IBM EMPLOYEE ATTRITION PREDICTION

The given data related to employee attrition at IBM company, and the goal is likely to predict which employees might leave (attrition = "Yes") or stay (attrition = "No"). The dataset contains various features about employees, their work environment, and personal attributes.

Importing all Librararies

In [51]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler,LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import warnings
warnings.filterwarnings("ignore")
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,precision_score,recall_score,f1_score,confusion_matrix,classification_
```

loading the dataset

In [9]:
```
df=pd.read_csv("C:/Users/user/Downloads/WA_Fn-UseC_-HR-Employee-Attrition.csv")
df
```

Out[9]:

yeeNumber	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsA
1	...	1	80	0	8	0	1	
2	...	4	80	1	10	3	3	
4	...	2	80	0	7	3	3	
5	...	3	80	0	8	3	3	
7	...	4	80	1	6	3	3	
...	...	...	...	...	...	...	...	
2061	...	3	80	1	17	3	3	
2062	...	1	80	1	9	5	3	
2064	...	2	80	1	6	0	3	
2065	...	4	80	0	17	3	2	
2068	...	1	80	0	6	3	4	

Column Explanation

1.Age: The age of the employee. This is a continuous numerical feature. Age can be a factor in attrition as it could correlate with job satisfaction and career stage.
2.Attrition: The target variable, indicating whether the employee left the company (Yes) or stayed (No). This is a categorical feature, where 'Yes' represents an employee who left (attrition) and 'No' represents an employee who stayed.
3.BusinessTravel: The frequency with which the employee travels for work. It can have three possible values:
1. Travel_Rarely
2.Trvael_Frequently
3.Non_Traveller
4.DailyRate: The daily rate of pay for the employee. This is a numerical feature and could be related to job satisfaction and retention.
5.Department: The department in which the employee works.certain departments can have higher or lower attrition rates.
6.DistanceFromHome: The distance (in miles) between the employee's home and the company. Employees with a long commute might experience higher job dissatisfaction, leading to higher attrition.

7.Education: The level of education the employee has completed.Education might correlate with career opportunities and job satisfaction.
8.EducationField: The field in which the employee obtained their education.
9.EmployeeCount: This is usually a constant value (e.g., 1) representing the count of employees. It's likely not a very useful feature for modeling.
10.EmployeeNumber: A unique identifier assigned to each employee. This is generally used for referencing the employee and doesn't provide meaningful information for prediction tasks.
11.EnvironmentSatisfaction: The employee's satisfaction with their work environment, on a scale from 1 to 4, where 1 is very dissatisfied and 4 is very satisfied. Higher satisfaction likely correlates with lower attrition.
12.Gender: The gender of the employee. It's a categorical feature with possible values such as Male and Female. Gender may have an indirect effect on attrition if there are any biases or systemic differences in how employees are treated.
13.HourlyRate: The hourly rate the employee earns. A higher hourly rate could be linked to job satisfaction and lower attrition.
14.JobInvolvement: The employee's level of involvement in their job, scored on a scale from 1 to 4, where 1 is low involvement and 4 is high involvement. More engaged employees may have lower attrition.
15.JobLevel: The job level the employee holds within the company, often ranging from 1 (entry-level) to 5 (senior-level). Job level can affect attrition, as employees in higher levels may be more likely to stay due to promotions and career growth opportunities.
16.JobRole: The role the employee holds within the company.Different roles might have varying attrition rates due to job demands and growth opportunities.
17.JobSatisfaction: The employee’s satisfaction with their job on a scale from 1 (very dissatisfied) to 4 (very satisfied). High job satisfaction is generally associated with lower attrition.
18.MaritalStatus: The marital status of the employee,might indirectly influence an employee’s work-life balance and, therefore, their likelihood of staying with the company.
19.MonthlyIncome: The monthly income of the employee. Higher income could be associated with lower attrition, as employees may be more likely to stay in their current job if the pay is competitive.
20.NumCompaniesWorked: The number of companies the employee has worked for in the past. A higher number of previous companies could indicate less loyalty or dissatisfaction with past employers, potentially leading to higher attrition.
21.OverTime: Whether the employee works overtime or not. It is a binary categorical feature (e.g., Yes or No). Working overtime might cause burnout or dissatisfaction, potentially leading to attrition.
22.PercentSalaryHike: The percentage increase in the employee's salary during the last year. Higher salary increases may correlate with higher employee satisfaction and lower attrition.
23.PerformanceRating: The employee’s performance rating, typically on a scale from 1 to 4. A higher performance rating could be associated with better career growth opportunities and lower attrition.
24.RelationshipSatisfaction: The employee's satisfaction with relationships with colleagues and supervisors, on a scale from 1 (very dissatisfied) to 4 (very satisfied). Strong relationships are often linked with lower attrition.
25.StandardHours: The standard number of hours an employee is expected to work. This is often constant and might not be a very useful feature.
26.StockOptionLevel: The level of stock options provided to the employee, typically 0, 1, or 2. Stock options could be a significant retention tool for employees.
27.TotalWorkingYears: The total number of years the employee has worked in their career. This might influence their likelihood of staying or leaving, as more experienced employees may have different retention patterns.
28.TrainingTimesLastYear: The number of training sessions the employee participated in during the past year. Higher training could indicate greater investment in the employee's growth, which may lead to lower attrition.
29.WorkLifeBalance: The employee's perception of their work-life balance, on a scale from 1 (bad) to 4 (good). A good work-life balance is often linked to higher job satisfaction and lower attrition.
