Project 3 : HR Data Analysis

Importing libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv('C:/Users/Ayush/Desktop/Afame Tech/DA Project Details/HR Data.csv')
```

In [3]:
```
df.shape
```

Out[3]:
```
(1470, 35)
```

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',
       'Over18', 'OverTime', 'PercentSalaryHike', 'PerformanceRating',
       'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
       'YearsWithCurrManager'],
      dtype='object')
```

In [5]:
```
df.dtypes
```

Out[5]:
```
Age                         int64
Attrition                  object
BusinessTravel             object
DailyRate                   int64
Department                 object
DistanceFromHome            int64
Education                   int64
EducationField             object
EmployeeCount               int64
EmployeeNumber              int64
EnvironmentSatisfaction     int64
Gender                     object
HourlyRate                  int64
JobInvolvement              int64
JobLevel                    int64
JobRole                    object
JobSatisfaction             int64
MaritalStatus              object
MonthlyIncome               int64
MonthlyRate                 int64
NumCompaniesWorked          int64
Over18                     object
OverTime                   object
PercentSalaryHike           int64
PerformanceRating           int64
RelationshipSatisfaction    int64
StandardHours               int64
StockOptionLevel            int64
TotalWorkingYears           int64
TrainingTimesLastYear       int64
WorkLifeBalance             int64
YearsAtCompany              int64
YearsInCurrentRole          int64
YearsSinceLastPromotion     int64
YearsWithCurrManager        int64
dtype: object
```

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
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 [7]:
```
df.describe(include='object')
```

Out[7]:

	Attrition	BusinessTravel	Department	EducationField	Gender	JobRole	MaritalStatus	Over18	OverTime
count	1470	1470	1470	1470	1470	1470	1470	1470	1470
unique	2	3	3	6	2	9	3	1	2
top	No	Travel_Rarely	Research & Development	Life Sciences	Male	Sales Executive	Married	Y	No
freq	1233	1043	961	606	882	326	673	1470	1054

In [8]:
```
df.describe()
```

Out[8]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvement	JobLevel	...	RelationshipSatisfaction	StandardHours	StockOptionLev
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	...	1470.000000	1470.0	1470.0000
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.721769	65.891156	2.729932	2.063946	...	2.712245	80.0	0.7938
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.093082	20.329428	0.711561	1.106940	...	1.081209	0.0	0.8520
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.000000	30.000000	1.000000	1.000000	...	1.000000	80.0	0.0000
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.000000	48.000000	2.000000	1.000000	...	2.000000	80.0	0.0000
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.000000	66.000000	3.000000	2.000000	...	3.000000	80.0	1.0000
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.000000	83.750000	3.000000	3.000000	...	4.000000	80.0	1.0000
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.000000	100.000000	4.000000	5.000000	...	4.000000	80.0	3.0000

8 rows × 26 columns

This dataset is containing information about employees of a company. Here is a description of each column:

- **Age**: The age of the employee (integer).
- **Attrition**: Whether the employee has left the company or not (object).
- **BusinessTravel**: Frequency of business travel (object).
- **DailyRate**: The daily rate of pay for the employee (integer).
- **Department**: Department in which the employee works (object).
- **DistanceFromHome**: Distance from home to work in miles (integer).
- **Education**: Level of education of the employee (integer).
- **EducationField**: Field of education of the employee (object).
- **EmployeeCount**: Number of employees (always 1) (integer).
- **EmployeeNumber**: Unique identifier for each employee (integer).
- **EnvironmentSatisfaction**: Satisfaction level with the work environment (integer).
- **Gender**: Gender of the employee (object).
- **HourlyRate**: Hourly rate of pay for the employee (integer).
- **JobInvolvement**: Level of job involvement (integer).
- **JobLevel**: Level of job within the company (integer).
- **JobRole**: Role of the employee in the company (object).
- **JobSatisfaction**: Satisfaction level with the job (integer).
- **MaritalStatus**: Marital status of the employee (object).
- **MonthlyIncome**: Monthly income of the employee (integer).
- **MonthlyRate**: Monthly rate of pay for the employee (integer).
- **NumCompaniesWorked**: Number of companies the employee has worked for (integer).
- **Over18**: Whether the employee is over 18 years old (object).
- **OverTime**: Whether the employee works overtime or not (object).
- **PercentSalaryHike**: Percentage increase in salary (integer).
- **PerformanceRating**: Performance rating of the employee (integer).
- **RelationshipSatisfaction**: Satisfaction level with work relationships (integer).
- **StandardHours**: Standard number of working hours (always 80) (integer).
- **StockOptionLevel**: Level of stock option (integer).
- **TotalWorkingYears**: Total number of years worked (integer).
- **TrainingTimesLastYear**: Number of training sessions attended last year (integer).

- **WorkLifeBalance**: Level of work-life balance (integer).
- **YearsAtCompany**: Number of years spent at the company (integer).
- **YearsInCurrentRole**: Number of years in the current role (integer).
- **YearsSinceLastPromotion**: Number of years since the last promotion (integer).
- **YearsWithCurrManager**: Number of years with the current manager (integer).

In [9]:
```
df.head()
```

Out[9]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	Trai
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	1	80	0	8	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	4	80	1	10	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...	2	80	0	7	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...	3	80	0	8	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...	4	80	1	6	

5 rows × 35 columns

In [10]:
```
df.columns
```

Out[10]:
```
Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',
       'Over18', 'OverTime', 'PercentSalaryHike', 'PerformanceRating',
       'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
       'YearsWithCurrManager'],
      dtype='object')
```

In [11]:
```
df['PerformanceSatisfactionIndex'] = df[['EnvironmentSatisfaction', 'JobSatisfaction', 'RelationshipSatisfaction']].mean(axis=1)
```

In [12]:
```
df['ExperiencePriorToCurrentJob'] = df['TotalWorkingYears'] - df['YearsAtCompany']
```

In [13]:
```
df['EmployeeEngagementScore'] = (df['JobInvolvement'] + df['PerformanceRating'] + df['WorkLifeBalance']) / 3
```

In [14]:
```
education_map = {
    1: 'Below College',
    2: 'College',
    3: 'Bachelor',
    4: 'Master',
    5: 'Doctor'
}

environment_satisfaction_map = {
    1: 'Low',
    2: 'Medium',
    3: 'High',
    4: 'Very High'
}
```

```
job_involvement_map = {
    1: 'Low',
    2: 'Medium',
    3: 'High',
    4: 'Very High'
}

job_satisfaction_map = {
    1: 'Low',
    2: 'Medium',
    3: 'High',
    4: 'Very High'
}

performance_rating_map = {
    1: 'Low',
    2: 'Good',
    3: 'Excellent',
    4: 'Outstanding'
}

relationship_satisfaction_map = {
    1: 'Low',
    2: 'Medium',
    3: 'High',
    4: 'Very High'
}

work_life_balance_map = {
    1: 'Bad',
    2: 'Good',
    3: 'Better',
    4: 'Best'
}
```

In [15]:
```
df['Education'] = df['Education'].map(education_map)
df['EnvironmentSatisfaction'] = df['EnvironmentSatisfaction'].map(environment_satisfaction_map)
df['JobInvolvement'] = df['JobInvolvement'].map(job_involvement_map)
df['JobSatisfaction'] = df['JobSatisfaction'].map(job_satisfaction_map)
df['PerformanceRating'] = df['PerformanceRating'].map(performance_rating_map)
df['RelationshipSatisfaction'] = df['RelationshipSatisfaction'].map(relationship_satisfaction_map)
df['WorkLifeBalance'] = df['WorkLifeBalance'].map(work_life_balance_map)
df['Overtime_Preference'] = df['OverTime'].map({'Yes': 1, 'No': 0})
```

In [16]:
```
def generation(age):
    if (age >= 97) & (age <= 102):
        return 'WWII'
    elif (age >= 79) & (age <= 96):
        return 'Post War'
    elif (age >= 70) & (age <= 78):
        return 'Boomers I*'
    elif (age >= 60) & (age <= 69):
        return 'Boomers II (a/k/a Generation Jones)*'
    elif (age >= 44) & (age <= 59):
        return 'Gen X'
    elif (age >= 28 ) & (age <= 43):
        return 'Millennials'
    elif (age >= 12 ) & (age <= 27):
        return 'Gen Z'
```

In [17]:
```
def distance_category(distance):
    if (distance >= 1) & (distance <= 10):
```

```
        return 'Short Distance'
    elif (distance >= 11) & (distance <= 20):
        return 'Medium Distance'
    elif (distance >= 21) & (distance <= 30):
        return 'Long Distance'
```

In [18]:
```
df['AgeGeneration'] = df['Age'].apply(generation)
df['DistanceFromHomeCategory'] = df['DistanceFromHome'].apply(distance_category)
```

In [19]:
```
df = df.reindex(sorted(df.columns), axis=1)
```

In [20]:
```
df.columns
```

Out[20]:
```
Index(['Age', 'AgeGeneration', 'Attrition', 'BusinessTravel', 'DailyRate',
       'Department', 'DistanceFromHome', 'DistanceFromHomeCategory',
       'Education', 'EducationField', 'EmployeeCount',
       'EmployeeEngagementScore', 'EmployeeNumber', 'EnvironmentSatisfaction',
       'ExperiencePriorToCurrentJob', 'Gender', 'HourlyRate', 'JobInvolvement',
       'JobLevel', 'JobRole', 'JobSatisfaction', 'MaritalStatus',
       'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked', 'Over18',
       'OverTime', 'Overtime_Preference', 'PercentSalaryHike',
       'PerformanceRating', 'PerformanceSatisfactionIndex',
       'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
       'YearsWithCurrManager'],
      dtype='object')
```

Exploratory Data Analysis (Whole Dataset)

Univariate Analysis

In [21]:
```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='Age', bins=20, kde=False, color='red', edgecolor='yellow')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

Most of the employees data is from 30 to 40 years old Age group

In [22]:
```
df['Age'].median()
```

Out[22]:
```
36.0
```

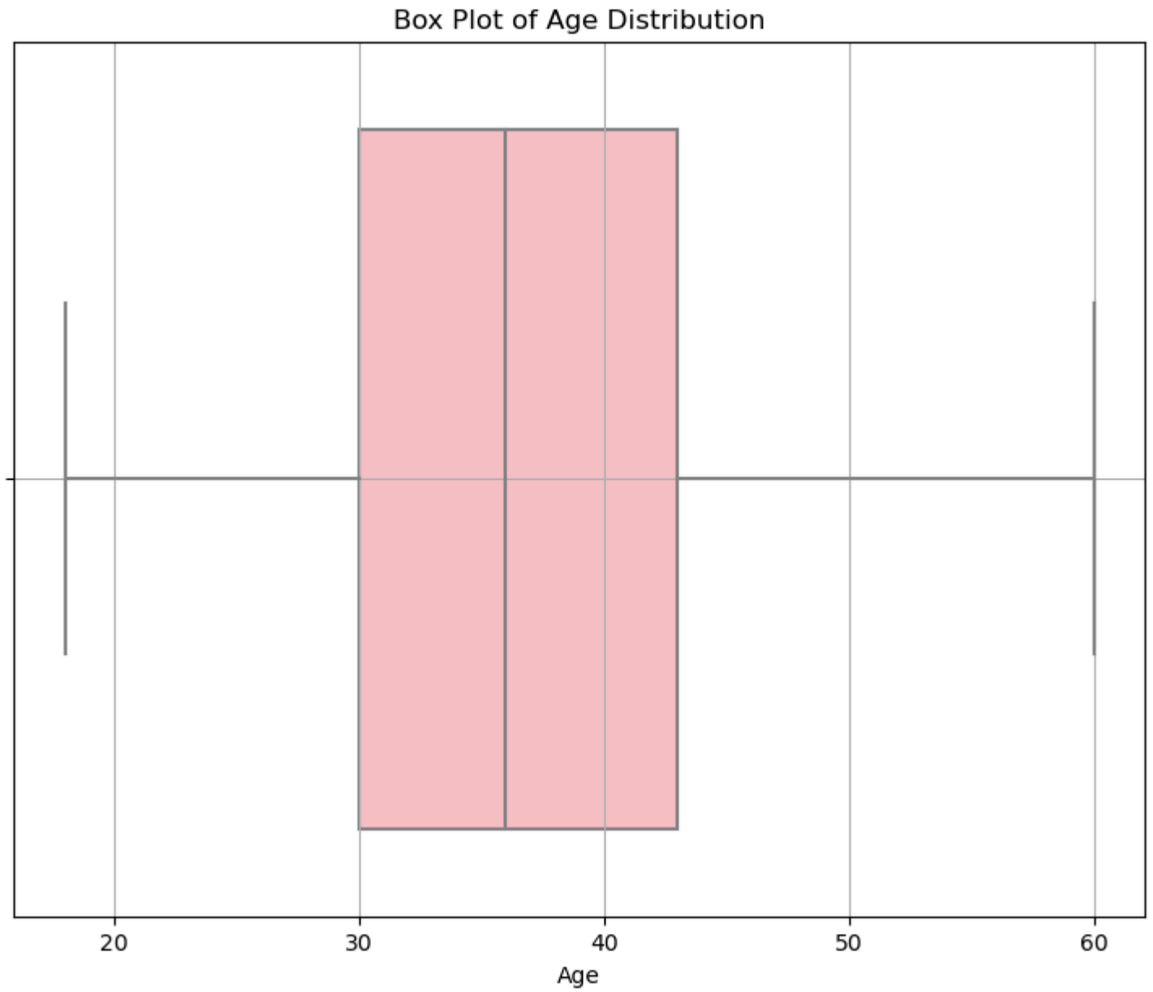
In [23]:
```
# Find quartiles
min_age = df['Age'].min()
max_age = df['Age'].max()

print("Minimum Age:", min_age)
print("Maximum Age:", max_age)
```

```
Minimum Age: 18
Maximum Age: 60
```

In [24]:
```
# Calculate quartiles
Q1 = df['Age'].quantile(0.25)
Q3 = df['Age'].quantile(0.75)

print("IQR Min:", Q1)
print("IQR Max:", Q3)
```

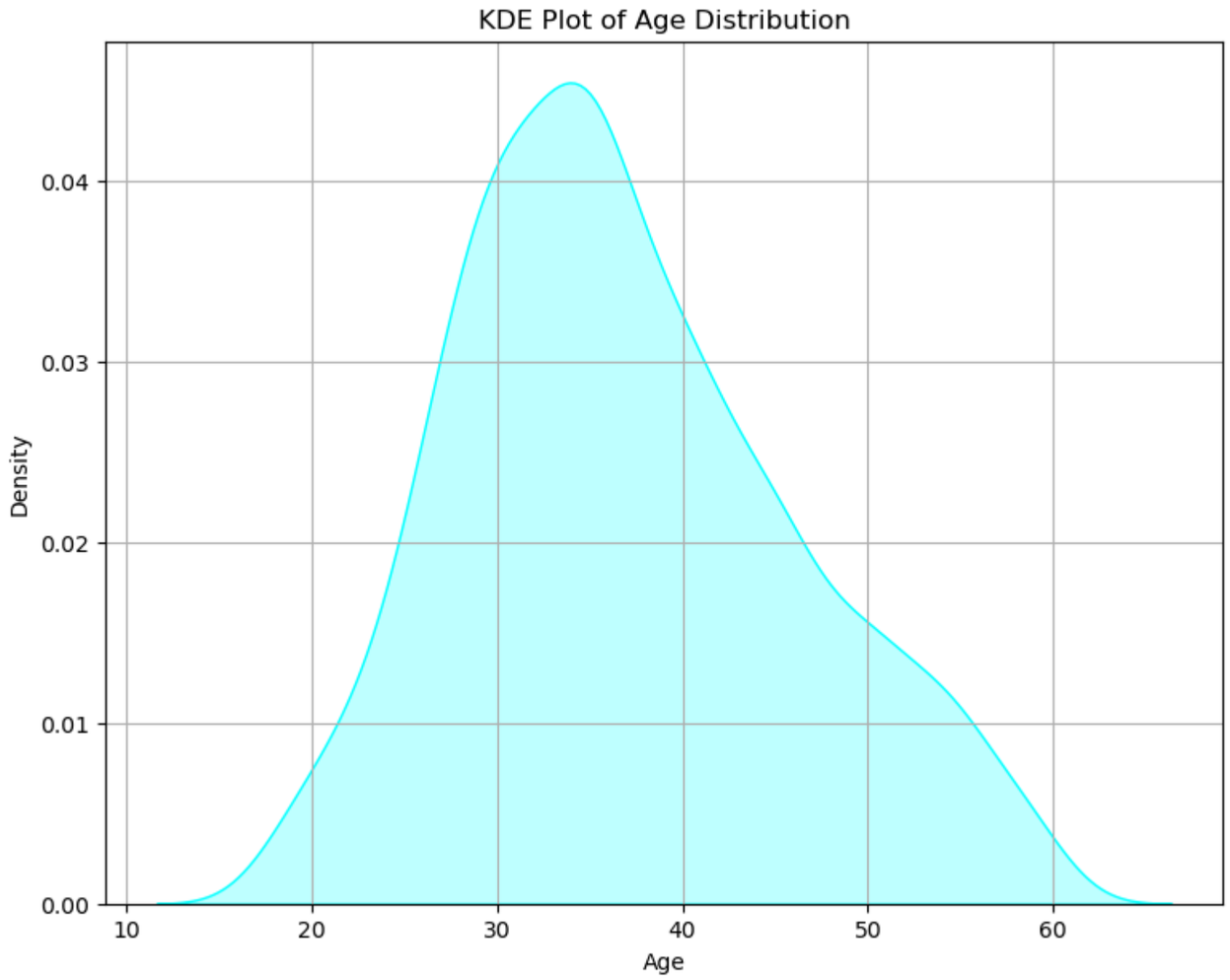
```
IQR Min: 30.0
IQR Max: 43.0
```

In [25]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='Age', color='lightpink')
plt.title('Box Plot of Age Distribution')
plt.xlabel('Age')
plt.grid(True)
plt.show()
```

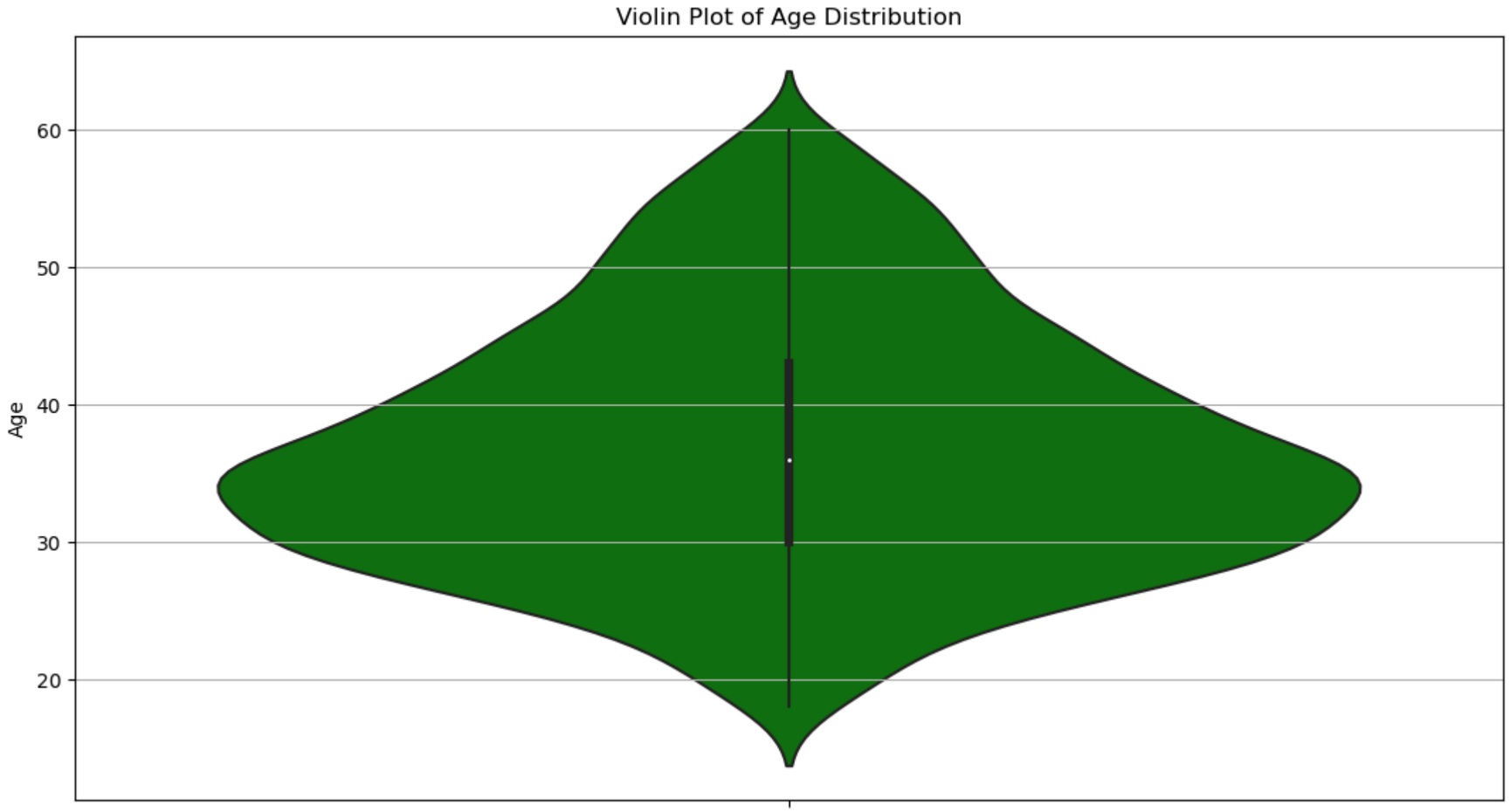


The box plot of the age distribution showing that the median age of employees are around 36 years old, with the interquartile range spanning from approximately 30 to 43 years old. The whiskers extend up to around 60 years old, indicating that most employees fall within this age category.

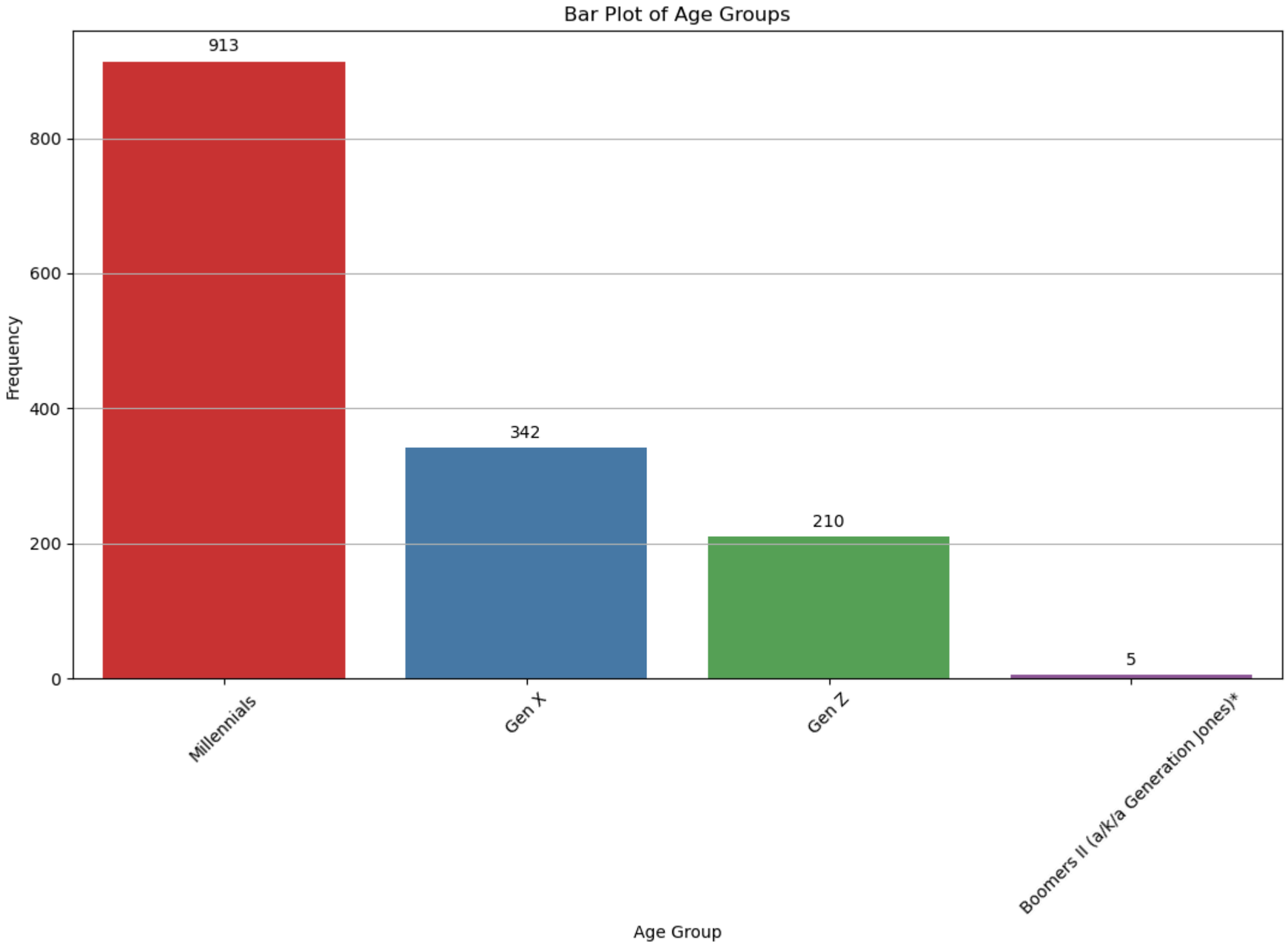
In [26]:
```
plt.figure(figsize=(9, 7))
sns.kdeplot(data=df, x='Age', color='cyan', fill=True)
plt.title('KDE Plot of Age Distribution')
plt.xlabel('Age')
plt.ylabel('Density')
plt.grid(True)
plt.show()
```



In [27]:
```
plt.figure(figsize=(13, 7))
sns.violinplot(data=df, y='Age', color='green')
plt.title('Violin Plot of Age Distribution')
plt.ylabel('Age')
plt.grid(axis='y')
plt.show()
```

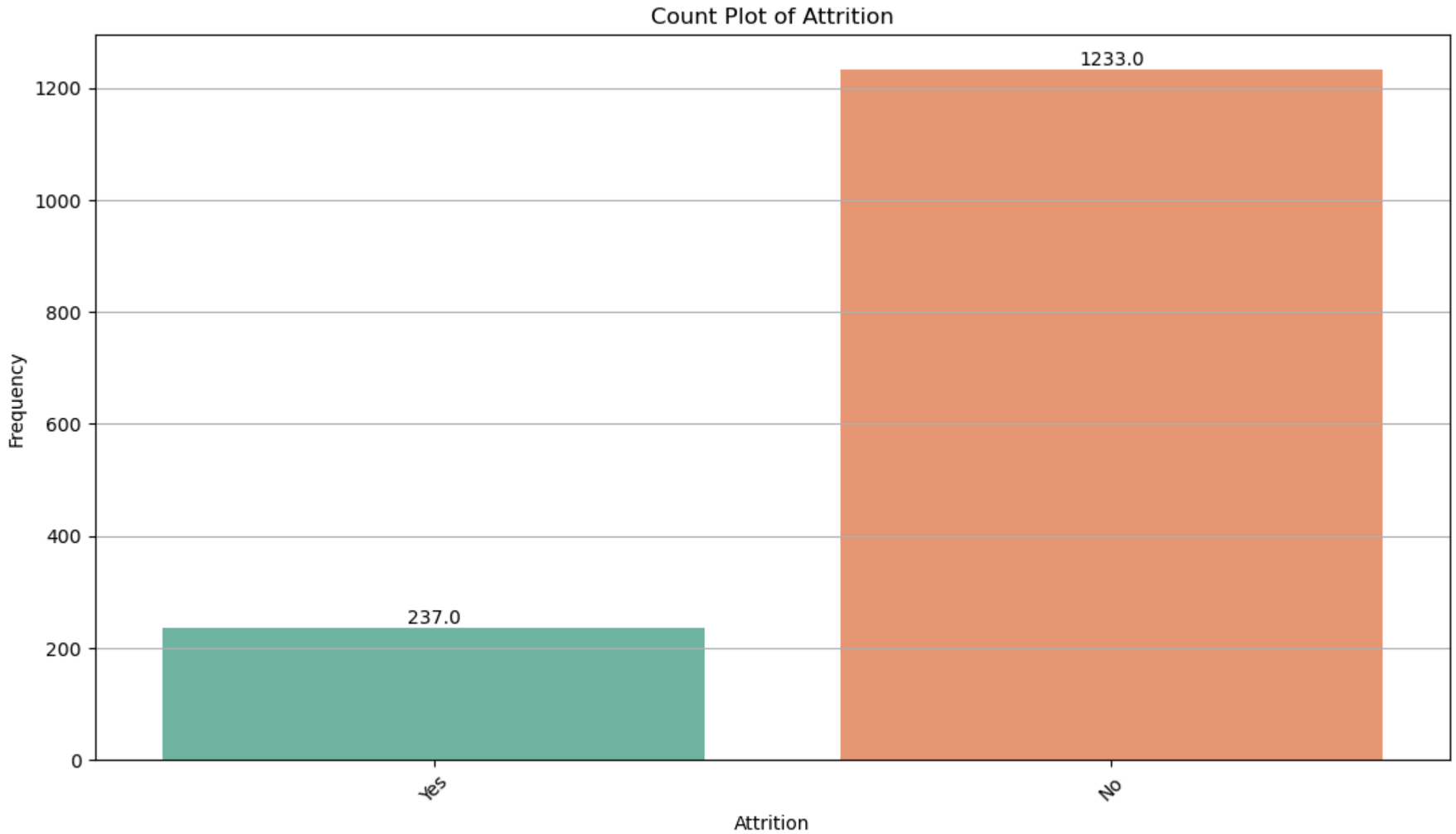


In [30]:
```
plt.figure(figsize=(13, 7))
ax = sns.countplot(data=df, x='AgeGeneration', palette='Set1')
plt.title('Bar Plot of Age Groups')
plt.xlabel('Age Group')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.grid(axis='y')

# Add count annotations to the bars
for p in ax.patches:
    ax.annotate(format(p.get_height(), '.0f'),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha = 'center', va = 'center',
                xytext = (0, 9),
                textcoords = 'offset points')

plt.show()
```

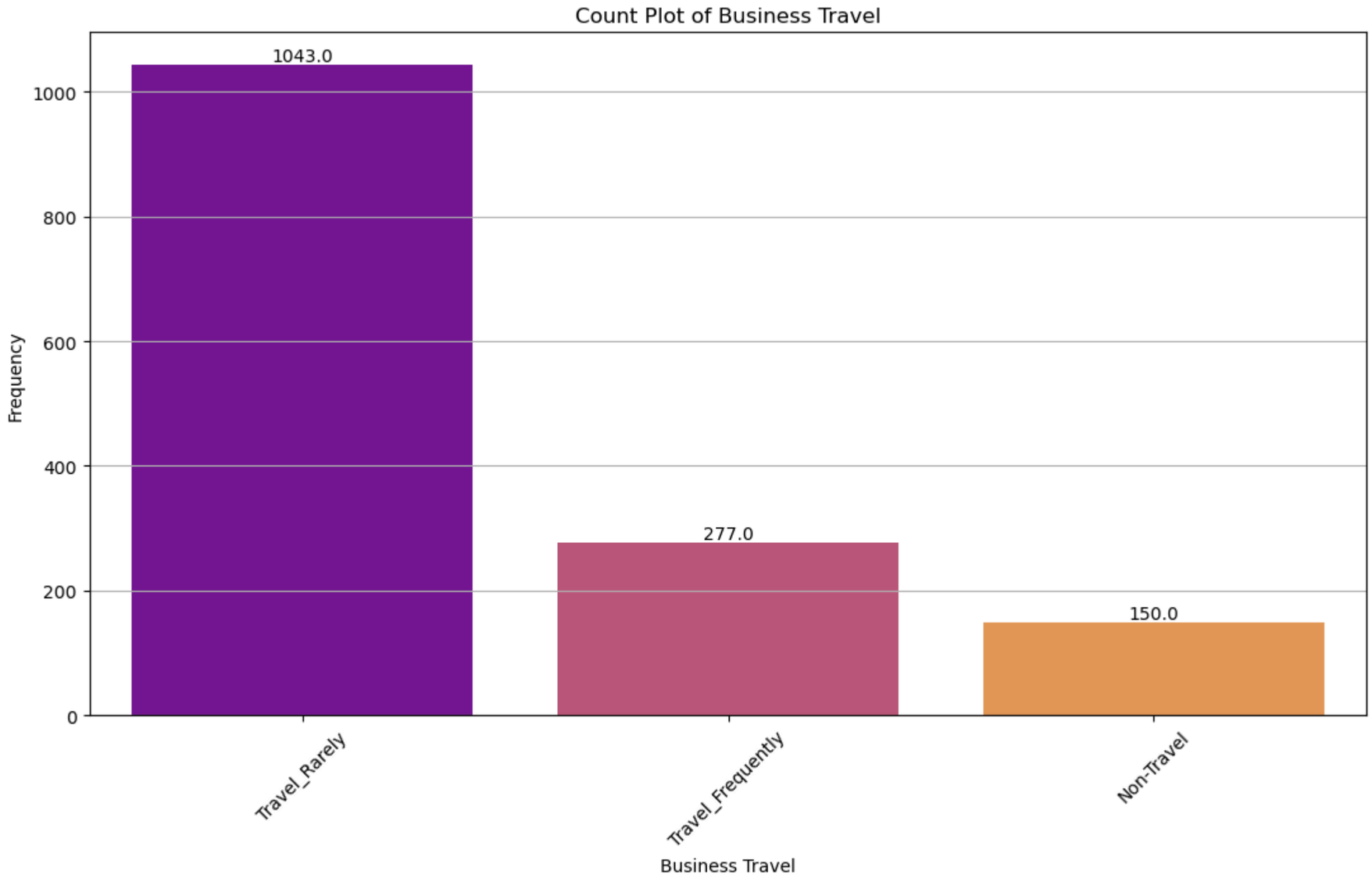


In [31]:
```
plt.figure(figsize=(13, 7))
ax = sns.countplot(data=df, x='Attrition', palette='Set2')

# Adding annotations
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.title('Count Plot of Attrition')
plt.xlabel('Attrition')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
```

