```
# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.impute import SimpleImputer
from sklearn.ensemble import IsolationForest
```

```
# importing dataset
df = pd.read_csv('/content/Titanic-Dataset.csv')
```

```
df.head(5)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fa
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.25
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.28
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.92
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.10
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.05

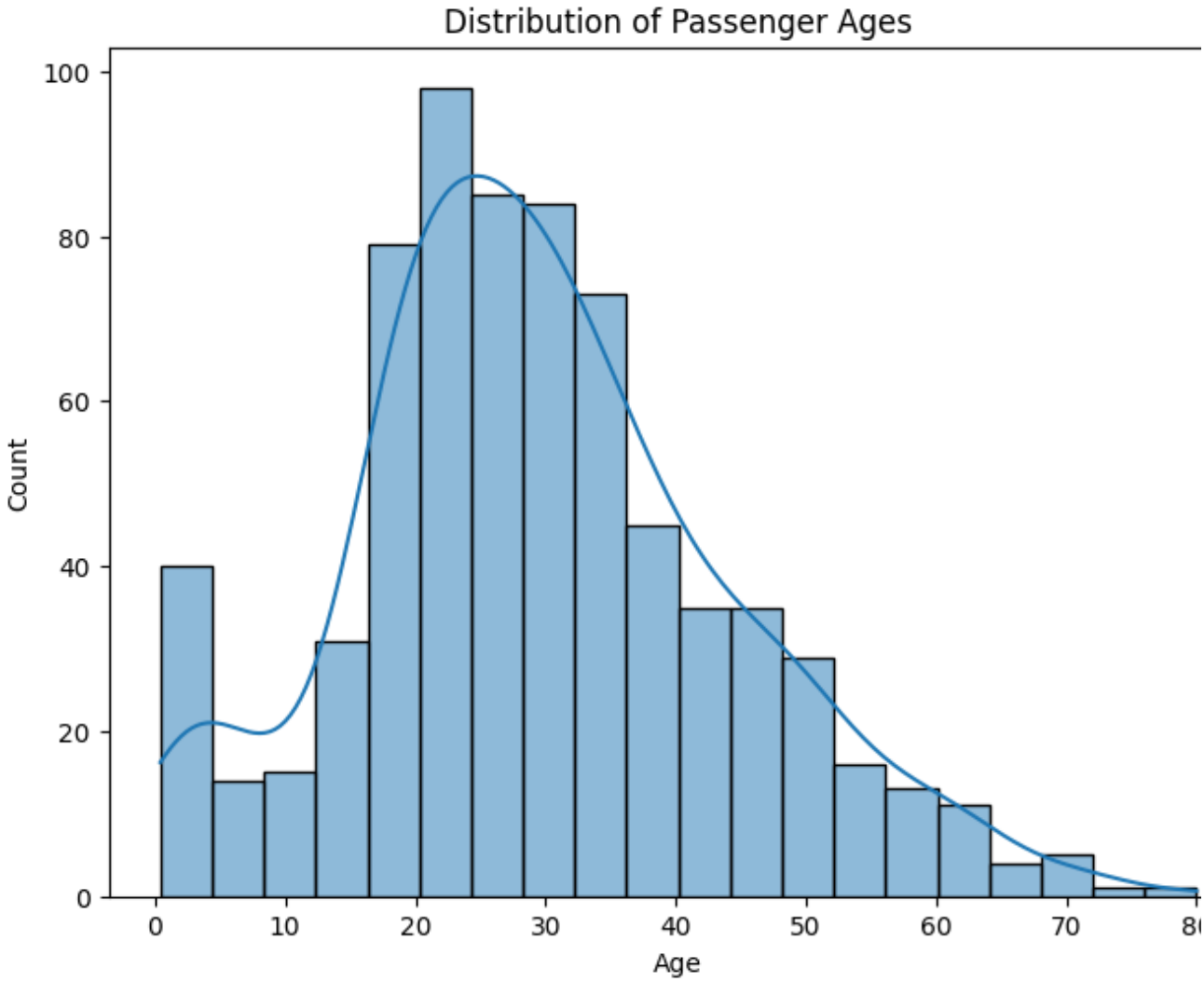
```
df.drop('Cabin', axis=1, inplace=True)
```

```
# Check for null values
null_values = df.isnull().sum()
print("Null Values:\n", null_values)
```

```
Null Values:
 PassengerId      0
Survived         0
Pclass           0
```

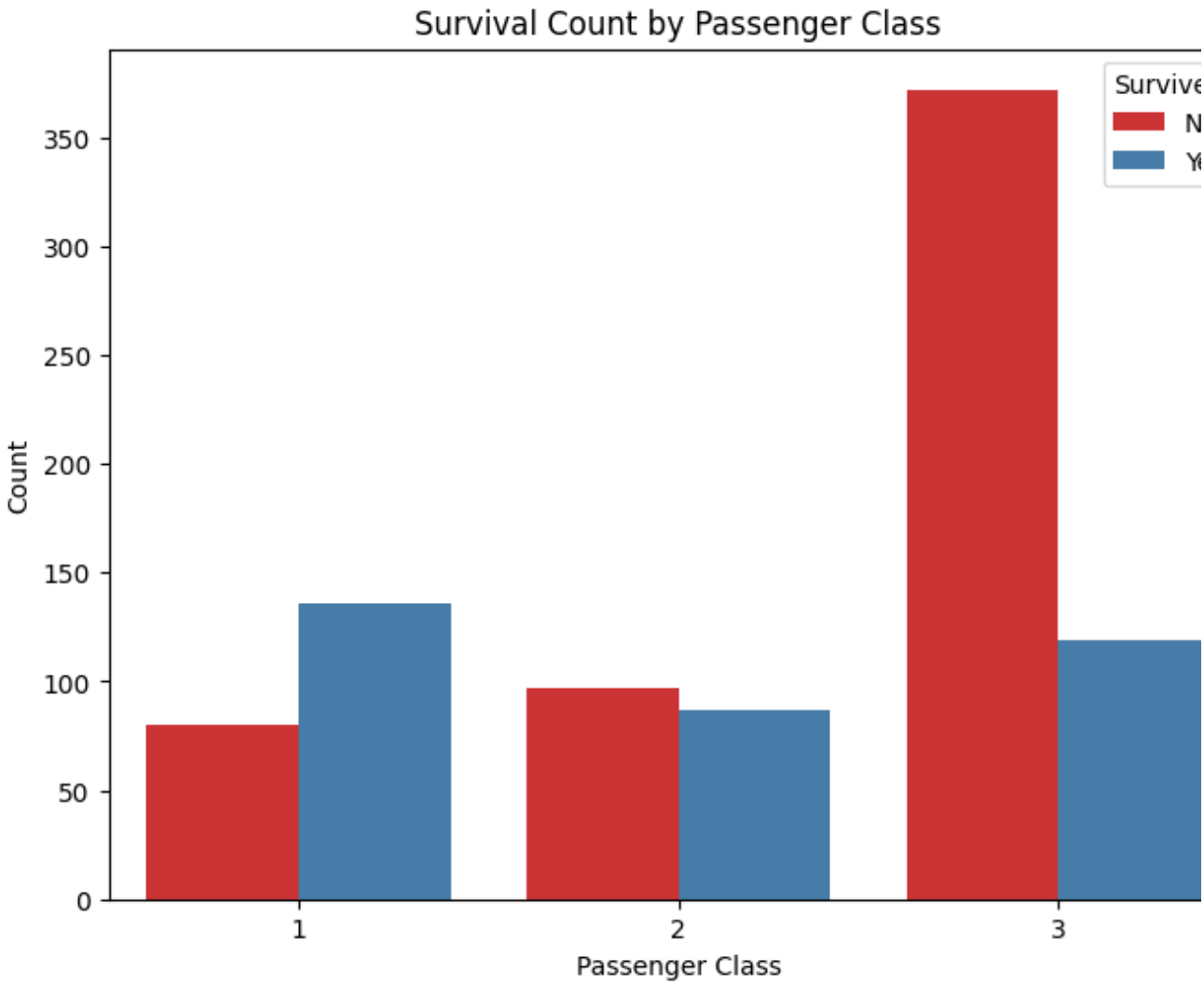
```
Name            0
Sex             0
Age           177
SibSp           0
Parch           0
Ticket          0
Fare            0
Embarked        2
dtype: int64
```

```
# Data Visualization
#1
plt.figure(figsize=(8, 6))
sns.histplot(data=df, x='Age', bins=20, kde=True)
plt.title('Distribution of Passenger Ages')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



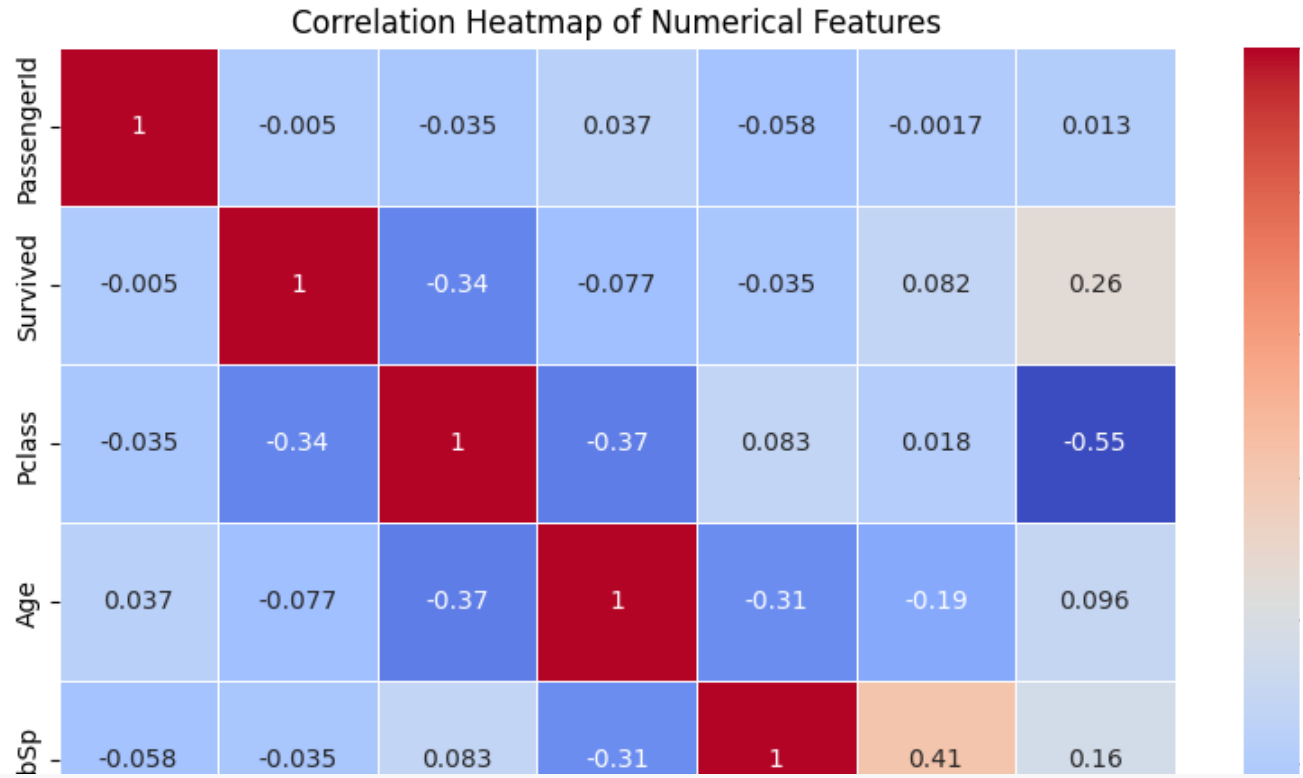

```
#2
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='Pclass', hue='Survived', palette='Set1')
plt.title('Survival Count by Passenger Class')
plt.xlabel('Passenger Class')
plt.ylabel('Count')
```

```