30.YearsAtCompany: The number of years the employee has worked at the company. Long-tenured employees may be less likely to leave due to higher loyalty or seniority.
31.YearsInCurrentRole: The number of years the employee has spent in their current role. Employees in the same role for many years might experience boredom or stagnation, potentially increasing attrition.
32.YearsSinceLastPromotion: The number of years since the employee was last promoted. Employees who haven't been promoted in a long time may feel undervalued and are more likely to leave.
33.YearsWithCurrManager: The number of years the employee has worked with their current manager. Strong relationships with managers may lead to lower attrition.

Exploratoty Data Analysis

In [10]:
```
df.head() # find the first 5 rows
```

Out[10]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumbe
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7

5 rows × 35 columns

In [11]:
```
df.tail() # Last 5 rows
```

Out[11]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNun
1465	36	No	Travel_Frequently	884	Research & Development	23	2	Medical	1	2
1466	39	No	Travel_Rarely	613	Research & Development	6	1	Medical	1	2
1467	27	No	Travel_Rarely	155	Research & Development	4	3	Life Sciences	1	2
1468	49	No	Travel_Frequently	1023	Sales	2	3	Medical	1	2
1469	34	No	Travel_Rarely	628	Research & Development	8	3	Medical	1	2

5 rows × 35 columns

In [12]:
```
df.shape # number of rows and columns
```

Out[12]: (1470, 35)

In [13]:
```
df.info() # gets the total columns, their data types
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64 
 9   EmployeeNumber            1470 non-null   int64 
 10  EnvironmentSatisfaction   1470 non-null   int64 
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64 
 13  JobInvolvement            1470 non-null   int64 
 14  JobLevel                  1470 non-null   int64 
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64 
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64 
 19  MonthlyRate               1470 non-null   int64 
 20  NumCompaniesWorked        1470 non-null   int64 
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64 
 24  PerformanceRating         1470 non-null   int64 
 25  RelationshipSatisfaction  1470 non-null   int64 
 26  StandardHours             1470 non-null   int64 
 27  StockOptionLevel          1470 non-null   int64 
 28  TotalWorkingYears         1470 non-null   int64 
 29  TrainingTimesLastYear     1470 non-null   int64 
 30  WorkLifeBalance           1470 non-null   int64 
 31  YearsAtCompany            1470 non-null   int64 
 32  YearsInCurrentRole        1470 non-null   int64 
 33  YearsSinceLastPromotion   1470 non-null   int64 
 34  YearsWithCurrManager      1470 non-null   int64 
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
```

In [14]:
```
df.describe().T # statistical analysis of the columns
```

Out[14]:

	count	mean	std	min	25%	50%	75%	max
Age	1470.0	36.923810	9.135373	18.0	30.00	36.0	43.00	60.0
DailyRate	1470.0	802.485714	403.509100	102.0	465.00	802.0	1157.00	1499.0
DistanceFromHome	1470.0	9.192517	8.106864	1.0	2.00	7.0	14.00	29.0
Education	1470.0	2.912925	1.024165	1.0	2.00	3.0	4.00	5.0
EmployeeCount	1470.0	1.000000	0.000000	1.0	1.00	1.0	1.00	1.0
EmployeeNumber	1470.0	1024.865306	602.024335	1.0	491.25	1020.5	1555.75	2068.0
EnvironmentSatisfaction	1470.0	2.721769	1.093082	1.0	2.00	3.0	4.00	4.0
HourlyRate	1470.0	65.891156	20.329428	30.0	48.00	66.0	83.75	100.0
JobInvolvement	1470.0	2.729932	0.711561	1.0	2.00	3.0	3.00	4.0
JobLevel	1470.0	2.063946	1.106940	1.0	1.00	2.0	3.00	5.0
JobSatisfaction	1470.0	2.728571	1.102846	1.0	2.00	3.0	4.00	4.0
MonthlyIncome	1470.0	6502.931293	4707.956783	1009.0	2911.00	4919.0	8379.00	19999.0
MonthlyRate	1470.0	14313.103401	7117.786044	2094.0	8047.00	14235.5	20461.50	26999.0
NumCompaniesWorked	1470.0	2.693197	2.498009	0.0	1.00	2.0	4.00	9.0
PercentSalaryHike	1470.0	15.209524	3.659938	11.0	12.00	14.0	18.00	25.0
PerformanceRating	1470.0	3.153741	0.360824	3.0	3.00	3.0	3.00	4.0
RelationshipSatisfaction	1470.0	2.712245	1.081209	1.0	2.00	3.0	4.00	4.0
StandardHours	1470.0	80.000000	0.000000	80.0	80.00	80.0	80.00	80.0
StockOptionLevel	1470.0	0.793878	0.852077	0.0	0.00	1.0	1.00	3.0
TotalWorkingYears	1470.0	11.279592	7.780782	0.0	6.00	10.0	15.00	40.0
TrainingTimesLastYear	1470.0	2.799320	1.289271	0.0	2.00	3.0	3.00	6.0
WorkLifeBalance	1470.0	2.761224	0.706476	1.0	2.00	3.0	3.00	4.0
YearsAtCompany	1470.0	7.008163	6.126525	0.0	3.00	5.0	9.00	40.0
YearsInCurrentRole	1470.0	4.229252	3.623137	0.0	2.00	3.0	7.00	18.0
YearsSinceLastPromotion	1470.0	2.187755	3.222430	0.0	0.00	1.0	3.00	15.0
YearsWithCurrManager	1470.0	4.123129	3.568136	0.0	2.00	3.0	7.00	17.0

In [15]:
```
num_col=df.select_dtypes(include="number")
num_col
```

Out[15]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolveme
0	41	1102	1	2	1	1	2	94	
1	49	279	8	1	1	2	3	61	
2	37	1373	2	2	1	4	4	92	
3	33	1392	3	4	1	5	4	56	
4	27	591	2	1	1	7	1	40	
...	...	...	...	...	...	...	...	...	
1465	36	884	23	2	1	2061	3	41	
1466	39	613	6	1	1	2062	4	42	
1467	27	155	4	3	1	2064	2	87	
1468	49	1023	2	3	1	2065	4	63	
1469	34	628	8	3	1	2068	2	82	

1470 rows × 26 columns

In [16]:
```
df.select_dtypes(include="object")
```

Out[16]:

	Attrition	BusinessTravel	Department	EducationField	Gender	JobRole	MaritalStatus	Over18	OverTime
0	Yes	Travel_Rarely	Sales	Life Sciences	Female	Sales Executive	Single	Y	Yes
1	No	Travel_Frequently	Research & Development	Life Sciences	Male	Research Scientist	Married	Y	No
2	Yes	Travel_Rarely	Research & Development	Other	Male	Laboratory Technician	Single	Y	Yes
3	No	Travel_Frequently	Research & Development	Life Sciences	Female	Research Scientist	Married	Y	Yes
4	No	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Married	Y	No
...	...	...	...	...	...	...	...	...	...
1465	No	Travel_Frequently	Research & Development	Medical	Male	Laboratory Technician	Married	Y	No
1466	No	Travel_Rarely	Research & Development	Medical	Male	Healthcare Representative	Married	Y	No
1467	No	Travel_Rarely	Research & Development	Life Sciences	Male	Manufacturing Director	Married	Y	Yes
1468	No	Travel_Frequently	Sales	Medical	Male	Sales Executive	Married	Y	No
1469	No	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Married	Y	No

1470 rows × 9 columns

In [17]:
```
df.isnull().sum() # finding null values
```

Out[17]:
```
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

