1. Loading Dataset and Exploratory Analysis

```
# --- Load Required Libraries ---
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# --- Install API and upload the kaggle.json file for authentication ---
!pip install -q kaggle
from google.colab import files

# --- Upload API credentials file ---
files.upload()

# --- Setup API credentials ---
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json

# --- Download the dataset ---
!kaggle datasets download -d pavansubhasht/ibm-hr-analytics-attrition-dataset

# --- Unzip the dataset ---
!unzip ibm-hr-analytics-attrition-dataset.zip

# --- Load the dataset ---
df = pd.read_csv('WA_Fn-UseC_-HR-Employee-Attrition.csv')
```

Choose Files kaggle.json

- **kaggle.json**(application/json) - 65 bytes, last modified: n/a - 100% done

```
Saving kaggle.json to kaggle.json
Dataset URL: https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-at
License(s): DbCL-1.0
Archive:  ibm-hr-analytics-attrition-dataset.zip
  inflating: WA_Fn-UseC_-HR-Employee-Attrition.csv
```

```
df
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome
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
...	...	...	...	...	...	...
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

1470 rows × 35 columns

```
# --- Basic Exploration ---
print(df.shape)
print(df.info())
print(df.describe())
```

```
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
```

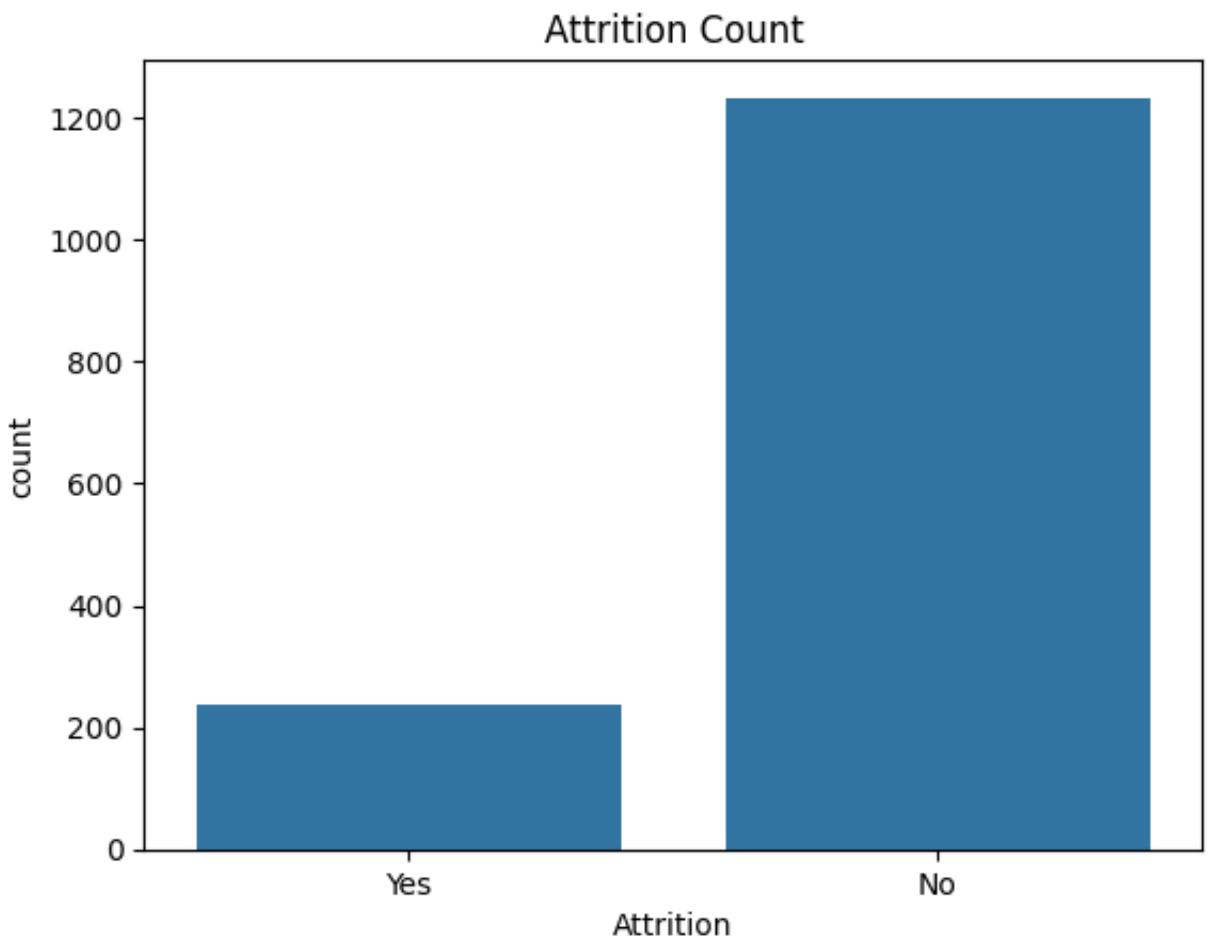
```
       EmployeeNumber  EnvironmentSatisfaction   HourlyRate  JobInvolvement  \
count     1470.000000              1470.000000  1470.000000     1470.000000
mean      1024.865306                 2.721769    65.891156        2.729932
std        602.024335                 1.093082    20.329428        0.711561
min          1.000000                 1.000000    30.000000        1.000000
25%        491.250000                 2.000000    48.000000        2.000000
50%       1020.500000                 3.000000    66.000000        3.000000
75%       1555.750000                 4.000000    83.750000        3.000000
max       2068.000000                 4.000000   100.000000        4.000000

          JobLevel  ...  RelationshipSatisfaction  StandardHours  \
count  1470.000000  ...               1470.000000         1470.0
mean      2.063946  ...                  2.712245           80.0
std       1.106940  ...                  1.081209            0.0
min       1.000000  ...                  1.000000           80.0
25%       1.000000  ...                  2.000000           80.0
50%       2.000000  ...                  3.000000           80.0
75%       3.000000  ...                  4.000000           80.0
max       5.000000  ...                  4.000000           80.0

       StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear  \
count       1470.000000        1470.000000            1470.000000
mean           0.793878          11.279592               2.799320
std            0.852077           7.780782               1.289271
min            0.000000           0.000000               0.000000
25%            0.000000           6.000000               2.000000
50%            1.000000          10.000000               3.000000
75%            1.000000          15.000000               3.000000
max            3.000000          40.000000               6.000000

       WorkLifeBalance  YearsAtCompany  YearsInCurrentRole  \
count      1470.000000     1470.000000         1470.000000
mean          2.761224        7.008163            4.229252
std           0.706476        6.126525            3.623137
min           1.000000        0.000000            0.000000
25%           2.000000        3.000000            2.000000
50%           3.000000        5.000000            3.000000
75%           3.000000        9.000000            7.000000
max           4.000000       40.000000           18.000000

       YearsSinceLastPromotion  YearsWithCurrManager
count              1470.000000           1470.000000
mean                  2.187755              4.123129
std                   3.222430              3.568136
min                   0.000000              0.000000
25%                   0.000000              2.000000
50%                   1.000000              3.000000
75%                   3.000000              7.000000
max                  15.000000             17.000000
```

