HR Data Analytics

A Summary Report

Libraries and Tools Used

- Matplotlib
- Seaborn
- Pandas
- Scikit-learn
- Microsoft Excel
- Jupyter Notebook

About the Data Set

- This is a data set with HR Data of employees for a certain firm.
- It has various columns with various details of the employees from various departments.
- These include values ranging from Numerical to Subjective attributes
- The data set has been visualized here and a model that predicts attrition based on this data set has also been implemented

Series of Steps

- Data Cleaning
- Getting the Data Set Mapped to Numerical Values
- Correlation and Visuals
- Relationships between Attributes
- Attrition Prediction Model

Data Cleaning

- Checking Null Values
  - Null Values removed using functionality under Pandas through Python
- Dropping Duplicates
  - Looking for duplicate columns and dropping them
- Dropping Columns with Redundant data
  - Columns with same data throughout all columns dropped

Data Cleaning

```
df.isnull().sum() # null values
```

```
# Dropping Duplicates
df = df.drop_duplicates()

# Removing null values
df = df.dropna()
```

Getting the Data Set Mapped to Numerical Values

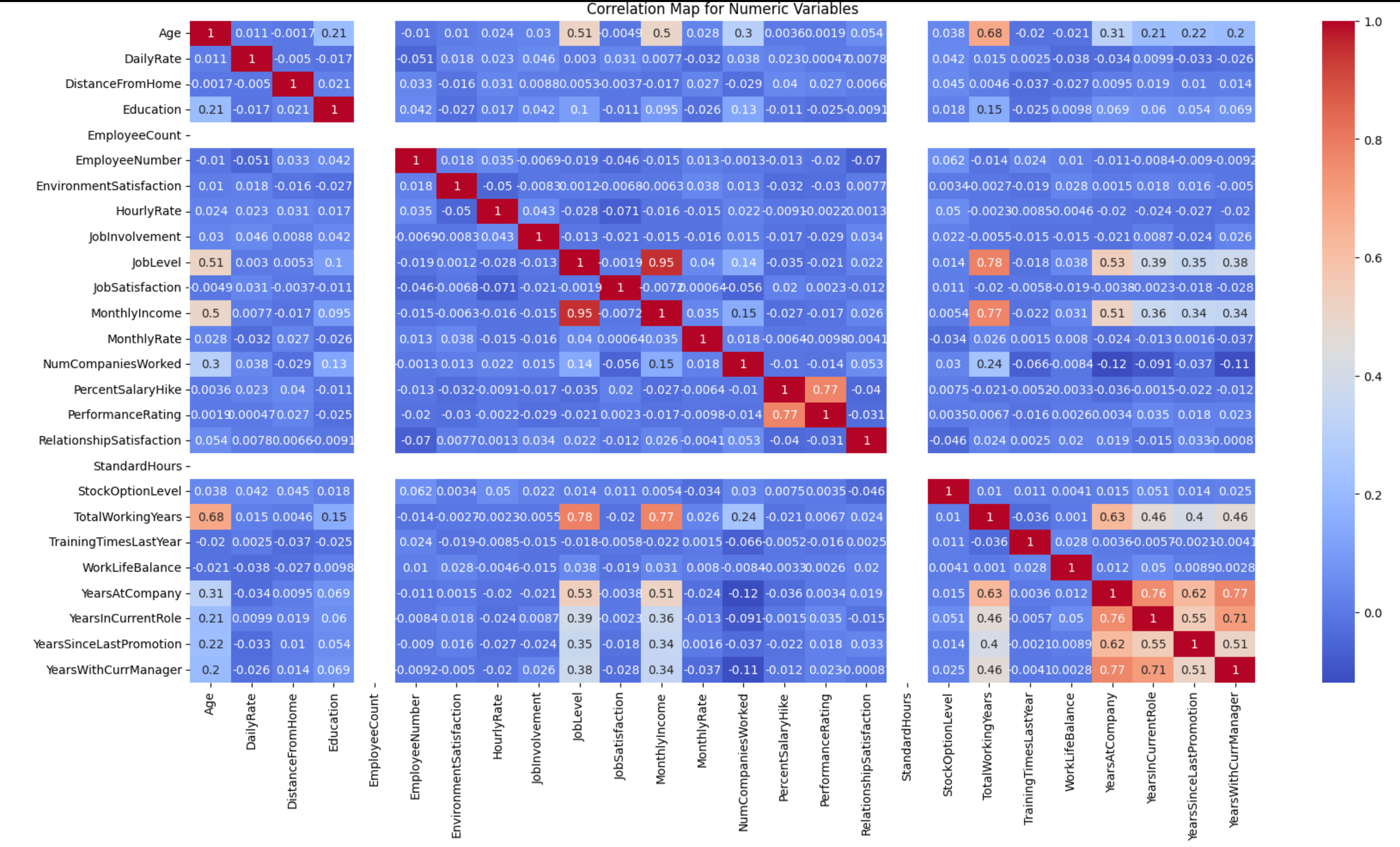
- Columns which contain qualitative data need to be mapped at some points of the analysis
- When we want our model to generate accurate results and we do not want to lose the information through qualitative columns we often use this mapping

Examples

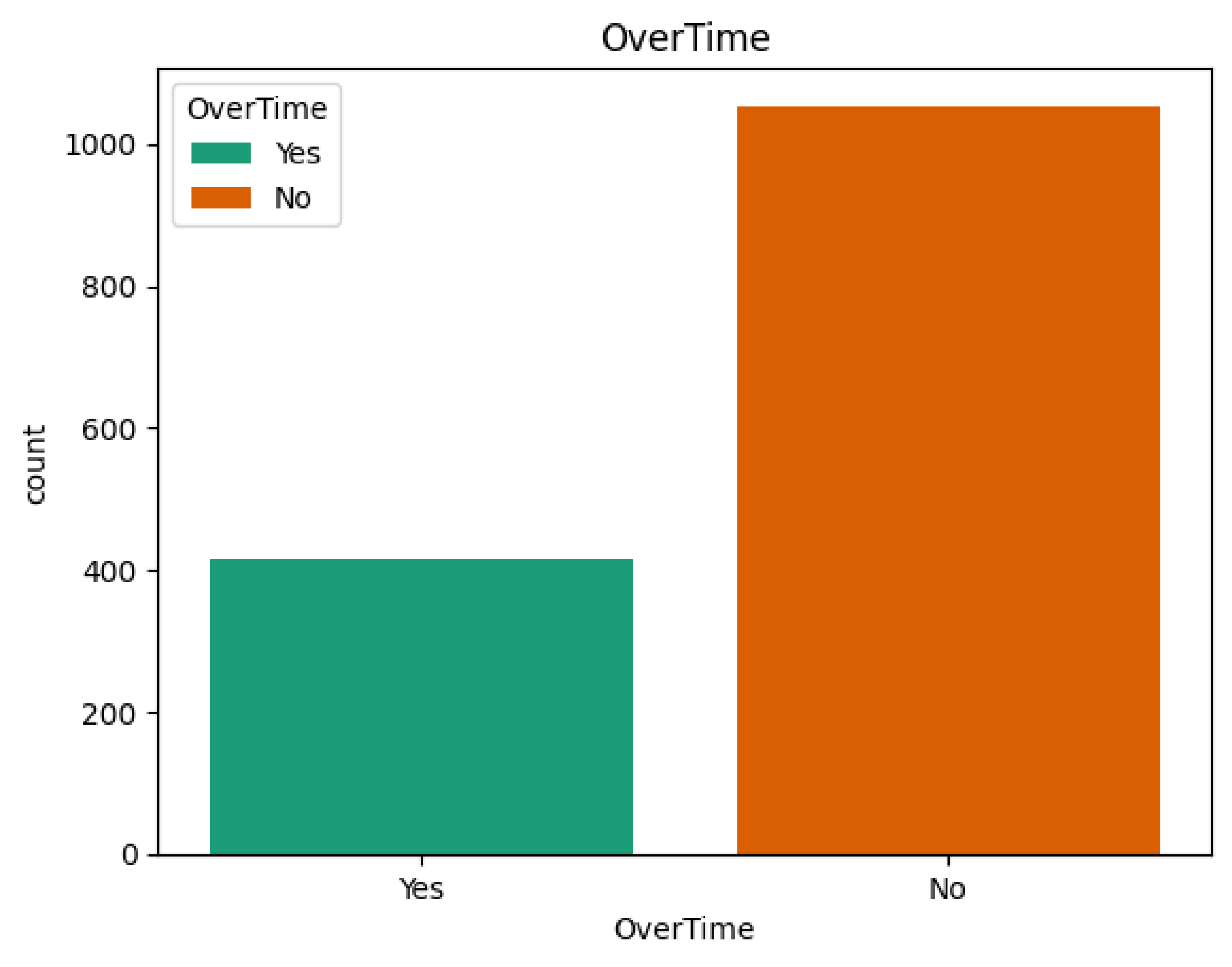
```
df['Attrition'] = df['Attrition'].map({'Yes': 1,
                                       'No': 0})
df['BusinessTravel'] = df['BusinessTravel'].map({'Non-Travel': 0,
                                                 'Travel_Rarely': 1,
                                                 'Travel_Frequently': 2})
df['Gender'] = df['Gender'].map({'Male': 1, 'Female': 0})
df['MaritalStatus'] = df['MaritalStatus'].map({'Married': 0,
                                               'Single': 1,
                                               'Divorced': 2})
df['OverTime'] = df['OverTime'].map({'Yes': 1,
                                     'No': 0})
```