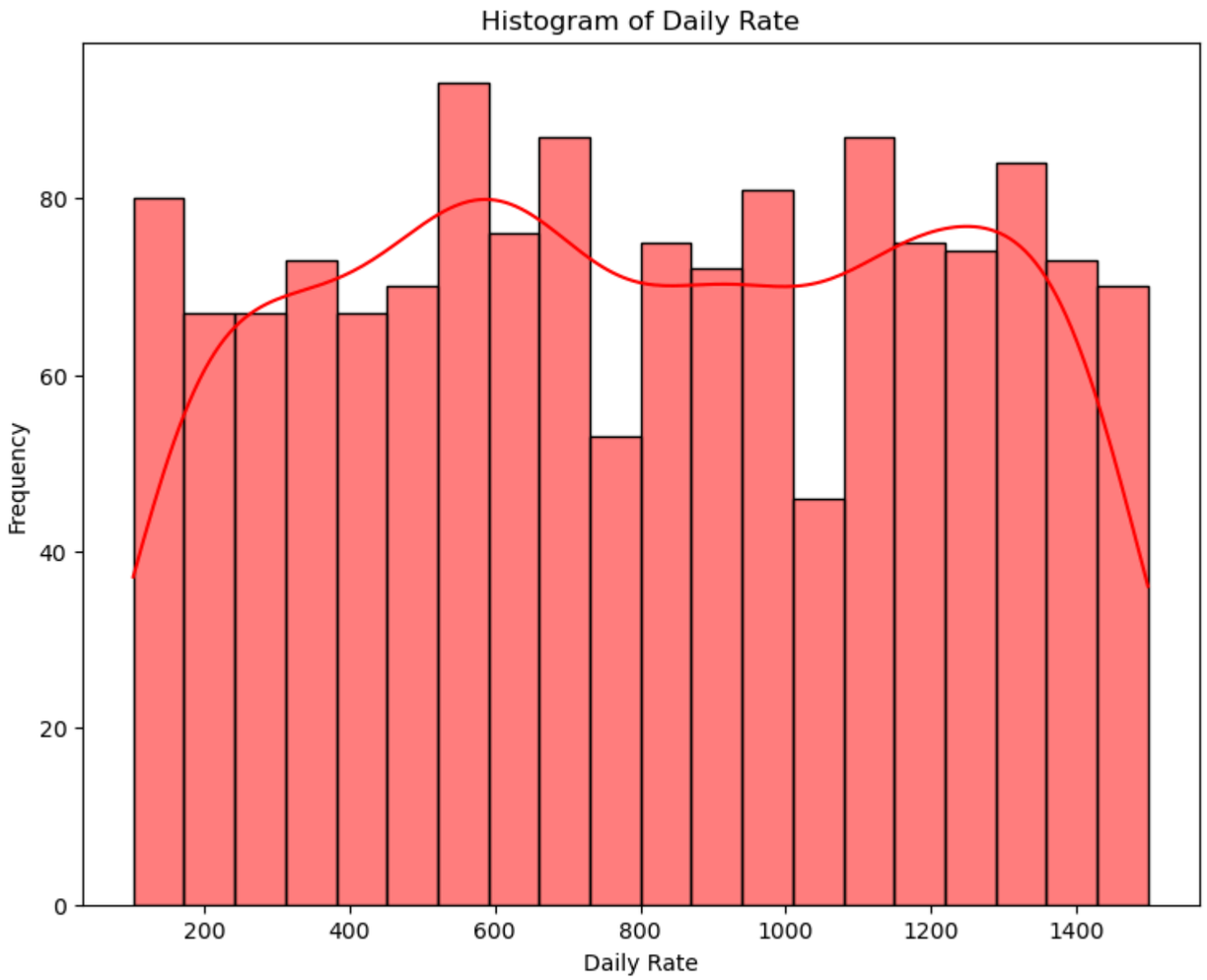
```
plt.grid(axis='y')
plt.show()
```



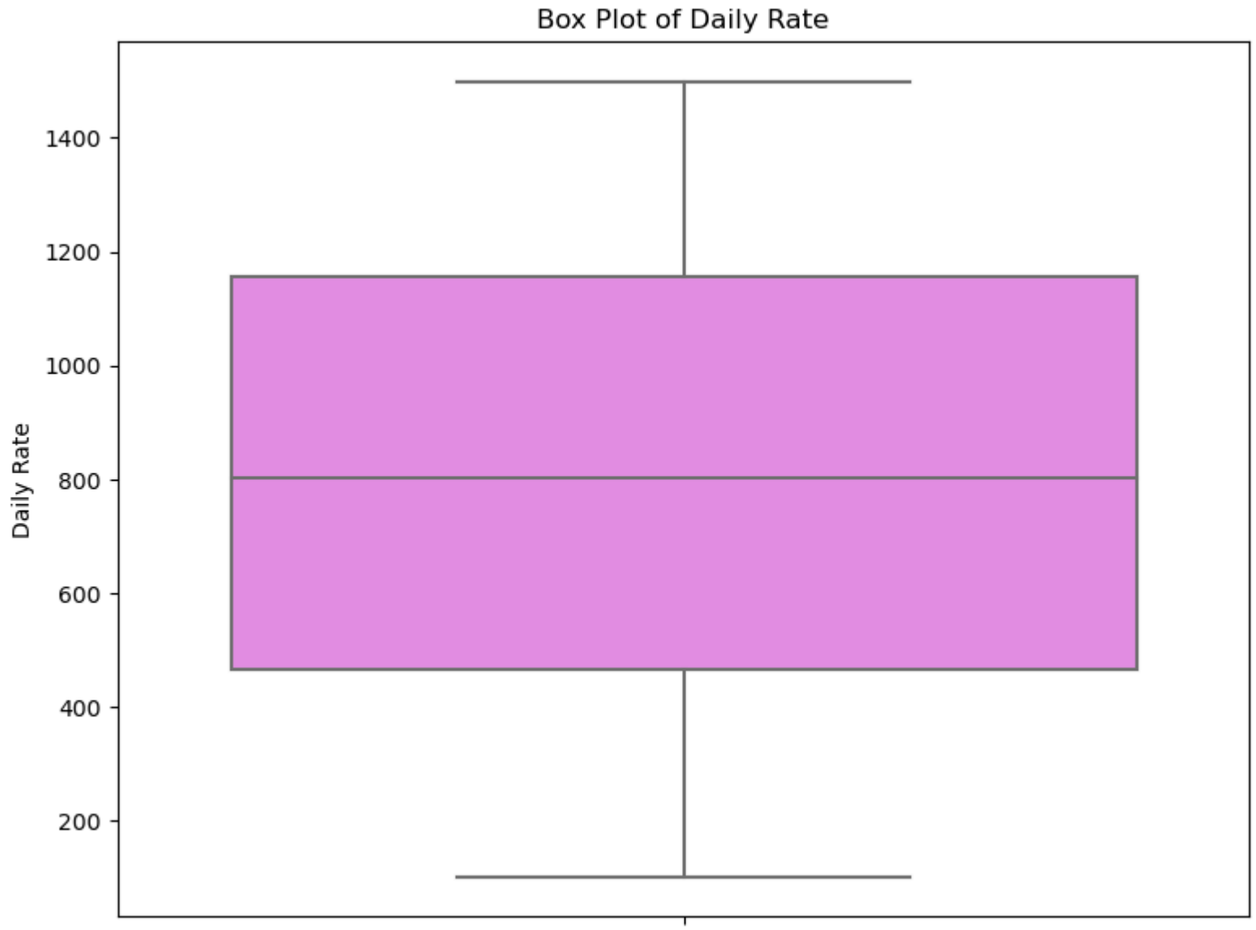
In [32]:
```
plt.figure(figsize=(13, 7))
ax = sns.countplot(data=df, x='BusinessTravel', palette='plasma')
plt.title('Count Plot of Business Travel')
plt.xlabel('Business Travel')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.grid(axis='y')

for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



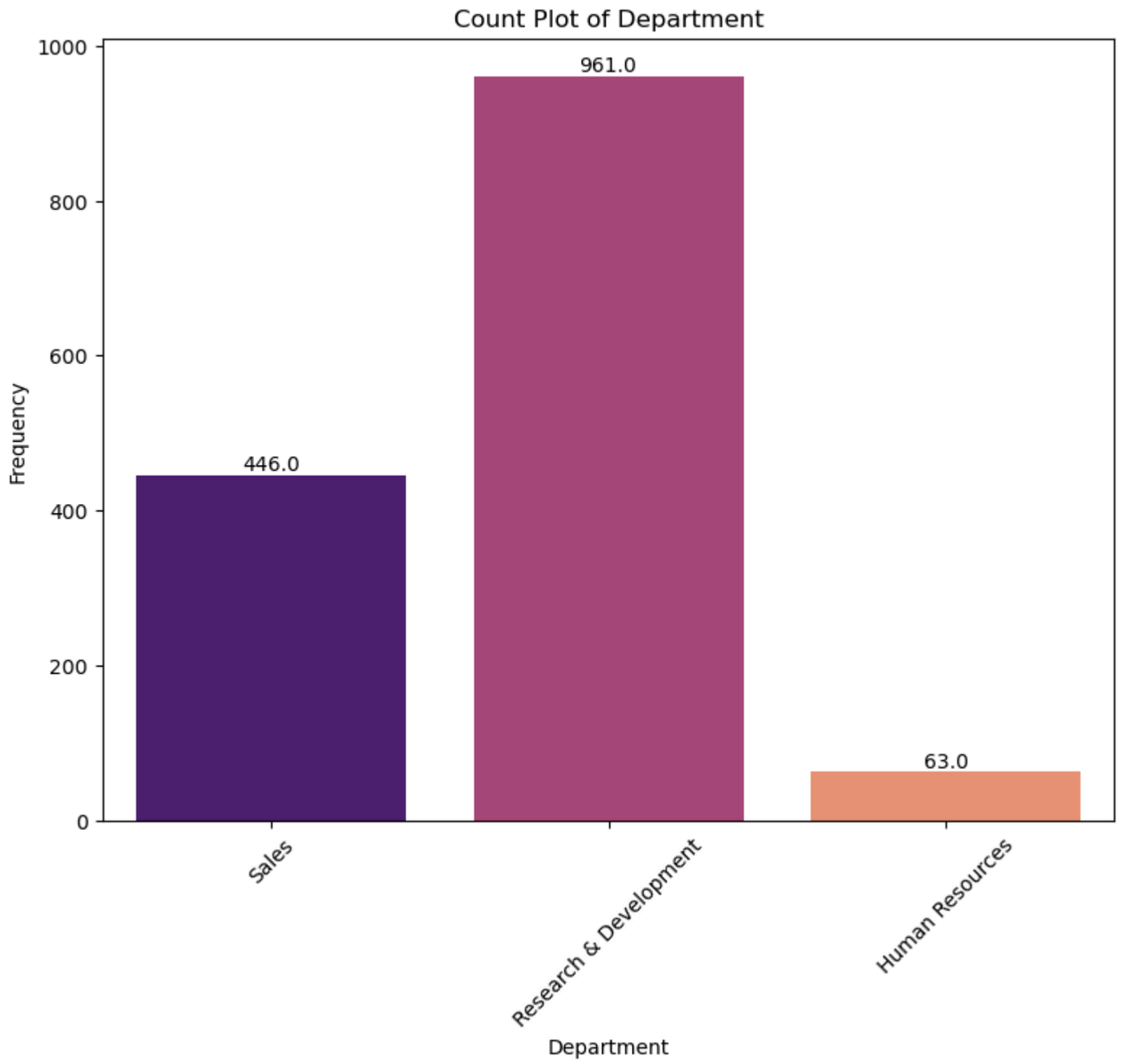
In [33]:
```
plt.figure(figsize=(9, 7))
sns.histplot(df['DailyRate'], bins=20, kde=True, color='red')
plt.title('Histogram of Daily Rate')
plt.xlabel('Daily Rate')
plt.ylabel('Frequency')
plt.show()
```



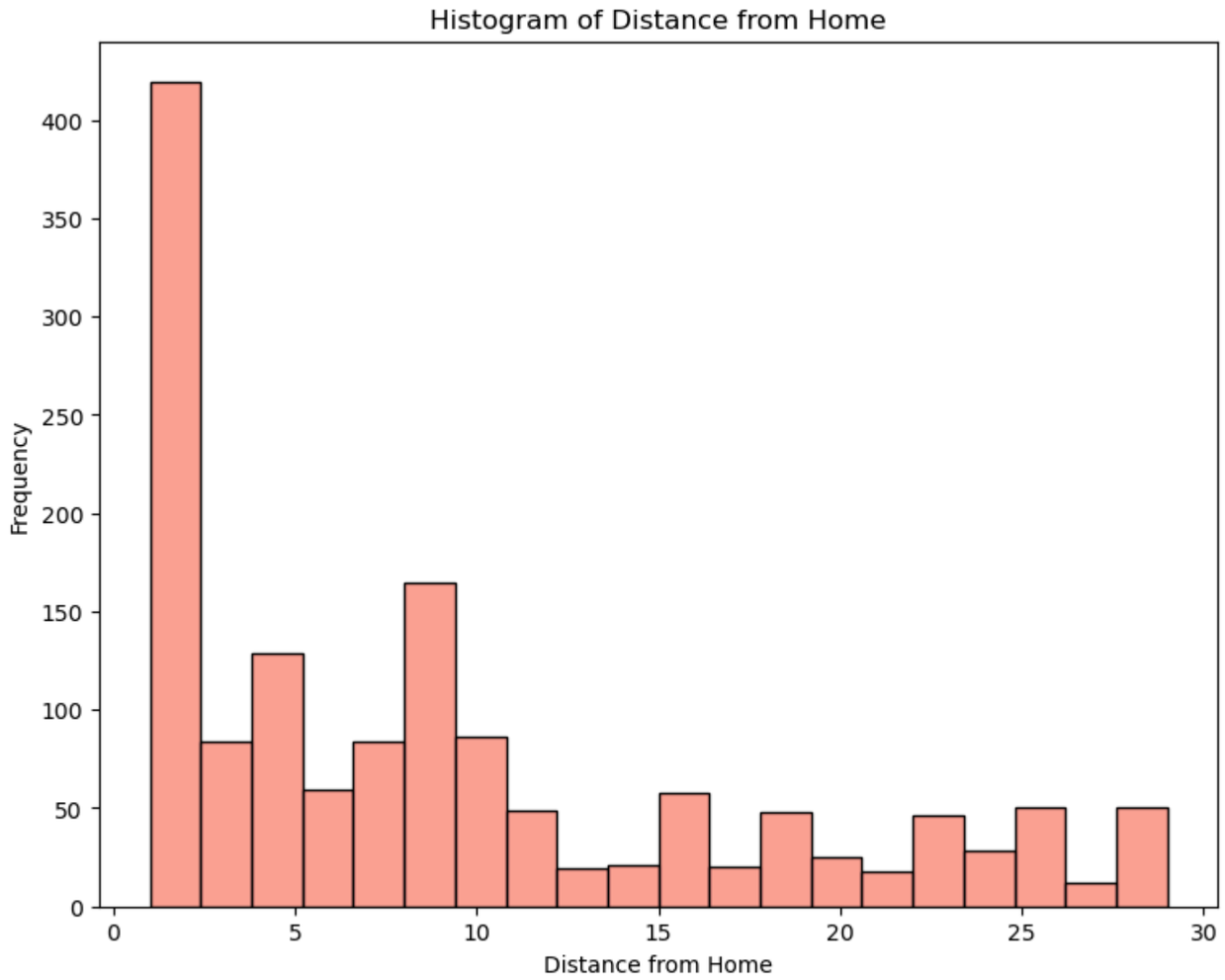
In [34]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, y='DailyRate', color='violet')
plt.title('Box Plot of Daily Rate')
plt.ylabel('Daily Rate')
plt.show()
```



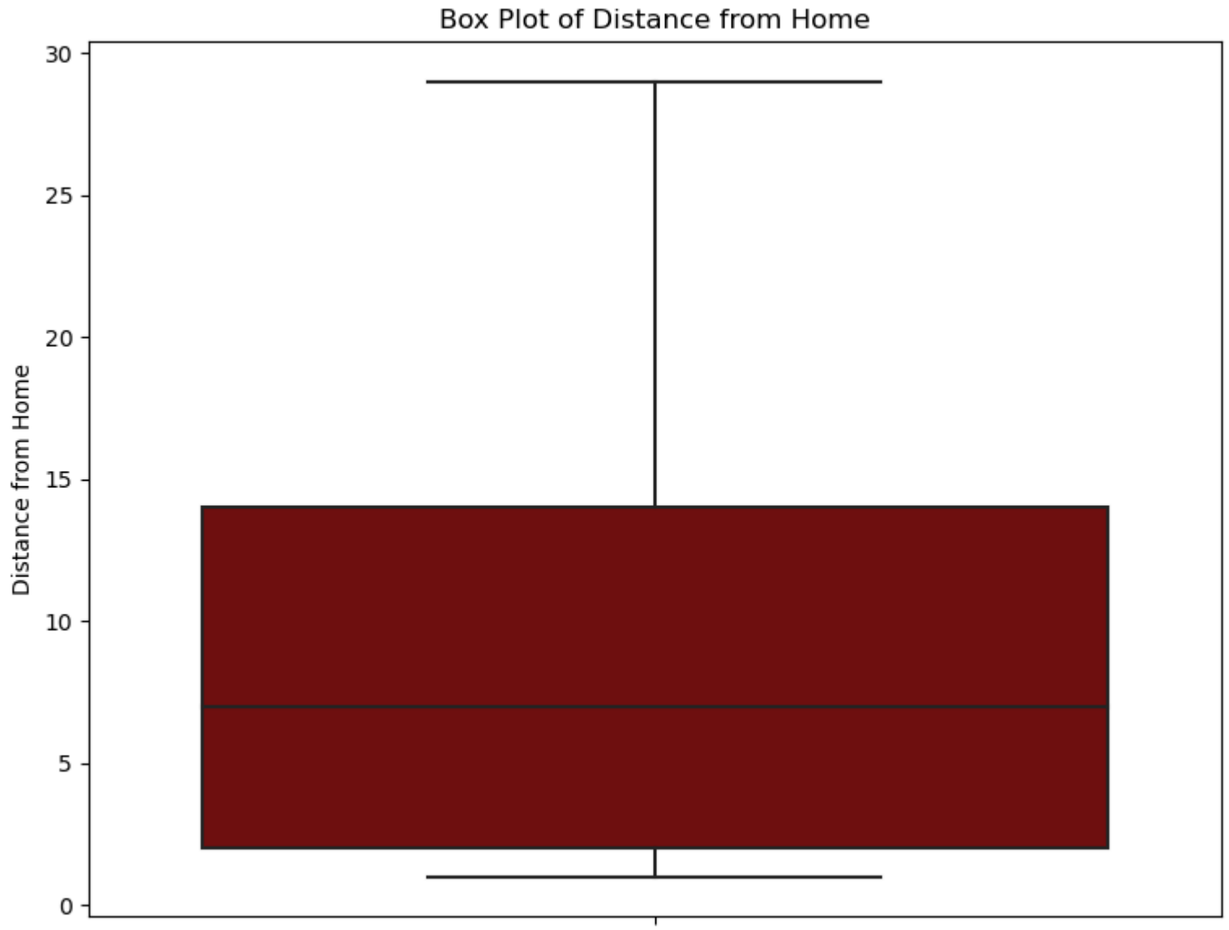
In [35]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='Department', palette='magma')
plt.title('Count Plot of Department')
plt.xlabel('Department')
plt.ylabel('Frequency')
plt.xticks(rotation=45)

for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()
```



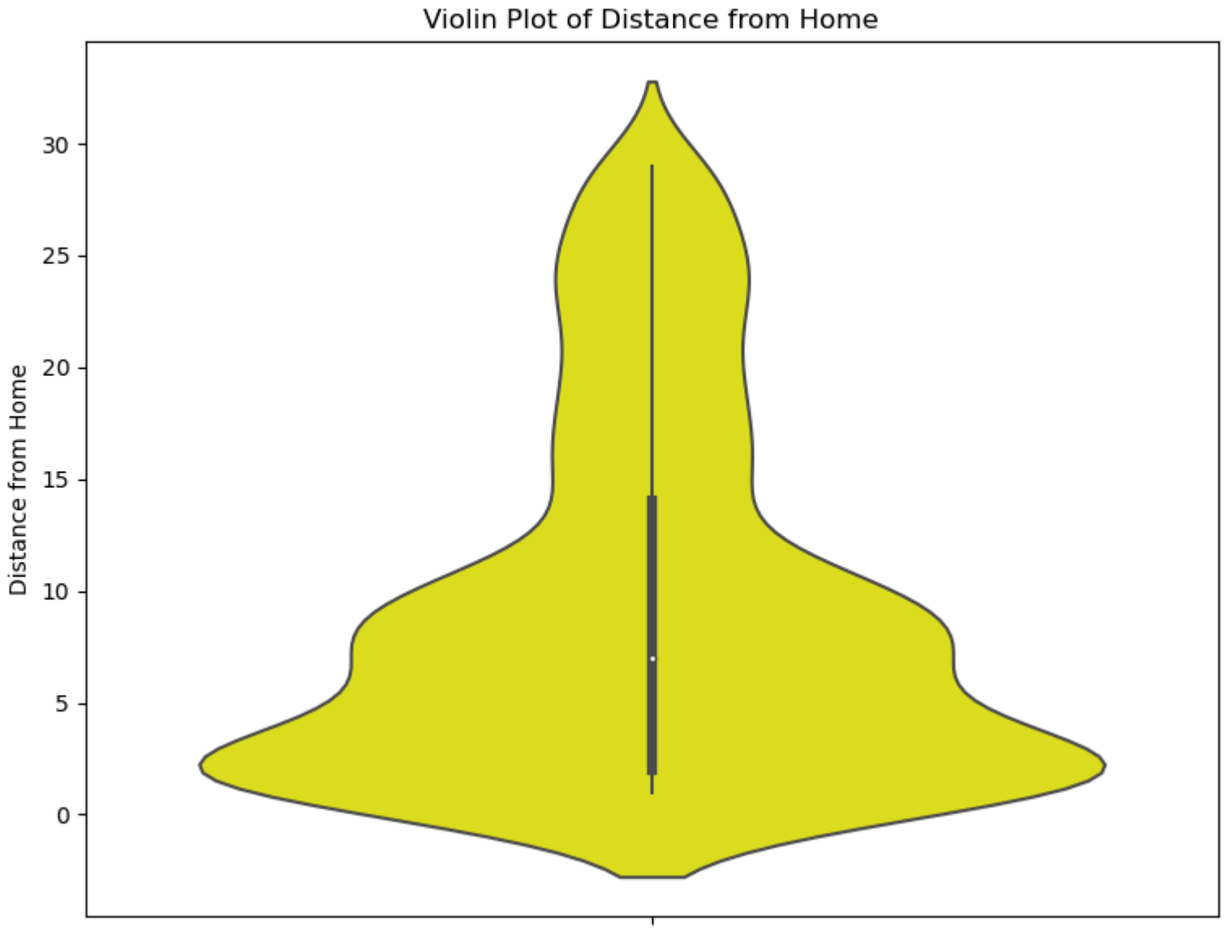
In [36]:
```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='DistanceFromHome', bins=20, kde=False, color='salmon')
plt.title('Histogram of Distance from Home')
plt.xlabel('Distance from Home')
plt.ylabel('Frequency')
plt.show()
```



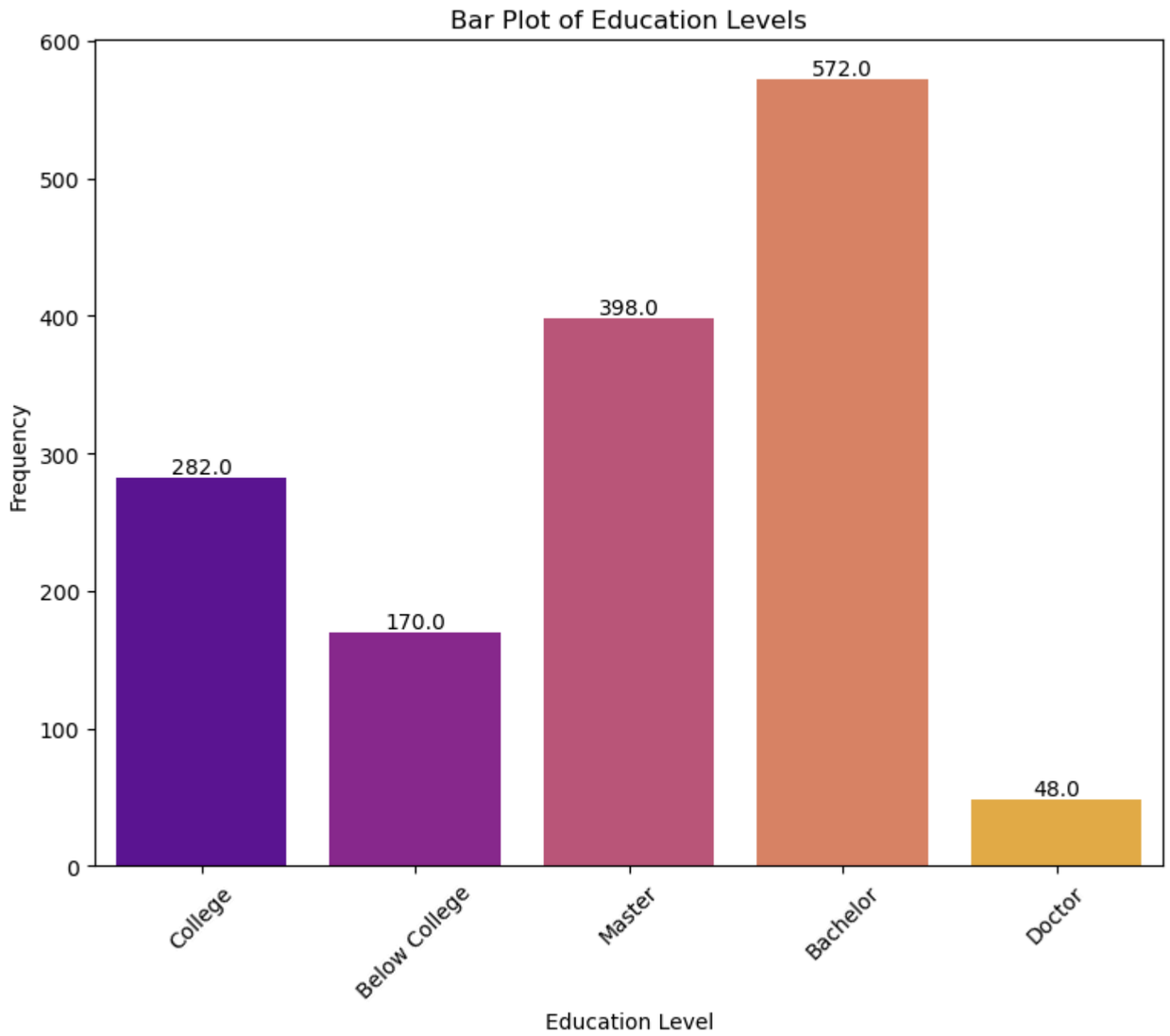
In [39]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, y='DistanceFromHome', color='maroon')
plt.title('Box Plot of Distance from Home')
plt.ylabel('Distance from Home')
plt.show()
```



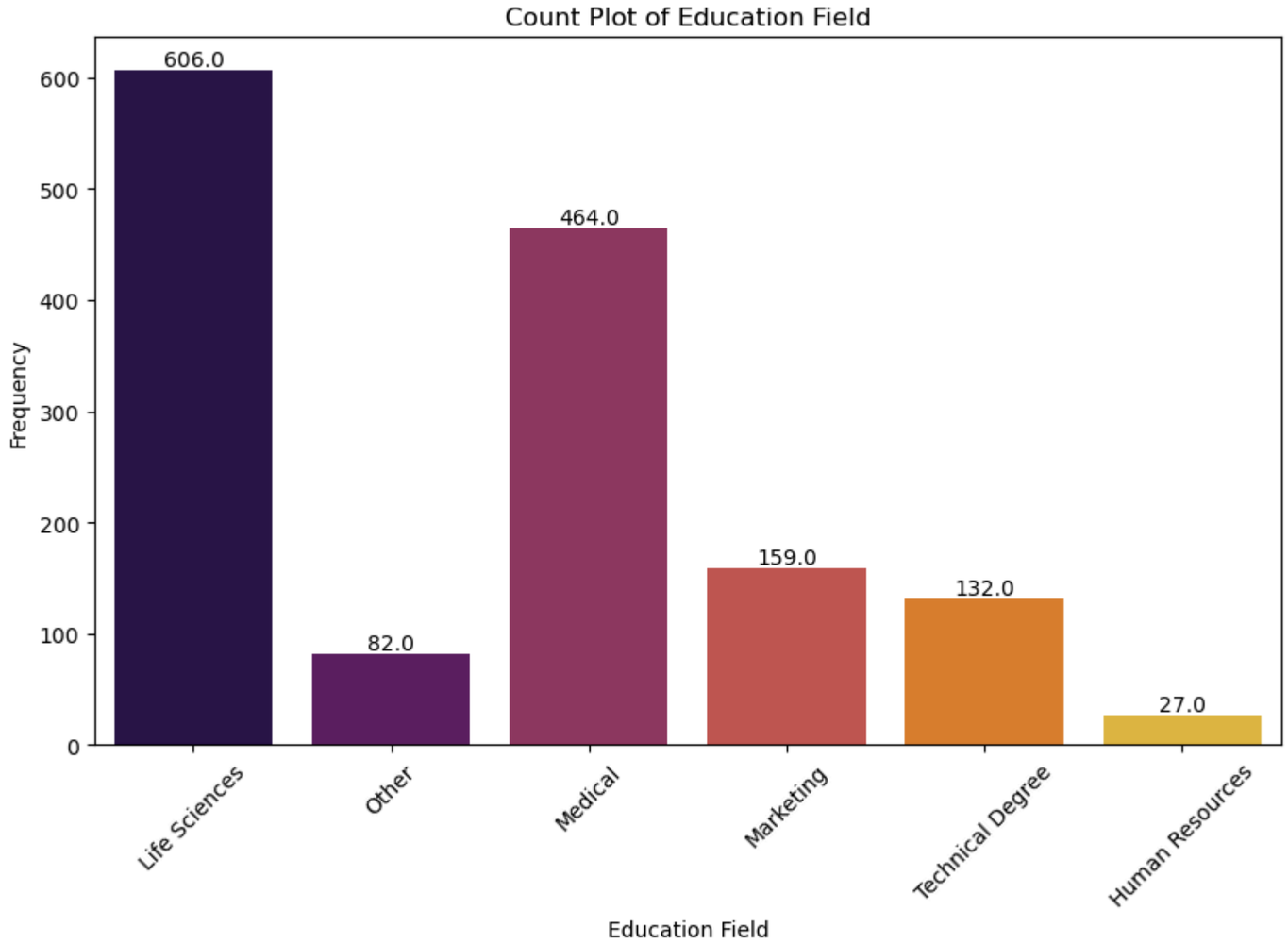
In [40]:
```
plt.figure(figsize=(9, 7))
sns.violinplot(data=df, y='DistanceFromHome', color='yellow')
plt.title('Violin Plot of Distance from Home')
plt.ylabel('Distance from Home')
plt.show()
```



In [44]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='Education', palette='plasma')
plt.title('Bar Plot of Education Levels')
plt.xlabel('Education Level')
plt.ylabel('Frequency')
plt.xticks(rotation=45)

