ASSIGNMENT-9: BINARY CLASSIFICATION

Objectives

1. Understand the concept and application of binary classification.
2. Implement different binary classification algorithms.
3. Evaluate the performance of the models using various metrics.

Dataset

Download the Employee dataset from Kaggle.

Link - https://www.kaggle.com/datasets/tawfikelmetwally/employee-dataset

Tasks

The task here is to find whether an employee is going to continue or leave the organization.

Task 1: Data Exploration and Preprocessing

1. Load the dataset and display the first few rows.
2. Perform basic statistical analysis to understand the distribution of the features.
3. Perform different visual exploratory data analysis such as
   i. Histograms
   ii. Correlations
   iii. Pair wise plots
   iv. Box plots
4. Check for missing values and handle them appropriately.
5. Check for outliers and handle them appropriately.
6. Check whether the dataset is balanced or not.
7. Split the data into training and testing sets.

```
import pandas as pd
df = pd.read_csv('Employee.csv')
df.head()
```

	Education	JoiningYear	City	PaymentTier	Age	Gender	EverBenched	Experienc
0	Bachelors	2017	Bangalore	3	34	Male	No	
1	Bachelors	2013	Pune	1	28	Female	No	
2	Bachelors	2014	New Delhi	3	38	Female	No	
3	Masters	2016	Bangalore	3	27	Male	No	
4	Masters	2017	Pune	3	24	Male	Yes	

```
# 2. Basic Statistical Analysis
# Statistical summary
print("\nBasic statistical analysis:")
print(df.describe())
```

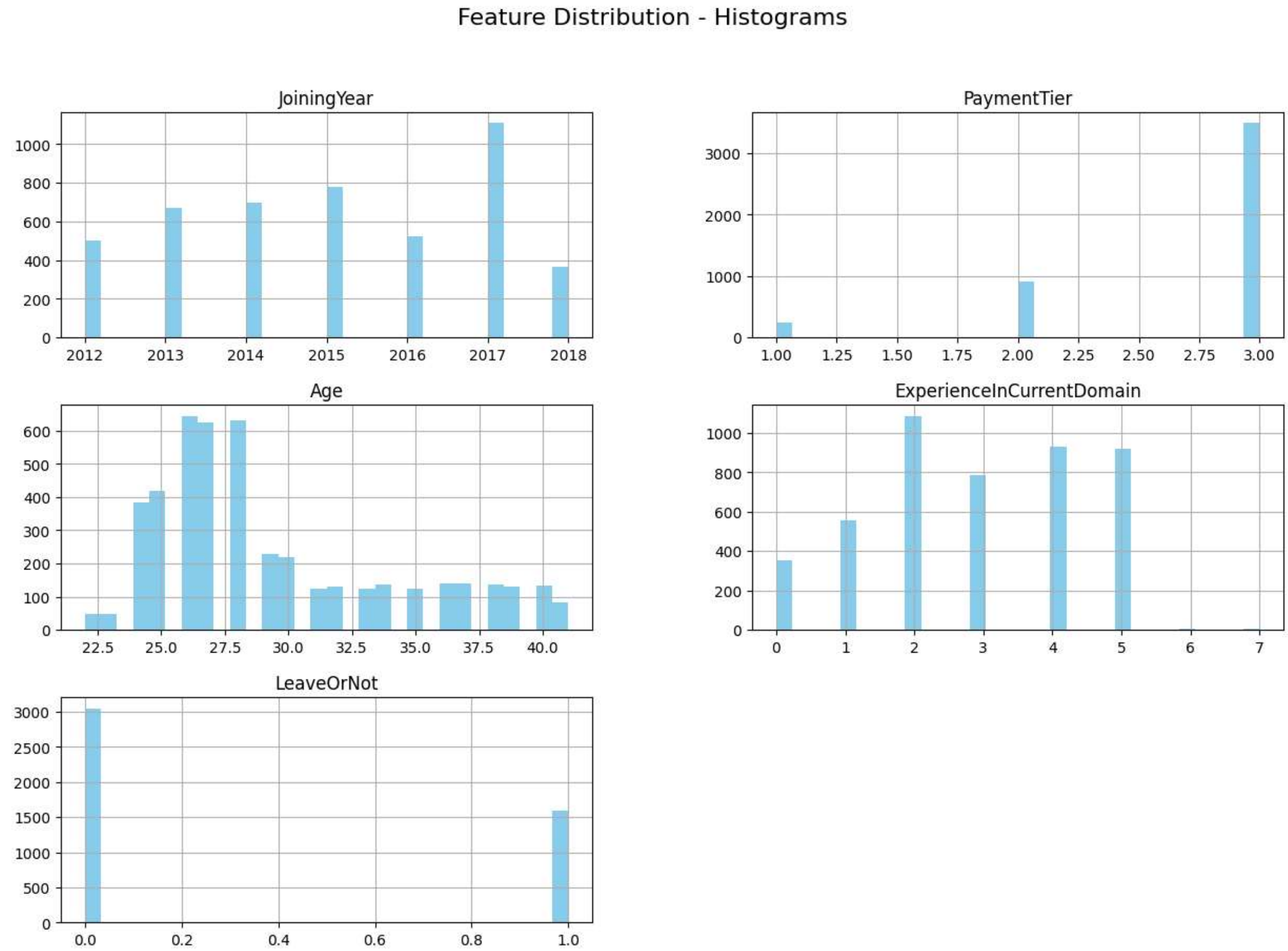
```
Basic statistical analysis:
       JoiningYear  PaymentTier          Age  ExperienceInCurrentDomain  \
count  4653.000000  4653.000000  4653.000000                4653.000000
mean   2015.062970     2.698259    29.393295                   2.905652
std       1.863377     0.561435     4.826087                   1.558240
min    2012.000000     1.000000    22.000000                   0.000000
25%    2013.000000     3.000000    26.000000                   2.000000
50%    2015.000000     3.000000    28.000000                   3.000000
75%    2017.000000     3.000000    32.000000                   4.000000
max    2018.000000     3.000000    41.000000                   7.000000

        LeaveOrNot
count  4653.000000
mean      0.343864
std       0.475047
min       0.000000
25%       0.000000
50%       0.000000
75%       1.000000
max       1.000000
```

```
# Check data types and non-null counts
print("\nData info:")
print(df.info())
```

```
Data info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4653 entries, 0 to 4652
Data columns (total 9 columns):
 #   Column                     Non-Null Count  Dtype 
---  ------                     --------------  ----- 
 0   Education                  4653 non-null   object
 1   JoiningYear                4653 non-null   int64 
 2   City                       4653 non-null   object
 3   PaymentTier                4653 non-null   int64 
 4   Age                        4653 non-null   int64 
 5   Gender                     4653 non-null   object
 6   EverBenched                4653 non-null   object
 7   ExperienceInCurrentDomain  4653 non-null   int64 
 8   LeaveOrNot                 4653 non-null   int64 
dtypes: int64(5), object(4)
memory usage: 327.3+ KB
None
```

```
# 3. Visual Exploratory Data Analysis
# i. Histograms
import seaborn as sns
import matplotlib.pyplot as plt
df.hist(bins=30, figsize=(15, 10), color='skyblue')
```

```python
plt.suptitle('Feature Distribution - Histograms', fontsize=16)
plt.show()
```




```python
####---- Label encoding for co relation plot ----####
from sklearn.preprocessing import LabelEncoder
```

