```
import pandas as pd
```

```
df1 = pd.read_csv('HR_Attrition Data.csv')
```

```
df1.head()
```

Employee No.	Gender	Education	Education Field	Job Role	Department	Age group	Work Location	Attrition	Attrition Date	...	Environment Satisfaction	Attrition Count
E_1	Female	Associates Degree	Life Sciences	Sales Executive	Sales	35-44	Banglore	Yes	4/30/2022	...	2	1.0
E_2	Male	High School	Life Sciences	Research Scientist	R & D	45-55	Hydrabad	No	NaN	...	3	NaN
E_4	Male	Associates Degree	Other	Laboratory Technician	R & D	35-44	Noida	Yes	05-03-2022	...	4	1.0
E_5	Female	Master's Degree	Life Sciences	Research Scientist	R & D	25-34	Pune	No	NaN	...	4	NaN
E_7	Male	High School	Medical	Laboratory Technician	R & D	25-34	Mumbai	No	NaN	...	1	NaN

ws × 26 columns

```
df2 = pd.read_csv('HR_New_data.csv')
```

```
df2.head()
```

	emp_id	last_evaluation	number_project	average_montly_hours	Work_accident	promotion_last_5years
0	E_1	0.53	2	157	0	0
1	E_4	0.86	5	262	0	0
2	E_19	0.88	7	272	0	0
3	E_27	0.87	5	223	0	0
4	E_31	0.52	2	159	0	0

Next steps: Generate code with df2 | View recommended plots | New interactive sheet

```
# Merging both datasets

df = pd.merge(left=df1,right=df2,left_on="Employee No.",right_on="emp_id")
```

```
df.head()
```

	Employee No.	Gender	Education	Education Field	Job Role	Department	Age group	Work Location	Attrition	Attrition Date	...	Performance Rating	Placeholde
0	E_1	Female	Associates Degree	Life Sciences	Sales Executive	Sales	35-44	Banglore	Yes	4/30/2022	...	3	
1	E_2	Male	High School	Life Sciences	Research Scientist	R & D	45-55	Hydrabad	No	NaN	...	4	
2	E_4	Male	Associates Degree	Other	Laboratory Technician	R & D	35-44	Noida	Yes	05-03-2022	...	3	
3	E_5	Female	Master's Degree	Life Sciences	Research Scientist	R & D	25-34	Pune	No	NaN	...	3	
4	E_7	Male	High School	Medical	Laboratory Technician	R & D	25-34	Mumbai	No	NaN	...	3	

5 rows × 32 columns

```
df.columns
```

```
Index(['Employee No.', 'Gender', 'Education', 'Education Field', 'Job Role',
       'Department', 'Age group', 'Work Location', 'Attrition',
       'Attrition Date', 'Attrition Label', 'Dynamic Date', 'Age',
       'Attrition.1', 'Avg. Satisfaction Score', 'Employee Count',
```

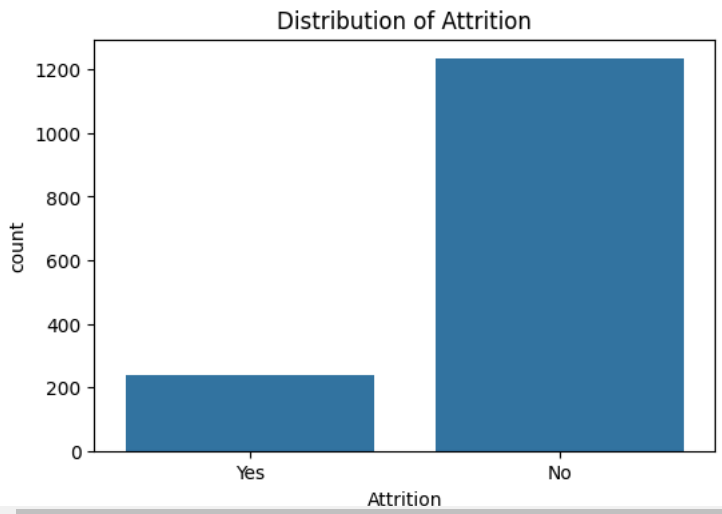
```
       'Environment Satisfaction', 'Attrition Count', 'Job Involvement',
       'Job Satisfaction', 'Monthly Income', 'Percent Salary Hike',
       'Performance Rating', 'Placeholder', 'Relationship Satisfaction',
       'Work Life Balance', 'emp_id', 'last_evaluation', 'number_project',
       'average_montly_hours', 'Work_accident', 'promotion_last_5years'],
      dtype='object')
```

Exploratry Data Analysis

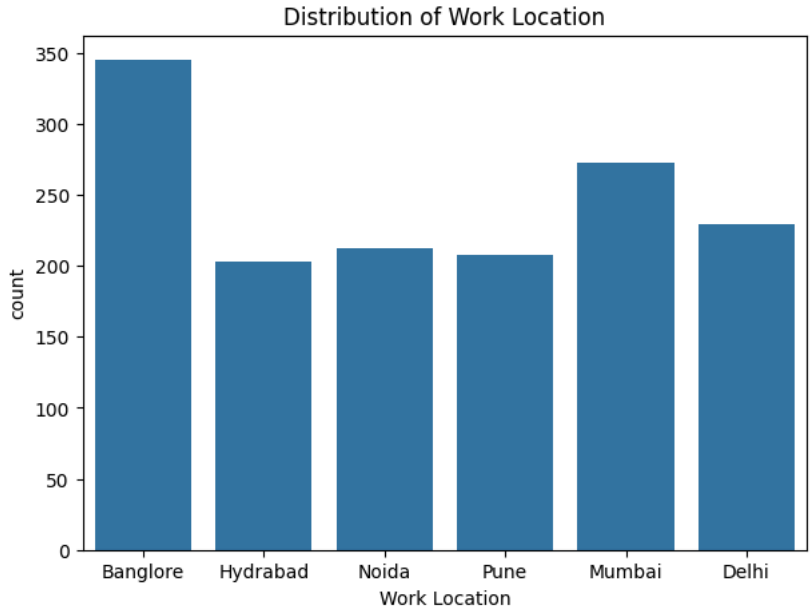
```
# Checking Target Variable

df['Attrition'].value_counts()
```

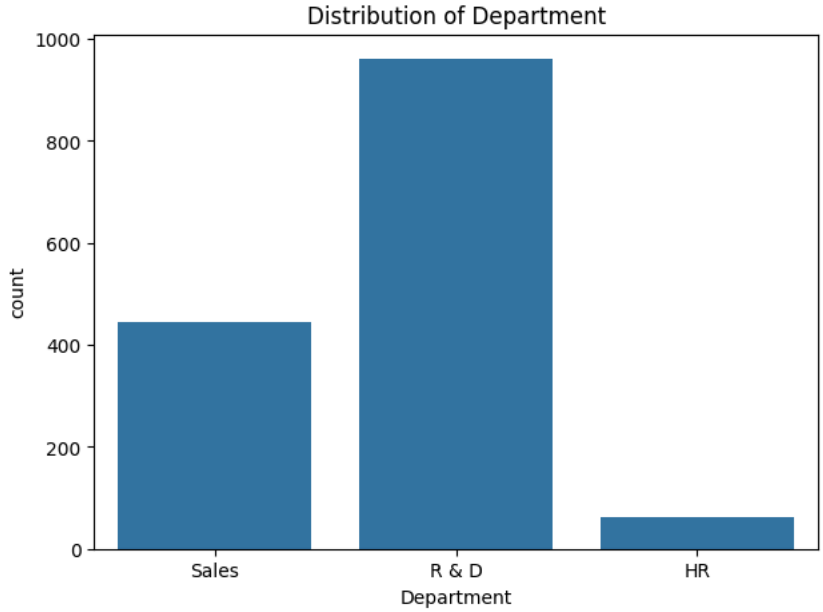
Attrition	count
No	1233
Yes	237

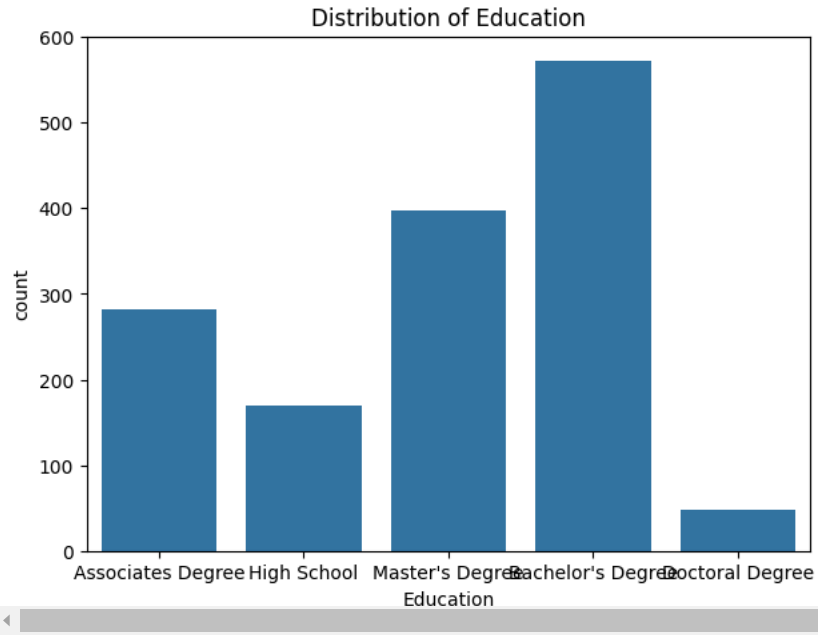
```
import matplotlib.pyplot as plt
import seaborn as sns


# Plot distribution of Attrition (target variable)
plt.figure(figsize=(6,4))
sns.countplot(x='Attrition', data=df)
plt.title('Distribution of Attrition')
plt.show()
```




```
# Plot distribution of a few key categorical variables
categorical_cols1 = ['Work Location', 'Department', 'Education']
for col in categorical_cols1:
    plt.figure(figsize=(7,5))
    sns.countplot(x=col, data=df)
    plt.title(f'Distribution of {col}')
    plt.show()
```



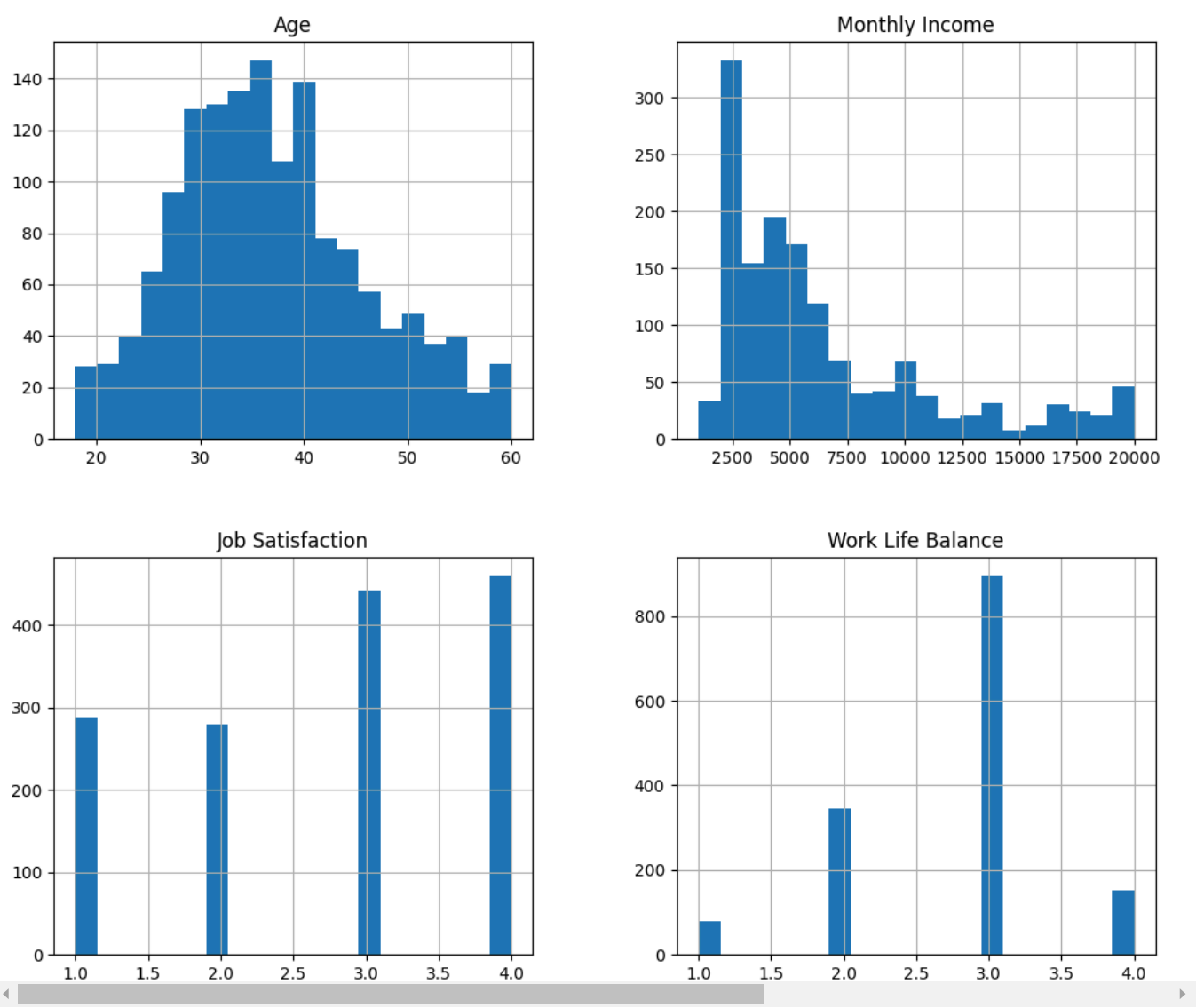





```
# Plot distribution of a few key categorical variables
categorical_cols2 = ['Job Role']
for col in categorical_cols2:
    plt.figure(figsize=(14,5))
    sns.countplot(x=col, data=df)
```

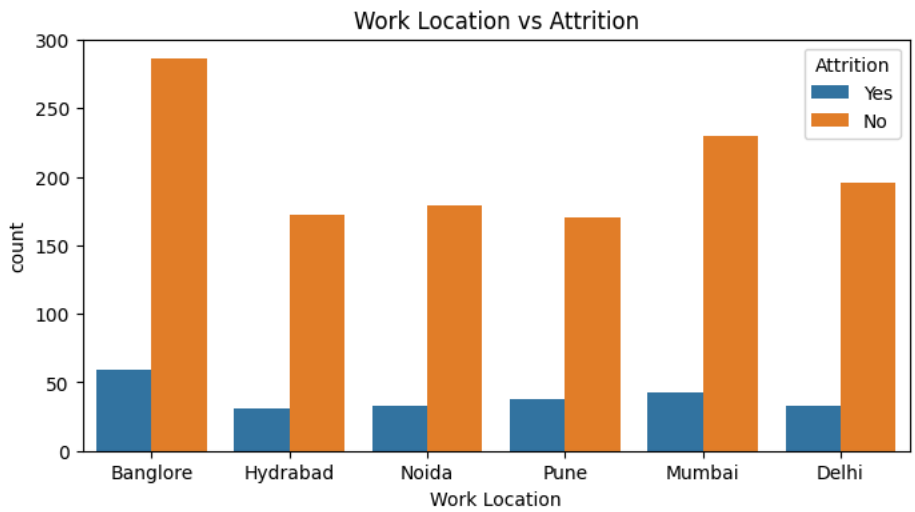
```
plt.title(f'Distribution of {col}')
plt.show()
```

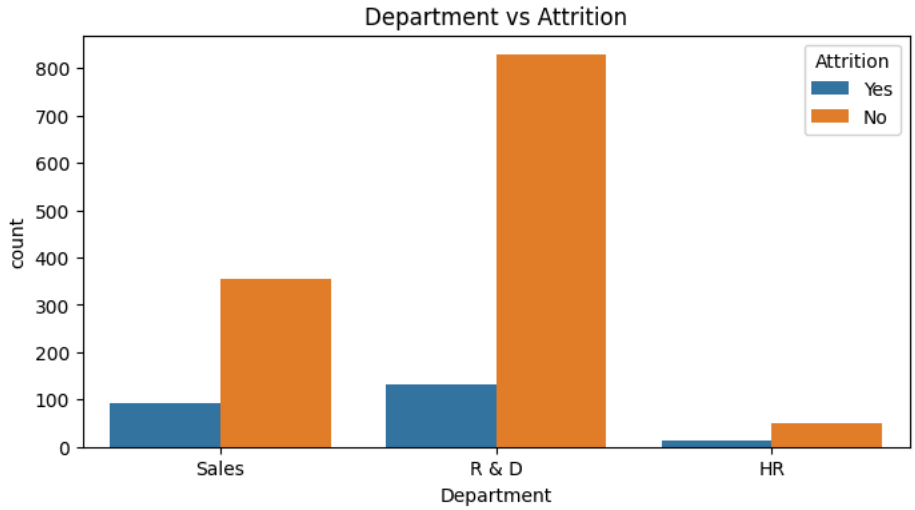

```
# Histograms for numerical features
numerical_cols = ['Age', 'Monthly Income', 'Job Satisfaction', 'Work Life Balance']
df[numerical_cols].hist(bins=20, figsize=(12, 10), layout=(2, 2))
plt.show()
```

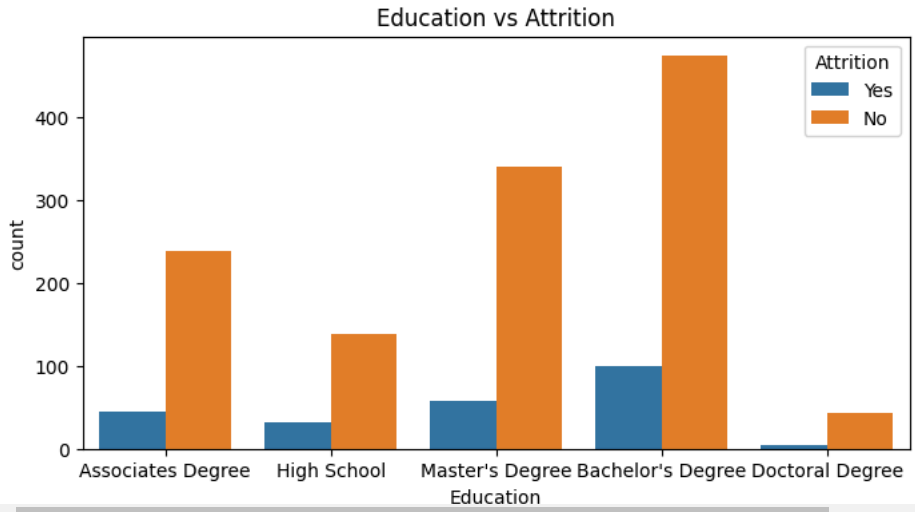



```
# Categorical vs Target (Attrition)
for col in categorical_cols1:
```

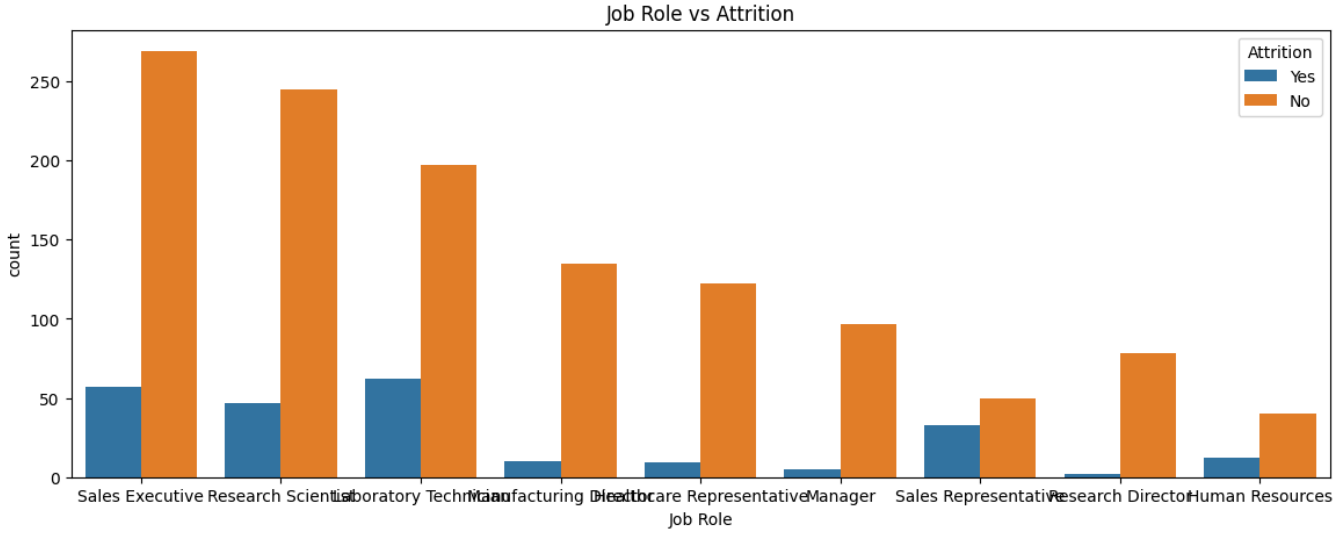
```
    plt.figure(figsize=(8,4))
    sns.countplot(x=col, hue='Attrition', data=df)
    plt.title(f'{col} vs Attrition')
    plt.show()
```