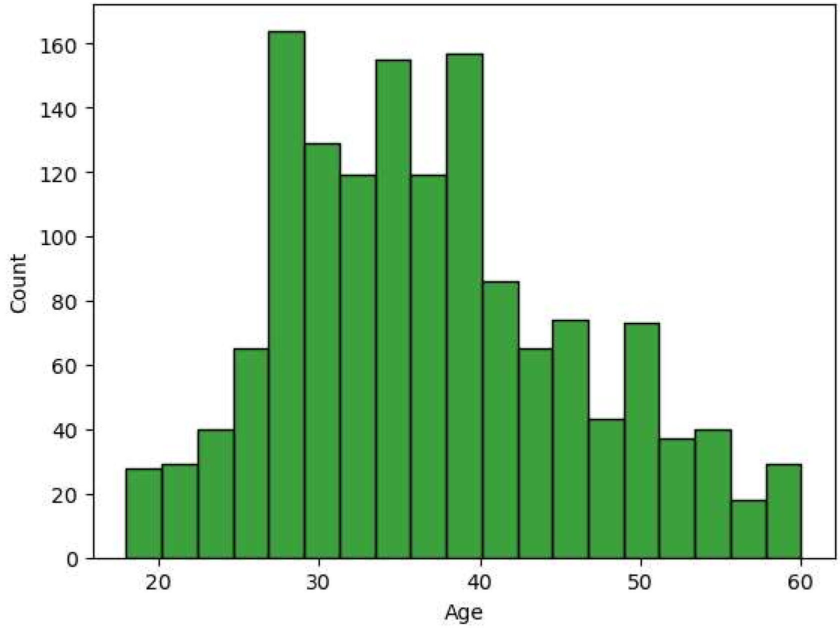
In [78]:
```
df.duplicated().sum() # finding duplicates
```