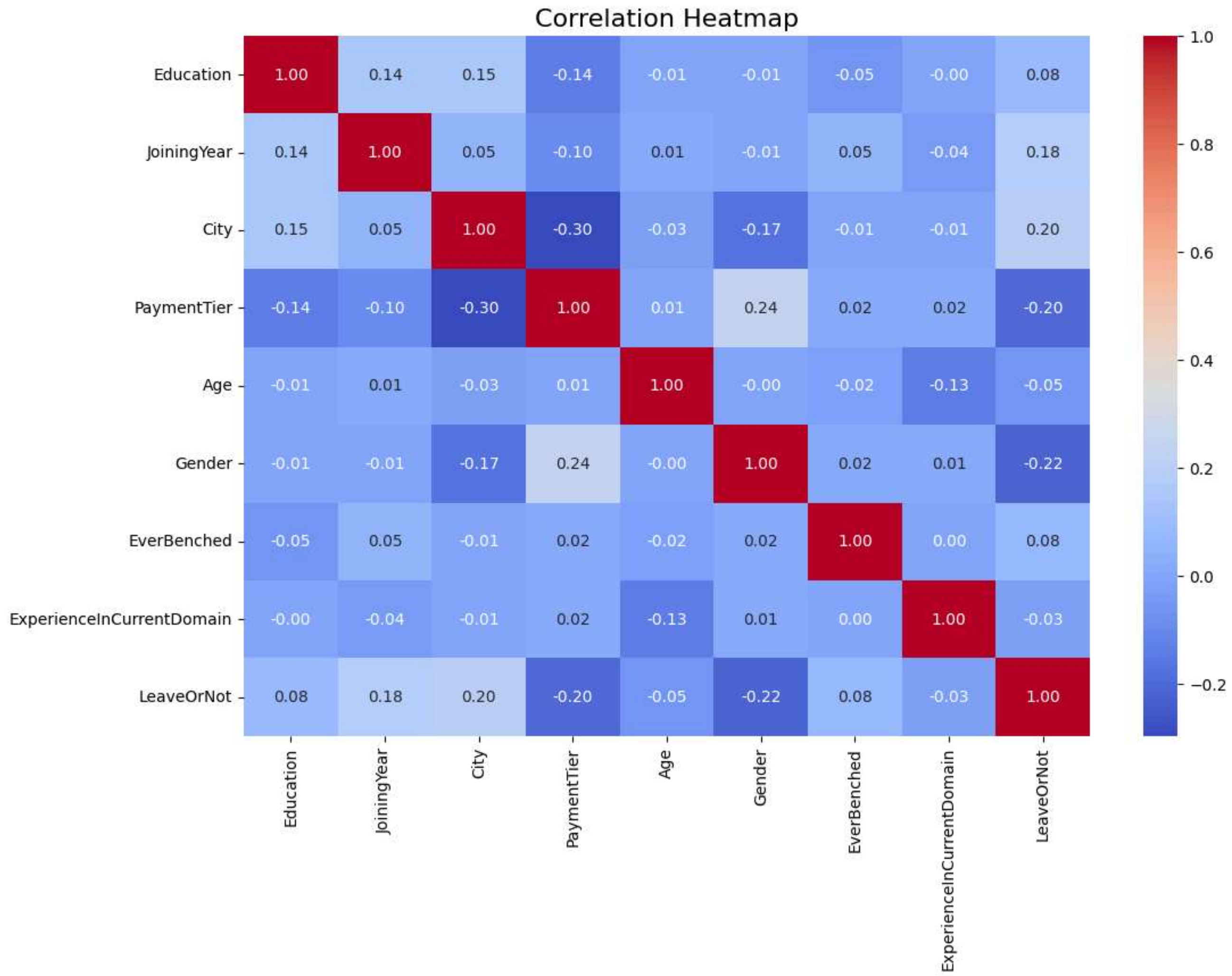
```
le = LabelEncoder()
df['Education'] = le.fit_transform(df['Education'])
#df.head()
df['City'] = le.fit_transform(df['City'])
#df.head()
df['Gender'] = le.fit_transform(df['Gender'])
df['EverBenched'] = le.fit_transform(df['EverBenched'])
df.head()
```

	Education	JoiningYear	City	PaymentTier	Age	Gender	EverBenched	ExperienceInC
0	0	2017	0	3	34	1	0	
1	0	2013	2	1	28	0	0	
2	0	2014	1	3	38	0	0	
3	1	2016	0	3	27	1	0	
4	1	2017	2	3	24	1	1	

