```python
#Import necessary libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df=pd.read_csv("D:\\Data_Science_Intern\\Attrition data.csv")

df.head()
```

```
   EmployeeID  Age Attrition     BusinessTravel
Department  \
0           1   51        No      Travel_Rarely
Sales
1           2   31       Yes  Travel_Frequently  Research &
Development
2           3   32        No  Travel_Frequently  Research &
Development
3           4   38        No         Non-Travel  Research &
Development
4           5   32        No      Travel_Rarely  Research &
Development

   DistanceFromHome  Education EducationField  EmployeeCount
Gender  ...  \
0                 6          2  Life Sciences              1
Female  ...
1                10          1  Life Sciences              1
Female  ...
2                17          4          Other              1
Male  ...
3                 2          5  Life Sciences              1
Male  ...
4                10          1        Medical              1
Male  ...

   TotalWorkingYears  TrainingTimesLastYear  YearsAtCompany  \
0                1.0                      6               1
1                6.0                      3               5
2                5.0                      2               5
3               13.0                      5               8
4                9.0                      2               6

   YearsSinceLastPromotion  YearsWithCurrManager
EnvironmentSatisfaction  \
0                        0                     0
3.0
1                        1                     4
3.0
2                        0                     3
2.0
```

```
3                          7                        5
4.0
4                          0                        4
4.0

   JobSatisfaction  WorkLifeBalance  JobInvolvement  PerformanceRating

0              4.0              2.0               3                  3

1              2.0              4.0               2                  4

2              2.0              1.0               3                  3

3              4.0              3.0               2                  3

4              1.0              3.0               3                  3


[5 rows x 29 columns]

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4410 entries, 0 to 4409
Data columns (total 29 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   EmployeeID               4410 non-null   int64
 1   Age                      4410 non-null   int64
 2   Attrition                4410 non-null   object
 3   BusinessTravel           4410 non-null   object
 4   Department               4410 non-null   object
 5   DistanceFromHome         4410 non-null   int64
 6   Education                4410 non-null   int64
 7   EducationField           4410 non-null   object
 8   EmployeeCount            4410 non-null   int64
 9   Gender                   4410 non-null   object
 10  JobLevel                 4410 non-null   int64
 11  JobRole                  4410 non-null   object
 12  MaritalStatus            4410 non-null   object
 13  MonthlyIncome            4410 non-null   int64
 14  NumCompaniesWorked       4391 non-null   float64
 15  Over18                   4410 non-null   object
 16  PercentSalaryHike        4410 non-null   int64
 17  StandardHours            4410 non-null   int64
 18  StockOptionLevel         4410 non-null   int64
 19  TotalWorkingYears        4401 non-null   float64
 20  TrainingTimesLastYear    4410 non-null   int64
 21  YearsAtCompany           4410 non-null   int64
 22  YearsSinceLastPromotion  4410 non-null   int64
```

```
 23  YearsWithCurrManager     4410 non-null   int64
 24  EnvironmentSatisfaction  4385 non-null   float64
 25  JobSatisfaction          4390 non-null   float64
 26  WorkLifeBalance          4372 non-null   float64
 27  JobInvolvement           4410 non-null   int64
 28  PerformanceRating        4410 non-null   int64
dtypes: float64(5), int64(16), object(8)
memory usage: 999.3+ KB

df.describe()

        EmployeeID          Age  DistanceFromHome    Education
EmployeeCount  \
count  4410.000000  4410.000000       4410.000000  4410.000000
4410.0
mean   2205.500000    36.923810          9.192517     2.912925
1.0
