A2_part2

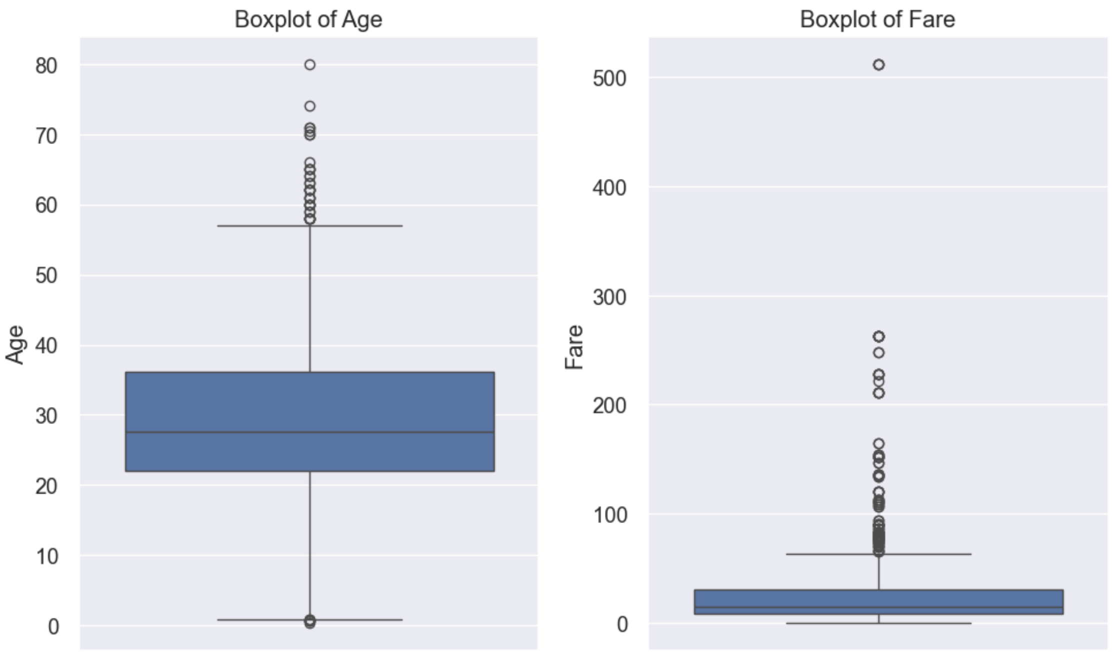
June 10, 2025

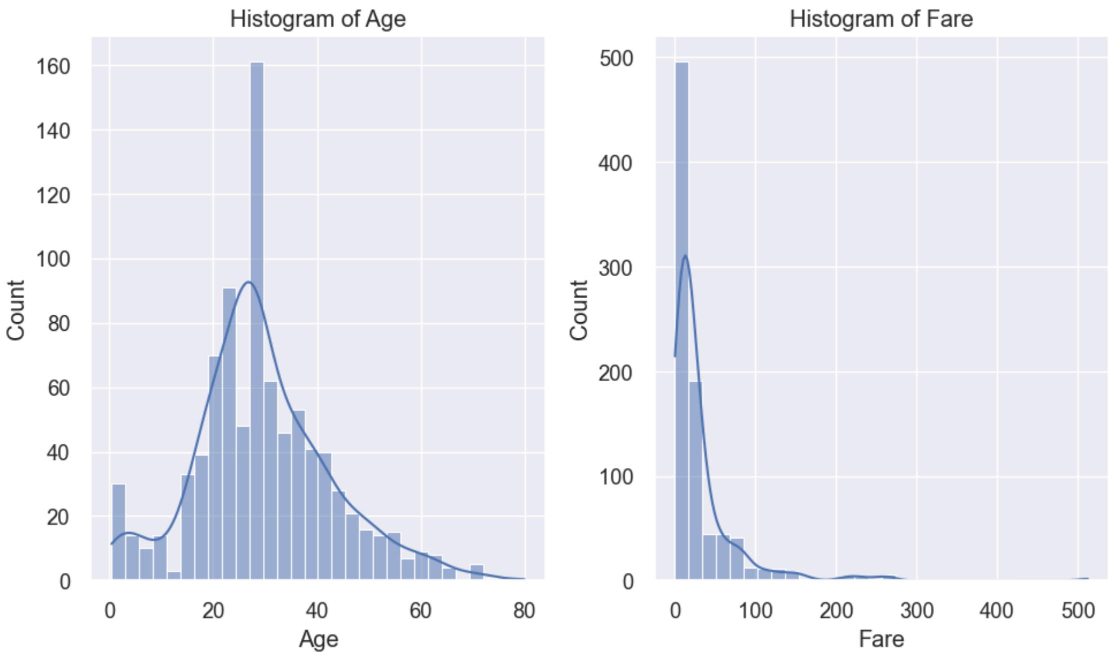
[1]:

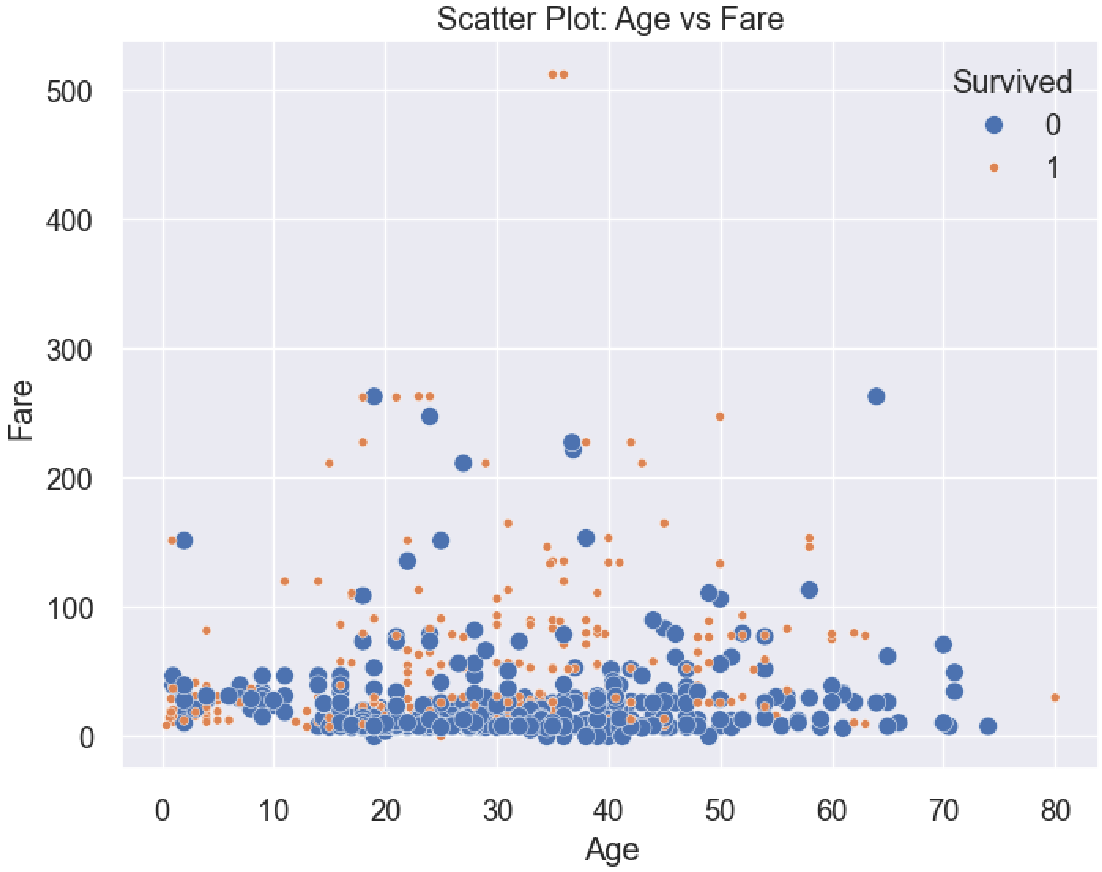
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import chi2_contingency
from sklearn.linear_model import LinearRegression
import os

