Titanic - Machine Learning from Disaster

1. Introduction

This project analyzes passenger data from the Titanic disaster to predict survival outcomes using machine learning. We use data cleaning, feature engineering, and a logistic regression model to predict whether a passenger survived.

Dataset: Kaggle Titanic Competition

2. Import Libraries

In [45]:
```
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.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

sns.set(style="whitegrid")
```

3. load the dataset

In [46]:
```
df = pd.read_csv(r"C:\Users\Roshni\Downloads\titanic-survival-prediction\data\train.csv")
df.head()
```

Out[46]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarke
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	

4. Basic data exploration

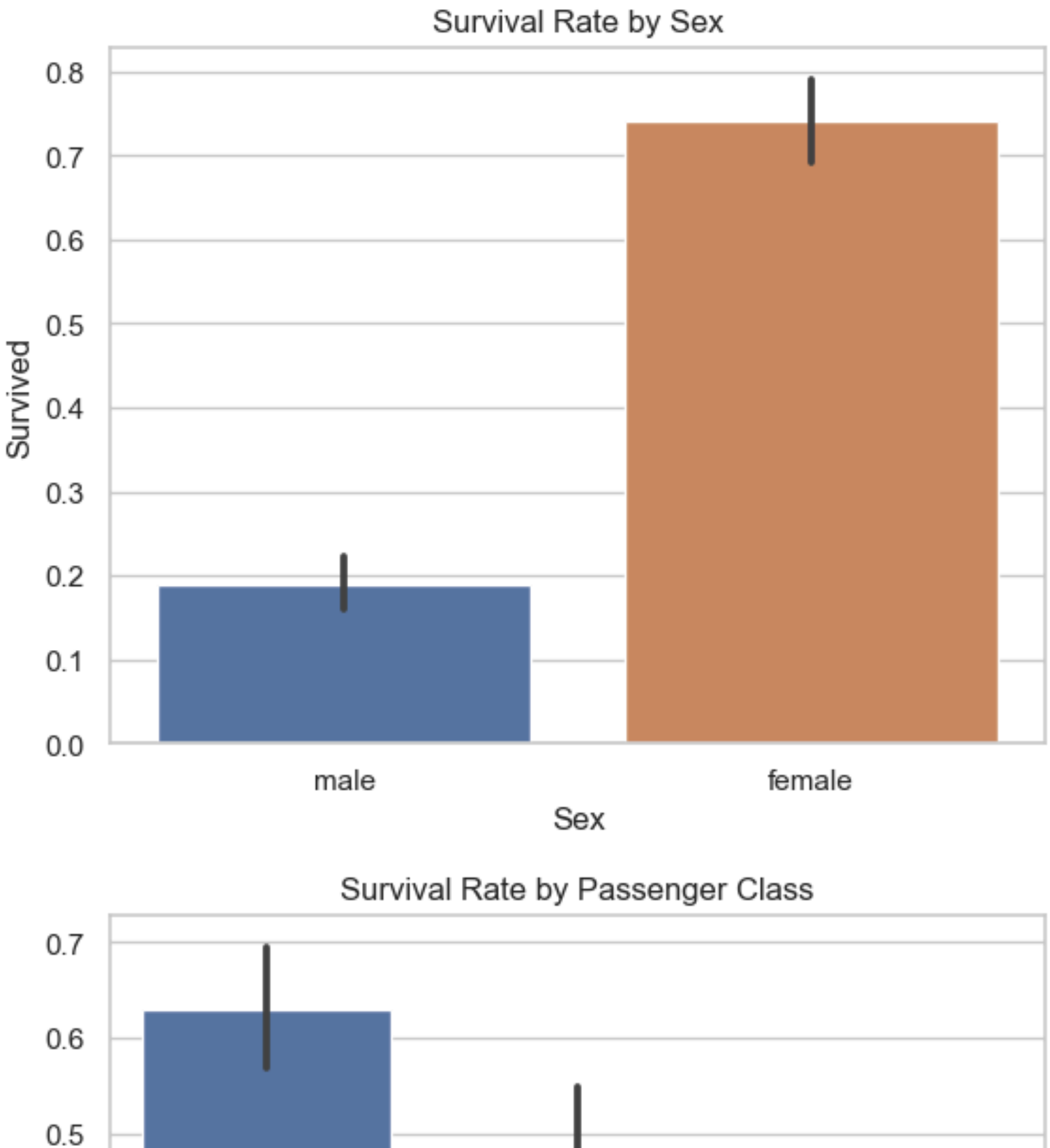
In [47]:
```
df.shape
df.info()
df.describe(include='all')
df.isnull().sum()
```