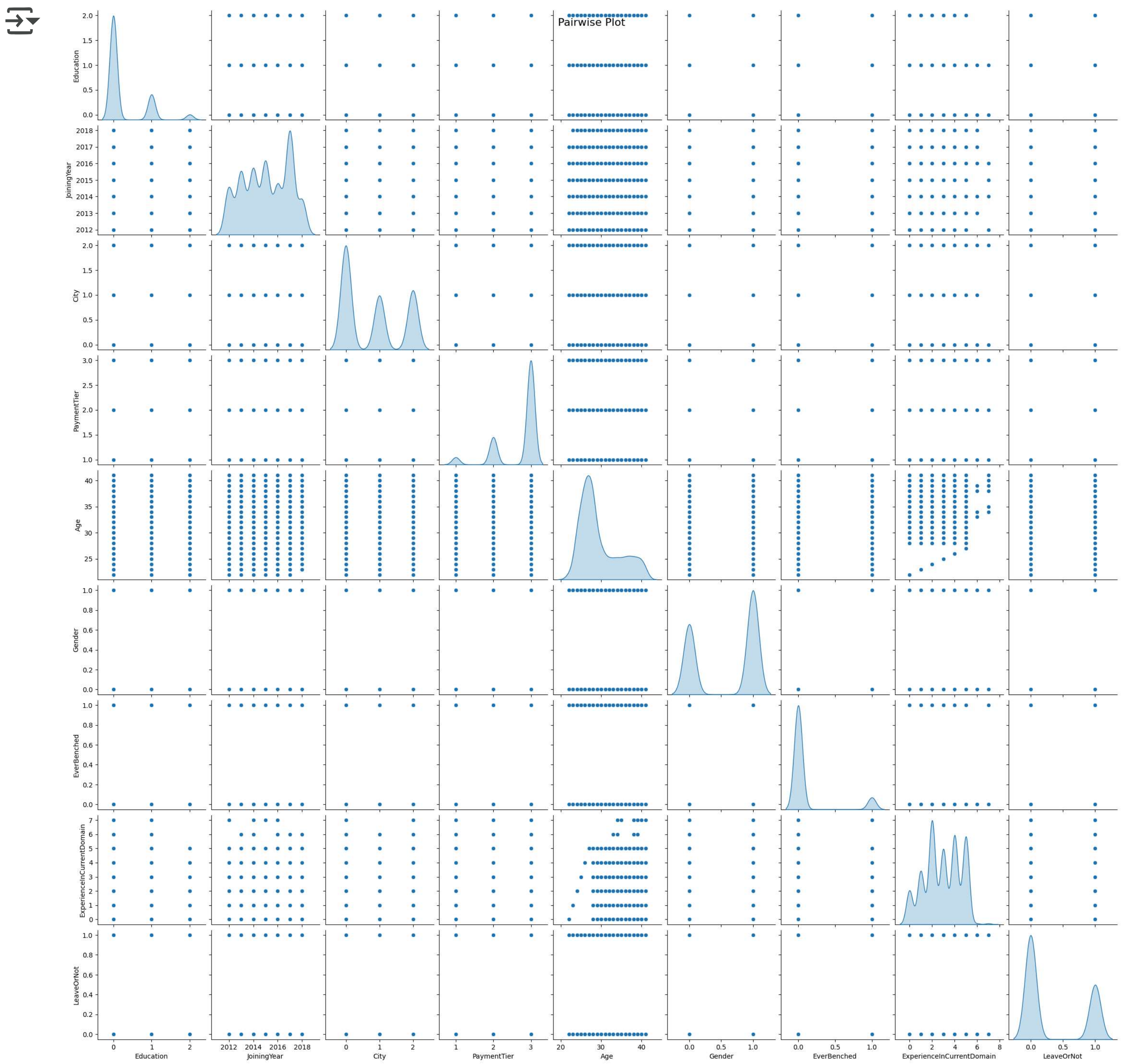
```
df.corr()
```

	Education	JoiningYear	City	PaymentTier	Age	(
Education	1.000000	0.142670	0.149903	-0.140741	-0.010611	-0.0
JoiningYear	0.142670	1.000000	0.051441	-0.096078	0.013165	-0.0
City	0.149903	0.051441	1.000000	-0.295884	-0.030706	-0.1
PaymentTier	-0.140741	-0.096078	-0.295884	1.000000	0.007631	0.2
Age	-0.010611	0.013165	-0.030706	0.007631	1.000000	-0.0
Gender	-0.010889	-0.012213	-0.168546	0.235119	-0.003866	1.0
EverBenched	-0.052249	0.049353	-0.007046	0.019207	-0.016135	0.0
ExperienceInCurrentDomain	-0.004463	-0.036525	-0.009925	0.018314	-0.134643	0.0
LeaveOrNot	0.080497	0.181705	0.201058	-0.197638	-0.051126	-0.2

