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

In [ ]:
```
df = pd.read_csv("/content/HR-Employee-Attrition.csv")
```

In [ ]:
```
# Set the maximum number of displayed columns to 40
pd.set_option('display.max_columns', 40)
```

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

Out[ ]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Educat
0	41	Yes	Travel_Rarely	1102	Sales	1	
1	49	No	Travel_Frequently	279	Research & Development	8	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	
3	33	No	Travel_Frequently	1392	Research & Development	3	
4	27	No	Travel_Rarely	591	Research & Development	2	

In [ ]:
```
df.tail()
```

Out[ ]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Ed
1465	36	No	Travel_Frequently	884	Research & Development	23	
1466	39	No	Travel_Rarely	613	Research & Development	6	
1467	27	No	Travel_Rarely	155	Research & Development	4	
1468	49	No	Travel_Frequently	1023	Sales	2	
1469	34	No	Travel_Rarely	628	Research & Development	8	

In [ ]:
```
df.sample()
```

Out[ ]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Edu
106	50	No	Travel_Frequently	1115	Research & Development	1	

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

Out[ ]:
```
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 [ ]:
```
df.shape
```

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

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

```
Out[ ]: 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 [ ]: df['Attrition']=df['Attrition'].astype('category')
        df['BusinessTravel']=df['BusinessTravel'].astype('category')
        df['Department']=df['Department'].astype('category')
        df['EducationField']=df['EducationField'].astype('category')
        df['Gender']=df['Gender'].astype('category')
```

```
In [ ]: df.info()
```

```
<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   category
 2   BusinessTravel            1470 non-null   category
 3   DailyRate                 1470 non-null   int64
 4   Department                1470 non-null   category
 5   DistanceFromHome          1470 non-null   int64
 6   Education                 1470 non-null   int64
 7   EducationField            1470 non-null   category
 8   EmployeeCount             1470 non-null   int64
 9   EmployeeNumber            1470 non-null   int64
 10  EnvironmentSatisfaction   1470 non-null   int64
 11  Gender                    1470 non-null   category
 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: category(5), int64(26), object(4)
memory usage: 352.5+ KB
```

In [ ]:
```
df['BusinessTravel'].unique()
```

Out[ ]:
```
array(['Travel_Rarely', 'Travel_Frequently', 'Non-Travel'], dtype=object)
```

In [ ]:
```
df['Department'].unique()
```

Out[ ]:
```
array(['Sales', 'Research & Development', 'Human Resources'], dtype=object)
```

In [ ]:
```
df['JobRole'].unique()
```

Out[ ]:
```
array(['Sales Executive', 'Research Scientist', 'Laboratory Technician',
       'Manufacturing Director', 'Healthcare Representative', 'Manager',
       'Sales Representative', 'Research Director', 'Human Resources'],
      dtype=object)
```

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

Out[ ]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	Em
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	
mean	36.923810	802.485714	9.192517	2.912925	1.0	
std	9.135373	403.509100	8.106864	1.024165	0.0	
min	18.000000	102.000000	1.000000	1.000000	1.0	
25%	30.000000	465.000000	2.000000	2.000000	1.0	
50%	36.000000	802.000000	7.000000	3.000000	1.0	
75%	43.000000	1157.000000	14.000000	4.000000	1.0	
max	60.000000	1499.000000	29.000000	5.000000	1.0	

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

Out[ ]:
```
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 [ ]:
```
df.duplicated().sum()
```

```
Out[ ]: 0
```

Overall attrition rate

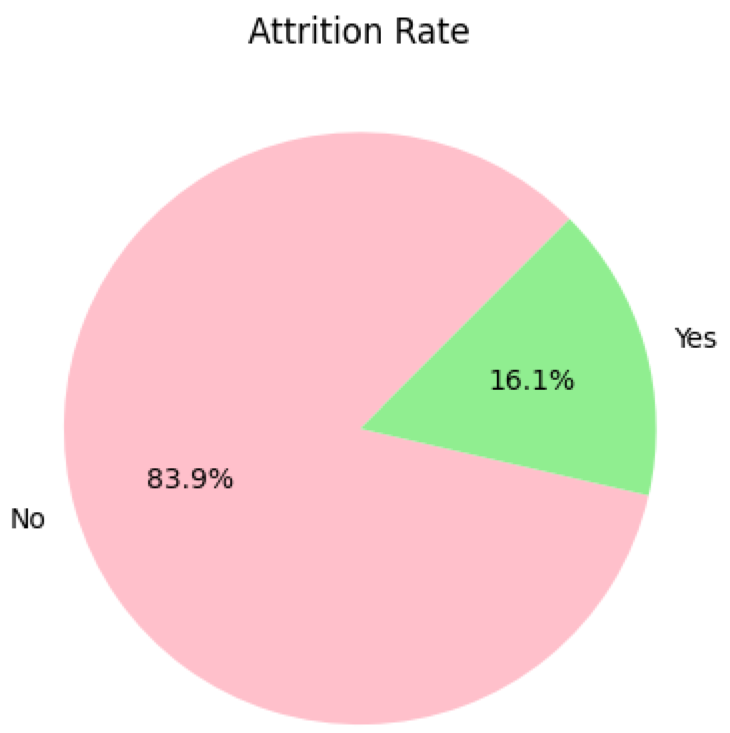
```
In [ ]: attrition_rate=((df[df['Attrition']=='Yes'].count()[0])/df.shape[0])*100
        attrition_rate
```

```
Out[ ]: 16.122448979591837
```

```
In [ ]: attrition_counts = df['Attrition'].value_counts()
        attrition_counts
```

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

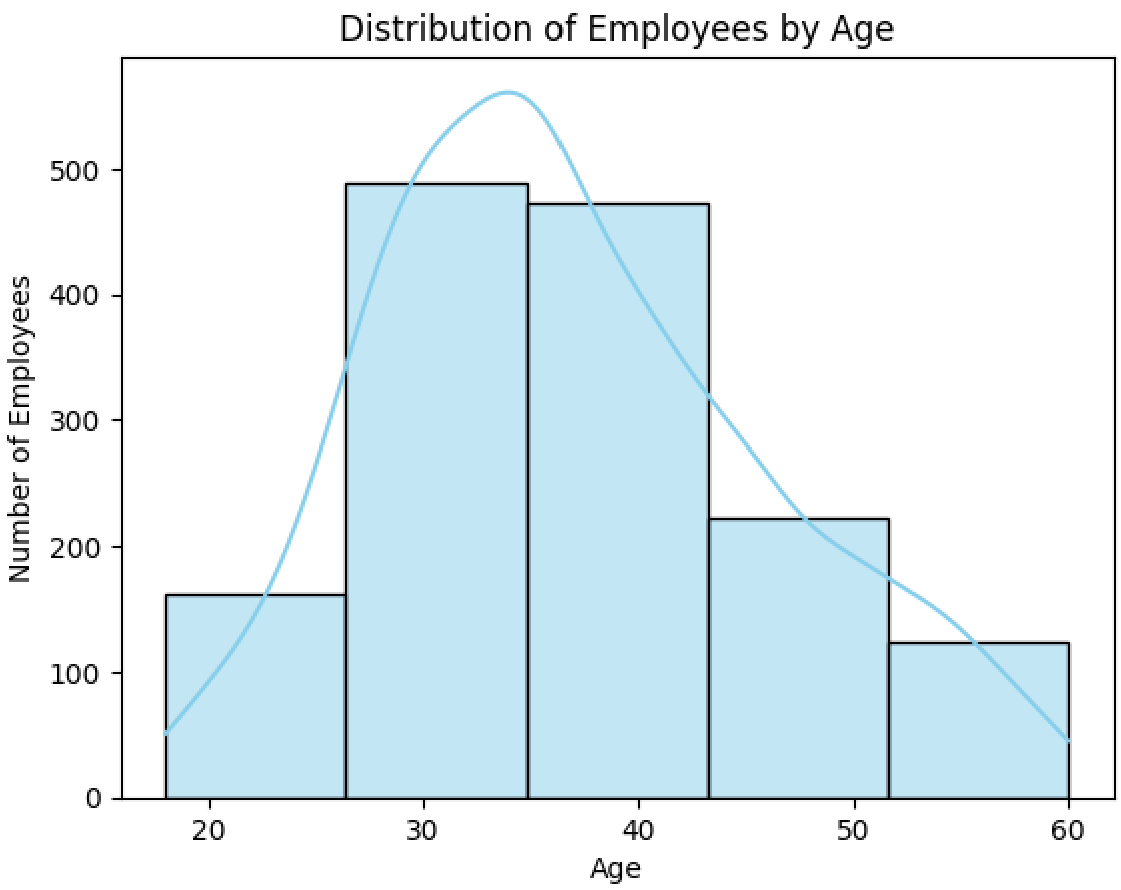
```
In [ ]: plt.pie(attrition_counts, labels=attrition_counts.index,autopct='%1.1f%%', start
        plt.title('Attrition Rate')
        plt.show()
```



Age Factor

Age Diversity

```
In [ ]: sns.histplot(data= df, x= 'Age', bins= 5, color= 'skyblue', kde=True)
        plt.ylabel('Number of Employees')
        plt.title('Distribution of Employees by Age')
        plt.show()
```



Impact of age on Attrition

In [ ]:
```
bins= [10,30,50,np.inf]
values= ['Young', 'Adult', 'Senile']
df['Age_category']= pd.cut(df['Age'], bins=bins,labels=values)
```