std    1273.201673     9.133301          8.105026     1.023933
0.0
min       1.000000    18.000000          1.000000     1.000000
1.0
25%    1103.250000    30.000000          2.000000     2.000000
1.0
50%    2205.500000    36.000000          7.000000     3.000000
1.0
75%    3307.750000    43.000000         14.000000     4.000000
1.0
max    4410.000000    60.000000         29.000000     5.000000
1.0

          JobLevel  MonthlyIncome  NumCompaniesWorked
PercentSalaryHike  \
count  4410.000000    4410.000000         4391.000000
4410.000000
mean      2.063946   65029.312925            2.694830
15.209524
std       1.106689   47068.888559            2.498887
3.659108
min       1.000000   10090.000000            0.000000
11.000000
25%       1.000000   29110.000000            1.000000
12.000000
50%       2.000000   49190.000000            2.000000
14.000000
75%       3.000000   83800.000000            4.000000
18.000000
max       5.000000  199990.000000            9.000000
25.000000

       StandardHours  ...  TotalWorkingYears  TrainingTimesLastYear  \
```

```
count         4410.0  ...        4401.000000             4410.000000
mean             8.0  ...          11.279936                2.799320
std              0.0  ...           7.782222                1.288978
min              8.0  ...           0.000000                0.000000
25%              8.0  ...           6.000000                2.000000
50%              8.0  ...          10.000000                3.000000
75%              8.0  ...          15.000000                3.000000
max              8.0  ...          40.000000                6.000000

       YearsAtCompany  YearsSinceLastPromotion
YearsWithCurrManager  \
count     4410.000000              4410.000000           4410.000000

mean         7.008163                 2.187755              4.123129

std          6.125135                 3.221699              3.567327

min          0.000000                 0.000000              0.000000

25%          3.000000                 0.000000              2.000000

50%          5.000000                 1.000000              3.000000

75%          9.000000                 3.000000              7.000000

max         40.000000                15.000000             17.000000


       EnvironmentSatisfaction  JobSatisfaction  WorkLifeBalance  \
count              4385.000000      4390.000000      4372.000000
mean                  2.723603         2.728246         2.761436
std                   1.092756         1.101253         0.706245
min                   1.000000         1.000000         1.000000
25%                   2.000000         2.000000         2.000000
50%                   3.000000         3.000000         3.000000
75%                   4.000000         4.000000         3.000000
max                   4.000000         4.000000         4.000000

       JobInvolvement  PerformanceRating
count     4410.000000        4410.000000
mean         2.729932           3.153741
std          0.711400           0.360742
min          1.000000           3.000000
25%          2.000000           3.000000
50%          3.000000           3.000000
75%          3.000000           3.000000
max          4.000000           4.000000

[8 rows x 21 columns]

df.shape
```

```
(4410, 29)
```

```
#Checking for missing values
df.isnull().sum()
```

```
EmployeeID                  0
Age                         0
Attrition                   0
BusinessTravel              0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
Gender                      0
JobLevel                    0
JobRole                     0
MaritalStatus               0
MonthlyIncome               0
NumCompaniesWorked         19
Over18                      0
PercentSalaryHike           0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           9
TrainingTimesLastYear       0
YearsAtCompany              0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
EnvironmentSatisfaction    25
JobSatisfaction            20
WorkLifeBalance            38
JobInvolvement              0
PerformanceRating           0
dtype: int64
```

```
df['Attrition'].value_counts()
```

```
Attrition
No     3699
Yes     711
Name: count, dtype: int64
```