Out[78]:
```
0
```

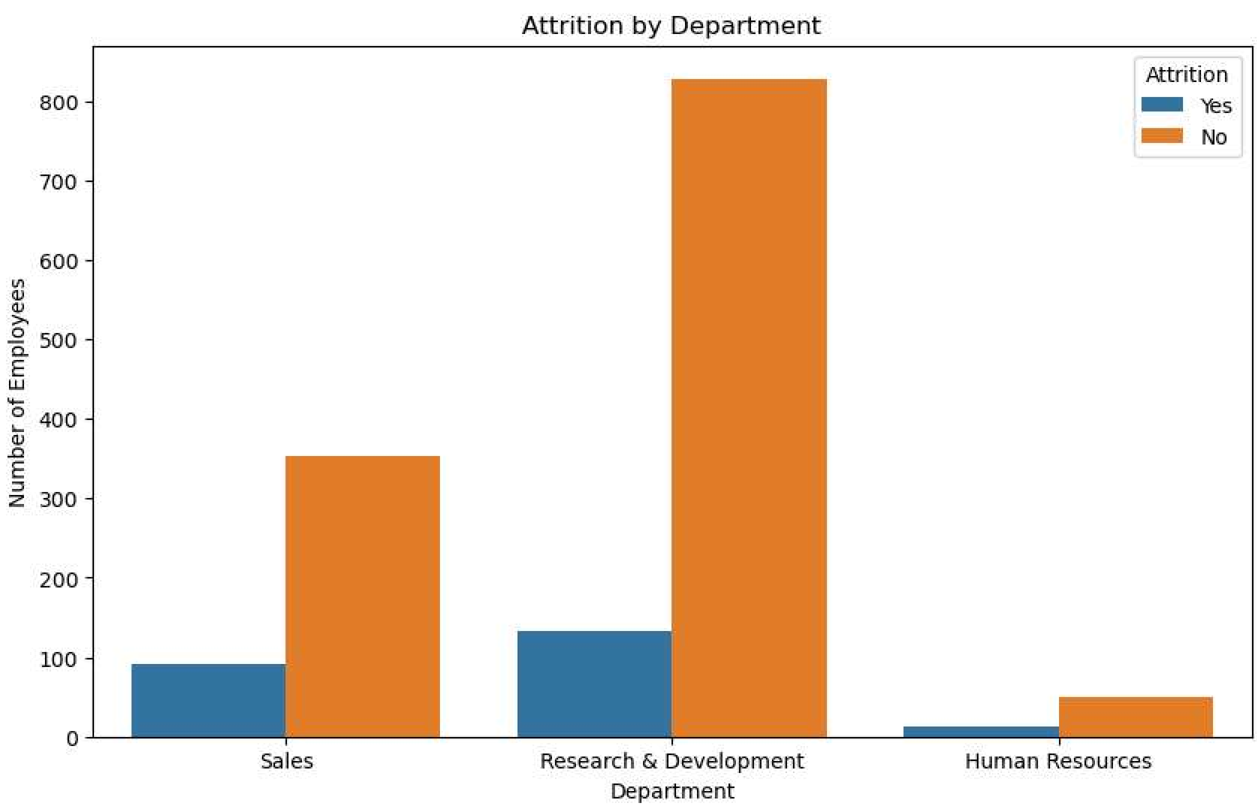
Data Visualizations

In [17]:
```
sns.histplot(x="Age",data=df,color="Green")
```

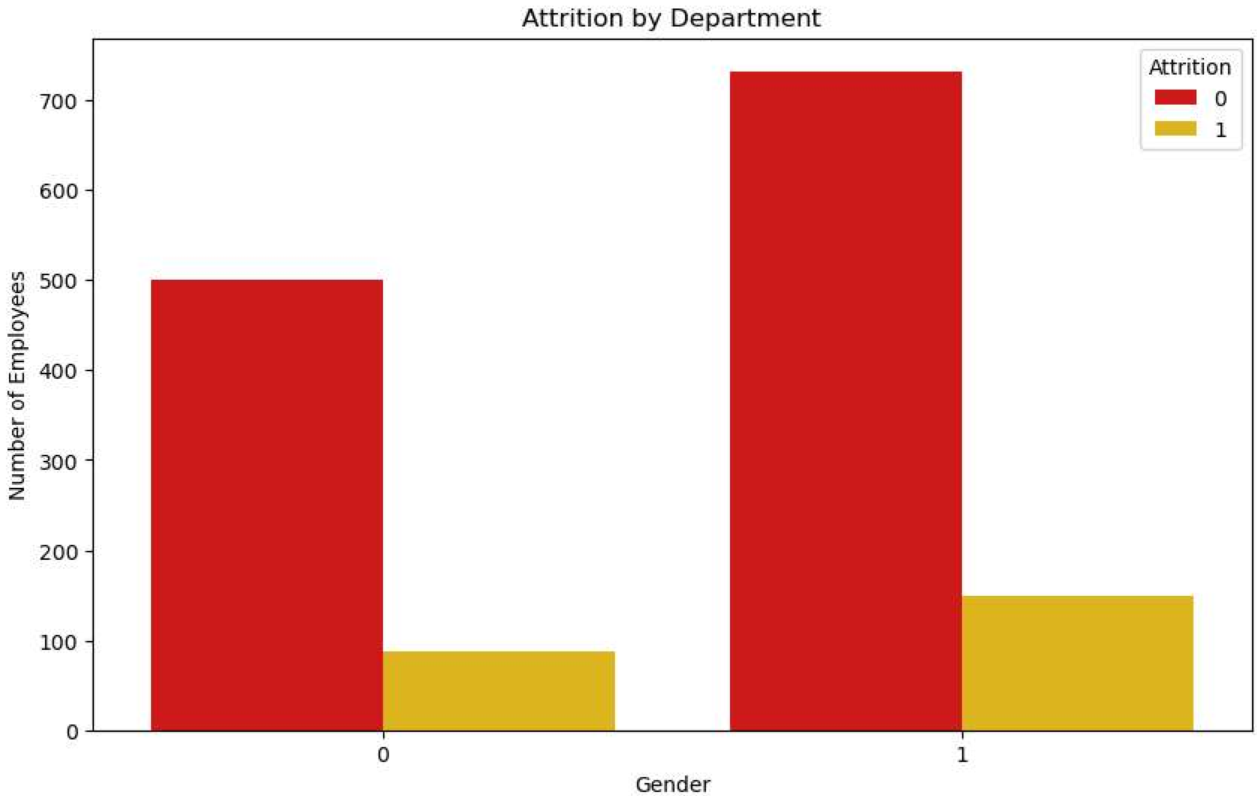
Out[17]: <Axes: xlabel='Age', ylabel='Count'>



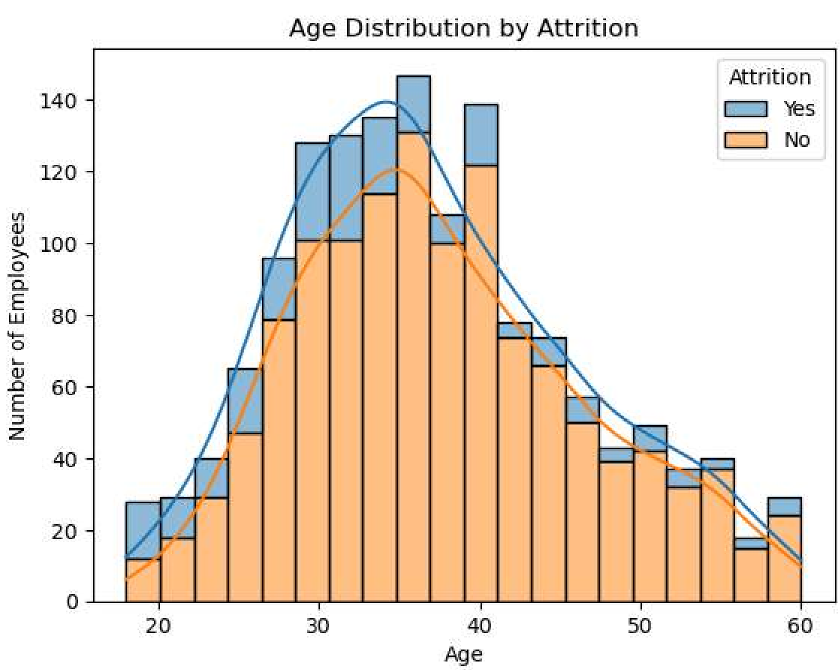
In [27]:
```
plt.figure(figsize=(10, 6))
sns.countplot(x='Department', hue='Attrition', data=df)
plt.title('Attrition by Department')
plt.xlabel('Department')
plt.ylabel('Number of Employees')
plt.show()
```