# Add annotations
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black
                ', xytext=(0, 5),
                textcoords='offset points')

plt.show()
```

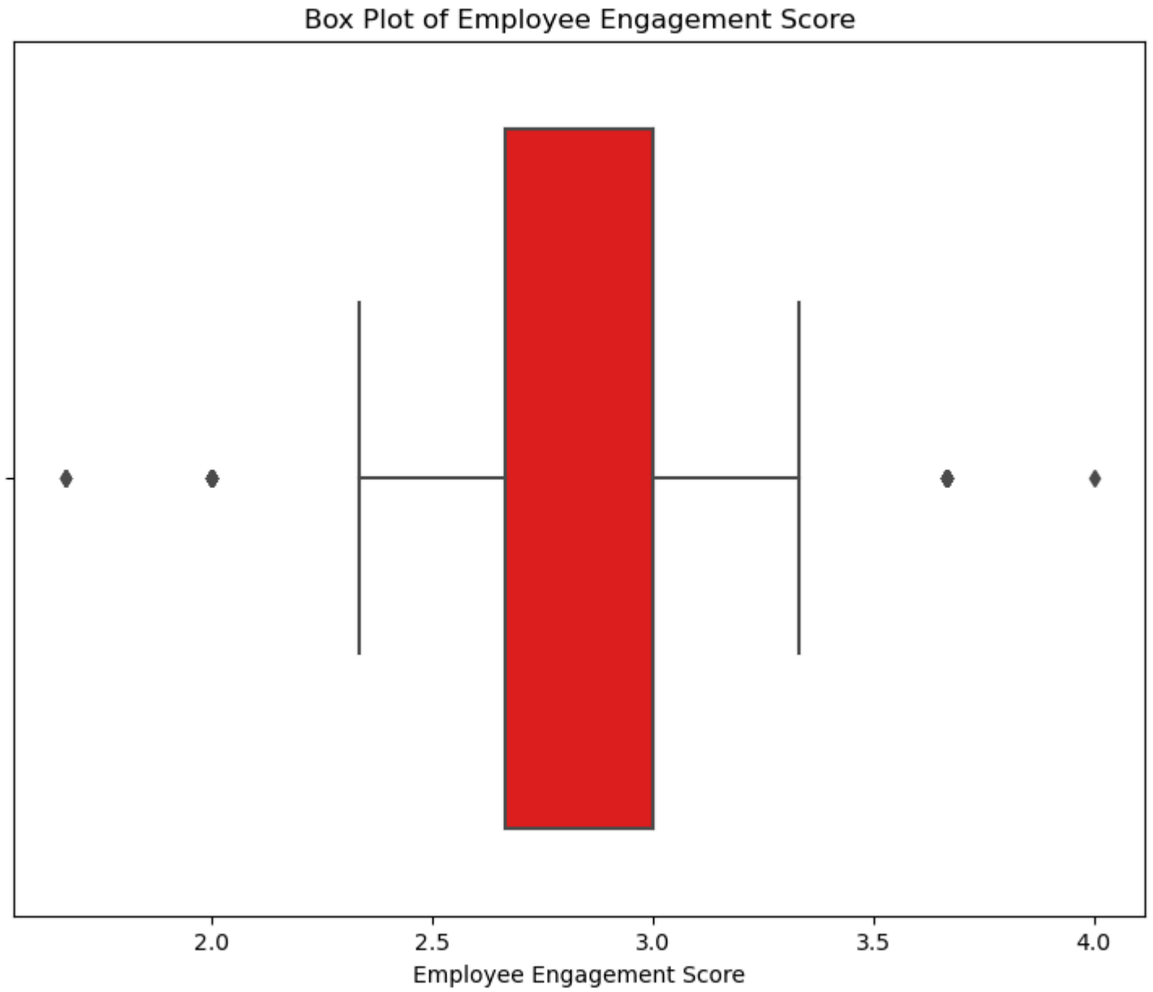


In [45]:
```
plt.figure(figsize=(10, 6))
ax = sns.countplot(data=df, x='EducationField', palette='inferno')
plt.title('Count Plot of Education Field')
plt.xlabel('Education Field')
plt.ylabel('Frequency')
plt.xticks(rotation=45)

for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()
```



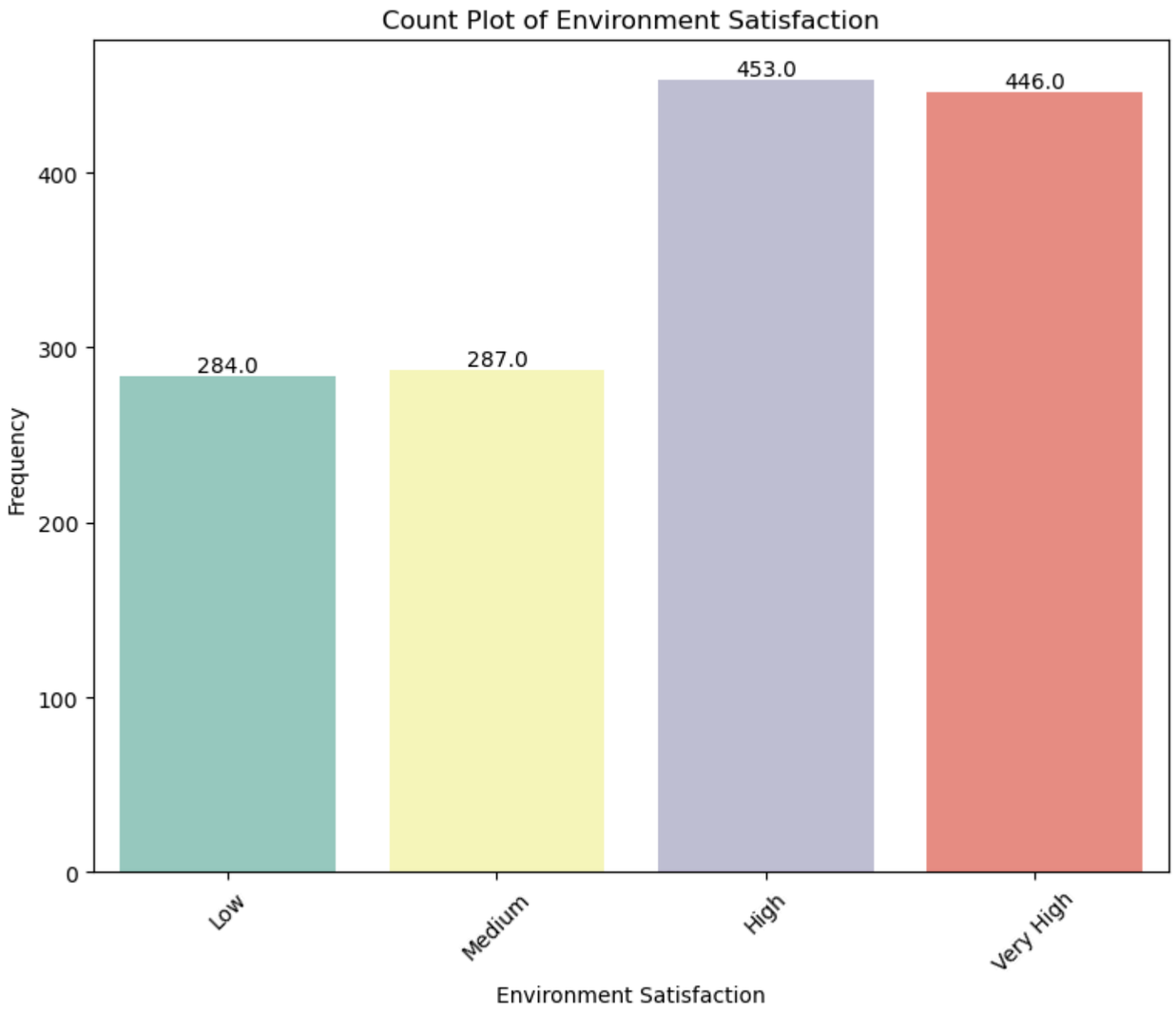
In [46]:
```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='EmployeeEngagementScore', bins=6, kde=True, color='pink')
plt.title('Histogram of Employee Engagement Score')
plt.xlabel('Employee Engagement Score')
plt.ylabel('Frequency')
plt.show()
```

In [47]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='EmployeeEngagementScore', color='red')
plt.title('Box Plot of Employee Engagement Score')
plt.xlabel('Employee Engagement Score')
plt.show()
```

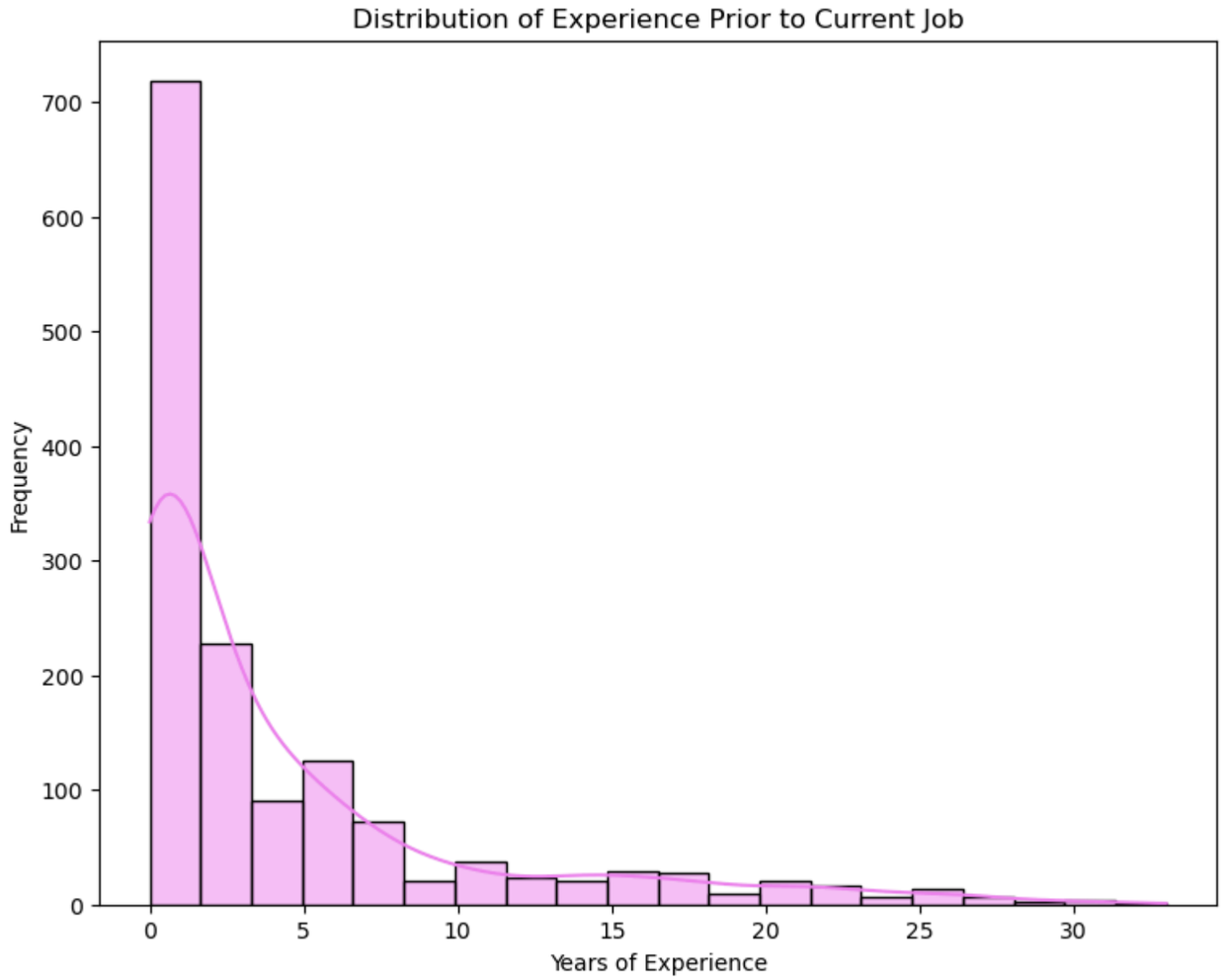


In [48]:
```
plt.figure(figsize=(9, 7))
sns.violinplot(data=df, x='EmployeeEngagementScore', color='brown')
plt.title('Violin Plot of Employee Engagement Score')
plt.xlabel('Employee Engagement Score')
plt.show()
```

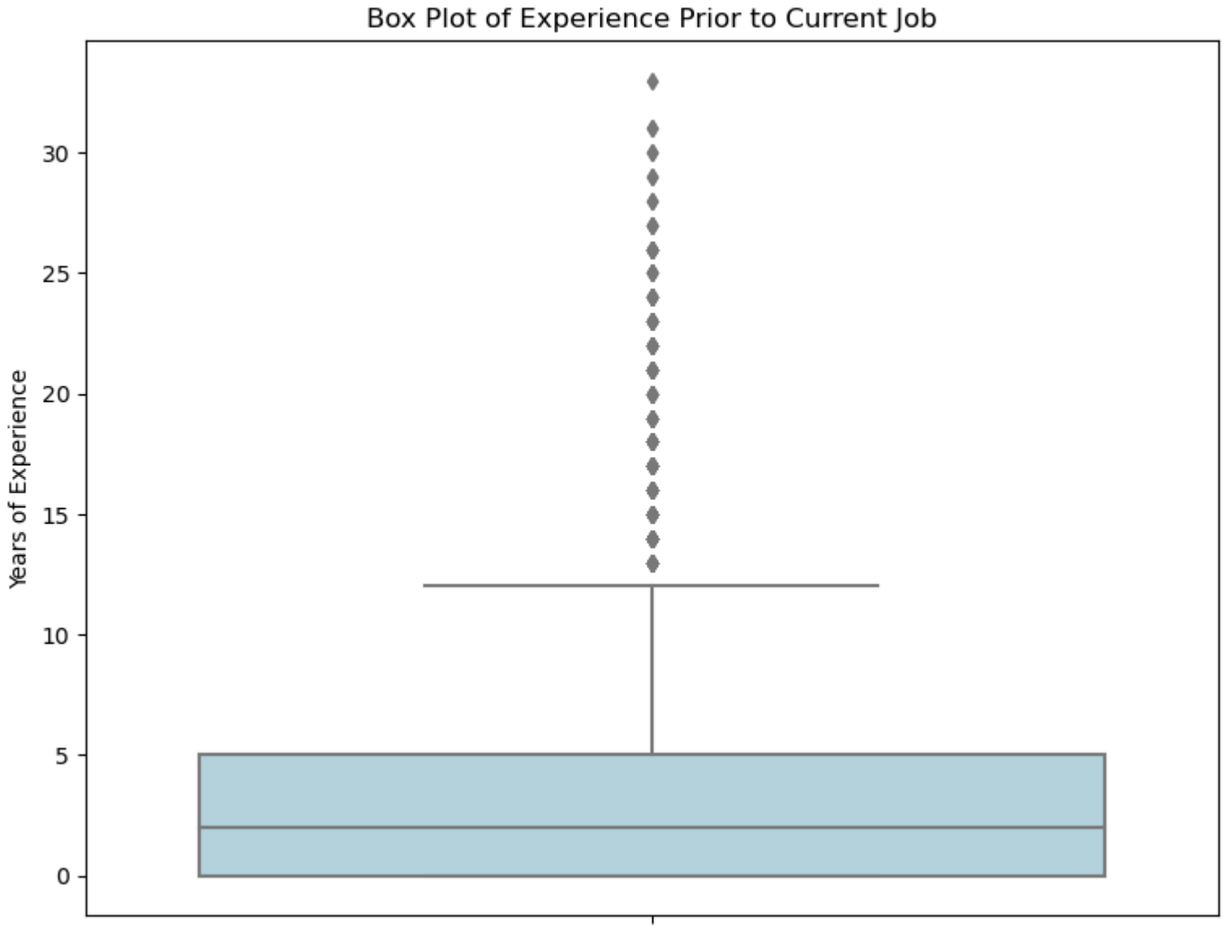
In [50]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='EnvironmentSatisfaction', palette='Set3', order=['Low', 'Medium', 'High', 'Very High'])
plt.title('Count Plot of Environment Satisfaction')
plt.xlabel('Environment Satisfaction')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()
```



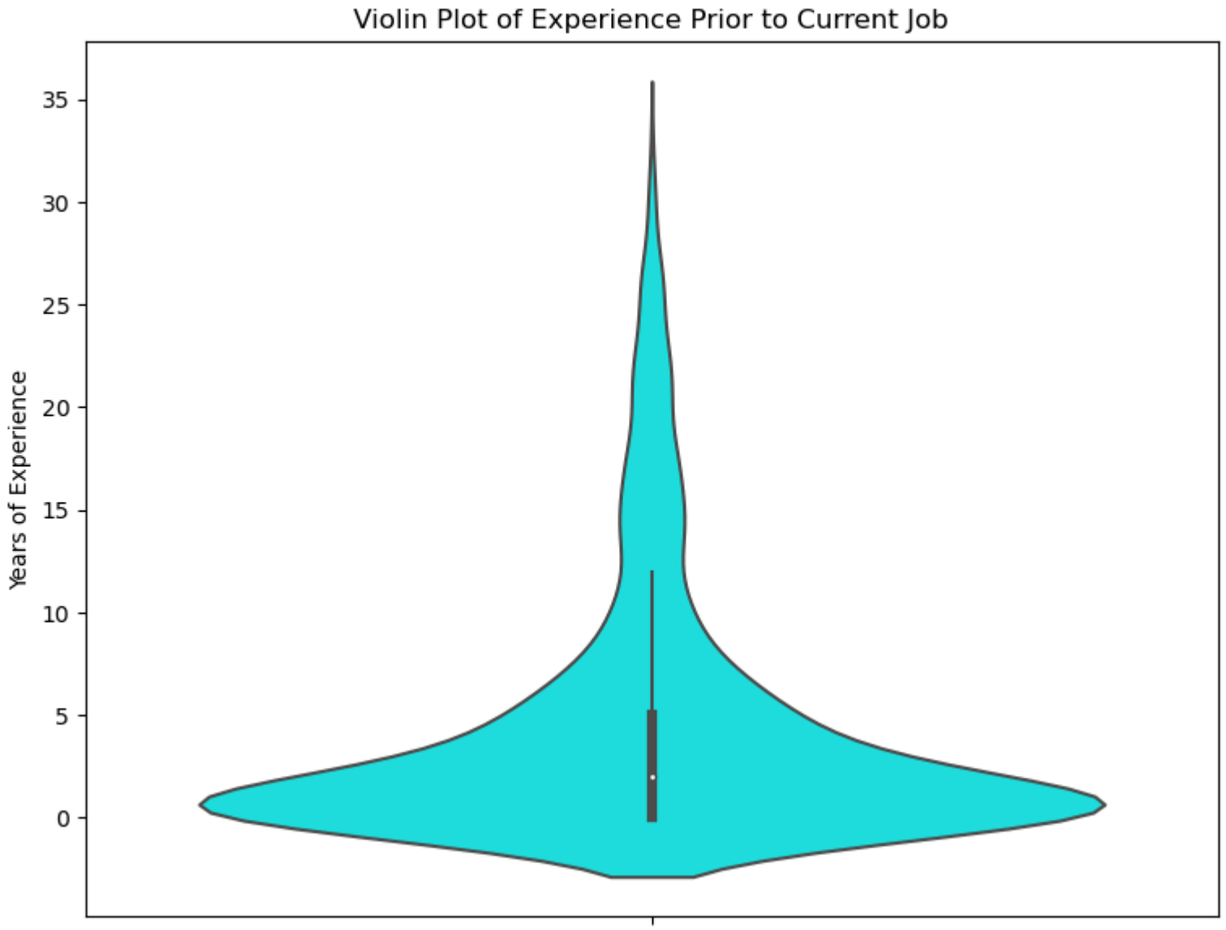
In [51]:
```
plt.figure(figsize=(9, 7))
sns.histplot(df['ExperiencePriorToCurrentJob'], bins=20, kde=True, color='violet')
plt.title('Distribution of Experience Prior to Current Job')
plt.xlabel('Years of Experience')
plt.ylabel('Frequency')
plt.show()
```



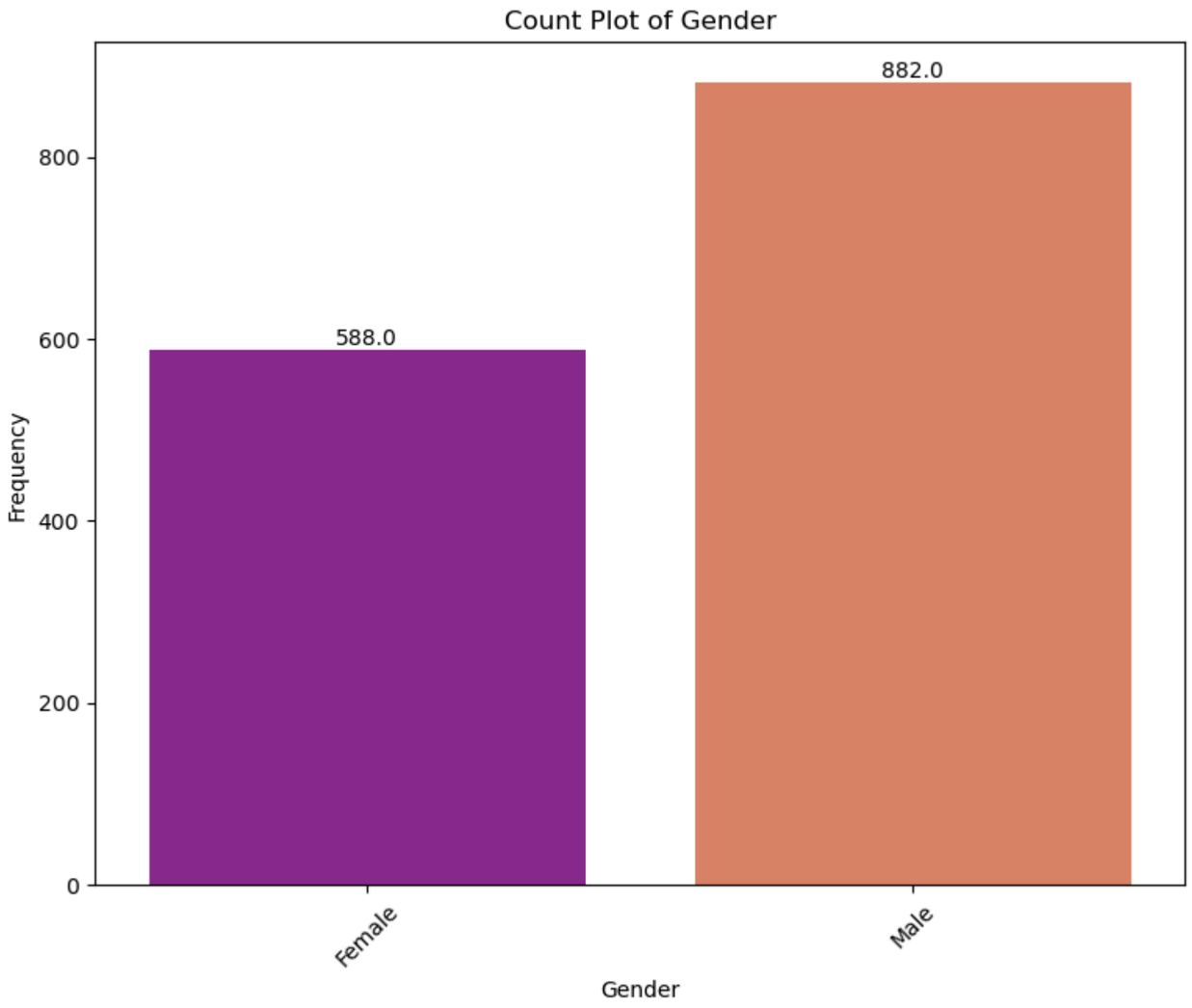
In [53]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, y='ExperiencePriorToCurrentJob', color='lightblue')
plt.title('Box Plot of Experience Prior to Current Job')
plt.ylabel('Years of Experience')
plt.show()
```



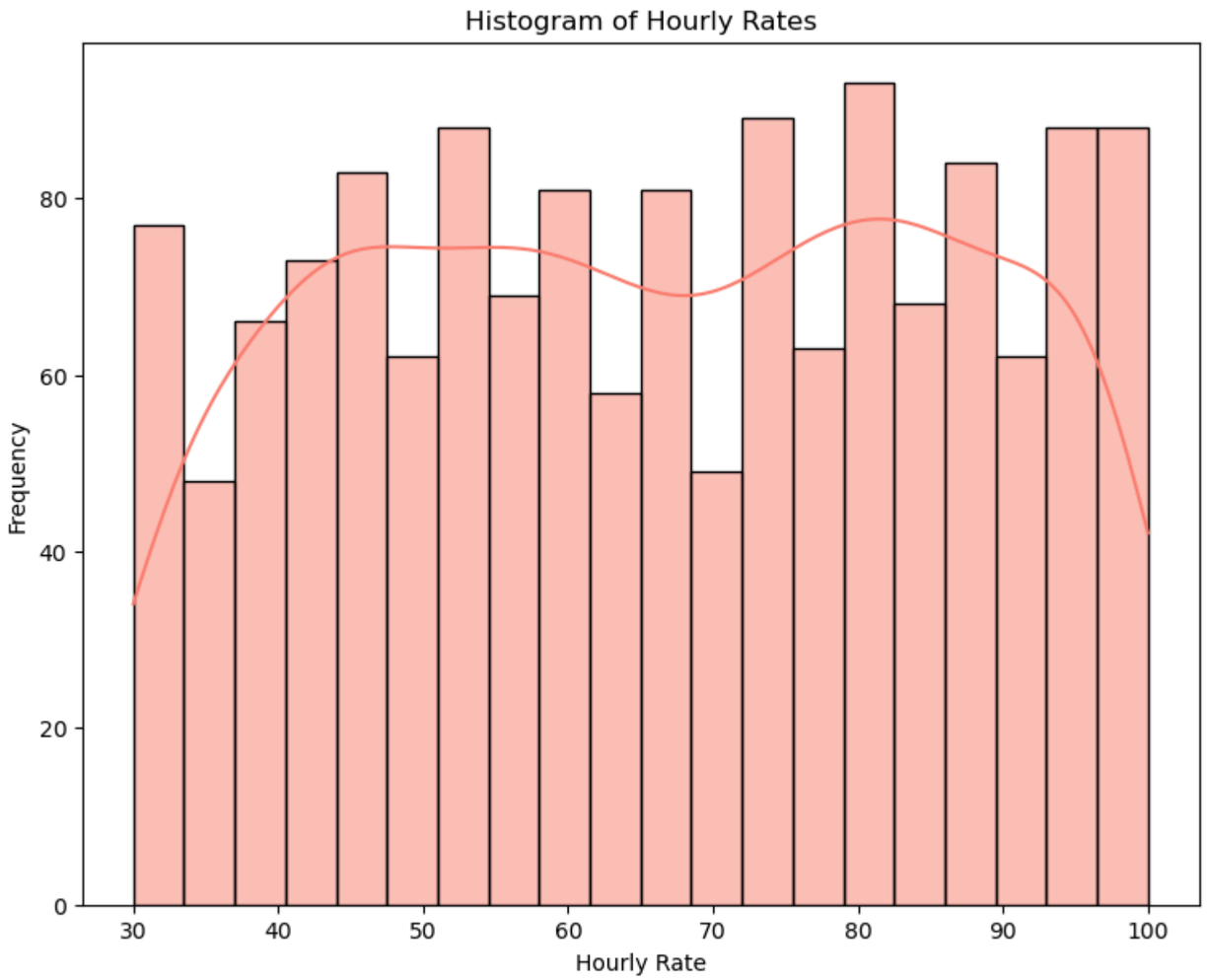
In [54]:
```
plt.figure(figsize=(9, 7))
sns.violinplot(data=df, y='ExperiencePriorToCurrentJob', color='cyan')
plt.title('Violin Plot of Experience Prior to Current Job')
plt.ylabel('Years of Experience')
plt.show()
```



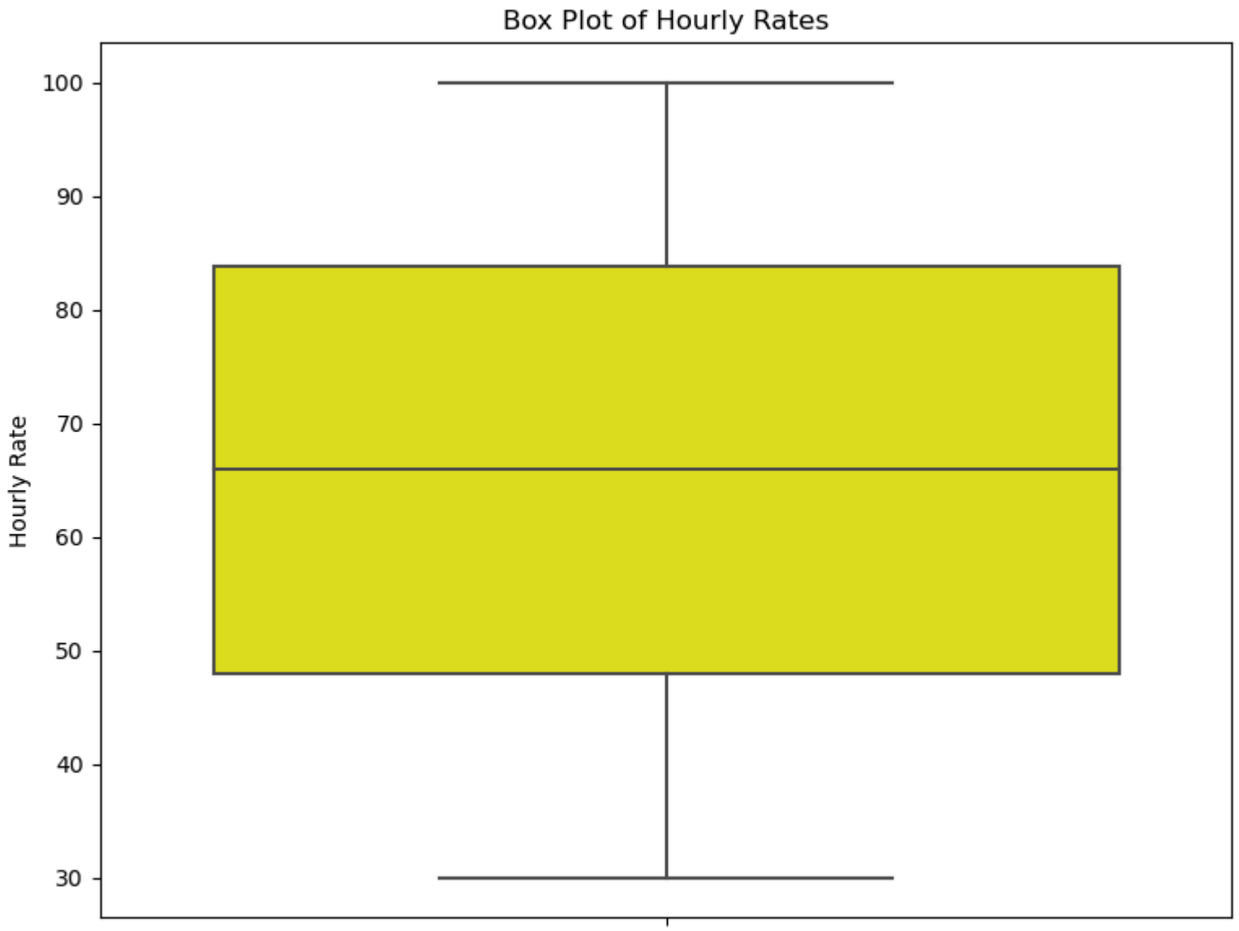
In [56]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='Gender', palette='plasma')
plt.title('Count Plot of Gender')
plt.xlabel('Gender')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()
```



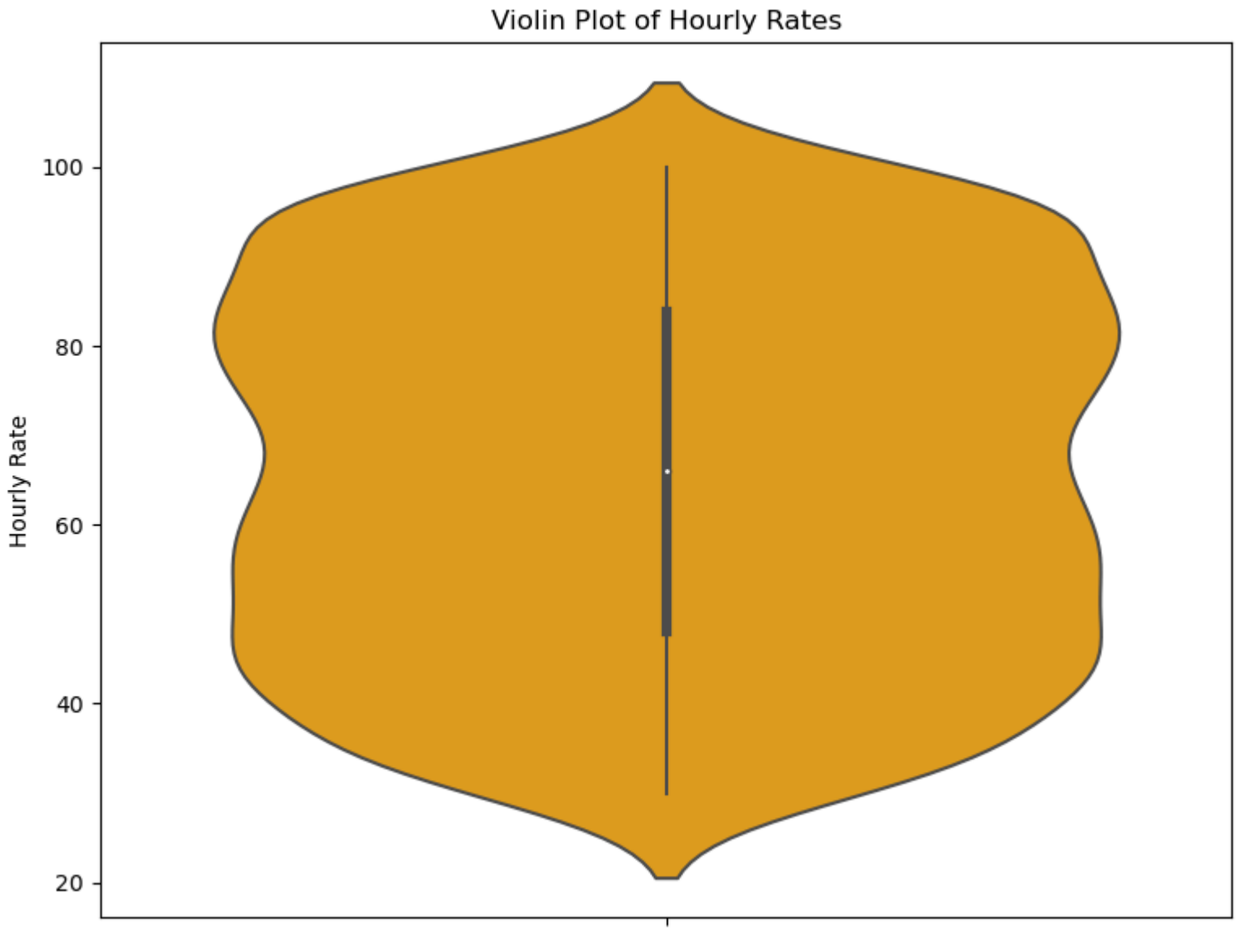
In [57]:
```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='HourlyRate', bins=20, kde=True, color='Salmon')
plt.title('Histogram of Hourly Rates')
plt.xlabel('Hourly Rate')
plt.ylabel('Frequency')
plt.show()
```



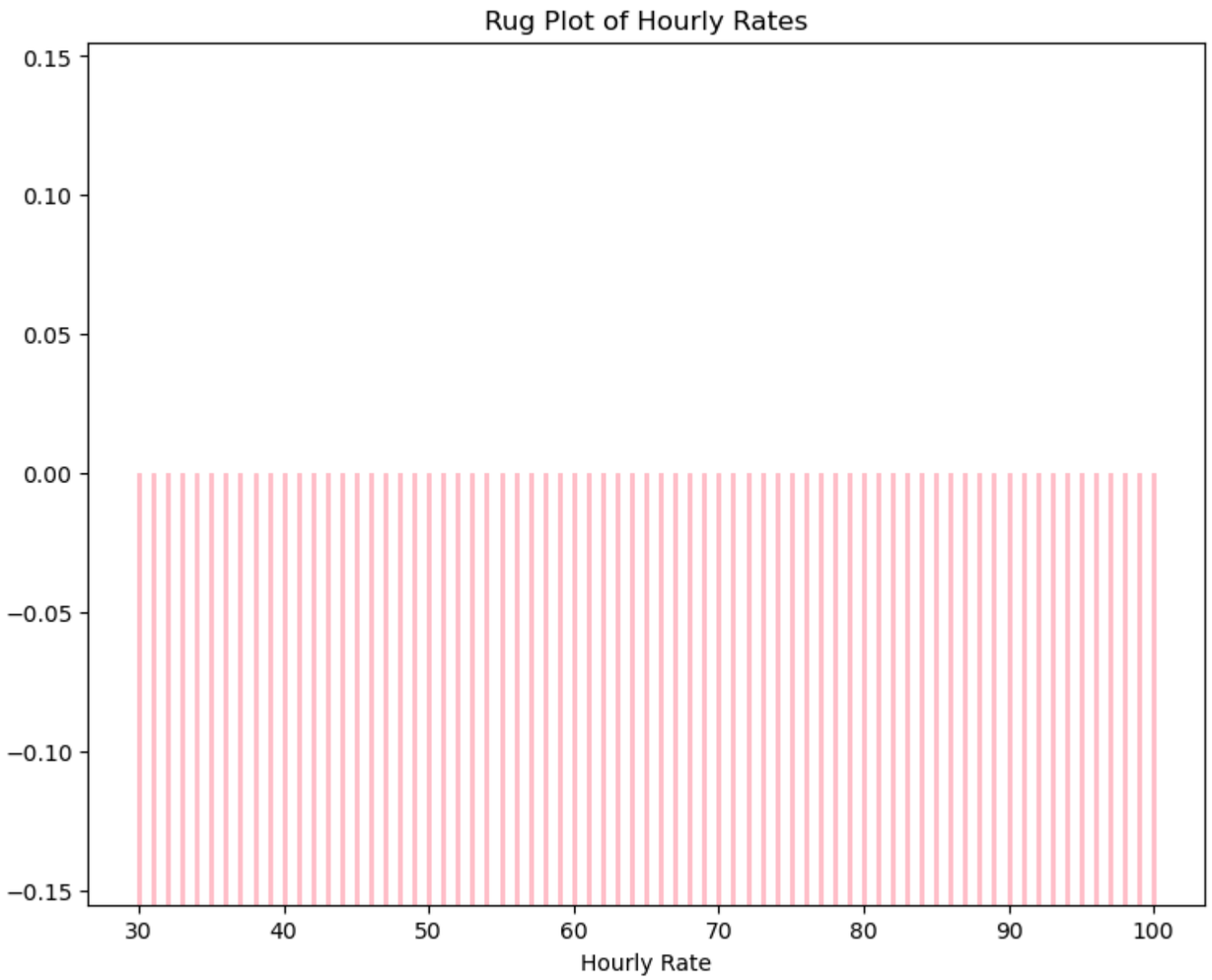
In [58]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, y='HourlyRate', color='yellow')
plt.title('Box Plot of Hourly Rates')
plt.ylabel('Hourly Rate')
plt.show()
```



