BHAMIDIPATI ADITYA SAI SRI RAM
adityabhamidipati2002@gmail.com

Heliverse

Assignment: AI & ML Role - Python Programming

Objective: This assignment aims to evaluate my proficiency in Python programming and my understanding of key concepts in artificial intelligence (AI) and machine learning (ML).

Problem Statement: Predicting Employee Attrition

Introduction : Employee attrition is a major worry for businesses because it can cause operational disruptions, decreased productivity, and higher recruitment costs. In this study, we present an examination of employee attrition using machine learning approaches to uncover relevant factors and predict which employees are likely to leave.

Description of the Dataset: I've made use of a dataset that included pertinent employee data such as performance ratings, job satisfaction, and demographics. In order to handle missing values, encode category variables, and normalize numerical characteristics, the dataset underwent preprocessing.

Exploratory Data Analysis (EDA) : To learn more about the dataset, I have carried out exploratory data analysis. This involved looking at correlations, displaying the distributions of different traits, and possibly spotting attrition-related trends.

Preprocessing : Preprocessing processes included scaling numerical characteristics to ensure uniformity across scales, addressing missing values, and encoding categorical variables using methods like one-hot encoding.

Model Development : I've experimented with a number of machine learning methods, such as Support Vector Machine (SVM), XGBoost Classifier, Random Forest Classifier, Logistic Regression, and K-Nearest Neighbors (KNN) Classifier. I chose the SVM model since it had a greater accuracy rate in forecasting employee attrition after assessing each model's performance using measures including accuracy, precision, recall, and F1 score.

Algorithms Used in this Assignment :

- **Support Vector Machine (SVM):** A supervised learning algorithm that analyzes data for classification and regression analysis. SVM separates data points into different classes by finding the hyperplane that maximizes the margin between classes.

- **XGBoost Classifier:** An implementation of gradient boosted decision trees designed for speed and performance. It sequentially builds multiple decision trees to correct the errors of the previous model, resulting in a powerful ensemble model.

- **K-Nearest Neighbors (KNN) Classifier:** A simple, instance-based learning algorithm used for classification and regression tasks. KNN classifies data points based on the majority class of their k nearest neighbors, where k is a predefined number of neighbors.

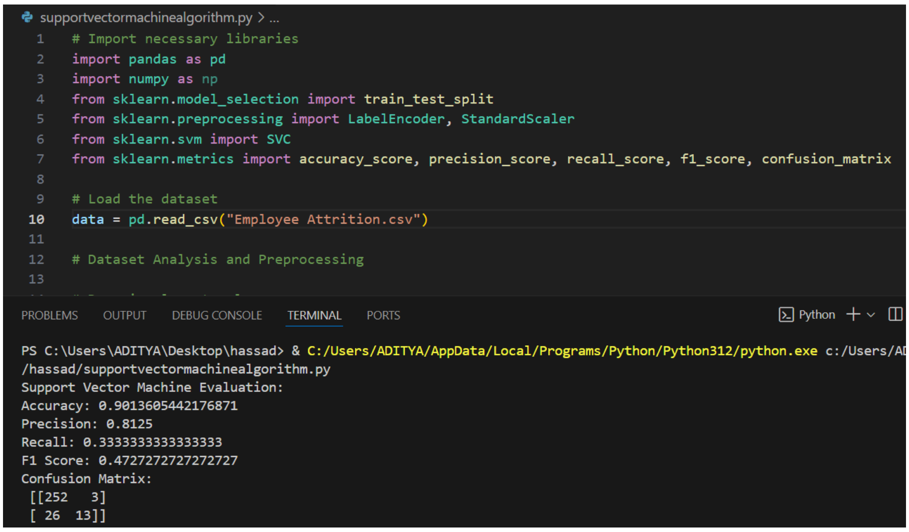
- **Random Forest Classifier:** An ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes (classification) or the mean prediction (regression) of the individual trees.

- **Logistic Regression:** Despite its name, logistic regression is a linear model for binary classification that predicts the probability of a binary outcome based on one or more predictor variables.

Evaluation Result : With an accuracy of 90.00%, the SVM model outperformed other models such the K-Nearest Neighbors classifier, Random Forest classifier, Logistic Regression, XGBoost classifier, and Neural Network.