Correlation and Visuals

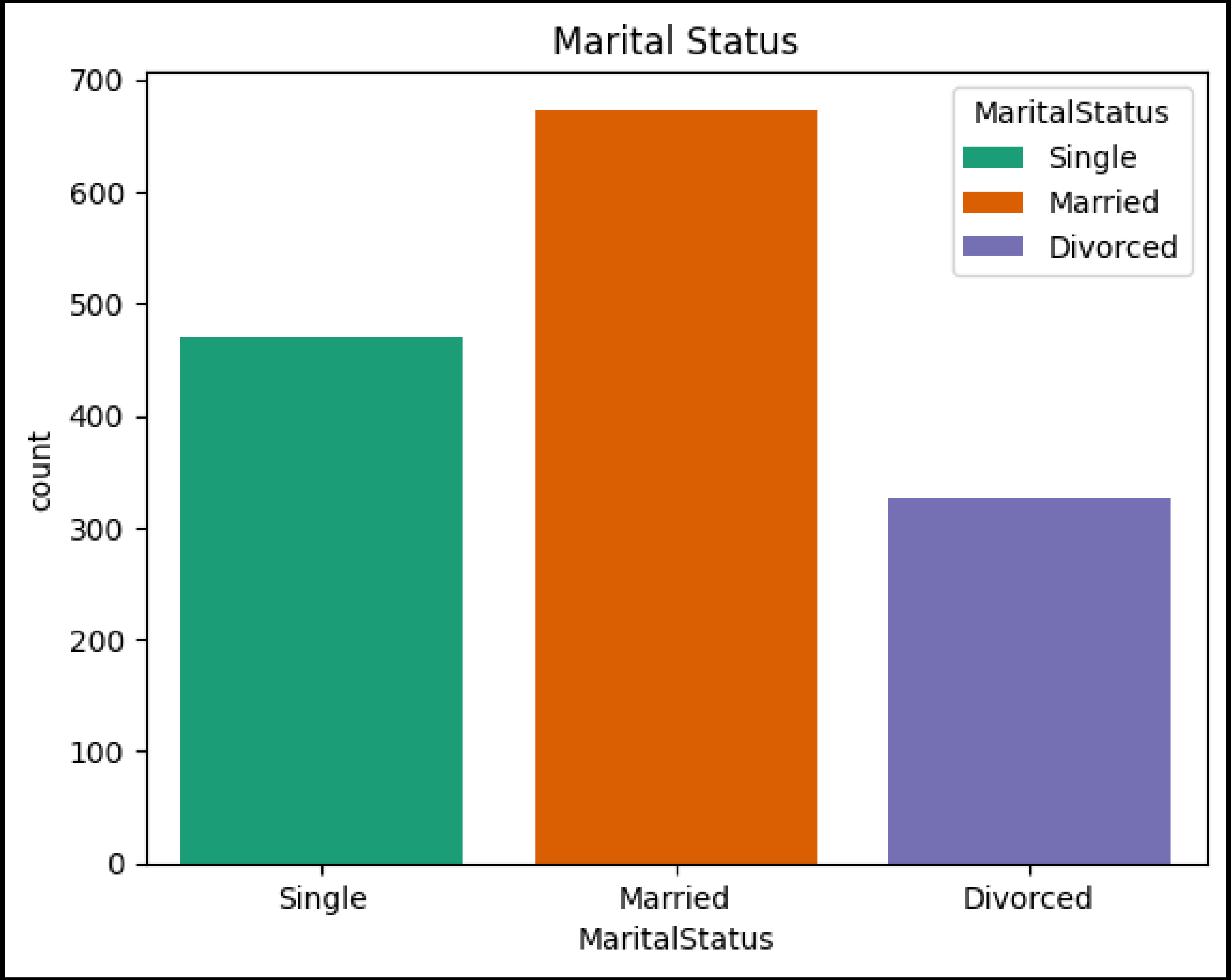
- Correlation between numeric data attributes has been depicted here.
- The relationships and visual representation of some of the relevant information has also been done
- This includes count-plots, histograms, boxplots etc.
- Some of the codes is there in this presentation but the complete source code can be downloaded from the GitHub Repository
  - GitHub Repo


Correlation Map for Numeric Variables
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
1.0
0.8
0.6
0.4
0.2
0.0