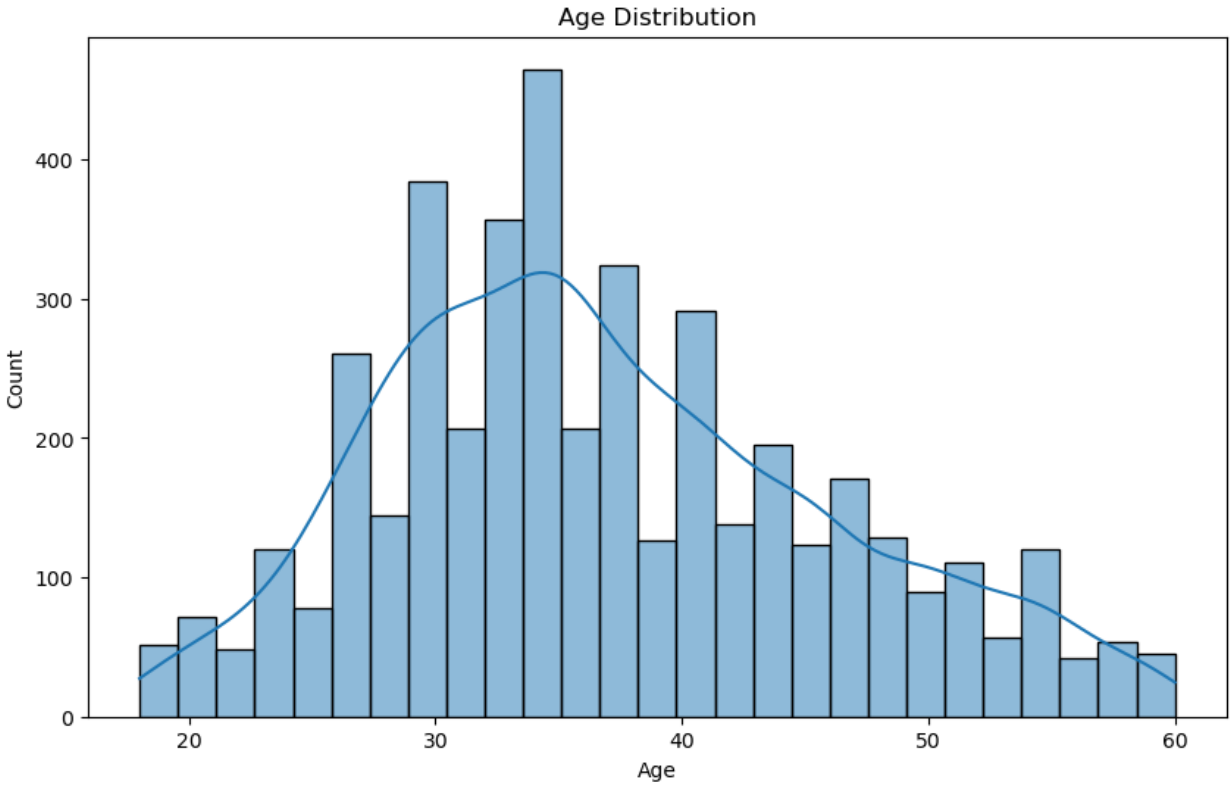
```
# Impute missing values for numerical columns with median
df['NumCompaniesWorked'].fillna(df['NumCompaniesWorked'].median(),
inplace=True)
df['TotalWorkingYears'].fillna(df['TotalWorkingYears'].median(),
inplace=True)

# Impute missing values for categorical columns with mode
```

```
df['EnvironmentSatisfaction'].fillna(df['EnvironmentSatisfaction'].mod
e()[0], inplace=True)
df['JobSatisfaction'].fillna(df['JobSatisfaction'].mode()[0],
inplace=True)
df['WorkLifeBalance'].fillna(df['WorkLifeBalance'].mode()[0],
inplace=True)

# Encode categorical variables
categorical_columns = df.select_dtypes(include=['object']).columns
# Convert categorical columns to category type
df[categorical_columns] = df[categorical_columns].astype('category')
# Encode categorical variables as integers
df[categorical_columns] = df[categorical_columns].apply(lambda x:
x.cat.codes)
# Verify that there are no missing values left
missing_values_after = df.isnull().sum()
print(missing_values_after[missing_values_after > 0])
```