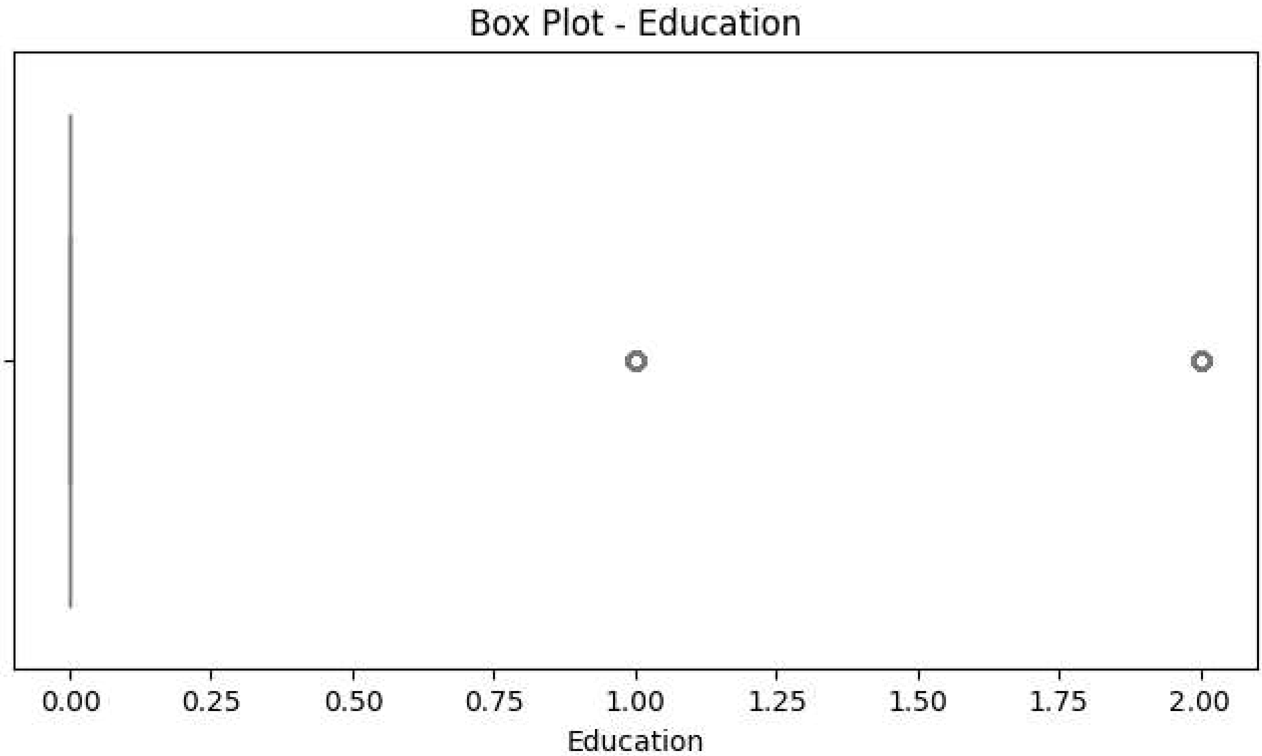
```
# ii. Correlation heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(), annot=True, fmt=".2f", cmap='coolwarm')
plt.title('Correlation Heatmap', fontsize=16)
plt.show()
```

