```
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 LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix



# Load the Titanic dataset
titanic_data = pd.read_csv("/content/archive.zip")

# Display the first 5 rows
print(titanic_data.head())
```

```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

Double-click (or enter) to edit

```
# Fill missing Age with median and Embarked with the mode
titanic_data['Age'].fillna(titanic_data['Age'].median(), inplace=True)
titanic_data['Embarked'].fillna(titanic_data['Embarked'].mode()[0], inplace=True)

# Drop irrelevant columns
titanic_data.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1, inplace=True)

# Encode categorical variables
label_encoder = LabelEncoder()
titanic_data['Sex'] = label_encoder.fit_transform(titanic_data['Sex'])
titanic_data['Embarked'] = label_encoder.fit_transform(titanic_data['Embarked'])
```

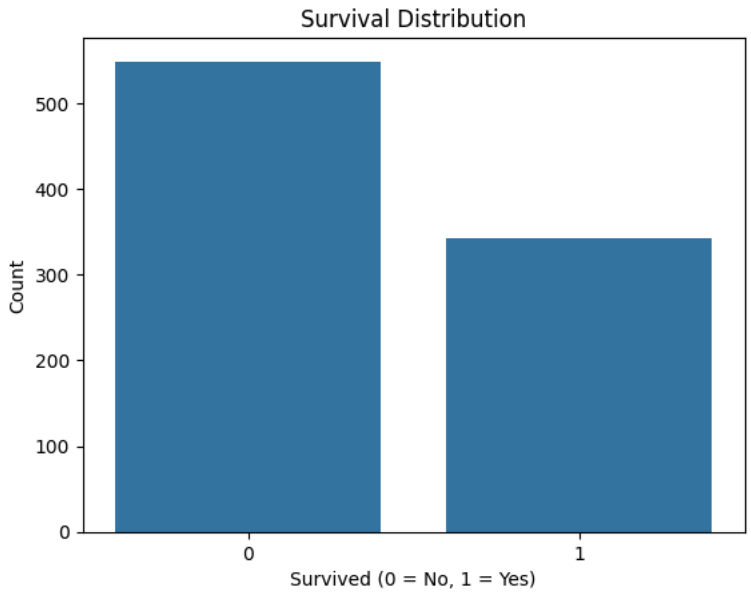
```
<ipython-input-4-f1a4abb7f328>:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  titanic_data['Age'].fillna(titanic_data['Age'].median(), inplace=True)
<ipython-input-4-f1a4abb7f328>:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  titanic_data['Embarked'].fillna(titanic_data['Embarked'].mode()[0], inplace=True)
```