plt.legend(title='Survived', labels=['No', 'Yes'])
plt.show()
```




```
#3
correlation_matrix = df.corr()
# Create a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap of Numerical Features')
plt.show()
```

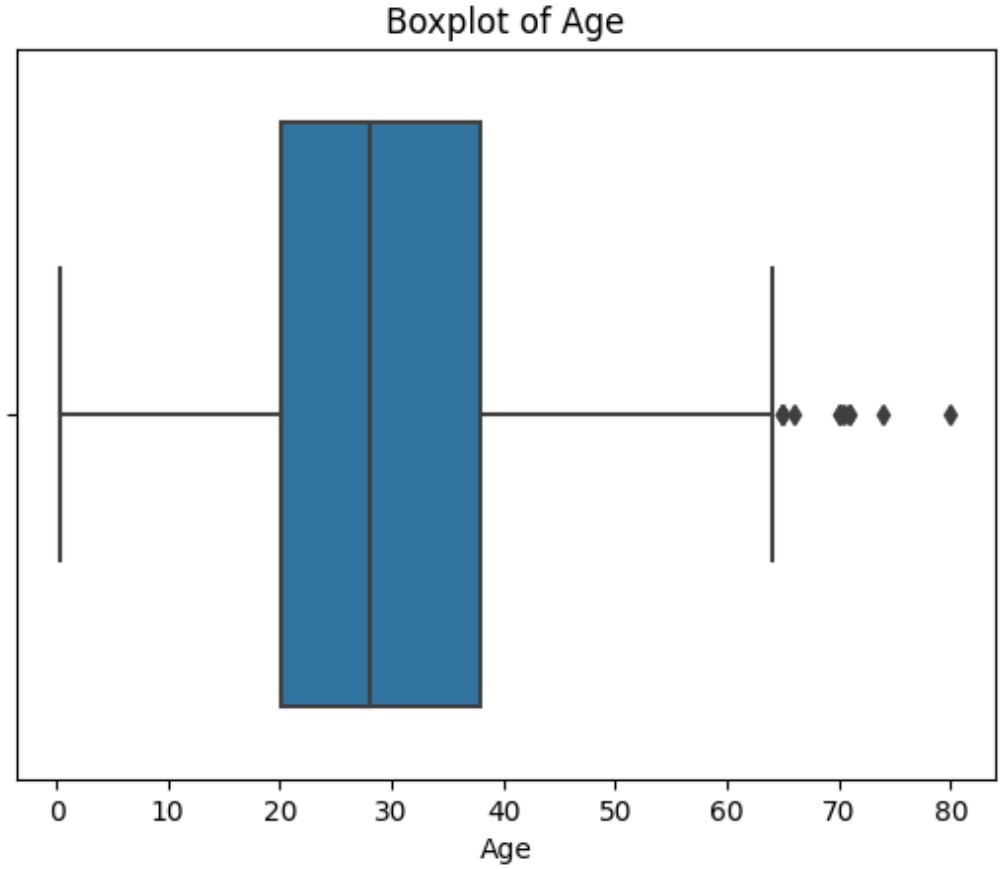
```
<ipython-input-12-e7b28d703787>:2: FutureWarning: The default value of numeric_only in
  correlation_matrix = df.corr()
```




```
# Outlier Detection
# Using Isolation Forest for outlier detection on the 'Fare' column
iso_forest = IsolationForest(contamination=0.05)
df['IsOutlier'] = iso_forest.fit_predict(df[['Fare']])
outliers = df[df['IsOutlier'] == -1]
print("Outliers Detected:\n", outliers)
```

```
337      0     16966  134.5000        C         -1
341      2     19950  263.0000        S         -1
373      0  PC 17760  135.6333        C         -1
377      2    113503  211.5000        C         -1
380      0  PC 17757  227.5250        C         -1
390      2    113760  120.0000        S         -1
435      2    113760  120.0000        S         -1
438      4     19950  263.0000        S         -1
498      2    113781  151.5500        S         -1
527      0  PC 17483  221.7792        S         -1
537      0  PC 17761  106.4250        C         -1
544      0  PC 17761  106.4250        C         -1
557      0  PC 17757  227.5250        C         -1
609      0  PC 17582  153.4625        S         -1
660      0  PC 17611  133.6500        S         -1
679      1  PC 17755  512.3292        C         -1
689      1     24160  211.3375        S         -1
700      0  PC 17757  227.5250        C         -1
708      0    113781  151.5500        S         -1
716      0  PC 17757  227.5250        C         -1
730      0     24160  211.3375        S         -1
737      0  PC 17755  512.3292        C         -1
742      2  PC 17608  262.3750        C         -1
763      2    113760  120.0000        S         -1
779      1     24160  211.3375        S         -1
802      2    113760  120.0000        S         -1
856      1     36928  164.8667        S         -1
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not
  warnings.warn(
```

```
# Outlier detection using boxplot and IQR for 'Age' feature
sns.boxplot(data=df, x='Age')
plt.title('Boxplot of Age')
plt.xlabel('Age')
plt.show()
# Calculate the IQR for 'Age'
Q1 = df['Age'].quantile(0.25)
Q3 = df['Age'].quantile(0.75)
IQR = Q3 - Q1
# Define the lower and upper bounds for outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
# Detect outliers
outliers_age = df[(df['Age'] < lower_bound) | (df['Age'] > upper_bound)]
print("Outliers Detected via IQR for 'Age':\n", outliers_age)
```




```
Outliers Detected via IQR for 'Age':
     PassengerId  Survived  Pclass                                     Name  \
33            34         0       2                  Wheadon, Mr. Edward H
54            55         0       1         Ostby, Mr. Engelhart Cornelius
96            97         0       1              Goldschmidt, Mr. George B
116          117         0       3                   Connors, Mr. Patrick