In [69]:
```
plt.figure(figsize=(10, 6))
sns.countplot(x='Gender', hue='Attrition', data=df,palette="hot")
plt.title('Attrition by Department')
plt.xlabel('Gender')
plt.ylabel('Number of Employees')
plt.show()
```

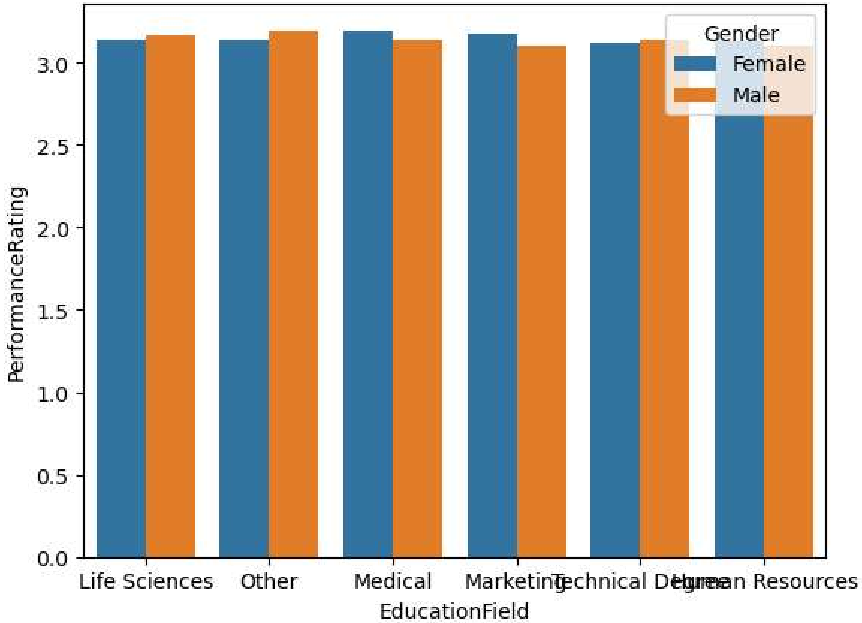


In [29]:
```
sns.histplot(data=df, x='Age', hue='Attrition', multiple='stack', bins=20, kde=True)
plt.title('Age Distribution by Attrition')
plt.xlabel('Age')
plt.ylabel('Number of Employees')
plt.show()
```



In [9]:
```
sns.barplot(x="EducationField",y="PerformanceRating",hue="Gender",data=df,ci=None)
plt.figure(figsize=(15,5))
```

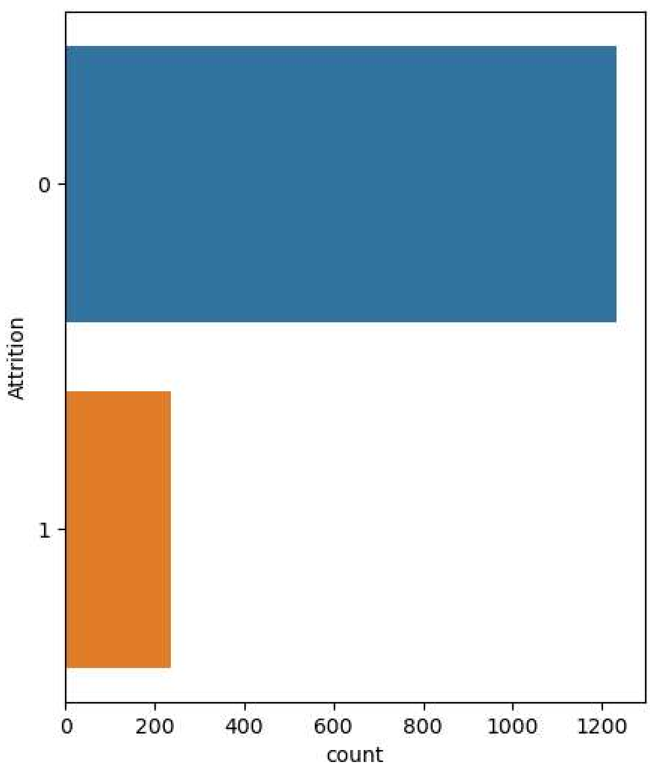
Out[9]: <Figure size 1500x500 with 0 Axes>