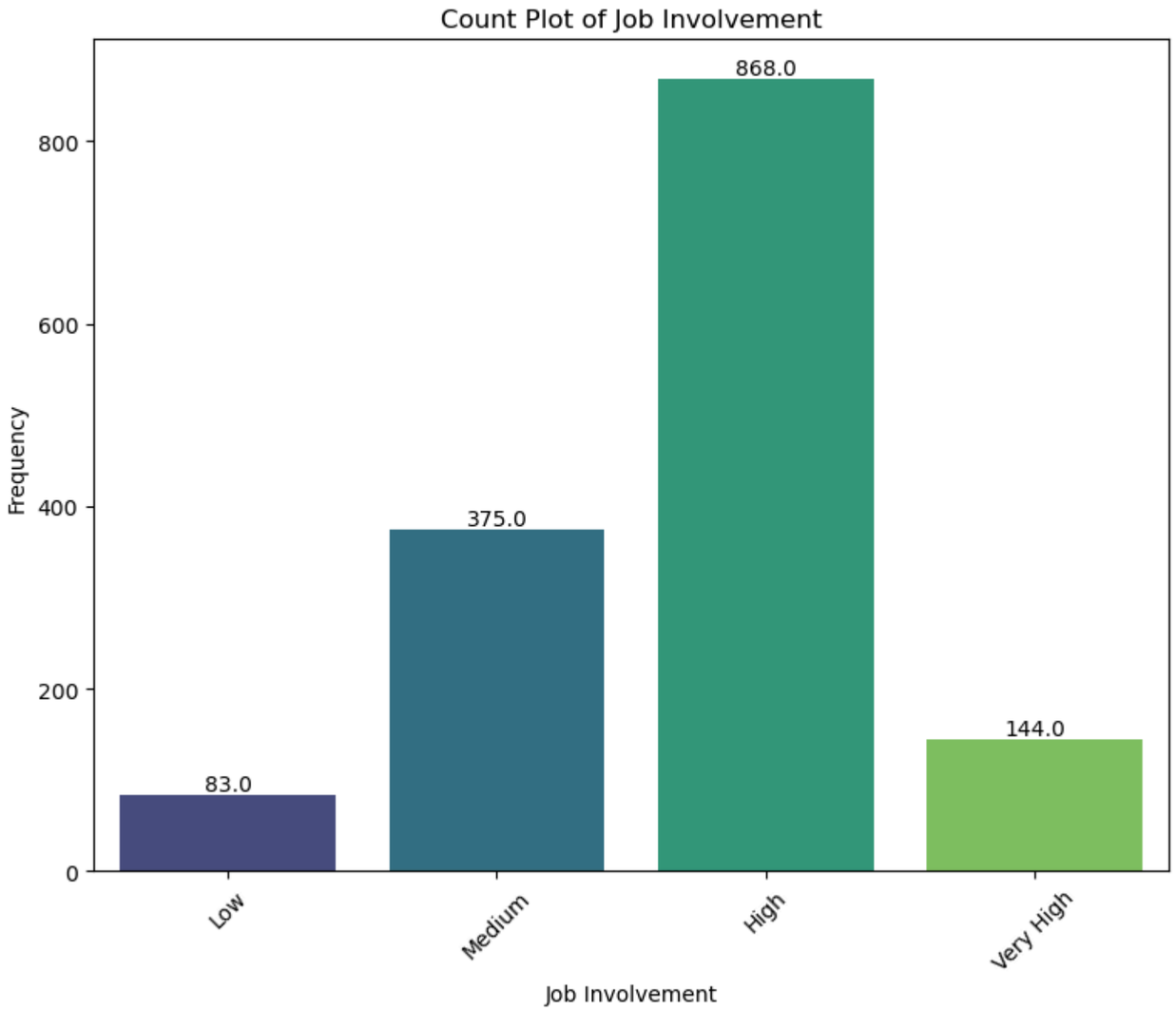
In [59]:
```
plt.figure(figsize=(9, 7))
sns.violinplot(data=df, y='HourlyRate', color='orange')
plt.title('Violin Plot of Hourly Rates')
plt.ylabel('Hourly Rate')
plt.show()
```



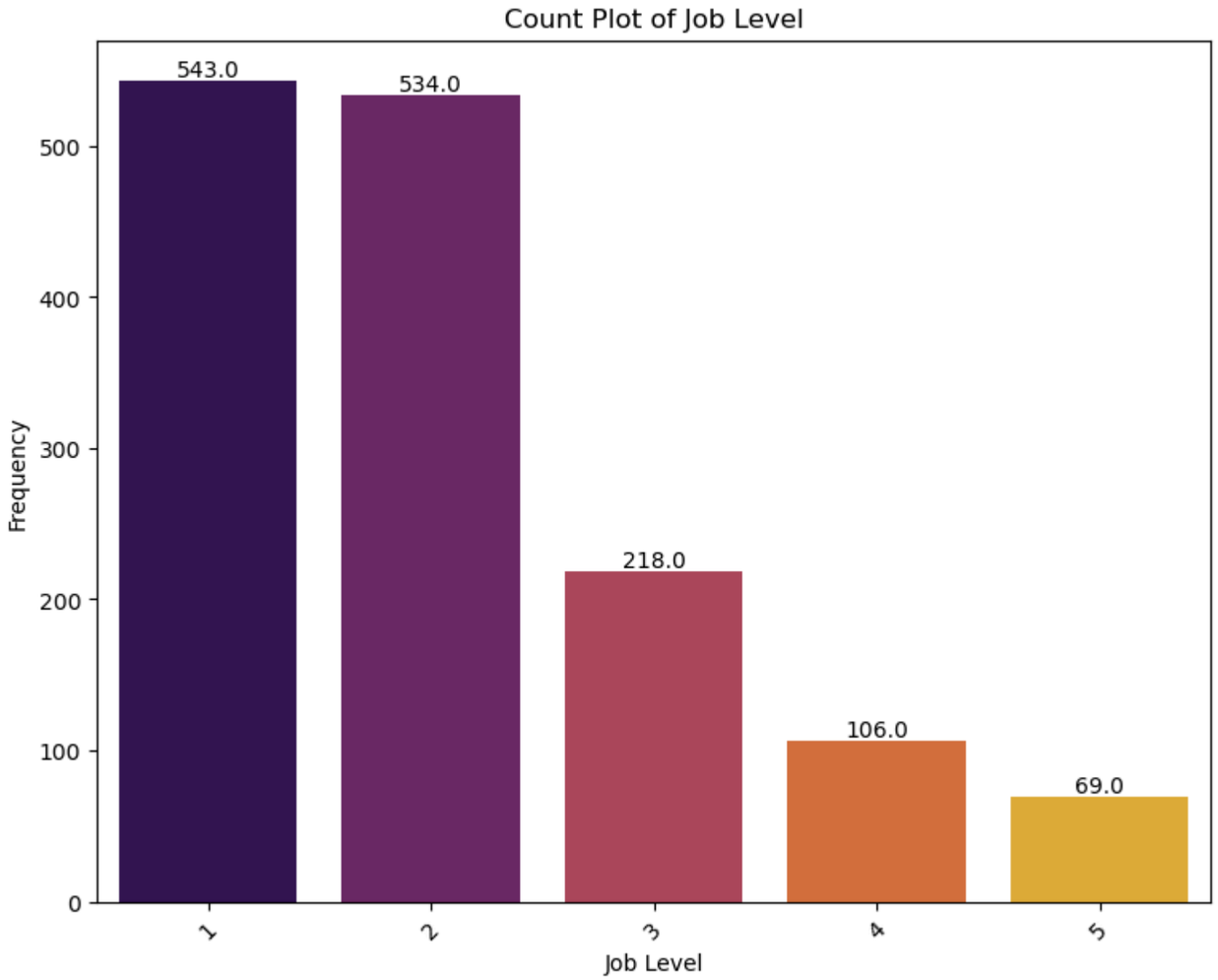
In [60]:
```
plt.figure(figsize=(9, 7))
sns.rugplot(data=df, x='HourlyRate', height=0.5, color='pink')
plt.title('Rug Plot of Hourly Rates')
plt.xlabel('Hourly Rate')
plt.show()
```



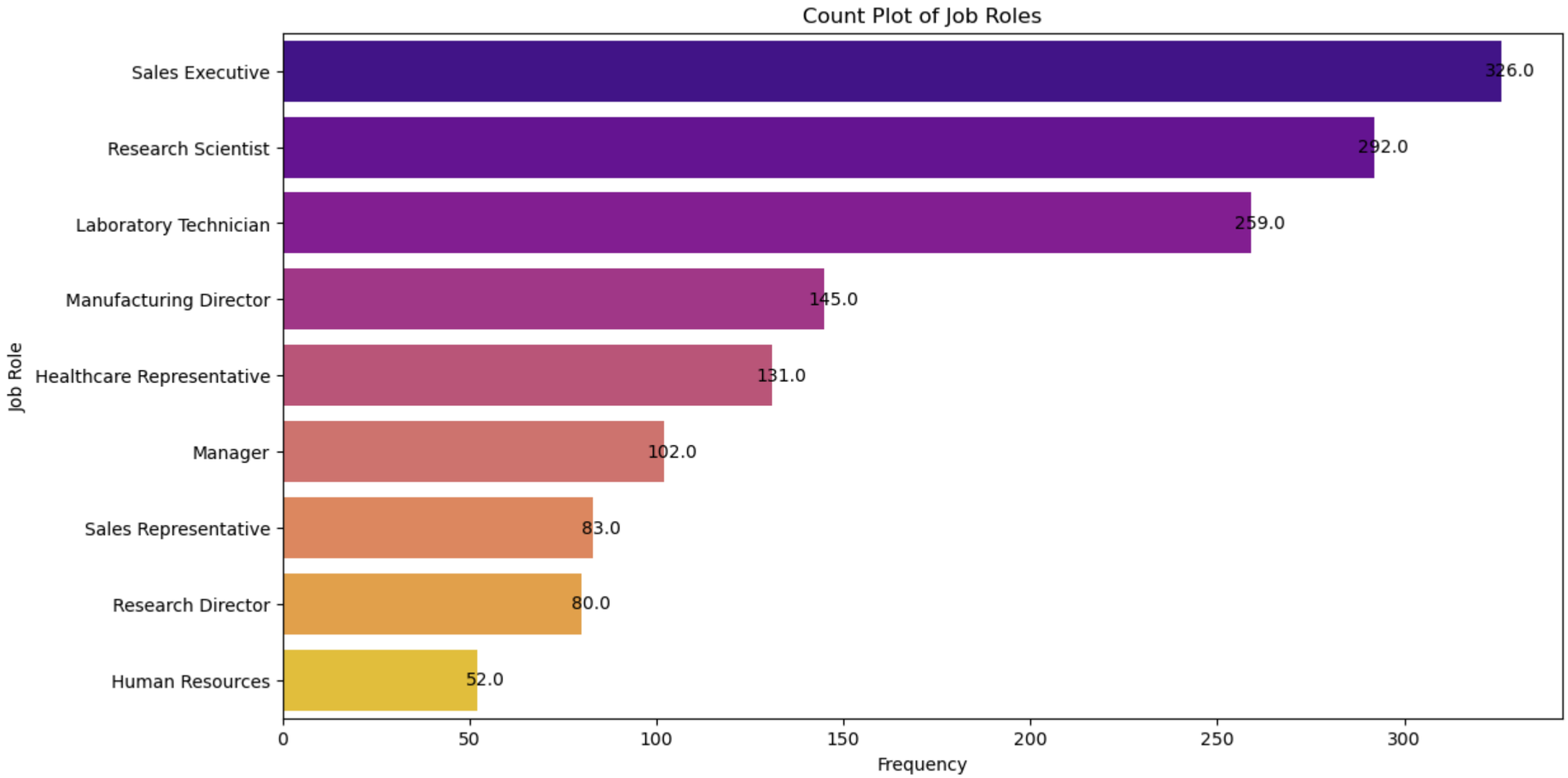
In [61]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='JobInvolvement', palette='viridis', order=['Low', 'Medium', 'High', 'Very High'])
plt.title('Count Plot of Job Involvement')
plt.xlabel('Job Involvement')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



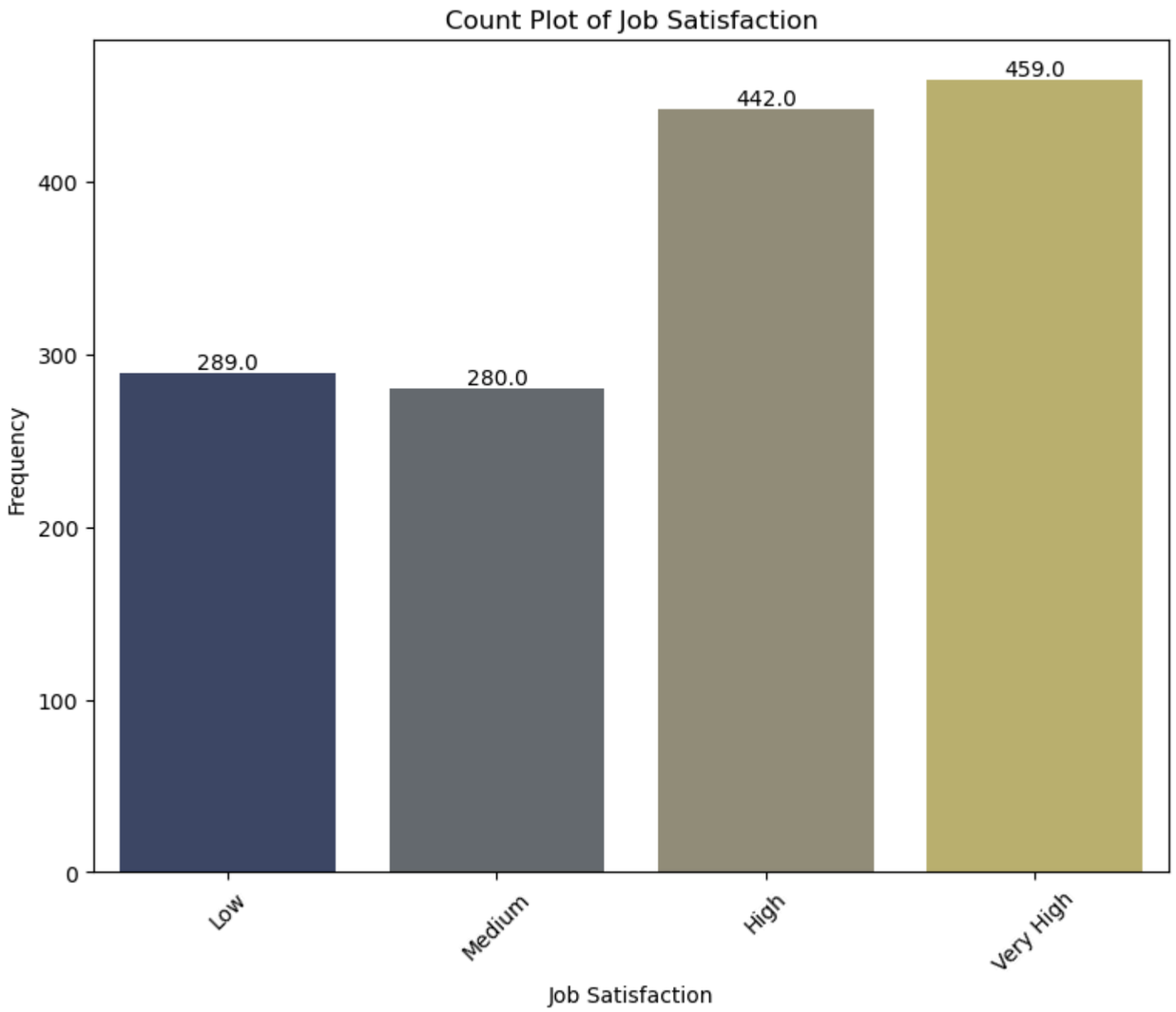
In [62]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='JobLevel', palette='inferno')
plt.title('Count Plot of Job Level')
plt.xlabel('Job Level')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



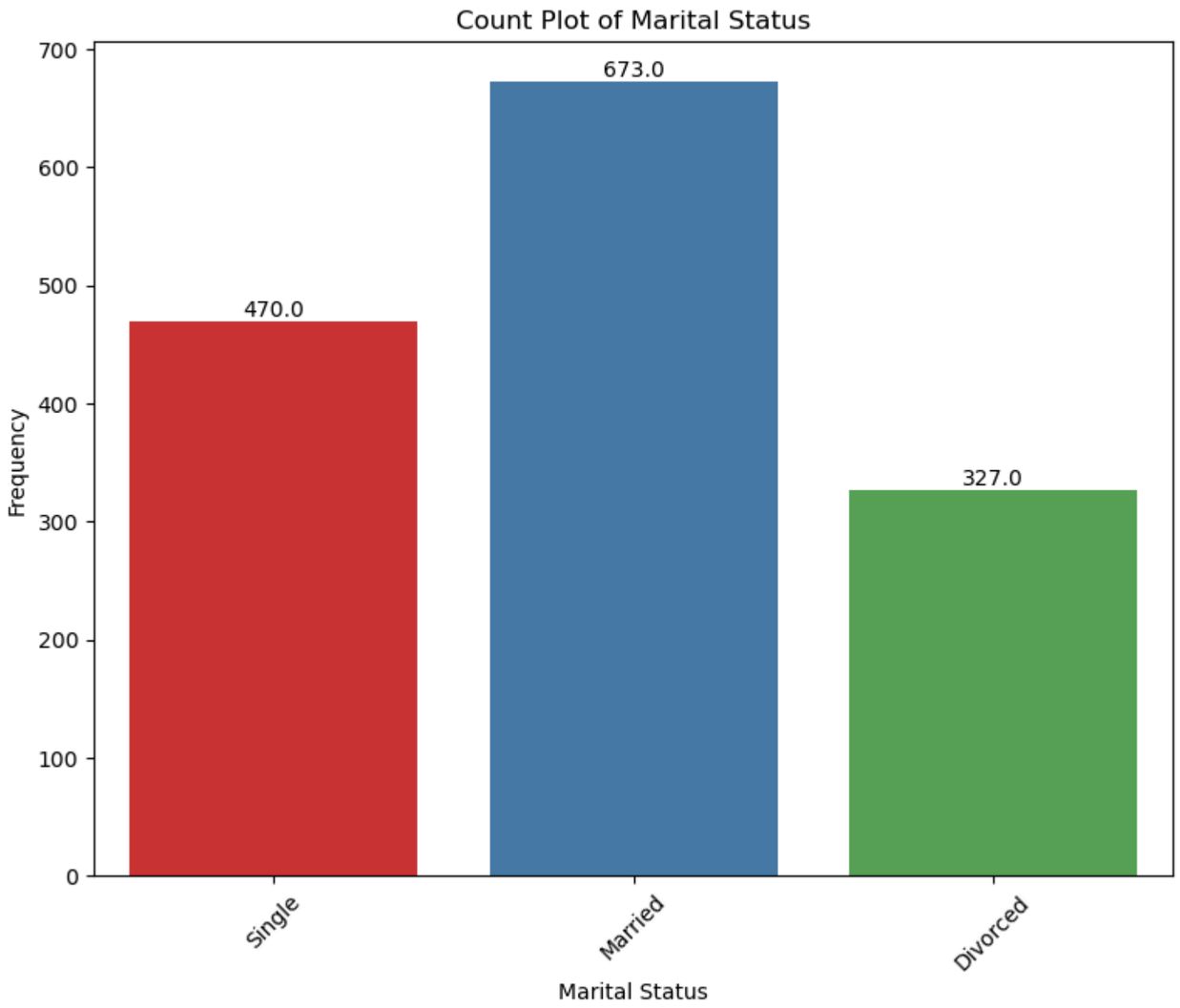
In [63]:
```
plt.figure(figsize=(13, 7))
ax = sns.countplot(data=df, y='JobRole', palette='plasma')
plt.title('Count Plot of Job Roles')
plt.xlabel('Frequency')
plt.ylabel('Job Role')

for p in ax.patches:
    ax.annotate(f'{p.get_width()}', (p.get_width(), p.get_y() + p.get_height() / 2.),
                ha='center', va='center', fontsize=10, color='black', xytext=(5, 0),
                textcoords='offset points')

plt.show()
```



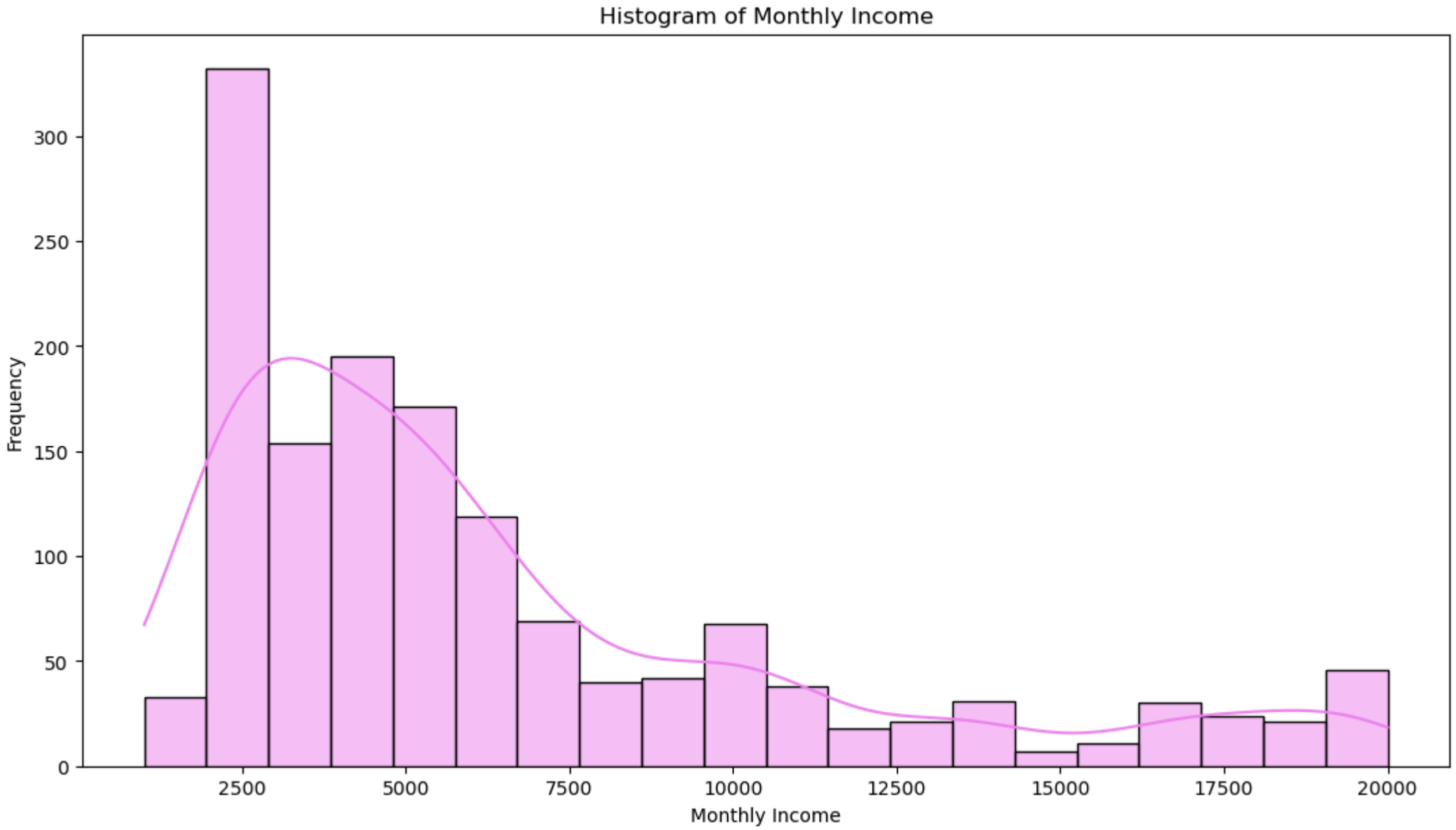
In [64]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='JobSatisfaction', palette='cividis', order=['Low', 'Medium', 'High', 'Very High'])
plt.title('Count Plot of Job Satisfaction')
plt.xlabel('Job Satisfaction')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



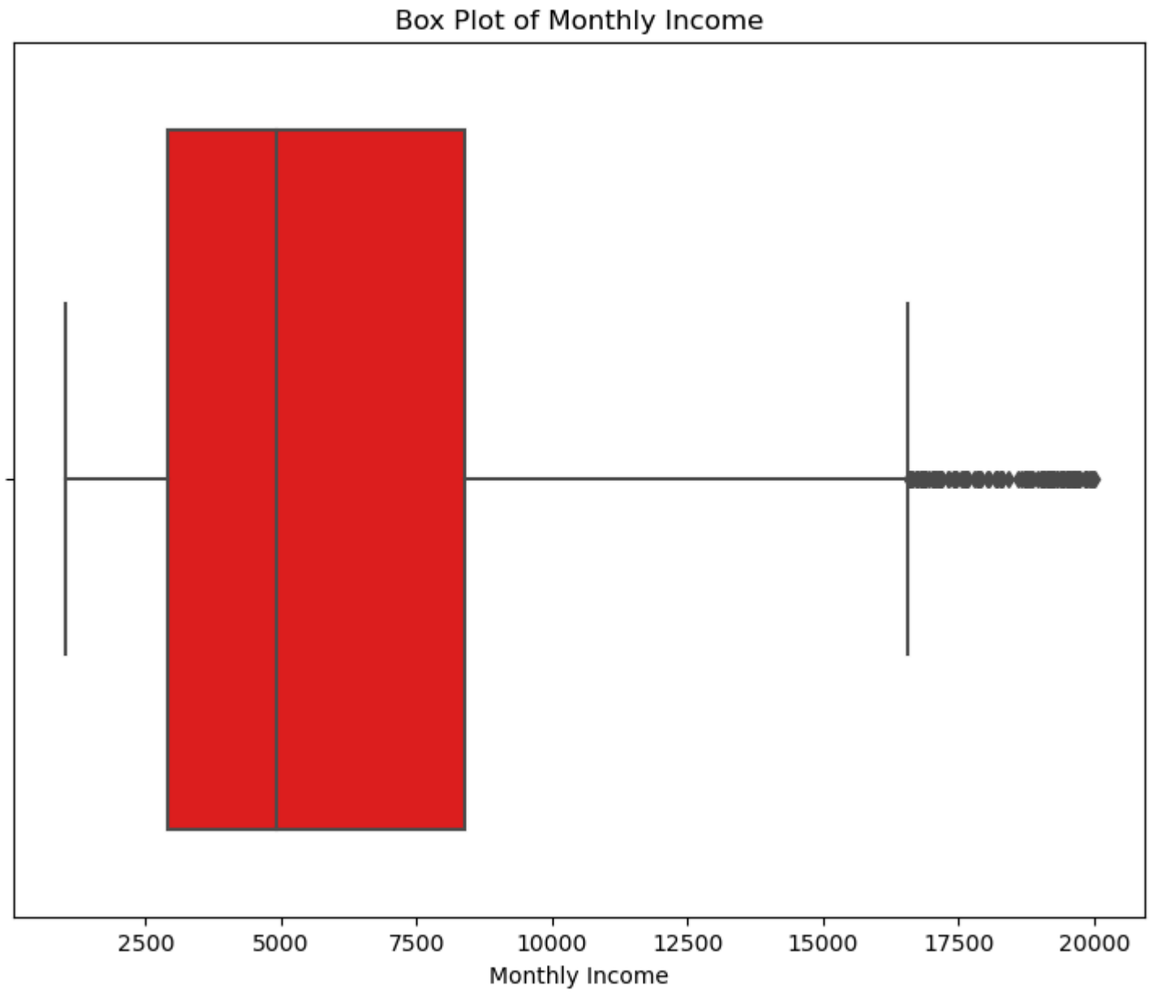
In [65]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='MaritalStatus', palette='Set1')
plt.title('Count Plot of Marital Status')
plt.xlabel('Marital Status')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



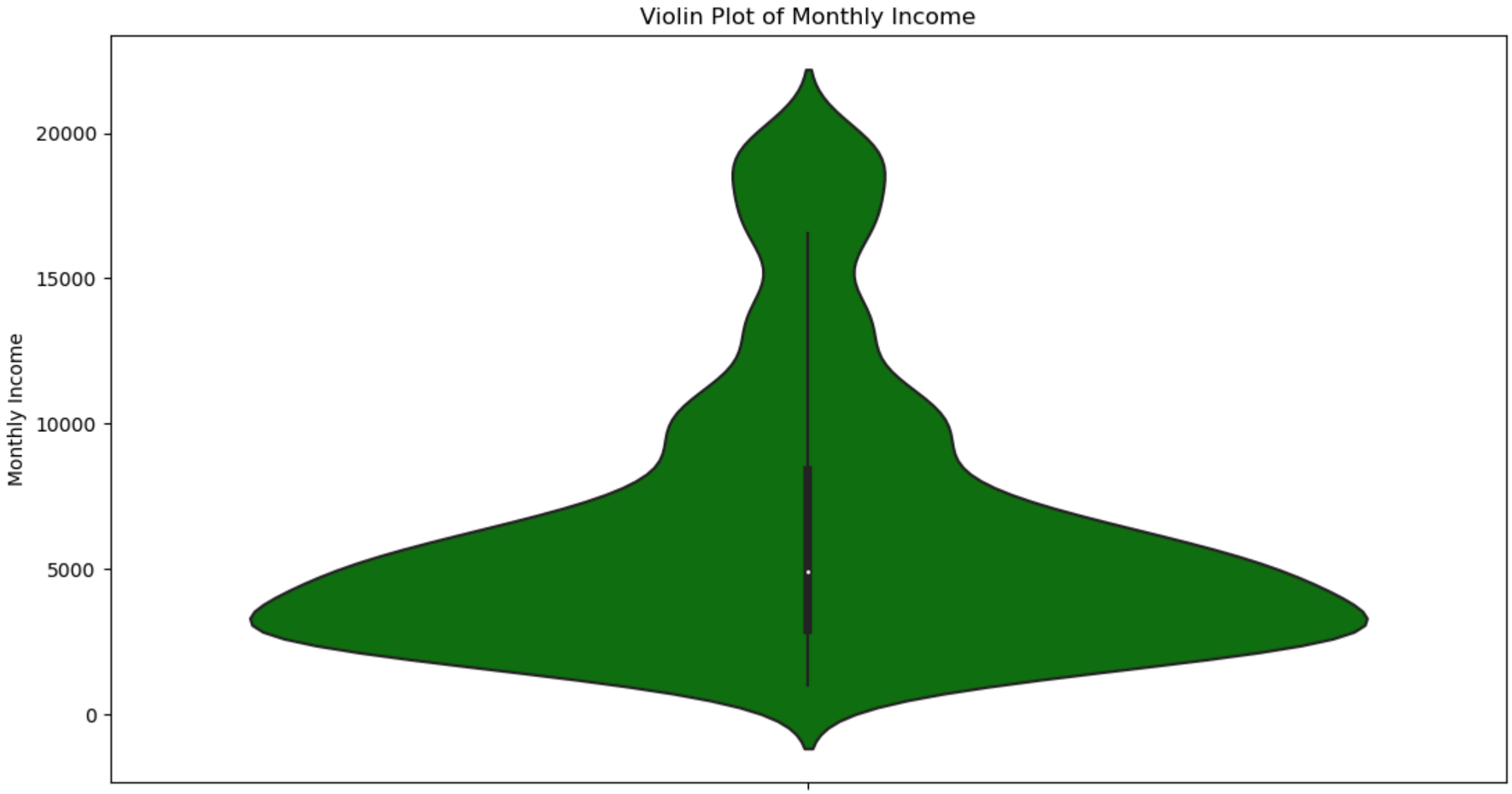
In [66]:
```
plt.figure(figsize=(13, 7))
sns.histplot(df['MonthlyIncome'], bins=20, kde=True, color='violet')
plt.title('Histogram of Monthly Income')
plt.xlabel('Monthly Income')
plt.ylabel('Frequency')
plt.show()
```