```
[8 rows x 26 columns]
```

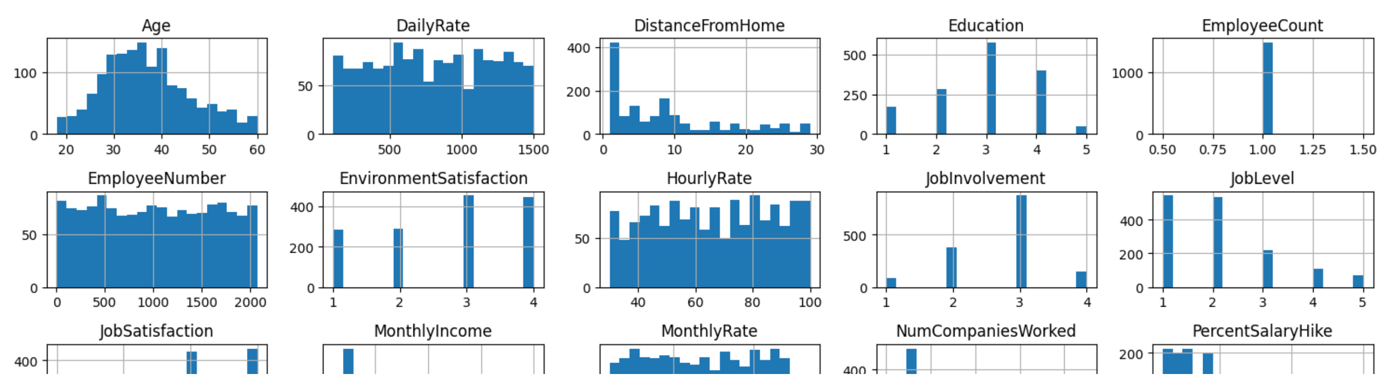
```
# --- Check for Missing Values ---
print(df.isnull().sum())
```

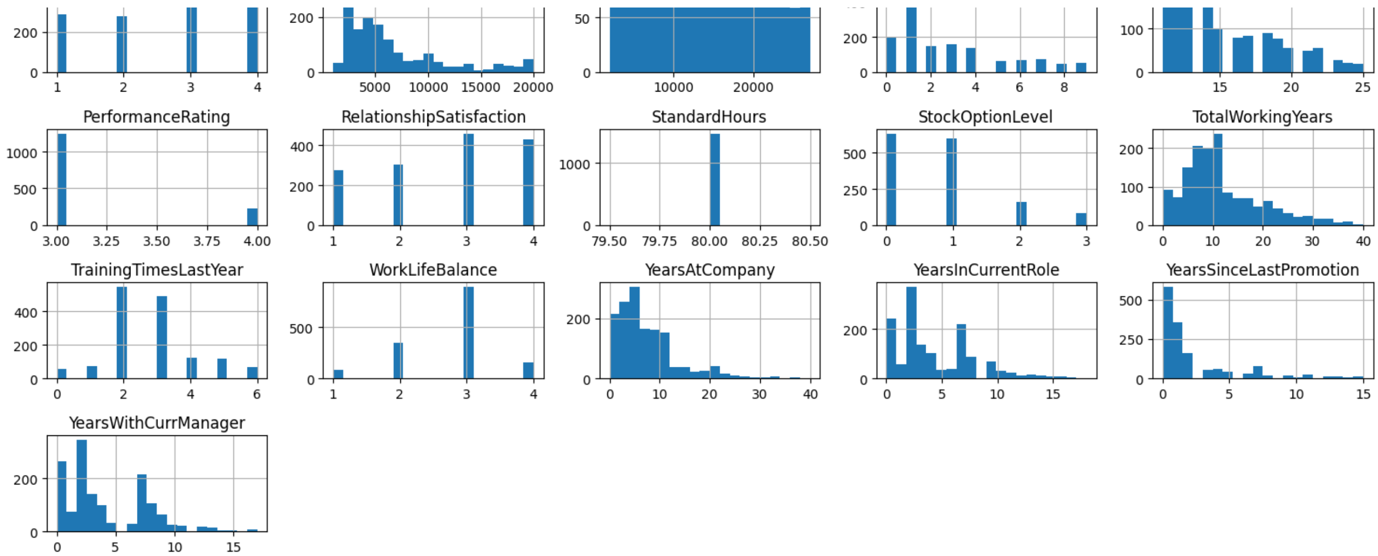
```
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              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
```

```
# --- Attrition Distribution ---
sns.countplot(x='Attrition', data=df)
plt.title('Attrition Count')
plt.show()
```



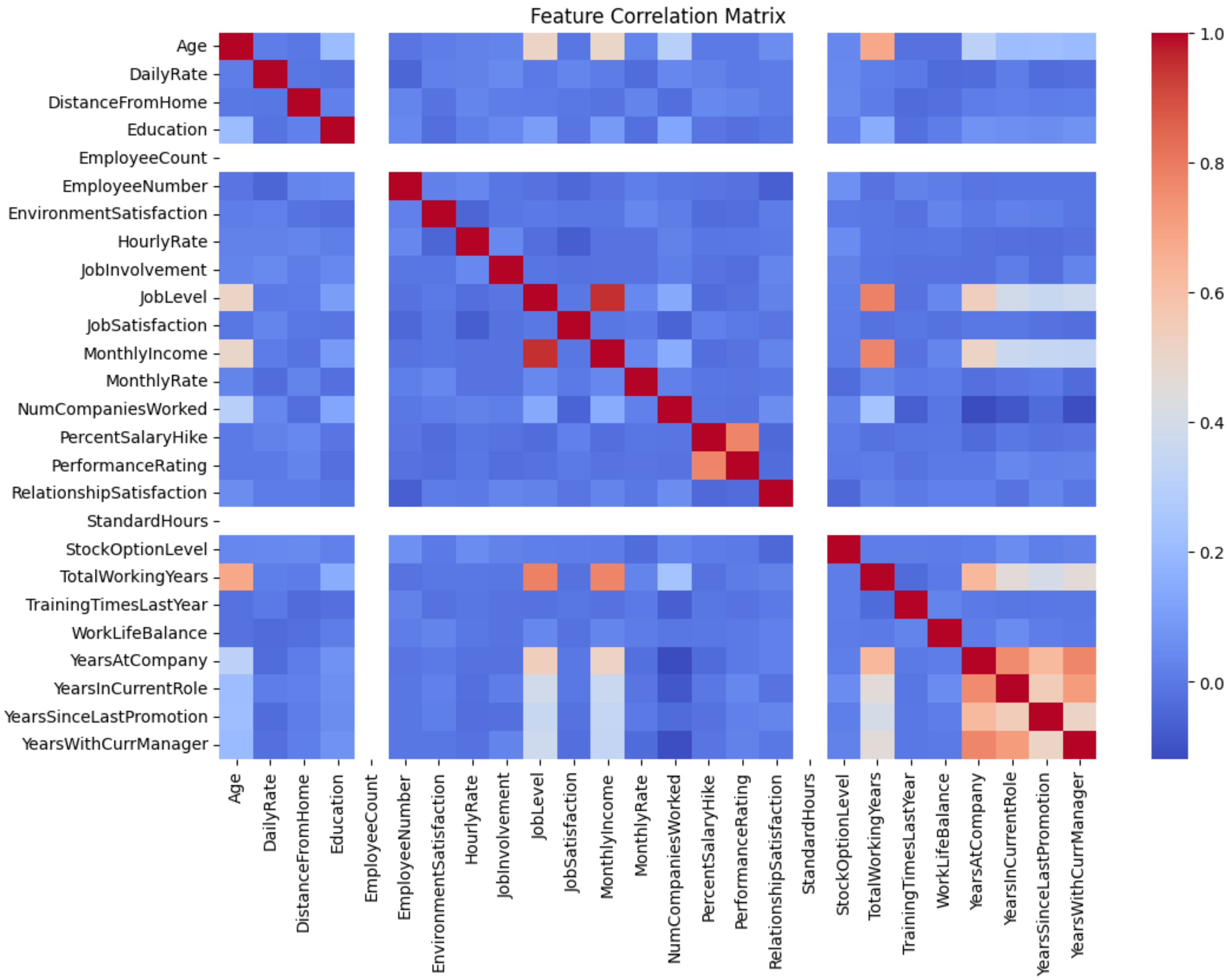