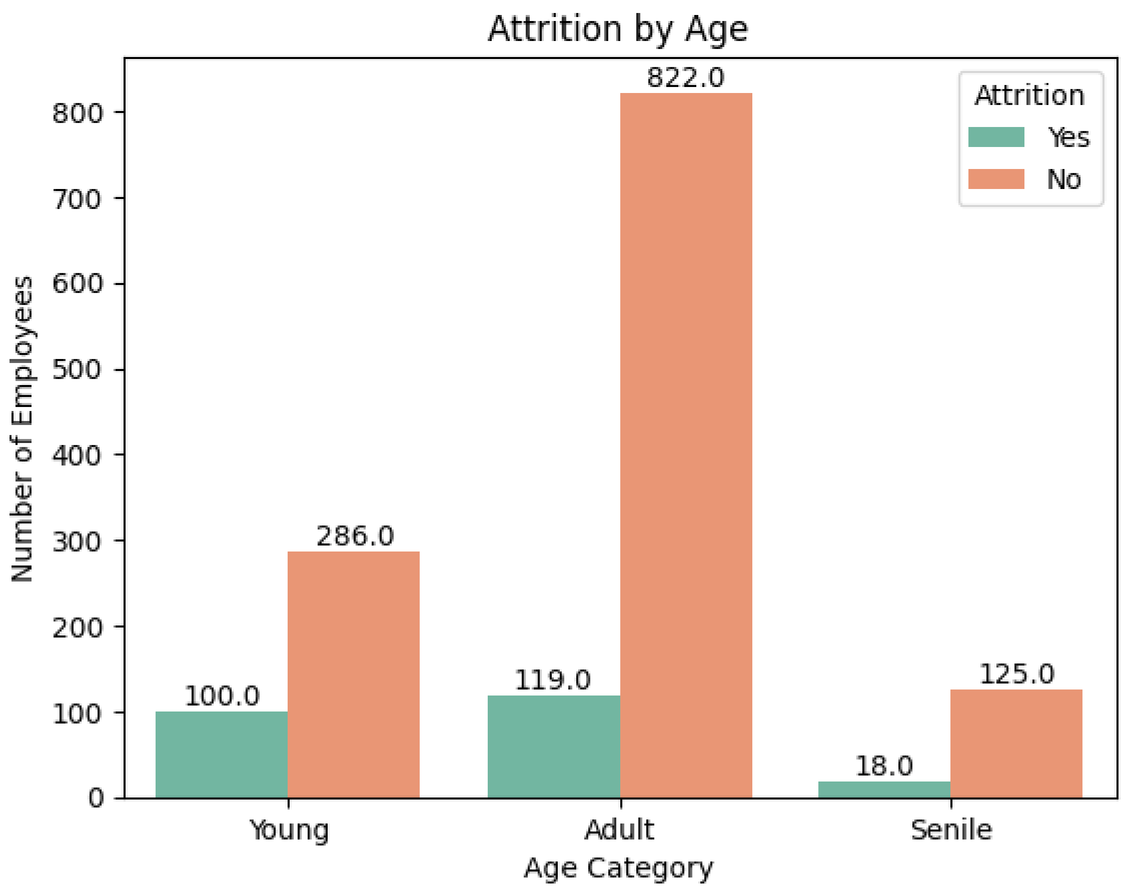
In [ ]:
```
attrition_age=df.pivot_table(index='Age_category',columns='Attrition',values='Em
attrition_age
```

Out[ ]:

Attrition	No	Yes
Age_category		
Young	286	100
Adult	822	119
Senile	125	18

In [ ]:
```
m=sns.countplot(data= df,x= 'Age_category', hue='Attrition',palette='Set2')
plt.xticks(rotation=0)
plt.xlabel('Age Category')
plt.ylabel('Number of Employees')
plt.title('Attrition by Age')

for p in m.patches:
    height = p.get_height()
    m.annotate(f'{height}', (p.get_x() + p.get_width() / 2, height),
                ha='center', va='bottom', fontsize=10)
plt.show()
```



Employee Distribution by department

```
In [ ]: df['Department'].value_counts()
```

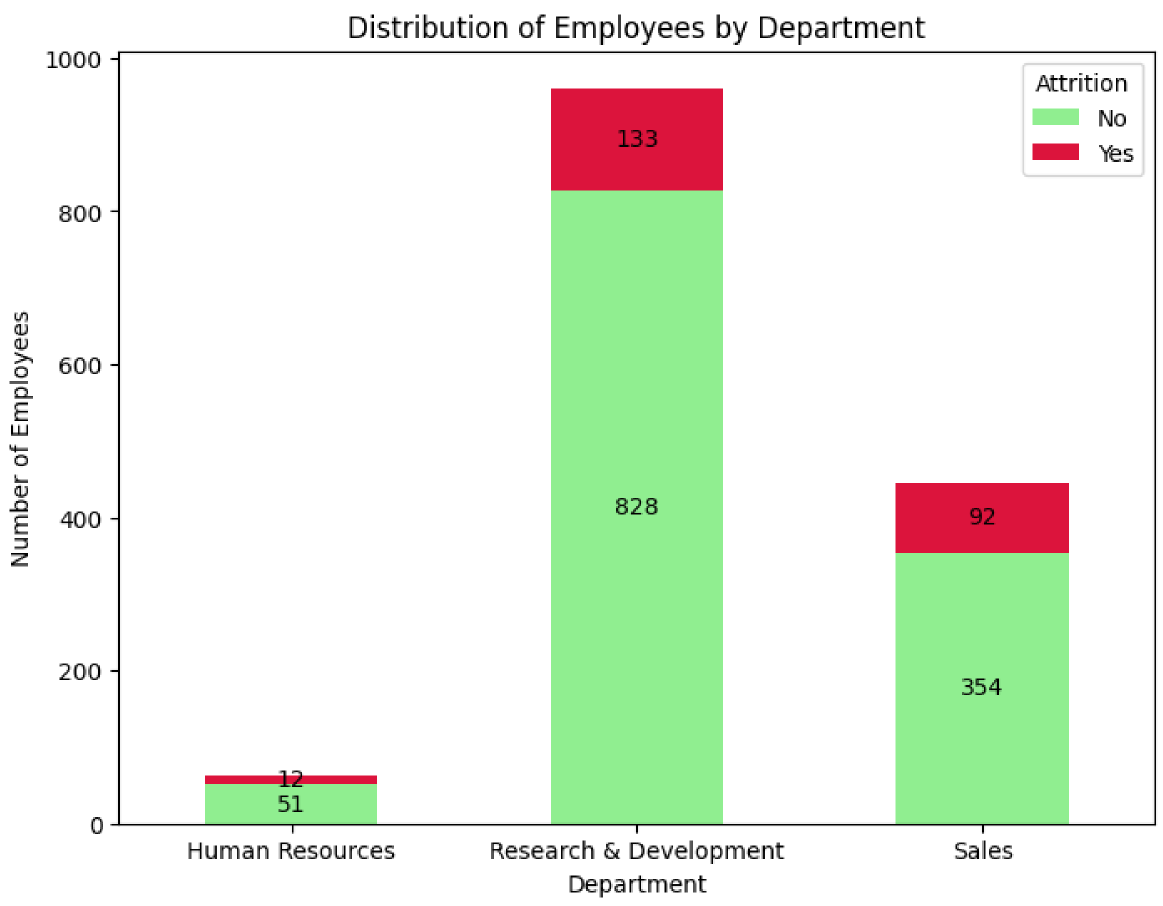
```
Out[ ]: Research & Development    961
        Sales                     446
        Human Resources            63
        Name: Department, dtype: int64
```

```
In [ ]: attrition_by_department = df.groupby(['Department', 'Attrition']).size().unstack

        # Create a stacked bar chart
        ax = attrition_by_department.plot(kind='bar', stacked=True, figsize=(8, 6),color
        plt.legend(title='Attrition', loc='upper right')
        plt.xticks(rotation=0)
        plt.xlabel('Department')
        plt.ylabel('Number of Employees')
        plt.title('Distribution of Employees by Department')
        m.bar_label(m.containers[0], fontsize=10)


        for p in ax.patches:
            width, height = p.get_width(), p.get_height()
            x, y = p.get_xy()
            ax.annotate(f'{int(height)}', (x + width/2, y + height/2), ha='center', va='

        plt.show()
```



```
In [ ]: #Percentage attrition departmentwise
        department_groups = df.groupby('Department')
        total_employees_by_department = department_groups.size().reset_index(name='Total
        attrition_count_by_department = department_groups['Attrition'].apply(lambda x: (
        attrition_percentage_by_department = pd.merge(total_employees_by_department, att
        attrition_percentage_by_department['AttritionPercentage'] = (attrition_percentag
        print(attrition_percentage_by_department)
```

```
                 Department  TotalEmployees  AttritionCount  AttritionPercentage
0           Human Resources              63              12            19.047619
1    Research & Development             961             133            13.839750
2                     Sales             446              92            20.627803
```

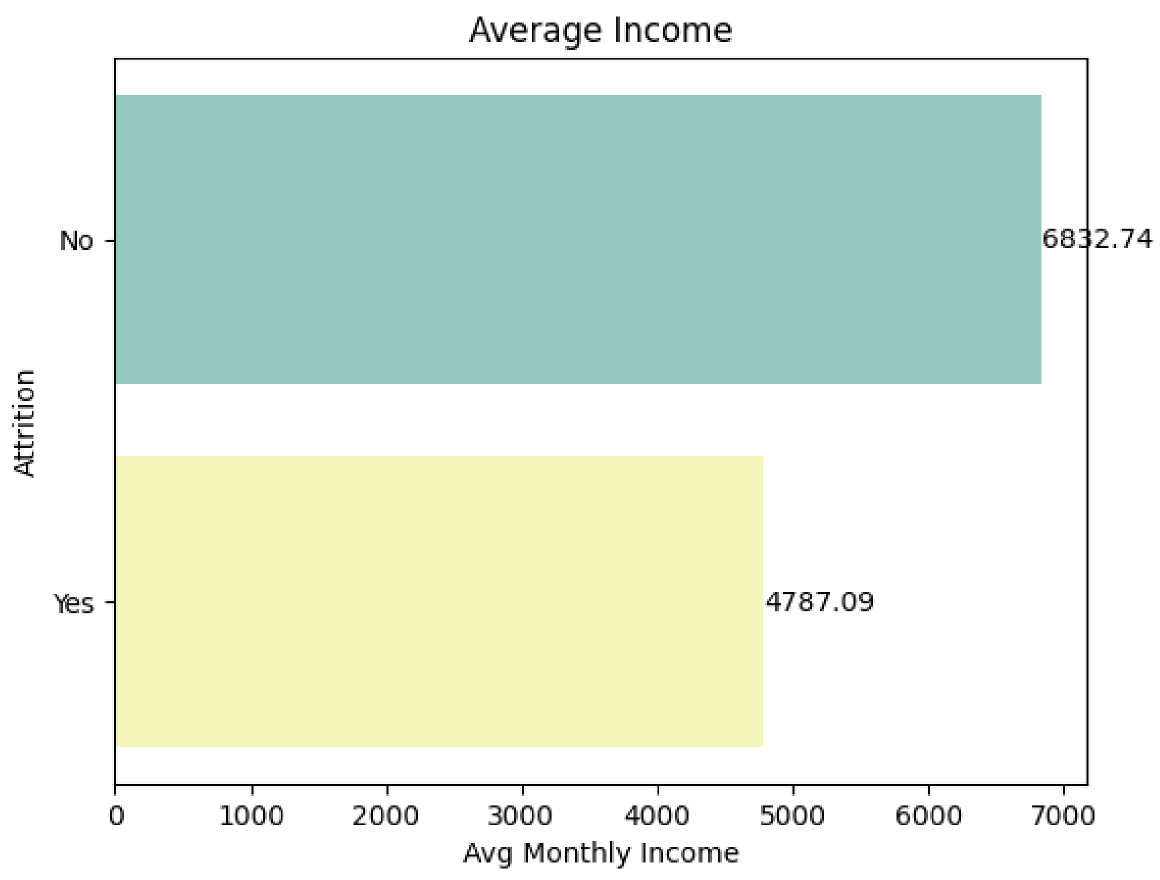
Conclusion:There are varying levels of attrition across departments, with Sales and Human Resources experiencing higher attrition rates compared to Research & Development. The Sales department experiences a higher attrition rate, with approximately 20.63% of employees leaving.The Research & Development department has a relatively lower attrition rate, with approximately 13.84% of employees leaving. This department seems to have better employee retention compared to Human Resources and Sales.

