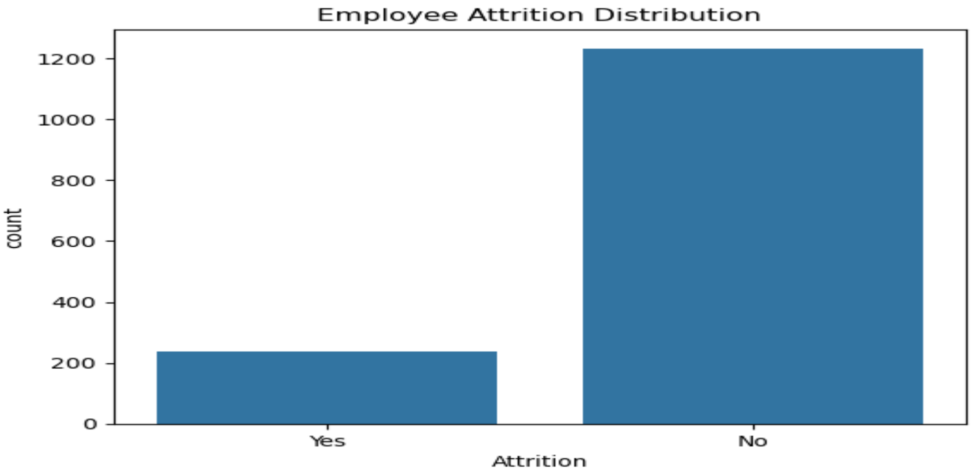
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# ML
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

# SHAP for explainability
import shap


import pandas as pd

df = pd.read_csv("C:/Users/Balambiga2910/Documents/hr dataset.csv")
df.head()
```

```
   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
3                  7                        3                     0
4                  2                        2                     2

[5 rows x 35 columns]
```

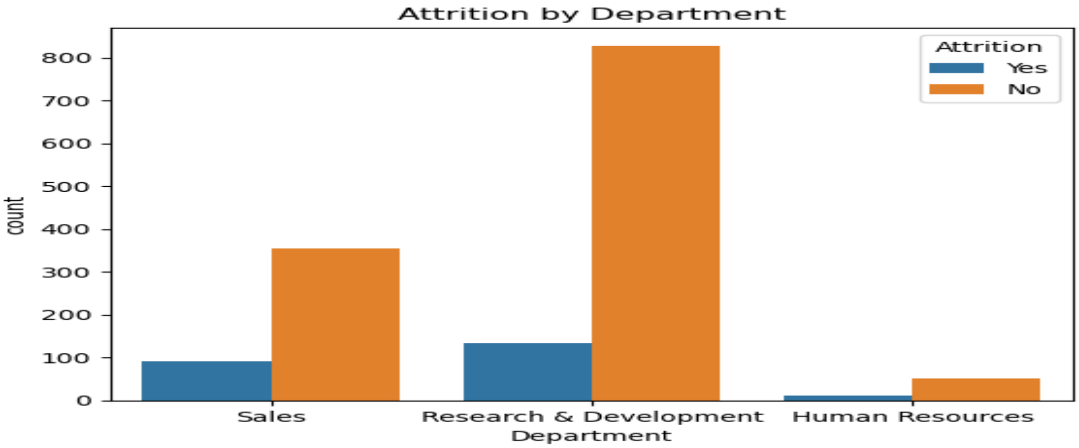
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.countplot(x='Attrition', data=df)
plt.title("Employee Attrition Distribution")
```

```
Text(0.5, 1.0, 'Employee Attrition Distribution')
```



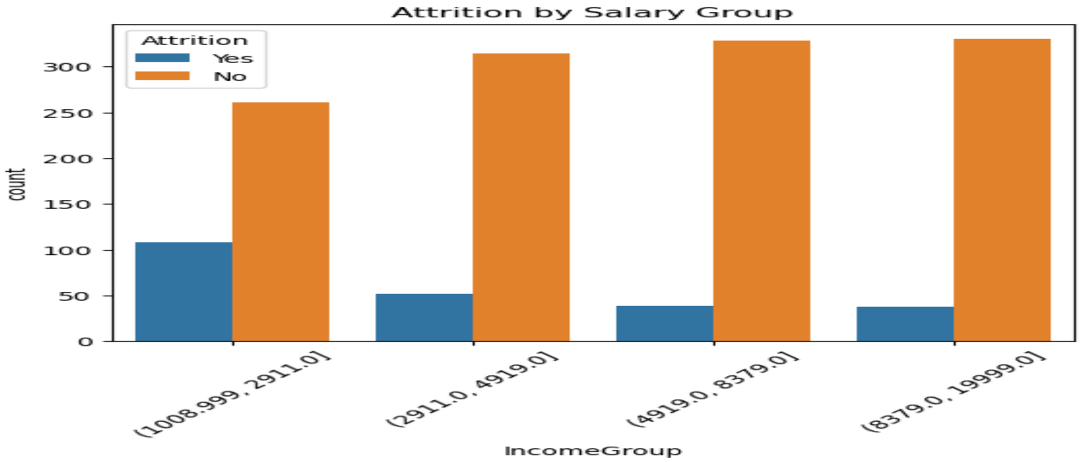
```
sns.countplot(x='Department', hue='Attrition', data=df)
plt.title("Attrition by Department")
```

```
Text(0.5, 1.0, 'Attrition by Department')
```



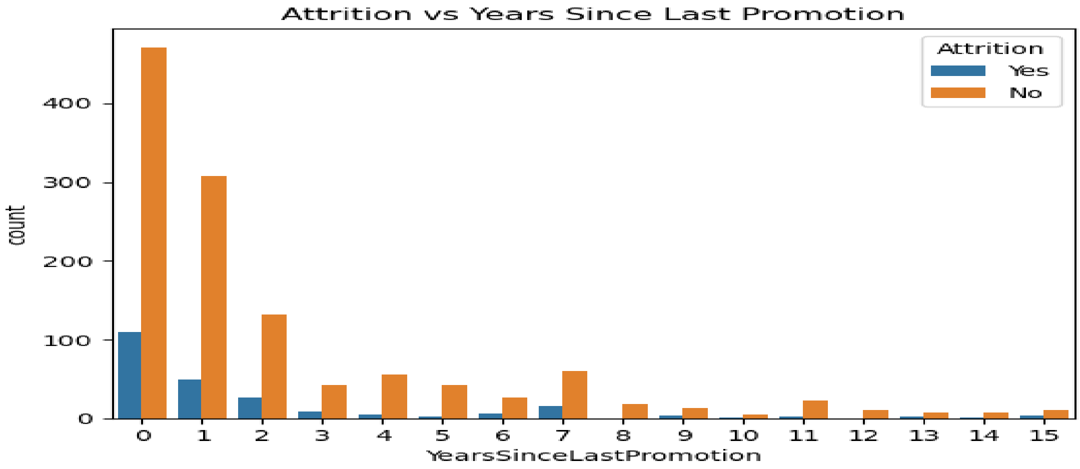
```
df['IncomeGroup'] = pd.qcut(df['MonthlyIncome'], 4)
sns.countplot(x='IncomeGroup', hue='Attrition', data=df)
plt.title("Attrition by Salary Group")
plt.xticks(rotation=45)
```

```
([0, 1, 2, 3],
 [Text(0, 0, '(1008.999, 2911.0]'),
  Text(1, 0, '(2911.0, 4919.0]'),
  Text(2, 0, '(4919.0, 8379.0]'),
  Text(3, 0, '(8379.0, 19999.0]')])
```



```
sns.countplot(x='YearsSinceLastPromotion', hue='Attrition', data=df)
plt.title("Attrition vs Years Since Last Promotion")
```

```
Text(0.5, 1.0, 'Attrition vs Years Since Last Promotion')
```



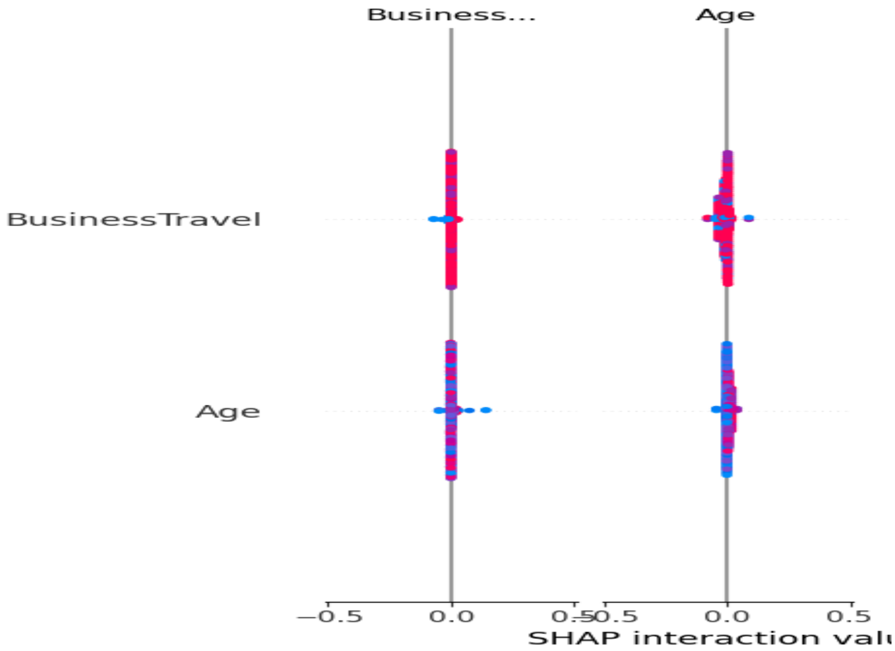
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
# Drop irrelevant columns
```