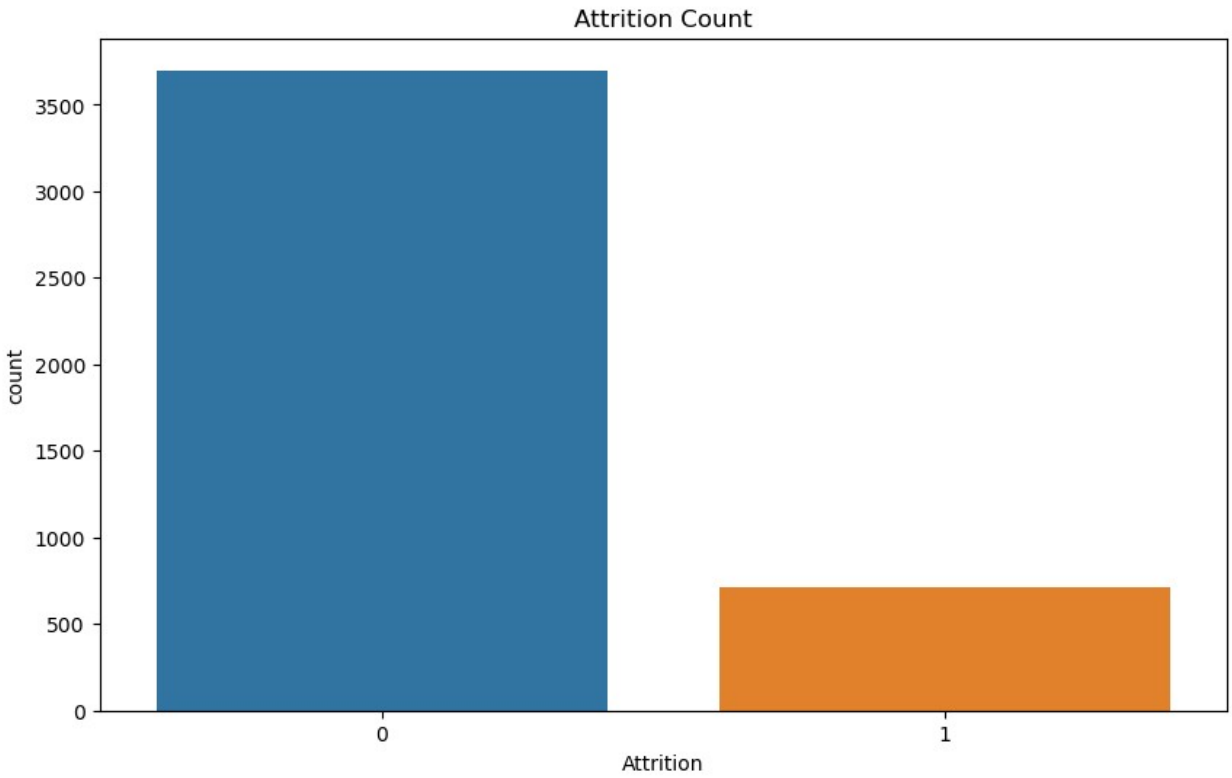
```
Series([], dtype: int64)
```

```
# Plot the distribution of age
plt.figure(figsize=(10, 6))
sns.histplot(df['Age'], kde=True)
plt.title('Age Distribution')
plt.show()
```

```
C:\Users\Admin\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

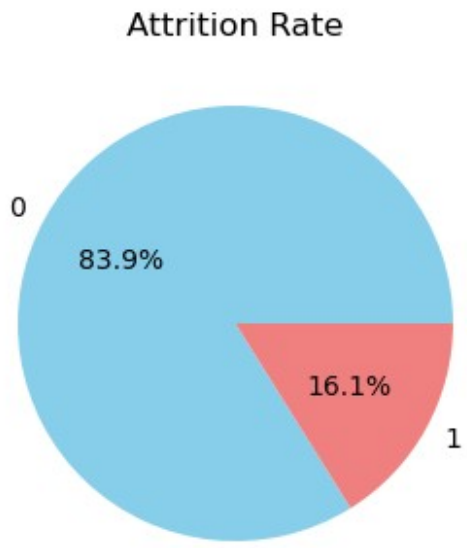