<Figure size 1500x500 with 0 Axes>

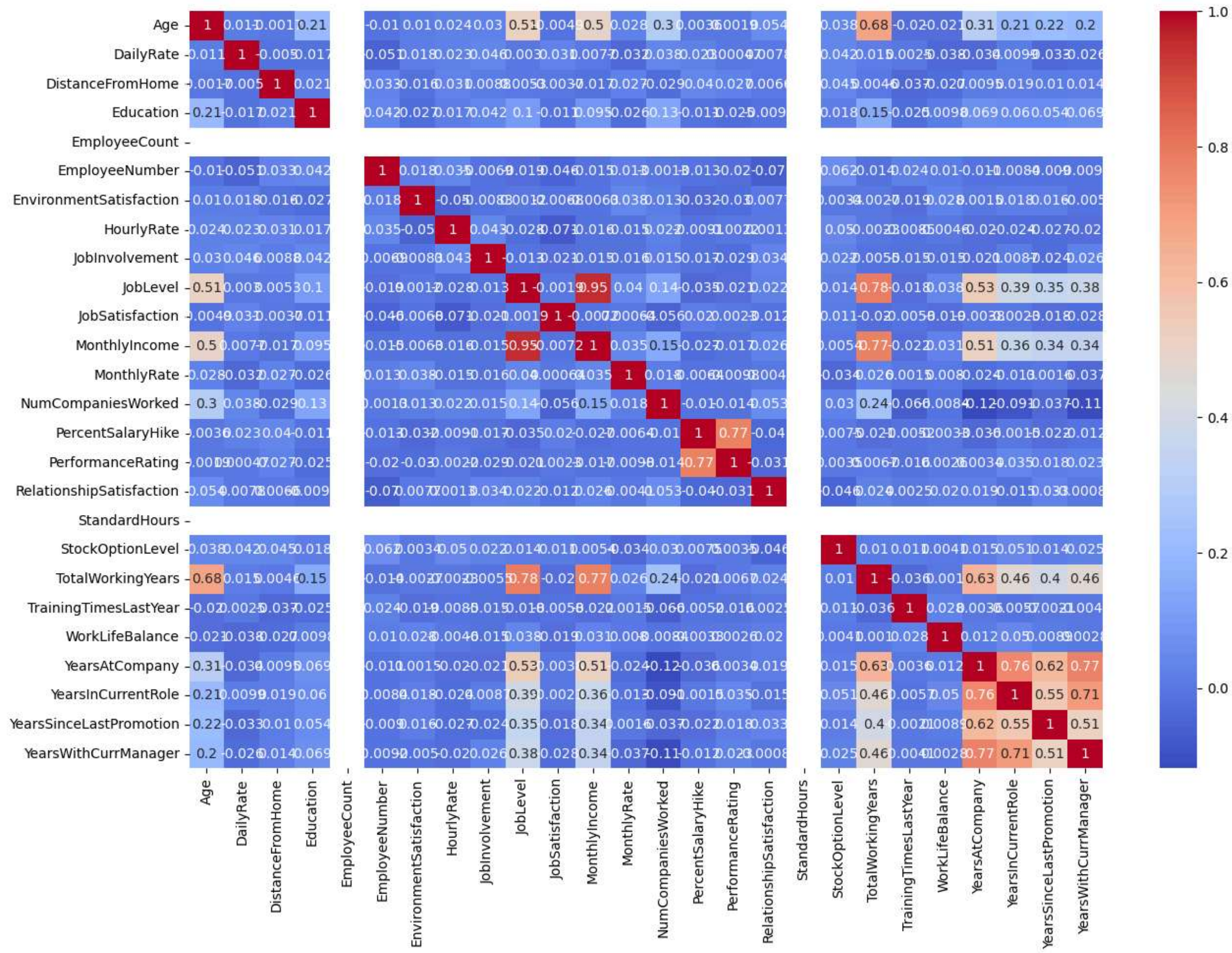
In [65]:
```
plt.figure(figsize=(5,6))
sns.countplot(y="Attrition",data=df)
```

Out[65]: <Axes: xlabel='count', ylabel='Attrition'>



In [103]:
```
plt.figure(figsize=(15,10))
corr=num_col.corr()
sns.heatmap(corr,annot=True,cmap="coolwarm")
```

Out[103]: <Axes: >



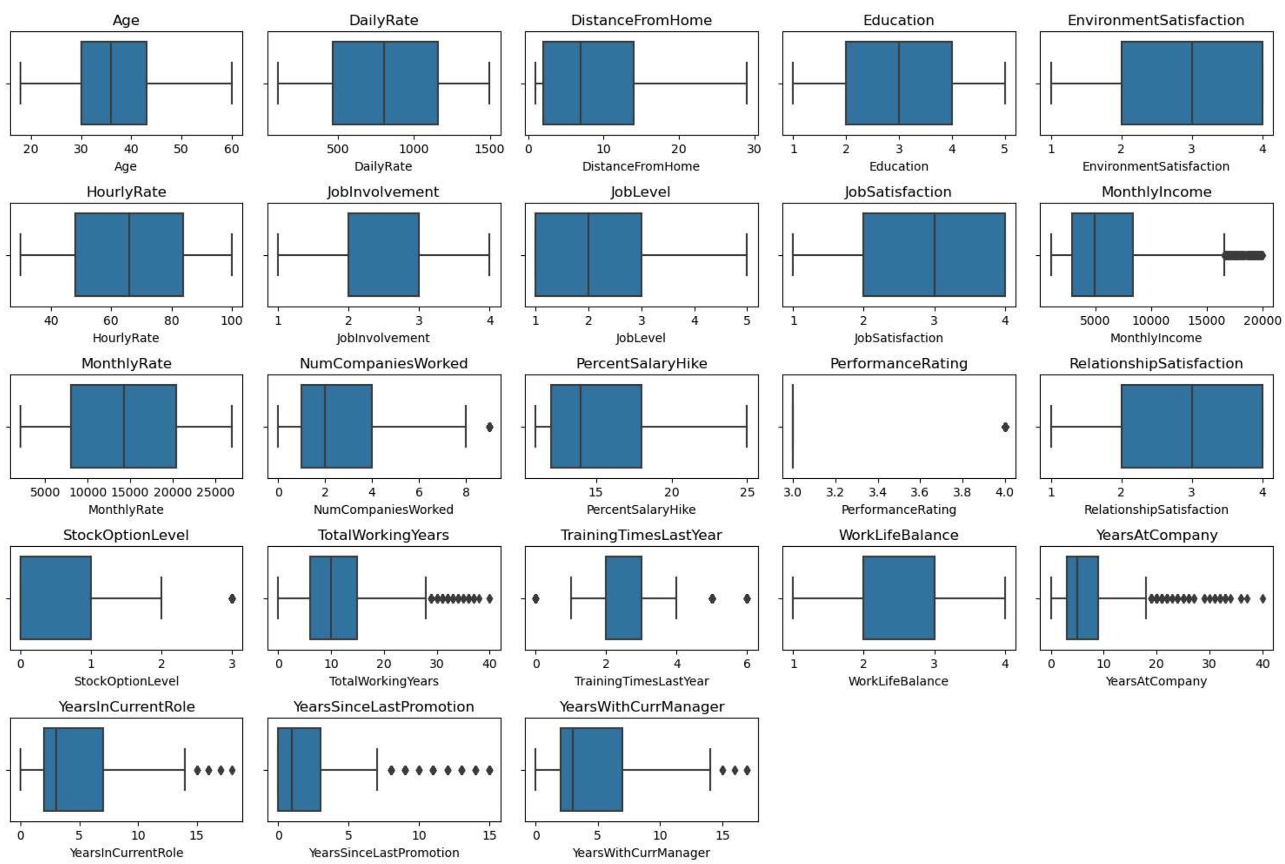
In [18]:
```
df.drop(columns=["EmployeeNumber","EmployeeCount","StandardHours","Over18"],axis=1,inplace=True)
df
```

