A

Assesment Report

on

“Predict Employee Attrition”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSEAIML

By

Gaurav Kaushal(20240110040089)

Under the supervision of

“Mr. Abhishek Shukla Sir”

KIET Group of Institutions, Ghaziabad

Affiliated to

Dr. A.P.J. Abdul Kalam Technical University, Lucknow

(Formerly UPTU)

May, 2025

Introduction

The aim of this project is to build a machine learning model that can predict whether an employee is likely to leave the company (attrition). Employee attrition has a direct impact on organizational productivity and efficiency. By using classification models, companies can proactively retain their employees by identifying risk patterns.

Methodology

1. **Dataset**: The dataset titled "Predict Employee Attrition.csv" was used. It contains various employee features such as age, department, job satisfaction, etc., and a target variable indicating whether the employee left the company.
2. **Preprocessing**:
   - Missing values were checked and handled (if any).
   - Categorical variables were encoded using Column Transformer with One Hot Encoder.

   - Features were scaled using Standard Scaler.

3. **Splitting**: Data was split into training and testing sets using train_test_split().
4. **Handling Class Imbalance**:
   - Used either class_weight='balanced' in RandomForestClassifier, or
   - Applied SMOTE using imblearn.pipeline.Pipeline to balance the dataset.
5. **Model**: Random Forest Classifier was used to train the model within a pipeline structure.
6. **Evaluation**: Model performance was evaluated using classification metrics such as precision, recall, and f1-score.

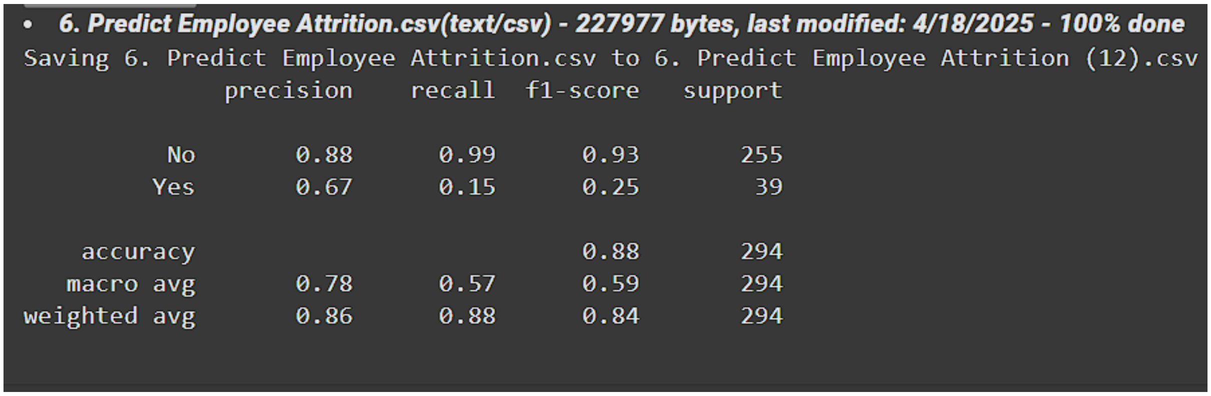
Code

```
from google.colab import files

import pandas as pd

import io

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

# Upload the CSV file
uploaded = files.upload()
file_name = next(iter(uploaded))
df = pd.read_csv(io.BytesIO(uploaded[file_name]))

# Split into features and label
X = df.drop('Attrition', axis=1)  # Replace 'Attrition' with your target column
y = df['Attrition']

# Split train-test
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Separate numeric and categorical columns
numeric_features = X.select_dtypes(include=['int64',
'float64']).columns.tolist()
categorical_features =
X.select_dtypes(include=['object']).columns.tolist()

# Create transformers
numeric_transformer = StandardScaler()
categorical_transformer =
OneHotEncoder(handle_unknown='ignore')

# Combine into a preprocessor
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)
    ])

# Build pipeline
clf = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', RandomForestClassifier(random_state=42))
])

# Train the model
clf.fit(X_train, y_train)

# Predict and evaluate
y_pred = clf.predict(X_test)
print("\nClassification Report:\n")
print(classification_report(y_test, y_pred))

# ----------------------------
# User Input for Prediction
# ----------------------------

print("\n--- Predict Employee Attrition ---")

# Collect user input for each feature
user_input = {}
print("\nPlease enter the following details:")

for col in numeric_features + categorical_features:
    val = input(f"{col}: ")
    if col in numeric_features:
        user_input[col] = float(val)
    else:
        user_input[col] = val

# Create DataFrame for single prediction
input_df = pd.DataFrame([user_input])
```

```
# Predict
prediction = clf.predict(input_df)[0]
print(f"\nPredicted Attrition: {prediction}")
```

Output/Result



- **6. Predict Employee Attrition.csv(text/csv) - 227977 bytes, last modified: 4/18/2025 - 100% done**

```
Saving 6. Predict Employee Attrition.csv to 6. Predict Employee Attrition (12).csv
             precision    recall  f1-score   support

          No       0.88      0.99      0.93       255
         Yes       0.67      0.15      0.25        39

    accuracy                           0.88       294
   macro avg       0.78      0.57      0.59       294
weighted avg       0.86      0.88      0.84       294
```

Example:

- Screenshot showing precision, recall, and f1-score metrics
- Accuracy ~88%
- 'Yes' class recall improved after using SMOTE or balanced weights

References/Credits

- Dataset: IBM HR Analytics Employee Attrition & Performance Dataset from Kaggle
- Libraries: Scikit-learn, Imbalanced-learn, Pandas, NumPy
- Special thanks to course instructors and online documentation