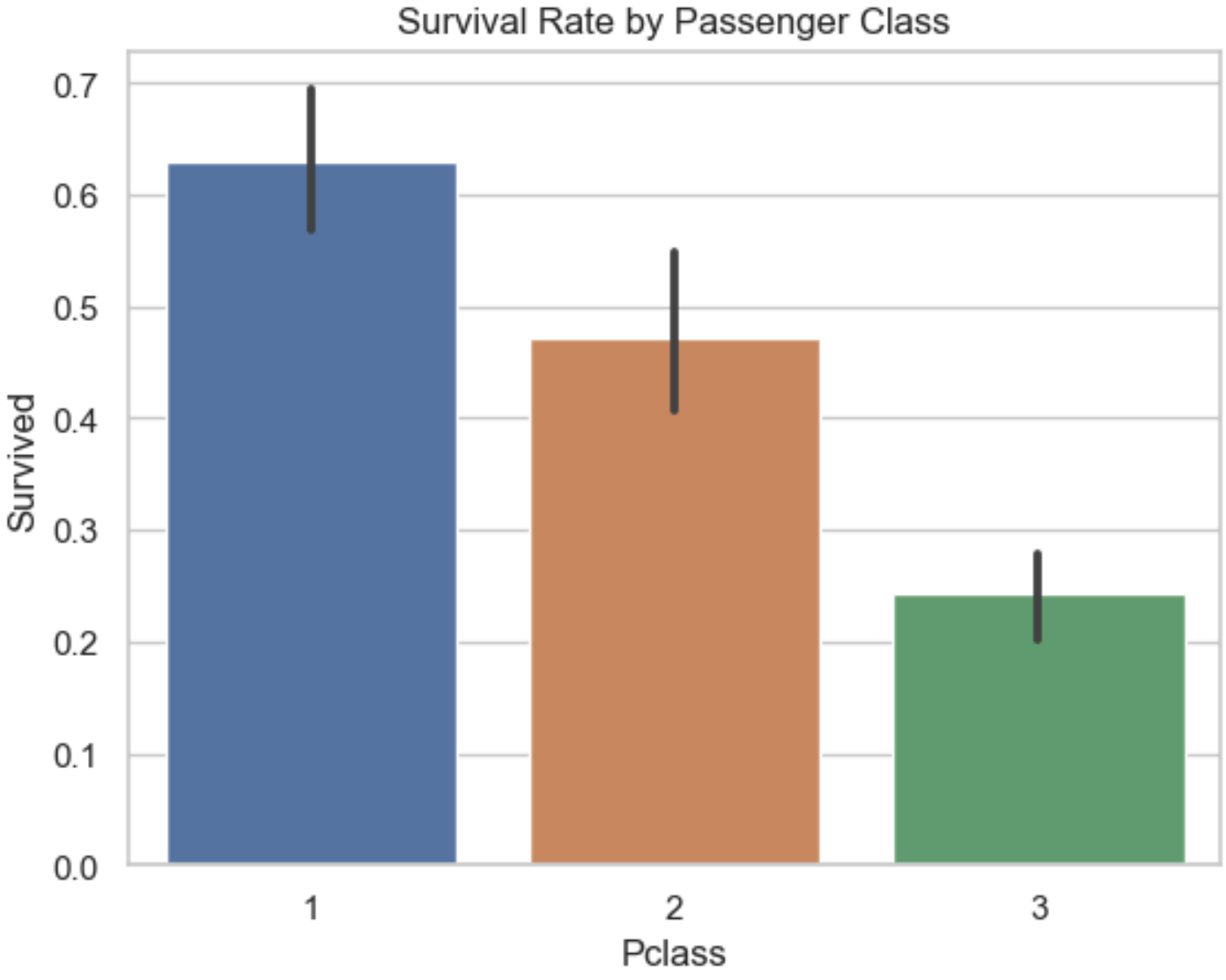
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