Effect of income on attrition

```
In [ ]: avg_income=df.groupby('Attrition')['MonthlyIncome'].mean()
        avg_income
```

```
Out[ ]: Attrition
        No     6832.739659
        Yes    4787.092827
        Name: MonthlyIncome, dtype: float64
```

```
In [ ]: p=sns.barplot(data=df, y= avg_income.index, x=avg_income.values,errorbar=None,pa
        p.bar_label(p.containers[0], fontsize=10)
        plt.title('Average Income')
        plt.xlabel('Avg Monthly Income')
        plt.show()
```

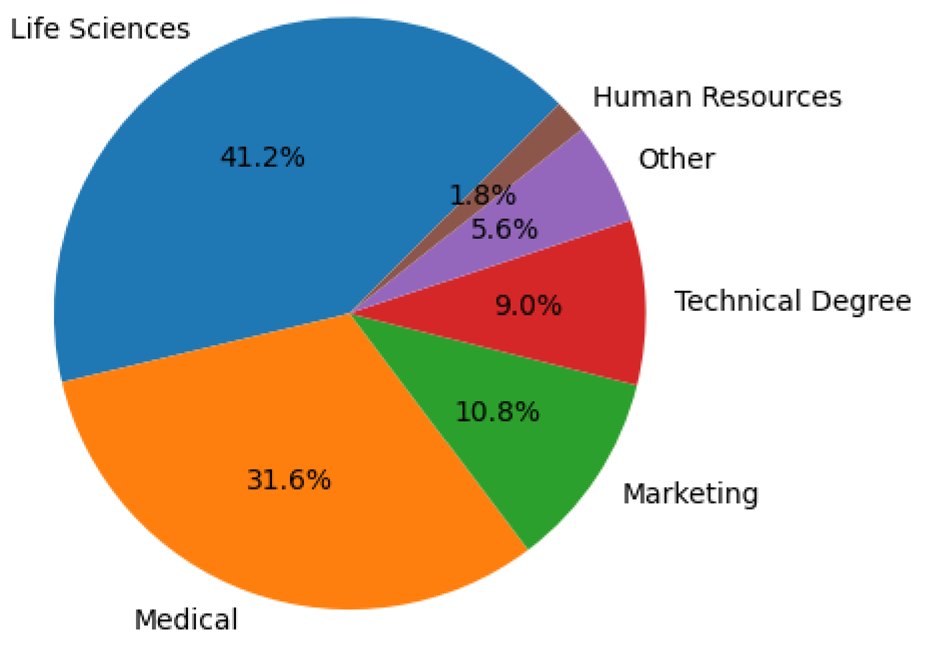


Education levels of employees

In [ ]:
```
edu_dist=df['EducationField'].value_counts()
edu_dist
```

Out[ ]:
```
Life Sciences       606
Medical             464
Marketing           159
Technical Degree    132
Other                82
Human Resources      27
Name: EducationField, dtype: int64
```

In [ ]:
```
plt.pie(edu_dist, labels=edu_dist.index,autopct='%1.1f%%', startangle=45)
plt.show()
```

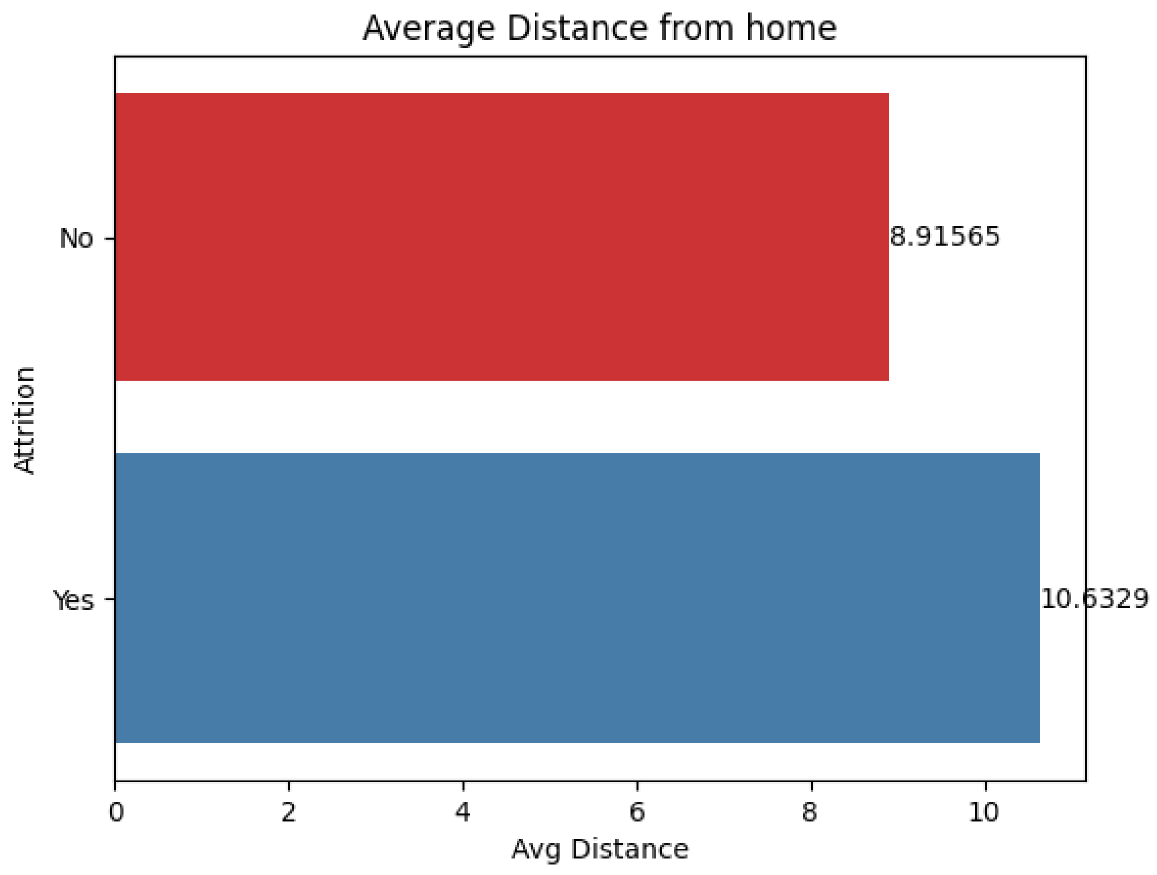


Location Proximity

```
In [ ]: avg_distance=df.groupby('Attrition')['DistanceFromHome'].mean()
        avg_distance
```

```
Out[ ]: Attrition
        No      8.915653
        Yes    10.632911
        Name: DistanceFromHome, dtype: float64
```

```
In [ ]: p=sns.barplot(data=df, y= avg_distance.index, x=avg_distance.values,errorbar=Non
        p.bar_label(p.containers[0], fontsize=10)
        plt.title('Average Distance from home')
        plt.xlabel('Avg Distance')
        plt.show()
```



Effect of overtime on employee attrition

In [ ]:
```
total_employees_with_overtime = df[df['OverTime'] == 'Yes']['EmployeeNumber'].cc
overtime_and_left = df[(df['OverTime'] == 'Yes') & (df['Attrition'] == 'Yes')]['
percentage_left_with_overtime = (overtime_and_left / total_employees_with_overti
print(percentage_left_with_overtime)
```

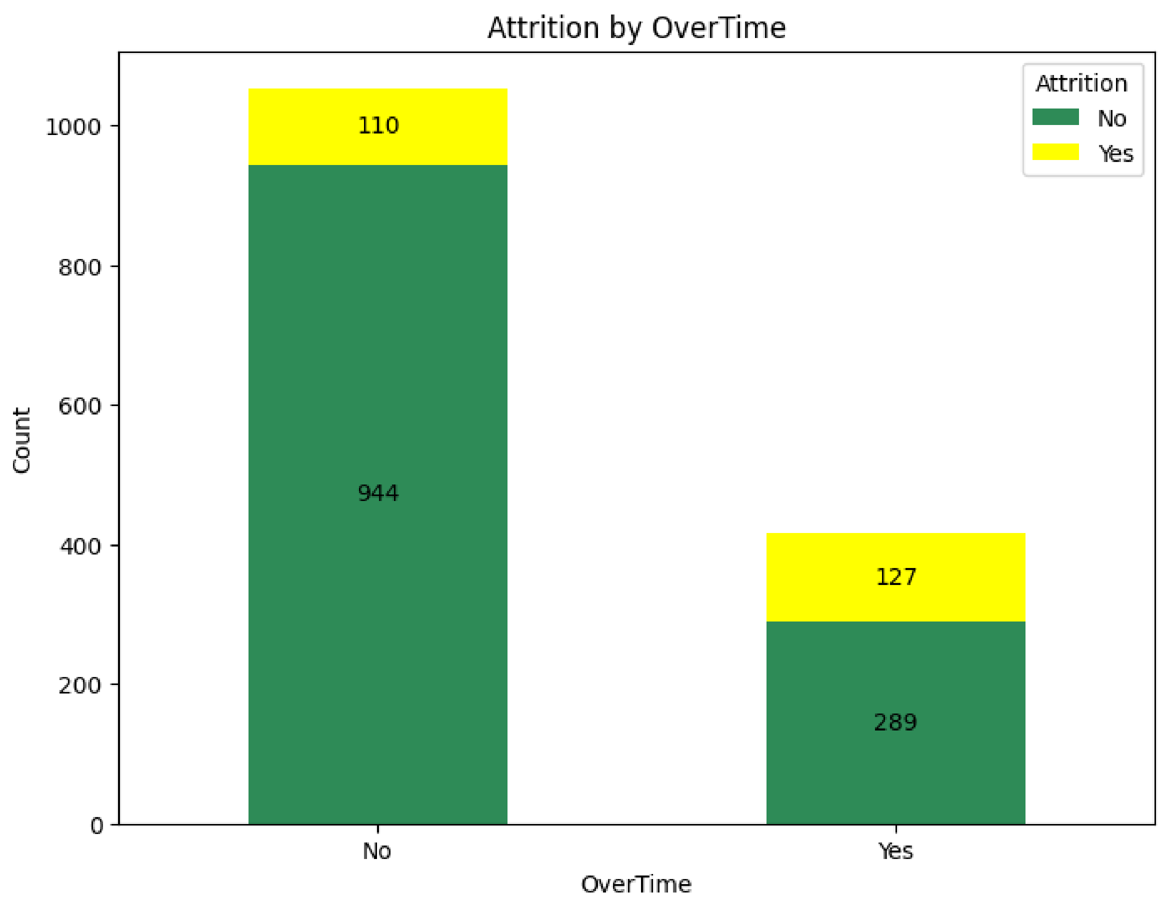
```
30.528846153846157
```

In [ ]:
```
attrition_by_overtime = df.groupby(['OverTime', 'Attrition']).size().unstack()

# Create a stacked bar chart
ax = attrition_by_overtime.plot(kind='bar', stacked=True, figsize=(8, 6),color=[
plt.title('Attrition by OverTime')
plt.xlabel('OverTime')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.legend(title='Attrition', loc='upper right')


for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x, y = p.get_xy()
    ax.annotate(f'{int(height)}', (x + width/2, y + height/2), ha='center', va='

plt.show()
```