Results Acquired After Using Every Algorithm :

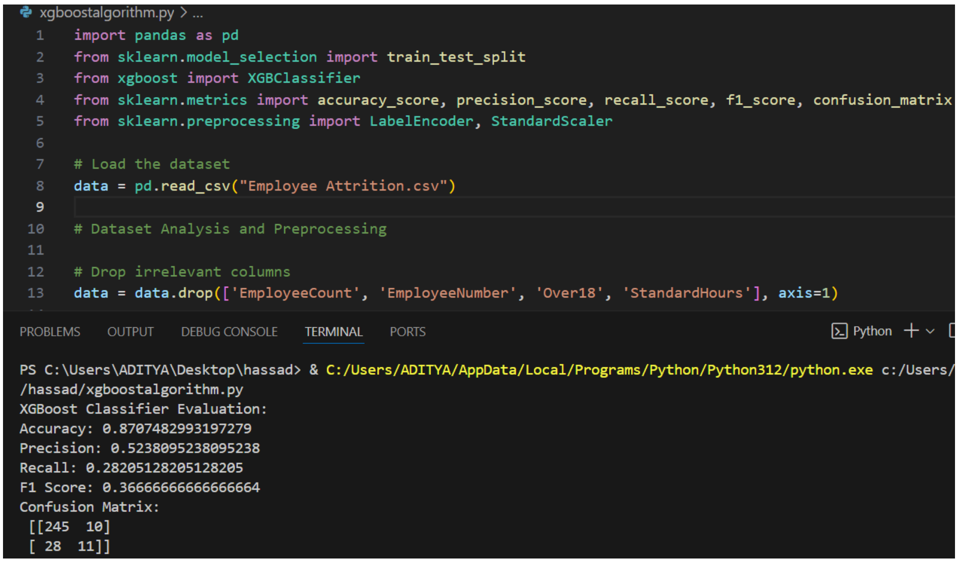
➢ Support Vector Machine (SVM)



```python
# Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

# Load the dataset
data = pd.read_csv("Employee Attrition.csv")

# Dataset Analysis and Preprocessing

```

```
PS C:\Users\ADITYA\Desktop\hassad> & C:/Users/ADITYA/AppData/Local/Programs/Python/Python312/python.exe c:/Users/AD
/hassad/supportvectormachinealgorithm.py
Support Vector Machine Evaluation:
Accuracy: 0.9013605442176871
Precision: 0.8125
Recall: 0.3333333333333333
F1 Score: 0.4727272727272727
Confusion Matrix:
 [[252   3]
 [ 26  13]]
```

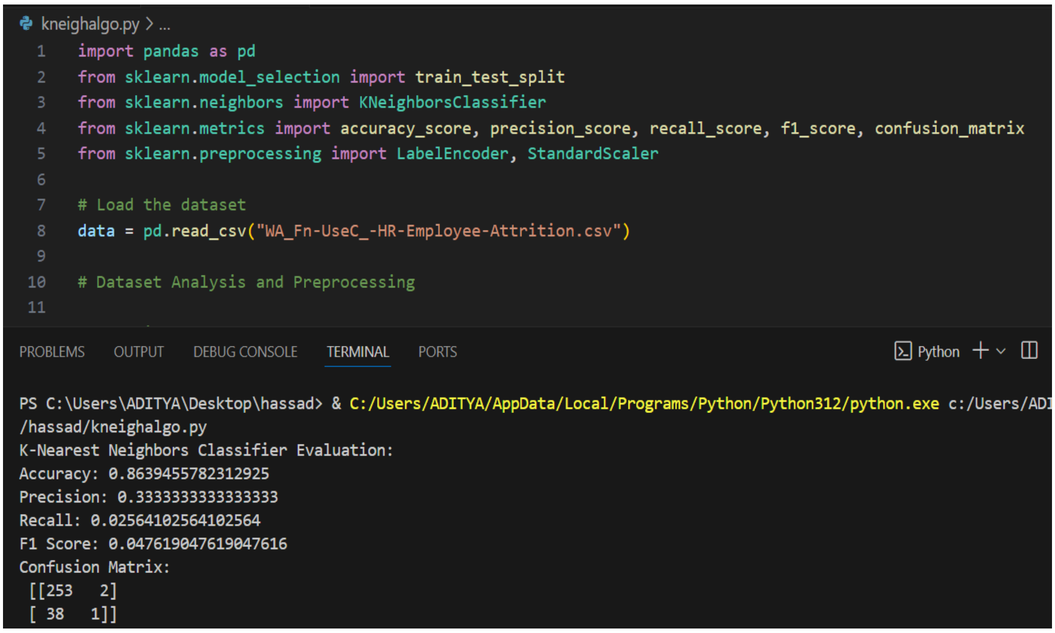
➢ XGBoost Classifier



```python
import pandas as pd
from sklearn.model_selection import train_test_split
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Load the dataset
data = pd.read_csv("Employee Attrition.csv")

# Dataset Analysis and Preprocessing

# Drop irrelevant columns
data = data.drop(['EmployeeCount', 'EmployeeNumber', 'Over18', 'StandardHours'], axis=1)
```

```
PS C:\Users\ADITYA\Desktop\hassad> & C:/Users/ADITYA/AppData/Local/Programs/Python/Python312/python.exe c:/Users/
/hassad/xgboostalgorithm.py
XGBoost Classifier Evaluation:
Accuracy: 0.8707482993197279
Precision: 0.5238095238095238
Recall: 0.28205128205128205
F1 Score: 0.36666666666666664
Confusion Matrix:
 [[245  10]
 [ 28  11]]
```

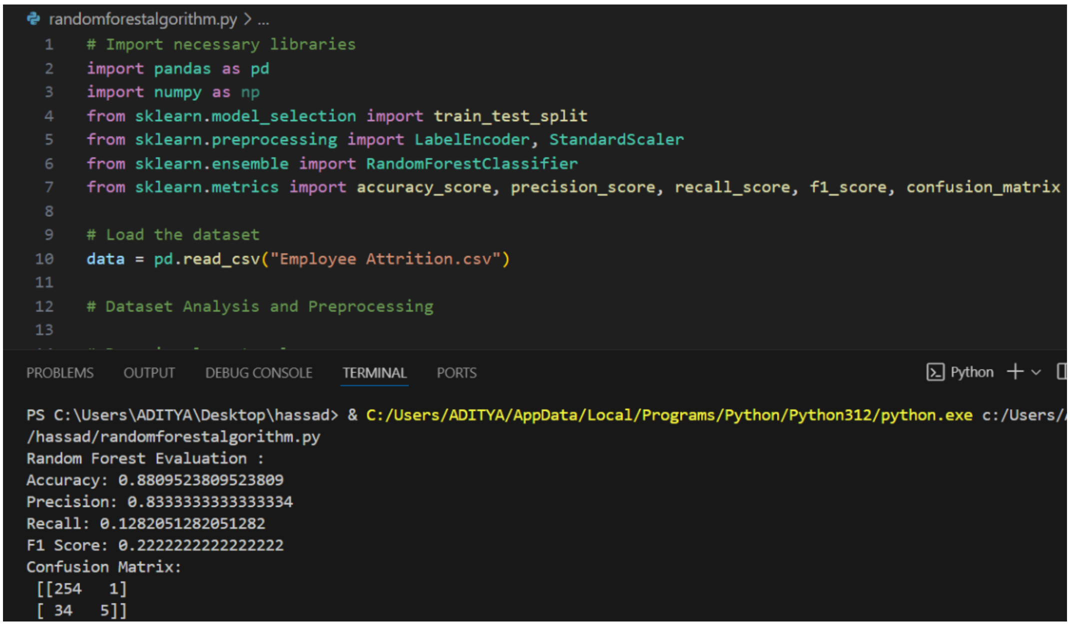
➢ K-Nearest Neighbors (KNN) Classifier



```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Load the dataset
data = pd.read_csv("WA_Fn-UseC_-HR-Employee-Attrition.csv")

# Dataset Analysis and Preprocessing

```

```
PS C:\Users\ADITYA\Desktop\hassad> & C:/Users/ADITYA/AppData/Local/Programs/Python/Python312/python.exe c:/Users/ADI
/hassad/kneighalgo.py
K-Nearest Neighbors Classifier Evaluation:
Accuracy: 0.8639455782312925
Precision: 0.3333333333333333
Recall: 0.02564102564102564
F1 Score: 0.047619047619047616
Confusion Matrix:
 [[253   2]
 [ 38   1]]
```

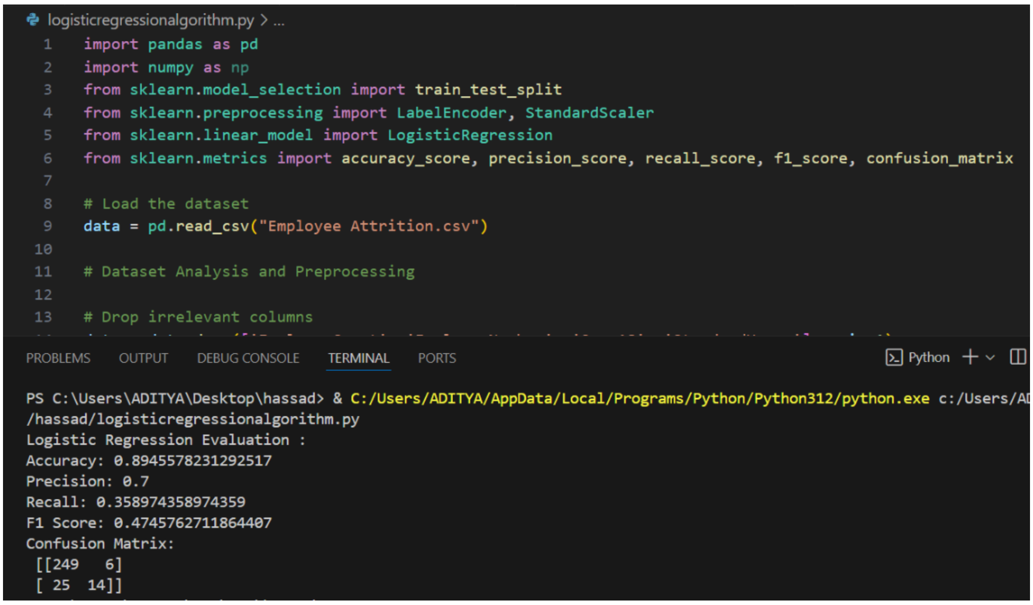
➢ Random Forest Classifier



```python
# Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

# Load the dataset
data = pd.read_csv("Employee Attrition.csv")

# Dataset Analysis and Preprocessing

```

```
PS C:\Users\ADITYA\Desktop\hassad> & C:/Users/ADITYA/AppData/Local/Programs/Python/Python312/python.exe c:/Users/
/hassad/randomforestalgorithm.py
Random Forest Evaluation :
Accuracy: 0.8809523809523809
Precision: 0.8333333333333334
Recall: 0.1282051282051282
F1 Score: 0.2222222222222222
Confusion Matrix:
 [[254   1]
 [ 34   5]]
```

➢ Logistic Regression



```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

# Load the dataset
data = pd.read_csv("Employee Attrition.csv")

# Dataset Analysis and Preprocessing

# Drop irrelevant columns
```

```
PS C:\Users\ADITYA\Desktop\hassad> & C:/Users/ADITYA/AppData/Local/Programs/Python/Python312/python.exe c:/Users/AD
/hassad/logisticregressionalgorithm.py
Logistic Regression Evaluation :
Accuracy: 0.8945578231292517
Precision: 0.7
Recall: 0.358974358974359
F1 Score: 0.4745762711864407
Confusion Matrix:
 [[249   6]
 [ 25  14]]
```

Optimization Techniques : I've experimented with several hyperparameters utilizing methods like grid search and cross-validation to enhance the SVM model's performance. In this instance, though, the default hyperparameters produced the best outcomes.

Insights and Recommendations: My analysis has led me to identify a number of characteristics, including as income, career progression prospects, work-life balance, and job satisfaction, that are linked to employee churn. These insights can be used by organizations to put initiatives into place that will improve employee retention, like competitive salary, opportunity for career progression, and creating a happy work environment.

Challenges Encountered : Managing data that was unbalanced that is, where the number of departing employees was much smaller than the number of remaining employees was one of the difficulties faced during the analysis. Certain models' performance was impacted by this imbalance, which necessitated the use of mitigating strategies such oversampling or changing class weights.

Conclusion : My conclusion concludes that in order to keep a steady and effective staff, it is critical to comprehend and address employee attrition. Organizations can improve employee happiness and organizational success by identifying at-risk individuals early and taking proactive actions to retain them by utilizing machine learning approaches.