```
# Plot the count of attrition
plt.figure(figsize=(10, 6))
sns.countplot(x='Attrition', data=df)
plt.title('Attrition Count')
plt.show()
```



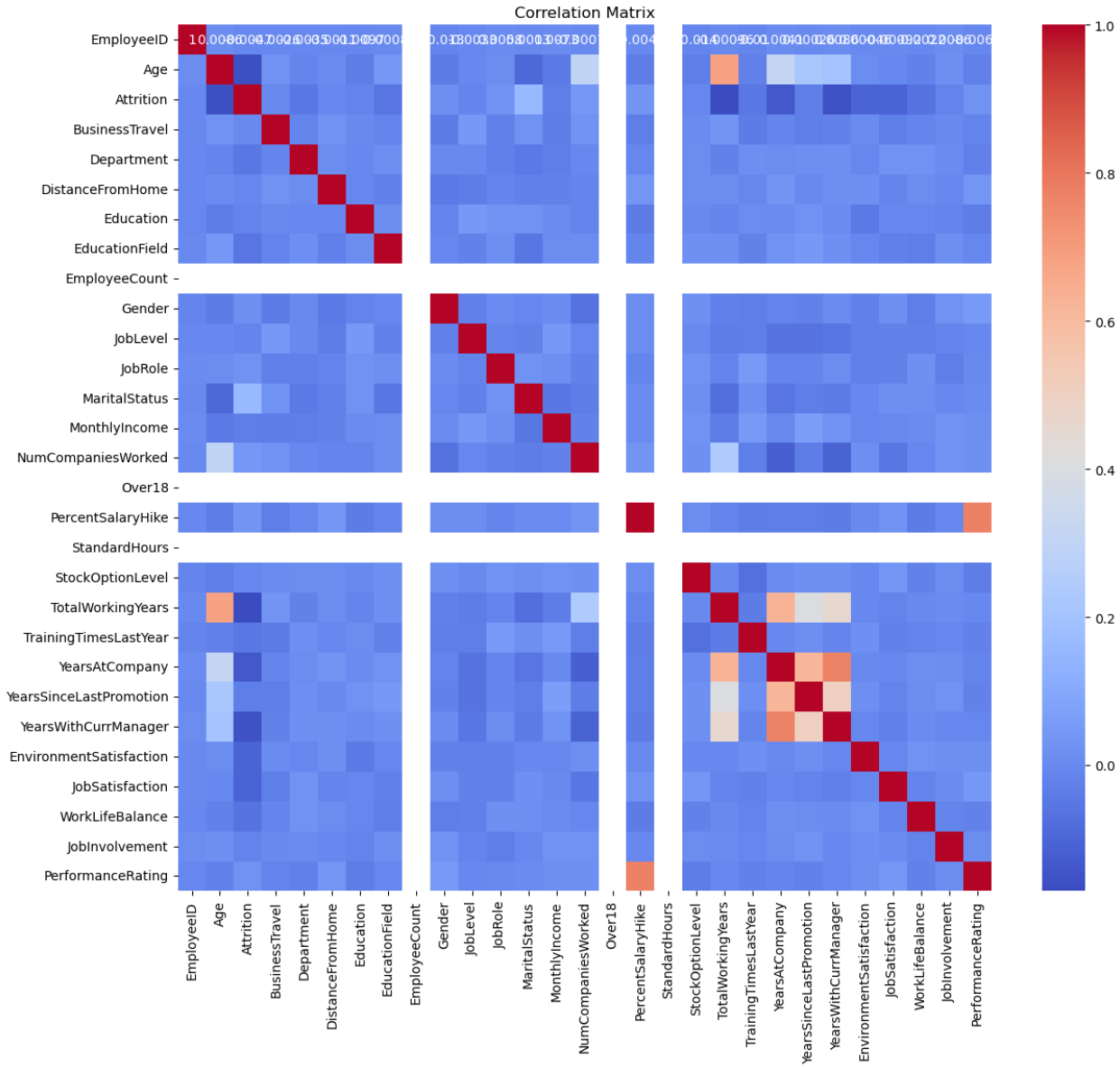