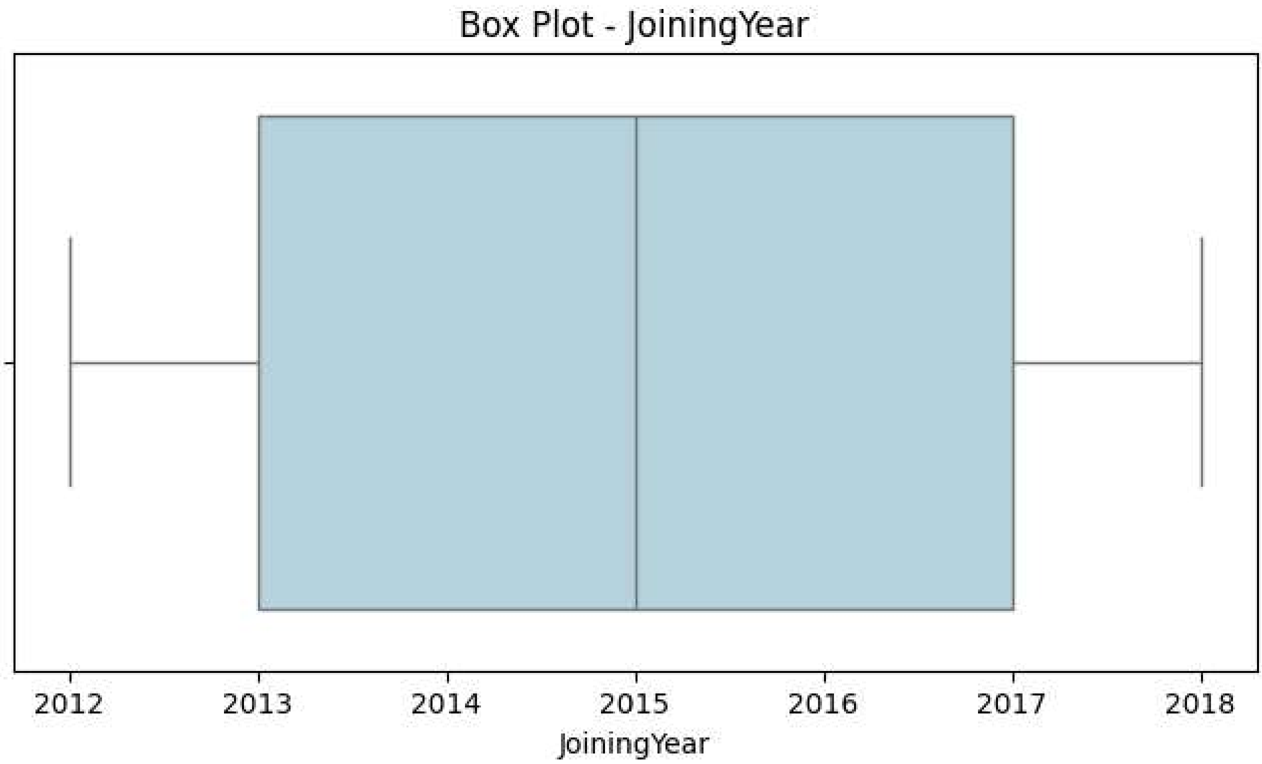


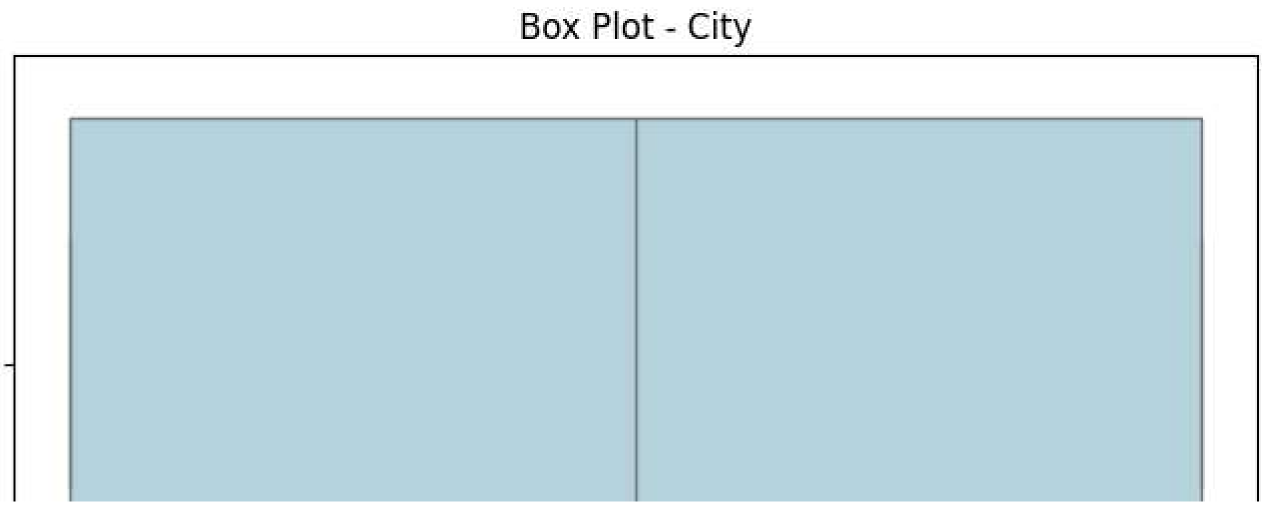

```
# iii. Pairwise plots
sns.pairplot(df, diag_kind='kde')
plt.suptitle('Pairwise Plot', fontsize=16)
plt.show()
```

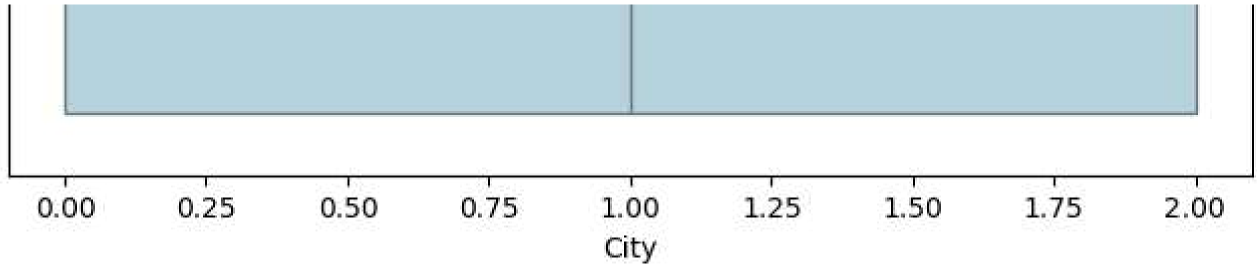

Pairwise Plot
Education
JoiningYear
City
PaymentTier
Age
Gender
EverBenched
ExperienceInCurrentDomain
LeaveOrNot

```
# iv. Box plots for numerical features
import numpy as np
numerical_columns = df.select_dtypes(include=np.number).columns
for col in numerical_columns:
    plt.figure(figsize=(8, 4))
    sns.boxplot(x=df[col], color='lightblue')
    plt.title(f'Box Plot - {col}')
    plt.show()
```

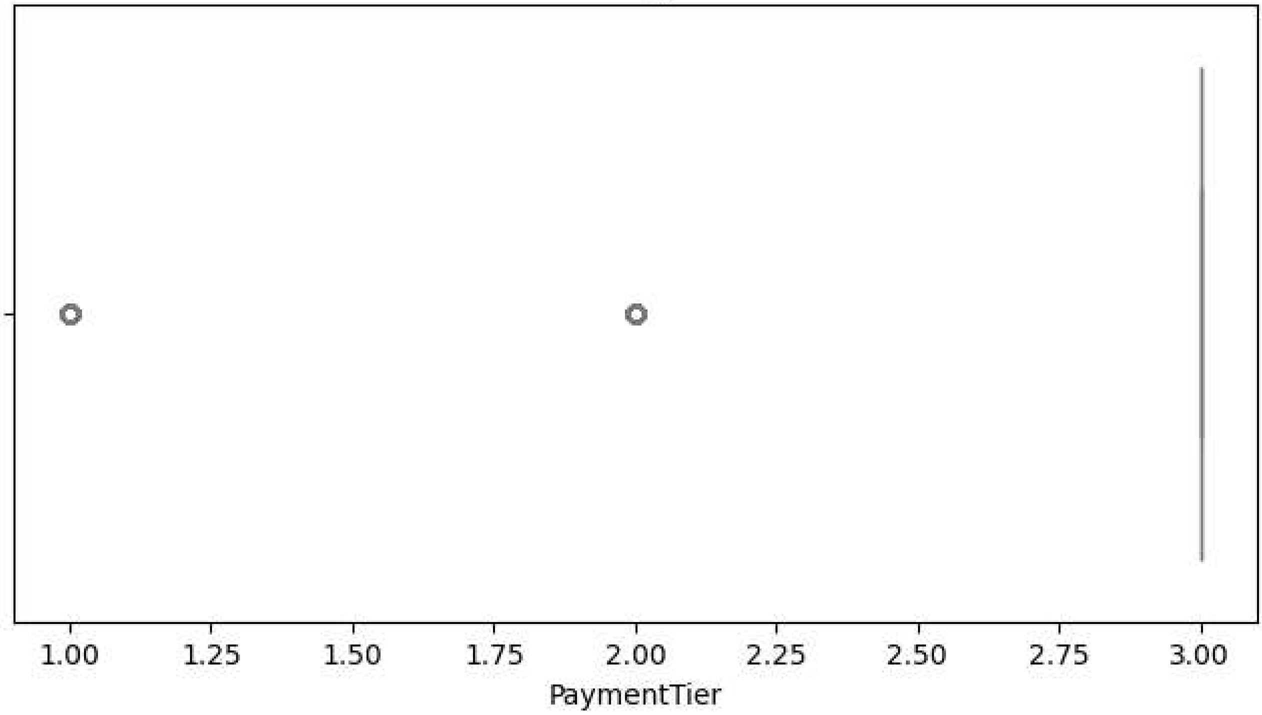

Box Plot - Education
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Education