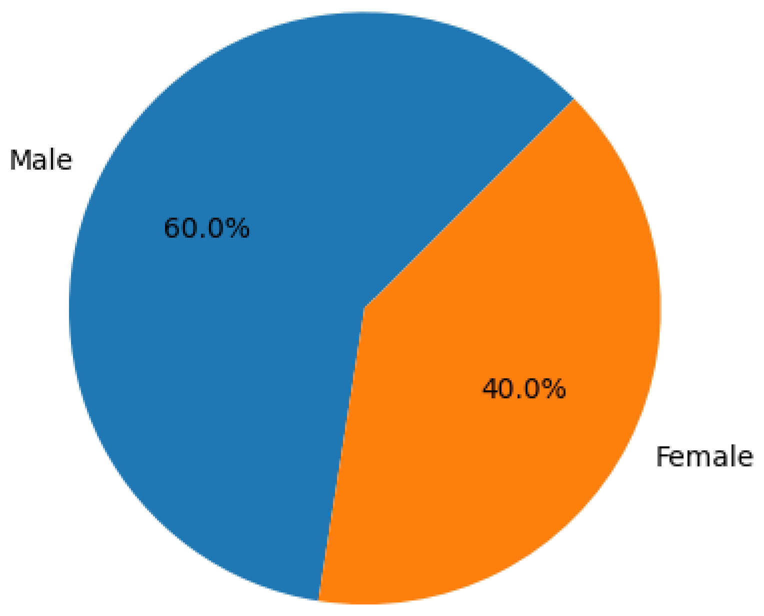


Gender Diversity and its impact on attrition

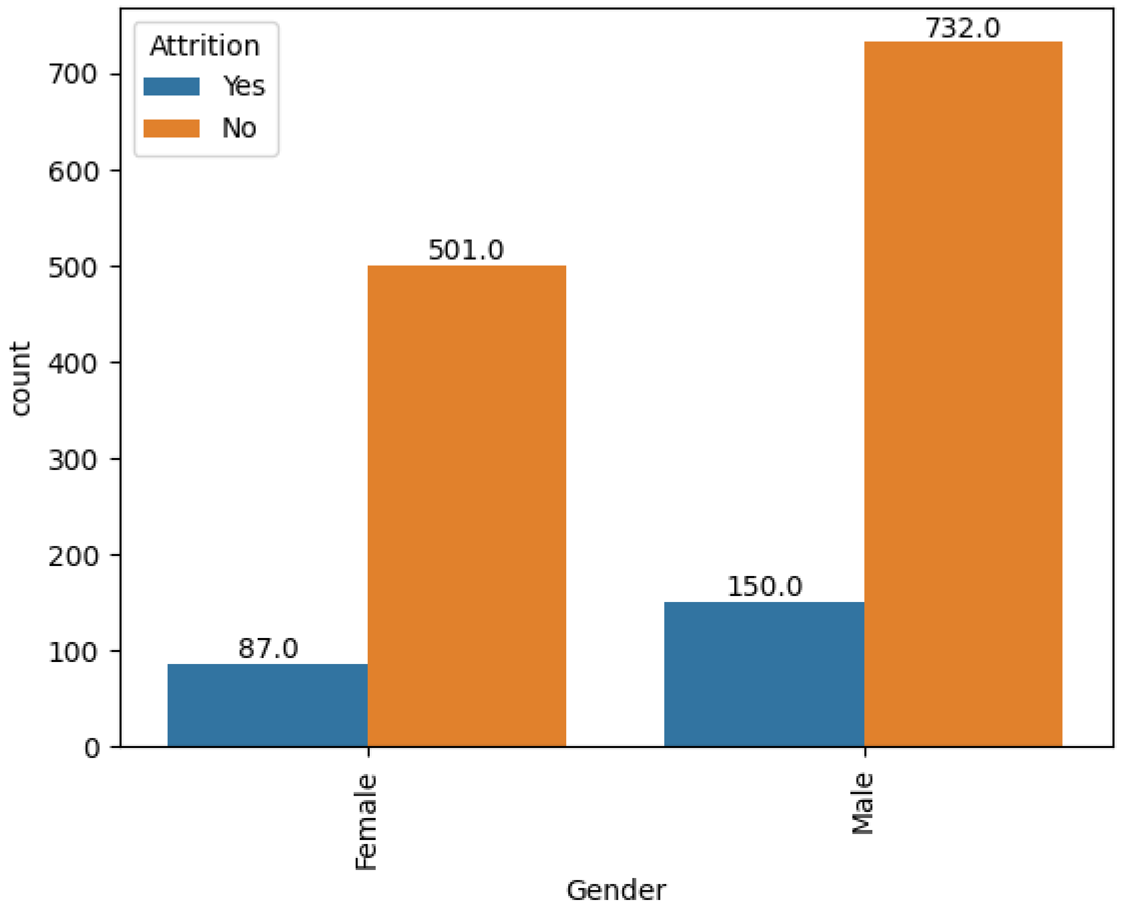
```
In [ ]: gend_dist=df['Gender'].value_counts()
        gend_dist
```

```
Out[ ]: Male      882
        Female    588
        Name: Gender, dtype: int64
```

```
In [ ]: plt.pie(gend_dist, labels=gend_dist.index,autopct='%1.1f%%', startangle=45)
        plt.show()
```



```
In [ ]: m=sns.countplot(data=df, x='Gender', hue='Attrition')
        plt.xticks(rotation=90)
        for p in m.patches:
            height = p.get_height()
            m.annotate(f'{height}', (p.get_x() + p.get_width() / 2, height),
                        ha='center', va='bottom', fontsize=10)
        plt.show()
```



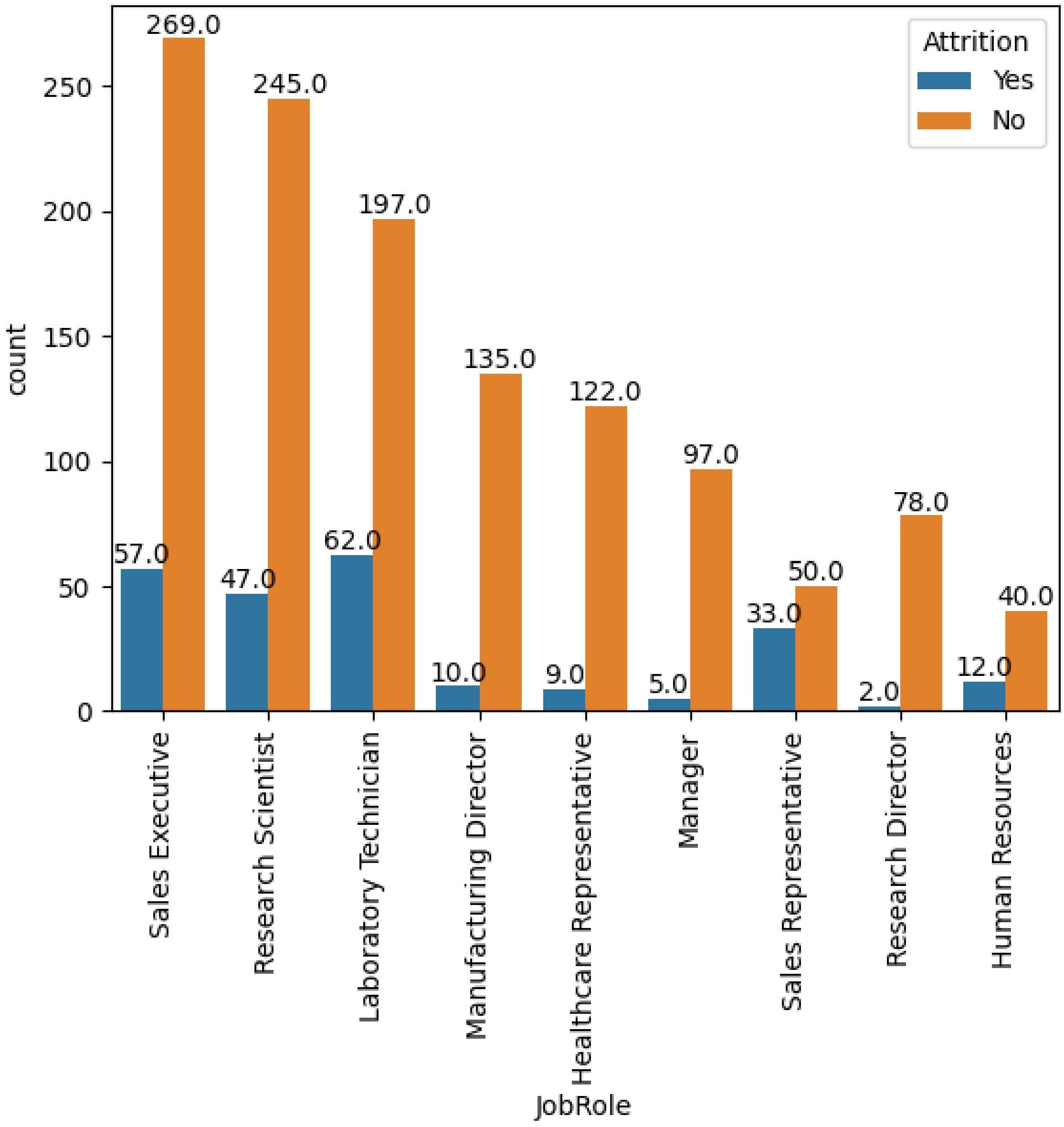
In [ ]:
```
#Percentage attrition by gender

gender_groups = df.groupby('Gender')
total_employees_by_gender = gender_groups.size().reset_index(name='TotalEmployee
attrition_count_by_gender = gender_groups['Attrition'].apply(lambda x: (x == 'Ye
attrition_percentage_by_gender = pd.merge(total_employees_by_gender, attrition_c
attrition_percentage_by_gender['AttritionPercentage'] = (attrition_percentage_by
print(attrition_percentage_by_gender)
```

```
   Gender  TotalEmployees  AttritionCount  AttritionPercentage
0  Female             588              87            14.795918
1    Male             882             150            17.006803
```