```
# --- Numeric Distributions ---
df.hist(figsize=(15, 10), bins=20)
plt.tight_layout()
plt.show()
```



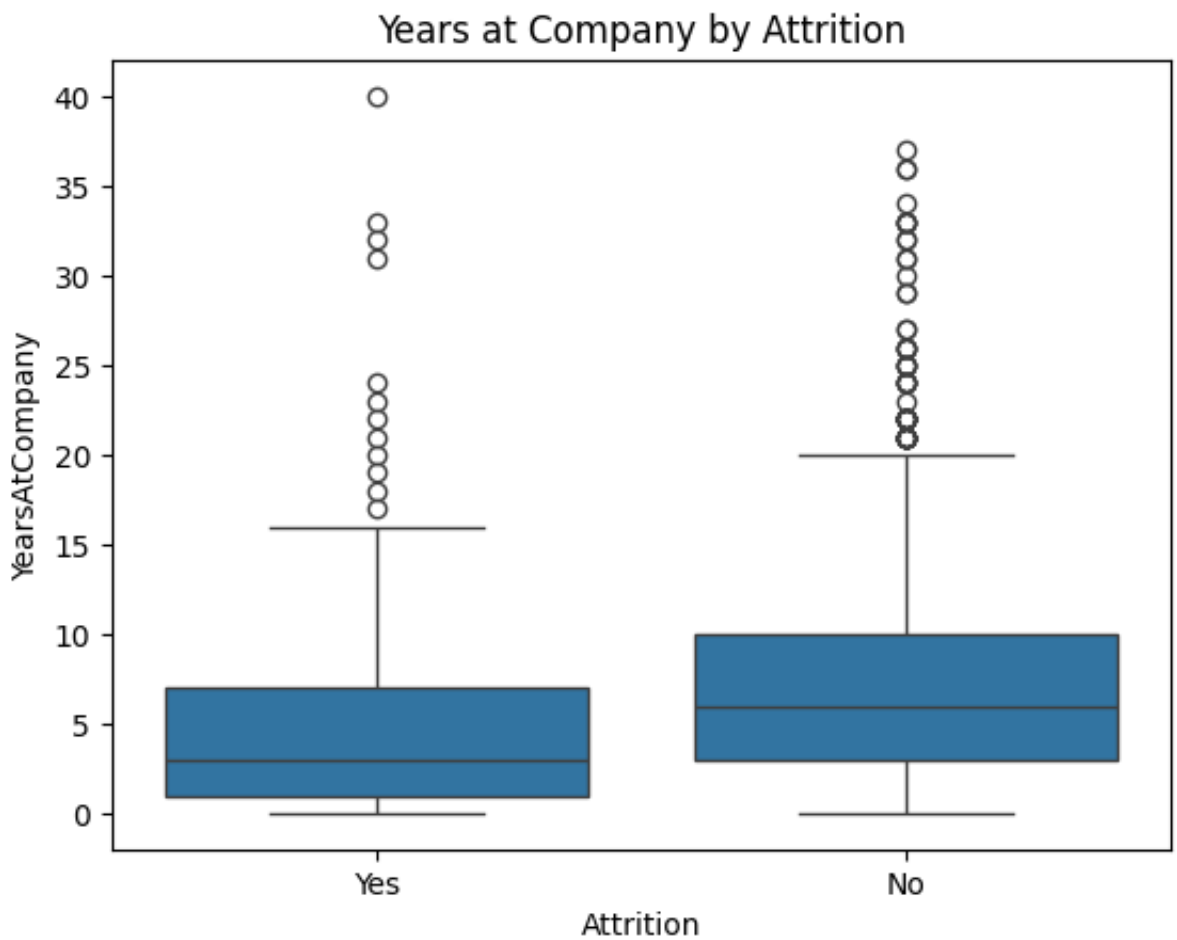


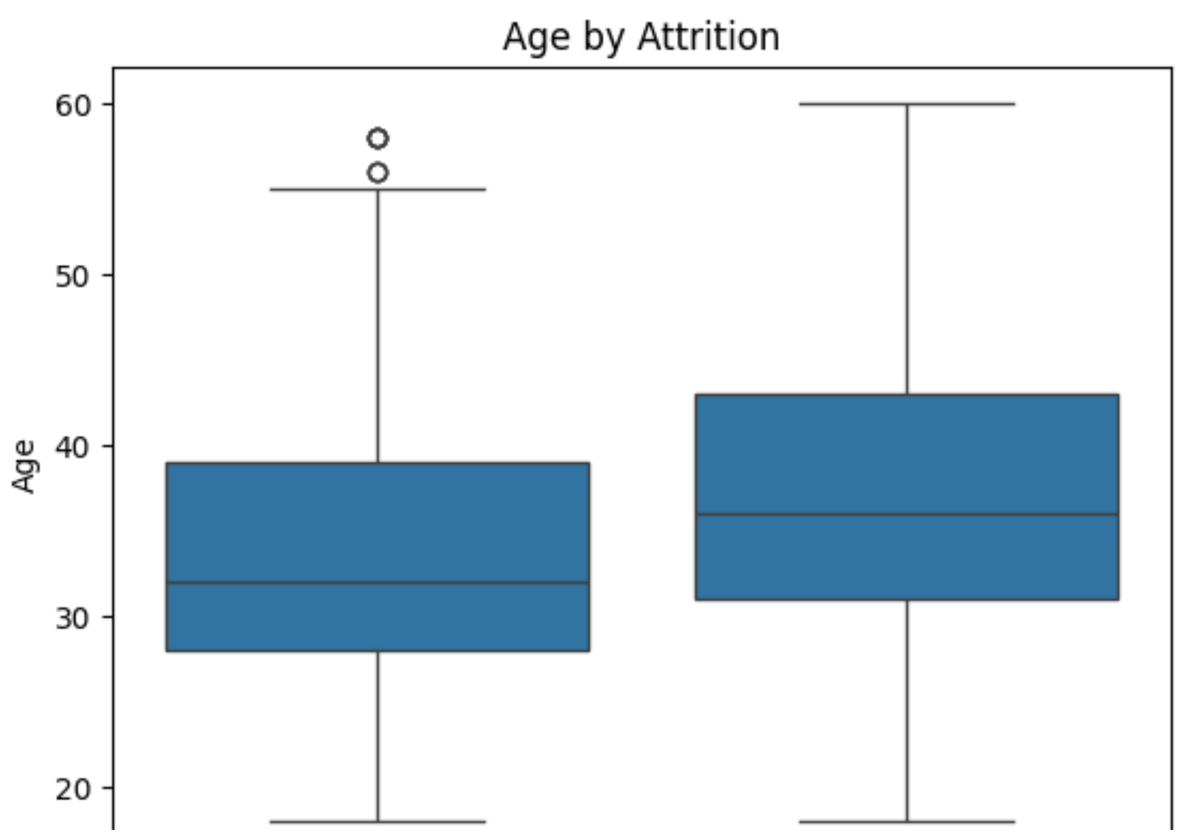

```
# --- Correlation Heatmap ---
plt.figure(figsize=(12, 8))
numeric_df = df.select_dtypes(include='number')
sns.heatmap(numeric_df.corr(), annot=False, cmap='coolwarm')
```

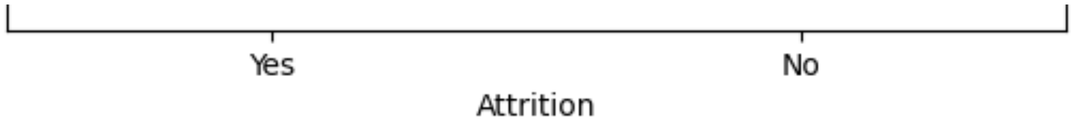
```
plt.title('Feature Correlation Matrix')
plt.show()
```

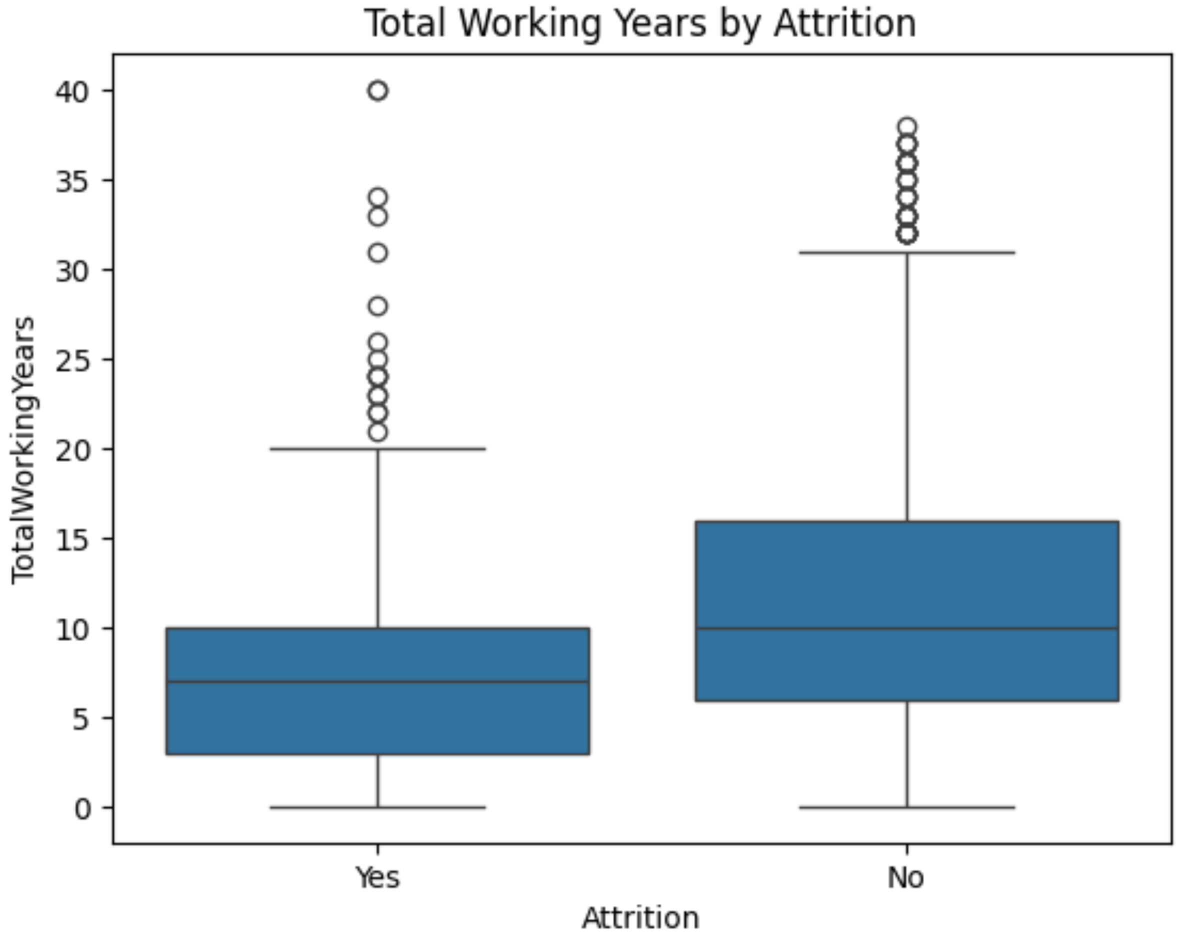


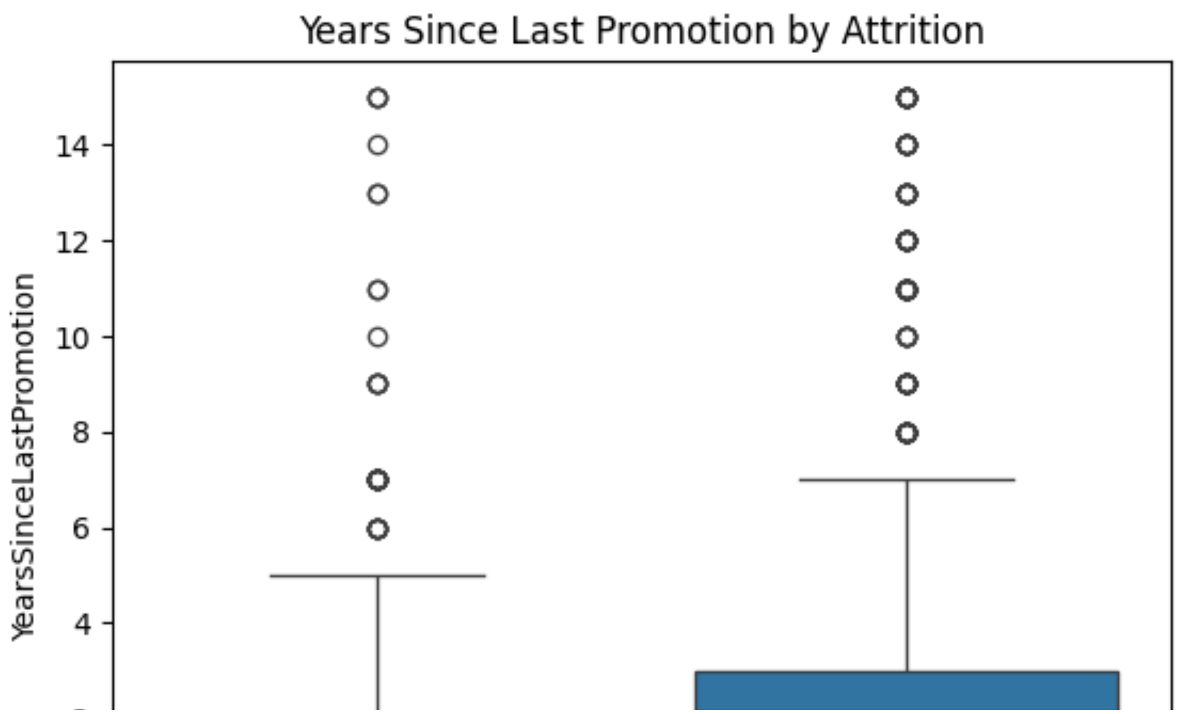
```
# --- Attrition by Key Numeric Features ---