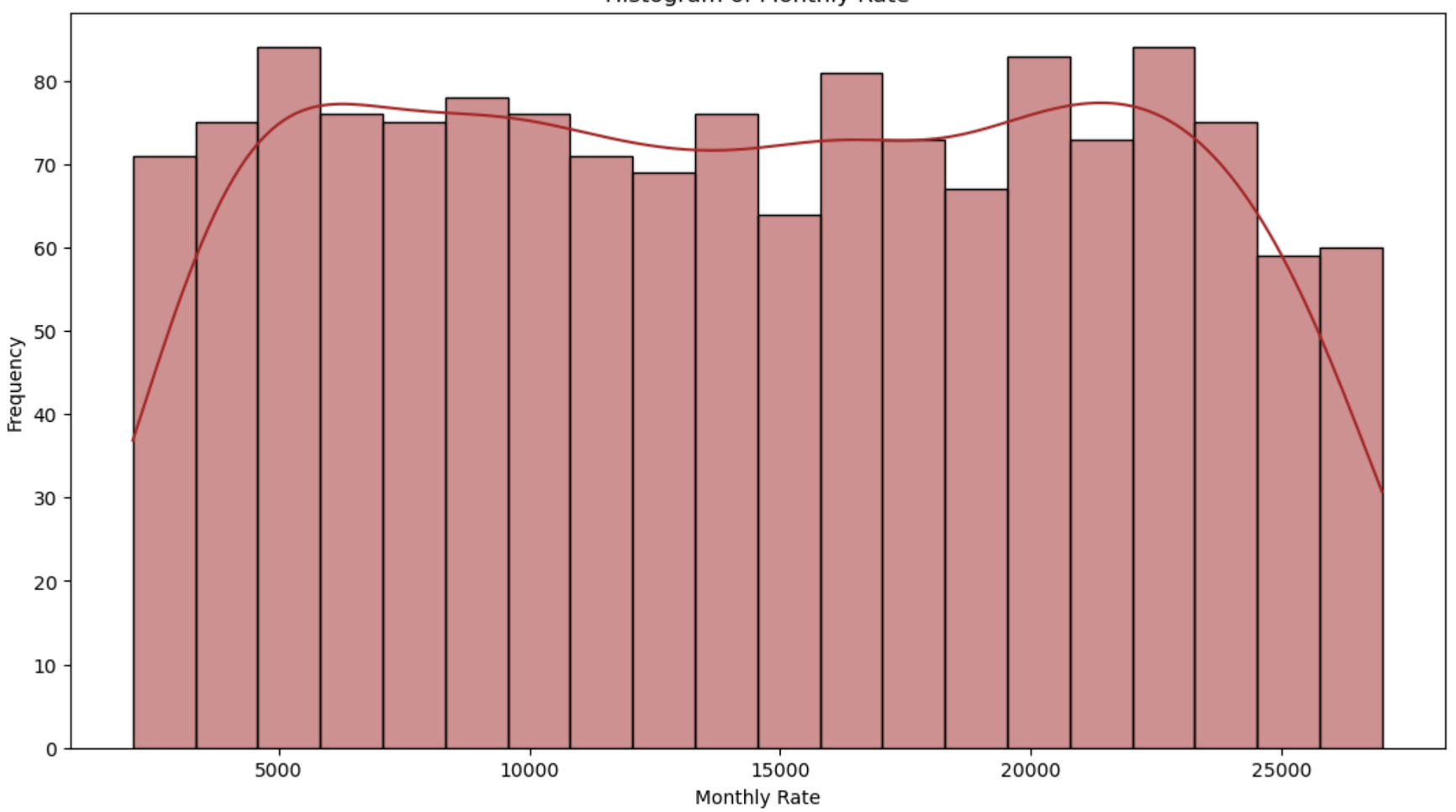
In [67]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='MonthlyIncome', color='red')
plt.title('Box Plot of Monthly Income')
plt.xlabel('Monthly Income')
plt.show()
```



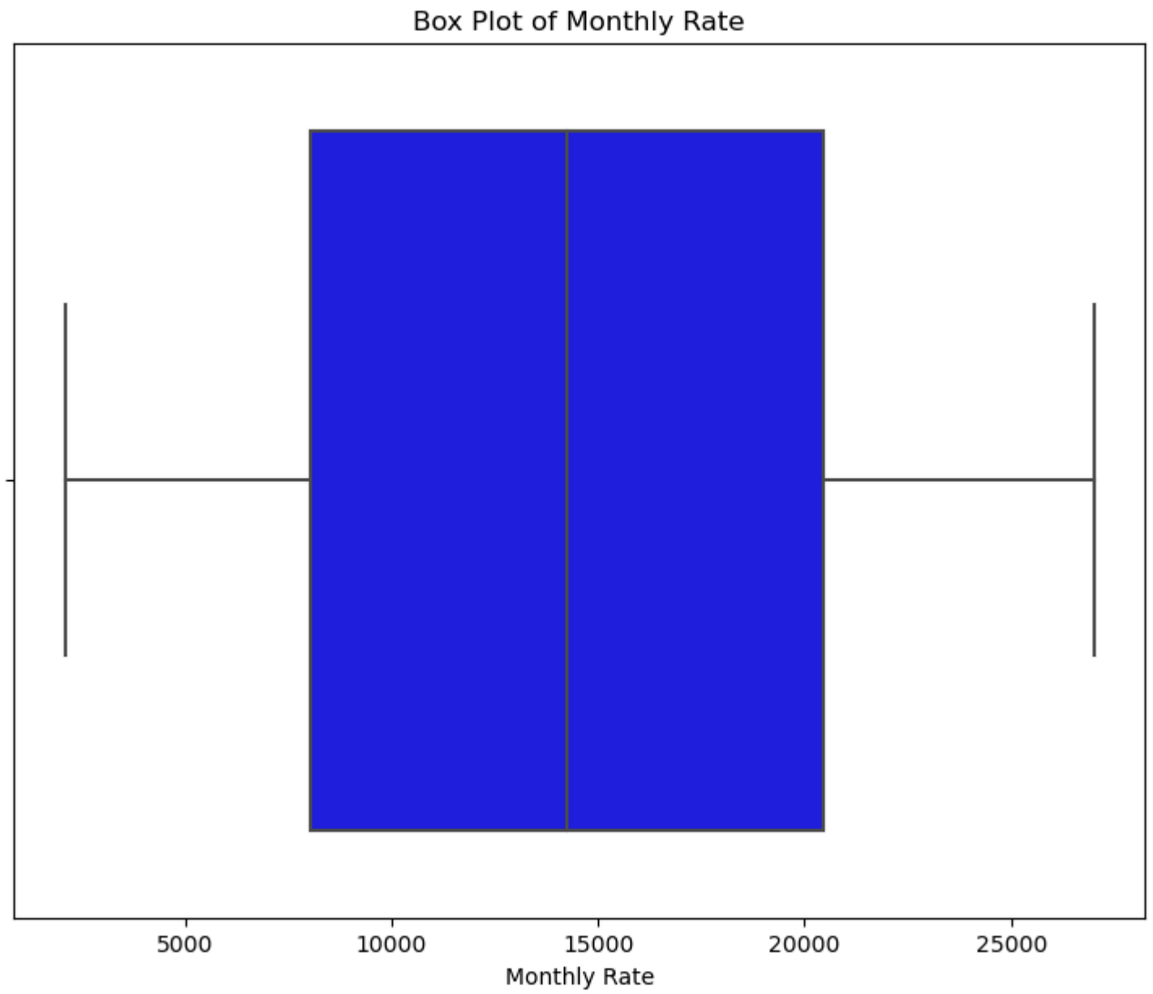
In [68]:
```
plt.figure(figsize=(13, 7))
sns.violinplot(data=df, y='MonthlyIncome', color='green')
plt.title('Violin Plot of Monthly Income')
plt.ylabel('Monthly Income')
plt.show()
```



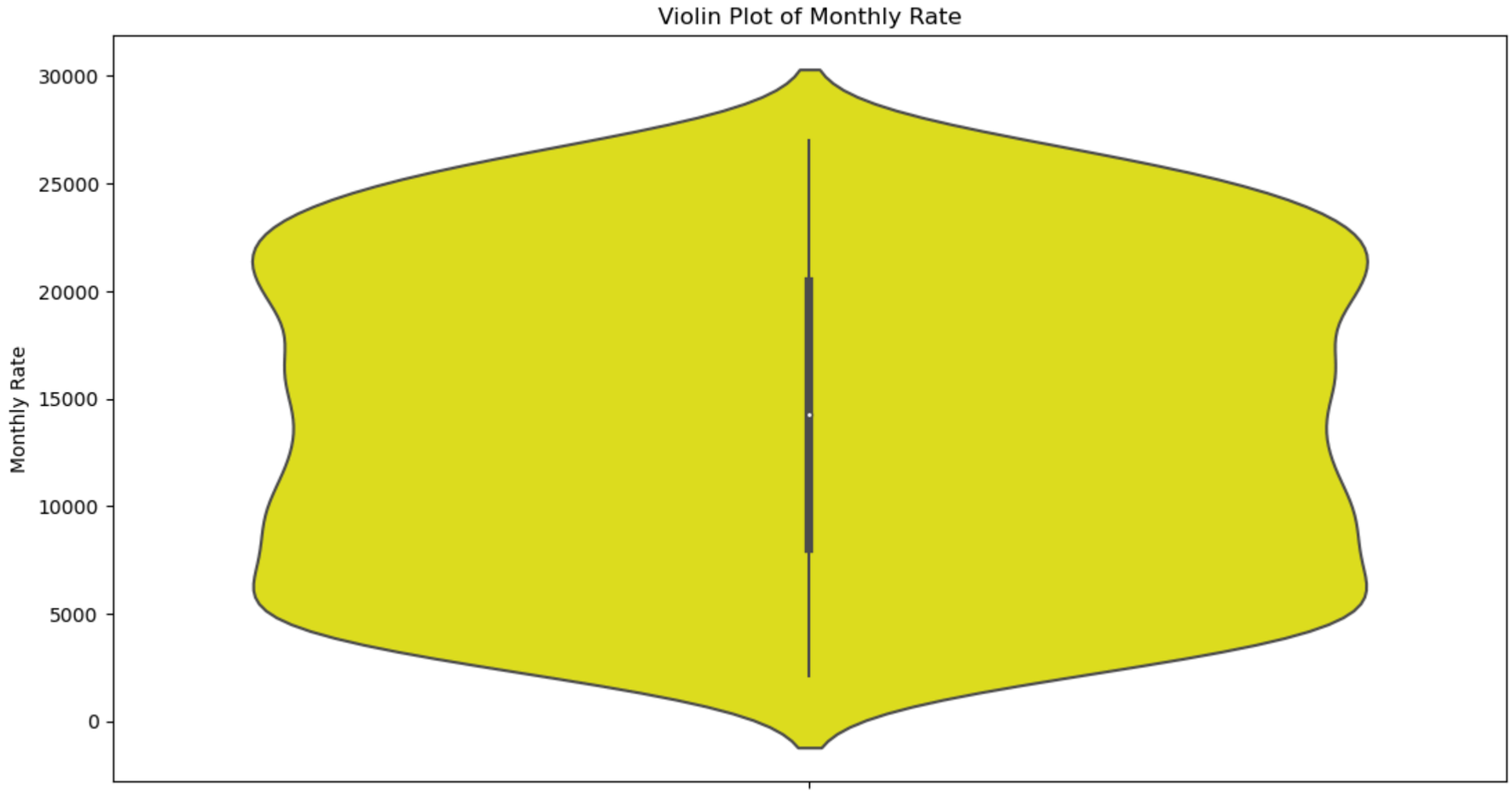
In [69]:
```
plt.figure(figsize=(13, 7))
sns.histplot(df['MonthlyRate'], bins=20, kde=True, color='brown')
plt.title('Histogram of Monthly Rate')
plt.xlabel('Monthly Rate')
plt.ylabel('Frequency')
plt.show()
```



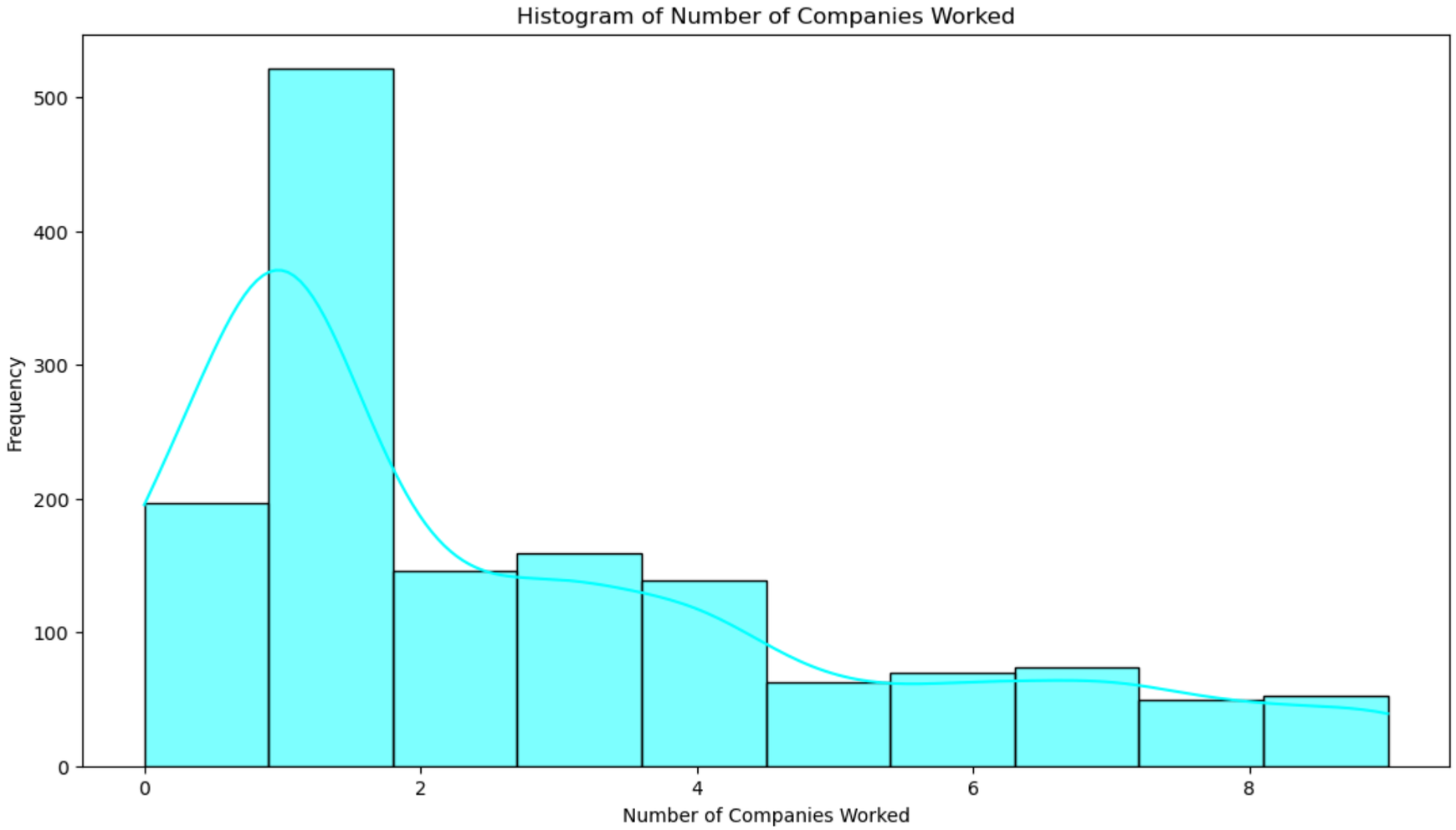
In [70]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='MonthlyRate', color='blue')
plt.title('Box Plot of Monthly Rate')
plt.xlabel('Monthly Rate')
plt.show()
```



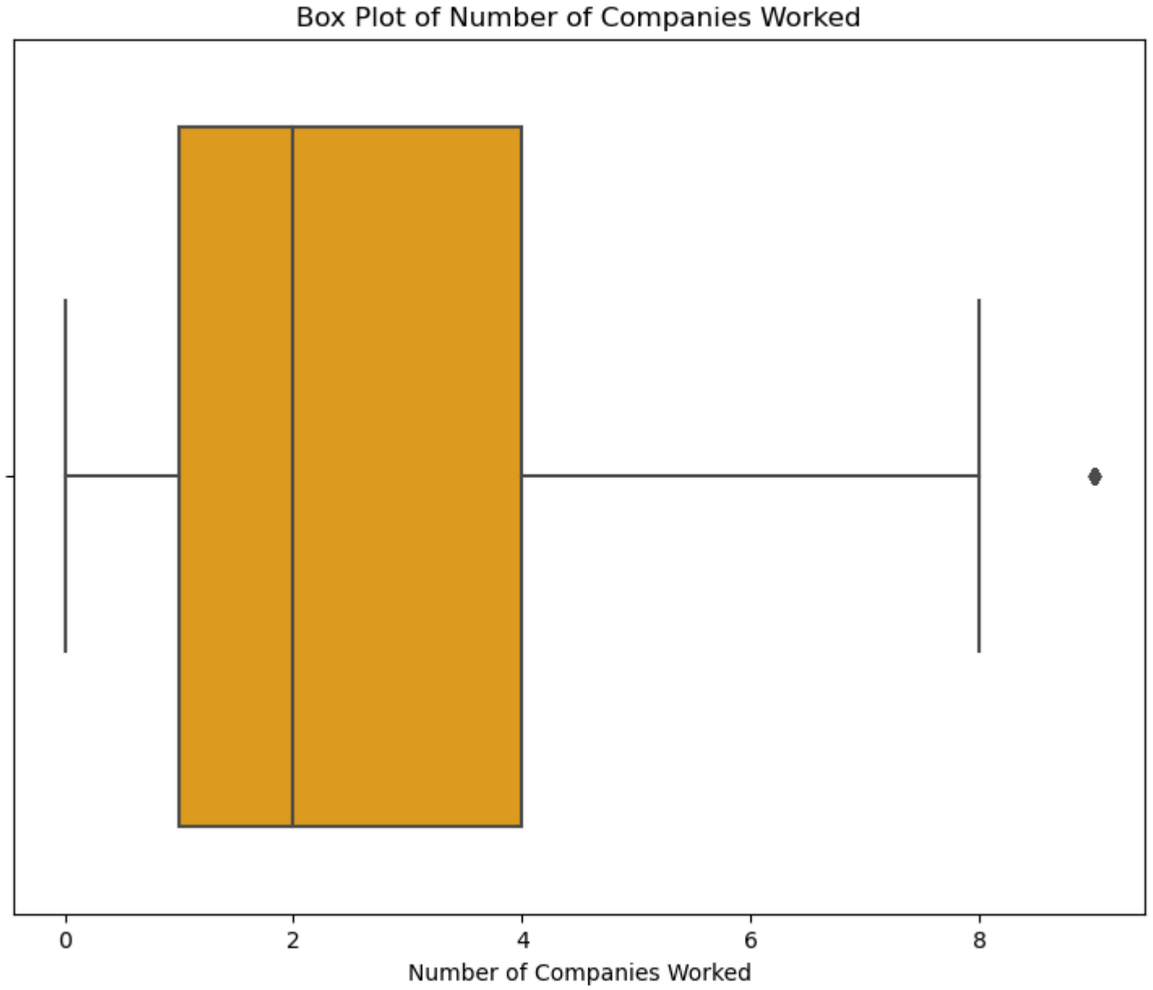
In [72]:
```
plt.figure(figsize=(13, 7))
sns.violinplot(data=df, y='MonthlyRate', color='yellow')
plt.title('Violin Plot of Monthly Rate')
plt.ylabel('Monthly Rate')
plt.show()
```



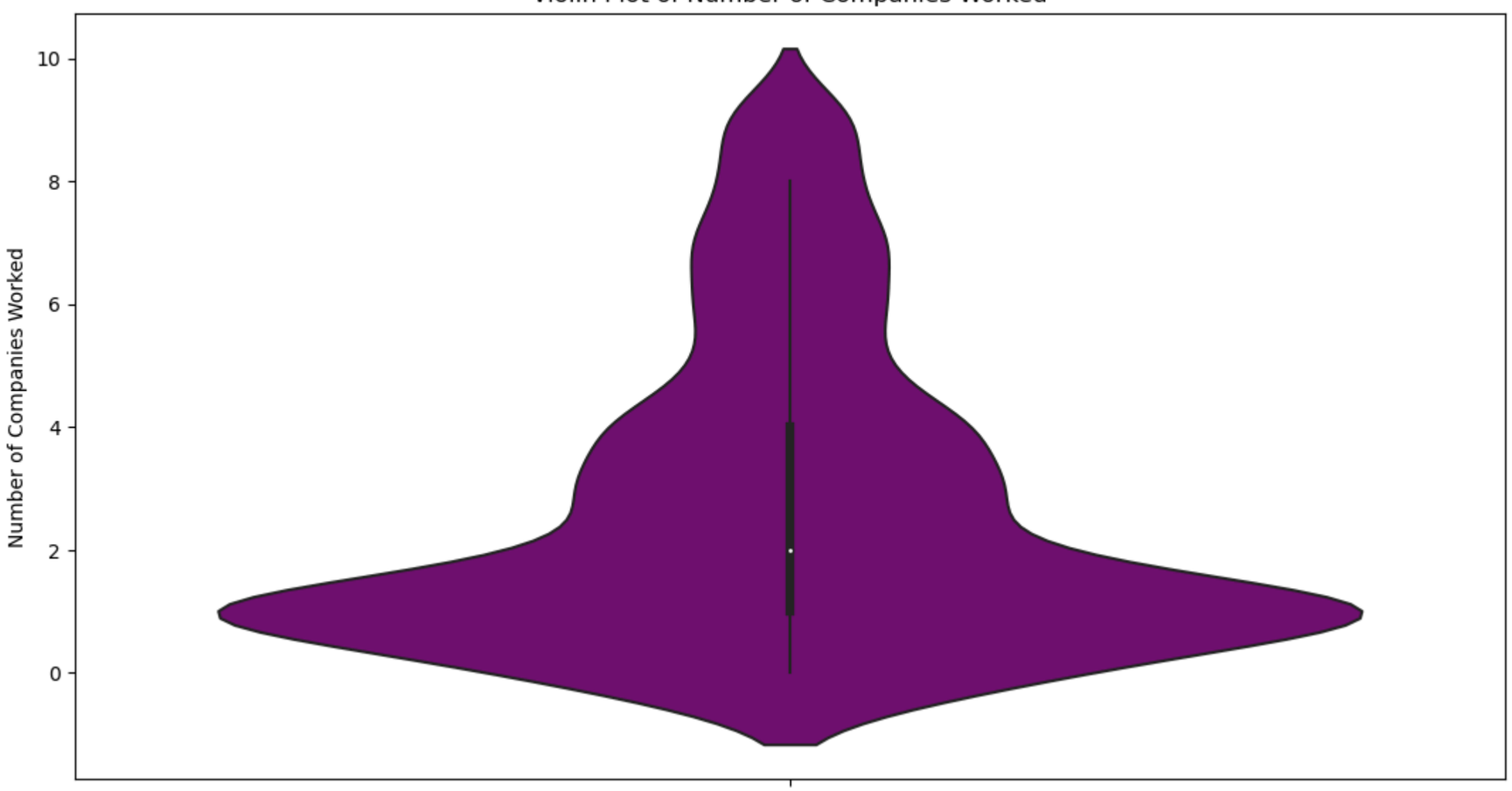
In [73]:
```
plt.figure(figsize=(13, 7))
sns.histplot(df['NumCompaniesWorked'], bins=10, kde=True, color='cyan')
plt.title('Histogram of Number of Companies Worked')
plt.xlabel('Number of Companies Worked')
plt.ylabel('Frequency')
plt.show()
```



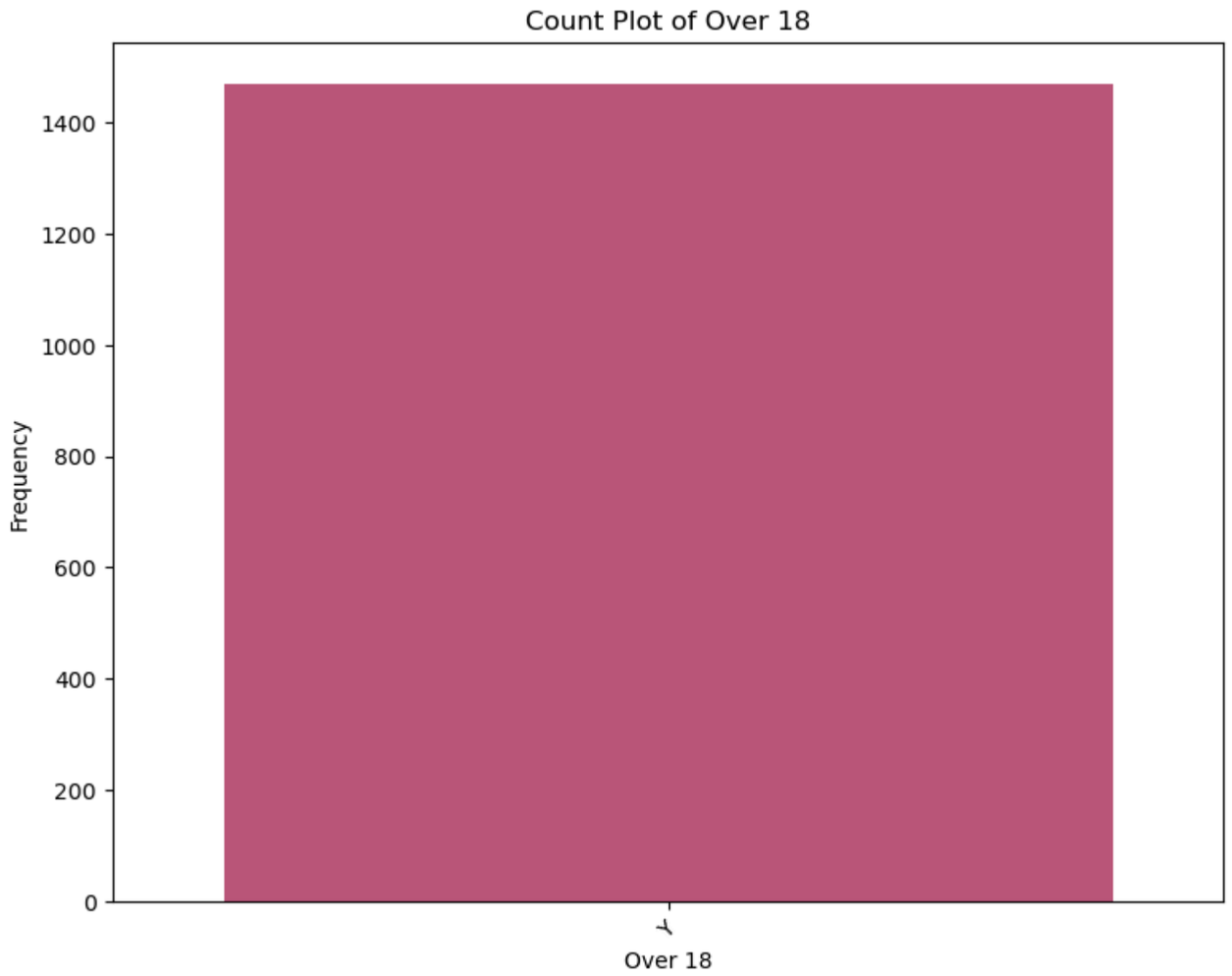
In [74]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='NumCompaniesWorked', color='orange')
plt.title('Box Plot of Number of Companies Worked')
plt.xlabel('Number of Companies Worked')
plt.show()
```



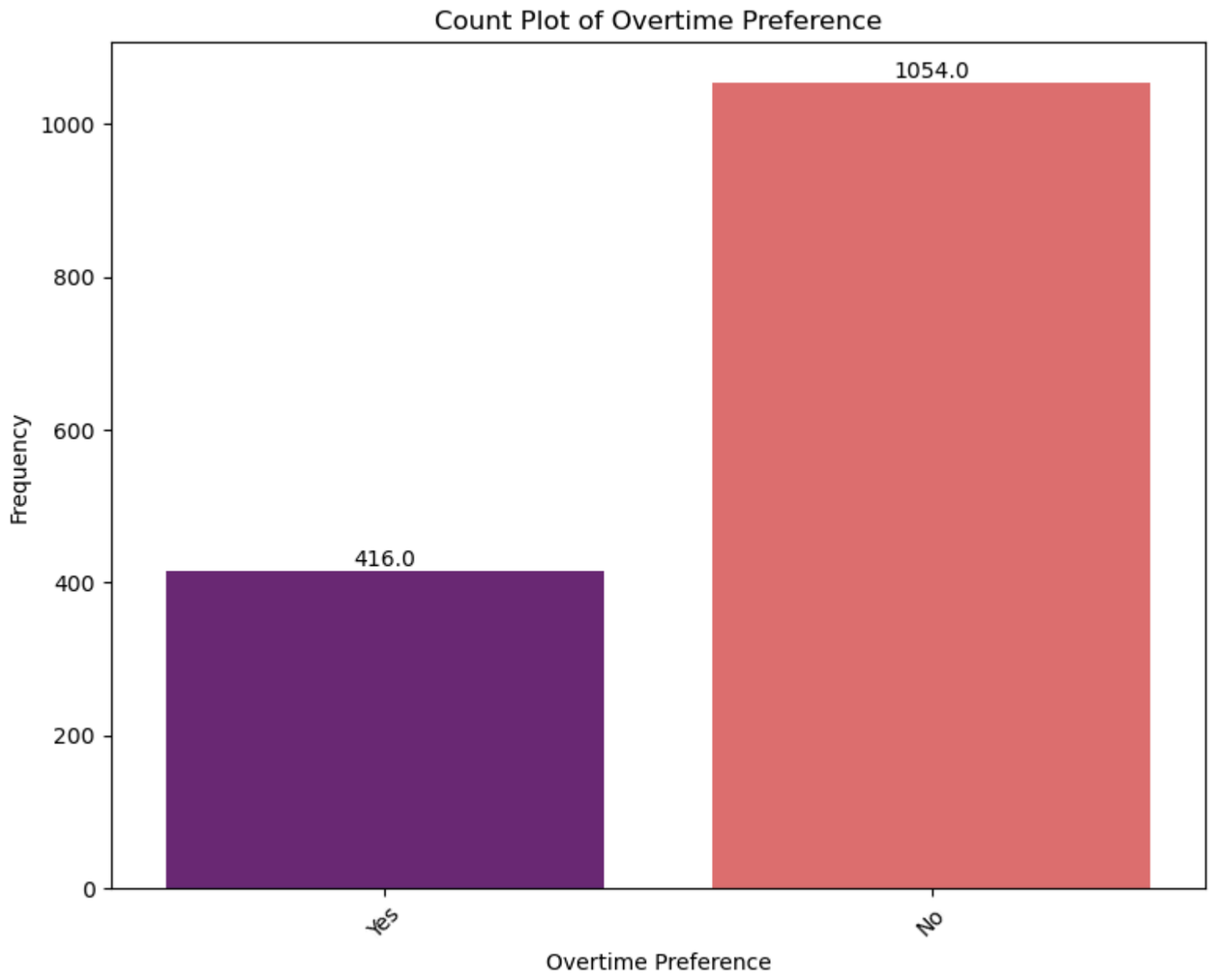
In [75]:
```
plt.figure(figsize=(13, 7))
sns.violinplot(data=df, y='NumCompaniesWorked', color='purple')
plt.title('Violin Plot of Number of Companies Worked')
plt.ylabel('Number of Companies Worked')
plt.show()
```



In [76]:
```
plt.figure(figsize=(9, 7))
sns.countplot(data=df, x='Over18', palette='plasma')
plt.title('Count Plot of Over 18')
plt.xlabel('Over 18')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.show()
```

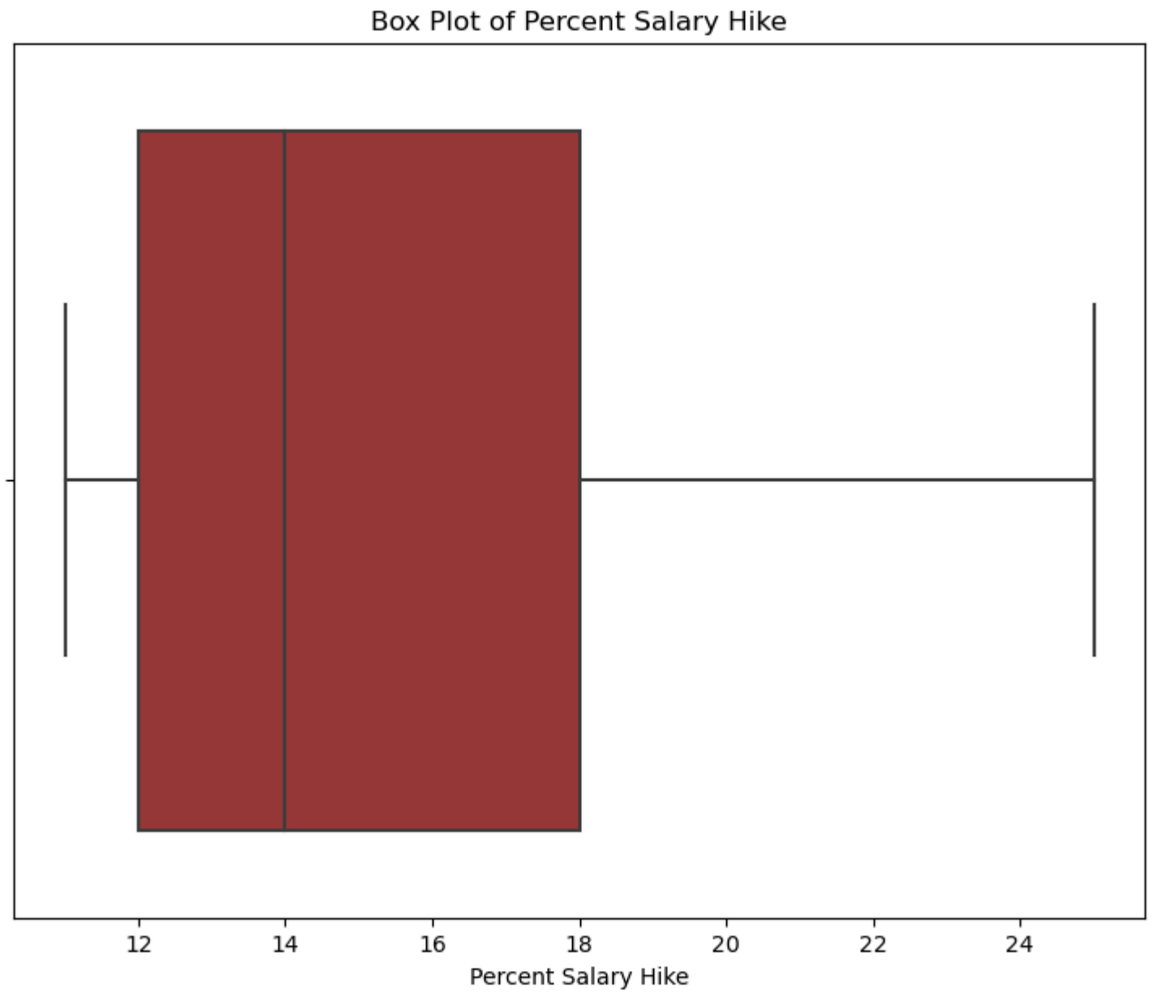


In [77]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='OverTime', palette='magma')
plt.title('Count Plot of Overtime Preference')
plt.xlabel('Overtime Preference')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```

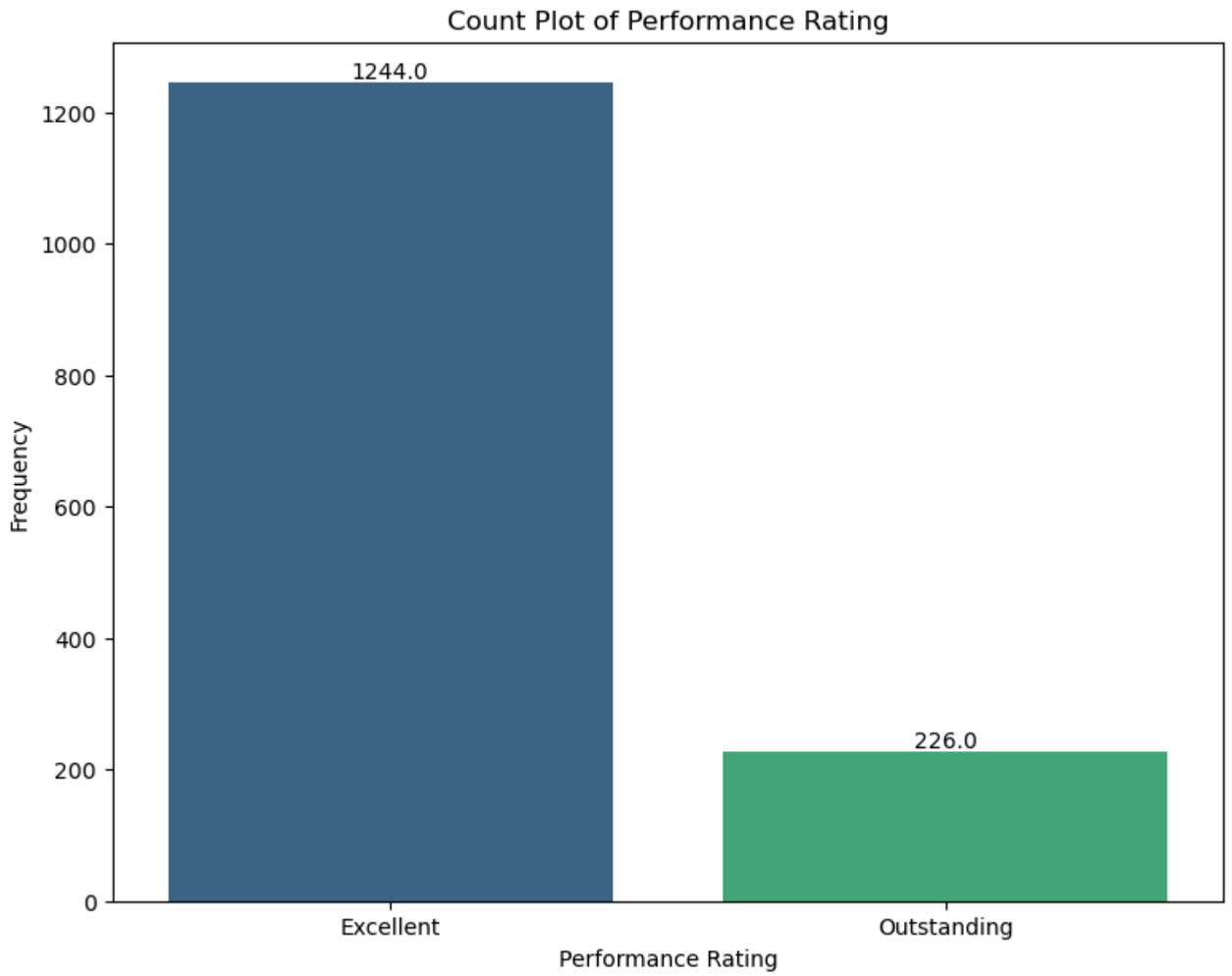


In [83]:
```
plt.figure(figsize=(13, 7))
sns.histplot(data=df, x='PercentSalaryHike', bins=15, kde=True, color='indigo')
plt.title('Histogram of Percent Salary Hike')
plt.xlabel('Percent Salary Hike')
plt.ylabel('Frequency')
plt.show()
```

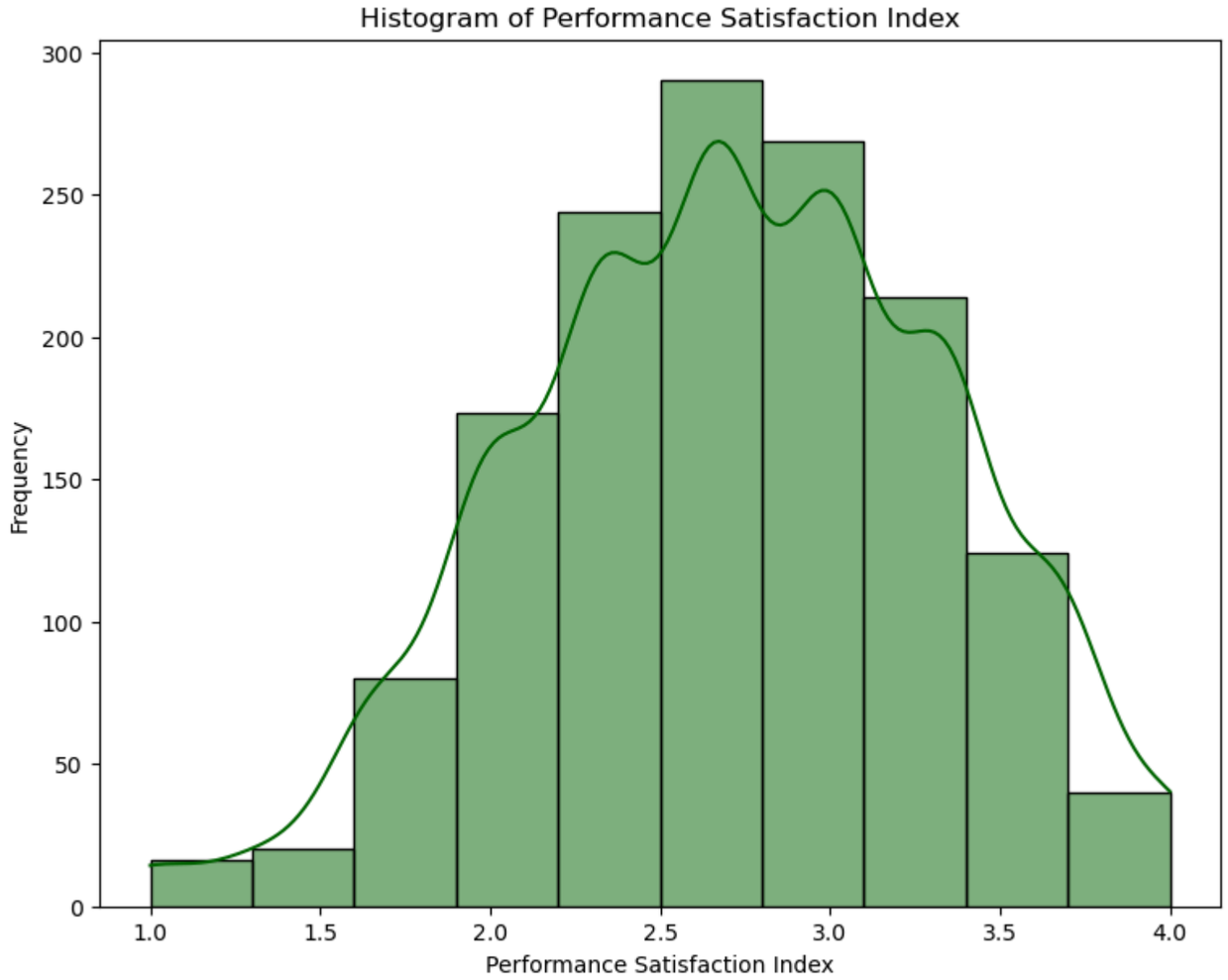
In [84]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='PercentSalaryHike', color='brown')
plt.title('Box Plot of Percent Salary Hike')
plt.xlabel('Percent Salary Hike')
plt.show()
```



