets using `train_test_split()`.
5. **Feature Scaling**:
    - Normalize the data using `StandardScaler`.
6. **Model Training**:
    - Train a Random Forest Classifier on the scaled training data.
7. **Prediction and Evaluation**:

- Make predictions on the test data and evaluate model performance using accuracy, classification report, and confusion matrix.

8. **Visualization**:
   - Display a heatmap of the confusion matrix for model evaluation.

CODE

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

# Step 1: Load the dataset (Download from Kaggle:
# https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-
attrition-dataset)
df = pd.read_csv('6. Predict Employee Attrition.csv')

# Step 2: Drop unnecessary columns
df = df.drop(['EmployeeCount', 'EmployeeNumber', 'Over18',
'StandardHours'], axis=1)

# Step 3: Encode categorical features
# Encode target variable first
df['Attrition'] = df['Attrition'].map({'Yes': 1, 'No': 0})
# Encode other categorical columns using LabelEncoder
le = LabelEncoder()
categorical_cols = df.select_dtypes(include='object').columns
for col in categorical_cols:
    df[col] = le.fit_transform(df[col])

# Step 4: Prepare features and labels
X = df.drop('Attrition', axis=1)
y = df['Attrition']
```

```
# Step 5: Train/test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)

# Step 6: Scale features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Step 7: Train a Random Forest model
model = RandomForestClassifier(random_state=42)
model.fit(X_train_scaled, y_train)

# Step 8: Make predictions
y_pred = model.predict(X_test_scaled)

# Step 9: Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: {:.2f}%".format(accuracy * 100))
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Step 10: Generate and display the confusion matrix heatmap
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 Heatmap')
plt.tight_layout()
plt.show()
```

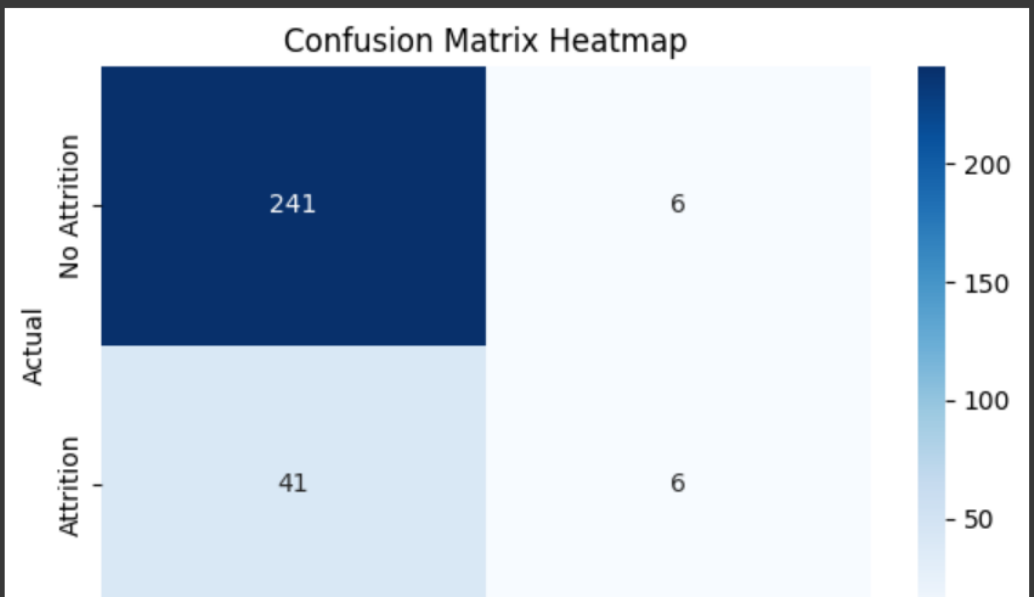
OUTPUT

```
Accuracy: 84.01%

Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.98      0.91       247
           1       0.50      0.13      0.20        47

    accuracy                           0.84       294
   macro avg       0.68      0.55      0.56       294
weighted avg       0.80      0.84      0.80       294
```



REFERENCE AND CREDITS

- **Dataset Credit**:
  - **IBM HR Analytics Attrition Dataset** from Kaggle: IBM HR Analytics Dataset.
- **Libraries**:
  - **Pandas**: McKinney, W. (2010). *Data structures for statistical computing in Python*.
  - **Matplotlib**: Hunter, J. D. (2007). *Matplotlib: A 2D graphics environment*.
  - **Seaborn**: Waskom, M. L., et al. (2020). *seaborn: statistical data visualization*.
  - **Scikit-Learn**: Pedregosa, F., et al. (2011). *Scikit-learn: Machine learning in Python*.