sns.boxplot(x='Attrition', y='YearsAtCompany', data=df)
plt.title('Years at Company by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='Age', data=df)
plt.title('Age by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='TotalWorkingYears', data=df)
plt.title('Total Working Years by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='YearsSinceLastPromotion', data=df)
plt.title('Years Since Last Promotion by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='JobLevel', data=df)
plt.title('Job Level(1-5 scale) by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='JobSatisfaction', data=df)
plt.title('Job Satisfaction(1-5 scale) by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='EnvironmentSatisfaction', data=df)
plt.title('Environment Satisfaction(1-5 scale) by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='NumCompaniesWorked', data=df)
plt.title('Number of Companies Worked by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='DistanceFromHome', data=df)
plt.title('Distance From Home by Attrition')
plt.show()

sns.boxplot(x='Attrition', y='MonthlyIncome', data=df)
plt.title('Monthly Income by Attrition')
plt.show()
```

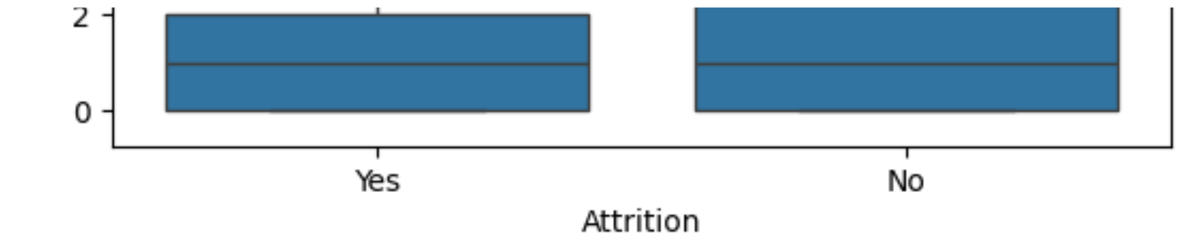

Years at Company by Attrition
YearsAtCompany
0
5
10
15
20
25
30
35
40
Yes
No
Attrition

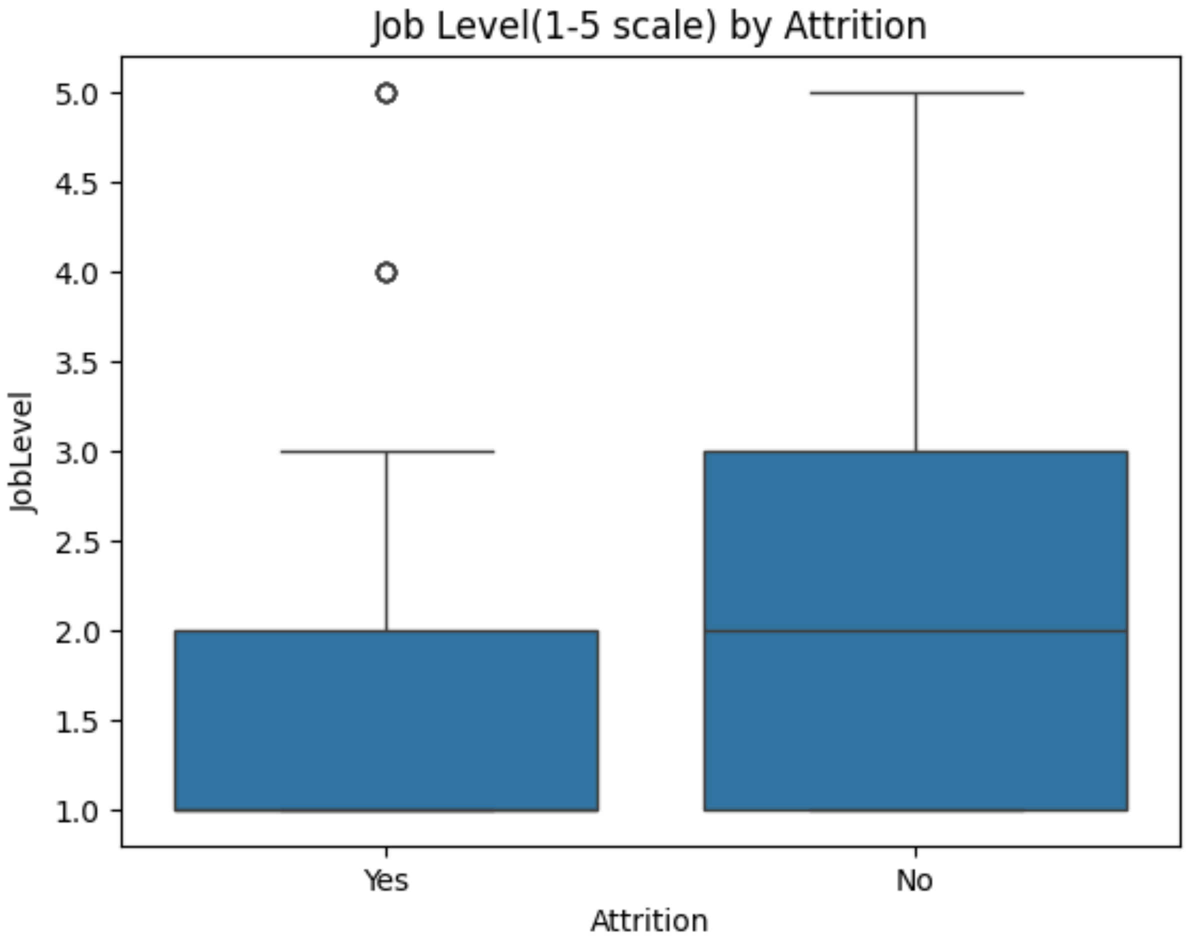

Age by Attrition
Age
20
30
40
50
60


Yes
No
Attrition

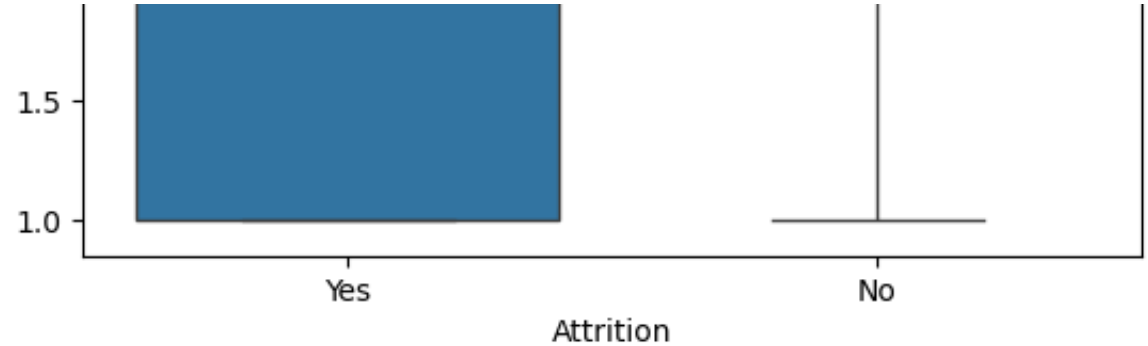

Total Working Years by Attrition
TotalWorkingYears
40
35
30
25
20
15
10
5
0
Yes
No
Attrition

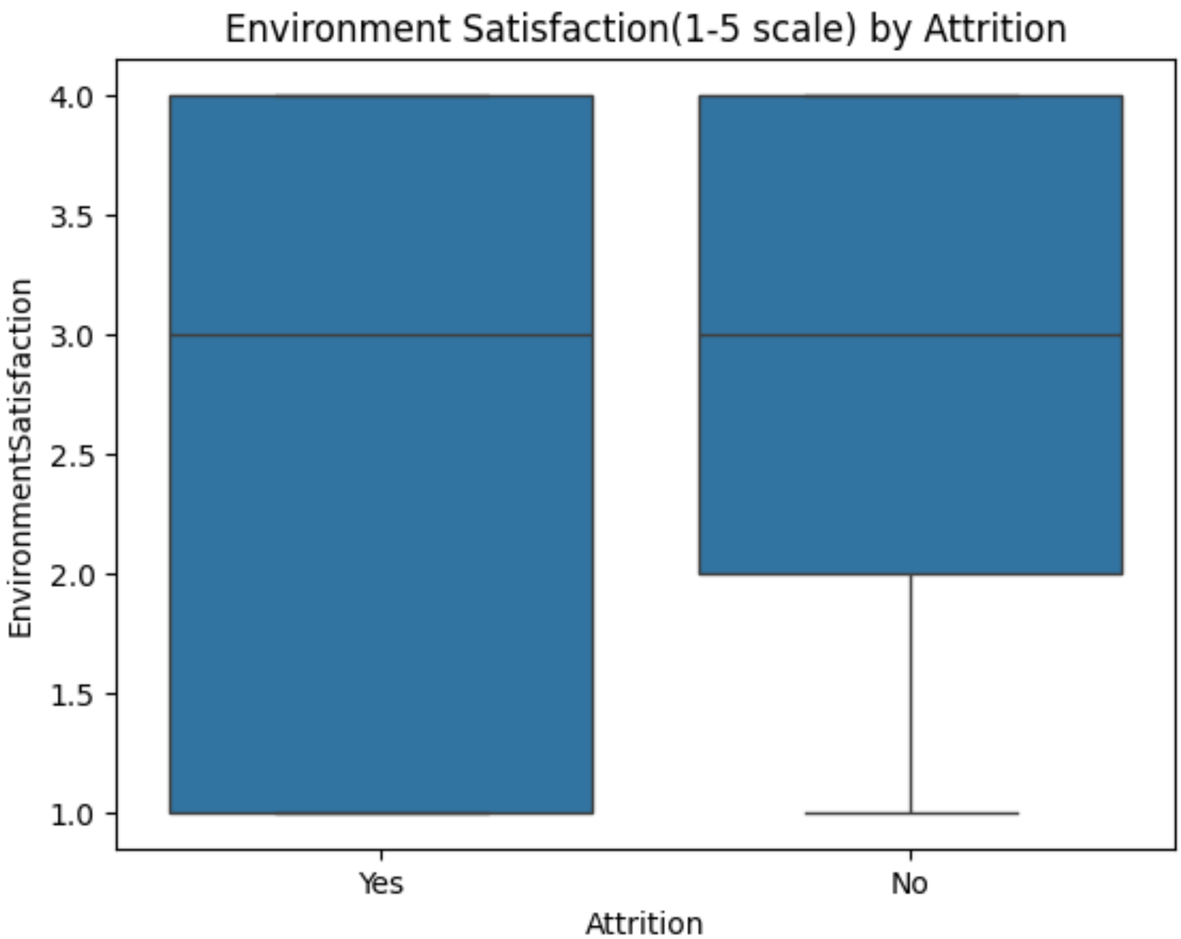

Years Since Last Promotion by Attrition
YearsSinceLastPromotion
14
12
10
8
6
4

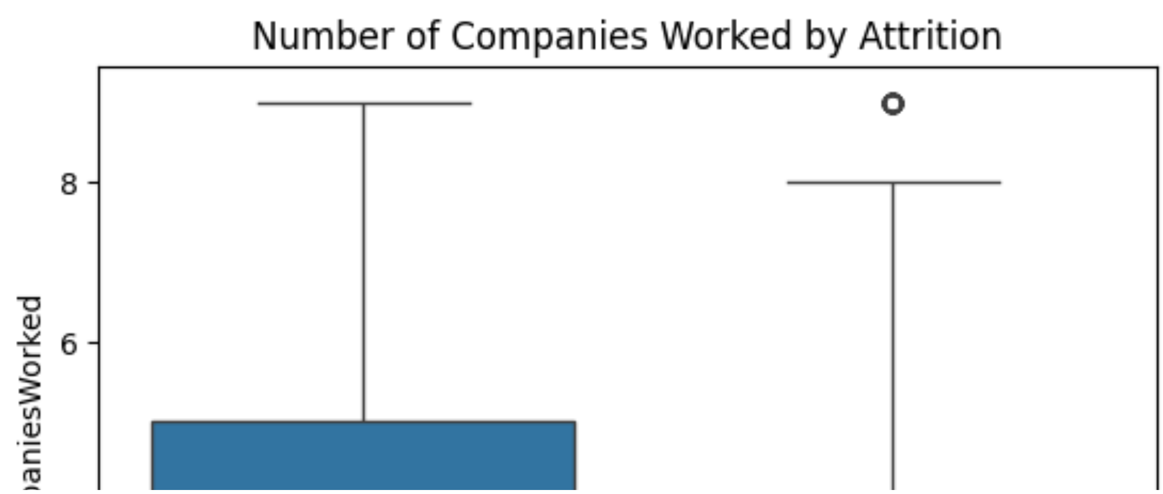

2
0
Yes
No
Attrition

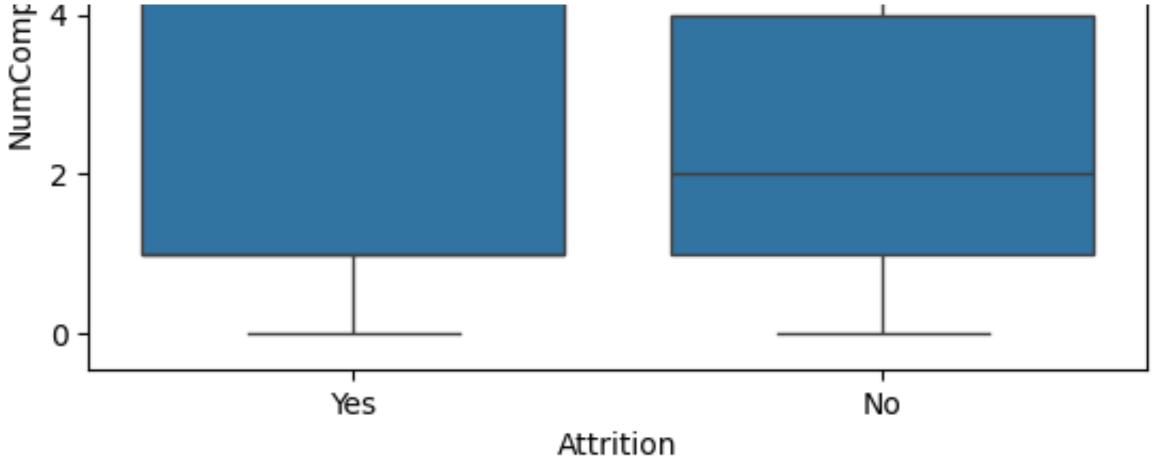