```
cols_to_drop = ['EmployeeNumber', 'Over18', 'StandardHours', 'EmployeeCount']
existing_cols = [col for col in cols_to_drop if col in df.columns]
df.drop(existing_cols, axis=1, inplace=True)

# Encode categorical columns
le = LabelEncoder()
for column in df.select_dtypes(include='object').columns:
    df[column] = le.fit_transform(df[column])

# Split data
X = df.drop('Attrition', axis=1)
y = df['Attrition']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
tree_model = DecisionTreeClassifier(max_depth=5, random_state=42)
tree_model.fit(X_train, y_train)
y_pred_tree = tree_model.predict(X_test)

print("Decision Tree Accuracy:", accuracy_score(y_test, y_pred_tree))
print(classification_report(y_test, y_pred_tree))
```

```
Decision Tree Accuracy: 0.826530612244898
              precision    recall  f1-score   support

           0       0.88      0.93      0.90       255
           1       0.25      0.15      0.19        39

    accuracy                           0.83       294
   macro avg       0.56      0.54      0.55       294
weighted avg       0.79      0.83      0.81       294
```

```
import shap

explainer = shap.TreeExplainer(tree_model)
shap_values = explainer.shap_values(X_test)

# Check structure
print("Length of shap_values:", len(shap_values))  # Debug print

# If it's a list of 2 elements, use [1]
if isinstance(shap_values, list) and len(shap_values) == 2:
    shap.summary_plot(shap_values[1], X_test)
else:
    shap.summary_plot(shap_values, X_test)
```

```
Length of shap_values: 294
```

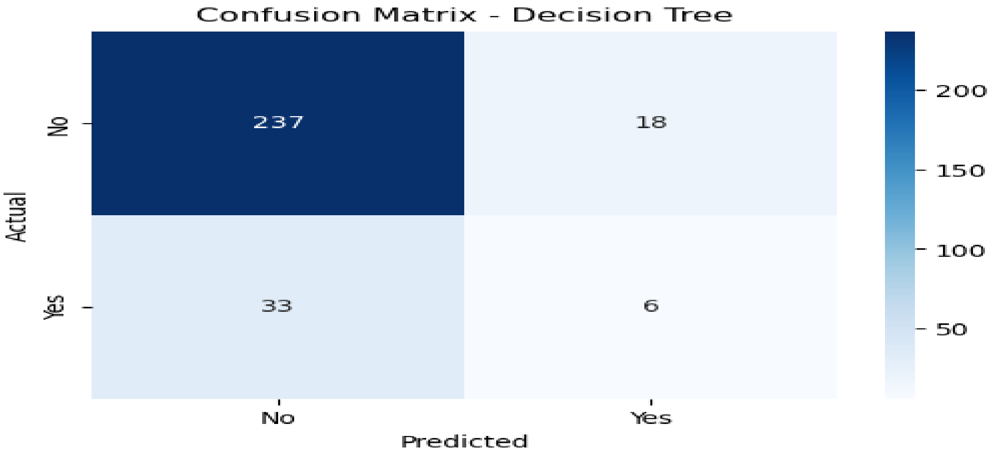


```
# 1. Import Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# 2. Load the Dataset
df = pd.read_csv("C:/Users/Balambiga2910/Documents/hr dataset.csv")
```

```
# 3. Drop Irrelevant Columns
cols_to_drop = ['EmployeeNumber', 'Over18', 'StandardHours', 'EmployeeCount']
df.drop([col for col in cols_to_drop if col in df.columns], axis=1, inplace=True)

# 4. Encode Categorical Columns
le = LabelEncoder()
for column in df.select_dtypes(include='object').columns:
    df[column] = le.fit_transform(df[column])

# 5. Split into Features and Target
X = df.drop('Attrition', axis=1)
y = df['Attrition']

# 6. Train/Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 7. Train the Model
tree_model = DecisionTreeClassifier(max_depth=5, random_state=42)
tree_model.fit(X_train, y_train)

# 8. Make Predictions
y_pred_tree = tree_model.predict(X_test)

# 9. Print Accuracy & Classification Report
print("Decision Tree Accuracy:", accuracy_score(y_test, y_pred_tree))
print(classification_report(y_test, y_pred_tree))
```

```
# 10. Confusion Matrix Plot
cm = confusion_matrix(y_test, y_pred_tree)
plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=['No', 'Yes'], yticklabels=['No', 'Yes'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - Decision Tree')
plt.show()
```

```
Decision Tree Accuracy: 0.826530612244898
              precision    recall  f1-score   support

           0       0.88      0.93      0.90       255
           1       0.25      0.15      0.19        39

    accuracy                           0.83       294
   macro avg       0.56      0.54      0.55       294
weighted avg       0.79      0.83      0.81       294
```


Confusion Matrix - Decision Tree
No
Yes
Actual
No
Yes
Predicted