```
# Display dataset info and summary statistics
print(titanic_data.info())
print(titanic_data.describe())
```

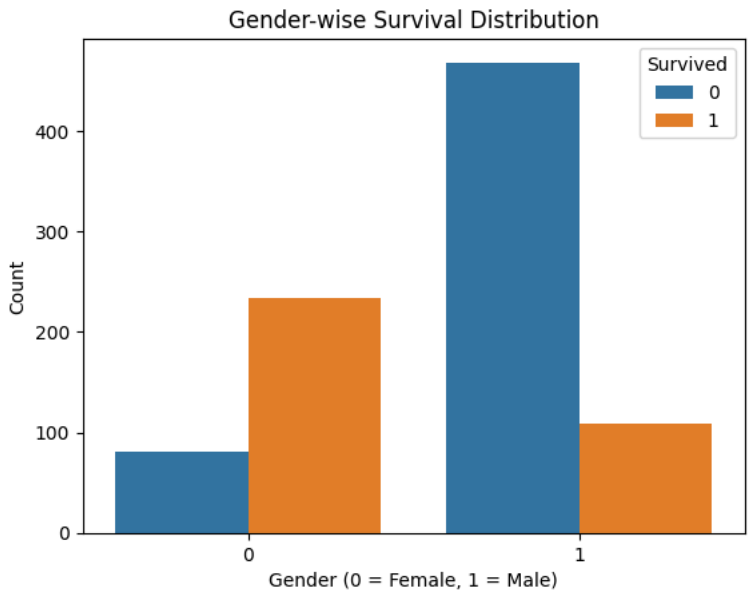
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Sex       891 non-null    int64
```

```
 3   Age       891 non-null    float64
 4   SibSp     891 non-null    int64
 5   Parch     891 non-null    int64
 6   Fare      891 non-null    float64
 7   Embarked  891 non-null    int64
dtypes: float64(2), int64(6)
memory usage: 55.8 KB
None
         Survived      Pclass         Sex         Age       SibSp       Parch  \
count  891.000000  891.000000  891.000000  891.000000  891.000000  891.000000
mean     0.383838    2.308642    0.647587   29.361582    0.523008    0.381594
std      0.486592    0.836071    0.477990   13.019697    1.102743    0.806057
min      0.000000    1.000000    0.000000    0.420000    0.000000    0.000000
25%      0.000000    2.000000    0.000000   22.000000    0.000000    0.000000
50%      0.000000    3.000000    1.000000   28.000000    0.000000    0.000000
75%      1.000000    3.000000    1.000000   35.000000    1.000000    0.000000
max      1.000000    3.000000    1.000000   80.000000    8.000000    6.000000

             Fare    Embarked
count  891.000000  891.000000
mean    32.204208    1.536476
std     49.693429    0.791503
min      0.000000    0.000000
25%      7.910400    1.000000
50%     14.454200    2.000000
75%     31.000000    2.000000
max    512.329200    2.000000
```

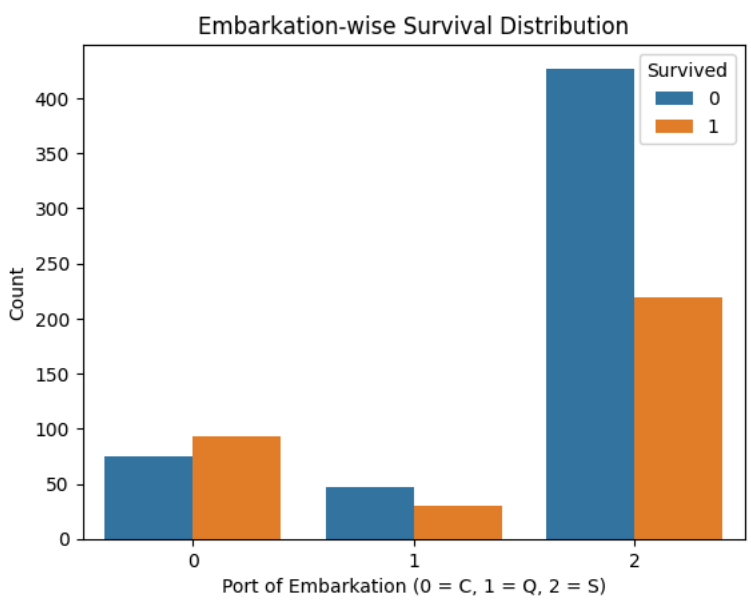
```
sns.countplot(data=titanic_data, x='Survived')
plt.title('Survival Distribution')
plt.xlabel('Survived (0 = No, 1 = Yes)')
plt.ylabel('Count')
plt.show()
```




```
sns.countplot(data=titanic_data, x='Sex', hue='Survived')
plt.title('Gender-wise Survival Distribution')
plt.xlabel('Gender (0 = Female, 1 = Male)')
plt.ylabel('Count')
plt.legend(title='Survived', loc='upper right')
plt.show()
```




```
sns.countplot(data=titanic_data, x='Embarked', hue='Survived')
plt.title('Embarkation-wise Survival Distribution')
plt.xlabel('Port of Embarkation (0 = C, 1 = Q, 2 = S)')
plt.ylabel('Count')
plt.legend(title='Survived', loc='upper right')
plt.show()
```