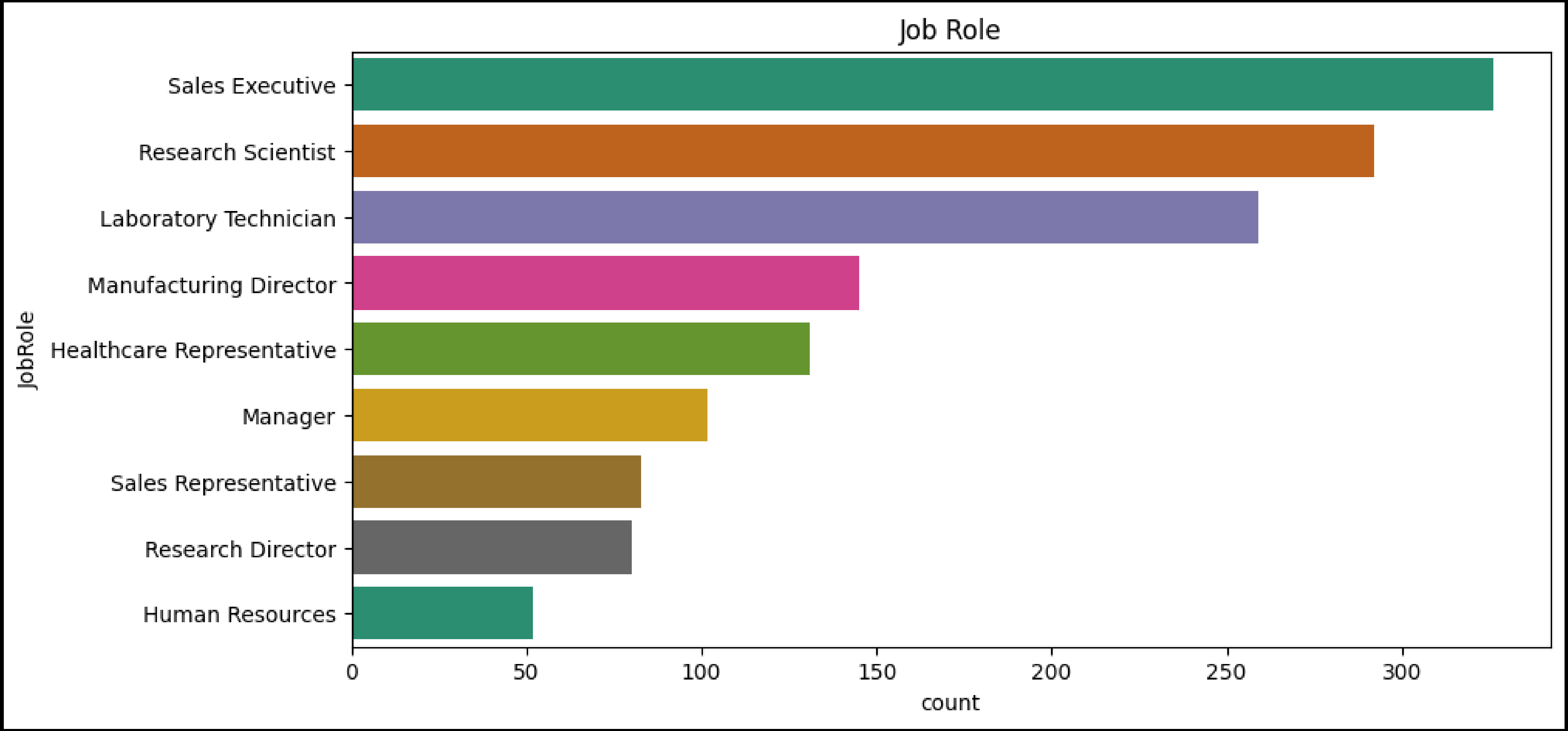
Overtime


OverTime
OverTime
Yes
No
1000
800
600
400
200
0
count
Yes
No
OverTime

Marital Status

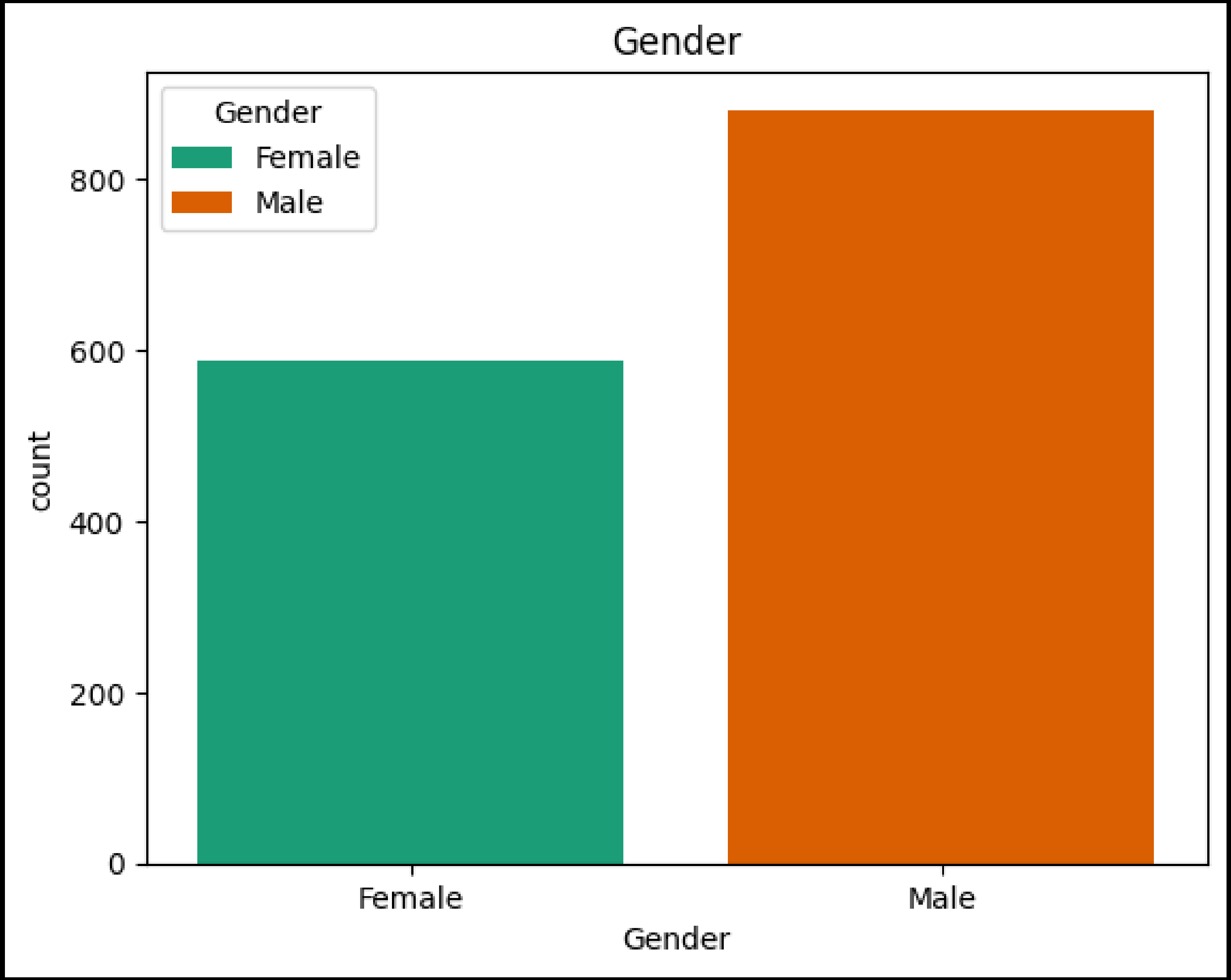


Job Role

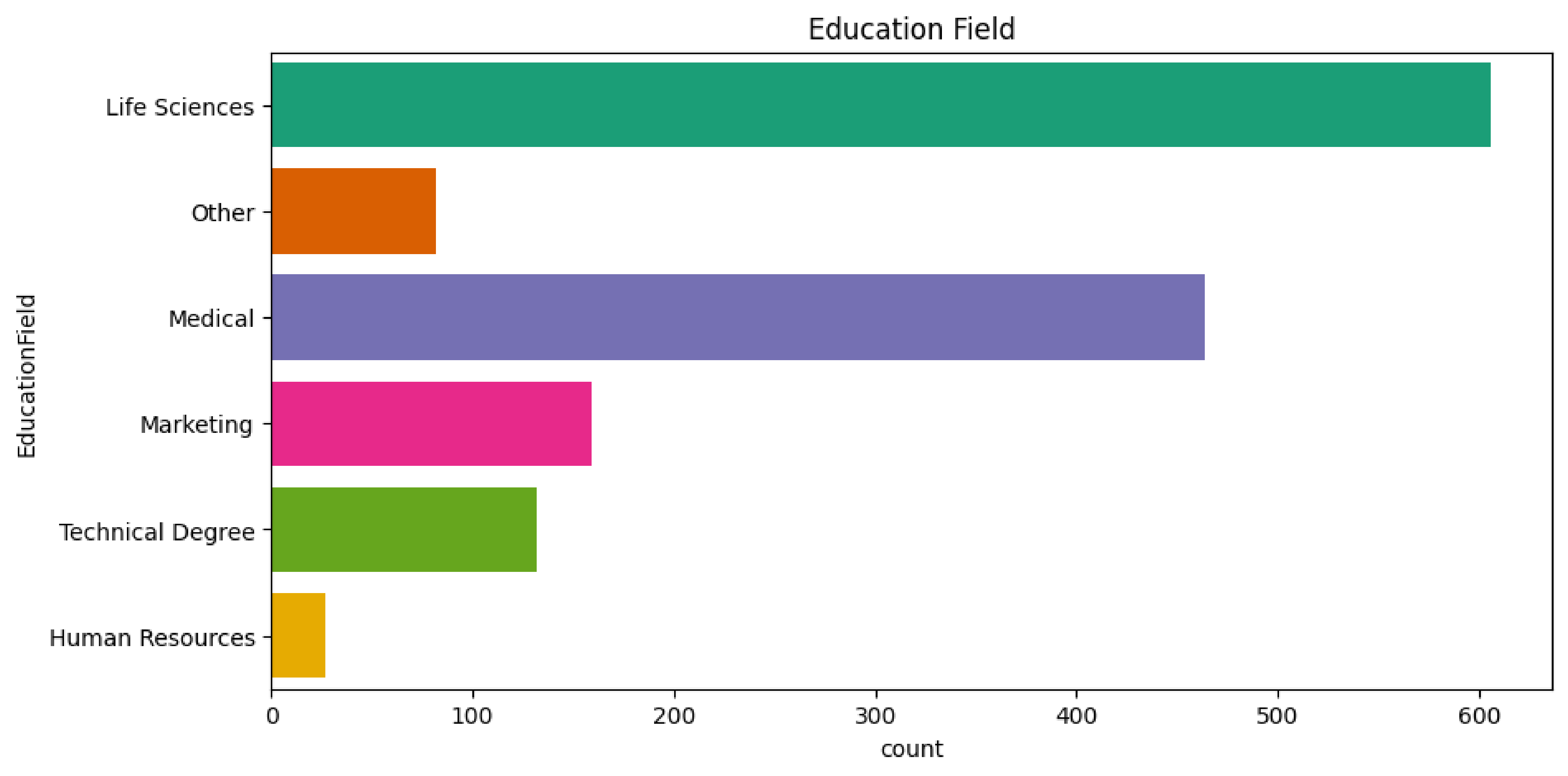


Job Role
Sales Executive
Research Scientist
Laboratory Technician
Manufacturing Director
Healthcare Representative
Manager
Sales Representative
Research Director
Human Resources
JobRole
0
50
100
150
200
250
300
count