Job Level(1-5 scale) by Attrition
JobLevel
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Yes
No
Attrition

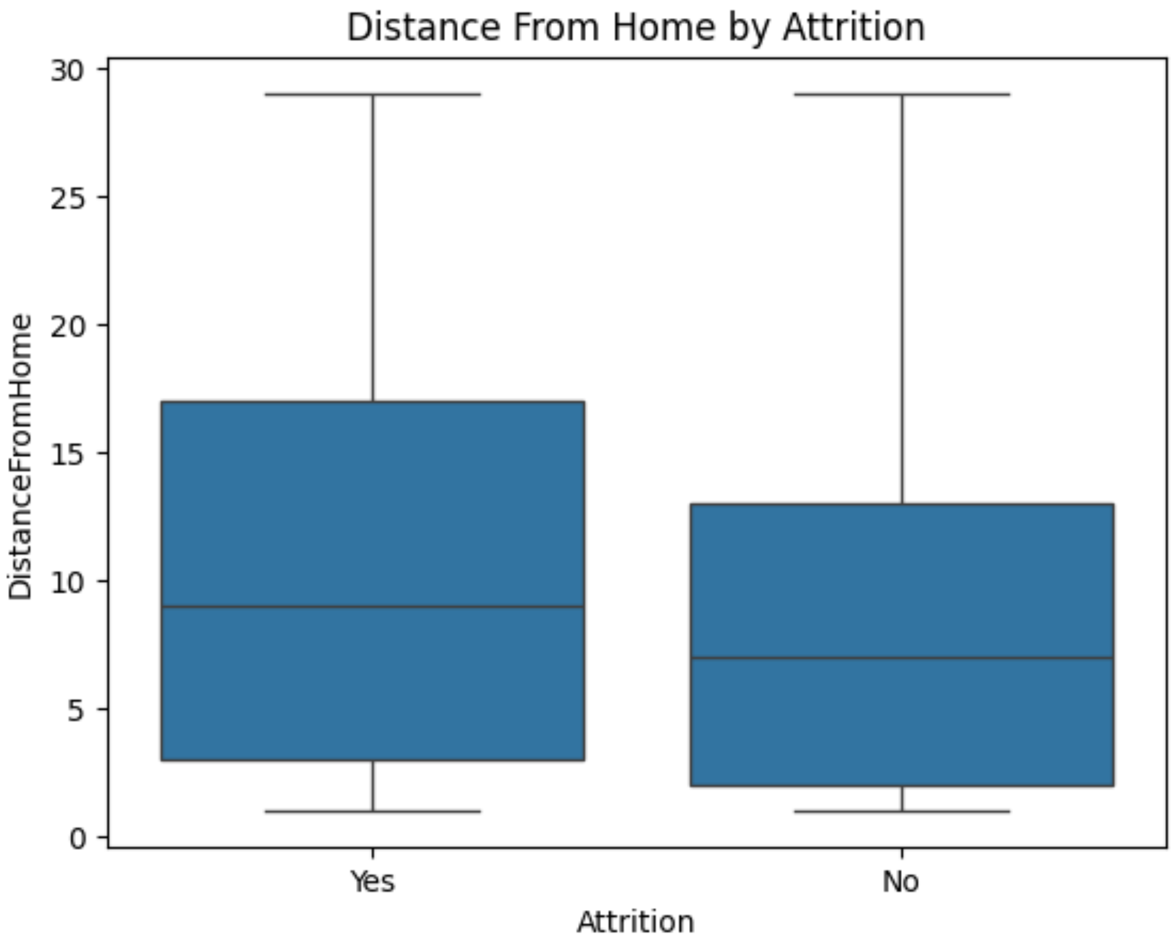

Job Satisfaction(1-5 scale) by Attrition
JobSatisfaction
4.0
3.5
3.0
2.5
2.0

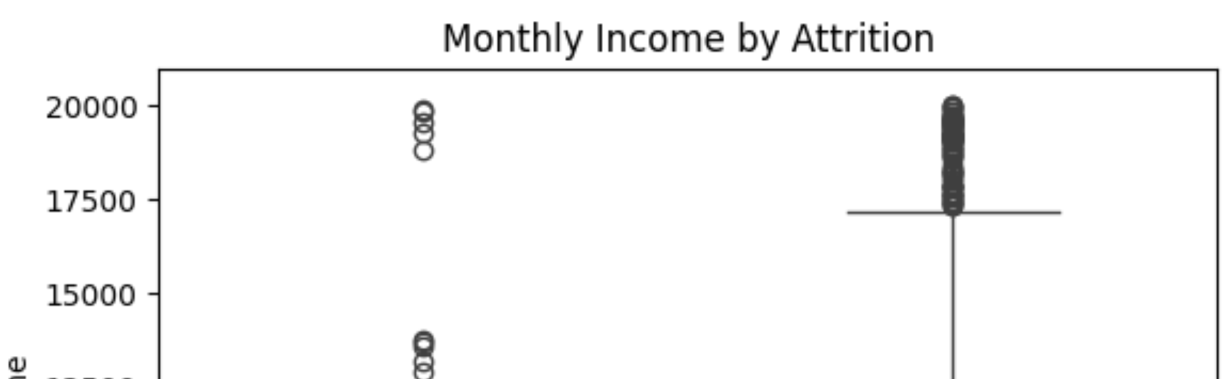

1.5
1.0
Yes
No
Attrition

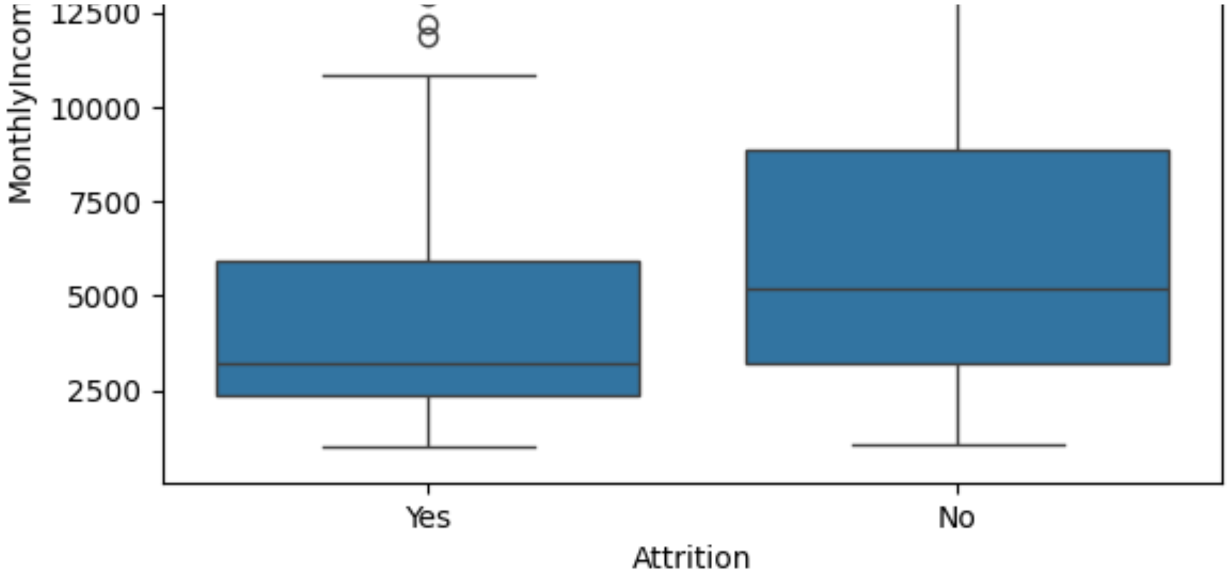

Environment Satisfaction(1-5 scale) by Attrition
4.0
3.5
3.0
2.5
2.0
1.5
1.0
EnvironmentSatisfaction
Yes
No
Attrition


Number of Companies Worked by Attrition
8
6
paniesWorked

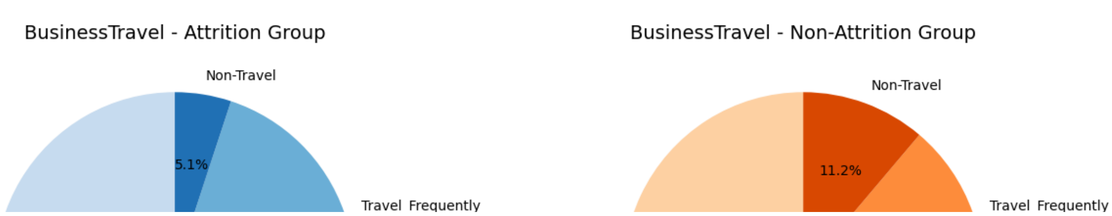

NumComp
4
2
0
Yes
No
Attrition

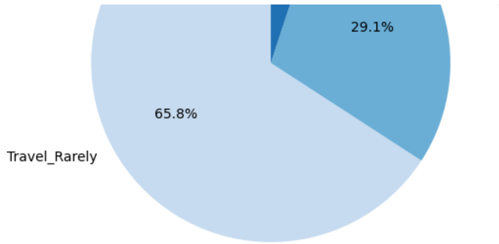

Distance From Home by Attrition
30
25
20
15
10
5
0
DistanceFromHome
Yes
No
Attrition

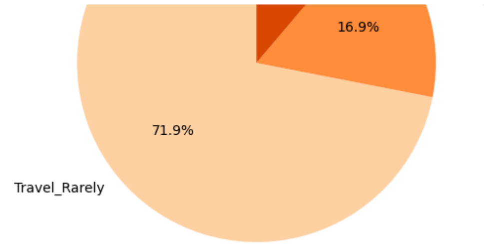

Monthly Income by Attrition
20000
17500
15000



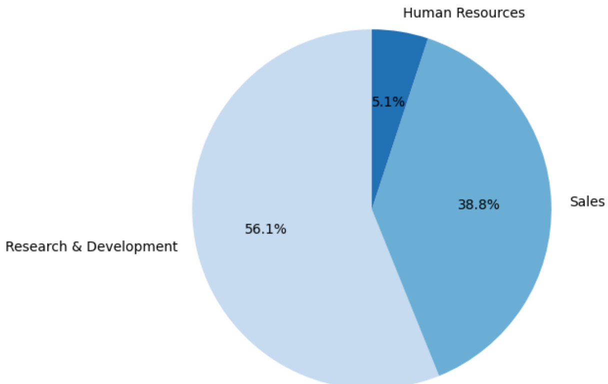

```
# --- Distribution of Key Categorical Figures by Attrition ---
categorical_variables = ['BusinessTravel', 'Department', 'JobRole', 'EducationFie

# Set up the figure for multiple subplots
fig, axes = plt.subplots(len(categorical_variables), 2, figsize=(14, len(categori

# Loop over each categorical variable to plot the dual pie charts
for i, var in enumerate(categorical_variables):
    # Count the occurrences of each category for both attrition and non-attrition
    attrition_counts = df[df['Attrition'] == 'Yes'][var].value_counts()
    non_attrition_counts = df[df['Attrition'] == 'No'][var].value_counts()

    # Plot Pie Chart for Attrition group
    axes[i, 0].pie(attrition_counts, labels=attrition_counts.index, autopct='%1.1
    axes[i, 0].set_title(f'{var} - Attrition Group', fontsize=14)

    # Plot Pie Chart for Non-Attrition group
    axes[i, 1].pie(non_attrition_counts, labels=non_attrition_counts.index, autop
    axes[i, 1].set_title(f'{var} - Non-Attrition Group', fontsize=14)

# Adjust layout to ensure subplots fit without overlapping
plt.tight_layout()
plt.show()
```



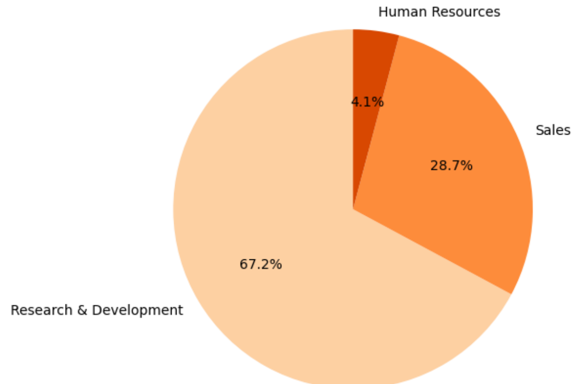




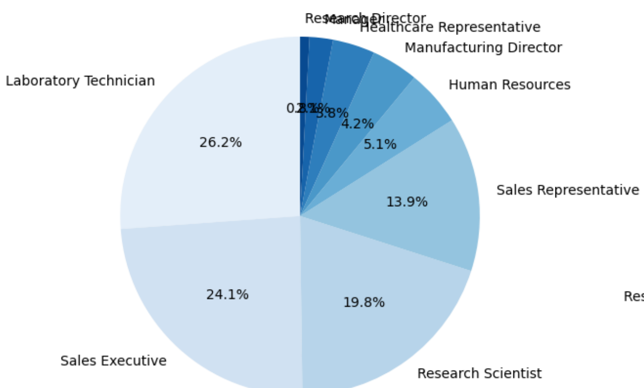
Department - Attrition Group



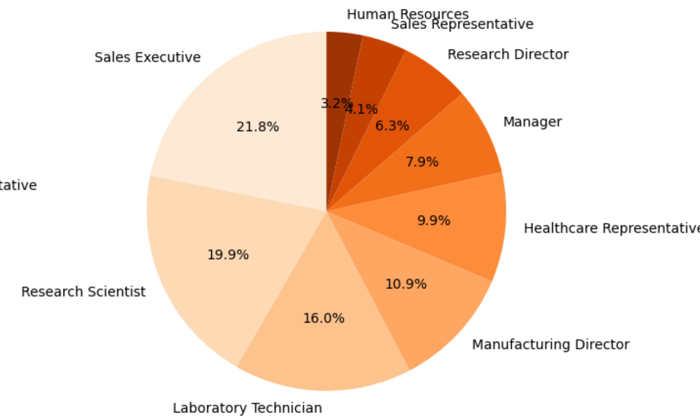
Department - Non-Attrition Group



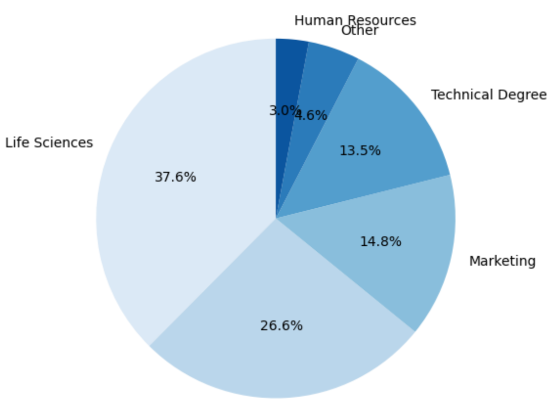
JobRole - Attrition Group



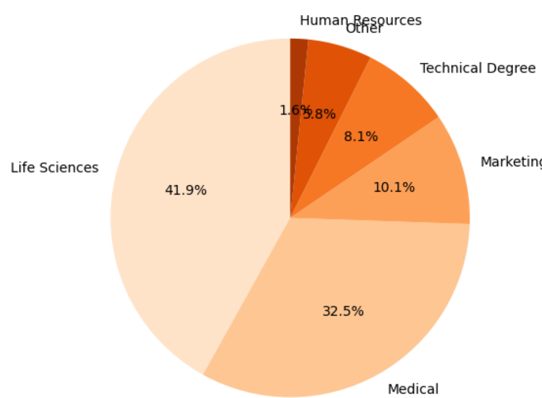
JobRole - Non-Attrition Group



EducationField - Attrition Group

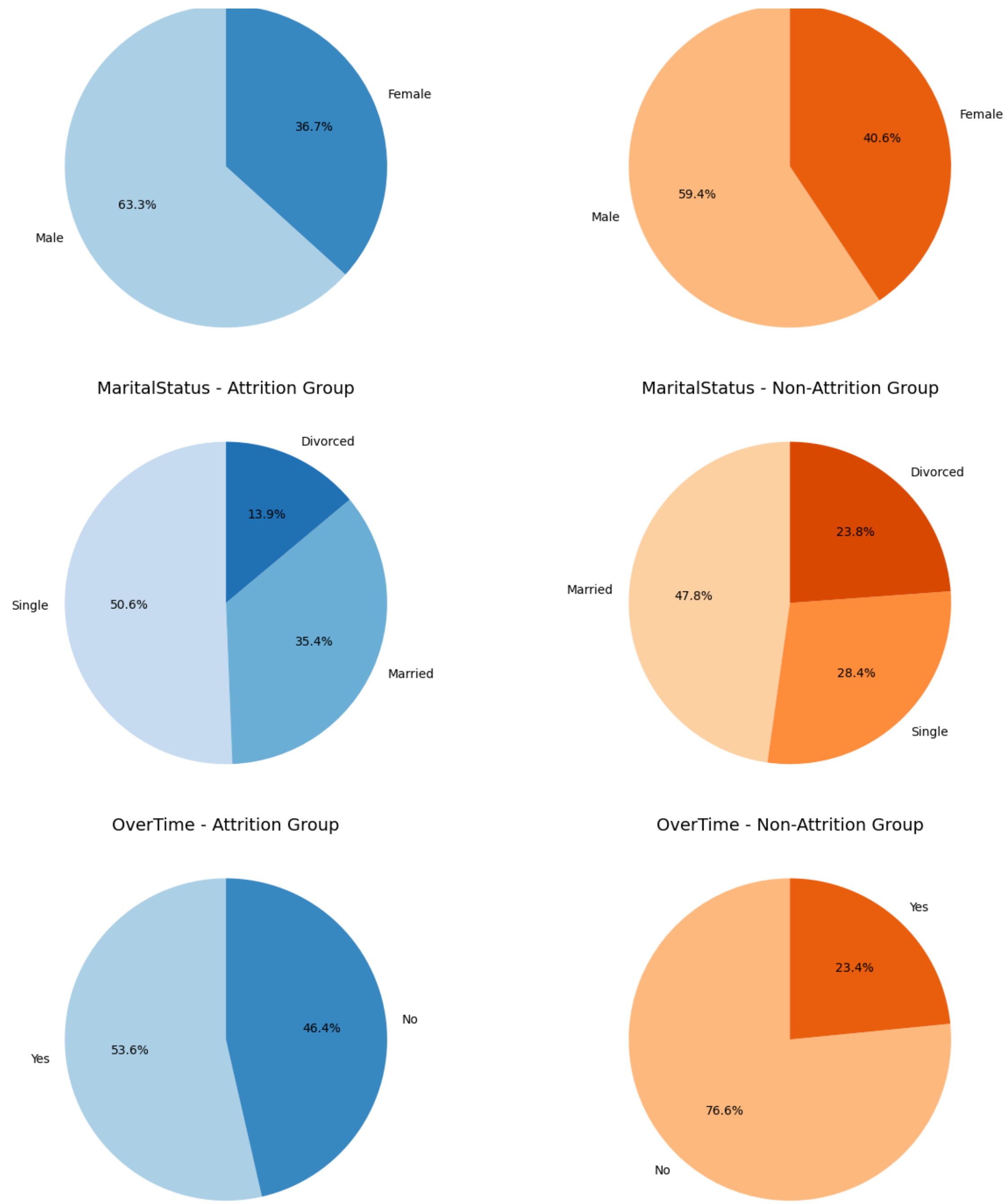


EducationField - Non-Attrition Group



Gender - Attrition Group

Gender - Non-Attrition Group


Female
36.7%
63.3%
Male
Female
40.6%
59.4%
Male
MaritalStatus - Attrition Group
Divorced
13.9%
Single
50.6%
35.4%
Married
MaritalStatus - Non-Attrition Group
Divorced
23.8%
Married
47.8%
28.4%
Single
OverTime - Attrition Group
No
46.4%
Yes
53.6%
OverTime - Non-Attrition Group
Yes
23.4%
76.6%
No

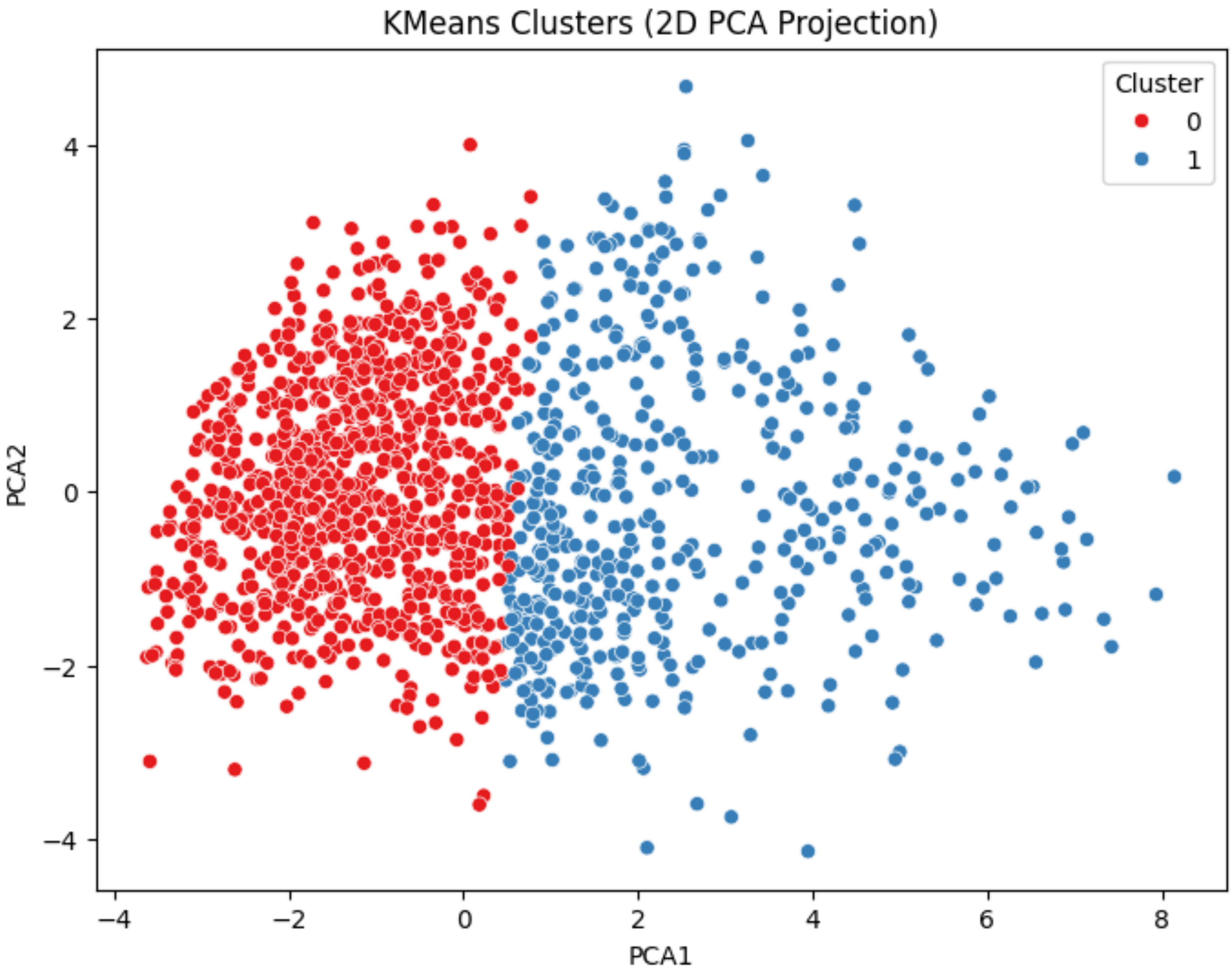
2. Data Preprocessing for Clustering

```
# --- Load Required Libraries ---
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split

# --- Encode Categorical Columns ---
df_encoded = df.copy()
label_cols = df_encoded.select_dtypes(include='object').columns