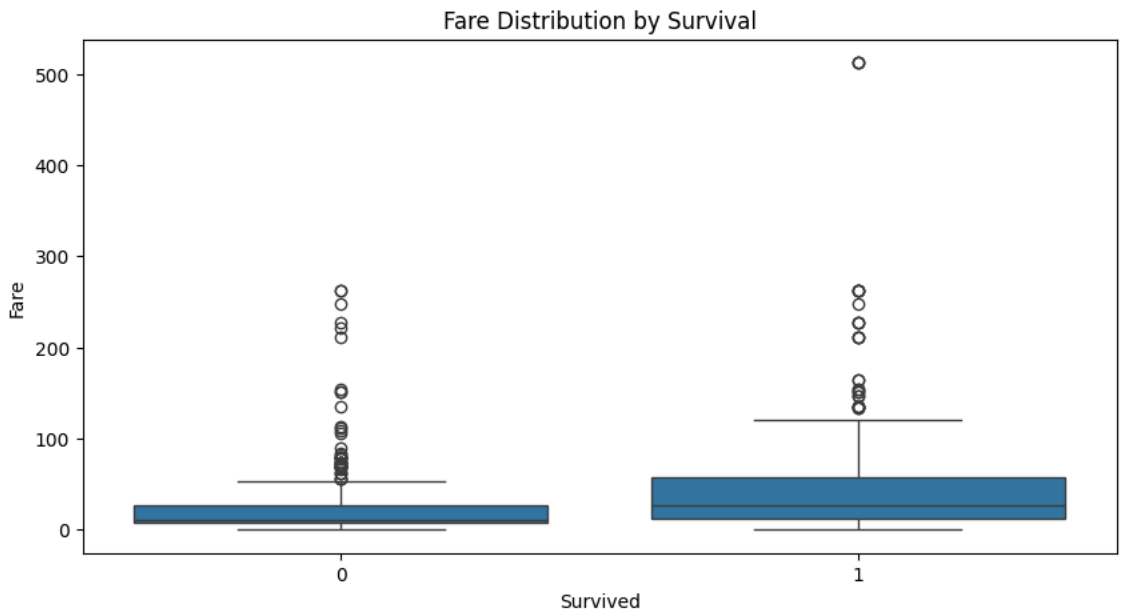


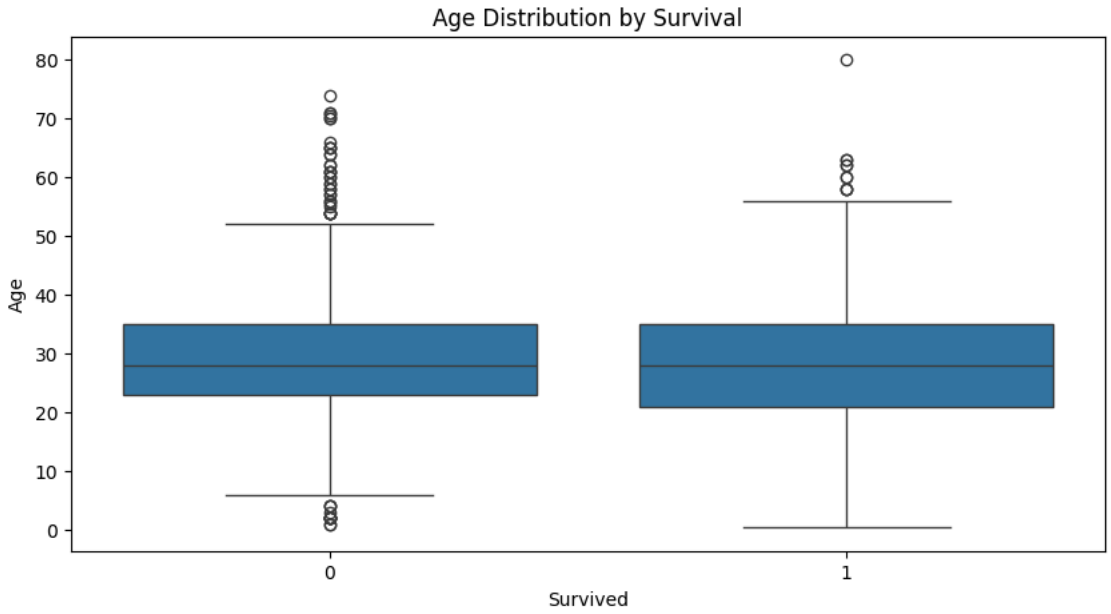

```
missing_values = titanic_data.isnull().sum()
print("Missing values in each column:")
print(missing_values)
```

```
Missing values in each column:
Survived    0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Embarked    0
dtype: int64
```