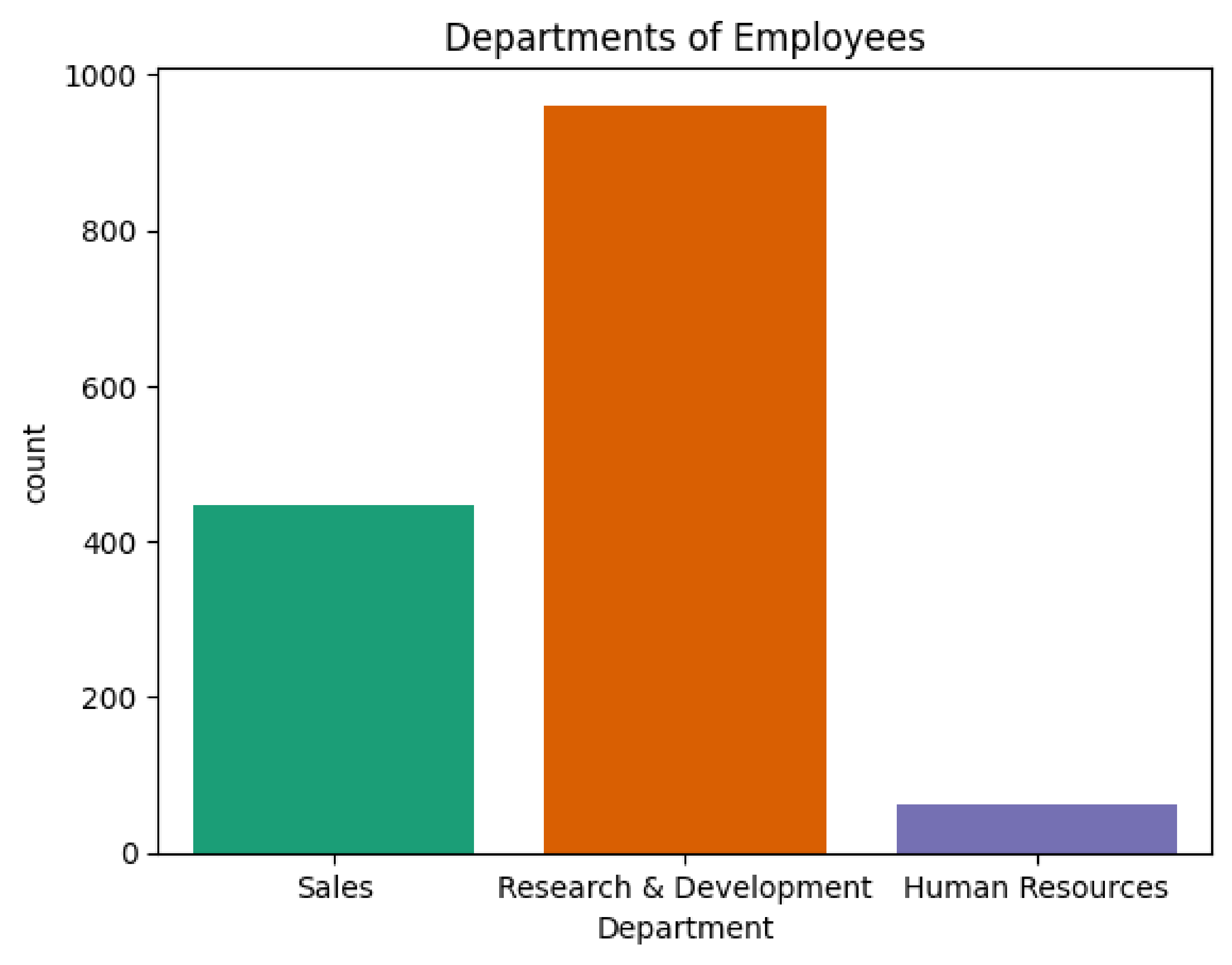
Gender


Gender
Gender
Female
Male
800
600
400
200
0
count
Female
Male
Gender

Education Field of Employees

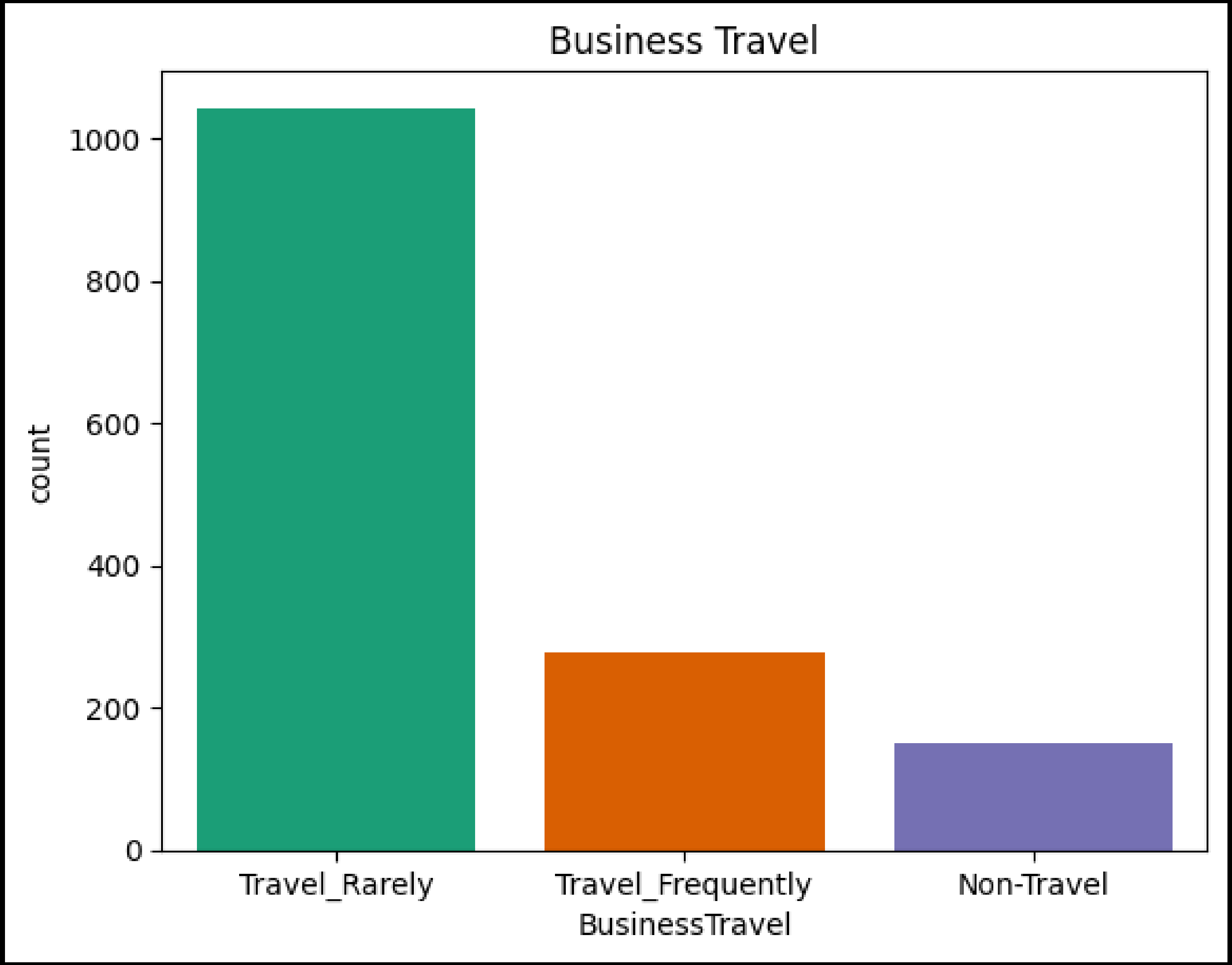


Department

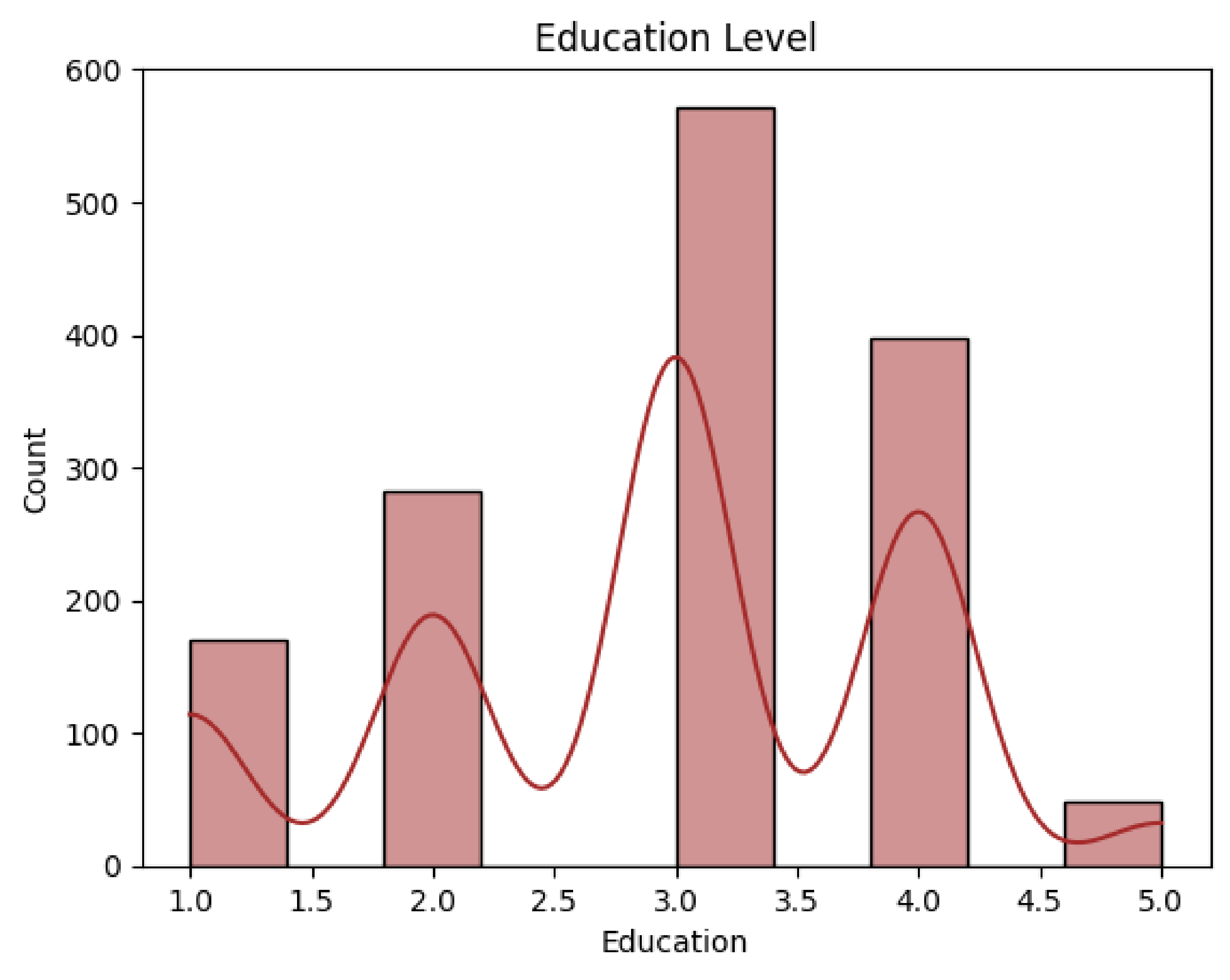


Departments of Employees
1000
800
600
400
200
0
count
Sales
Research & Development
Human Resources
Department