280          281         0       3                       Duane, Mr. Frank
456          457         0       1              Millet, Mr. Francis Davis
493          494         0       1                Artagaveytia, Mr. Ramon
630          631         1       1  Barkworth, Mr. Algernon Henry Wilson
672          673         0       2            Mitchell, Mr. Henry Michael
745          746         0       1           Crosby, Capt. Edward Gifford
851          852         0       3                     Svensson, Mr. Johan

      Sex   Age  SibSp  Parch      Ticket     Fare Embarked  IsOutlier
33   male  66.0      0      0  C.A. 24579  10.5000        S          1
```

```
# Splitting into dependent and independent variables
X = df.drop(['Survived'], axis=1)  # Independent variables
y = df['Survived']  # Dependent variable
```

```
# Encoding categorical variables (e.g., 'Sex' and 'Embarked')
label_encoder = LabelEncoder()
X['Sex'] = label_encoder.fit_transform(X['Sex'])
X['Embarked'] = label_encoder.fit_transform(X['Embarked'])
```

```
X.head(10)
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	3	Braund, Mr. Owen Harris	1	22.0	1	0	A/5 21171	7.2500	2
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	38.0	1	0	PC 17599	71.2833	0
2	3	3	Heikkinen, Miss. Laina	0	26.0	0	0	STON/O2. 3101282	7.9250	2
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	35.0	1	0	113803	53.1000	2
4	5	3	Allen, Mr. William Henry	1	35.0	0	0	373450	8.0500	2
5	6	3	Moran, Mr. James	1	NaN	0	0	330877	8.4583	1