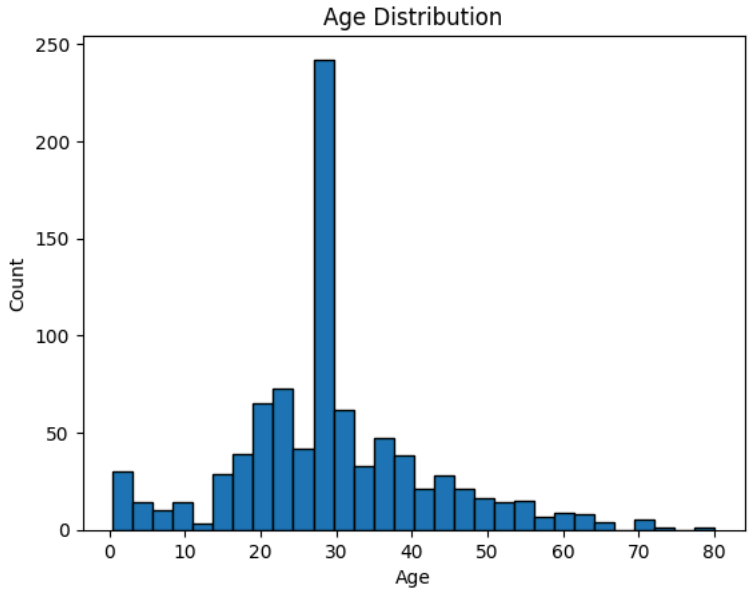
```
plt.figure(figsize=(10, 5))
sns.boxplot(data=titanic_data, x='Survived', y='Fare')
plt.title('Fare Distribution by Survival')
plt.show()

plt.figure(figsize=(10, 5))
sns.boxplot(data=titanic_data, x='Survived', y='Age')
```

```
plt.title('Age Distribution by Survival')
plt.show()
```