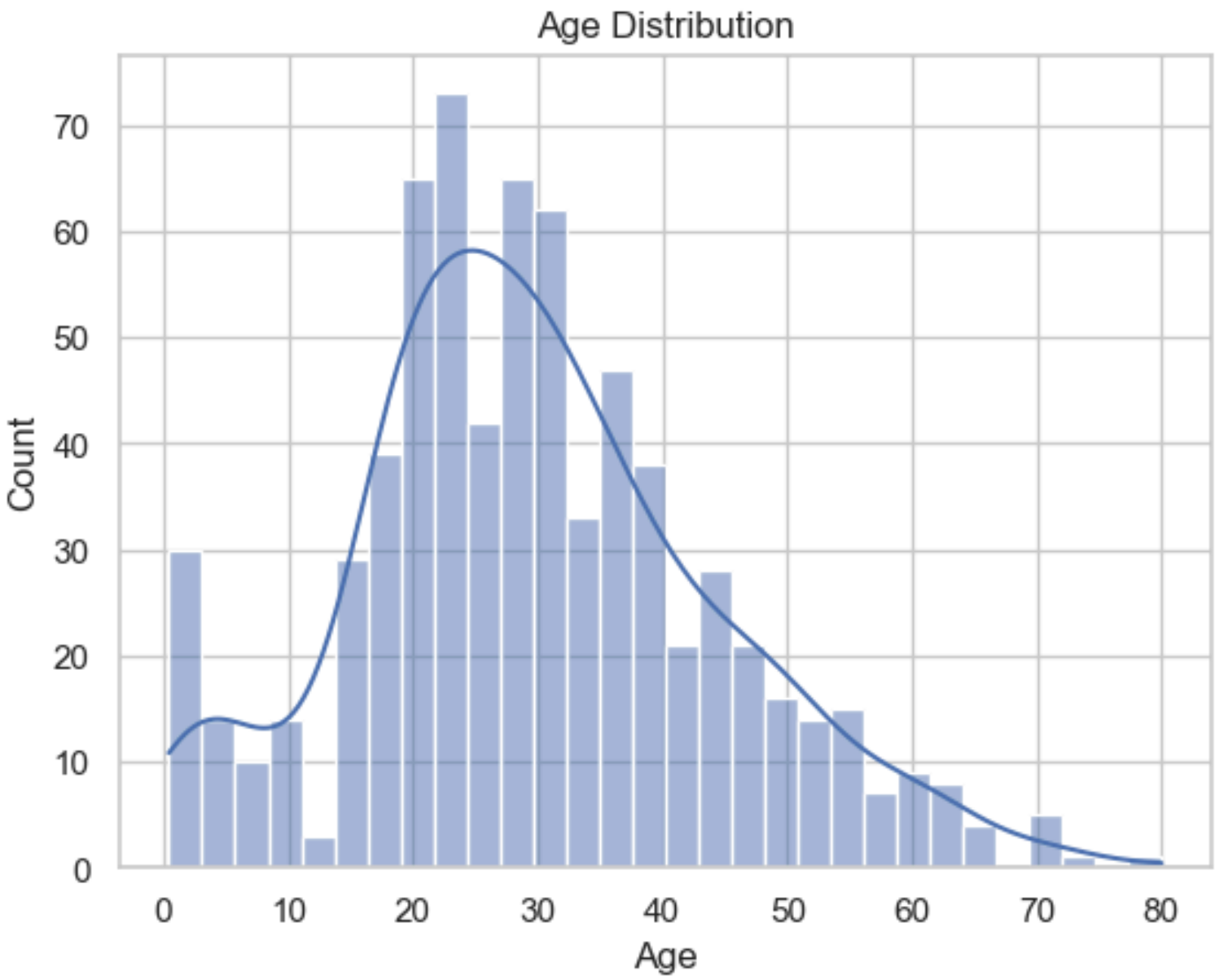
Out[47]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

5. Data Visualization

In [48]:
```
# Survival by Sex
sns.barplot(x="Sex", y="Survived", data=df)
plt.title("Survival Rate by Sex")
plt.show()

# Survival by Passenger Class
sns.barplot(x="Pclass", y="Survived", data=df)
plt.title("Survival Rate by Passenger Class")
plt.show()

# Age distribution
sns.histplot(df["Age"].dropna(), kde=True, bins=30)
plt.title("Age Distribution")
plt.show()
```


Survival Rate by Sex
Survived
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
male
female
Sex


Survival Rate by Passenger Class
Survived
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
1
2
3
Pclass



6. Data Cleaning

In [49]:
```
# Fill missing Age with median
df['Age'].fillna(df['Age'].median(), inplace=True)

# Fill missing Embarked with mode
df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)

# Drop Cabin (too many missing values)
df.drop('Cabin', axis=1, inplace=True)

# Drop Ticket and Name for simplicity
df.drop(['Ticket', 'Name'], axis=1, inplace=True)
```

7. Feature Engineering

In [50]:
```
# Convert Sex to numeric
df['Sex'] = df['Sex'].map({'male': 0, 'female': 1})

# One-hot encode Embarked
df = pd.get_dummies(df, columns=['Embarked'], drop_first=True)
```