6	7	1	McCarthy, Mr. Timothy J	1	54.0	0	0	17463	51.8625	2
7	8	3	Palsson, Master. Gosta Leonard	1	2.0	3	1	349909	21.0750	2
8	9	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	0	27.0	0	2	347742	11.1333	2
9	10	2	Nasser, Mrs. Nicholas	0	14.0	1	0	237736	30.0708	0

```
# Feature Scaling (Standardization)
scaler = StandardScaler()
X[['Age', 'Fare']] = scaler.fit_transform(X[['Age', 'Fare']])
```

```
X.head(10)
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Eml
0	1	3	Braund, Mr. Owen Harris	1	-0.530377	1	0	A/5 21171	-0.502445	
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	0.571831	1	0	PC 17599	0.786845	
2	3	3	Heikkinen, Miss. Laina	0	-0.254825	0	0	STON/O2. 3101282	-0.488854	
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	0.365167	1	0	113803	0.420730	
4	5	3	Allen, Mr. William Henry	1	0.365167	0	0	373450	-0.486337	
5	6	3	Moran, Mr. James	1	NaN	0	0	330877	-0.478116	
6	7	1	McCarthy, Mr. Timothy J	1	1.674039	0	0	17463	0.395814	
7	8	3	Palsson, Master. Gosta Leonard	1	-1.908136	3	1	349909	-0.224083	
8	9	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	0	-0.185937	0	2	347742	-0.424256	
9	10	2	Nasser, Mrs. Nicholas	0	-1.081480	1	0	237736	-0.042956	

```
# Splitting data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Display shapes of the training and testing sets
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

```
X_train shape: (712, 11)
X_test shape: (179, 11)
y_train shape: (712,)
y_test shape: (179,)
```

✓ 0s completed at 12:10 PM