```
# Set up the matplotlib figure
plt.figure(figsize=(14, 8))
# Attrition rate pie chart
attrition_counts = df['Attrition'].value_counts()
plt.subplot(2, 2, 1)
plt.pie(attrition_counts, labels=attrition_counts.index,
autopct='%1.1f%%', colors=['skyblue', 'lightcoral'])
plt.title('Attrition Rate')
```

```
Text(0.5, 1.0, 'Attrition Rate')
```

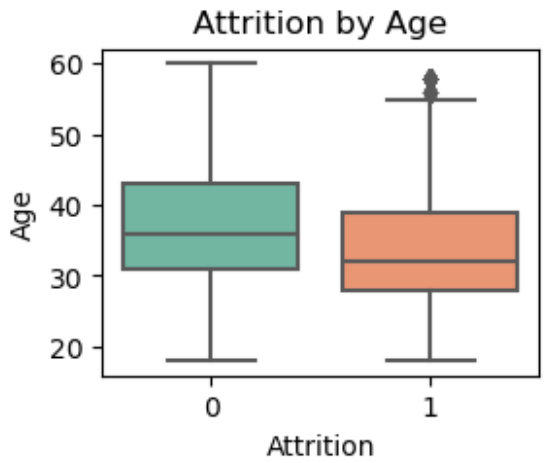



```
# Correlation matrix
plt.figure(figsize=(14, 12))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

```
C:\Users\Admin\anaconda3\Lib\site-packages\seaborn\matrix.py:260:
FutureWarning: Format strings passed to MaskedConstant are ignored,
but in future may error or produce different behavior
  annotation = ("{:" + self.fmt + "}").format(val)
```




```
# Box plot for Age
plt.subplot(2, 2, 4)
sns.boxplot(x='Attrition', y='Age', data=df, palette='Set2')
plt.title('Attrition by Age')

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




```
# Identify Key Factors Contributing to Attrition
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix

# Define features and target
X = df.drop(columns=['Attrition'])
y = df['Attrition']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Train a Random Forest Classifier
model = RandomForestClassifier(n_estimators=100, random_state=40)
model.fit(X_train, y_train)
```

```
RandomForestClassifier(random_state=40)
```

```
# Predict on the test set
y_pred = model.predict(X_test)
```

```
y_pred
```

```
array([0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0,
0,
```

```
        0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0,
        1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0,
        1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1,
        0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
        0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0,
        0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
        1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
0,
        0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,
1,
        0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
        1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
        0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0,
        0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0,
        0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0,
0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0,
        1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
0,
        0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
        0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
        0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1,
0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1,
        0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1,
```

```
0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1,
0,
       0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
1,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0,
       0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
1,
       0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1,
1,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1,
       0, 0], dtype=int8)
```

```python
# Display classification report and confusion matrix
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