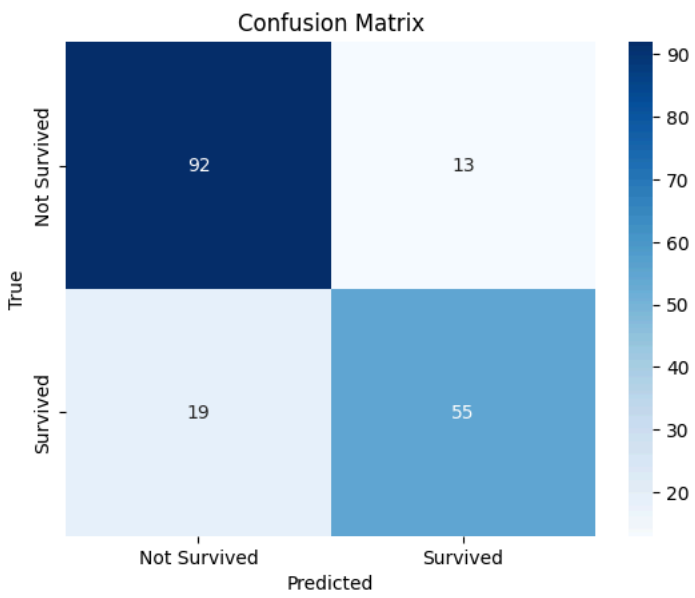




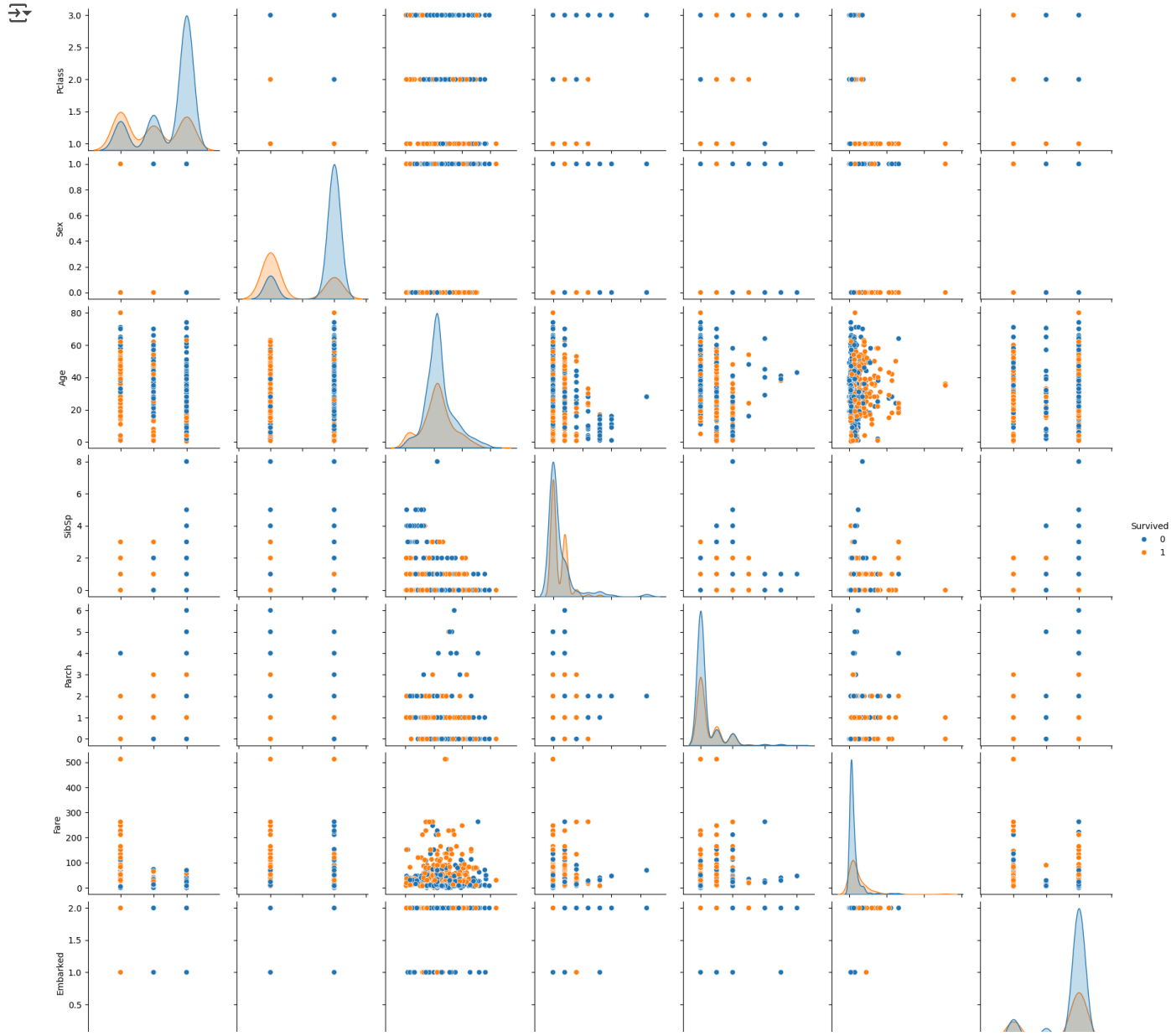

```
plt.hist(titanic_data['Age'], bins=30, edgecolor='black')
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```




```
# Split and train the model first
X = titanic_data.drop('Survived', axis=1)
y = titanic_data['Survived']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

# Confusion Matrix Heatmap
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['Not Survived', 'Survived'], yticklabels=['Not Survived', 'Surv
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```




```
sns.pairplot(titanic_data, hue='Survived', diag_kind='kde')
plt.show()
```


Pclass
Sex
Age
SibSp
Parch
Fare
Embarked
Survived
0
1