8. Prepare Data for Modeling

In [51]:
```
X = df.drop(['PassengerId', 'Survived'], axis=1)
y = df['Survived']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

9. Train Logistic Regression Model

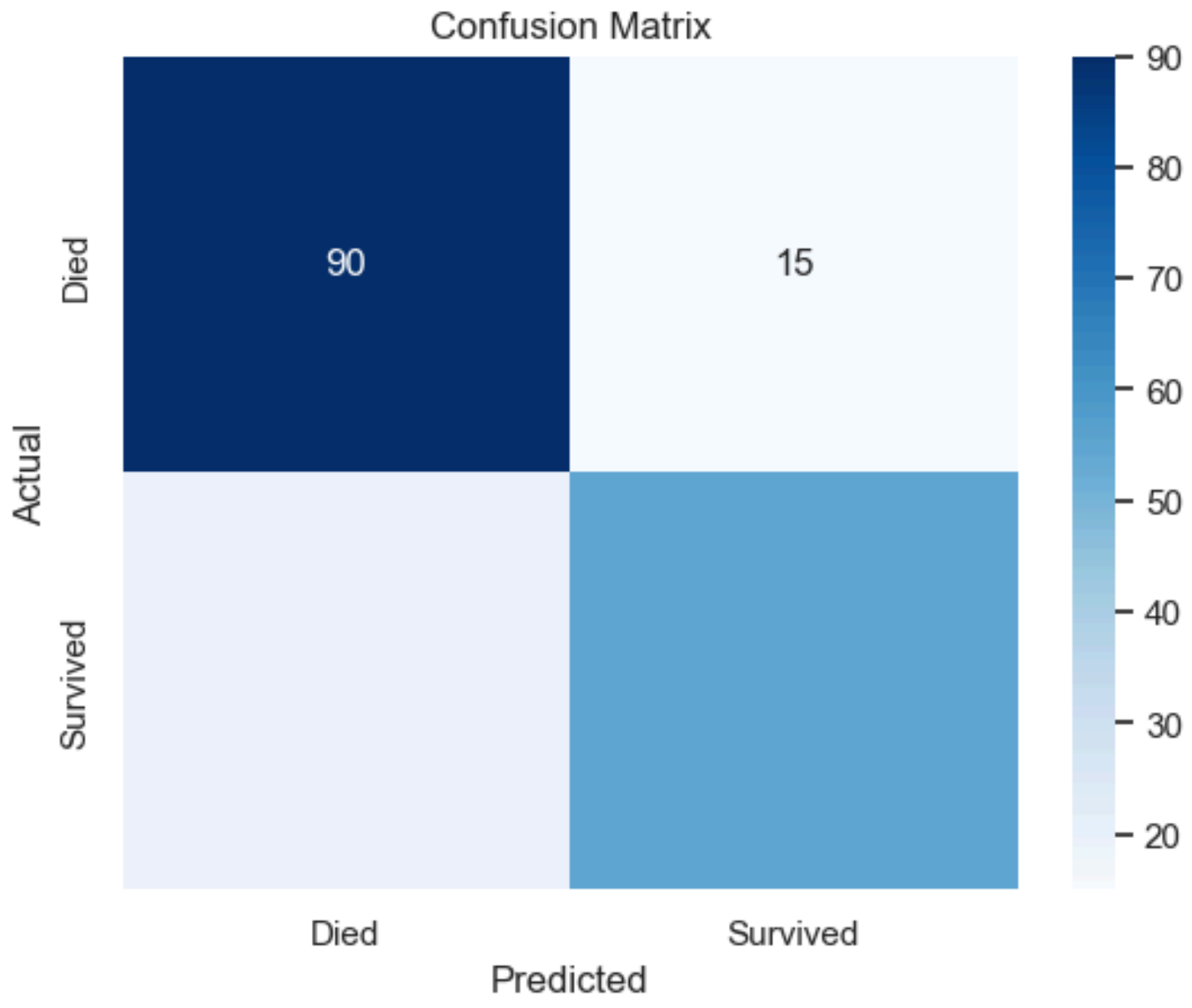
In [52]:
```
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

```
y_pred = model.predict(X_test)
```

10. Evaluate Model

In [53]:
```
# Accuracy and metrics
print("Accuracy Score:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Died", "Survived"], ytickl
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

```
Accuracy Score: 0.8100558659217877

Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.86      0.84       105
           1       0.79      0.74      0.76        74

    accuracy                           0.81       179
   macro avg       0.81      0.80      0.80       179
weighted avg       0.81      0.81      0.81       179
```



11. Conclusion

- The logistic regression model achieved an accuracy of around **81%**.
- Gender and passenger class had strong influence on survival chances.
- I cleaned missing values, encoded categorical variables, and created a simple prediction model.
- Future improvements: try Random Forests, feature selection, hyperparameter tuning.

In [ ]: