```python
# Import necessary libraries for data analysis and visualization
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import urllib.request

# Download dataset
url = "https://sp8138-heart-attack-dataset.s3.us-east-
2.amazonaws.com/heart_attack_data.csv"
local_file = "heart_attack_data.csv"
urllib.request.urlretrieve(url, local_file)
```

```
('heart_attack_data.csv', <http.client.HTTPMessage at 0x7e8d87b8d5d0>)
```

```python
# Load the dataset into a pandas DataFrame
df = pd.read_csv(local_file)

# Display basic information about the dataset
print("===== DATASET INFO =====")
print(df.info())
print("\n===== FIRST 5 ROWS =====")
display(df.head())
```

```
===== DATASET INFO =====
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 372974 entries, 0 to 372973
Data columns (total 32 columns):
 #   Column                 Non-Null Count   Dtype
---  ------                 --------------   -----
 0   Age                    372974 non-null  int64
 1   Gender                 372974 non-null  object
 2   Cholesterol            372974 non-null  int64
 3   BloodPressure          372974 non-null  int64
 4   HeartRate              372974 non-null  int64
 5   BMI                    372974 non-null  float64
 6   Smoker                 372974 non-null  int64
 7   Diabetes               372974 non-null  int64
 8   Hypertension           372974 non-null  int64
 9   FamilyHistory          372974 non-null  int64
 10  PhysicalActivity       372974 non-null  int64
 11  AlcoholConsumption     372974 non-null  int64
 12  Diet                   372974 non-null  object
 13  StressLevel            372974 non-null  int64
 14  Ethnicity              372974 non-null  object
 15  Income                 372974 non-null  int64
 16  EducationLevel         372974 non-null  object
 17  Medication             372974 non-null  object
 18  ChestPainType          372974 non-null  object
 19  ECGResults             372974 non-null  object
```

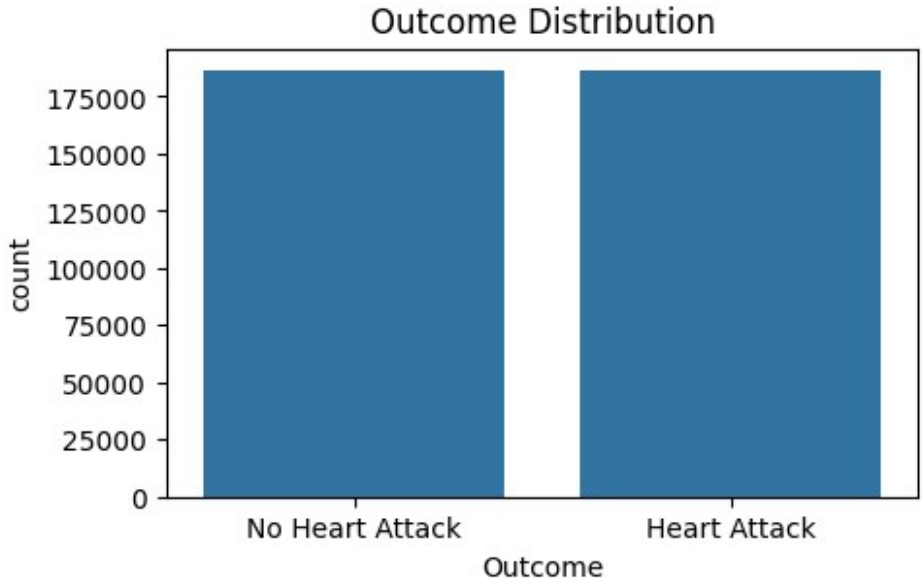
```
 20  MaxHeartRate            372974 non-null  int64
 21  ST_Depression           372974 non-null  float64
 22  ExerciseInducedAngina   372974 non-null  object
 23  Slope                   372974 non-null  object
 24  NumberOfMajorVessels    372974 non-null  int64
 25  Thalassemia             372974 non-null  object
 26  PreviousHeartAttack     372974 non-null  int64
 27  StrokeHistory           372974 non-null  int64
 28  Residence               372974 non-null  object
 29  EmploymentStatus        372974 non-null  object
 30  MaritalStatus           372974 non-null  object
 31  Outcome                 372974 non-null  object
dtypes: float64(2), int64(16), object(14)
memory usage: 91.1+ MB
None

===== FIRST 5 ROWS =====

{"type":"dataframe"}
```

```
# Check for missing values in the dataset
print("\n===== MISSING VALUES =====")
print(df.isnull().sum())
```

```
===== MISSING VALUES =====
Age                       0
Gender                    0
Cholesterol               0
BloodPressure             0
HeartRate                 0
BMI                       0
Smoker                    0
Diabetes                  0
Hypertension              0
FamilyHistory             0
PhysicalActivity          0
AlcoholConsumption        0
Diet                      0
StressLevel               0
Ethnicity                 0
Income                    0
EducationLevel            0
Medication                0
ChestPainType             0
ECGResults                0
MaxHeartRate              0
ST_Depression             0
ExerciseInducedAngina     0
Slope                     0
```

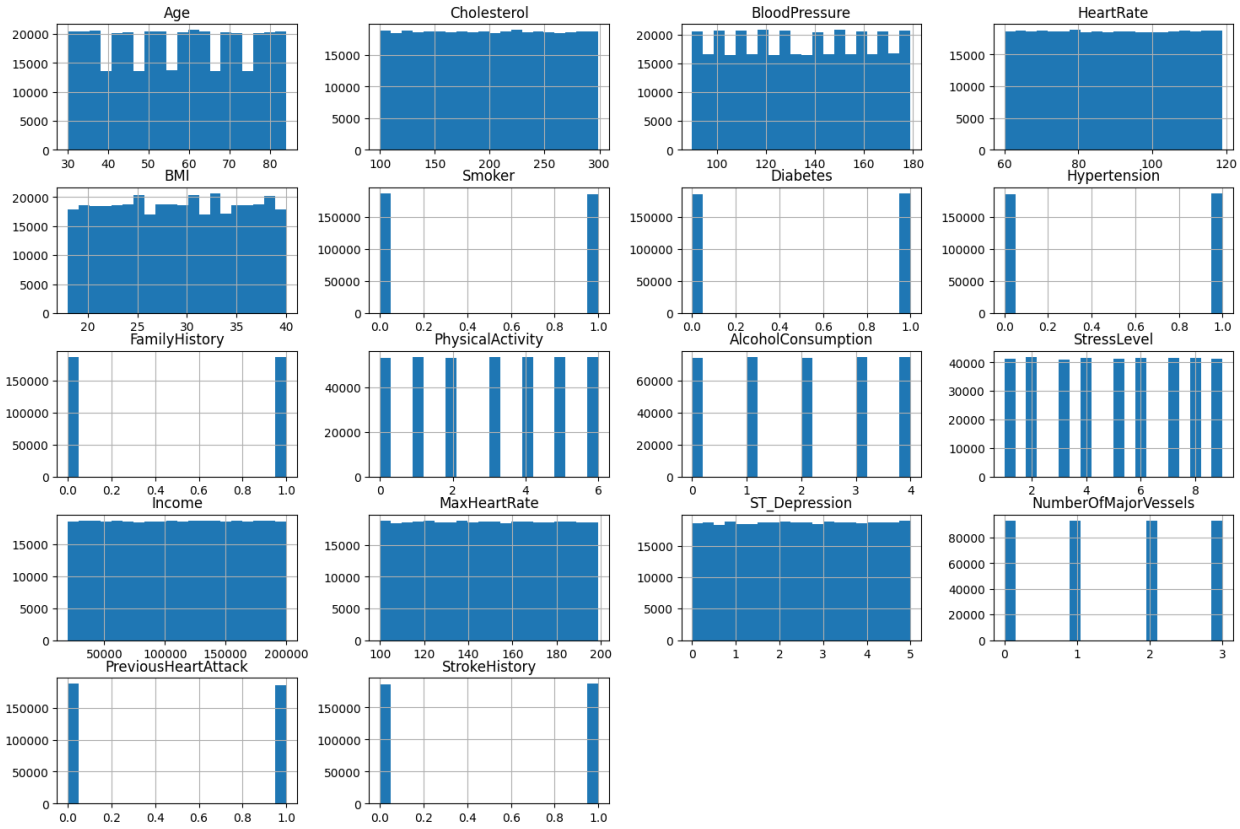
```
NumberOfMajorVessels     0
Thalassemia              0
PreviousHeartAttack      0
StrokeHistory            0
Residence                0
EmploymentStatus         0
MaritalStatus            0
Outcome                  0
dtype: int64
```

```
# Visualize the distribution of the outcome variable
plt.figure(figsize=(5,3))
sns.countplot(data=df, x="Outcome")
plt.title("Outcome Distribution")
plt.show()
print(df['Outcome'].value_counts())
```

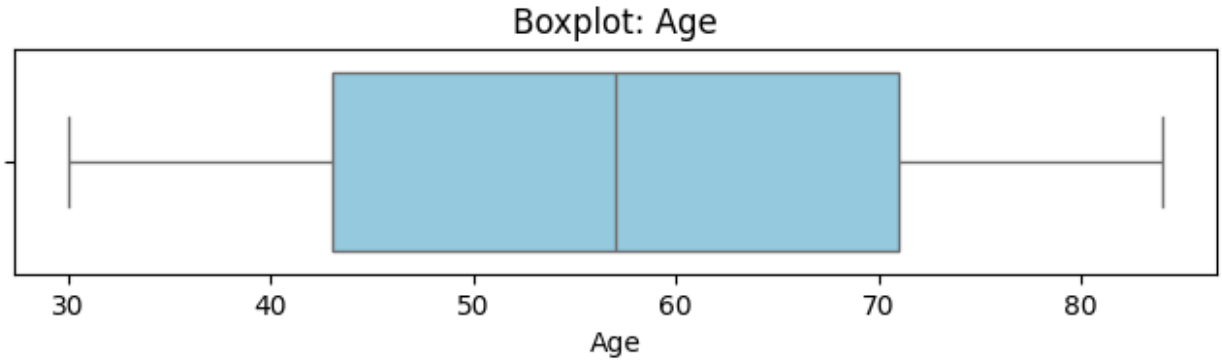


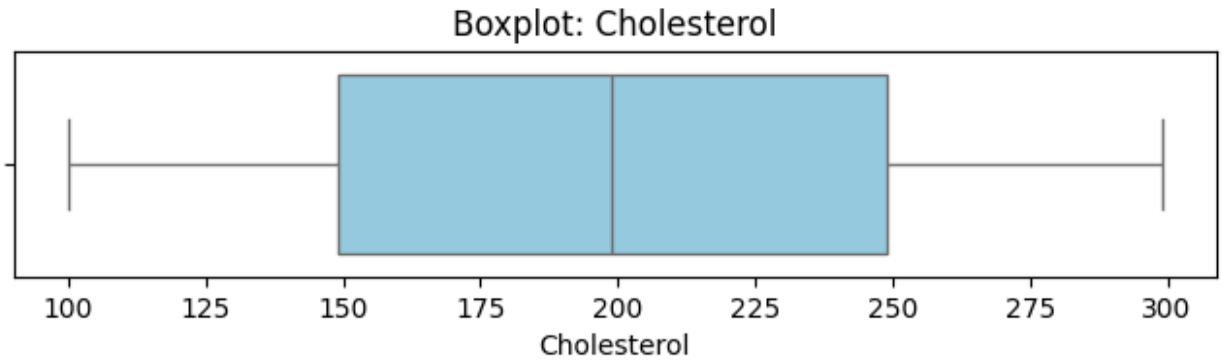

```
Outcome
No Heart Attack    186658
Heart Attack       186316
Name: count, dtype: int64
```

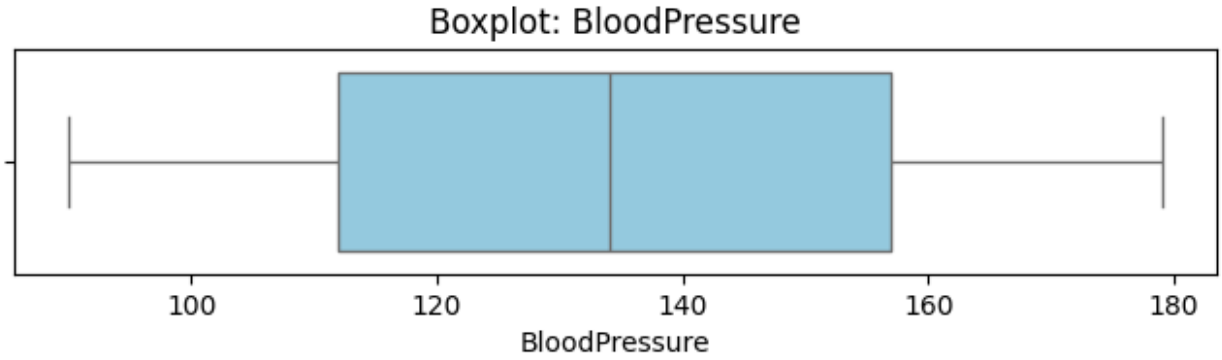
```
# Plot histograms for all numeric features
num_cols = df.select_dtypes(include=[np.number]).columns
df[num_cols].hist(figsize=(18, 12), bins=20)
plt.suptitle("Numeric Feature Distributions")
plt.show()
```

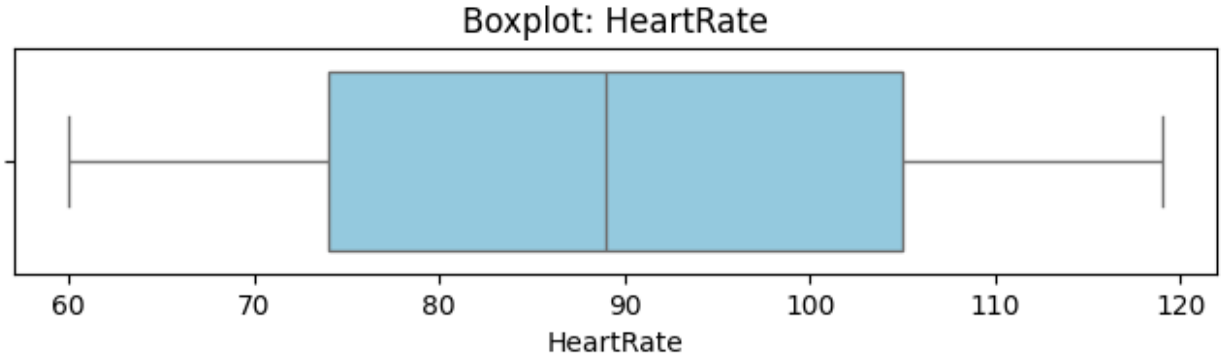


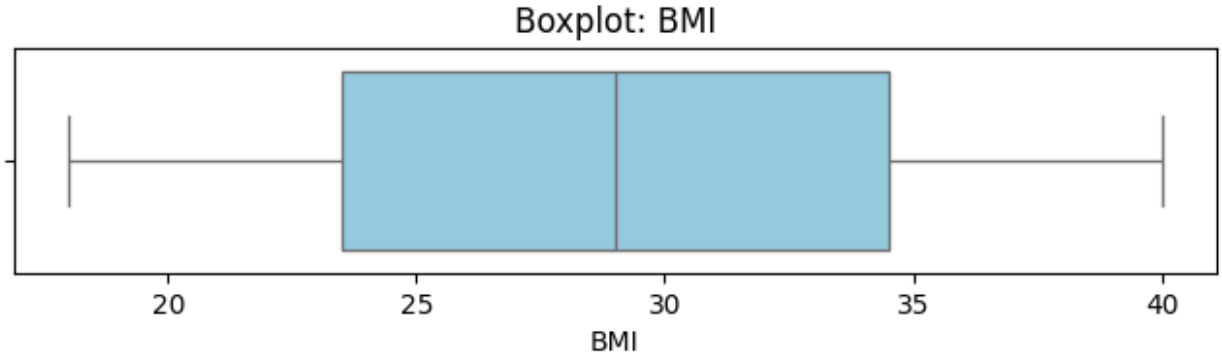

```python
# Create boxplots for each numeric feature to identify outliers
for col in num_cols:
    plt.figure(figsize=(8,1.5))
    sns.boxplot(x=df[col], color='skyblue')
    plt.title(f"Boxplot: {col}")
    plt.show()
```

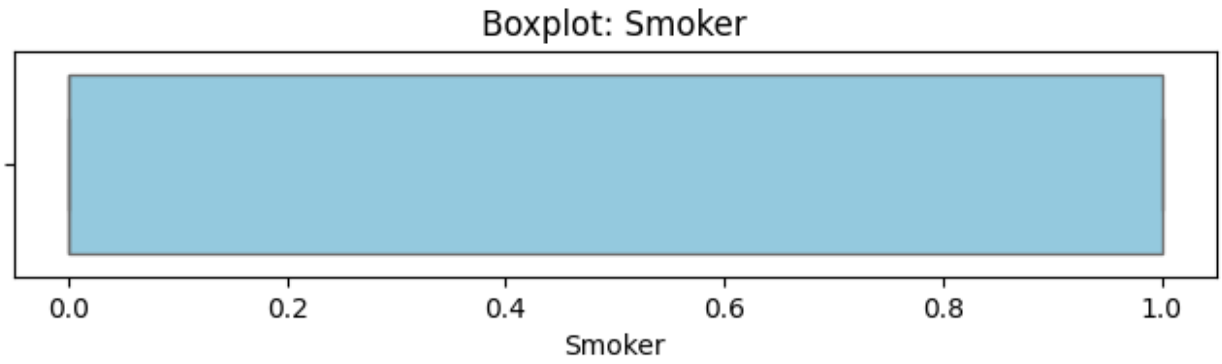


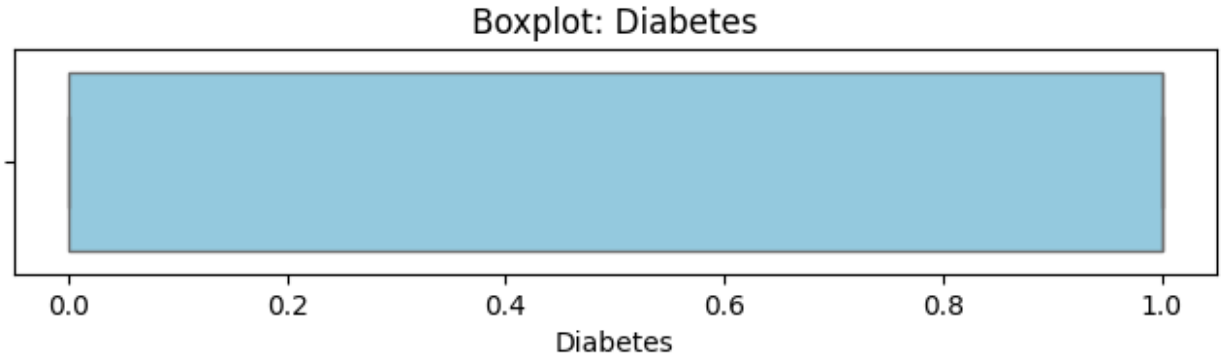

Boxplot: Cholesterol
100
125
150
175
200
225
250
275
300
Cholesterol

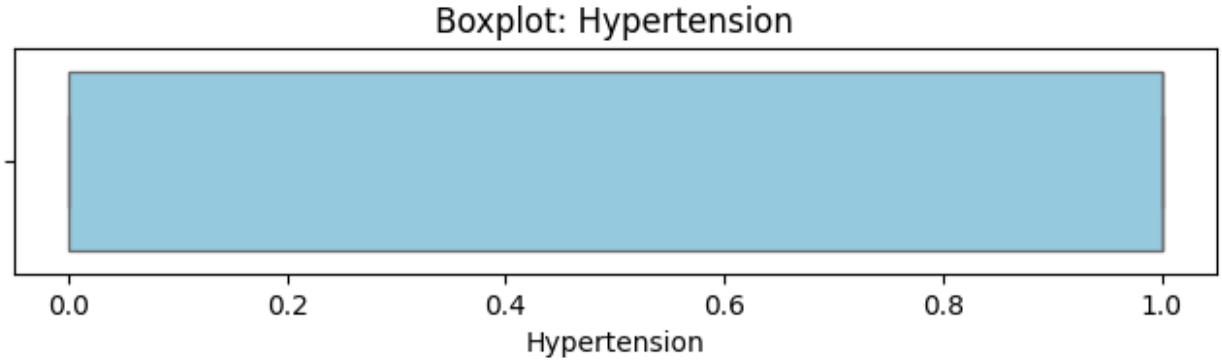

Boxplot: BloodPressure
100
120
140
160
180
BloodPressure

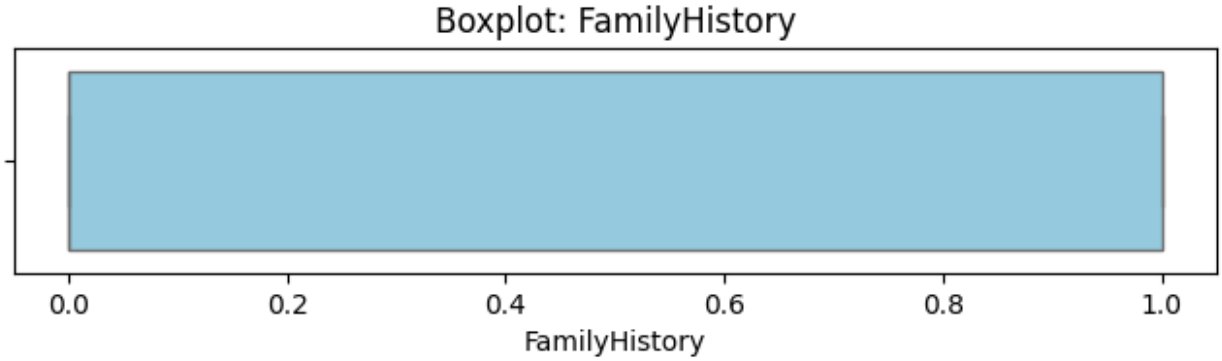

Boxplot: HeartRate
60
70
80
90
100
110
120
HeartRate

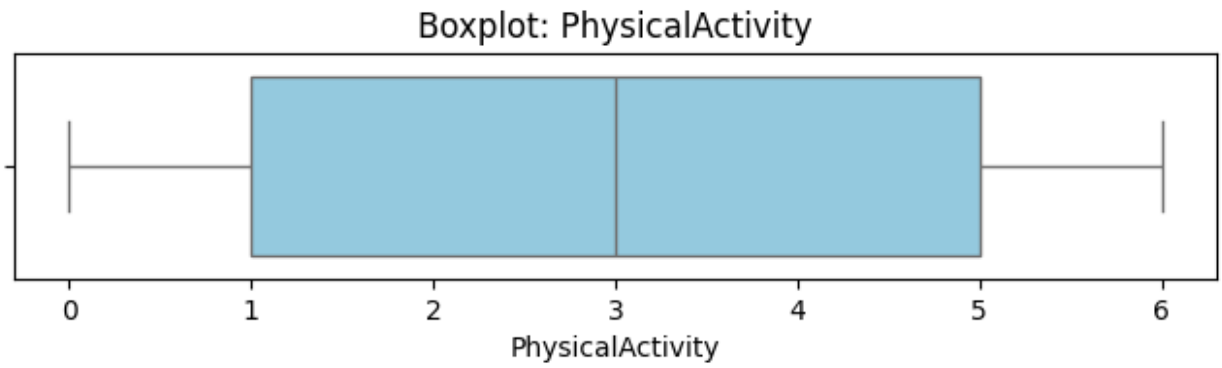

Boxplot: BMI
20
25
30
35
40
BMI


Boxplot: Smoker
0.0
0.2
0.4
0.6
0.8
1.0
Smoker

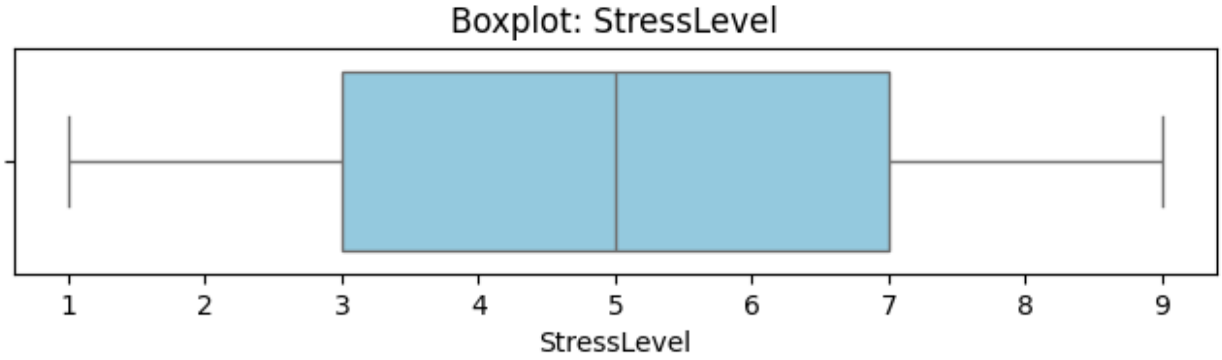