le = LabelEncoder()
for col in label_cols:
    df_encoded[col] = le.fit_transform(df_encoded[col])

# --- Feature and Target Split ---
X = df_encoded.drop(['Attrition'], axis=1)
y = df_encoded['Attrition']

# --- Feature Scaling ---
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

3. Clustering

```
# --- Load Required Libraries ---
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import numpy as np

# --- Drop target column before clustering ---
X_cluster = pd.DataFrame(X_scaled, columns=X.columns)

# --- Apply KMeans with 2 clusters (likely to match Attrition = Yes/No) ---
kmeans = KMeans(n_clusters=2, random_state=42)
clusters = kmeans.fit_predict(X_cluster)

# --- Attach cluster label ---
df_encoded['Cluster'] = clusters

# --- PCA for 2D projection ---
pca = PCA(n_components=2)
pca_components = pca.fit_transform(X_cluster)
df_encoded['PCA1'] = pca_components[:, 0]
df_encoded['PCA2'] = pca_components[:, 1]

# --- Visualize clusters ---
```

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='PCA1', y='PCA2', hue='Cluster', data=df_encoded, palette='Set1
plt.title('KMeans Clusters (2D PCA Projection)')
plt.show()
```



```
# --- Add original Attrition label back ---
df_encoded['Attrition'] = y.values

# --- Compare attrition rates in each cluster ---
cluster_attrition = df_encoded.groupby('Cluster')['Attrition'].value_counts(norma
print(cluster_attrition)
```

```
Attrition         0         1
Cluster
0          0.804082  0.195918
1          0.908163  0.091837
```

5. Downloading Final Datasets

```
# --- Start from original DataFrame ---
df_export = df.copy()

# --- Add cluster labels ---
df_export['Cluster'] = df_encoded['Cluster']

# --- Add PCA components ---
df_export['PCA1'] = df_encoded['PCA1']
df_export['PCA2'] = df_encoded['PCA2']

# --- Rename and relabel Cluster column ---
df_export.rename(columns={'Cluster': 'AttritionLikelyCluster'}, inplace=True)
df_export['AttritionLikelyCluster'] = df_export['AttritionLikelyCluster'].map({0: '

# --- Save to CSV ---
df_export.to_csv('final_attrition_project_data.csv', index=False)

# --- Download the file ---
files.download('final_attrition_project_data.csv')
```