Box Plot - JoiningYear
2012
2013
2014
2015
2016
2017
2018
JoiningYear

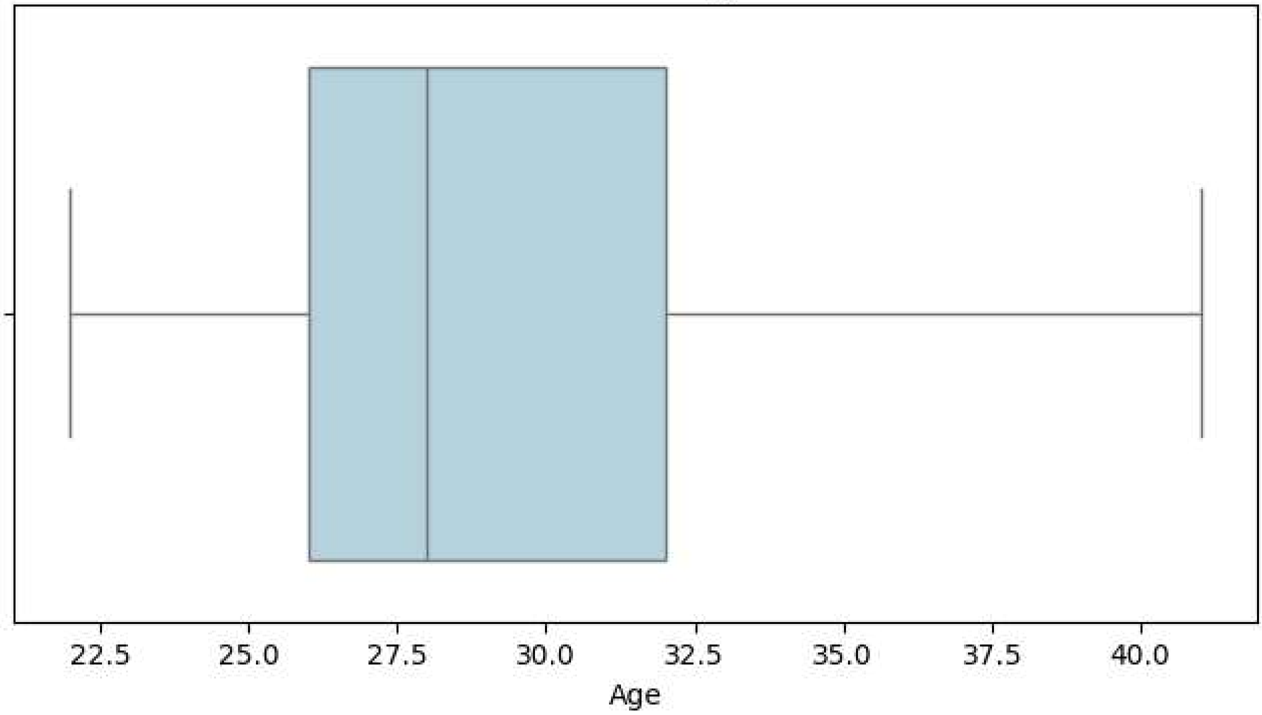

Box Plot - City



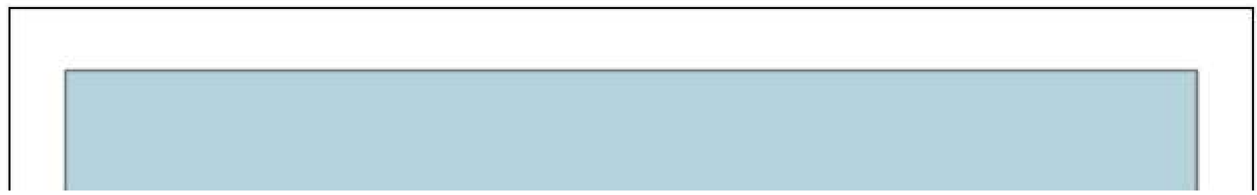
Box Plot - PaymentTier

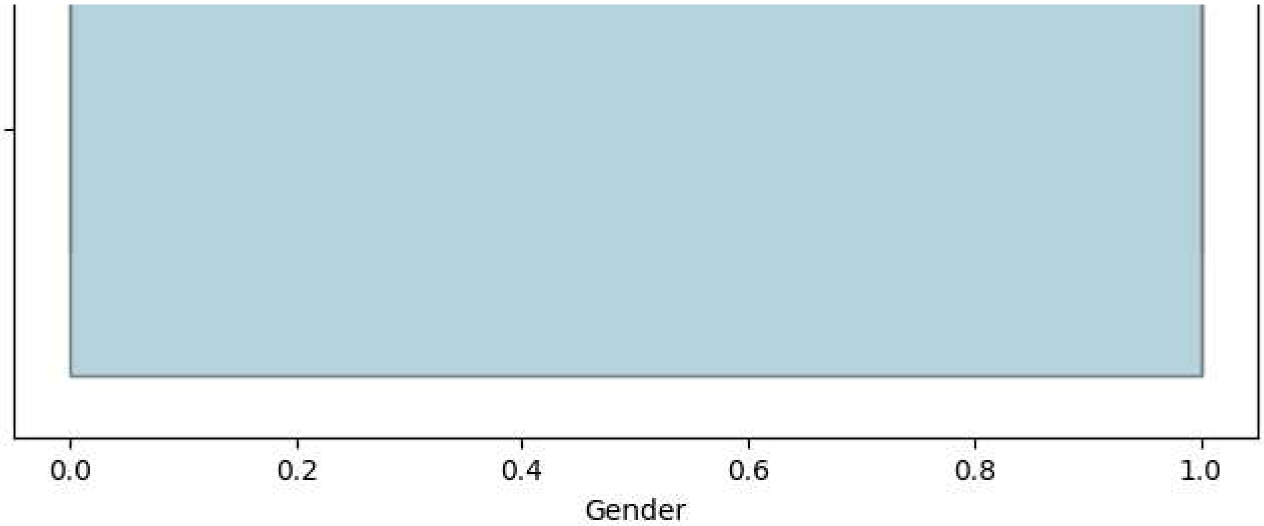


Box Plot - Age

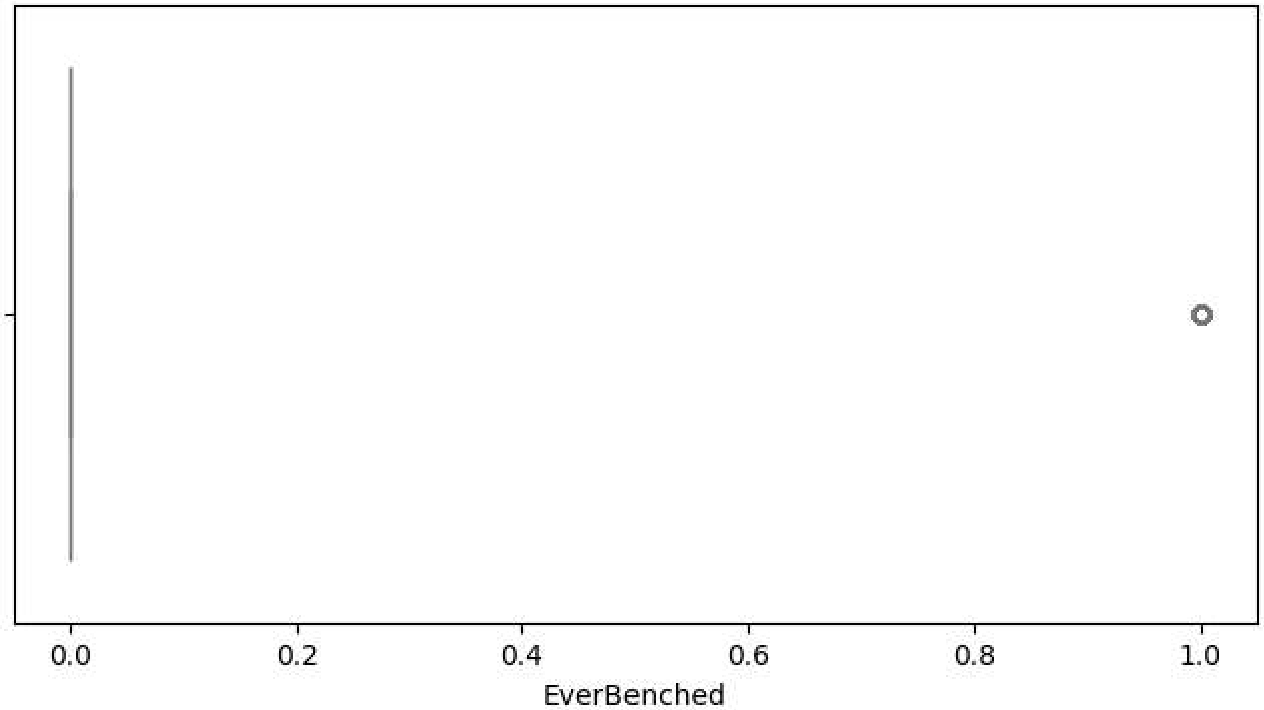


Box Plot - Gender

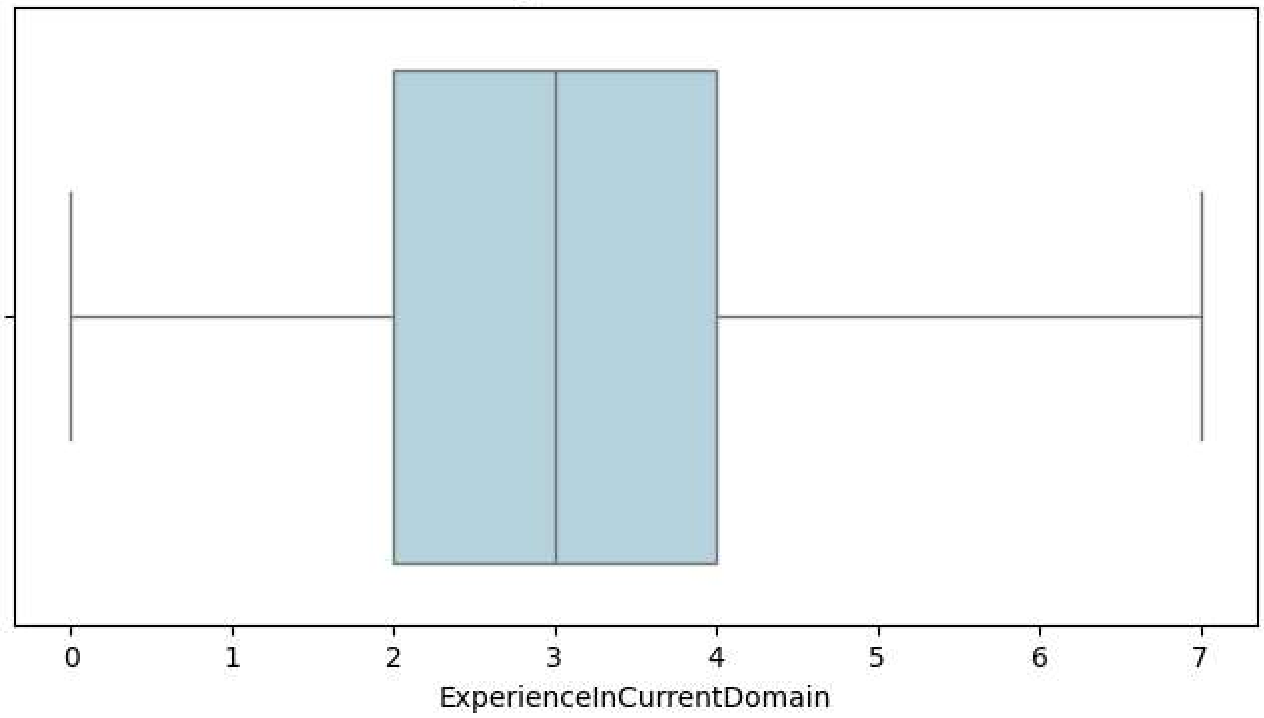


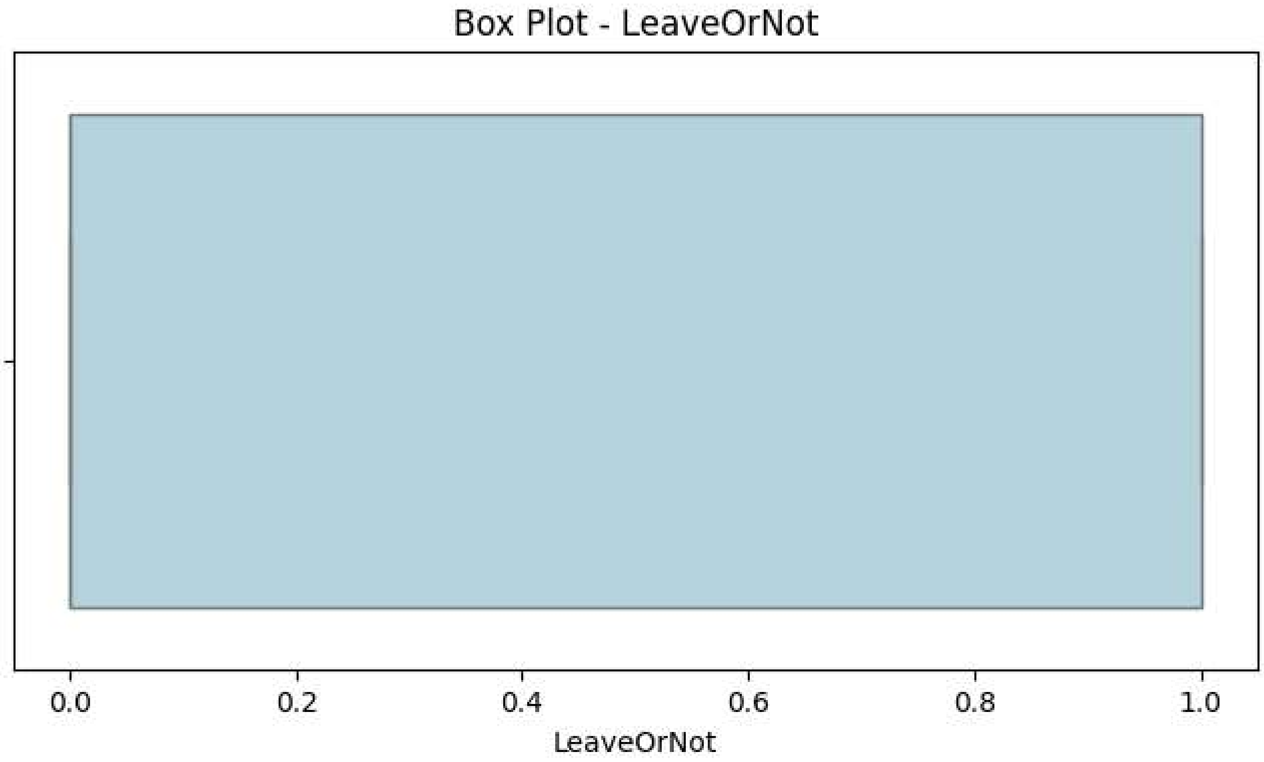


Box Plot - EverBenched



Box Plot - ExperienceInCurrentDomain




Box Plot - LeaveOrNot
0.0
0.2
0.4
0.6
0.8
1.0
LeaveOrNot

```
# 4. Handle Missing Values
print("\nMissing values in the dataset:")
print(df.isnull().sum())
```

```
Missing values in the dataset:
Education                    0
JoiningYear                  0
City                         0
PaymentTier                  0
Age                          0
Gender                       0
EverBenched                  0
ExperienceInCurrentDomain    0
LeaveOrNot                   0
dtype: int64
```

```
####---- DO NOT RUN ----#####
# WE DON'T HAVE MISSING VALUES #
# Fill missing values with mean for numerical and mode for categorical columns
from sklearn.impute import SimpleImputer
imputer_num = SimpleImputer(strategy='mean')
imputer_cat = SimpleImputer(strategy='most_frequent')

for col in df.columns:
    if df[col].dtype == 'object':
        df[col] = imputer_cat.fit_transform(df[[col]])
    else:
        df[col] = imputer_num.fit_transform(df[[col]])

print("\nMissing values handled. Updated dataset info:")
print(df.info())
```

```
# 5. Handle Outliers
for col in numerical_columns:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Replace outliers with upper and lower bounds
    df[col] = np.where(df[col] < lower_bound, lower_bound, df[col])
    df[col] = np.where(df[col] > upper_bound, upper_bound, df[col])
print("\nOutliers handled in numerical features.")
```

```
# 6. Check Dataset Balance
target_col = 'LeaveOrNot'
print(f"\nTarget column distribution:")
print(df[target_col].value_counts())
```

```
Target column distribution:
```

```
LeaveOrNot
0.0    3053
1.0    1600
Name: count, dtype: int64
```

```
# Visualize class distribution
sns.countplot(x=df[target_col], palette='pastel')
plt.title('Target Class Distribution')
plt.show()
```

```
<ipython-input-21-5fca87e12bf9>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.

  sns.countplot(x=df[target_col], palette='pastel')
```