Job Profile and its impact

In [ ]:
```
jp=sns.countplot(data=df, x='JobRole', hue='Attrition')
plt.xticks(rotation=90)
for p in jp.patches:
    height = p.get_height()
    jp.annotate(f'{height}', (p.get_x() + p.get_width() / 2, height),
                ha='center', va='bottom', fontsize=10)
plt.show()
```



```
In [ ]: job_profile_groups = df.groupby('JobRole')
        total_employees_by_jobrole = job_profile_groups.size().reset_index(name='TotalEm
        attrition_count_by_jobrole = job_profile_groups['Attrition'].apply(lambda x: (x
        attrition_percentage_by_jobrole = pd.merge(total_employees_by_jobrole, attrition
        attrition_percentage_by_jobrole['AttritionPercentage'] = (attrition_percentage_b
        print(attrition_percentage_by_jobrole)
```

```
                    JobRole  TotalEmployees  AttritionCount  \
0  Healthcare Representative             131               9
1            Human Resources              52              12
2      Laboratory Technician             259              62
3                    Manager             102               5
4     Manufacturing Director             145              10
5          Research Director              80               2
6         Research Scientist             292              47
7            Sales Executive             326              57
8       Sales Representative              83              33

   AttritionPercentage
0             6.870229
1            23.076923
2            23.938224
3             4.901961
4             6.896552
5             2.500000
6            16.095890
7            17.484663
8            39.759036
```

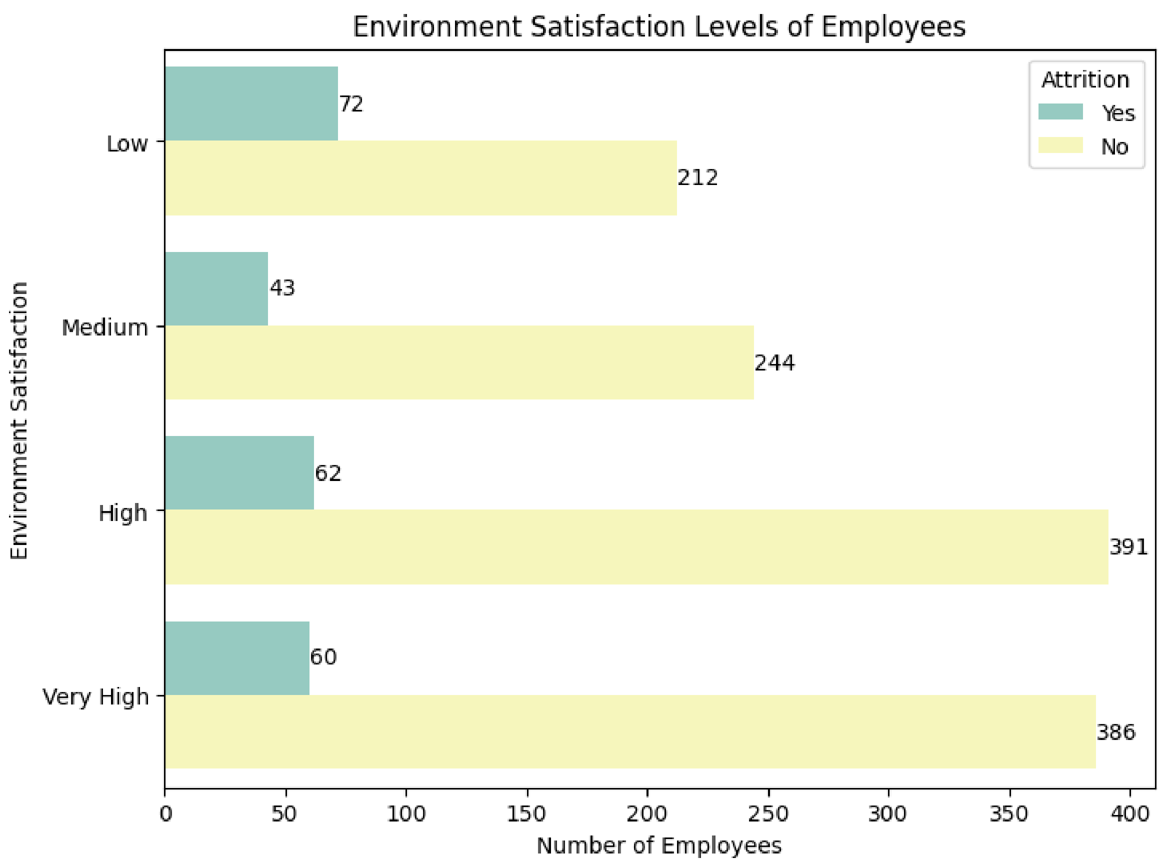
Environment Satisfaction Rating

```
In [ ]: plt.figure(figsize=(8, 6))
        env_sat = sns.countplot(data=df, y='EnvironmentSatisfaction', hue='Attrition',pa
        plt.title('Environment Satisfaction Levels of Employees')
        plt.xlabel('Number of Employees')
        plt.ylabel('Environment Satisfaction')

        for p in env_sat.patches:
            width = p.get_width()
            height = p.get_height()
            x, y = p.get_x() + width, p.get_y() + height/2
            plt.annotate(f'{int(width)}', (x, y), ha='left', va='center', fontsize=10)

        env_sat.set_yticklabels(['Low', 'Medium', 'High', 'Very High'])

        plt.show()
```



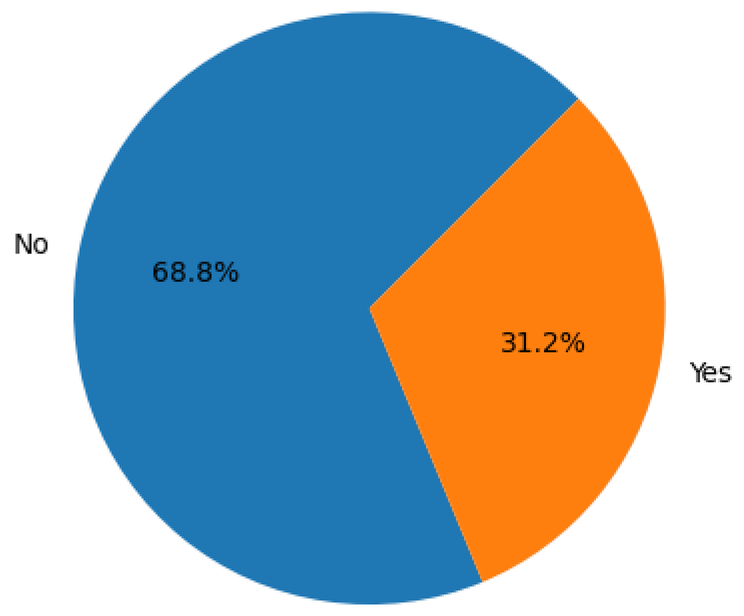
Work-Life Balance

```
In [ ]: df1= df[df['WorkLifeBalance']==1]
```

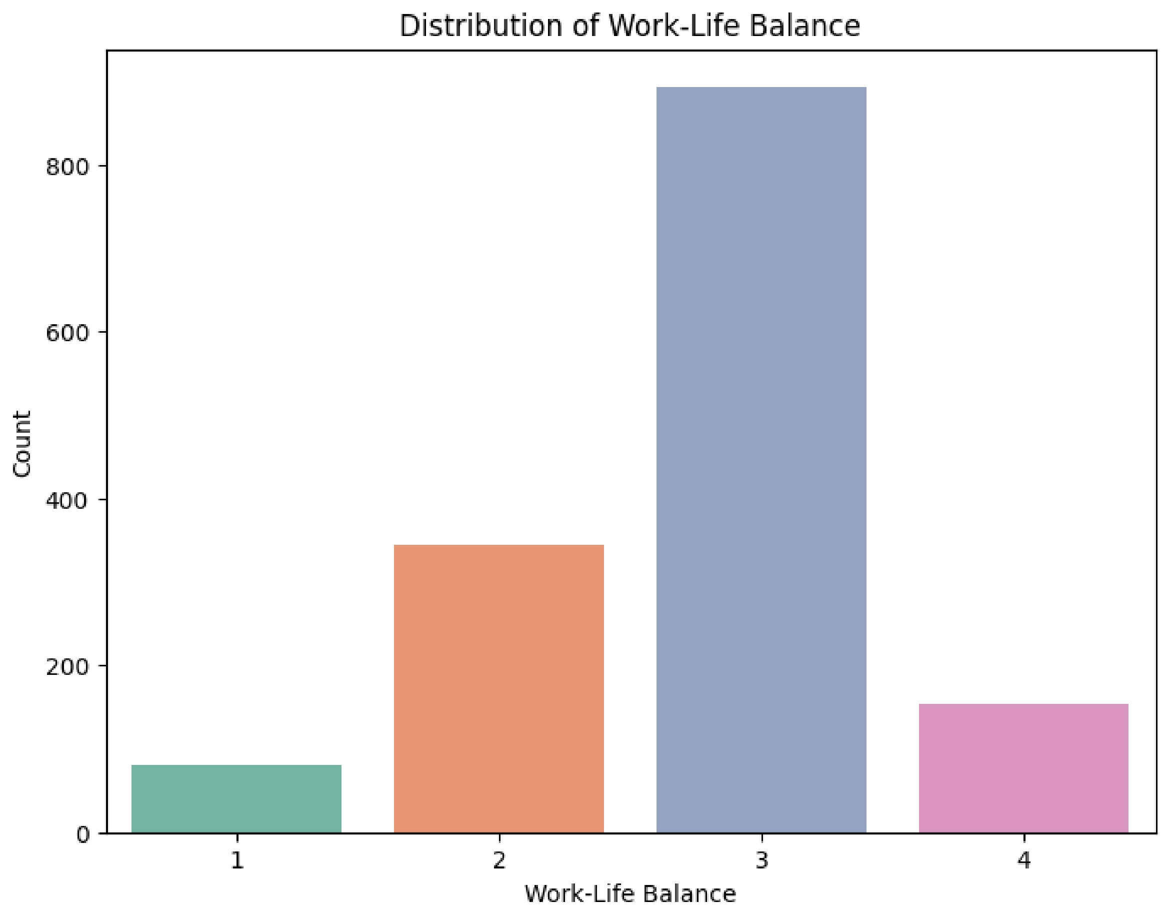
```
In [ ]: att_cnt=df1['Attrition'].value_counts()
        att_cnt
```

```
Out[ ]: No      55
        Yes     25
        Name: Attrition, dtype: int64
```

```
In [ ]: plt.pie(att_cnt, labels=att_cnt.index,autopct='%1.1f%%', startangle=45)
        plt.show()
```



```
In [ ]: plt.figure(figsize=(8, 6))
        sns.countplot(data=df, x='WorkLifeBalance', palette='Set2')
        plt.title('Distribution of Work-Life Balance')
        plt.xlabel('Work-Life Balance')
        plt.ylabel('Count')
        plt.show()
```