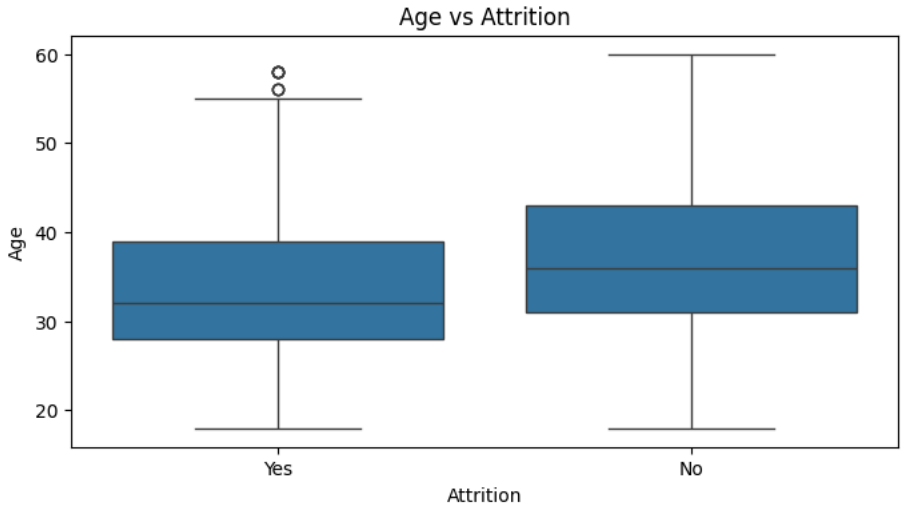


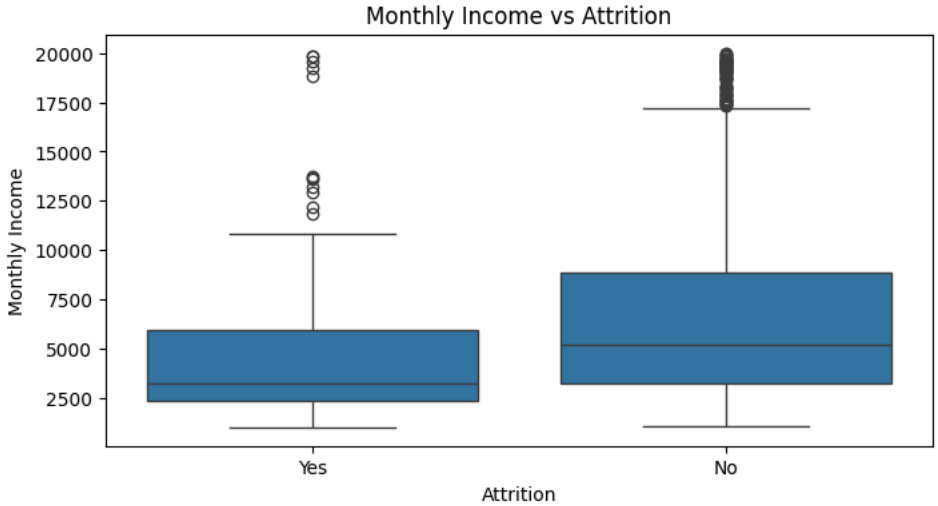


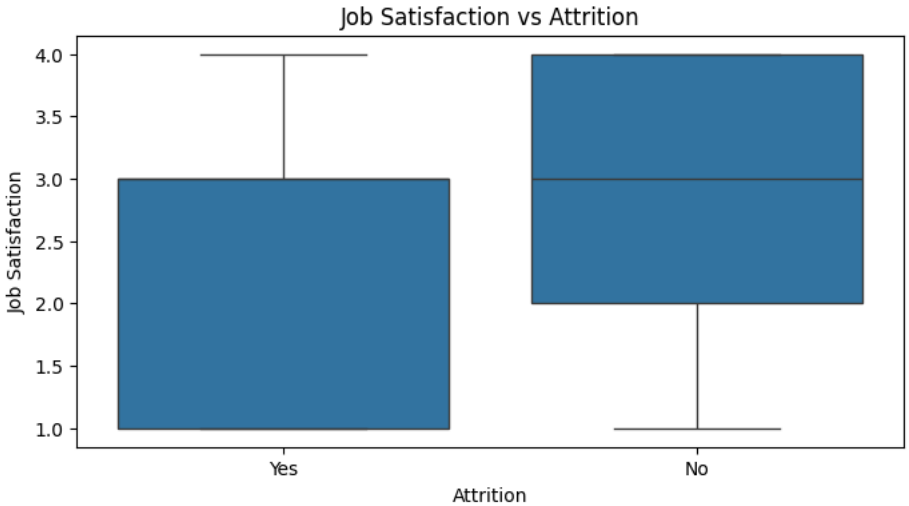

```
# Categorical vs Target (Attrition)
for col in categorical_cols2:
    plt.figure(figsize=(14,5))
    sns.countplot(x=col, hue='Attrition', data=df)
    plt.title(f'{col} vs Attrition')
    plt.show()
```

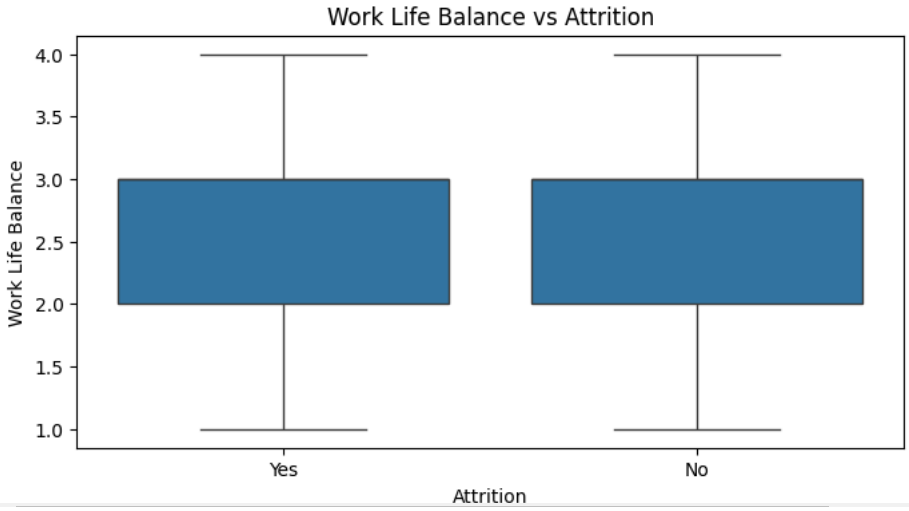



```
# Boxplots for numerical features vs Attrition
for col in numerical_cols:
    plt.figure(figsize=(8,4))
    sns.boxplot(x='Attrition', y=col, data=df)
    plt.title(f'{col} vs Attrition')
    plt.show()
```

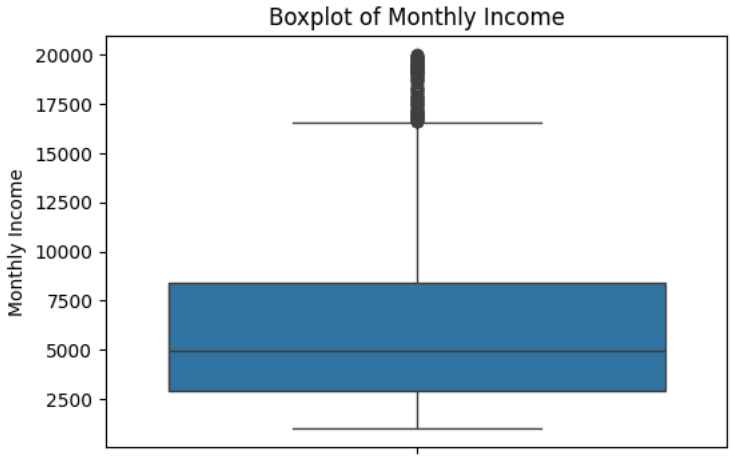

Age vs Attrition
Age
60
50
40
30
20
Yes
No
Attrition

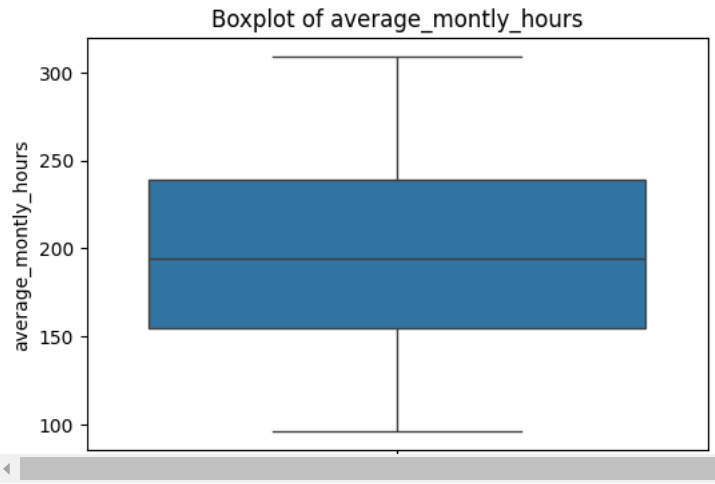

Monthly Income vs Attrition
Monthly Income
20000
17500
15000
12500
10000
7500
5000
2500
Yes
No
Attrition


Job Satisfaction vs Attrition
Job Satisfaction
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Yes
No
Attrition


Work Life Balance vs Attrition
Work Life Balance
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Yes
No
Attrition

```
# Detecting outliers using boxplots
for col in ['Monthly Income', 'average_montly_hours']:
    plt.figure(figsize=(6,4))
    sns.boxplot(data=df[col])
    plt.title(f'Boxplot of {col}')
    plt.show()
```





Data Preprocessing

```
# Check Missing Values
df.isna().sum()
```

	0
Employee No.	0
Gender	0
Education	0
Education Field	0
Job Role	0
Department	0
Age group	0
Work Location	0
Attrition	0
Attrition Date	1233
Attrition Label	0
Dynamic Date	1233
Age	0
Attrition.1	0
Avg. Satisfaction Score	0
Employee Count	0
Environment Satisfaction	0
Attrition Count	1233
Job Involvement	0
Job Satisfaction	0
Monthly Income	0
Percent Salary Hike	0
Performance Rating	0
Placeholder	0
Relationship Satisfaction	0
Work Life Balance	0
emp_id	0
last_evaluation	0
number_project	0
average_montly_hours	0
Work_accident	0
promotion_last_5years	0

```
df = df[['Gender', 'Education', 'Education Field', 'Job Role',
       'Department', 'Work Location','Attrition', 'Age',
       'Avg. Satisfaction Score',
       'Environment Satisfaction', 'Job Involvement',
       'Job Satisfaction', 'Monthly Income', 'Percent Salary Hike',
       'Performance Rating', 'Relationship Satisfaction',
       'Work Life Balance','last_evaluation', 'number_project',
       'average_montly_hours', 'Work_accident', 'promotion_last_5years']]
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 22 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Gender                     1470 non-null   object
 1   Education                  1470 non-null   object
 2   Education Field            1470 non-null   object
 3   Job Role                   1470 non-null   object
 4   Department                 1470 non-null   object
 5   Work Location              1470 non-null   object
 6   Attrition                  1470 non-null   object
```

```
 7   Age                       1470 non-null   int64
 8   Avg. Satisfaction Score   1470 non-null   float64
 9   Environment Satisfaction  1470 non-null   int64
 10  Job Involvement           1470 non-null   int64
 11  Job Satisfaction          1470 non-null   int64
 12  Monthly Income            1470 non-null   int64
 13  Percent Salary Hike       1470 non-null   int64
 14  Performance Rating        1470 non-null   int64
 15  Relationship Satisfaction 1470 non-null   int64
 16  Work Life Balance         1470 non-null   int64
 17  last_evaluation           1470 non-null   float64
 18  number_project            1470 non-null   int64
 19  average_montly_hours      1470 non-null   int64
 20  Work_accident             1470 non-null   int64
 21  promotion_last_5years     1470 non-null   int64
dtypes: float64(2), int64(13), object(7)
memory usage: 252.8+ KB
```

```
df.head()
```

	Gender	Education	Education Field	Job Role	Department	Work Location	Attrition	Age	Avg. Satisfaction Score	Environment Satisfaction	...	Monthly Income	Percent Salary Hike
0	Female	Associates Degree	Life Sciences	Sales Executive	Sales	Banglore	Yes	41	2.2	2	...	5993	11
1	Male	High School	Life Sciences	Research Scientist	R & D	Hydrabad	No	49	2.8	3	...	5130	23
2	Male	Associates Degree	Other	Laboratory Technician	R & D	Noida	Yes	37	2.8	4	...	2090	15
3	Female	Master's Degree	Life Sciences	Research Scientist	R & D	Pune	No	33	3.2	4	...	2909	11
4	Male	High School	Medical	Laboratory Technician	R & D	Mumbai	No	27	2.6	1	...	3468	12

5 rows × 22 columns

```
#Drop rows with missing values
df.dropna(inplace=True)
```

```
# One-hot encoding for categorical variables
df = pd.get_dummies(df, columns=['Gender', 'Education', 'Education Field', 'Job Role', 'Department', 'Work Location'], drop_first=True)
```

```
df.head()
```

	Attrition	Age	Avg. Satisfaction Score	Environment Satisfaction	Job Involvement	Job Satisfaction	Monthly Income	Percent Salary Hike	Performance Rating	Relationship Satisfaction	...	Role_Rese Scier
0	Yes	41	2.2	2	3	4	5993	11	3	1	...	
1	No	49	2.8	3	2	2	5130	23	4	4	...	
2	Yes	37	2.8	4	2	3	2090	15	3	2	...	
3	No	33	3.2	4	3	3	2909	11	3	3	...	
4	No	27	2.6	1	3	2	3468	12	3	4	...	

5 rows × 41 columns

```
# Convert binary columns to 0 and 1
df['Attrition'] = df['Attrition'].apply(lambda x: 1 if x == 'Yes' else 0)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 41 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Attrition                       1470 non-null   int64
 1   Age                             1470 non-null   int64
 2   Avg. Satisfaction Score         1470 non-null   float64
 3   Environment Satisfaction        1470 non-null   int64
 4   Job Involvement                 1470 non-null   int64
 5   Job Satisfaction                1470 non-null   int64
```

```
 6   Monthly Income                    1470 non-null   int64
 7   Percent Salary Hike               1470 non-null   int64
 8   Performance Rating                1470 non-null   int64
 9   Relationship Satisfaction         1470 non-null   int64
 10  Work Life Balance                 1470 non-null   int64
 11  last_evaluation                   1470 non-null   float64
 12  number_project                    1470 non-null   int64
 13  average_montly_hours              1470 non-null   int64
 14  Work_accident                     1470 non-null   int64
 15  promotion_last_5years             1470 non-null   int64
 16  Gender_Male                       1470 non-null   bool
 17  Education_Bachelor's Degree       1470 non-null   bool
 18  Education_Doctoral Degree         1470 non-null   bool
 19  Education_High School             1470 non-null   bool
 20  Education_Master's Degree         1470 non-null   bool
 21  Education Field_Life Sciences     1470 non-null   bool
 22  Education Field_Marketing         1470 non-null   bool
 23  Education Field_Medical           1470 non-null   bool
 24  Education Field_Other             1470 non-null   bool
 25  Education Field_Technical Degree  1470 non-null   bool
 26  Job Role_Human Resources          1470 non-null   bool
 27  Job Role_Laboratory Technician    1470 non-null   bool
 28  Job Role_Manager                  1470 non-null   bool
 29  Job Role_Manufacturing Director   1470 non-null   bool
 30  Job Role_Research Director        1470 non-null   bool
 31  Job Role_Research Scientist       1470 non-null   bool
 32  Job Role_Sales Executive          1470 non-null   bool
 33  Job Role_Sales Representative     1470 non-null   bool
 34  Department_R & D                  1470 non-null   bool
 35  Department_Sales                  1470 non-null   bool
 36  Work Location_Delhi               1470 non-null   bool
 37  Work Location_Hydrabad            1470 non-null   bool
 38  Work Location_Mumbai              1470 non-null   bool
 39  Work Location_Noida               1470 non-null   bool
 40  Work Location_Pune                1470 non-null   bool
dtypes: bool(25), float64(2), int64(14)
memory usage: 219.8 KB
```

```
df.describe()
```

	Attrition	Age	Avg. Satisfaction Score	Environment Satisfaction	Job Involvement	Job Satisfaction	Monthly Income	Percent Salary Hike	Performance Rating	Relati Satisf
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.
mean	0.161224	36.923810	2.730748	2.721769	2.729932	2.728571	6502.931293	15.209524	3.153741	2.
std	0.367863	9.135373	0.428551	1.093082	0.711561	1.102846	4707.956783	3.659938	0.360824	1.
min	0.000000	18.000000	1.200000	1.000000	1.000000	1.000000	1009.000000	11.000000	3.000000	1.
25%	0.000000	30.000000	2.400000	2.000000	2.000000	2.000000	2911.000000	12.000000	3.000000	2.
50%	0.000000	36.000000	2.800000	3.000000	3.000000	3.000000	4919.000000	14.000000	3.000000	3.
75%	0.000000	43.000000	3.000000	4.000000	3.000000	4.000000	8379.000000	18.000000	3.000000	4.
max	1.000000	60.000000	4.000000	4.000000	4.000000	4.000000	19999.000000	25.000000	4.000000	4.

```
from sklearn.preprocessing import StandardScaler

# Scaling numerical features
scaler = StandardScaler()
numerical_features = ['Age', 'Monthly Income', 'Percent Salary Hike', 'average_montly_hours', 'last_evaluation', 'number_project']
df[numerical_features] = scaler.fit_transform(df[numerical_features])
```

Model Building

```
from sklearn.model_selection import train_test_split

# Define features and target variable
X = df.drop(columns=['Attrition'])
y = df['Attrition']

# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Model Training

```
from sklearn.ensemble import RandomForestClassifier

# Initialize and train the model
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

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

Model Evaluation

```
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, roc_auc_score

# Predictions on test set
y_pred = model.predict(X_test)

# Evaluation metrics
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
print("ROC-AUC Score:", roc_auc_score(y_test, model.predict_proba(X_test)[:, 1]))
```

```
Accuracy: 0.9455782312925171
Confusion Matrix:
 [[372   4]
 [ 20  45]]
Classification Report:
               precision    recall  f1-score   support

           0       0.95      0.99      0.97       376
           1       0.92      0.69      0.79        65

    accuracy                           0.95       441
   macro avg       0.93      0.84      0.88       441
weighted avg       0.94      0.95      0.94       441

ROC-AUC Score: 0.947340425531915
```

Model Export

```
import joblib

# Save the trained model
joblib.dump(model, 'employee_attrition_model.pkl')
```

```
['employee_attrition_model.pkl']
```

Deployment and Reporting

```
# Load the model
loaded_model = joblib.load('employee_attrition_model.pkl')
```

```
import numpy as np
import pandas as pd

# Example new data (for 5 employees)
new_employee_data = {
    'Age': [28, 35, 40, 50, 25],
    'Avg. Satisfaction Score': [3.5, 4.2, 3.0, 4.0, 3.8],
    'Environment Satisfaction': [3, 4, 2, 4, 3],
    'Job Involvement': [3, 3, 4, 3, 2],
```