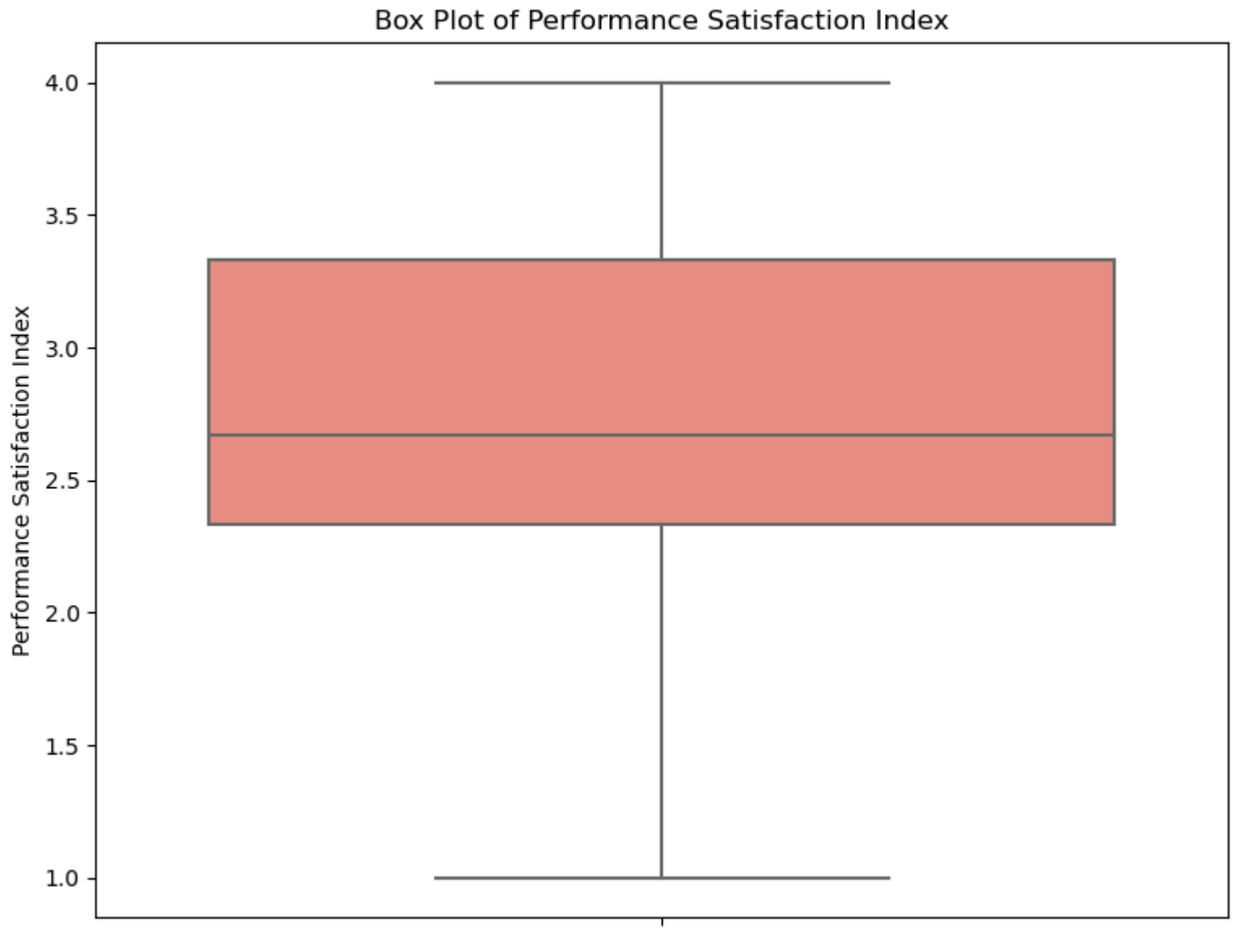
In [85]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='PerformanceRating', palette='viridis')
plt.title('Count Plot of Performance Rating')
plt.xlabel('Performance Rating')
plt.ylabel('Frequency')
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



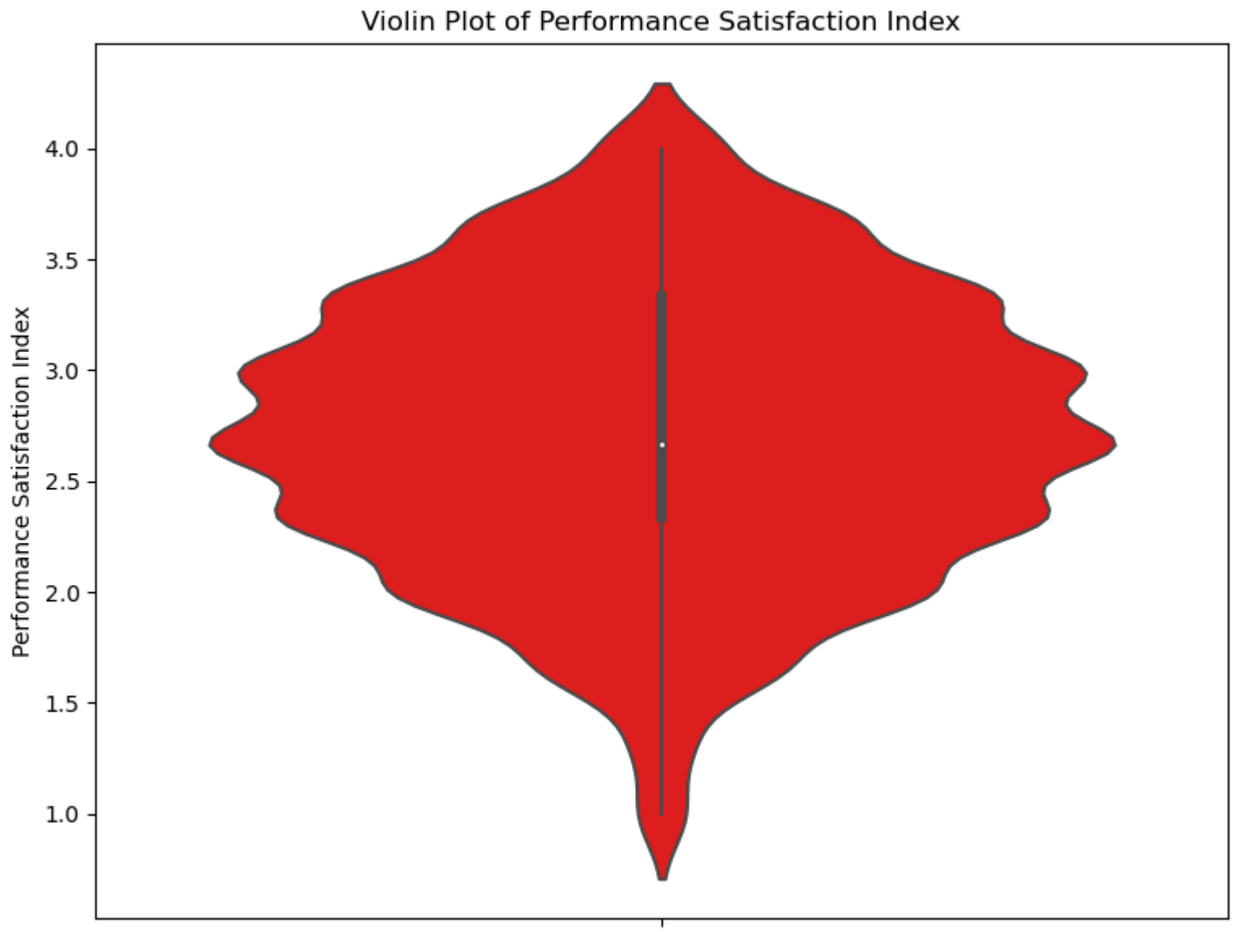
In [95]:
```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='PerformanceSatisfactionIndex', bins=10, kde=True, color='darkgreen')
plt.title('Histogram of Performance Satisfaction Index')
plt.xlabel('Performance Satisfaction Index')
plt.ylabel('Frequency')
plt.show()
```



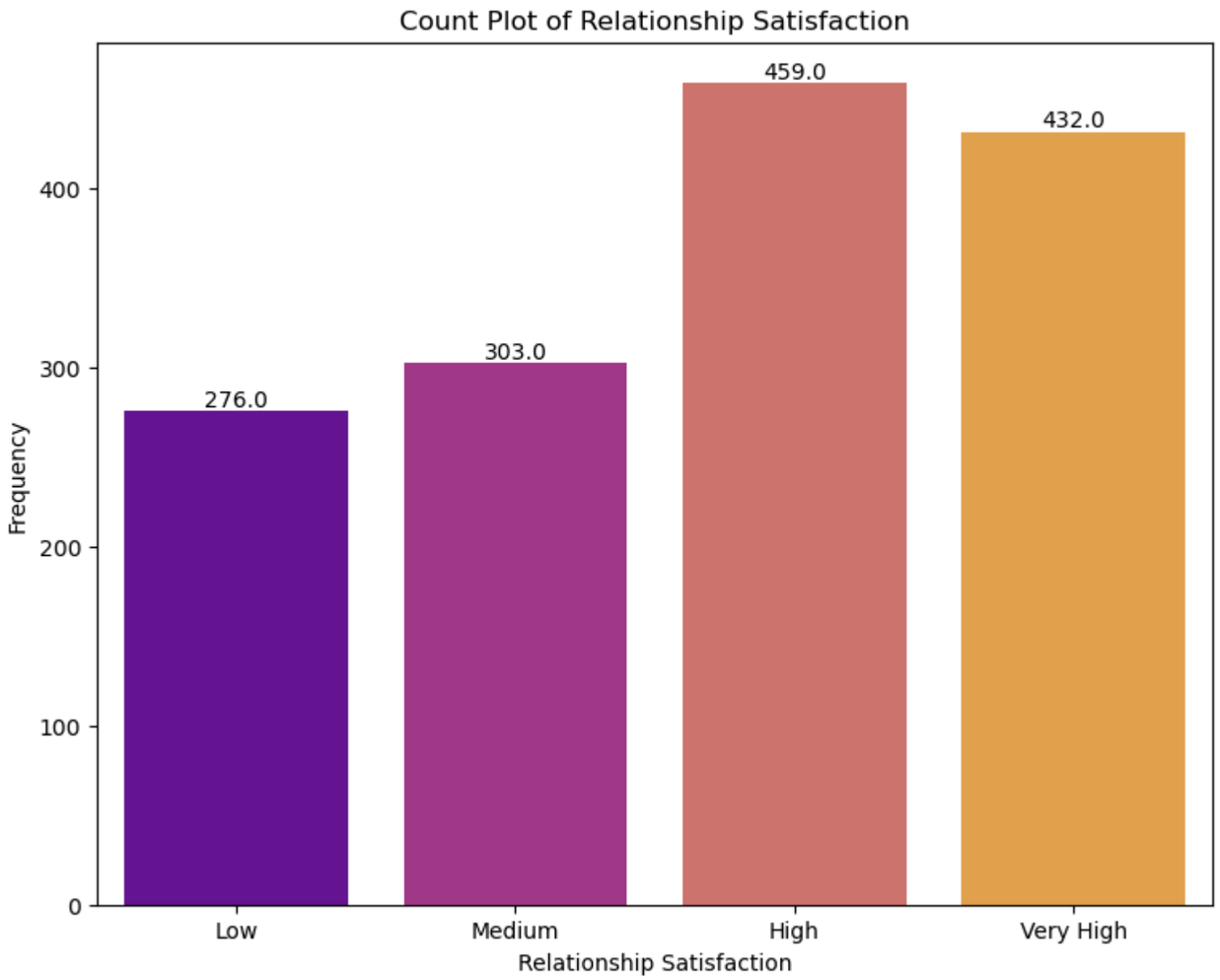
In [96]:
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, y='PerformanceSatisfactionIndex', color='salmon')
plt.title('Box Plot of Performance Satisfaction Index')
plt.ylabel('Performance Satisfaction Index')
plt.show()
```



In [97]:
```
plt.figure(figsize=(9, 7))
sns.violinplot(data=df, y='PerformanceSatisfactionIndex', color='red')
plt.title('Violin Plot of Performance Satisfaction Index')
plt.ylabel('Performance Satisfaction Index')
plt.show()
```

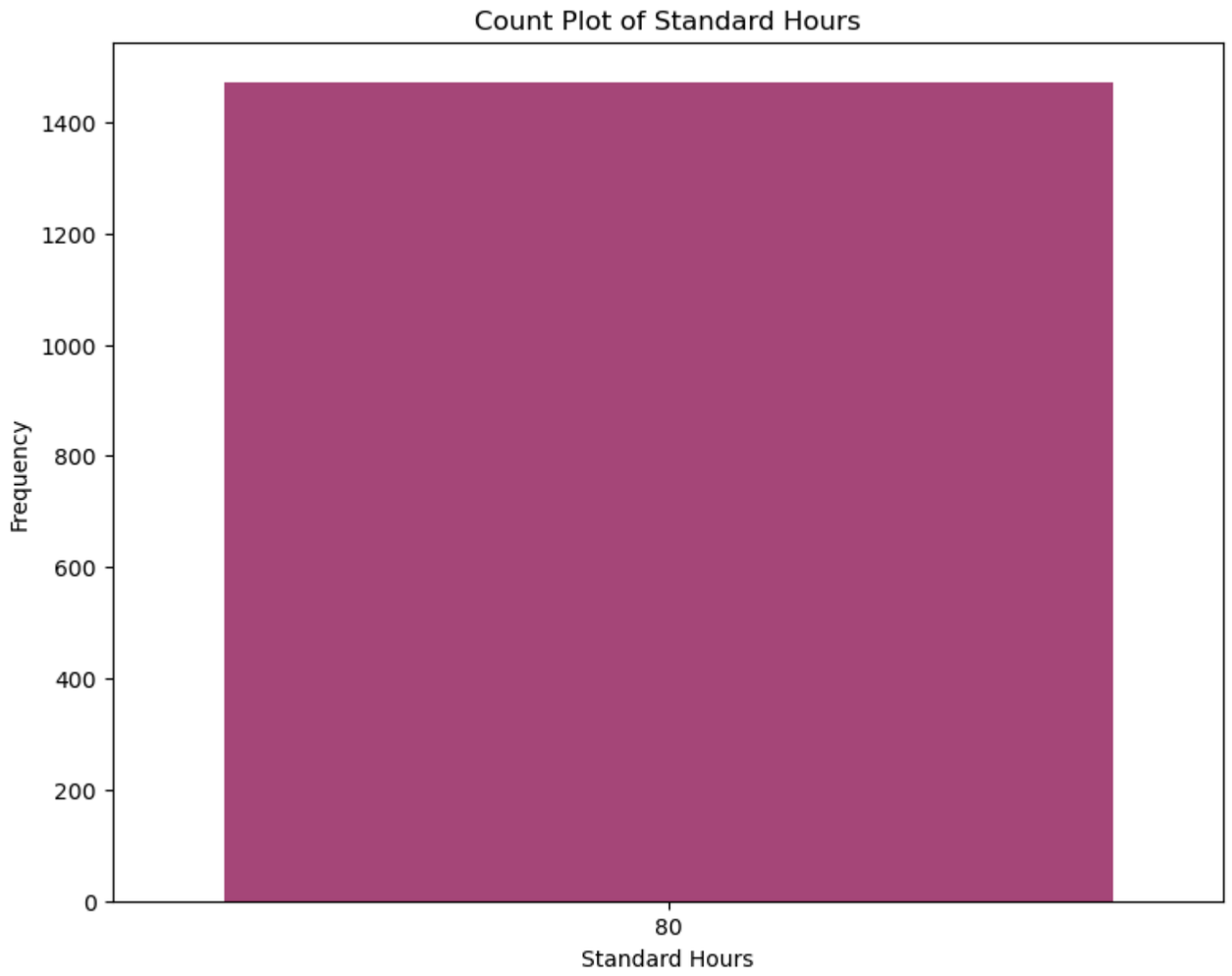


In [98]:
```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='RelationshipSatisfaction', palette='plasma', order=['Low', 'Medium', 'High', 'Very High'])
plt.title('Count Plot of Relationship Satisfaction')
plt.xlabel('Relationship Satisfaction')
plt.ylabel('Frequency')
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



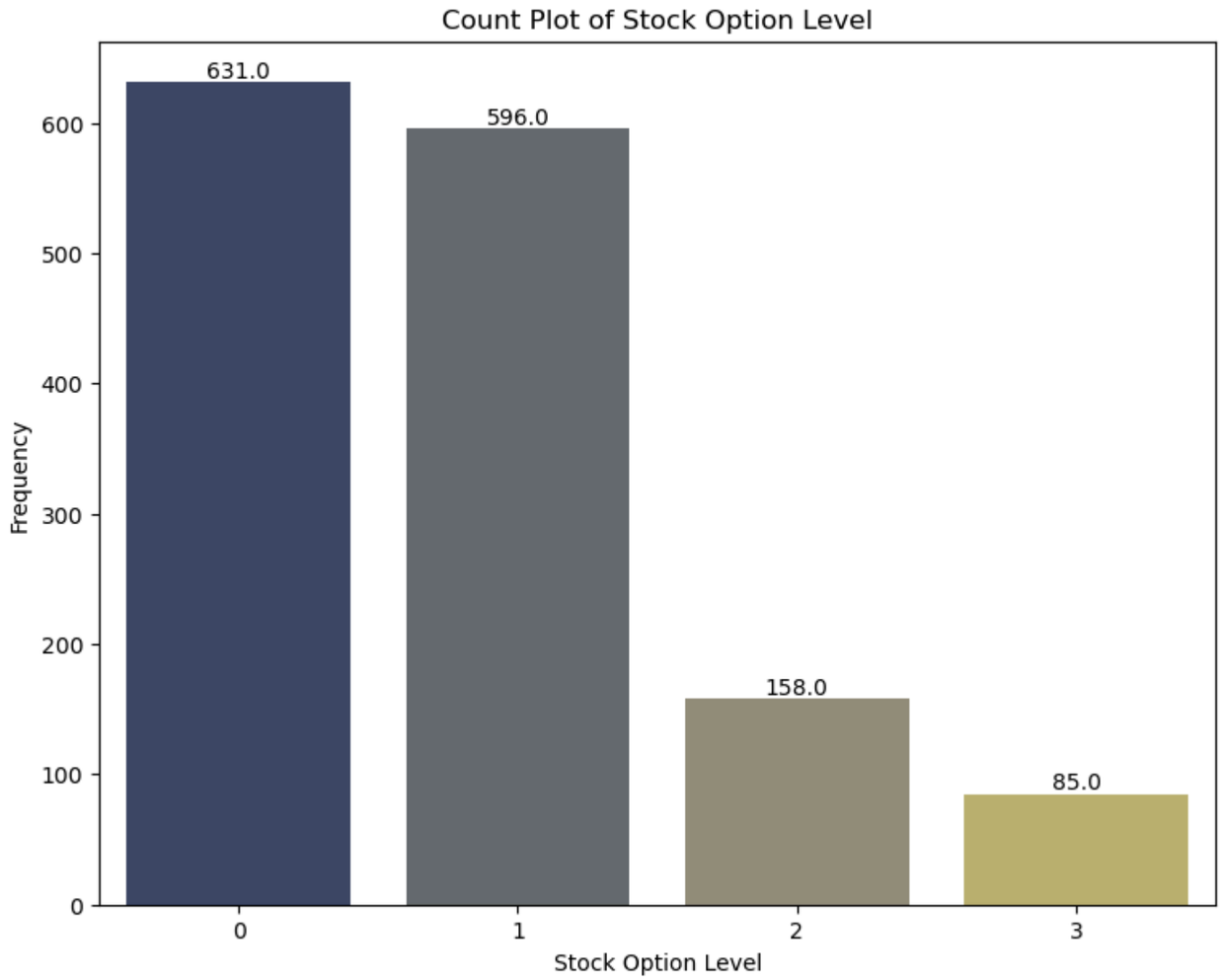
In [101…

```
plt.figure(figsize=(9, 7))
sns.countplot(data=df, x='StandardHours', palette='magma')
plt.title('Count Plot of Standard Hours')
plt.xlabel('Standard Hours')
plt.ylabel('Frequency')
plt.show()
```



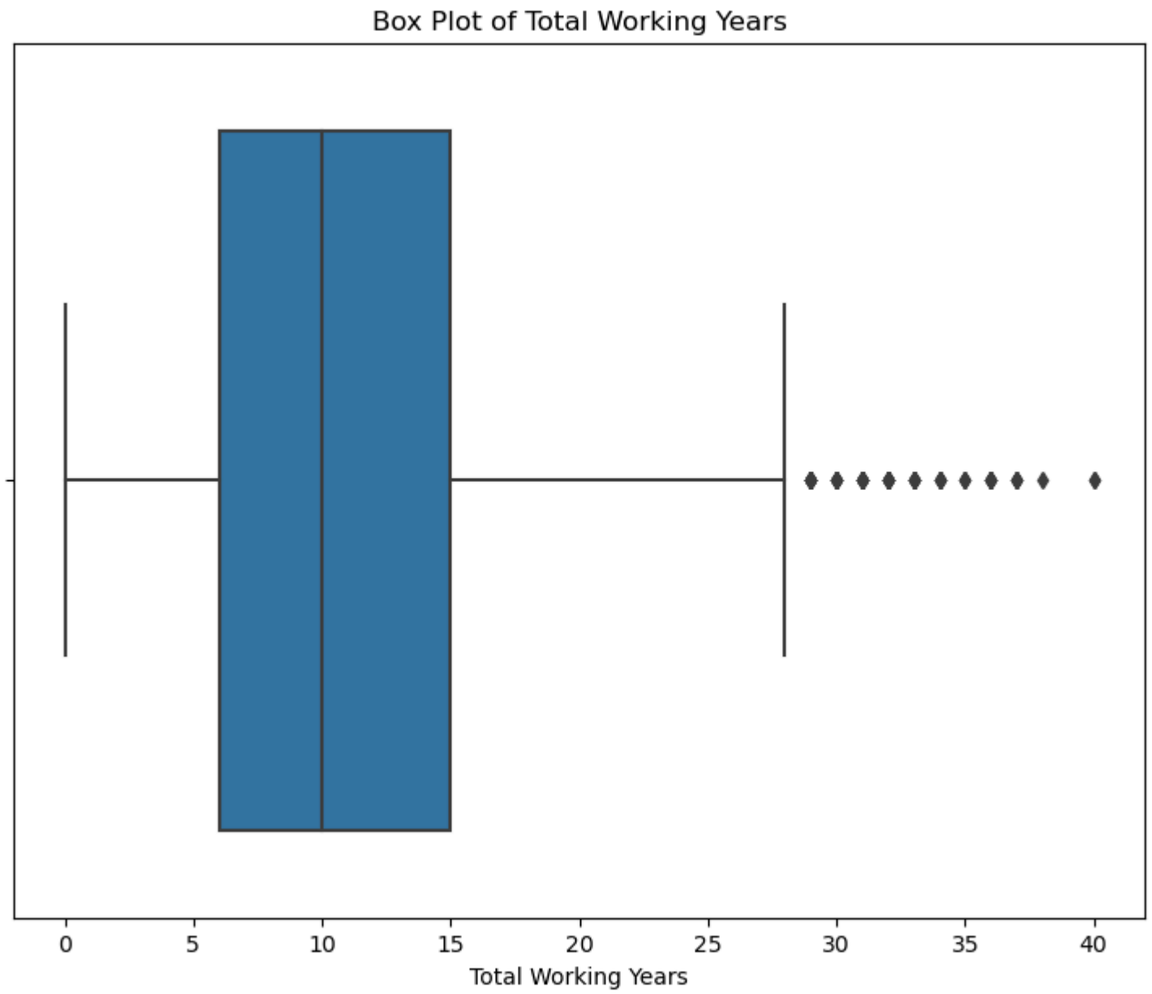
In [104…

```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='StockOptionLevel', palette='cividis')
plt.title('Count Plot of Stock Option Level')
plt.xlabel('Stock Option Level')
plt.ylabel('Frequency')
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



In [107…

```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='TotalWorkingYears')
plt.title('Box Plot of Total Working Years')
plt.xlabel('Total Working Years')
plt.show()
```

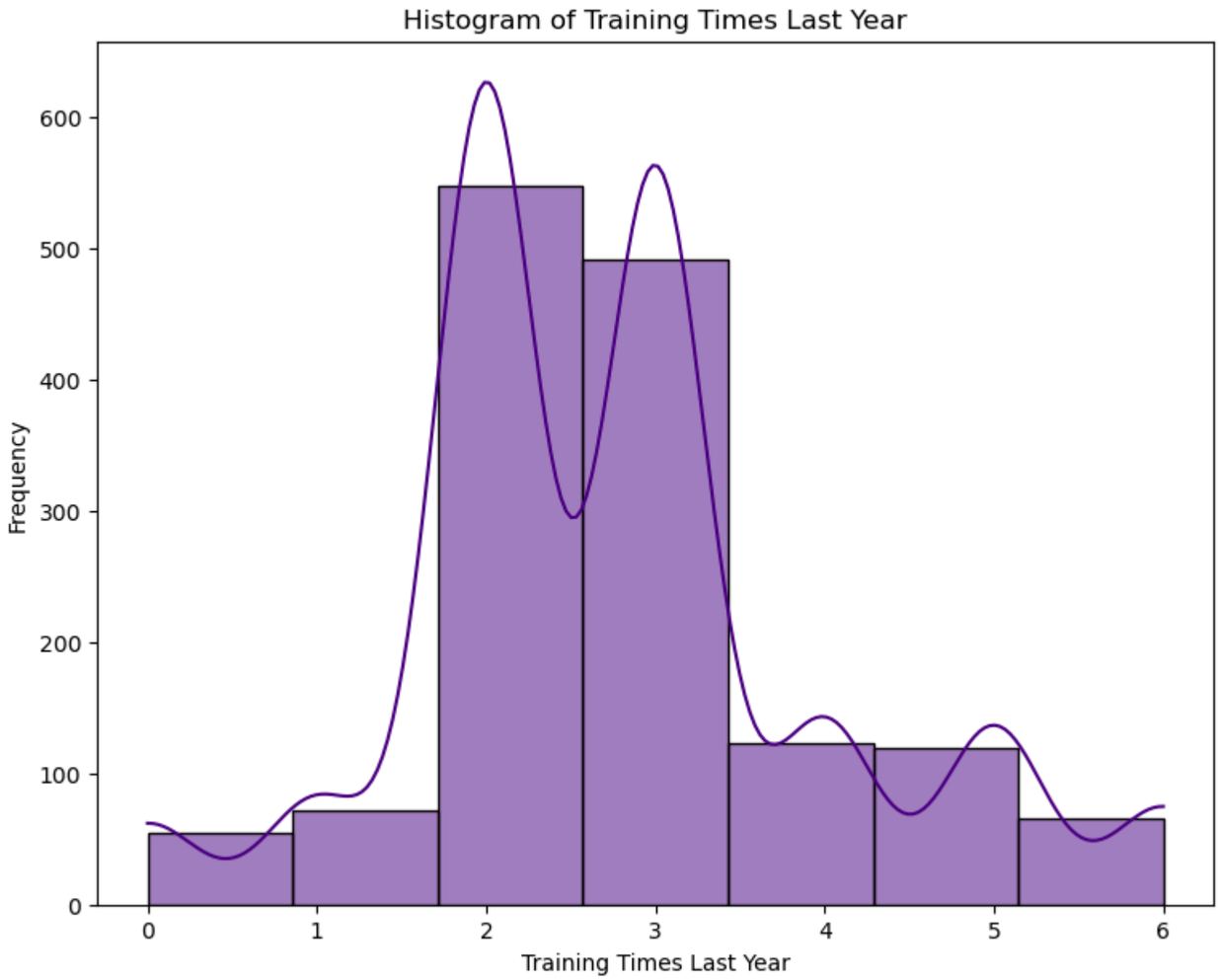


In [108…