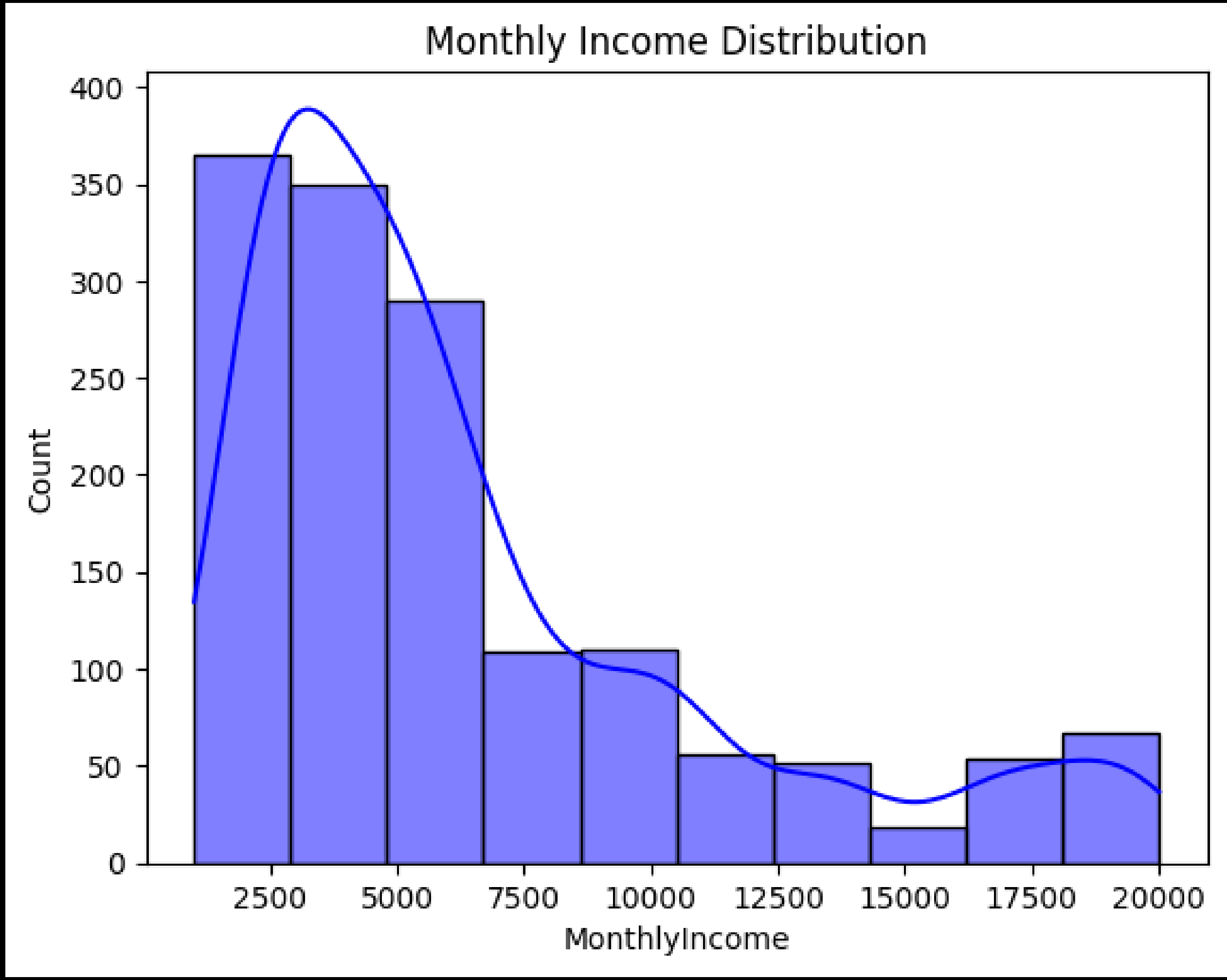
Business Travels of Employees



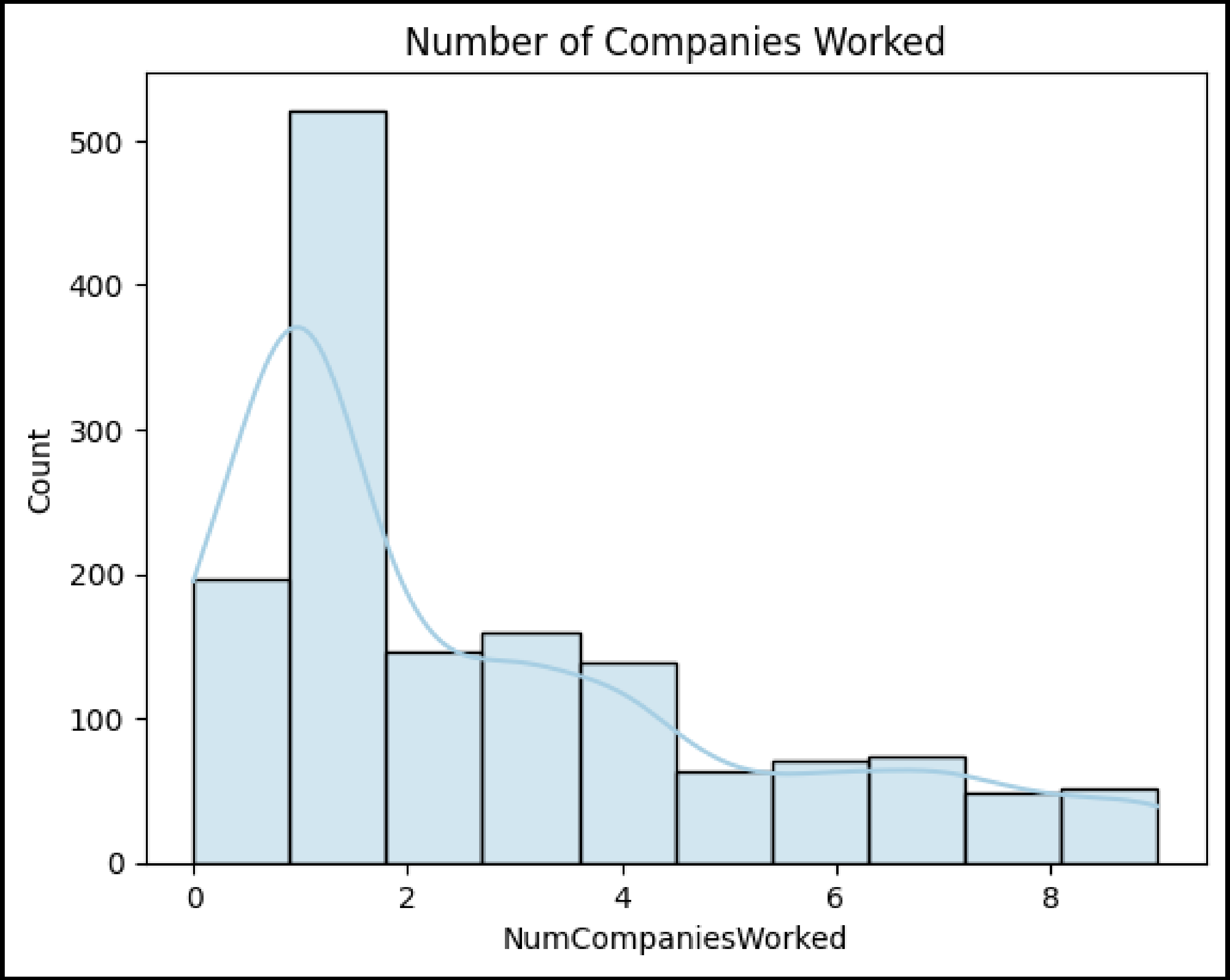
Education Level of Employees



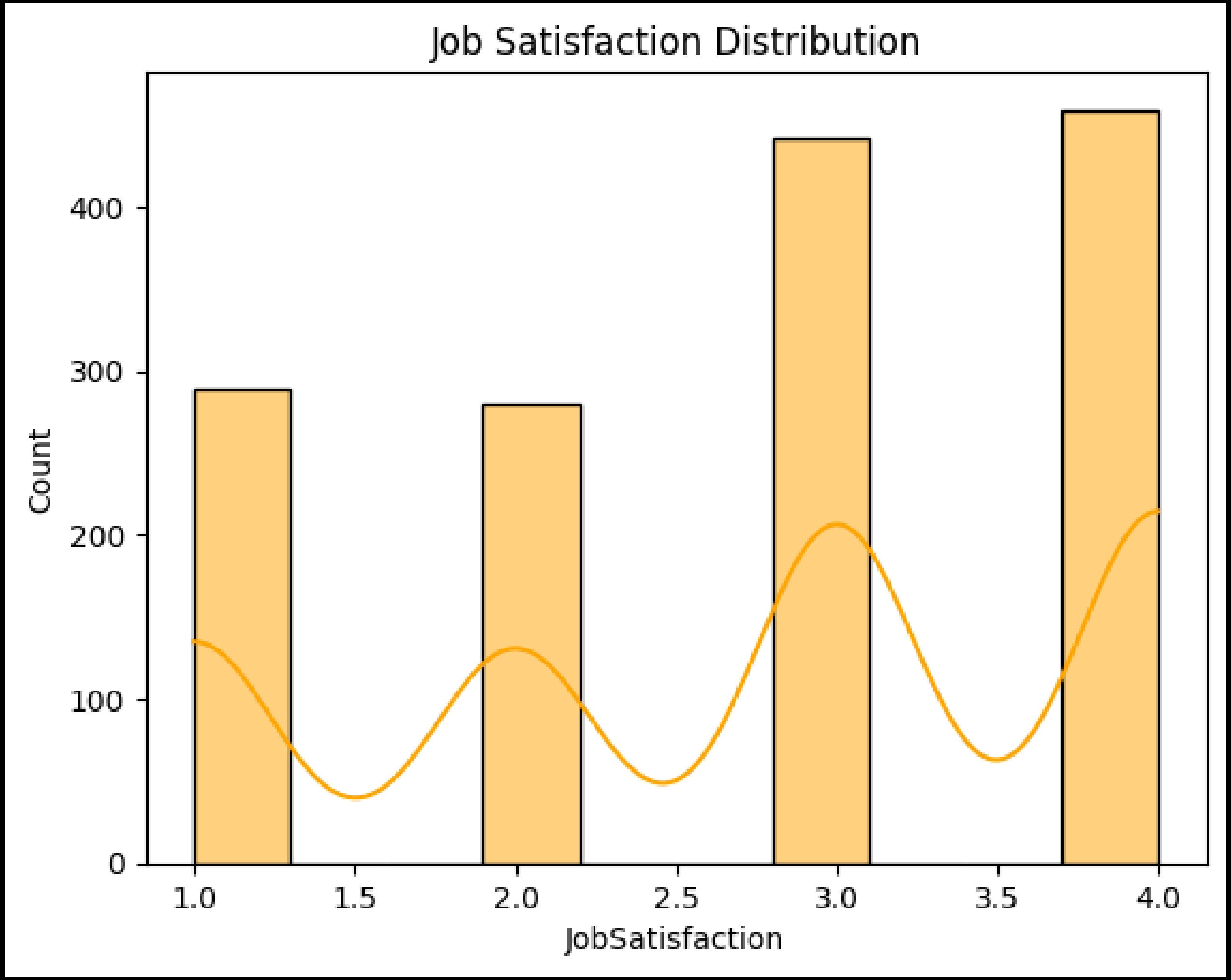
Monthly Income of Employees



Number of Companies Worked

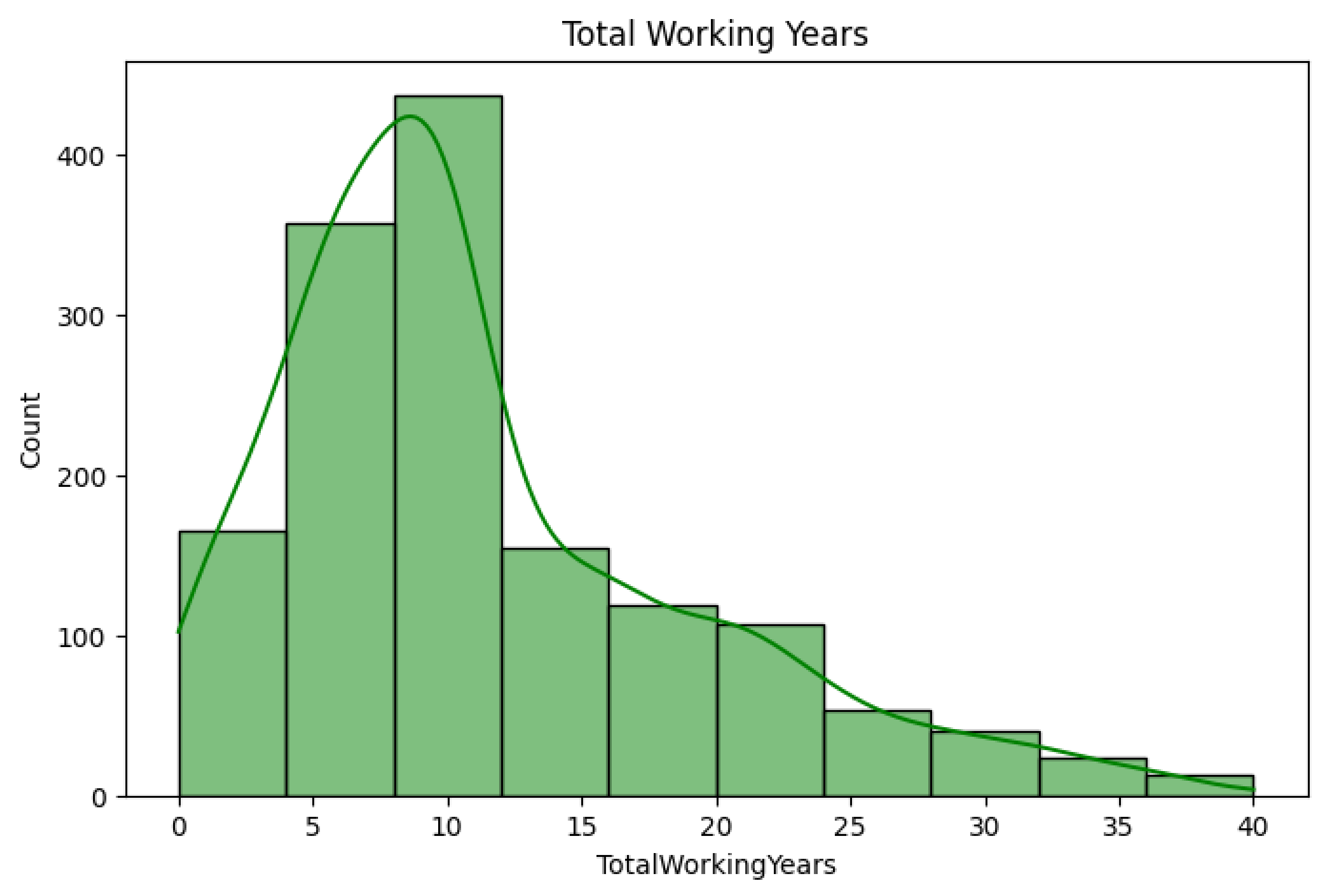


Job Satisfaction

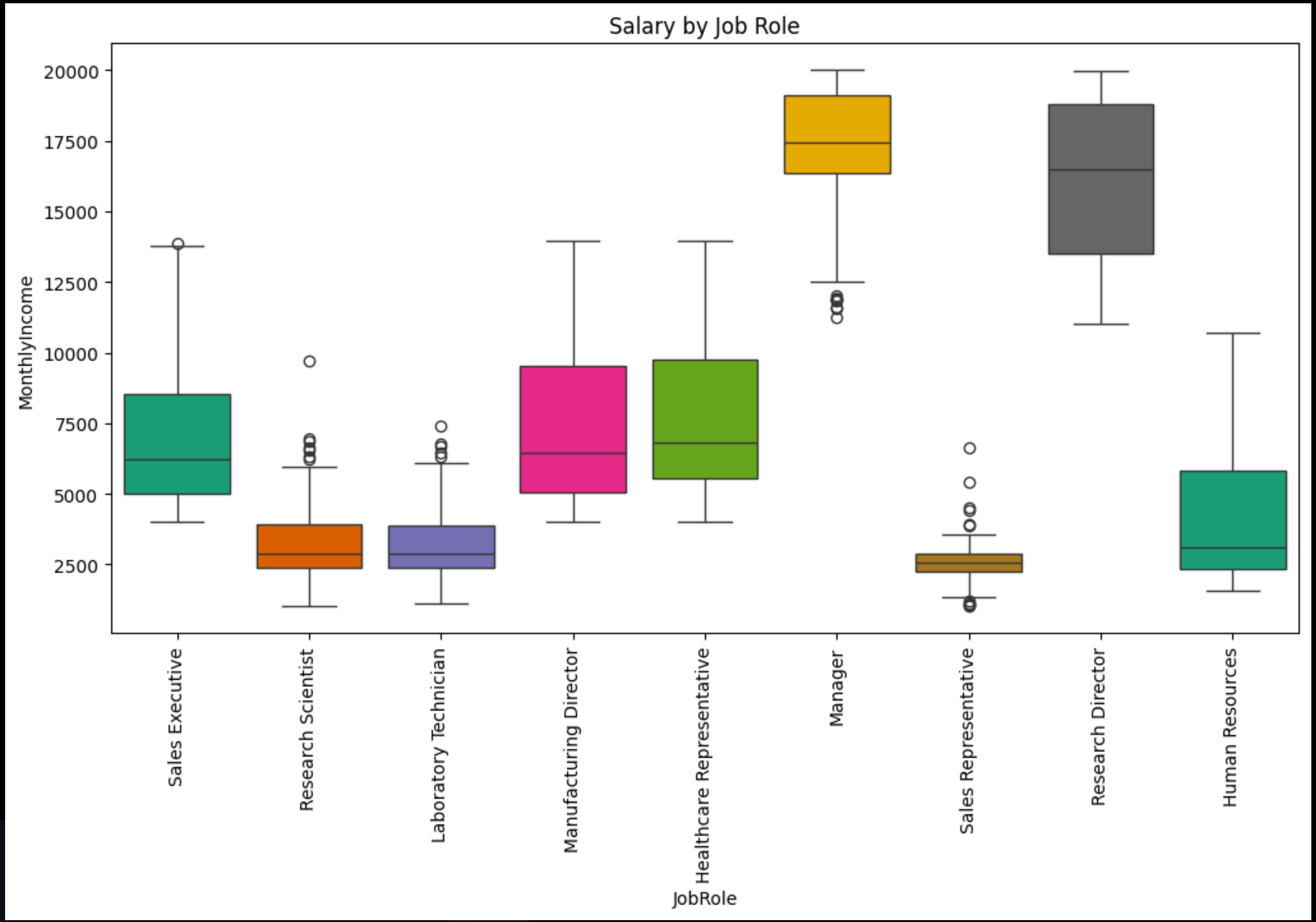


```python
# Total working Years
plt.figure(figsize=(8, 5))
sns.histplot(data = df, x = "TotalWorkingYears",
             bins = 10, kde = True, color = 'green')
plt.title('Total Working Years')
plt.show()
```

Total Working Years Distribution



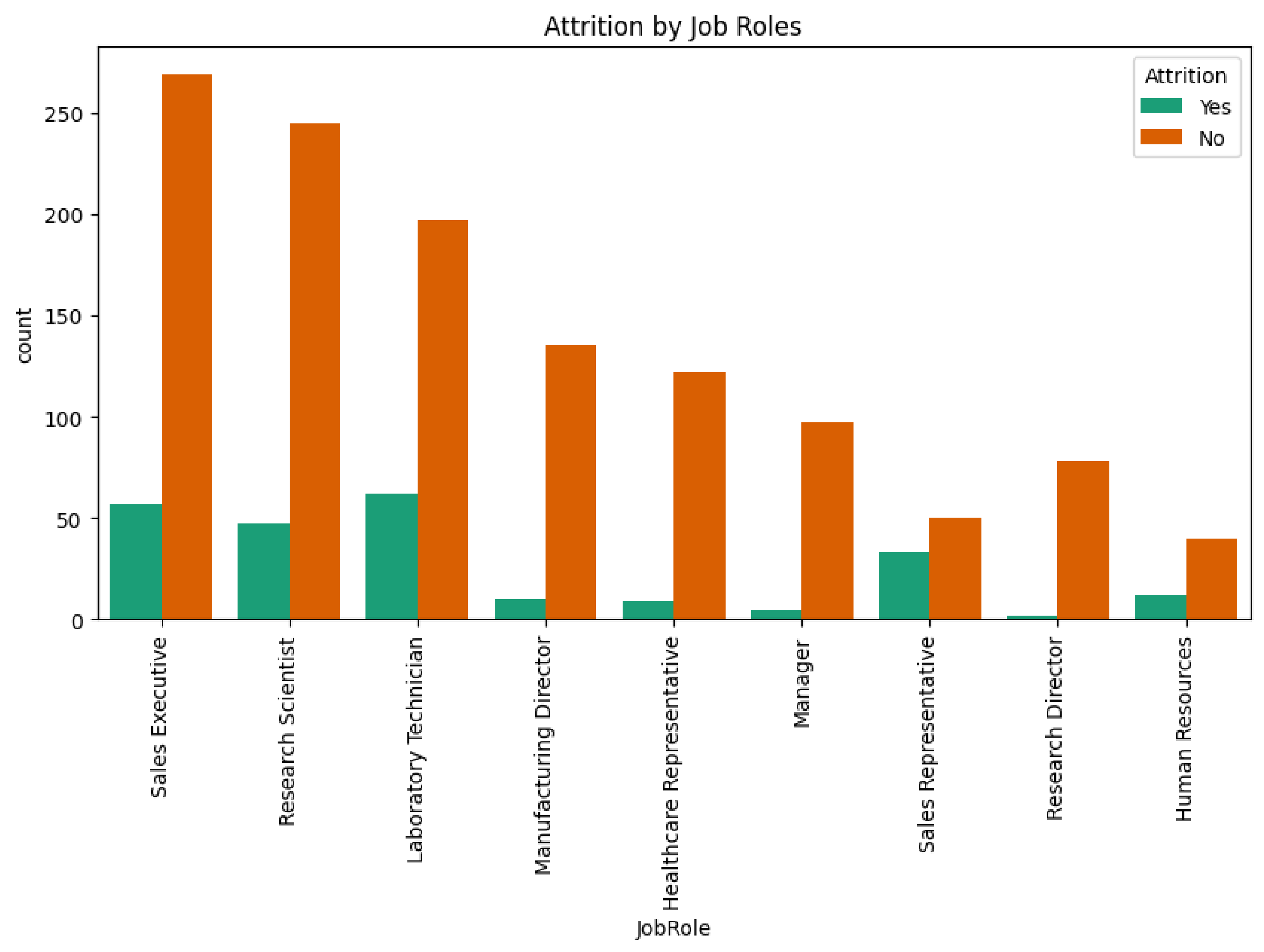
Comparison of Salary and Job Role




```python