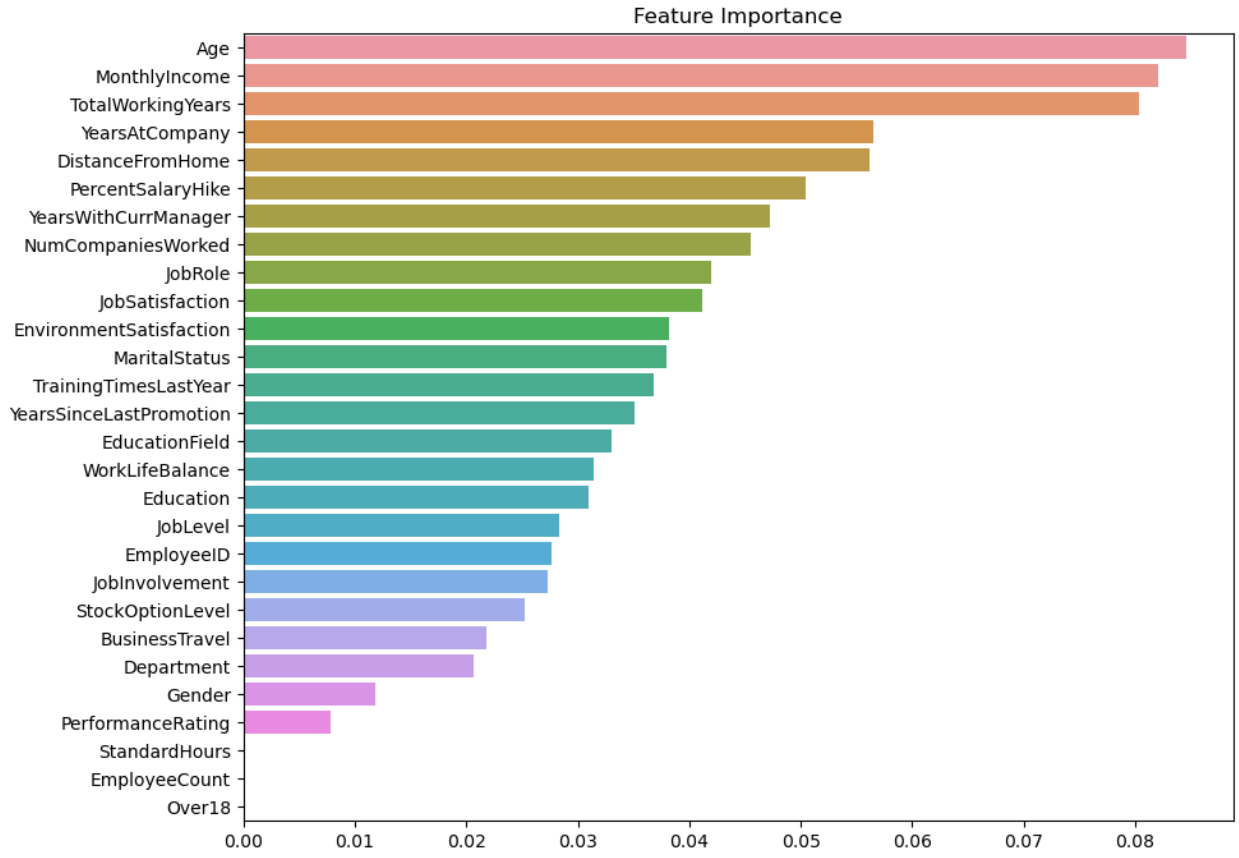
```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       741
           1       1.00      0.91      0.95       141

    accuracy                           0.99       882
   macro avg       0.99      0.95      0.97       882
weighted avg       0.99      0.99      0.98       882

[[741   0]
 [ 13 128]]
```

```python
# Feature importance
feature_importances = pd.Series(model.feature_importances_,
index=X.columns).sort_values(ascending=False)
print(feature_importances)
```

```
Age                         0.084539
MonthlyIncome               0.082058
TotalWorkingYears           0.080378
YearsAtCompany              0.056553
DistanceFromHome            0.056119
PercentSalaryHike           0.050465
YearsWithCurrManager        0.047292
NumCompaniesWorked          0.045470
```

```
JobRole                     0.041968
JobSatisfaction             0.041207
EnvironmentSatisfaction     0.038152
MaritalStatus               0.038006
TrainingTimesLastYear       0.036764
YearsSinceLastPromotion     0.035050
EducationField              0.033013
WorkLifeBalance             0.031374
Education                   0.030920
JobLevel                    0.028382
EmployeeID                  0.027636
JobInvolvement              0.027352
StockOptionLevel            0.025212
BusinessTravel              0.021759
Department                  0.020687
Gender                      0.011824
PerformanceRating           0.007821
StandardHours               0.000000
EmployeeCount               0.000000
Over18                      0.000000
dtype: float64
```

```python
# Plot feature importance
plt.figure(figsize=(10, 8))
sns.barplot(x=feature_importances, y=feature_importances.index)
plt.title('Feature Importance')
plt.show()
```


Feature Importance
Age
MonthlyIncome
TotalWorkingYears
YearsAtCompany
DistanceFromHome
PercentSalaryHike
YearsWithCurrManager
NumCompaniesWorked
JobRole
JobSatisfaction
EnvironmentSatisfaction
MaritalStatus
TrainingTimesLastYear
YearsSinceLastPromotion
EducationField
WorkLifeBalance
Education
JobLevel
EmployeeID
JobInvolvement
StockOptionLevel
BusinessTravel
Department
Gender
PerformanceRating
StandardHours
EmployeeCount
Over18
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08