```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='TotalWorkingYears', bins=10, kde=True, color='red')
plt.title('Histogram of Total Working Years')
plt.xlabel('Total Working Years')
plt.ylabel('Frequency')
plt.show()
```

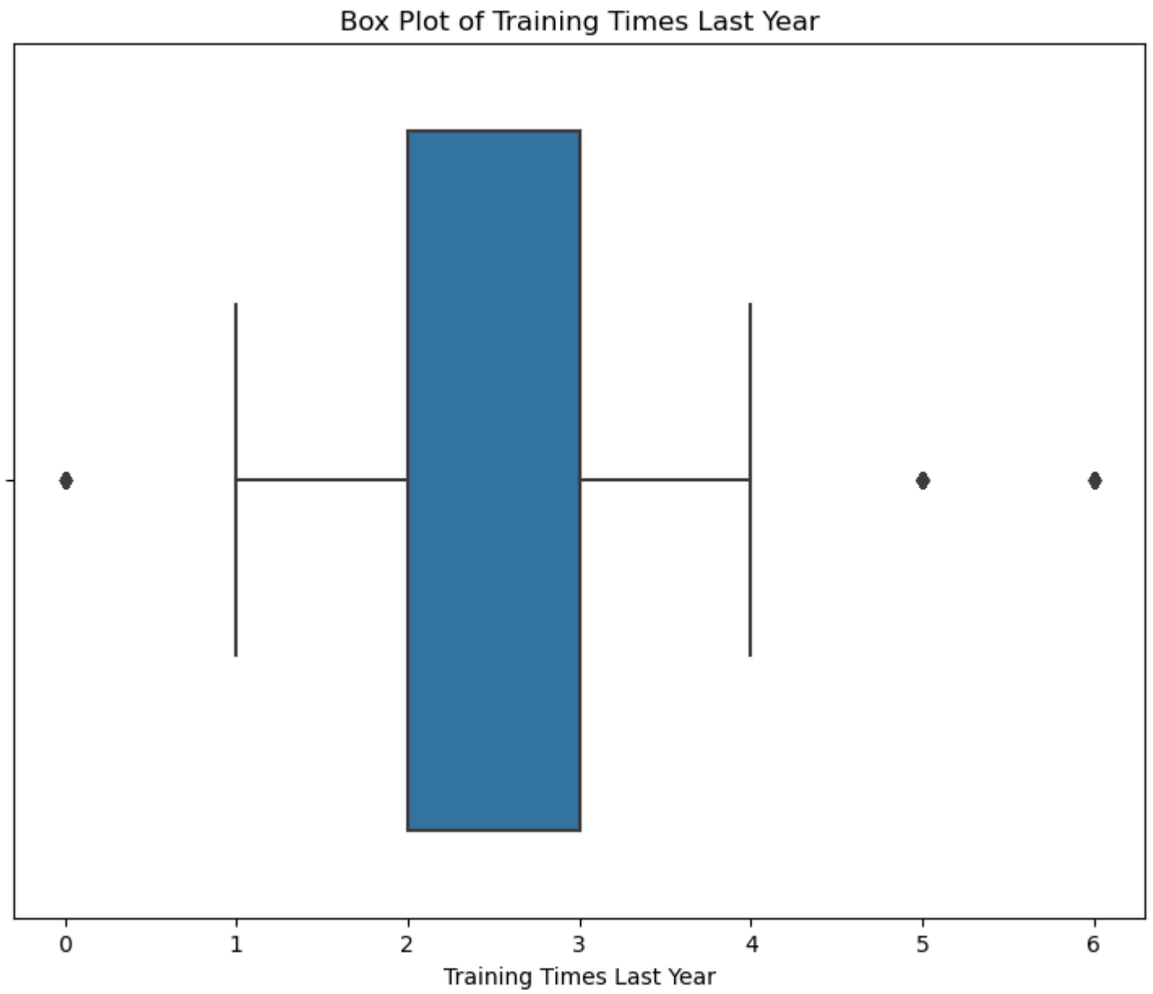
In [110…

```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='TrainingTimesLastYear', bins=7, kde=True,color='indigo')
plt.title('Histogram of Training Times Last Year')
plt.xlabel('Training Times Last Year')
plt.ylabel('Frequency')
plt.show()
```



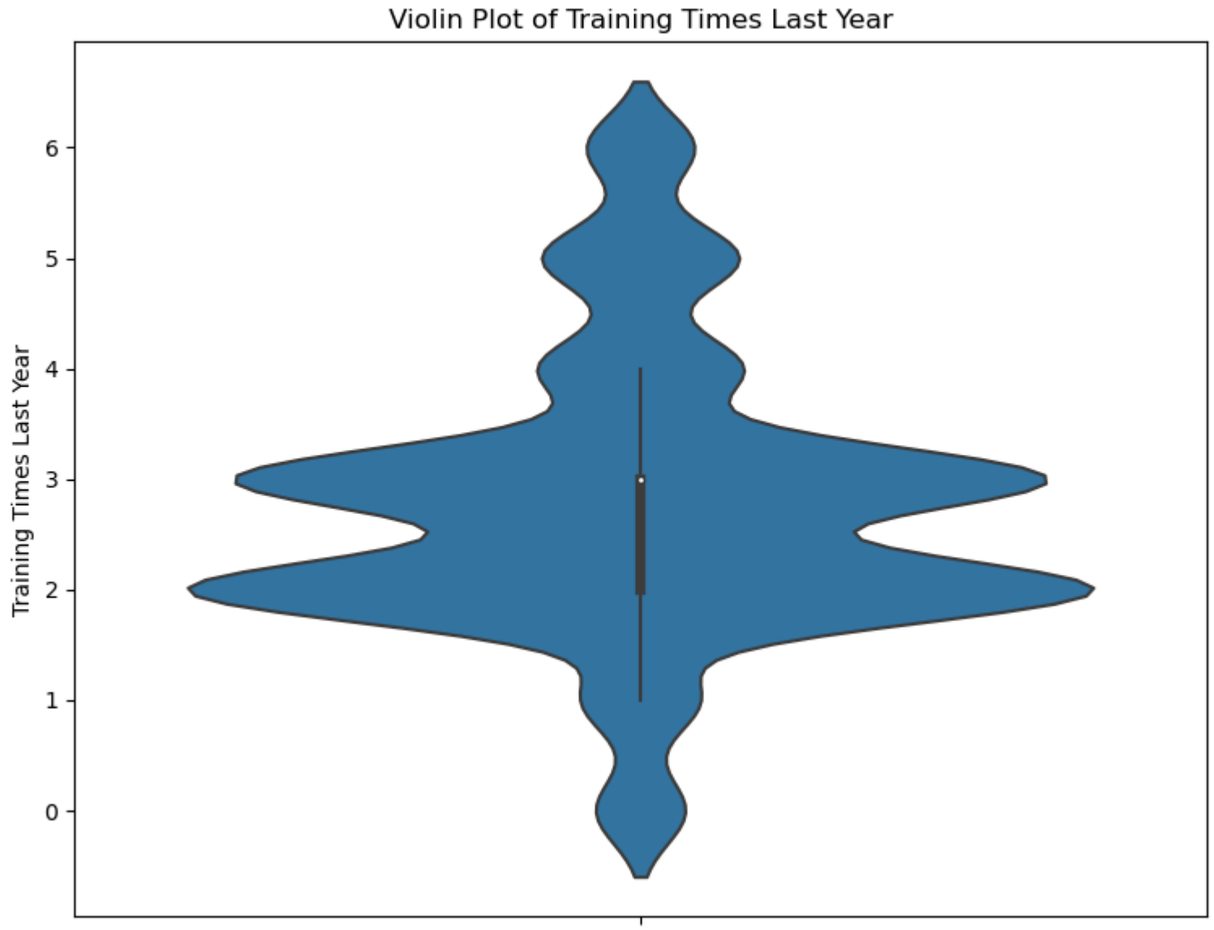
In [111…

```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='TrainingTimesLastYear')
plt.title('Box Plot of Training Times Last Year')
plt.xlabel('Training Times Last Year')
plt.show()
```



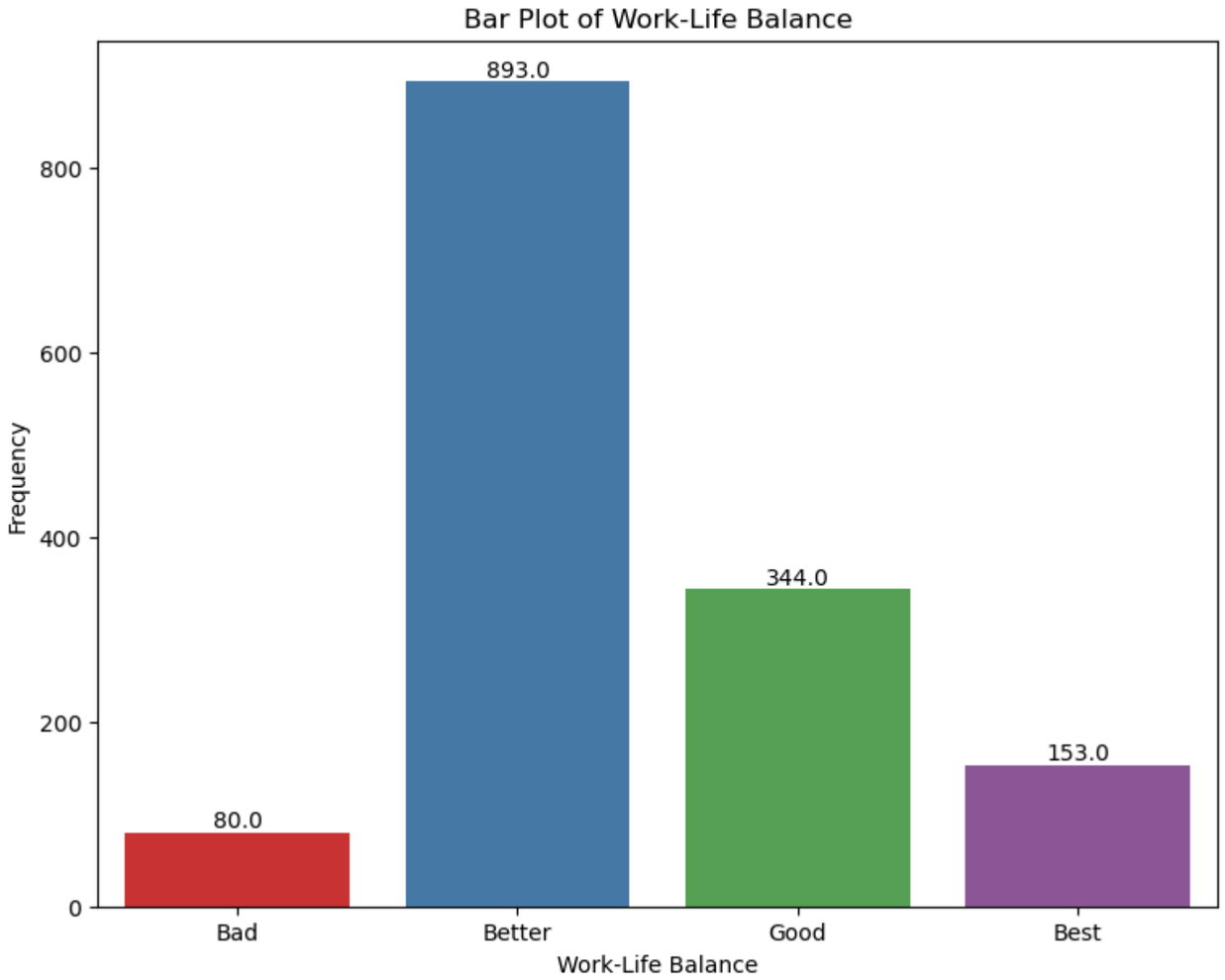
In [112…

```
plt.figure(figsize=(9, 7))
sns.violinplot(data=df, y='TrainingTimesLastYear')
plt.title('Violin Plot of Training Times Last Year')
plt.ylabel('Training Times Last Year')
plt.show()
```



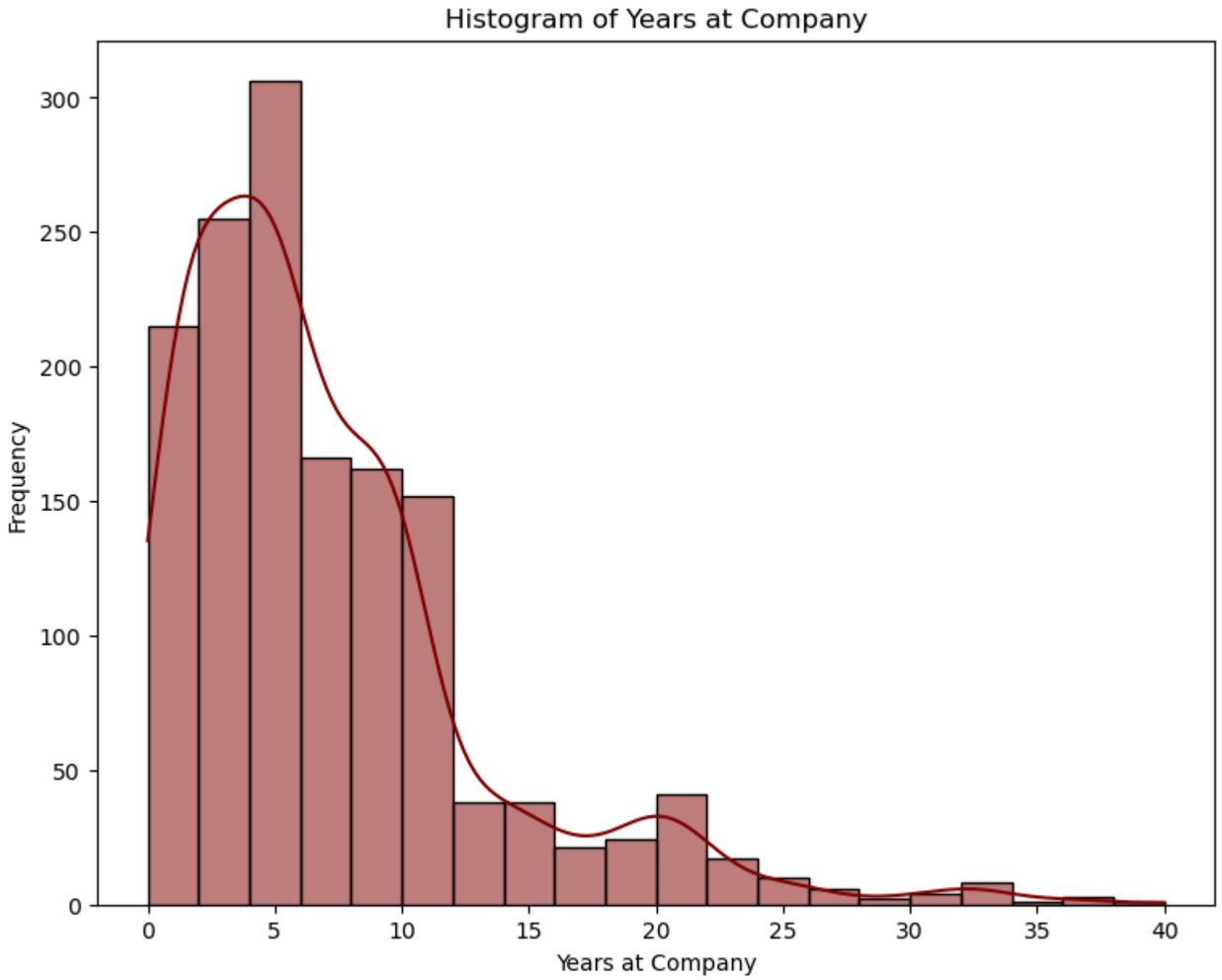
In [113…

```
plt.figure(figsize=(9, 7))
ax = sns.countplot(data=df, x='WorkLifeBalance', palette='Set1')
plt.title('Bar Plot of Work-Life Balance')
plt.xlabel('Work-Life Balance')
plt.ylabel('Frequency')
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.show()
```



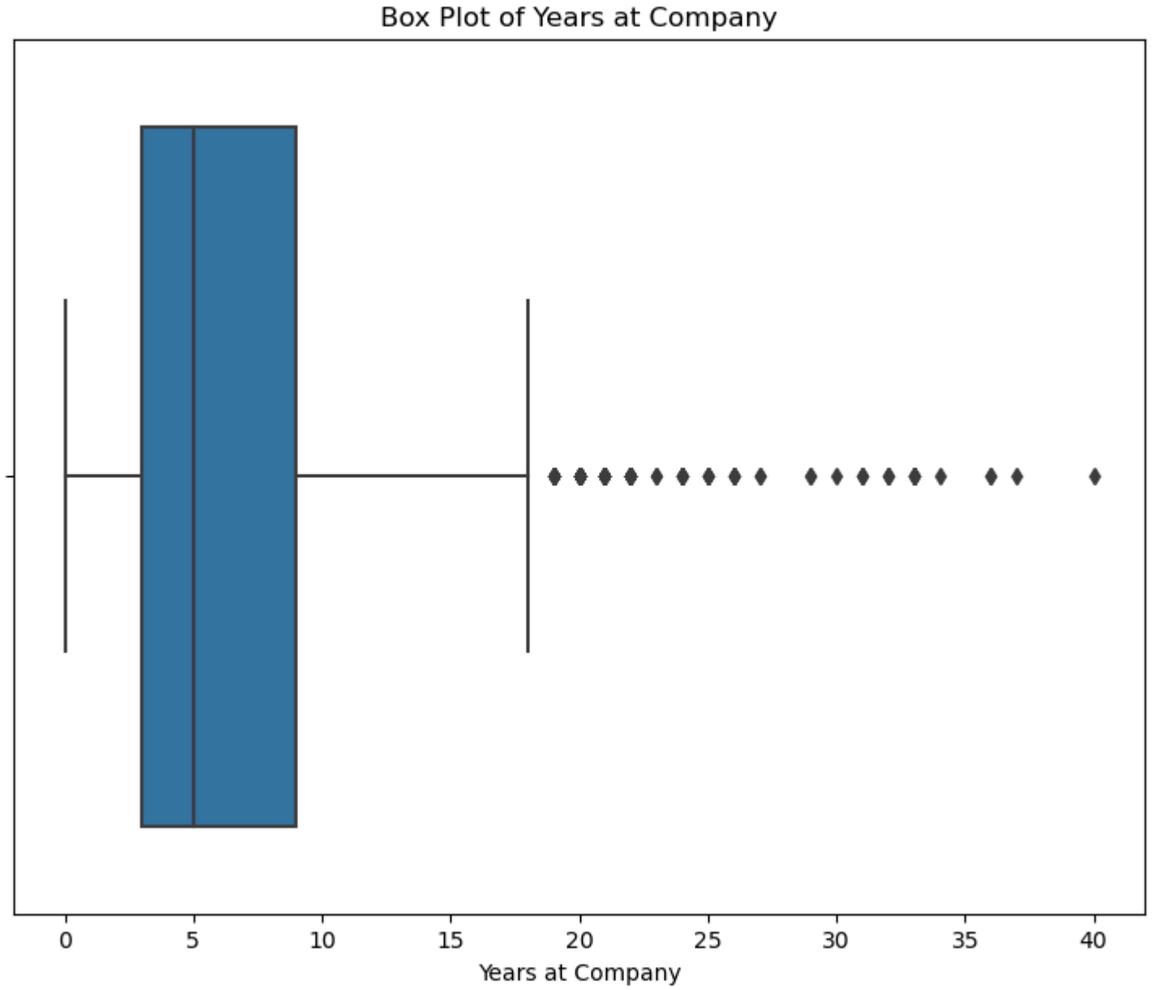
In [114…

```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='YearsAtCompany', bins=20, kde=True,color='maroon')
plt.title('Histogram of Years at Company')
plt.xlabel('Years at Company')
plt.ylabel('Frequency')
plt.show()
```



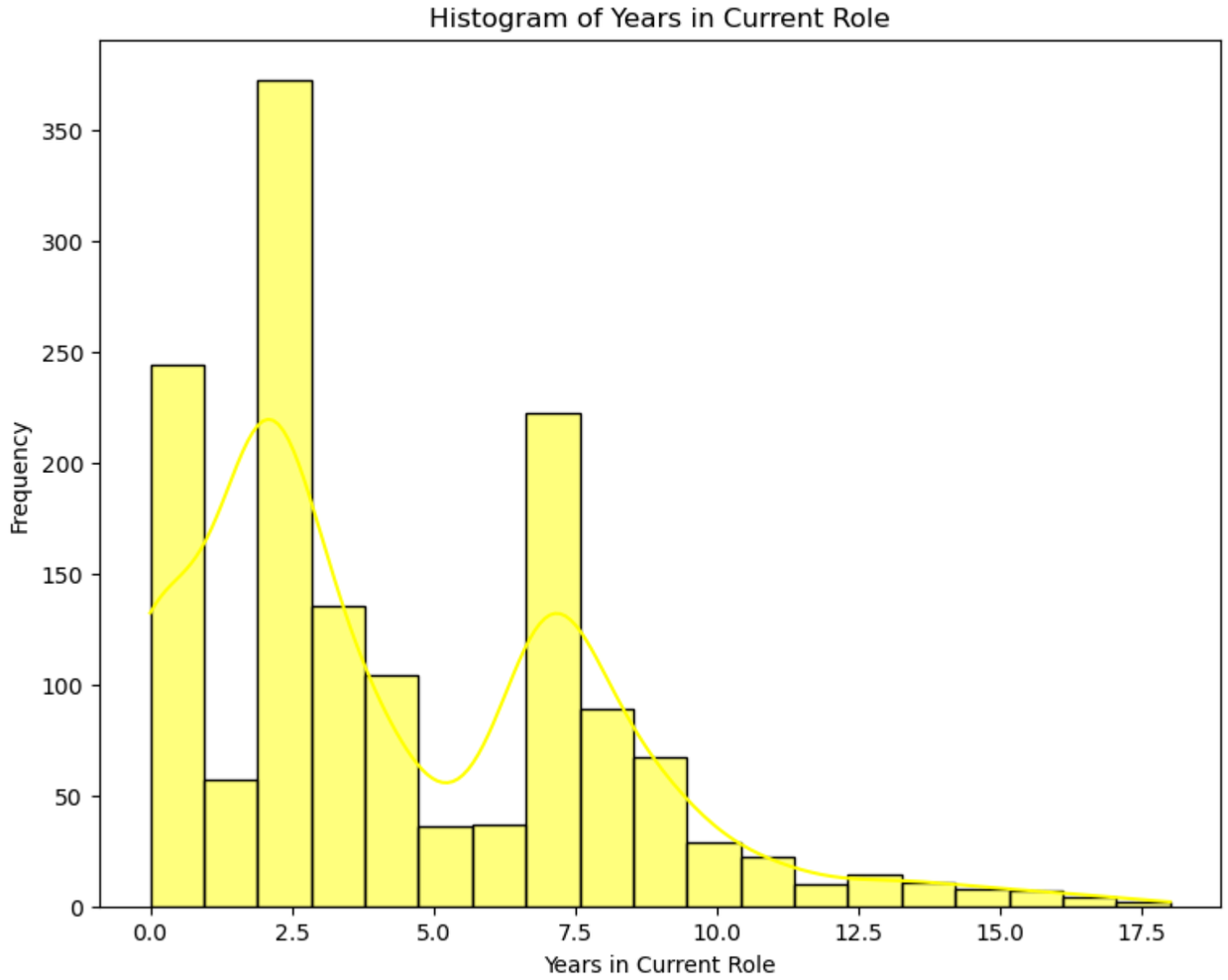
In [115…

```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='YearsAtCompany')
plt.title('Box Plot of Years at Company')
plt.xlabel('Years at Company')
plt.show()
```



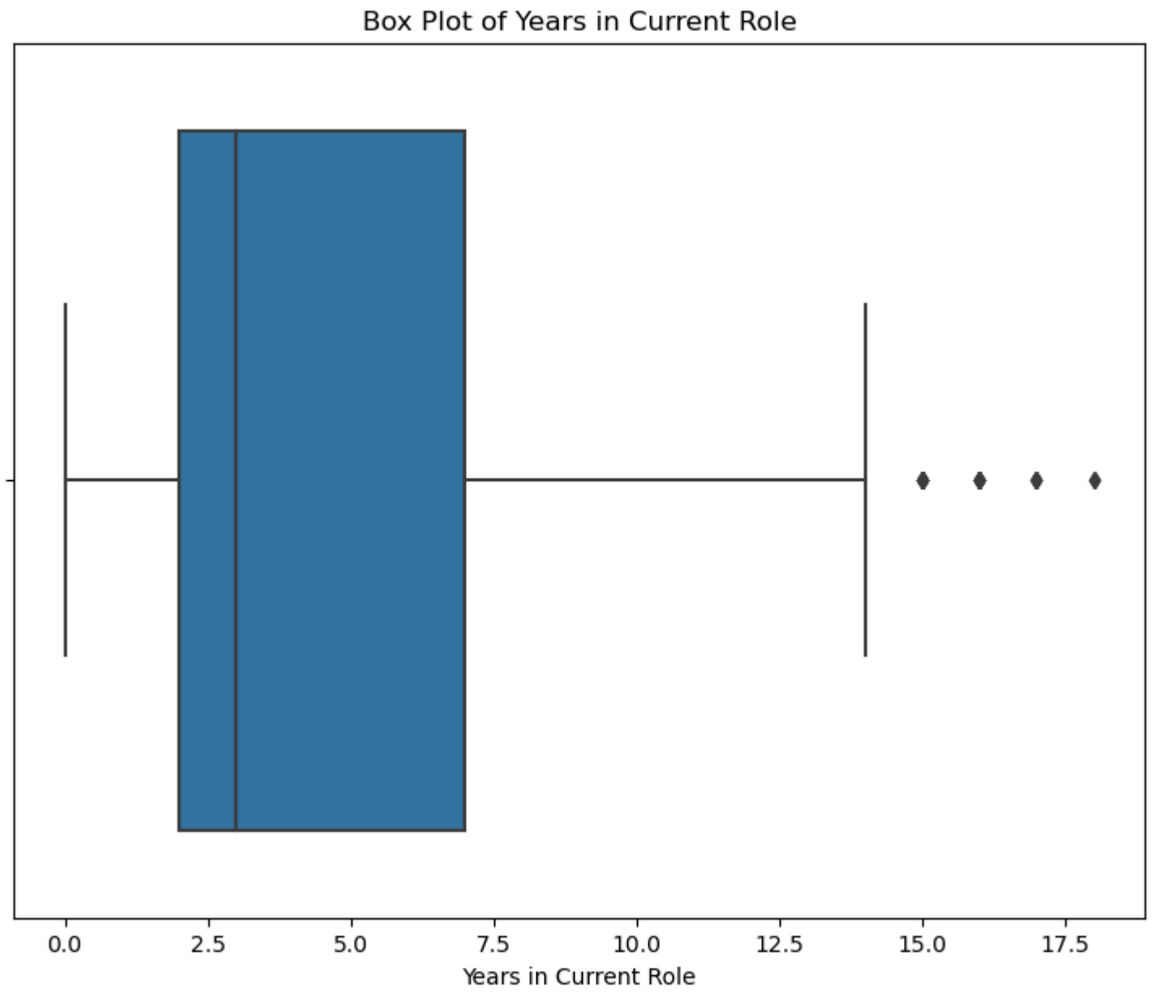
In [117…

```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='YearsInCurrentRole', bins=19, kde=True,color='yellow')
plt.title('Histogram of Years in Current Role')
plt.xlabel('Years in Current Role')
plt.ylabel('Frequency')
plt.show()
```



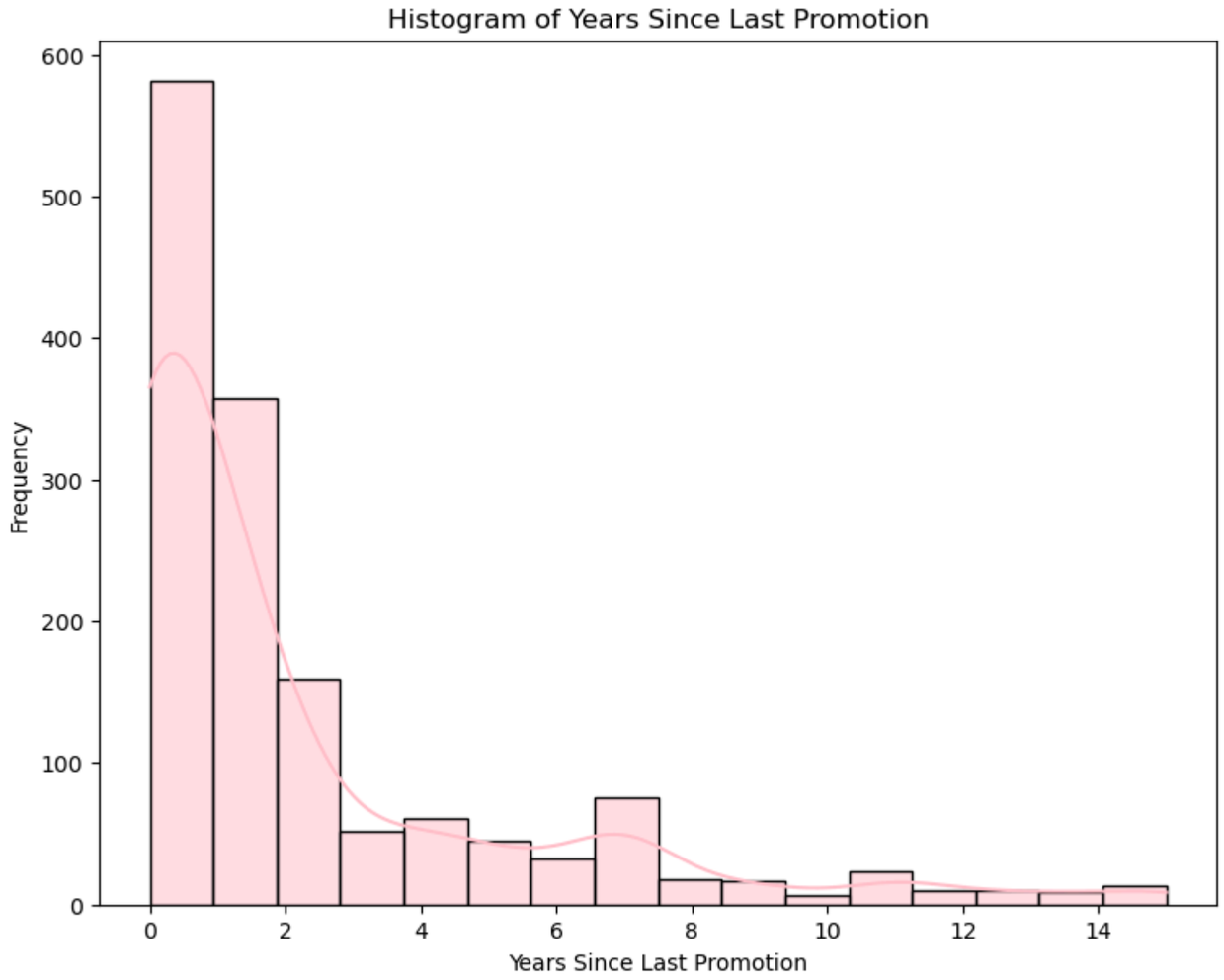
In [118…

```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='YearsInCurrentRole')
plt.title('Box Plot of Years in Current Role')
plt.xlabel('Years in Current Role')
plt.show()
```



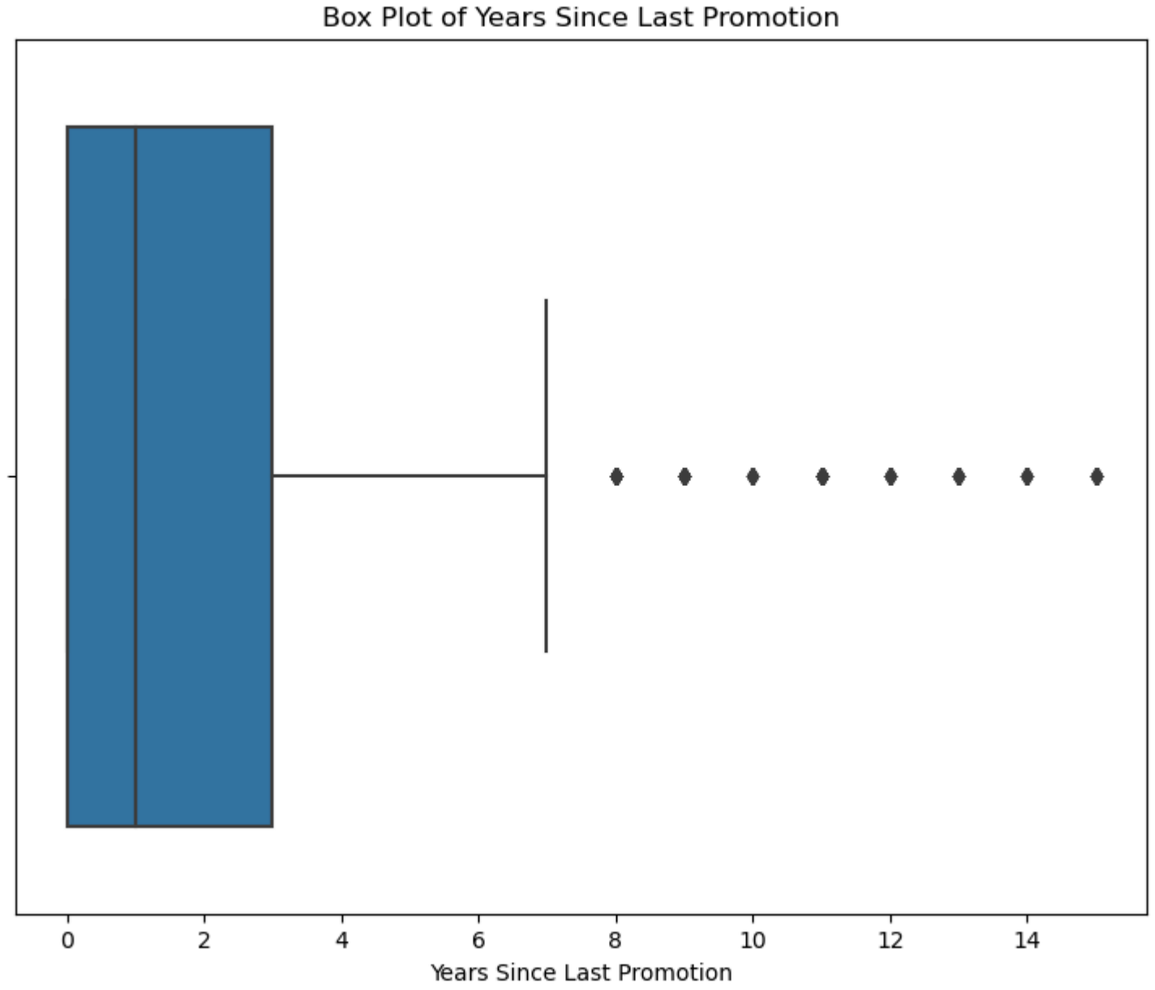
In [119…

```
plt.figure(figsize=(9, 7))
sns.histplot(data=df, x='YearsSinceLastPromotion', bins=16, kde=True, color='pink')
plt.title('Histogram of Years Since Last Promotion')
plt.xlabel('Years Since Last Promotion')
plt.ylabel('Frequency')
plt.show()
```



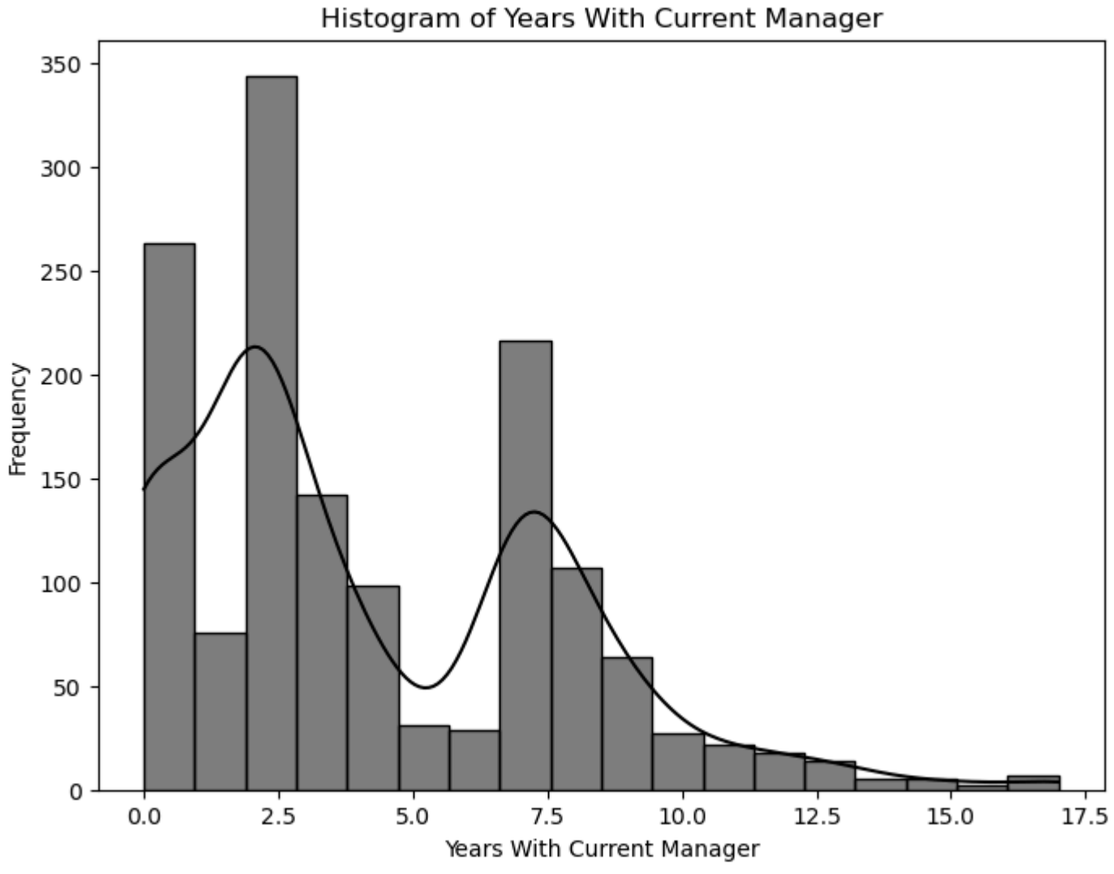
In [120…

```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='YearsSinceLastPromotion')
plt.title('Box Plot of Years Since Last Promotion')
plt.xlabel('Years Since Last Promotion')
plt.show()
```



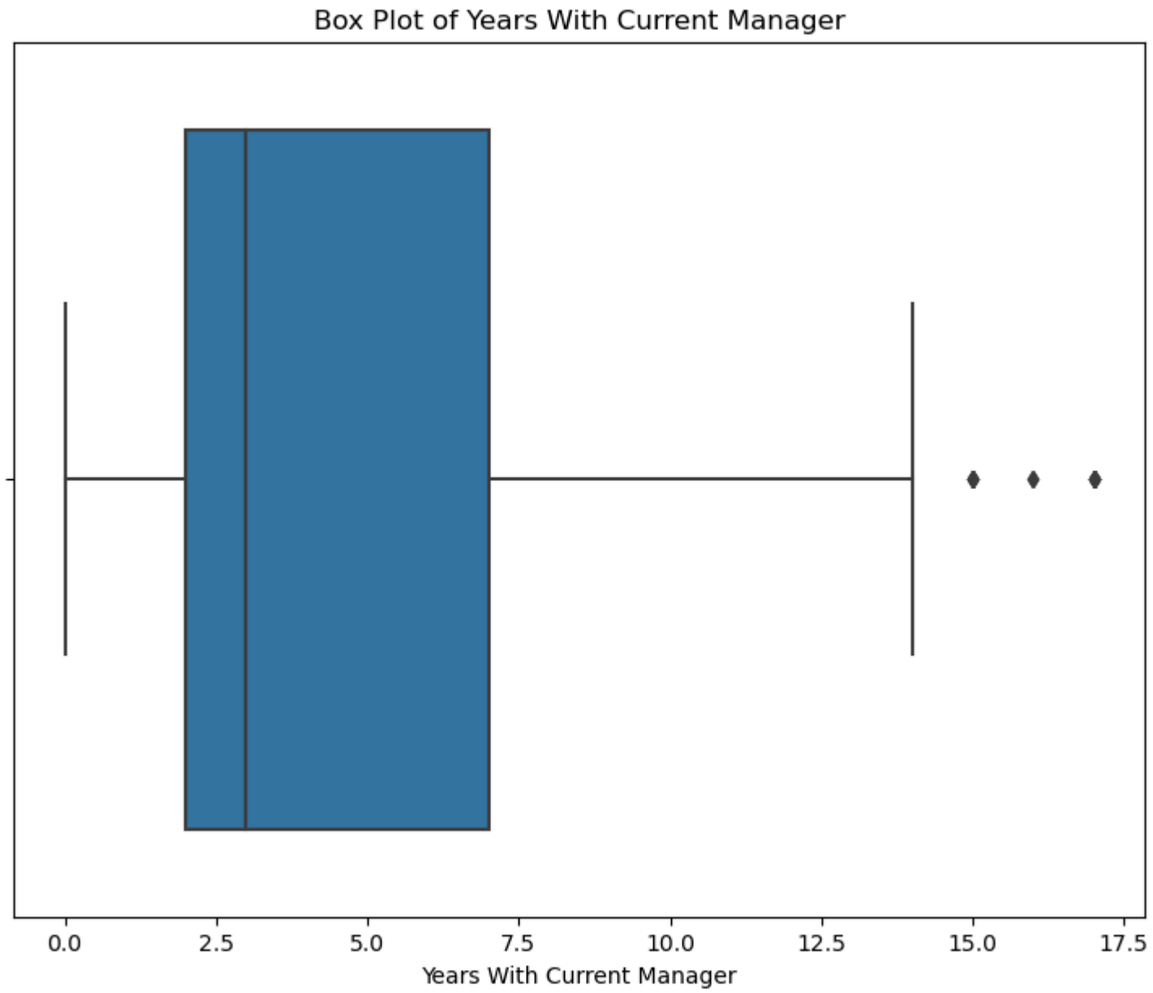
In [121…

```
plt.figure(figsize=(8, 6))
sns.histplot(data=df, x='YearsWithCurrManager', bins=18, kde=True,color='Black')
plt.title('Histogram of Years With Current Manager')
plt.xlabel('Years With Current Manager')
plt.ylabel('Frequency')
plt.show()
```



In [122…

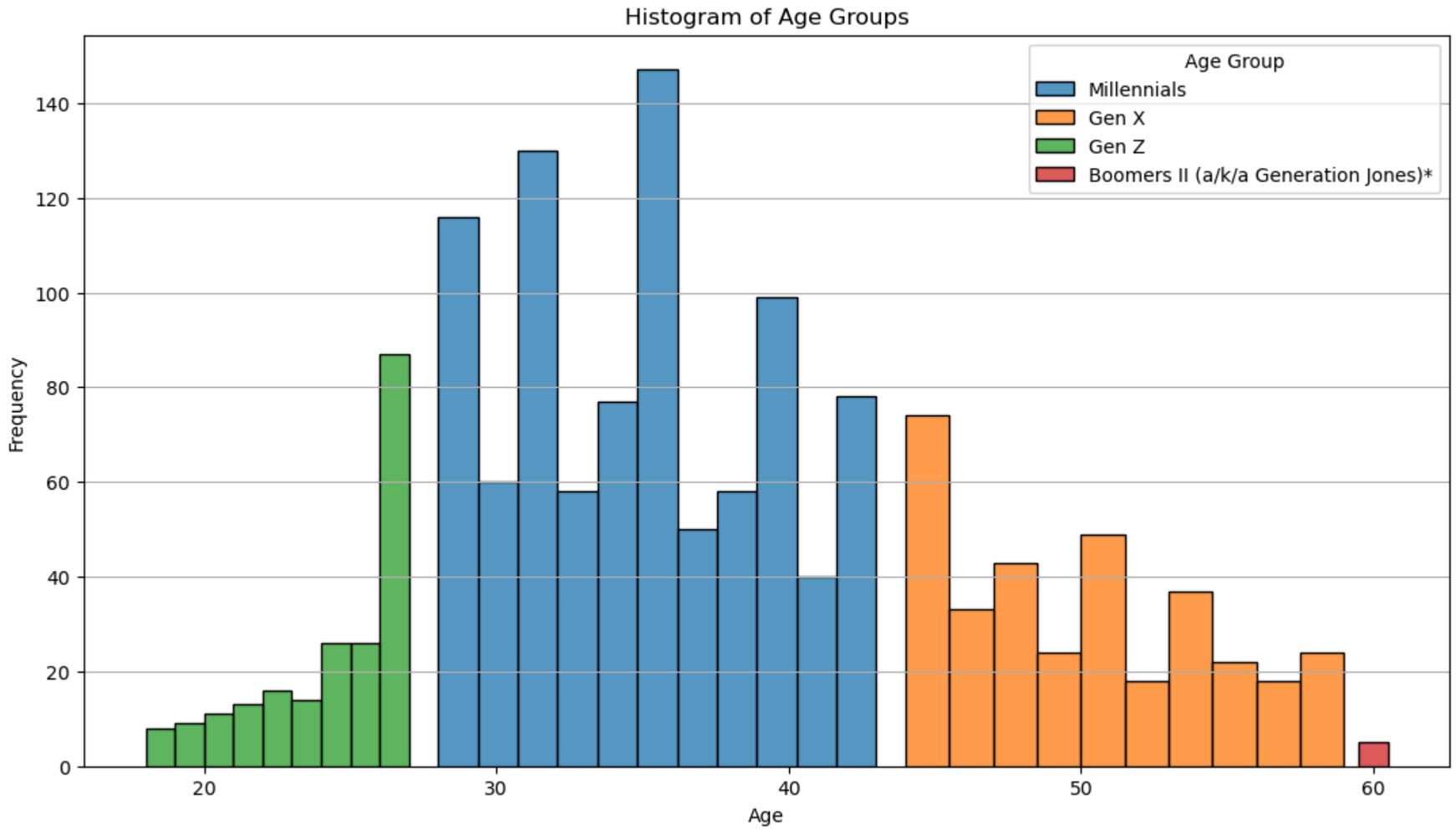
```
plt.figure(figsize=(9, 7))
sns.boxplot(data=df, x='YearsWithCurrManager')
plt.title('Box Plot of Years With Current Manager')
plt.xlabel('Years With Current Manager')
plt.show()
```



Bivariate Analysis

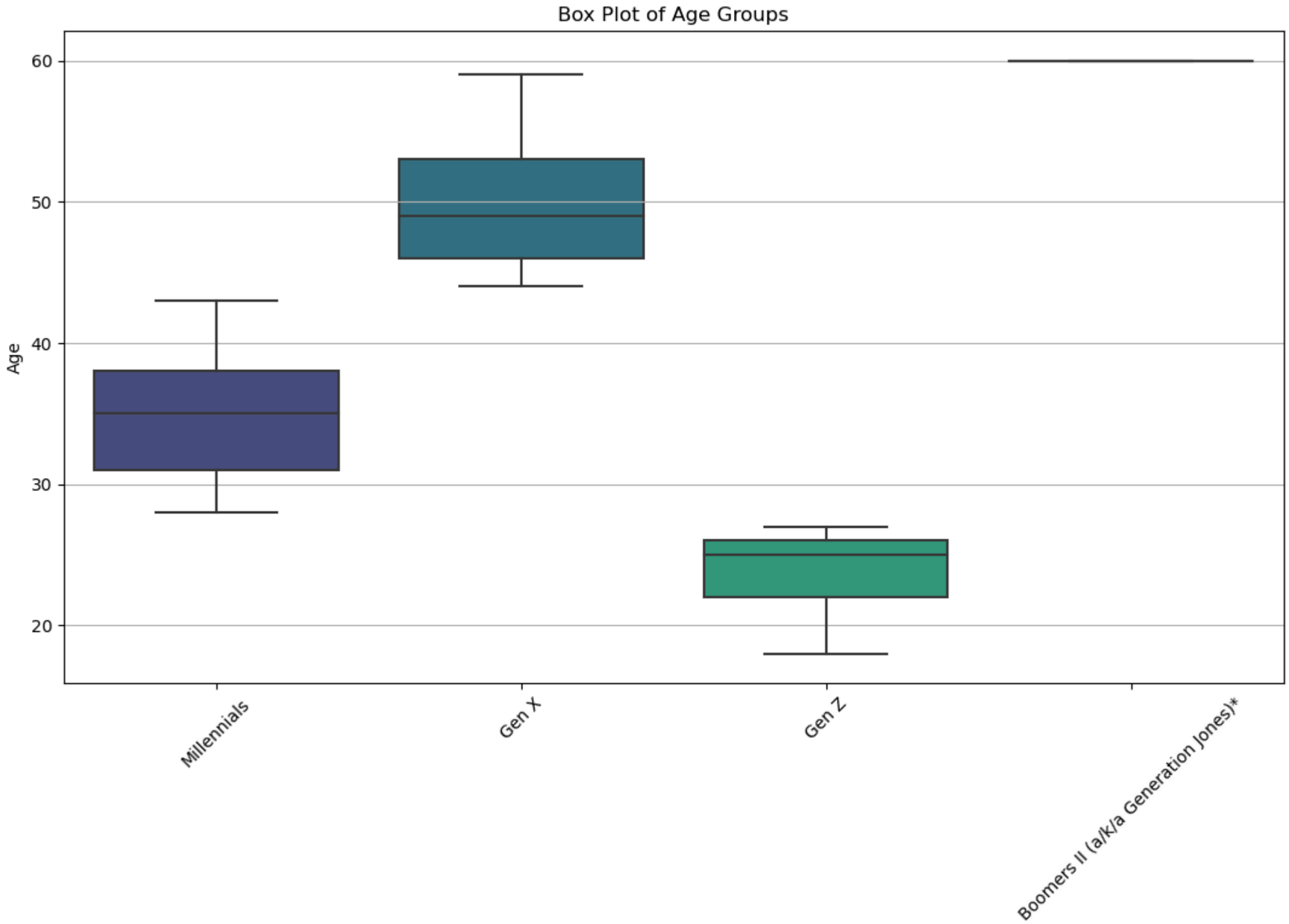
In [126…

```
plt.figure(figsize=(13, 7))
for age_group in df['AgeGeneration'].unique():
    sns.histplot(df[df['AgeGeneration'] == age_group]['Age'], kde=False, label=age_group)
plt.title('Histogram of Age Groups')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.legend(title='Age Group')
plt.grid(axis='y')
plt.show()
```



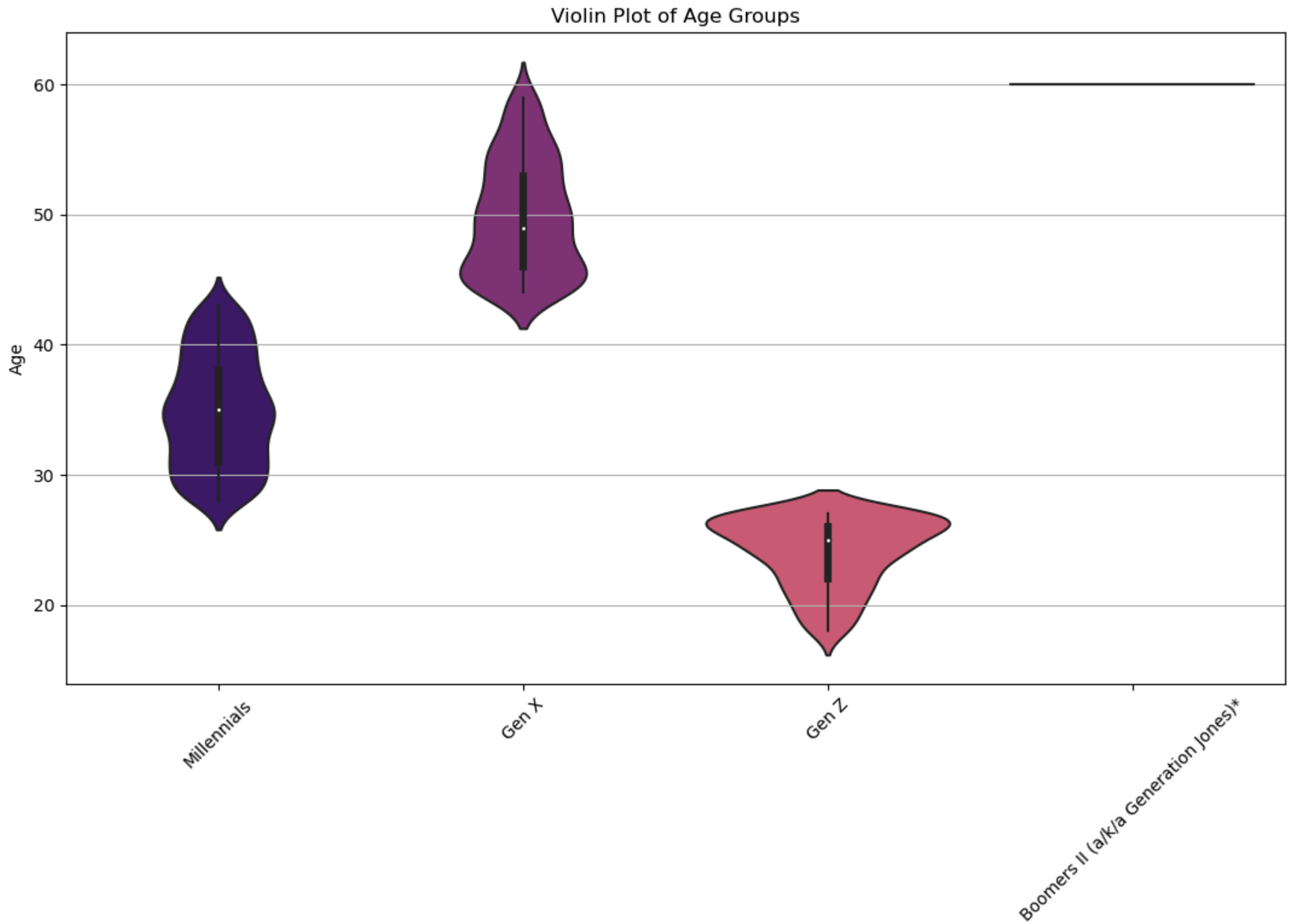
In [127…

```
plt.figure(figsize=(13, 7))
sns.boxplot(data=df, x='AgeGeneration', y='Age', palette='viridis')
plt.title('Box Plot of Age Groups')
plt.xlabel('Age Group')
plt.ylabel('Age')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.show()
```



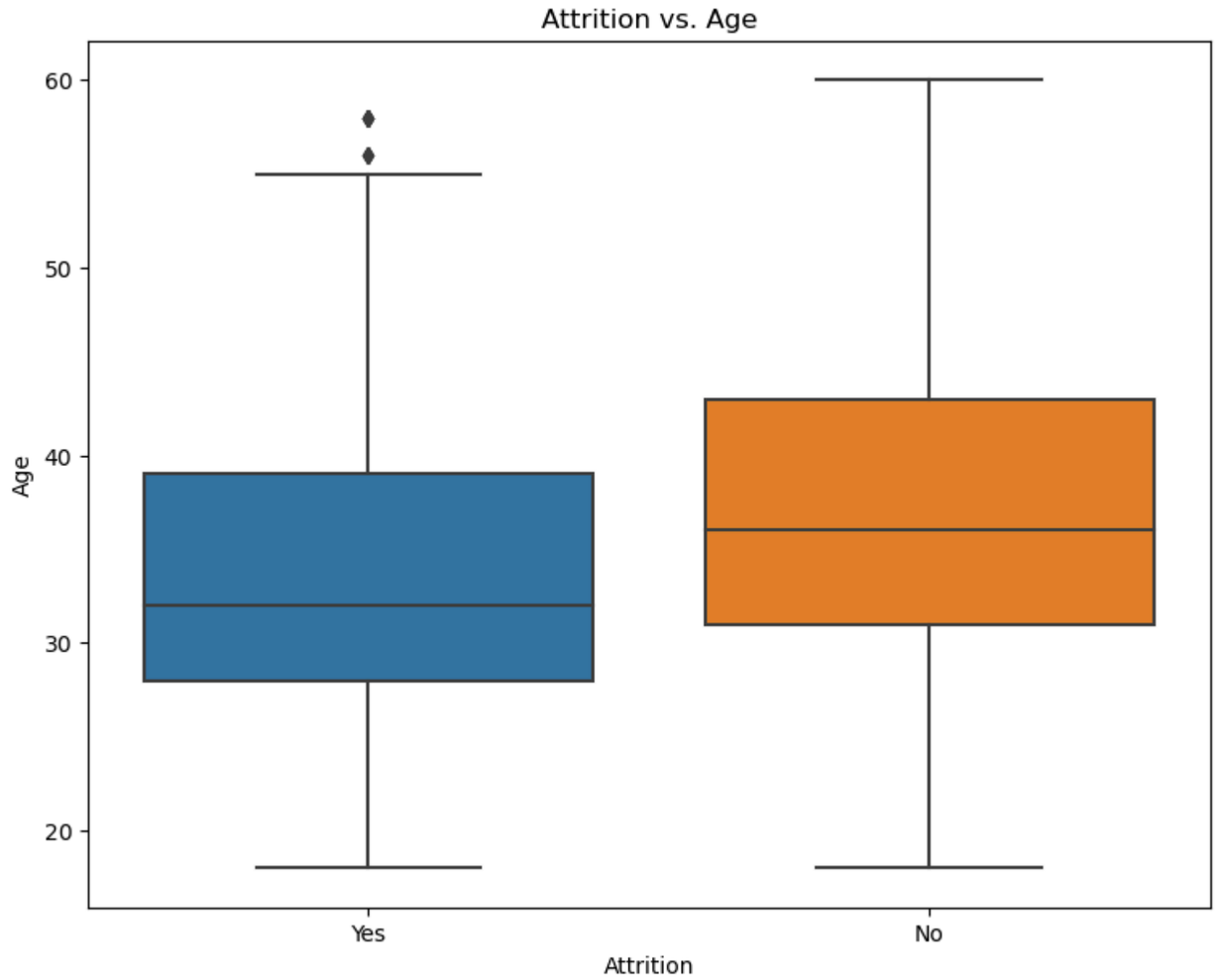
In [128…

```
plt.figure(figsize=(13, 7))
sns.violinplot(data=df, x='AgeGeneration', y='Age', palette='magma')
plt.title('Violin Plot of Age Groups')
plt.xlabel('Age Group')
plt.ylabel('Age')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.show()
```



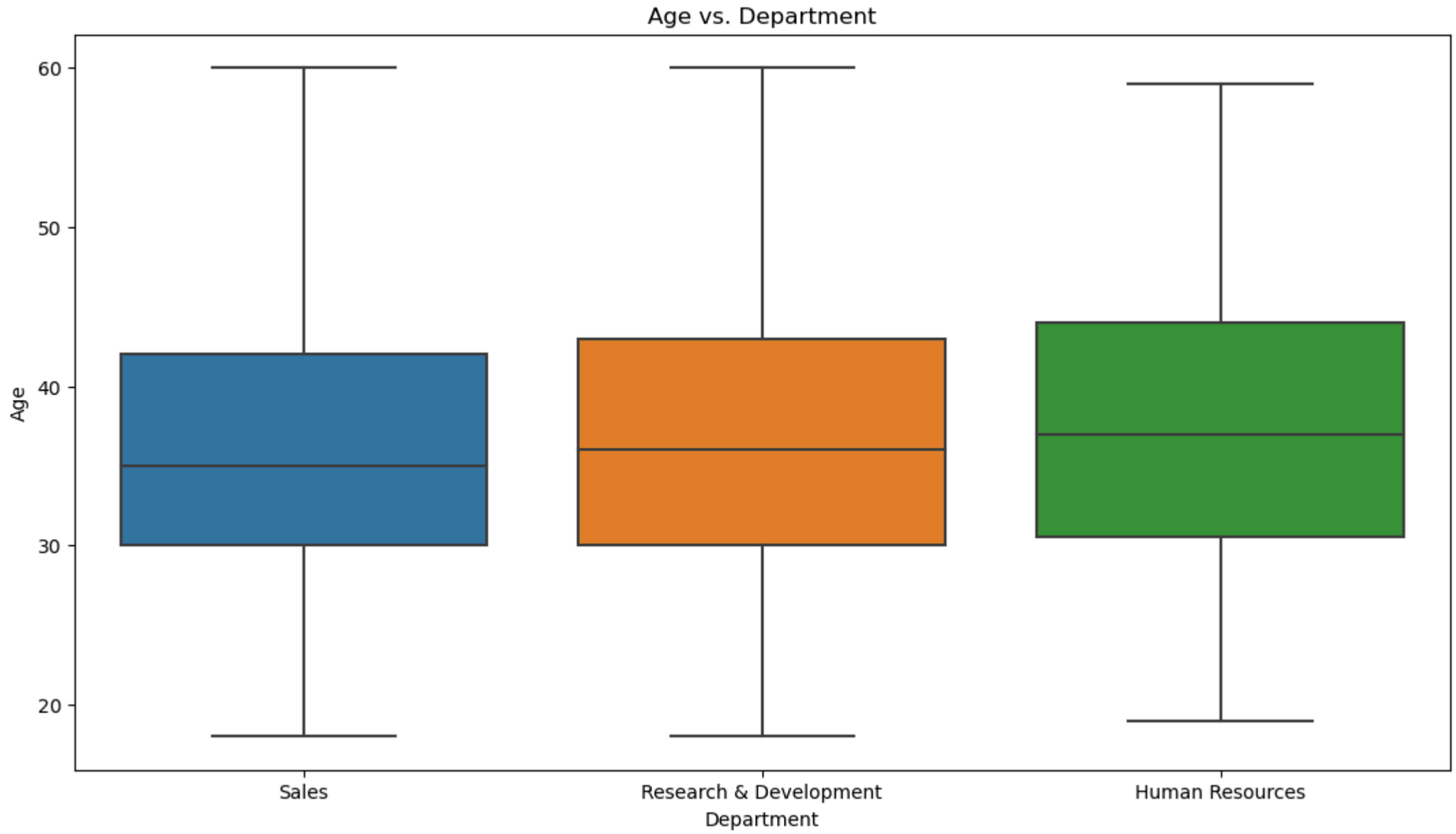
In [129…

```
plt.figure(figsize=(9, 7))
sns.boxplot(x='Attrition', y='Age', data=df)
plt.title('Attrition vs. Age')
plt.xlabel('Attrition')
plt.ylabel('Age')
plt.show()
```



In [130…

```
plt.figure(figsize=(13, 7))
sns.boxplot(data=df, x='Department', y='Age')
plt.title('Age vs. Department')
plt.xlabel('Department')
plt.ylabel('Age')
plt.show()
```

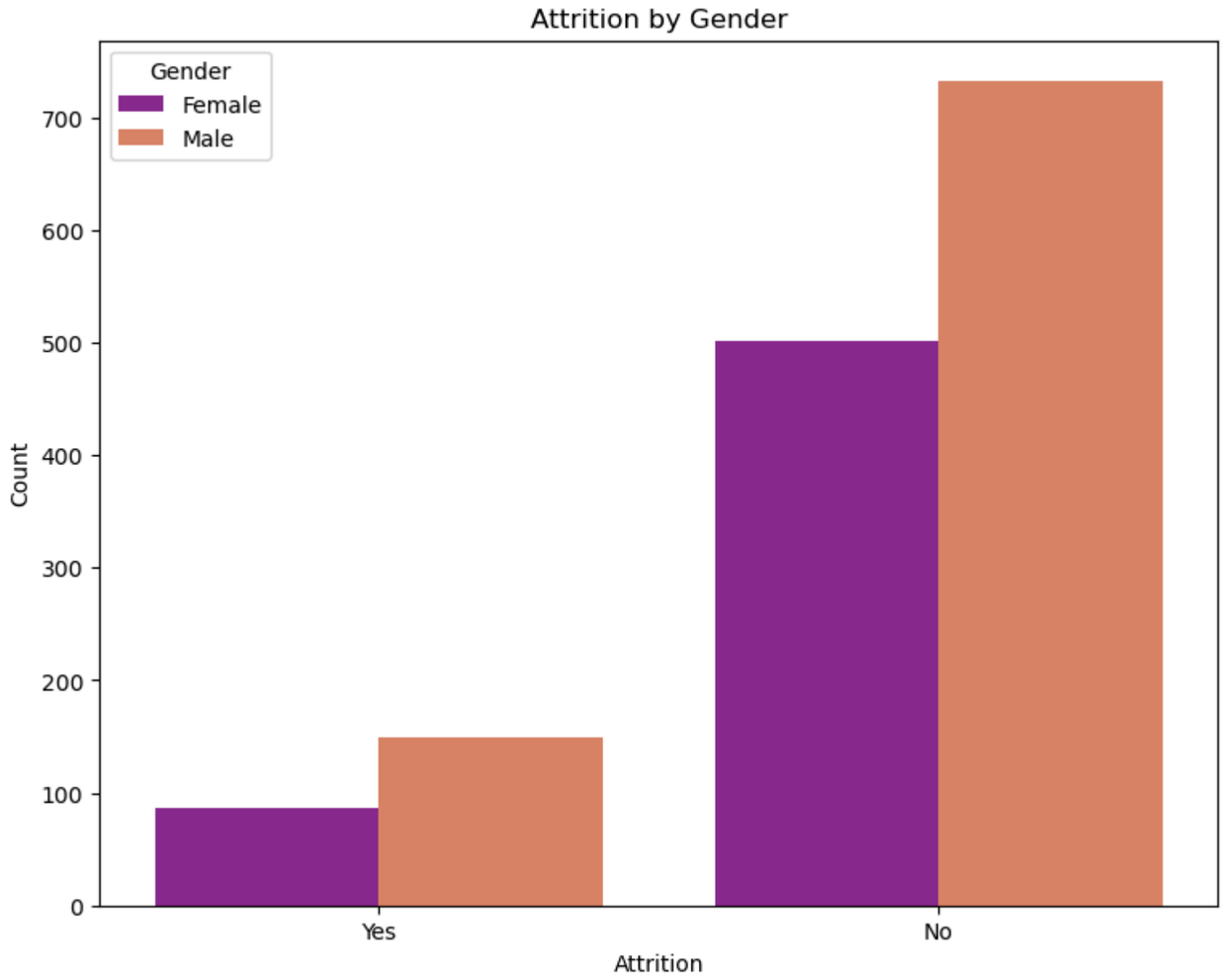


In [132…

```
plt.figure(figsize=(13, 7))
sns.scatterplot(data=df, x='MonthlyIncome', y='Age', color="orange")
plt.title('Age vs. Monthly Income')
plt.xlabel('Monthly Income')
plt.ylabel('Age')
plt.show()
```

In [133…

```
plt.figure(figsize=(9, 7))
sns.countplot(x='Attrition', hue='Gender', data=df, palette='plasma')
plt.title('Attrition by Gender')
plt.xlabel('Attrition')
plt.ylabel('Count')
plt.show()
```


Attrition by Gender
Gender
Female
Male
Count
0
100
200
300
400
500
600
700
Yes
No
Attrition