Out[18]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Genc
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	2	Fema
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	3	Ma
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	4	Ma
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	4	Fema
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	Ma
...	...	...	...	...	...	...	...	...	...	
1465	36	No	Travel_Frequently	884	Research & Development	23	2	Medical	3	Ma
1466	39	No	Travel_Rarely	613	Research & Development	6	1	Medical	4	Ma
1467	27	No	Travel_Rarely	155	Research & Development	4	3	Life Sciences	2	Ma
1468	49	No	Travel_Frequently	1023	Sales	2	3	Medical	4	Ma
1469	34	No	Travel_Rarely	628	Research & Development	8	3	Medical	2	Ma

1470 rows × 31 columns

In [59]:
```
plt.figure(figsize=(15, 10))
for i, col in enumerate(num_col):
    plt.subplot(5, 5, i + 1)
    sns.boxplot(x=df[col], data=df)
    plt.title(col)
plt.tight_layout()
plt.show()
```



In [21]:
```
df["Attrition"].value_counts()
```

Out[21]:
```
Attrition
No     1233
Yes     237
Name: count, dtype: int64
```

In [22]:
```
df["Attrition"]=df["Attrition"].replace({"No":0,"Yes":1})
```

Encoder

An encoder is a tool or method used to convert categorical data into a numerical format so that it can be used in machine learning models, which typically require numerical input.
Label Encoding:
Converts each category in a feature into a unique integer.
Useful for ordinal data (where the categories have an inherent order).
One-Hot Encoding:
Creates binary columns for each category in a categorical feature.
For each category, the column will have a 1 for rows belonging to that category, and 0 otherwise.
This is ideal for nominal (non-ordinal) data, where no inherent order exists.

In [24]:
```
encoder=LabelEncoder()
```

In [25]:
```
df['Gender']=encoder.fit_transform(df["Gender"])
df['MaritalStatus']=encoder.fit_transform(df["MaritalStatus"])
```

In [26]:
```
data=pd.get_dummies(df,columns=['BusinessTravel', 'Department', 'EducationField', 'JobRole', 'OverTime'])
```

In [27]:
```
data
```

Out[27]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EnvironmentSatisfaction	Gender	HourlyRate	JobInvolvement	JobLevel	...
0	41	1	1102	1	2	2	0	94	3	2	...
1	49	0	279	8	1	3	1	61	2	2	...
2	37	1	1373	2	2	4	1	92	2	1	...
3	33	0	1392	3	4	4	0	56	3	1	...
4	27	0	591	2	1	1	1	40	3	1	...
...	...	...	...	...	...	...	...	...	...	...	...
1465	36	0	884	23	2	3	1	41	4	2	...
1466	39	0	613	6	1	4	1	42	2	3	...
1467	27	0	155	4	3	2	1	87	4	2	...
1468	49	0	1023	2	3	4	1	63	2	2	...
1469	34	0	628	8	3	2	1	82	4	2	...

1470 rows × 49 columns

In [28]:
```
data = data.astype(int)
```

In [30]:
```
x=data.drop("Attrition",axis=1)
y=data["Attrition"]
```

Standard Scaler

The StandardScaler is a data preprocessing technique used to normalize or standardize the features of your dataset so that they all have the same scale, which is important for many machine learning algorithms. It transforms the data by scaling the features to have zero mean and unit variance.

In [33]:
```
scaler=StandardScaler()
```

In [34]:
```
x_scaled=scaler.fit_transform(x)
```

RandomSampler
A Random Sampler refers to a technique used to select a random subset of data points from a larger dataset. This is often used for tasks like splitting data, bootstrapping, cross-validation, or to obtain representative subsets of data for model training, validation, or testing.
Random sampling helps to reduce bias and ensures that your model is not overfitting to a specific subset of the data.

In [71]:
```
from imblearn.over_sampling import RandomOverSampler
```

In [72]:
```
sampler=RandomOverSampler()
```

In [73]:
```
x_sampled,y_sampled=sampler.fit_resample(x_scaled,y)
```

Spliting Dataset

In [75]:
```
x_train,x_test,y_train,y_test=train_test_split(x_sampled,y_sampled,test_size=0.2,random_state=42)
```

Model Building

LOGISTIC REGRESSION
Logistic Regression is a statistical method used for binary classification problems (i.e., when the output variable has two possible outcomes).Despite its name, it is used for classification, not regression. It models the probability that a given input point belongs to a particular class.

In [76]:
```
model=LogisticRegression()
```

In [77]:
```
model.fit(x_train,y_train)
```

Out[77]:
```
▾ LogisticRegression
LogisticRegression()
```
(https://scikit-learn.org/1.6/modules/generated/sklearn.linear_model.LogisticRegression.html)