Boxplot: Diabetes
0.0
0.2
0.4
0.6
0.8
1.0
Diabetes

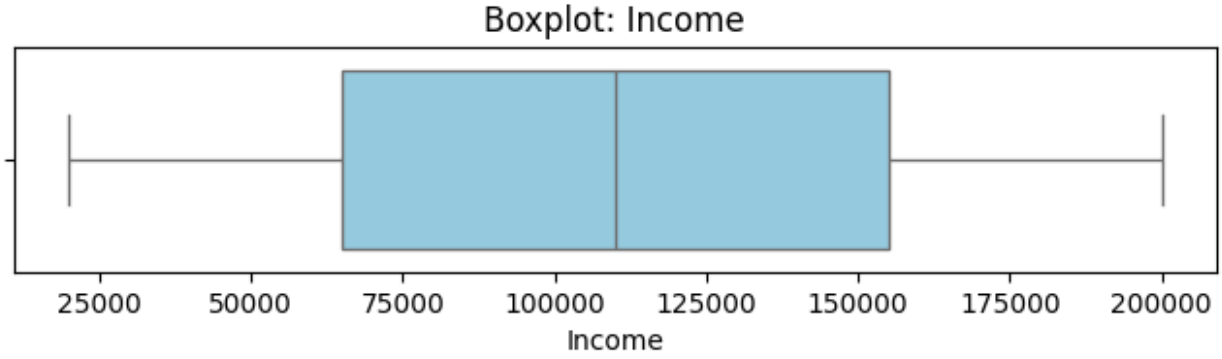

Boxplot: Hypertension
0.0
0.2
0.4
0.6
0.8
1.0
Hypertension

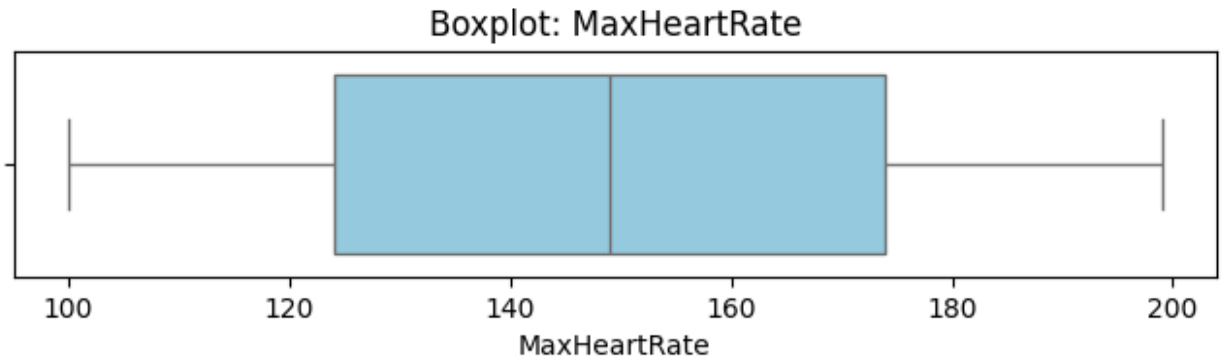

Boxplot: FamilyHistory
0.0
0.2
0.4
0.6
0.8
1.0
FamilyHistory

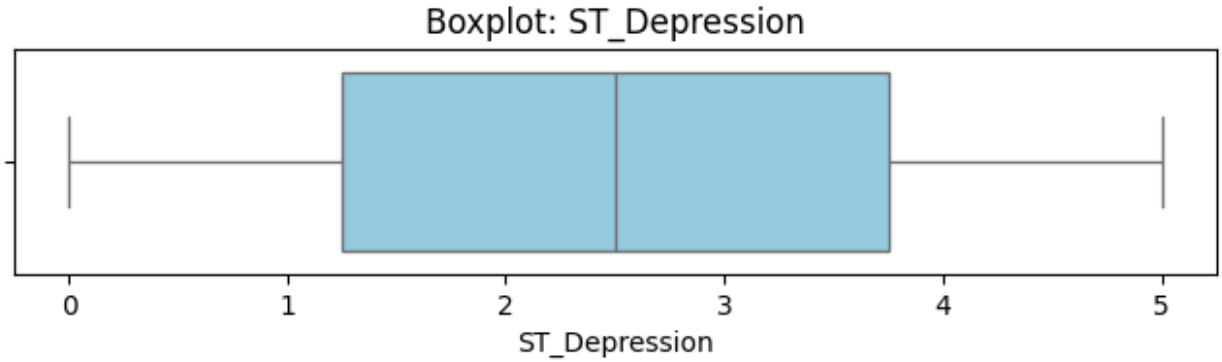

Boxplot: PhysicalActivity
0
1
2
3
4
5
6
PhysicalActivity

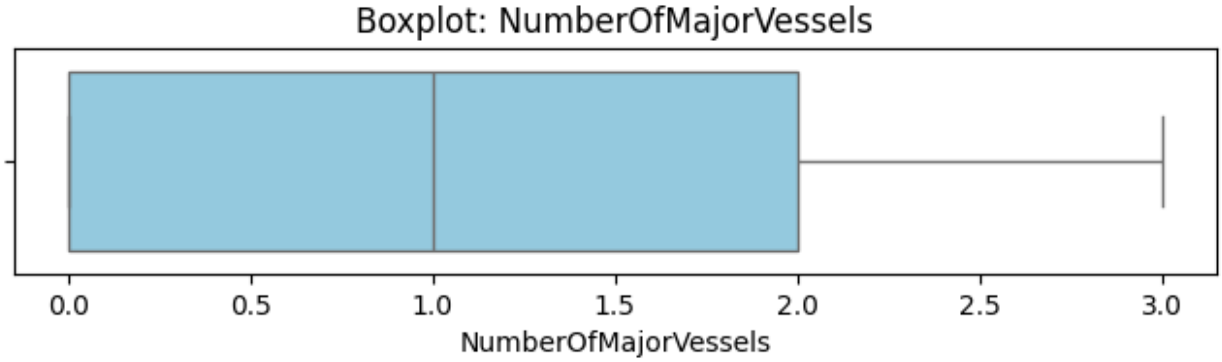

Boxplot: AlcoholConsumption
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
AlcoholConsumption

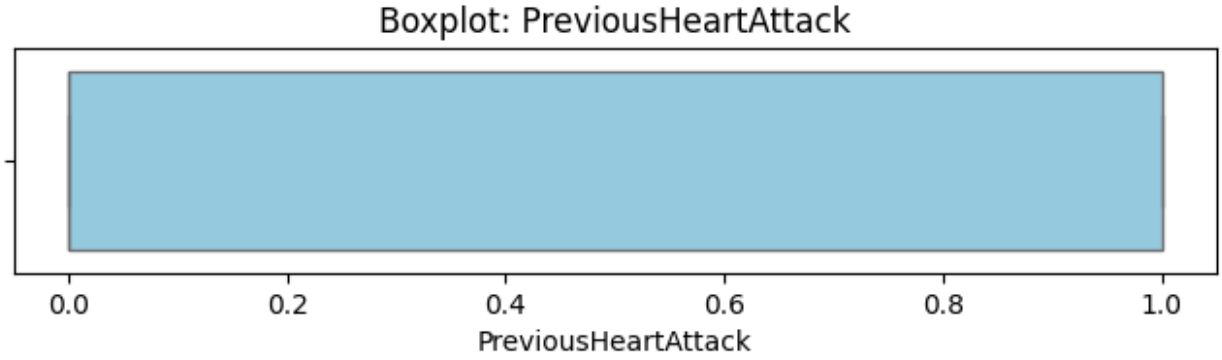

Boxplot: StressLevel
1
2
3
4
5
6
7
8
9
StressLevel

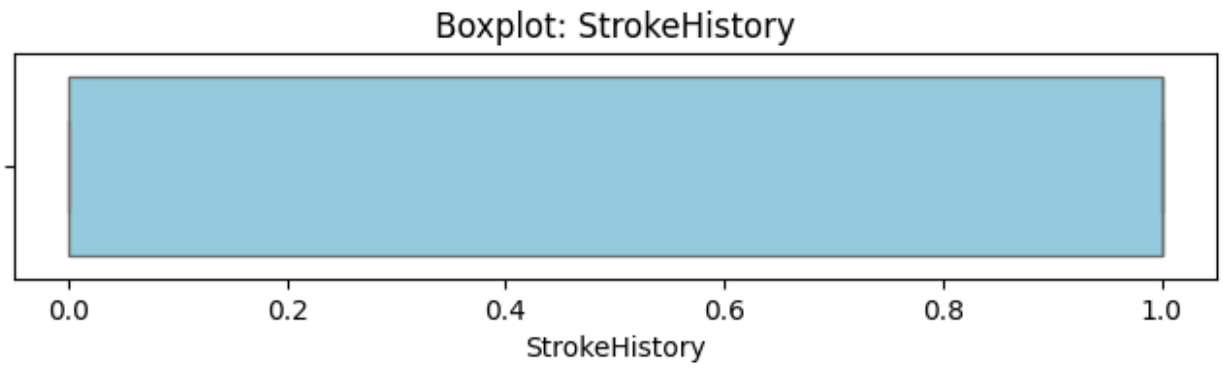

Boxplot: Income
25000
50000
75000
100000
125000
150000
175000
200000
Income


Boxplot: MaxHeartRate
100
120
140
160
180
200
MaxHeartRate

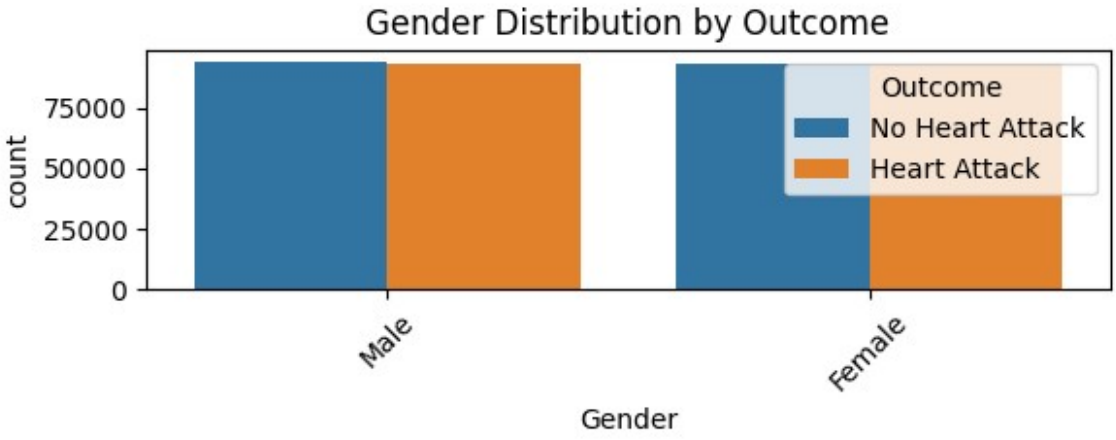

Boxplot: ST_Depression
0
1
2
3
4
5
ST_Depression

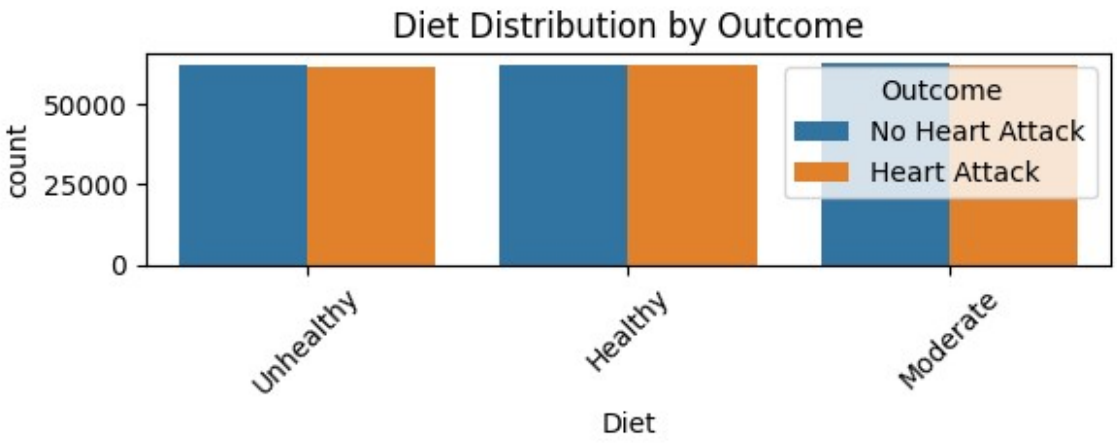

Boxplot: NumberOfMajorVessels
0.0
0.5
1.0
1.5
2.0
2.5
3.0
NumberOfMajorVessels

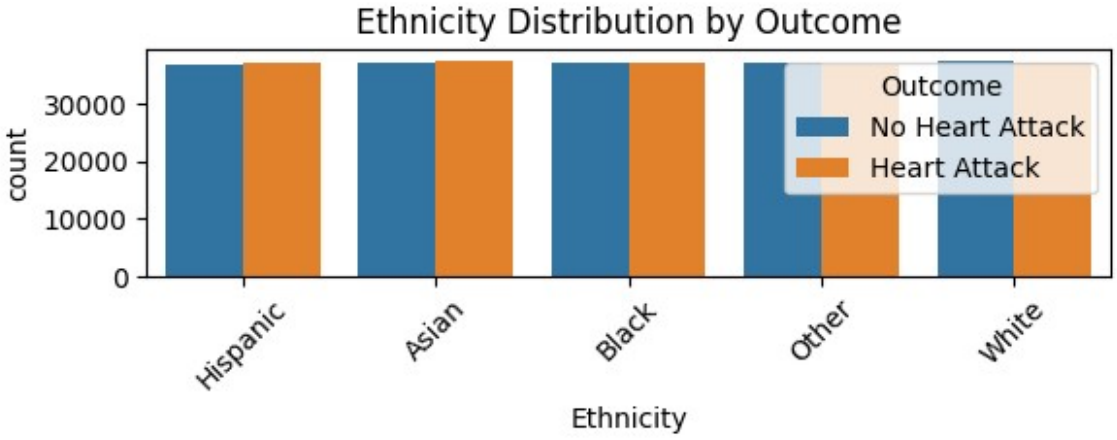

Boxplot: PreviousHeartAttack
0.0
0.2
0.4
0.6
0.8
1.0
PreviousHeartAttack

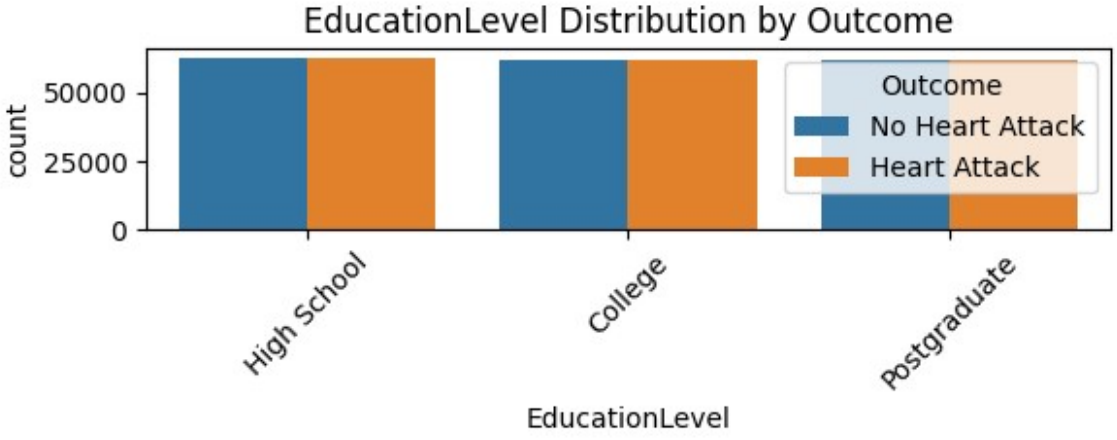


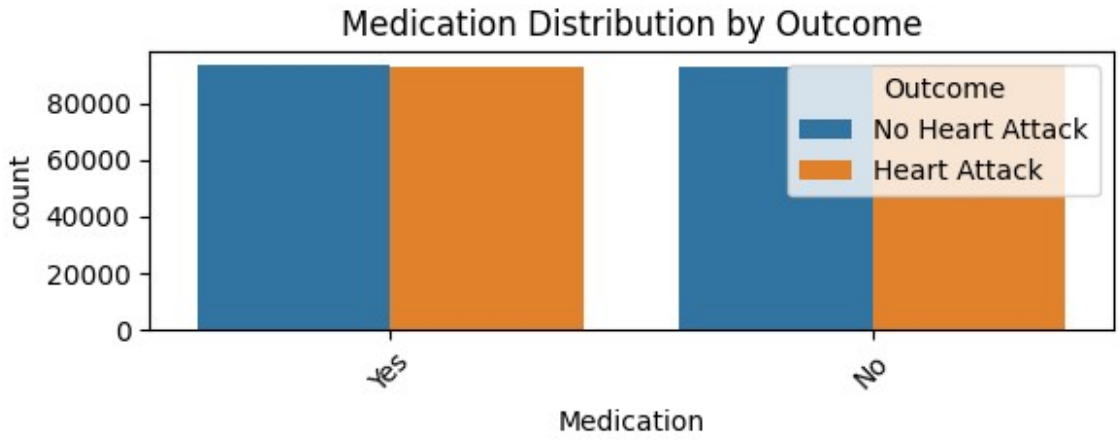

```
# Visualize the distribution of categorical features by outcome
cat_cols = df.select_dtypes(include='object').columns.drop('Outcome')
for col in cat_cols:
    plt.figure(figsize=(6,2.5))
    sns.countplot(data=df, x=col, hue="Outcome")
    plt.title(f"{col} Distribution by Outcome")
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()
```

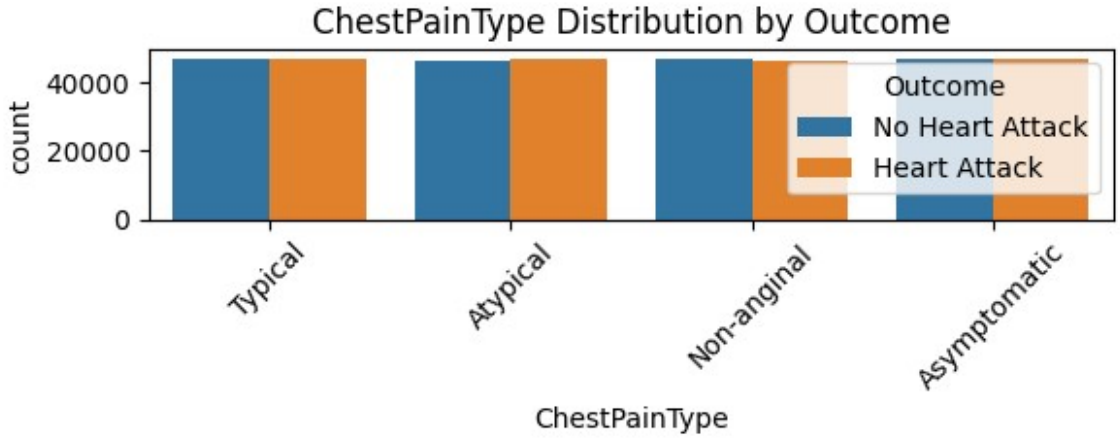


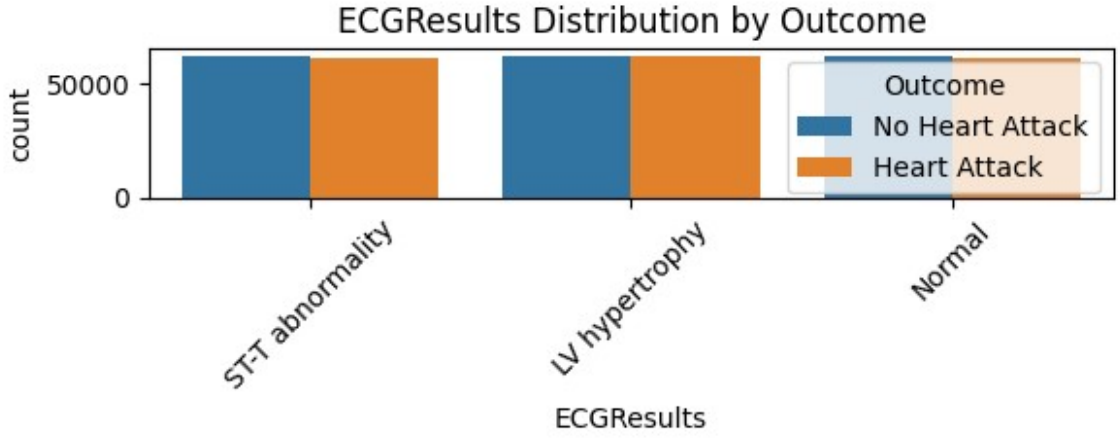

Diet Distribution by Outcome
count
50000
25000
0
Outcome
No Heart Attack
Heart Attack
Unhealthy
Healthy
Moderate
Diet

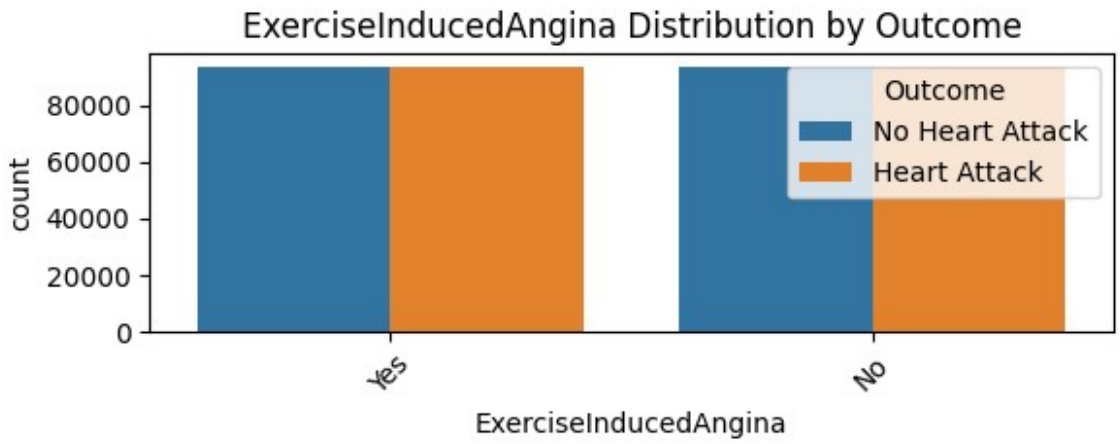

Ethnicity Distribution by Outcome
count
30000
20000
10000
0
Outcome
No Heart Attack
Heart Attack
Hispanic
Asian
Black
Other
White
Ethnicity

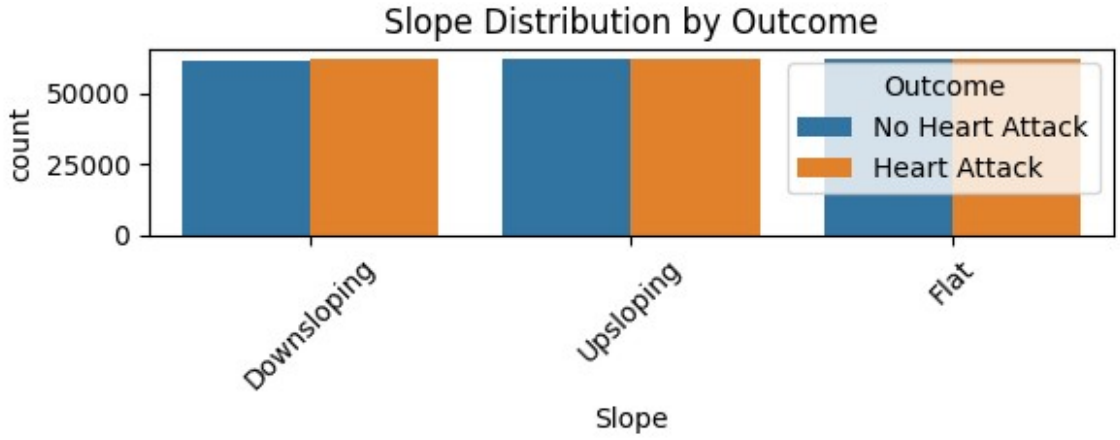

EducationLevel Distribution by Outcome
count
50000
25000
0
Outcome
No Heart Attack
Heart Attack
High School
College
Postgraduate
EducationLevel

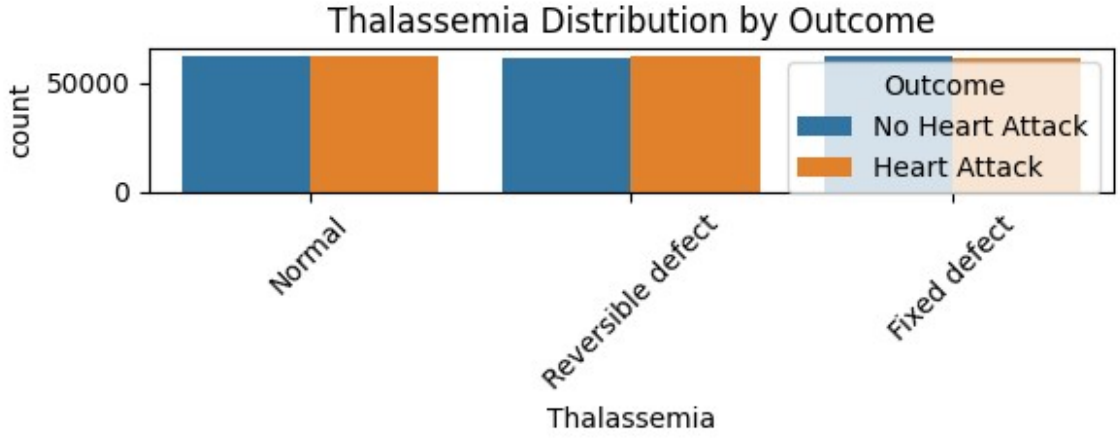

Medication Distribution by Outcome
count
80000
60000
40000
20000
0
Outcome
No Heart Attack
Heart Attack
Yes
No
Medication

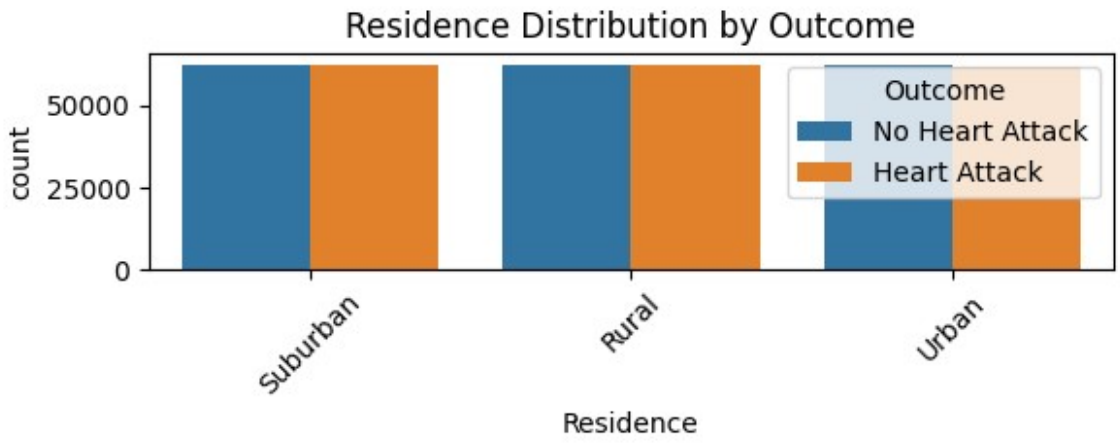

ChestPainType Distribution by Outcome
count
40000
20000
0
Outcome
No Heart Attack
Heart Attack
Typical
Atypical
Non-anginal
Asymptomatic
ChestPainType

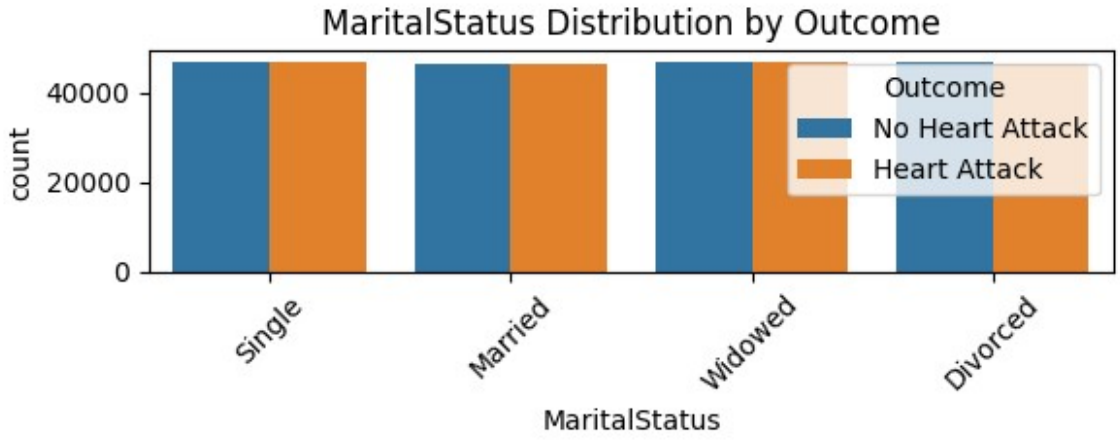

ECGResults Distribution by Outcome
count
50000
0
Outcome
No Heart Attack
Heart Attack
ST-T abnormality
LV hypertrophy
Normal
ECGResults


ExerciseInducedAngina Distribution by Outcome
count
80000
60000
40000
20000
0
Outcome
No Heart Attack
Heart Attack
Yes
No
ExerciseInducedAngina

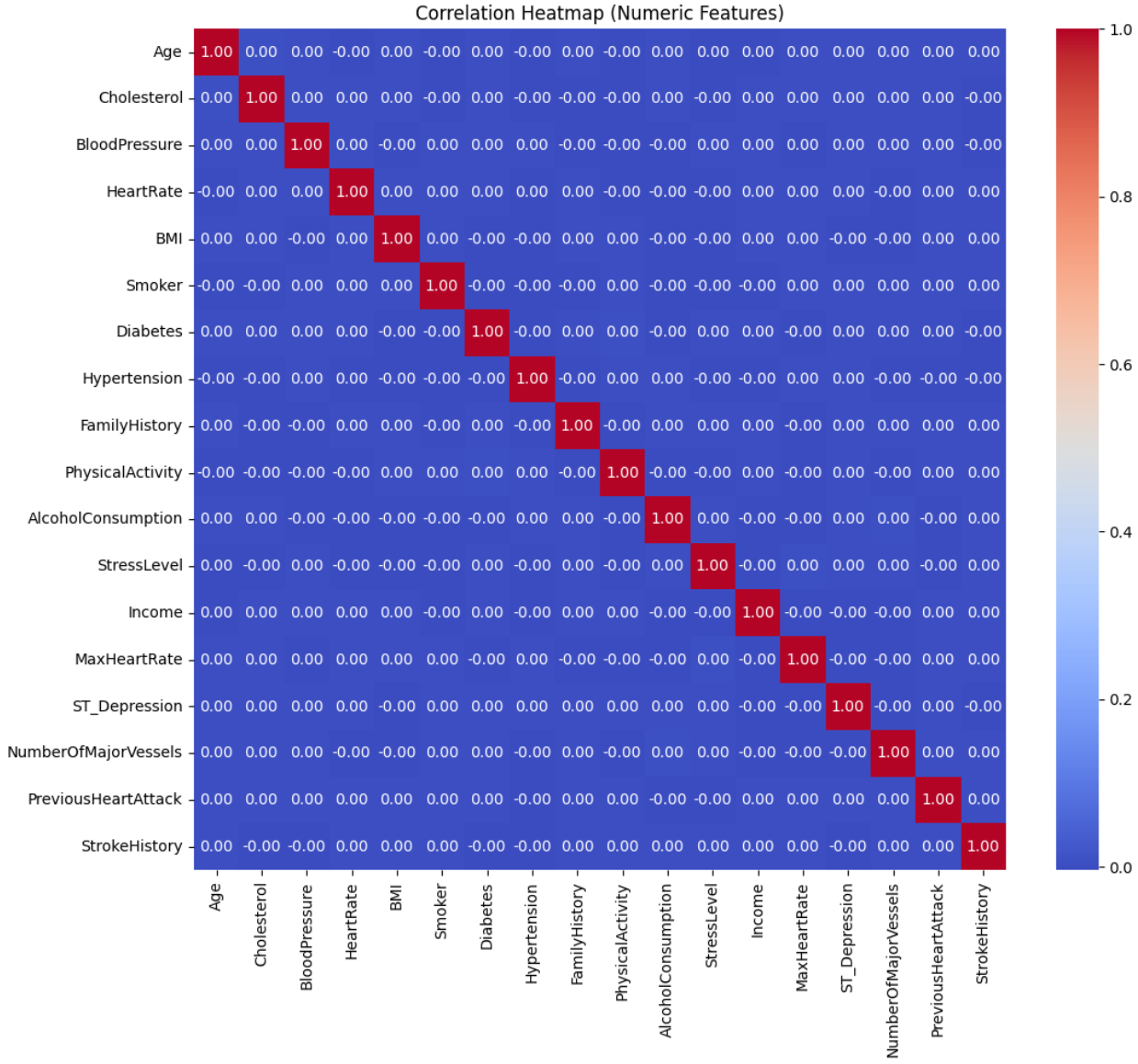

Slope Distribution by Outcome
count
50000
25000
0
Outcome
No Heart Attack
Heart Attack
Downsloping
Upsloping
Flat
Slope


Thalassemia Distribution by Outcome
count
50000
0
Outcome
No Heart Attack
Heart Attack
Normal
Reversible defect
Fixed defect
Thalassemia