# Salary by Job Role
plt.figure(figsize = (12, 6))
sns.boxplot(data = df, x = "JobRole", y = "MonthlyIncome",
            saturation = 1, hue = "JobRole",
            palette = custom_palette[:len(df["JobRole"].unique())])
plt.title("Salary by Job Role")
plt.xticks(rotation = 90)
plt.show()
```

```
# Attrition by Job Role
plt.figure(figsize = (10, 5))
sns.countplot(data = df, x = "JobRole",
              hue = "Attrition", saturation = 1,
              palette = custom_palette[:len(df["Attrition"].unique())])
plt.title("Attrition by Job Roles")
plt.xticks(rotation = 90)
plt.show()
```

Comparison of Attritions under different Job Roles



Attrition Prediction Model

- Algorithm used :- Random Forest Classification
- Target :- Attrition
- Accuracy 0.8639455782312925

```python
# Assuming 'Attrition' is the target variable
X = df.drop(['Attrition','Department','EducationField', 'JobRole', 'Over18'], axis=1)#
```

```python
X.fillna(0, inplace=True)  # Fill missing values with 0 of each column
```

```python
y = df['Attrition']  # Target variable
y.fillna(0, inplace=True)  # Fill missing values with 0 of each column
```

```python
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

```python
# Making predictions
predictions = model.predict(X_test)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
Accuracy: 0.8639455782312925
```

Thank You!