# Load the dataset
df = pd.read_csv('titanic.csv')

# 1. Identify attributes, their types, distinct values, mean, median, std, and range
def analyze_attributes(df):
    print("Attribute Analysis:")
    print("=" * 50)
    for column in df.columns:
        print(f"\nColumn: {column}")
        print(f"Type: {df[column].dtype}")
        print(f"Distinct Values: {df[column].nunique()}")
        if df[column].dtype in ['int64', 'float64']:
            print(f"Mean: {df[column].mean():.2f}")
            print(f"Median: {df[column].median():.2f}")
            print(f"Standard Deviation: {df[column].std():.2f}")
            print(f"Range: [{df[column].min()}, {df[column].max()}]")
        else:
            print("Most frequent values:")
            print(df[column].value_counts().head())

# 2. Data Preprocessing
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])  # Fill missing Embarked
df = df.drop('Cabin', axis=1)  # Drop Cabin due to many missing values

# Predict missing Age values using Linear Regression
def predict_missing_age(df):
    features = ['Pclass', 'SibSp', 'Parch', 'Fare']
    df_with_age = df[df['Age'].notna()].dropna(subset=features)
    df_no_age = df[df['Age'].isna()]

    X_train = df_with_age[features]
    y_train = df_with_age['Age']
    X_test = df_no_age[features]

    model = LinearRegression()
    model.fit(X_train, y_train)

    df.loc[df['Age'].isna(), 'Age'] = model.predict(X_test)
    return df

df = predict_missing_age(df)
df = df[df['Age'] >= 0]  # Remove invalid age entries

# 3. Visualizations
try:
    plt.style.use('seaborn-v0_8')
except:
    plt.style.use('ggplot')
sns.set(font_scale=1.2)

# Boxplots for Age and Fare
plt.figure(figsize=(10, 6))
plt.subplot(1, 2, 1)
sns.boxplot(y=df['Age'])
plt.title('Boxplot of Age')
plt.subplot(1, 2, 2)
sns.boxplot(y=df['Fare'])
plt.title('Boxplot of Fare')
plt.tight_layout()
plt.show()

# Histograms for Age and Fare
plt.figure(figsize=(10, 6))
plt.subplot(1, 2, 1)
sns.histplot(df['Age'], bins=30, kde=True)
plt.title('Histogram of Age')
plt.subplot(1, 2, 2)
sns.histplot(df['Fare'], bins=30, kde=True)
plt.title('Histogram of Fare')
plt.tight_layout()
plt.show()

# Scatter plot: Age vs Fare
```

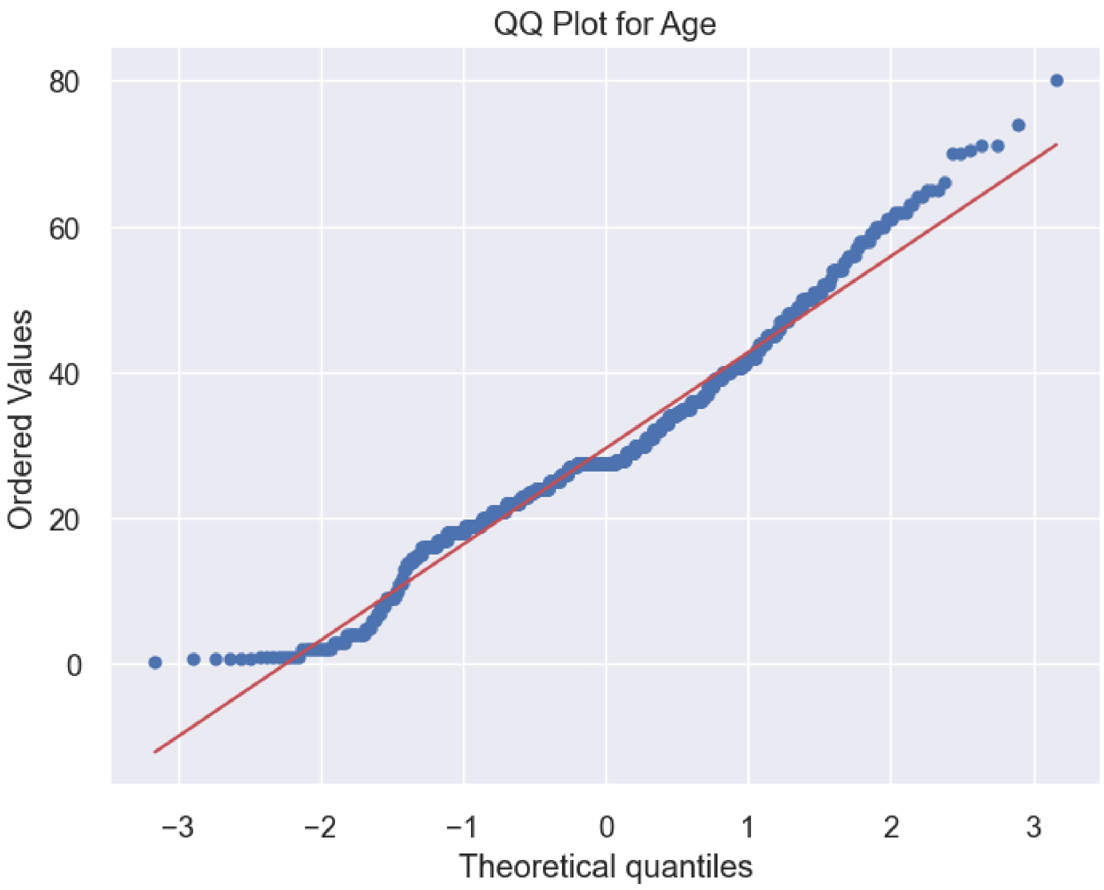
```python
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Age', y='Fare', hue='Survived', size='Survived', data=df)
plt.title('Scatter Plot: Age vs Fare')
plt.show()

# QQ plot for Age
plt.figure(figsize=(8, 6))
stats.probplot(df['Age'], dist="norm", plot=plt)
plt.title('QQ Plot for Age')
plt.show()

# 4. Chi-square tests
def chi_square_test(df, col1, col2):
    contingency_table = pd.crosstab(df[col1], df[col2])
    chi2, p, dof, expected = chi2_contingency(contingency_table)
    print(f"\nChi-square Test between {col1} and {col2}:")
    print(f"Chi2 Statistic: {chi2:.2f}")
    print(f"P-value: {p:.4f}")
    print(f"Degrees of Freedom: {dof}")

# Perform chi-square tests
chi_square_test(df, 'Sex', 'Survived')
chi_square_test(df, 'Pclass', 'Survived')
chi_square_test(df, 'Embarked', 'Survived')

# 5. EDA: Who is more likely to survive?
df['Sex'] = df['Sex'].map({'male': 1, 'female': 0})  # Binary encoding

# Countplot: Survival by Sex
plt.figure(figsize=(6, 4))
sns.countplot(x='Sex', hue='Survived', data=df)
plt.title('Survival by Sex (0=Female, 1=Male)')
plt.show()

# Countplot: Survival by Pclass
plt.figure(figsize=(6, 4))
sns.countplot(x='Pclass', hue='Survived', data=df)
plt.title('Survival by Pclass')
plt.show()

# Countplot: Survival by Embarked
plt.figure(figsize=(6, 4))
sns.countplot(x='Embarked', hue='Survived', data=df)
plt.title('Survival by Embarked (C, Q, S)')
plt.show()

# Survival rate by Sex and Pclass
```

```python
survival_rate = df.groupby(['Sex', 'Pclass'])['Survived'].mean().reset_index()
print("\nSurvival Rate by Sex and Pclass:")
print(survival_rate)

# Survival rate by Age Group
df['AgeGroup'] = pd.cut(df['Age'], bins=[0, 18, 30, 50, 100], labels=['Child', 'YoungAdult', 'Adult', 'Senior'])
# Survival rate by Age Group (fixed for FutureWarning)
survival_rate_age = df.groupby('AgeGroup', observed=False)['Survived'].mean().reset_index()
print("\nSurvival Rate by Age Group:")
print(survival_rate_age)

# Barplot: Survival Rate by Age Group
plt.figure(figsize=(6, 4))
sns.barplot(x='AgeGroup', y='Survived', data=survival_rate_age)
plt.title('Survival Rate by Age Group')
plt.show()

# Final attribute analysis
analyze_attributes(df)
```

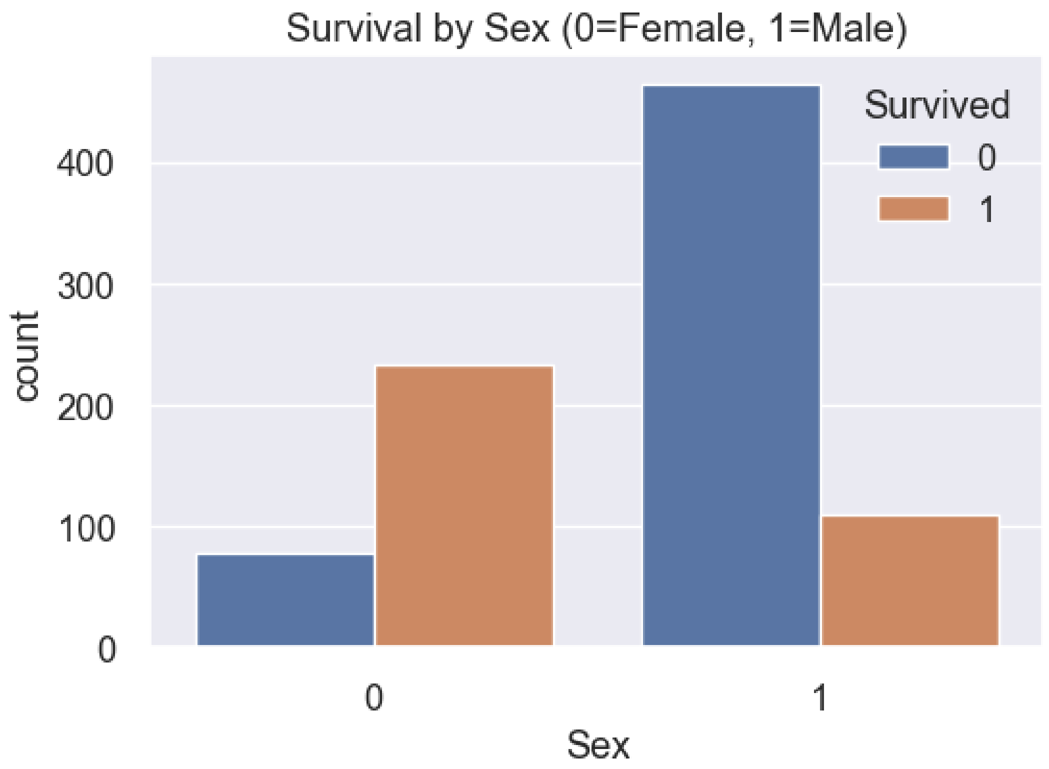


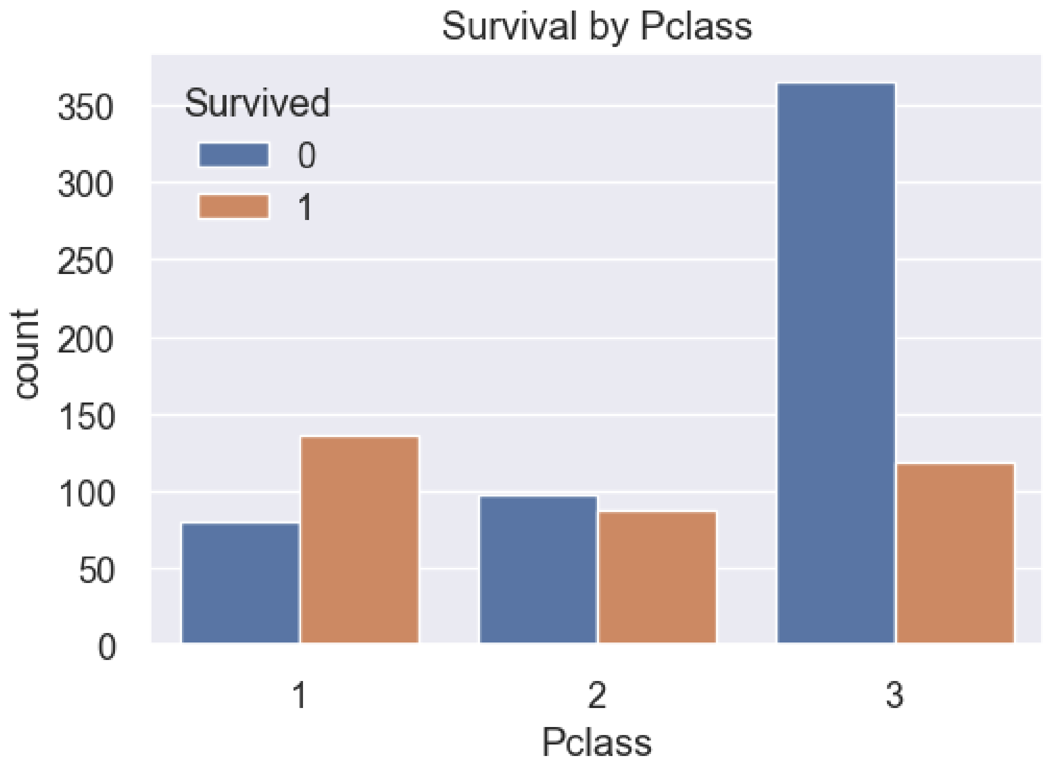

Histogram of Age
Count
160
140
120
100
80
60
40
20
0
0
20
40
60
80
Age
Histogram of Fare
Count
500
400
300
200
100
0
0
100
200
300
400
500
Fare

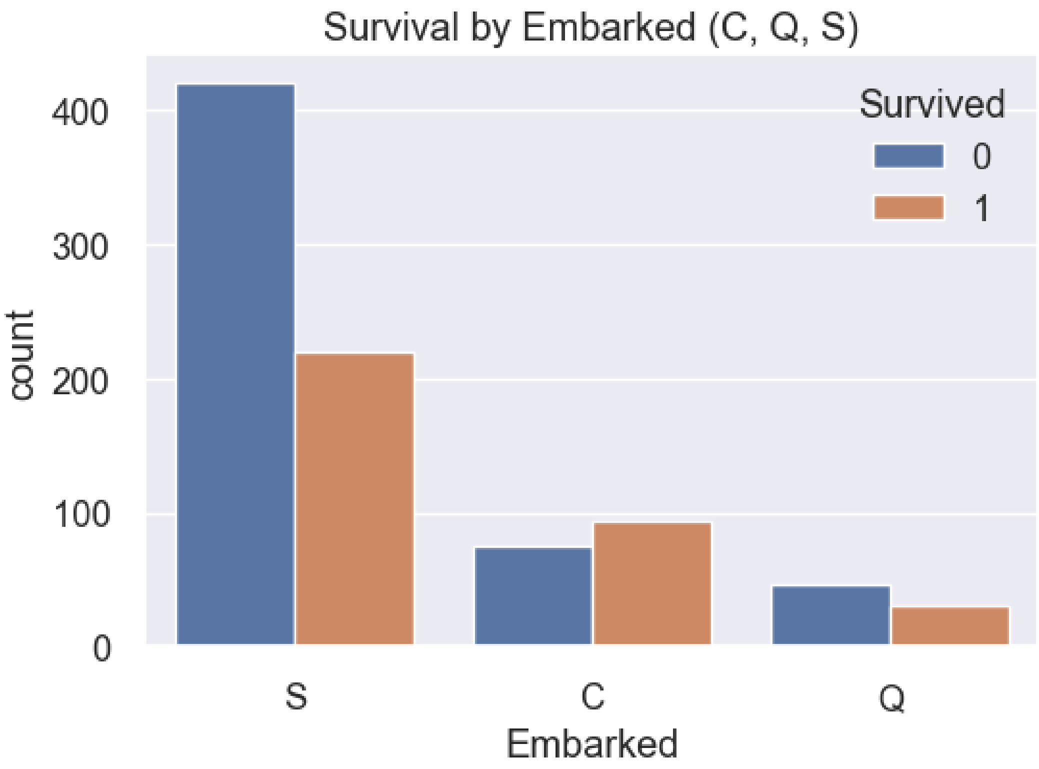

Scatter Plot: Age vs Fare
Fare
500
400
300
200
100
0
0
10
20
30
40
50
60
70
80
Age
Survived
● 0
● 1



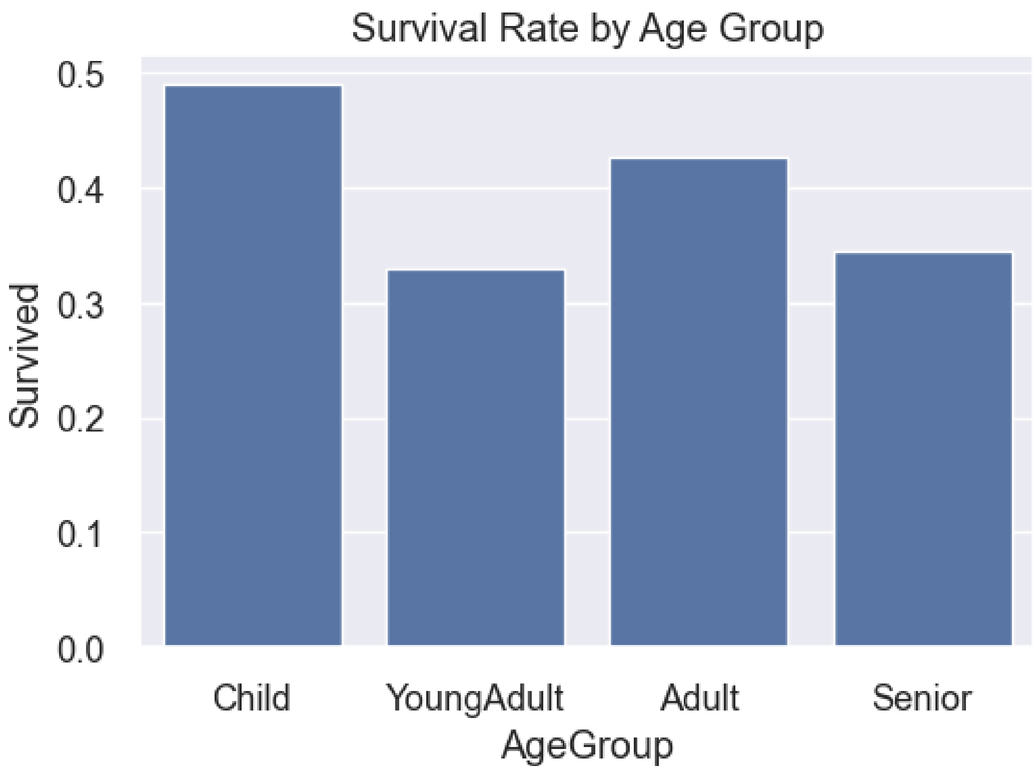
```
Chi-square Test between Sex and Survived:
Chi2 Statistic: 263.18
P-value: 0.0000
Degrees of Freedom: 1

Chi-square Test between Pclass and Survived:
Chi2 Statistic: 99.96
P-value: 0.0000
Degrees of Freedom: 2

Chi-square Test between Embarked and Survived:
Chi2 Statistic: 24.93
P-value: 0.0000
Degrees of Freedom: 2
```


Survival by Sex (0=Female, 1=Male)
count
0
100
200
300
400
0
1
Sex
Survived
0
1


Survival by Pclass
count
0
50
100
150
200
250
300
350
1
2
3
Pclass
Survived
0
1



```
Survival Rate by Sex and Pclass:
   Sex  Pclass  Survived
0    0       1  0.968085
1    0       2  0.921053
2    0       3  0.510638
3    1       1  0.368852
4    1       2  0.157407
5    1       3  0.137026

Survival Rate by Age Group:
     AgeGroup  Survived
0       Child  0.489510
1  YoungAdult  0.329114
2       Adult  0.425532
3      Senior  0.343750
```



```
Attribute Analysis:
==================================================

Column: PassengerId
Type: int64
Distinct Values: 884
Mean: 445.72
Median: 446.50
Standard Deviation: 256.87
Range: [1, 891]

Column: Survived
Type: int64
Distinct Values: 2
Mean: 0.39
Median: 0.00
Standard Deviation: 0.49
Range: [0, 1]

Column: Pclass
Type: int64
Distinct Values: 3
```

```
Mean: 2.30
Median: 3.00
Standard Deviation: 0.84
Range: [1, 3]

Column: Name
Type: object
Distinct Values: 884
Most frequent values:
Name
Braund, Mr. Owen Harris             1
Boulos, Mr. Hanna                   1
Frolicher-Stehli, Mr. Maxmillian    1
Gilinski, Mr. Eliezer               1
Murdlin, Mr. Joseph                 1
Name: count, dtype: int64

Column: Sex
Type: int64
Distinct Values: 2
Mean: 0.65
Median: 1.00
Standard Deviation: 0.48
Range: [0, 1]

Column: Age
Type: float64
Distinct Values: 169
Mean: 29.57
Median: 27.54
Standard Deviation: 13.30
Range: [0.42, 80.0]

Column: SibSp
Type: int64
Distinct Values: 6
Mean: 0.46
Median: 0.00
Standard Deviation: 0.88
Range: [0, 5]

Column: Parch
Type: int64
Distinct Values: 7
Mean: 0.37
Median: 0.00
Standard Deviation: 0.80
Range: [0, 6]
```

```
Column: Ticket
Type: object
Distinct Values: 680
Most frequent values:
Ticket
347082     7
1601       7
3101295    6
CA 2144    6
347088     6
Name: count, dtype: int64

Column: Fare
Type: float64
Distinct Values: 247
Mean: 31.91
Median: 14.45
Standard Deviation: 49.78
Range: [0.0, 512.3292]

Column: Embarked
Type: object
Distinct Values: 3
Most frequent values:
Embarked
S    639
C    168
Q     77
Name: count, dtype: int64

Column: AgeGroup
Type: category
Distinct Values: 4
Most frequent values:
AgeGroup
YoungAdult    395
Adult         282
Child         143
Senior         64
Name: count, dtype: int64
```