Business Travel

In [ ]:

```
pivot_table = pd.pivot_table(df, values='Attrition', index='BusinessTravel', agg
pivot_table.columns = ['TotalEmployees', 'AttritionCount']
pivot_table['AttritionPercentage'] = (pivot_table['AttritionCount'] / pivot_tabl
pivot_table.reset_index(inplace=True)

print(pivot_table)
```

```
      BusinessTravel  TotalEmployees  AttritionCount  AttritionPercentage
0         Non-Travel             150              12             8.000000
1  Travel_Frequently             277              69            24.909747
2      Travel_Rarely            1043             156            14.956855
```

Number of years spent in the company

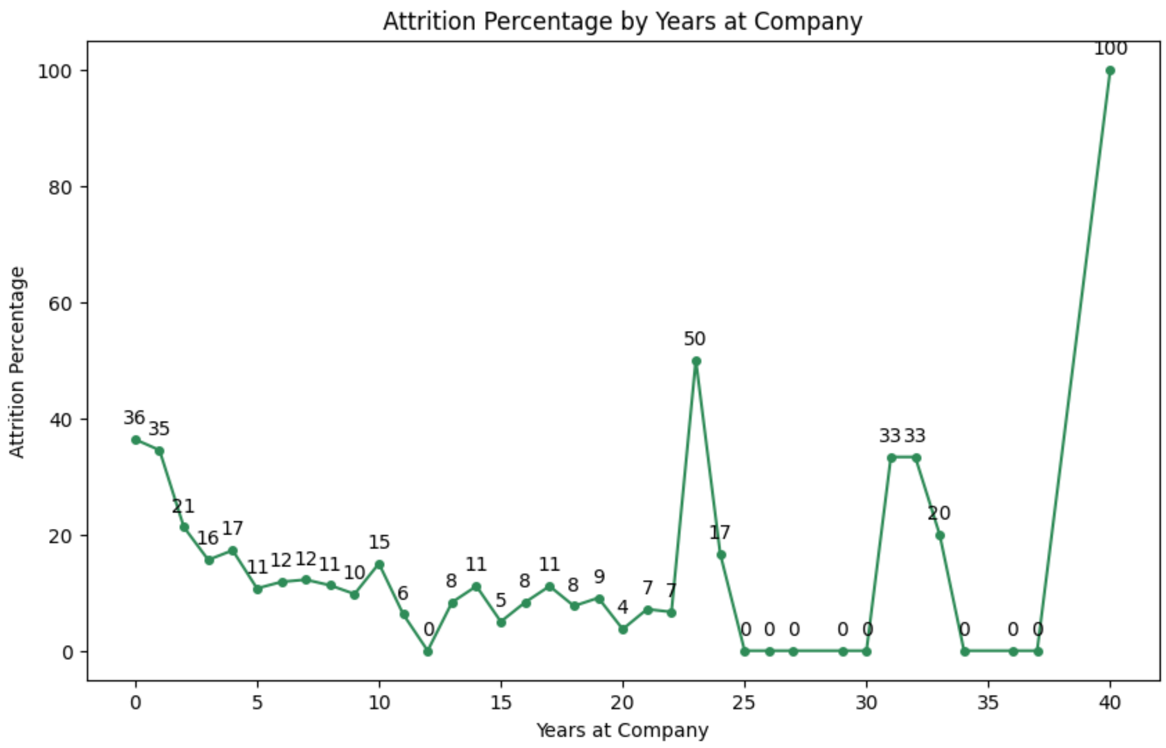
In [ ]:

```
attrition_by_years = df.groupby('YearsAtCompany')['Attrition'].apply(lambda x: (

# Create a line chart
plt.figure(figsize=(10, 6))
plt.plot(attrition_by_years.index, attrition_by_years.values, marker='o', linest
plt.title('Attrition Percentage by Years at Company')
plt.xlabel('Years at Company')
plt.ylabel('Attrition Percentage')

label_offset = 2  # Adjust this value to control label height
for x, y in zip(attrition_by_years.index, attrition_by_years.values):
    label = f'{int(round(y))}'  # Round the percentage and convert it to an inte
    plt.text(x, y + label_offset, label, ha='center', va='bottom', fontsize=10)

plt.show()
```



In [ ]:

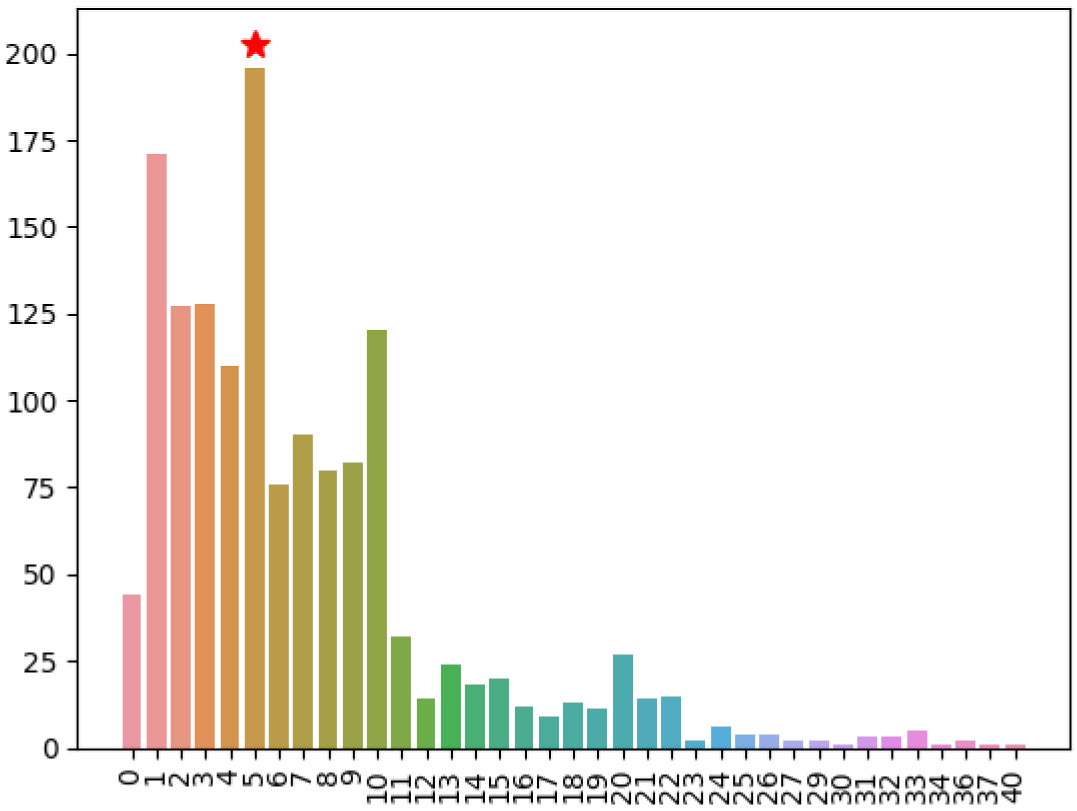
```
x = df['YearsAtCompany'].value_counts()
```

```
x
```

```
Out[ ]: 5     196
        1     171
        3     128
        2     127
        10    120
        4     110
        7      90
        9      82
        8      80
        6      76
        0      44
        11     32
        20     27
        13     24
        15     20
        14     18
        22     15
        12     14
        21     14
        18     13
        16     12
        19     11
        17      9
        24      6
        33      5
        25      4
        26      4
        31      3
        32      3
        27      2
        36      2
        29      2
        23      2
        37      1
        40      1
        34      1
        30      1
        Name: YearsAtCompany, dtype: int64
```

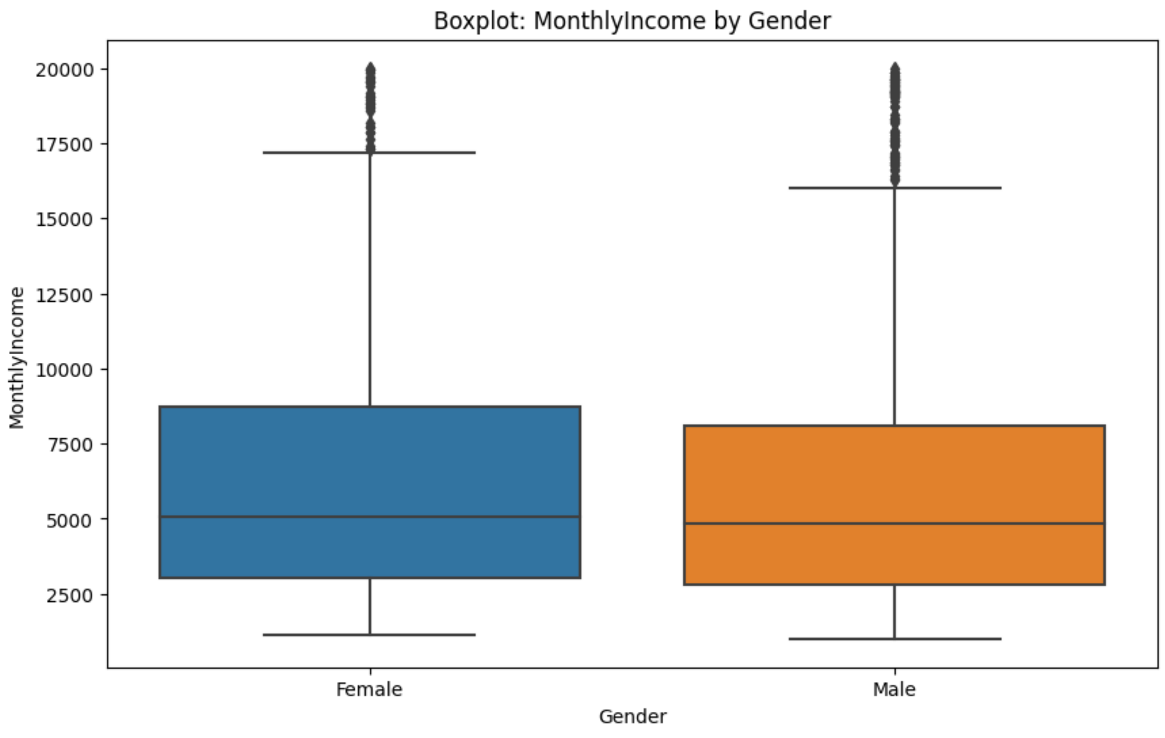
```
In [ ]: ax = sns.barplot(
             x=x.index, y=x.values,
               errorbar=None,
        )
        plt.xticks(rotation=90)

        ax.plot(5, 203, "*", markersize=10, color="r")
        plt.show()
```



Gender vs Monthly Income

```
In [ ]: plt.figure(figsize=(10, 6))
        sns.boxplot(x='Gender', y='MonthlyIncome', data=df)
        plt.title('Boxplot: MonthlyIncome by Gender')
        plt.show()
```



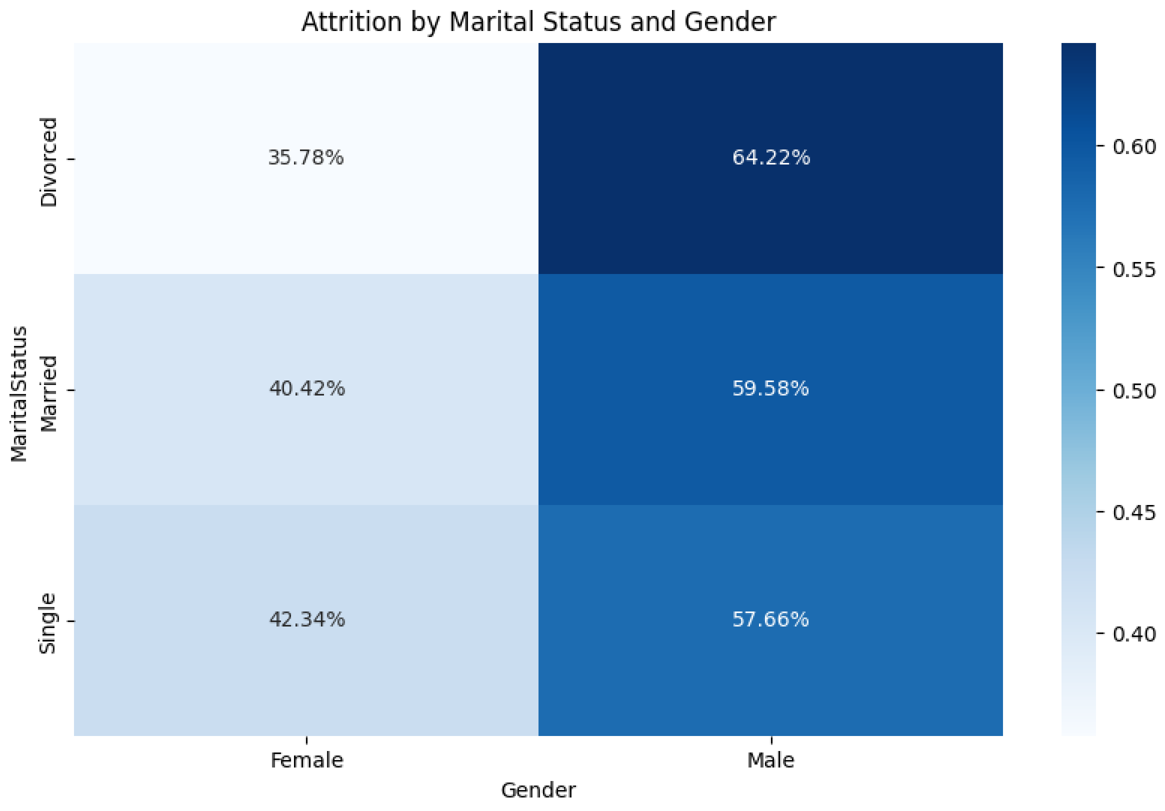
```
In [ ]: df.pivot_table(index='MaritalStatus',columns='JobInvolvement',values='EmployeeNu
```

Out[ ]:

JobInvolvement	1	2	3	4
MaritalStatus				
Divorced	22	71	200	34
Married	33	175	391	74
Single	28	129	277	36

Attrition by Marital Status and Gender

In [ ]:
```
cross_tab = pd.crosstab(df['MaritalStatus'], df['Gender'], values=df['Attrition'
plt.figure(figsize=(10, 6))
sns.heatmap(cross_tab, annot=True, cmap='Blues', fmt=".2%", cbar=True)
plt.title('Attrition by Marital Status and Gender')
plt.show()
```



Relationship satisfaction vs. Job satisfaction

In [ ]:
```
df.pivot_table(index='RelationshipSatisfaction', columns='JobSatisfaction', valu
```

Out[ ]:

JobSatisfaction	1	2	3	4
RelationshipSatisfaction				
1	54	53	85	84
2	57	57	94	95
3	91	81	133	154
4	87	89	130	126

Is there a correlation between performance rating and percent salary hike?

In [ ]:
```
df['PerformanceRating'].corr(df['PercentSalaryHike'])
```

Out[ ]:
```
0.773549996401268
```

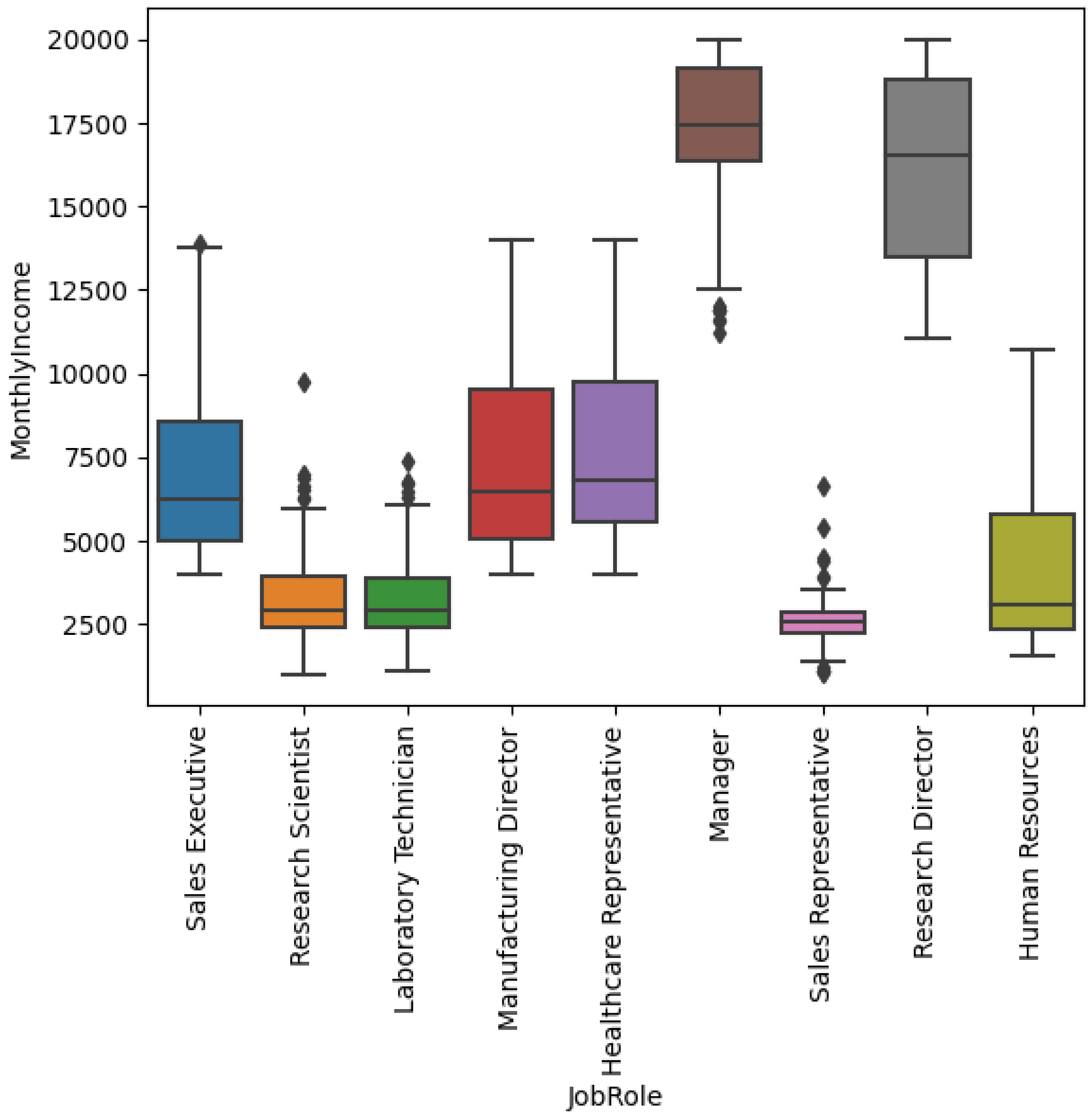
How many employees have never received a promotion?

In [ ]:
```
df[df['YearsSinceLastPromotion']==0]['EmployeeNumber'].count()
```

Out[ ]:
```
581
```

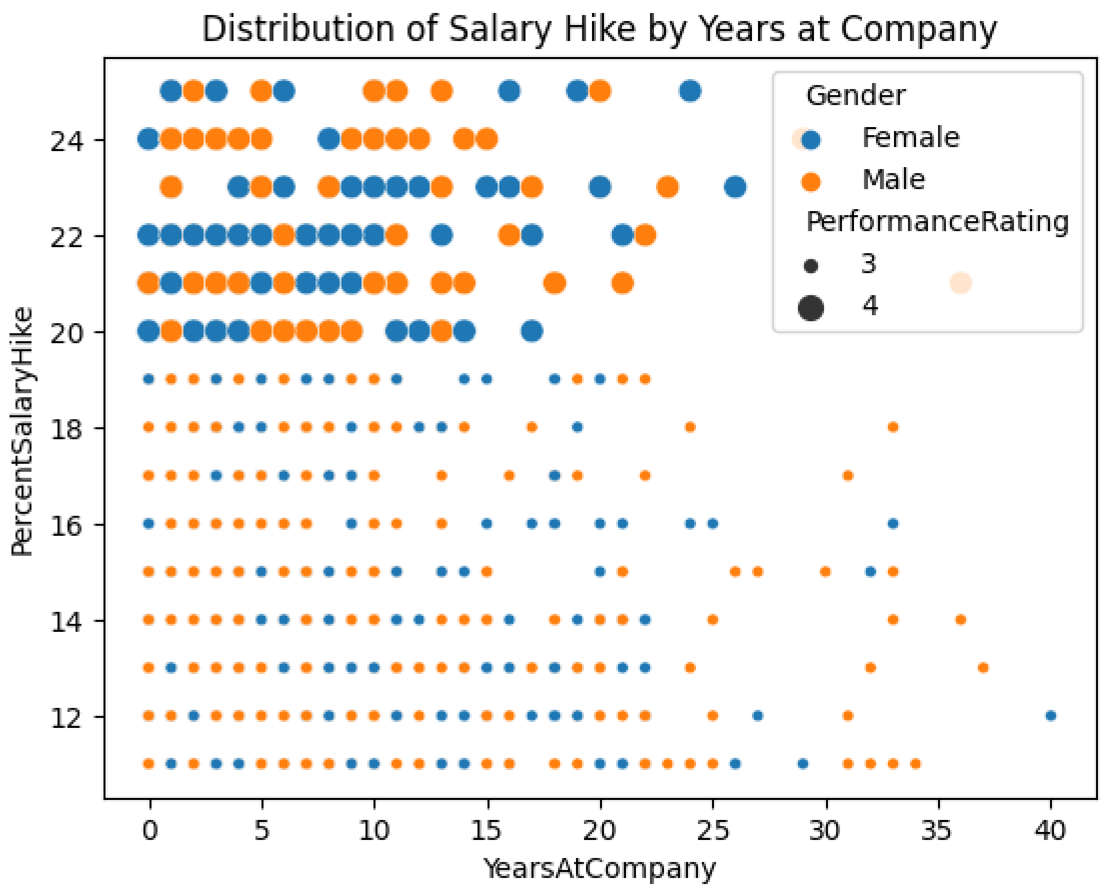
Distribution of Monthly income amongst various job roles

In [ ]:
```
sns.boxplot(data=df, x='JobRole', y='MonthlyIncome')
plt.xticks(rotation=90)
plt.show()
```



Salary hike by Performance Rating and Years spent at company

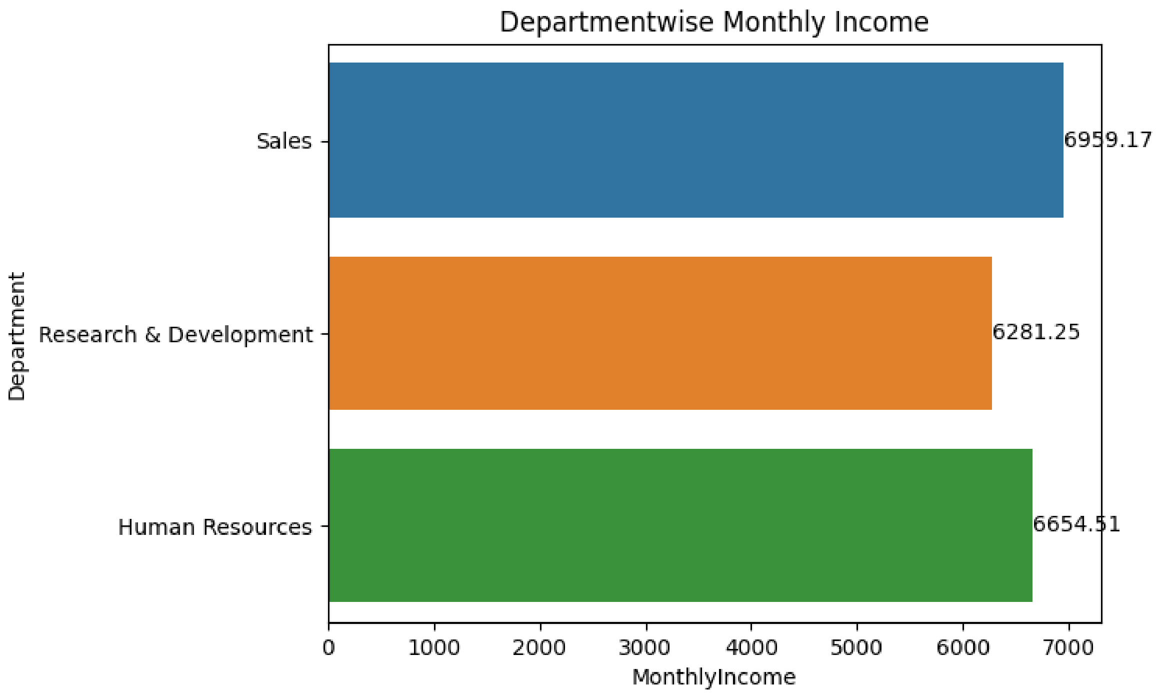
```
In [ ]: sns.scatterplot(data=df, x='YearsAtCompany', y='PercentSalaryHike', hue='Gender'
        plt.title('Distribution of Salary Hike by Years at Company')
        plt.show()
```



Departmentwise Monthly Income

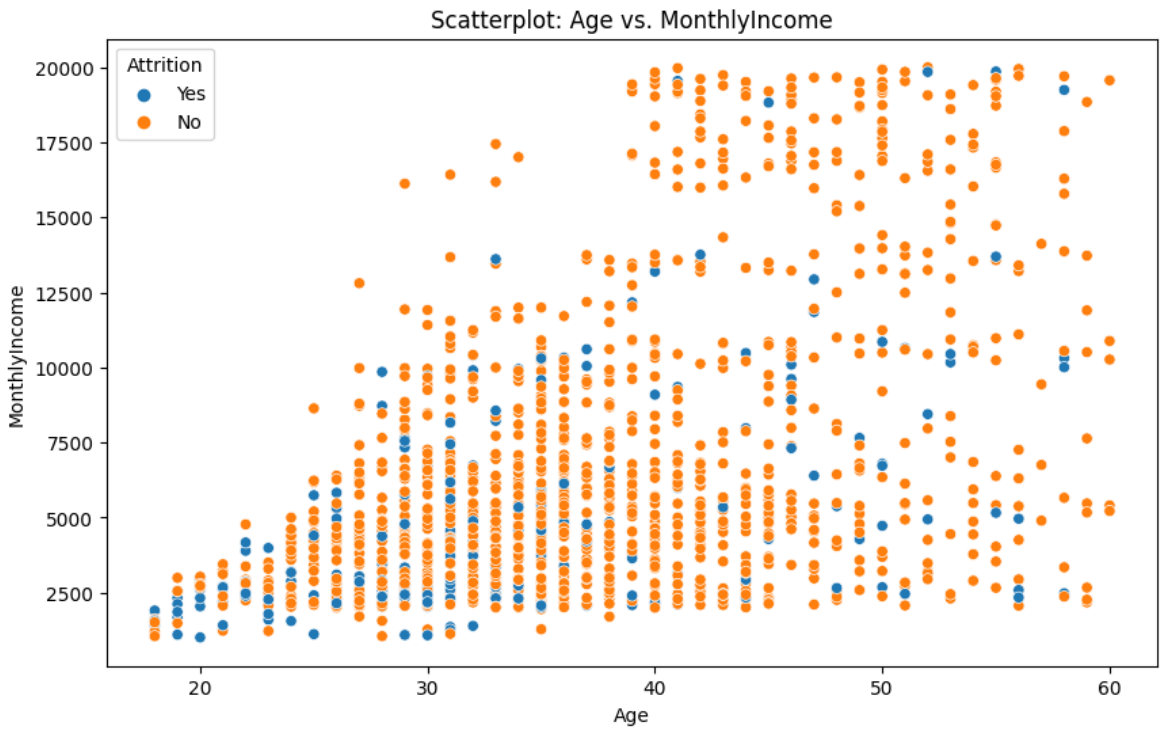
```
In [ ]: p=sns.barplot(data=df, y='Department', x='MonthlyIncome',errorbar=None)
        p.bar_label(p.containers[0], fontsize=10)
        plt.title('Departmentwise Monthly Income')
```

```
Out[ ]: Text(0.5, 1.0, 'Departmentwise Monthly Income')
```



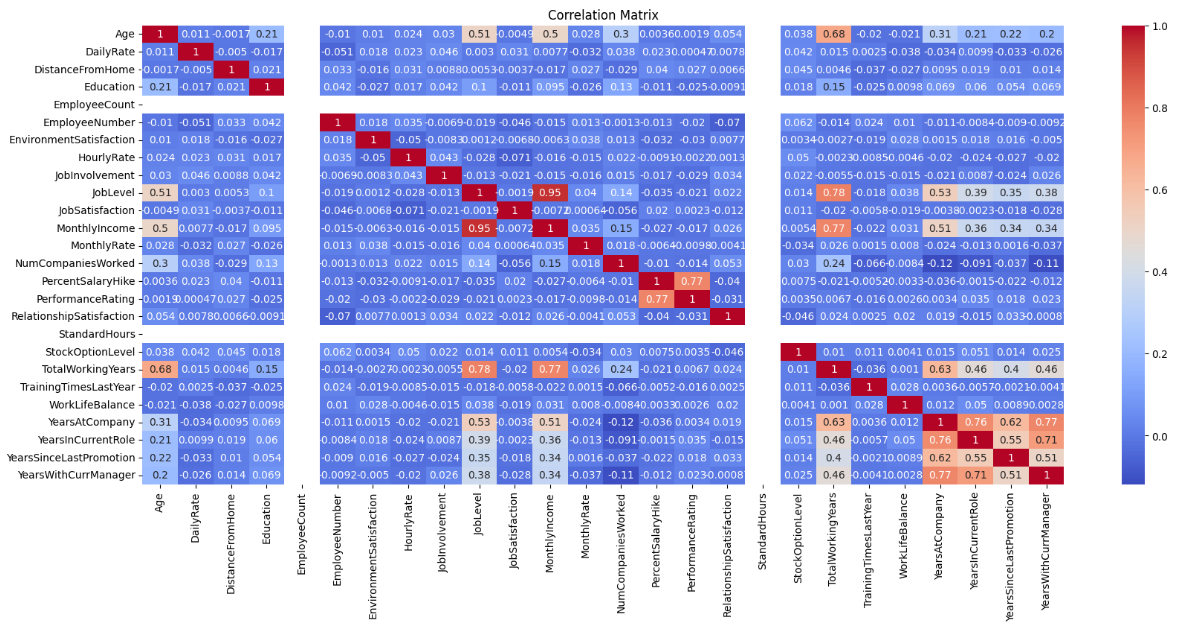
Age vs. Monthly Income

```
In [ ]: plt.figure(figsize=(10, 6))
        sns.scatterplot(x='Age', y='MonthlyIncome', data=df, hue='Attrition')
        plt.title('Scatterplot: Age vs. MonthlyIncome')
        plt.show()
```



Correlation Matrix

```
In [ ]: correlation_matrix = df.corr(numeric_only=True)
        plt.figure(figsize=(20, 8))
        sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
        plt.title('Correlation Matrix')
        plt.show()
```


Correlation Matrix
Age
DailyRate
DistanceFromHome
Education
EmployeeCount
EmployeeNumber
EnvironmentSatisfaction
HourlyRate
JobInvolvement
JobLevel
JobSatisfaction
MonthlyIncome
MonthlyRate
NumCompaniesWorked
PercentSalaryHike
PerformanceRating
RelationshipSatisfaction
StandardHours
StockOptionLevel
TotalWorkingYears
TrainingTimesLastYear
WorkLifeBalance
YearsAtCompany
YearsInCurrentRole
YearsSinceLastPromotion
YearsWithCurrManager