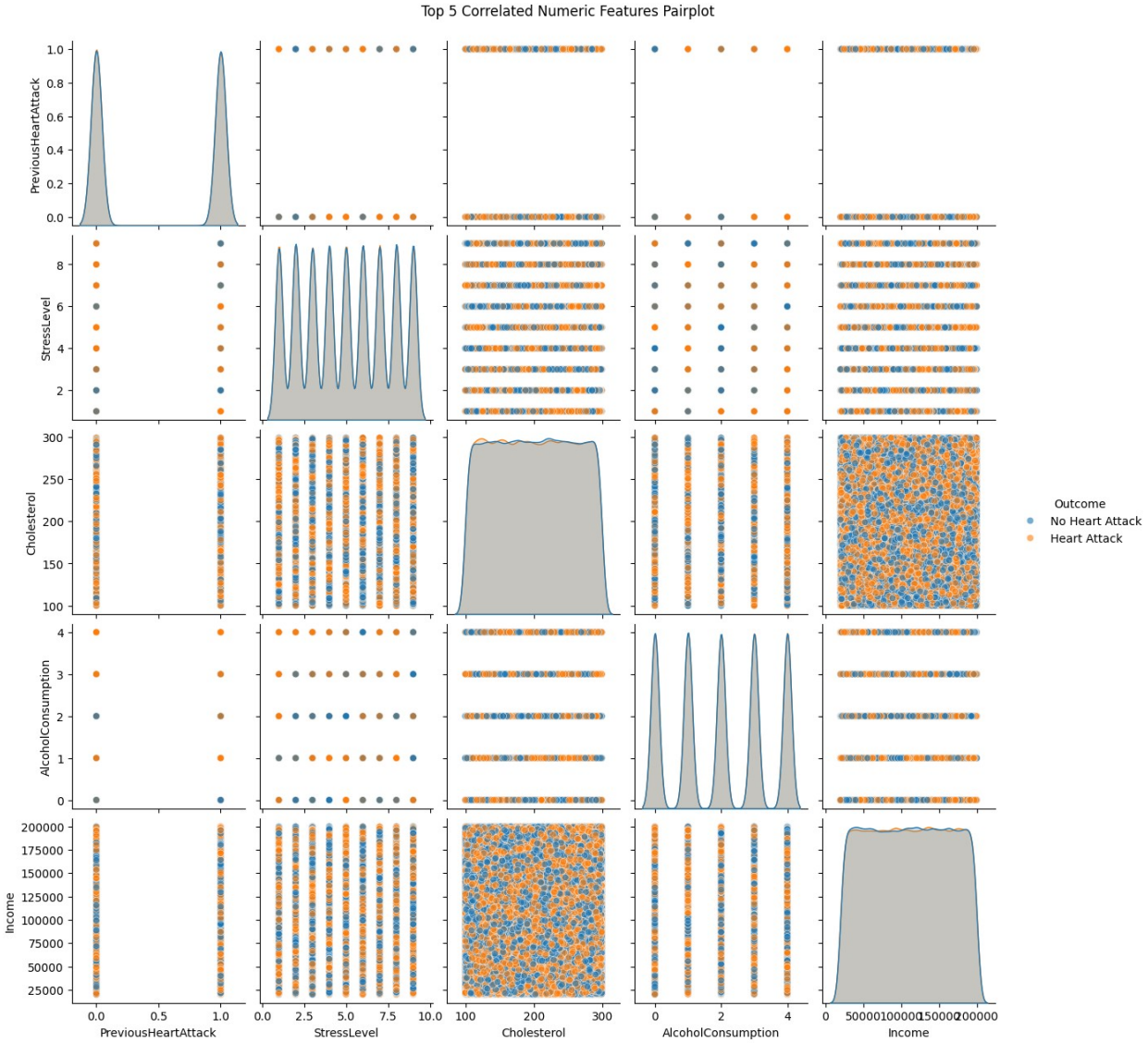

```
# Create a heatmap to visualize correlations between numeric features
plt.figure(figsize=(12,10))
corr = df[num_cols].corr()
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap (Numeric Features)")
plt.show()
```




```
# Calculate and display the top features correlated with the outcome
df_corr = df.copy()
df_corr['OutcomeCode'] =
df_corr['Outcome'].astype('category').cat.codes
corr_with_outcome =
df_corr[num_cols].corrwith(df_corr['OutcomeCode']).abs().sort_values(a
scending=False)
print("\n===== TOP 10 NUMERIC FEATURES CORRELATED WITH OUTCOME =====")
print(corr_with_outcome.head(10))


===== TOP 10 NUMERIC FEATURES CORRELATED WITH OUTCOME =====
PreviousHeartAttack    0.003791
StressLevel            0.002316
Cholesterol            0.001962
```

```
AlcoholConsumption     0.001679
Income                 0.001613
StrokeHistory          0.001546
HeartRate              0.001471
Age                    0.000947
Smoker                 0.000830
Diabetes               0.000710
dtype: float64
```

```
# Create a pairplot for the top 5 correlated numeric features
top5 = corr_with_outcome.head(5).index.tolist()
sns.pairplot(df, vars=top5, hue="Outcome", plot_kws={'alpha':0.6})
plt.suptitle("Top 5 Correlated Numeric Features Pairplot", y=1.02)
plt.show()
```



```python
# Display the distribution of categorical features by outcome
print("\n===== CATEGORICAL FEATURE DISTRIBUTION BY OUTCOME =====")
for col in cat_cols:
    cross = pd.crosstab(df[col], df['Outcome'], normalize='index')
    print(f"\n{col} (proportion of Outcome):")
    print(cross)
```

```
===== CATEGORICAL FEATURE DISTRIBUTION BY OUTCOME =====

Gender (proportion of Outcome):
Outcome  Heart Attack  No Heart Attack
Gender
Female       0.500489         0.499511
Male         0.498597         0.501403

Diet (proportion of Outcome):
Outcome    Heart Attack  No Heart Attack
Diet
Healthy        0.500004         0.499996
Moderate       0.499700         0.500300
Unhealthy      0.498918         0.501082

Ethnicity (proportion of Outcome):
Outcome    Heart Attack  No Heart Attack
Ethnicity
Asian          0.500534         0.499466
Black          0.498309         0.501691
Hispanic       0.502004         0.497996
Other          0.498982         0.501018
White          0.497886         0.502114

EducationLevel (proportion of Outcome):
Outcome         Heart Attack  No Heart Attack
EducationLevel
College             0.500750         0.499250
High School         0.498527         0.501473
Postgraduate        0.499355         0.500645

Medication (proportion of Outcome):
Outcome     Heart Attack  No Heart Attack
Medication
No              0.500568         0.499432
Yes             0.498514         0.501486

ChestPainType (proportion of Outcome):
Outcome        Heart Attack  No Heart Attack
ChestPainType
Asymptomatic       0.497164         0.502836
Atypical           0.502144         0.497856
```

```
Non-anginal        0.498639         0.501361
Typical            0.500226         0.499774

ECGResults (proportion of Outcome):
Outcome           Heart Attack  No Heart Attack
ECGResults
LV hypertrophy        0.501034         0.498966
Normal                0.498520         0.501480
ST-T abnormality      0.499062         0.500938

ExerciseInducedAngina (proportion of Outcome):
Outcome                 Heart Attack  No Heart Attack
ExerciseInducedAngina
No                          0.499349         0.500651
Yes                         0.499735         0.500265

Slope (proportion of Outcome):
Outcome      Heart Attack  No Heart Attack
Slope
Downsloping      0.500568         0.499432
Flat             0.498731         0.501269
Upsloping        0.499329         0.500671

Thalassemia (proportion of Outcome):
Outcome            Heart Attack  No Heart Attack
Thalassemia
Fixed defect           0.497120         0.502880
Normal                 0.498489         0.501511
Reversible defect      0.503018         0.496982

Residence (proportion of Outcome):
Outcome    Heart Attack  No Heart Attack
Residence
Rural          0.498456         0.501544
Suburban       0.500702         0.499298
Urban          0.499463         0.500537

EmploymentStatus (proportion of Outcome):
Outcome           Heart Attack  No Heart Attack
EmploymentStatus
Employed              0.500222         0.499778
Retired               0.499843         0.500157
Unemployed            0.498565         0.501435

MaritalStatus (proportion of Outcome):
Outcome        Heart Attack  No Heart Attack
MaritalStatus
Divorced           0.498700         0.501300
Married            0.500894         0.499106
```

```
Single              0.499653          0.500347
Widowed             0.498925          0.501075
```

```
print("\n===== EDA COMPLETE =====")
```

```
===== EDA COMPLETE =====
```