In [78]:
```
y_pred=model.predict(x_test)
```

In [79]:
```
accuracy_lr=accuracy_score(y_test,y_pred)
accuracy_lr
```

Out[79]:
```
0.7631578947368421
```

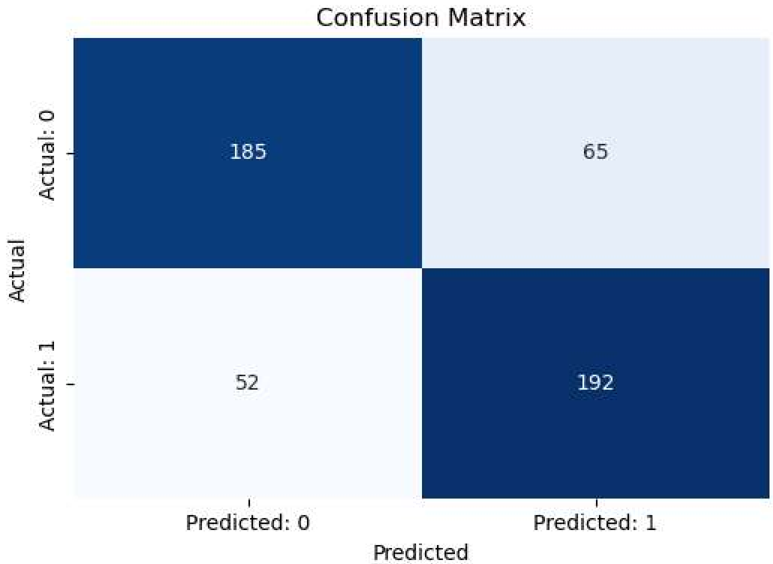
In [92]:
```
print("confusion metrix of Logistic Regression")
cm=confusion_matrix(y_test,y_pred)
cm
```

```
confusion metrix of Logistic Regression
```

Out[92]:
```
array([[185,  65],
       [ 52, 192]], dtype=int64)
```

In [94]:
```
plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", cbar=False,
            xticklabels=["Predicted: 0", "Predicted: 1"],
            yticklabels=["Actual: 0", "Actual: 1"])

plt.title("Confusion Matrix")
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```



In [81]:
```
print("The classification report og Logistic Regression is:")
print(classification_report(y_test,y_pred))
```

```
The classification report og Logistic Regression is:
              precision    recall  f1-score   support

           0       0.78      0.74      0.76       250
           1       0.75      0.79      0.77       244

    accuracy                           0.76       494
   macro avg       0.76      0.76      0.76       494
weighted avg       0.76      0.76      0.76       494
```

In [82]:
```
print("AUC:", roc_auc_score(y_test, y_pred))
```

```
AUC: 0.7634426229508197
```

RANDOMFOREST CLASSIFIER

The Random Forest Classifier is an ensemble learning method that combines multiple decision trees to create a more robust and accurate classification model. It is based on the concept of bagging (Bootstrap Aggregating), where multiple models are trained on different subsets of the data and their results are aggregated to improve the overall performance.

In [83]:
```
model_2=RandomForestClassifier()
```

In [84]:
```
model_2.fit(x_train,y_train)
```

Out[84]:
```
RandomForestClassifier
RandomForestClassifier()
```
(https://scikit-learn.org/1.6/modules/generated/sklearn.ensemble.RandomForestClassifier.html)

In [85]:
```
y_pred_2=model_2.predict(x_test)
```

In [86]:
```
accuracy_rf=accuracy_score(y_test,y_pred_2)
accuracy_rf
```

Out[86]:
```
0.9635627530364372
```

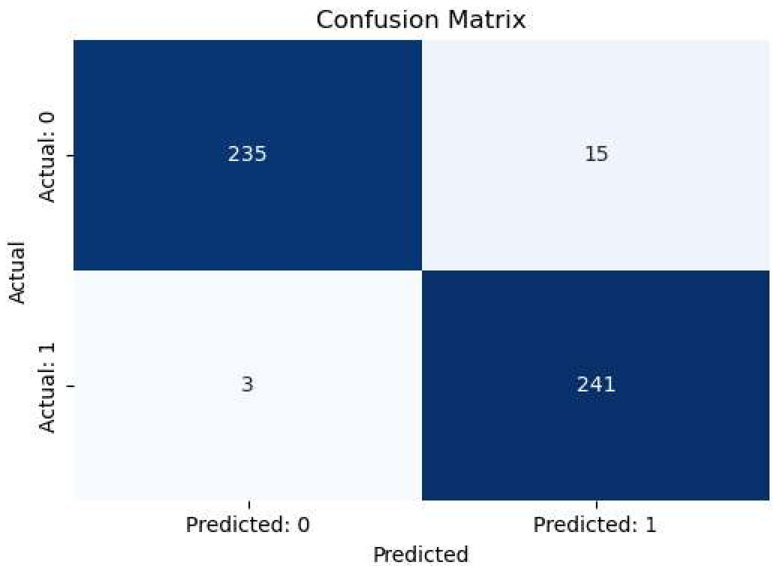
In [95]:
```
print("confusion metrix of Random Forest")
cm2=confusion_matrix(y_test,y_pred_2)
cm2
```

```
confusion metrix of Random Forest
```

Out[95]:
```
array([[235,  15],
       [  3, 241]], dtype=int64)
```

In [96]:
```
plt.figure(figsize=(6, 4))
sns.heatmap(cm2, annot=True, fmt="d", cmap="Blues", cbar=False,
            xticklabels=["Predicted: 0", "Predicted: 1"],
            yticklabels=["Actual: 0", "Actual: 1"])

plt.title("Confusion Matrix")
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```



In [88]:
```
print("The classification report og Logistic Regression is:")
print(classification_report(y_test,y_pred_2))
```

```
The classification report og Logistic Regression is:
              precision    recall  f1-score   support

           0       0.99      0.94      0.96       250
           1       0.94      0.99      0.96       244

    accuracy                           0.96       494
   macro avg       0.96      0.96      0.96       494
weighted avg       0.96      0.96      0.96       494
```

CONCLUSION

Random Forest Classifier performed much better than Logistic Regression in this case. This is typically due to the ensemble nature of Random Forest, where multiple decision trees are built, each learning from a different subset of the data. As a result, it is better equipped to handle complexity, non-linearity, and outliers, which may be present in the data.
Logistic Regression, while interpretable and effective for linearly separable data, might struggle to perform well when the data has complex relationships, as it assumes a linear relationship between the features and the target variable. In contrast, Random Forest doesn’t have this limitation, as it can model non-linear relationships between features.

In [ ]: