Step 1: Import Libraries & Load Dataset

```
import pandas as pd

train_df = pd.read_csv(r"C:\Users\Asus\Desktop\Elevate_Labs_day5_11-08-2025_\Datasets\train.csv")
test_df = pd.read_csv(r"C:\Users\Asus\Desktop\Elevate_Labs_day5_11-08-2025_\Datasets\test.csv")
gender_df = pd.read_csv(r"C:\Users\Asus\Desktop\Elevate_Labs_day5_11-08-2025_\Datasets\gender_submission.csv")
```

Head of train DataFrame

```
train_df.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
```

Step 2: Initial Data Inspection

```
train_df.info()
```

```
<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
```

Step 3: Data Cleaning

Filling missing 'Embarked' with mode

```
train_df['Embarked'] =
train_df['Embarked'].fillna(train_df['Embarked'].mode()[0])
```

Fill missing 'Age' with median

```
train_df['Age'] = train_df['Age'].fillna(train_df['Age'].median())
```

Creating a feature for Cabin availability

```
train_df['Has_Cabin'] = train_df['Cabin'].notnull().astype(int)
```

Droping 'Cabin' column (too many missing values)

```
train_df = train_df.drop('Cabin', axis=1)
```

Drop irrelevant columns for EDA

```
train_df = train_df.drop(['PassengerId', 'Name', 'Ticket'], axis=1)
```

Verify changes

```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 9 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Survived   891 non-null    int64  
 1   Pclass     891 non-null    int64  
 2   Sex        891 non-null    object 
 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    object 
 8   Has_Cabin  891 non-null    int64  
dtypes: float64(2), int64(5), object(2)
memory usage: 62.8+ KB
```

Step 4: Summary Statistics

Numeric summary

```
train_df.describe()
```

```
         Survived      Pclass         Age       SibSp       Parch  
Fare  \
count  891.000000  891.000000  891.000000  891.000000  891.000000  
891.000000   
mean     0.383838    2.308642   29.361582    0.523008    0.381594   
32.204208   
std      0.486592    0.836071   13.019697    1.102743    0.806057   
49.693429   
min      0.000000    1.000000    0.420000    0.000000    0.000000   
0.000000   
25%      0.000000    2.000000   22.000000    0.000000    0.000000   
7.910400   
50%      0.000000    3.000000   28.000000    0.000000    0.000000   
14.454200   
75%      1.000000    3.000000   35.000000    1.000000    0.000000   
31.000000   
max      1.000000    3.000000   80.000000    8.000000    6.000000   
512.329200   

        Has_Cabin  
count  891.000000  
mean     0.228956  
std      0.420397  
min      0.000000  
25%      0.000000  
```

```
50%        0.000000
75%        0.000000
max        1.000000
```

Summary Statistics – Observations

- **Survival Rate**: Mean of `Survived` = 0.38 → About 38% of passengers survived.
- **Passenger Class**: Mean `Pclass` = 2.31 → Most passengers were in 2nd or 3rd class.
- **Age**: Median age = 28, youngest = 0.42 years (infant), oldest = 80 years.
- **SibSp**: Median = 0 → Most passengers traveled without siblings/spouses.
- **Parch**: Median = 0 → Most passengers traveled without parents/children.
- **Fare**: Highly skewed, median = 14.45 but max = 512.33 → Indicates a few very expensive tickets.
- **Has_Cabin**: Mean = 0.23 → Only about 23% had cabin information (likely higher-class passengers).

Categorical summary

```
categorical_cols = ['Sex', 'Embarked', 'Pclass']
for col in categorical_cols:
    print(f"\nValue counts for {col}:")
    print(train_df[col].value_counts())


Value counts for Sex:
Sex
male      577
female    314
Name: count, dtype: int64

Value counts for Embarked:
Embarked
S    646
C    168
Q     77
Name: count, dtype: int64

Value counts for Pclass:
Pclass
3    491
1    216
2    184
Name: count, dtype: int64
```

Categorical Summary – Observations

- **Sex**: Majority were male (577), fewer females (314).
- **Embarked**:
  - Most passengers boarded from Southampton (S) – 646

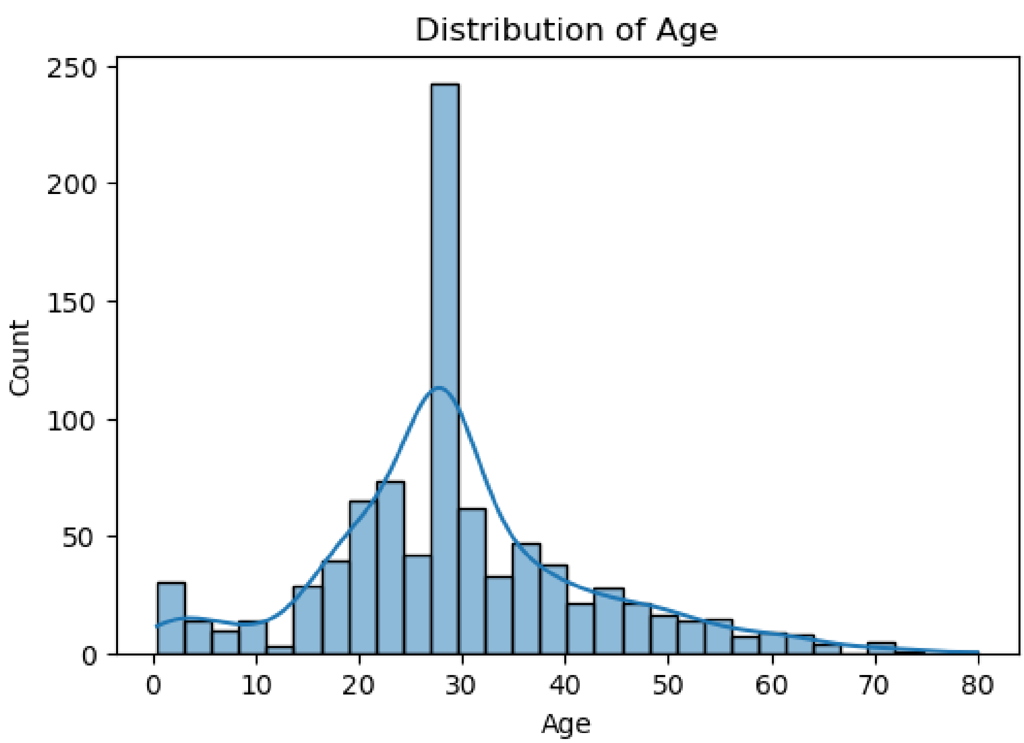
- Followed by Cherbourg (C) – 168
- Least from Queenstown (Q) – 77

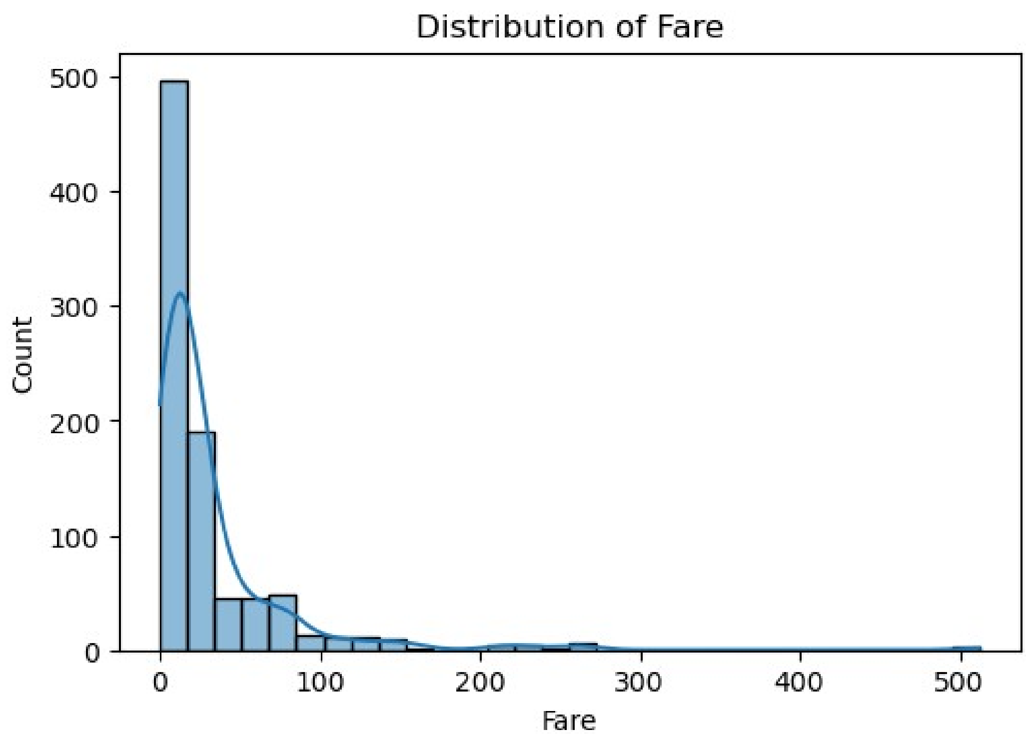
- **Pclass**:
  - Most passengers traveled in 3rd class – 491
  - 1st class – 216
  - 2nd class – 184

Step 5: Univariate Visualizations

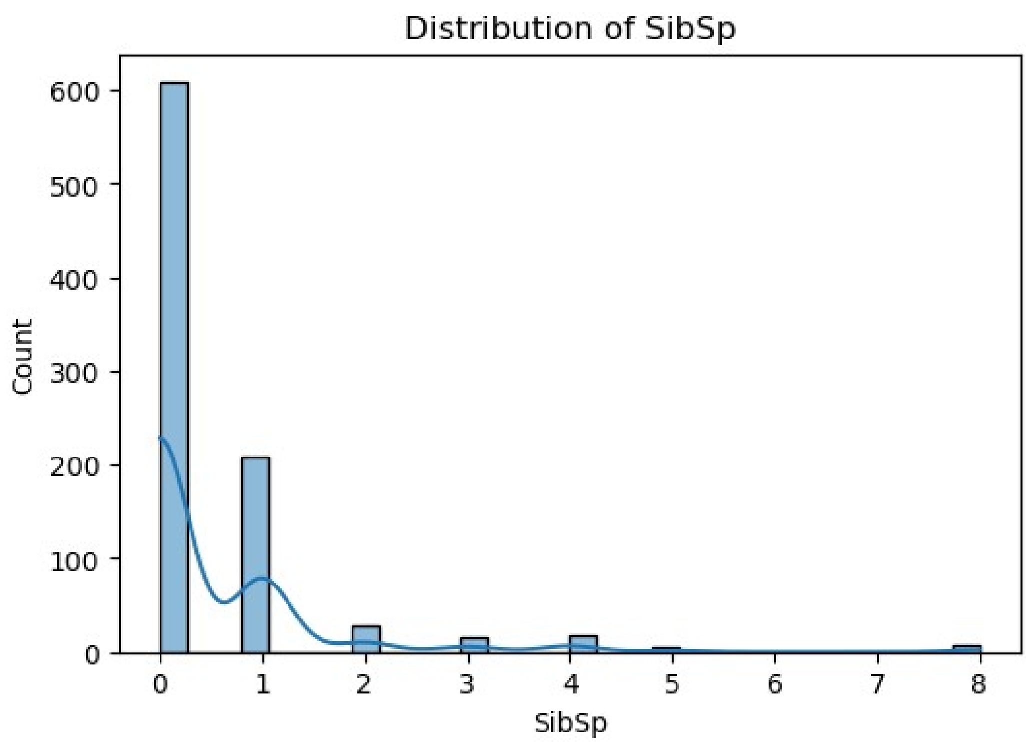
5.1 Numeric Columns – Histograms

```
import matplotlib.pyplot as plt
import seaborn as sns

numeric_cols = ['Age', 'Fare', 'SibSp', 'Parch']

for col in numeric_cols:
    plt.figure(figsize=(6,4))
    sns.histplot(train_df[col], kde=True, bins=30)
    plt.title(f'Distribution of {col}')
    plt.show()
```

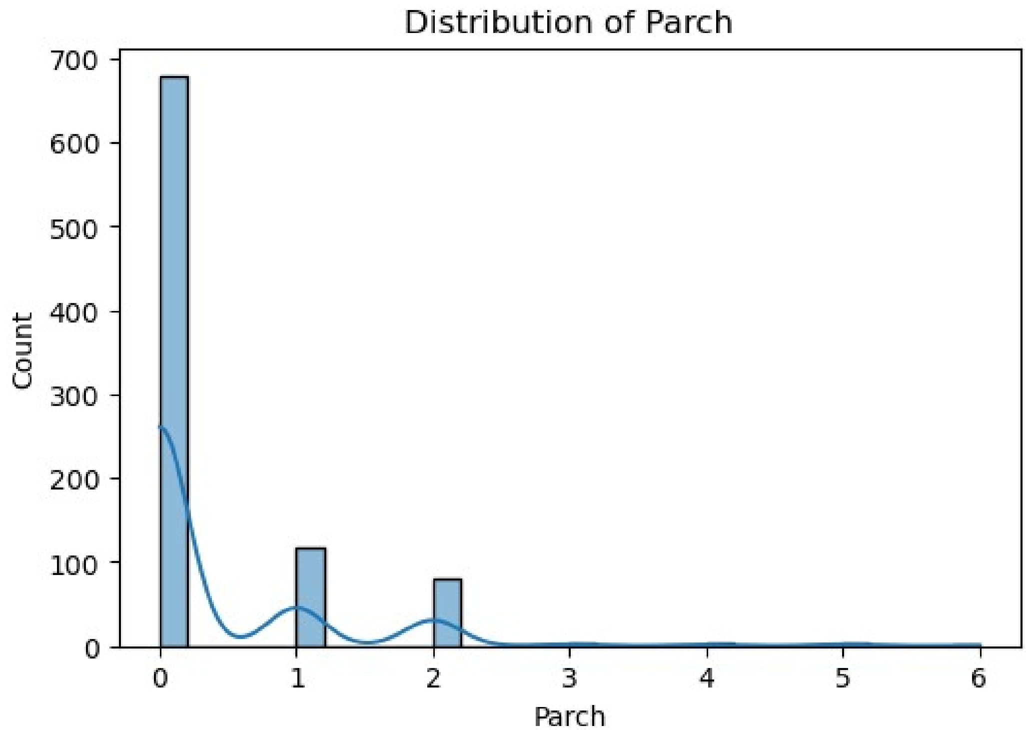




Distribution of Fare
Count
0
100
200
300
400
500
0
100
200
300
400
500
Fare



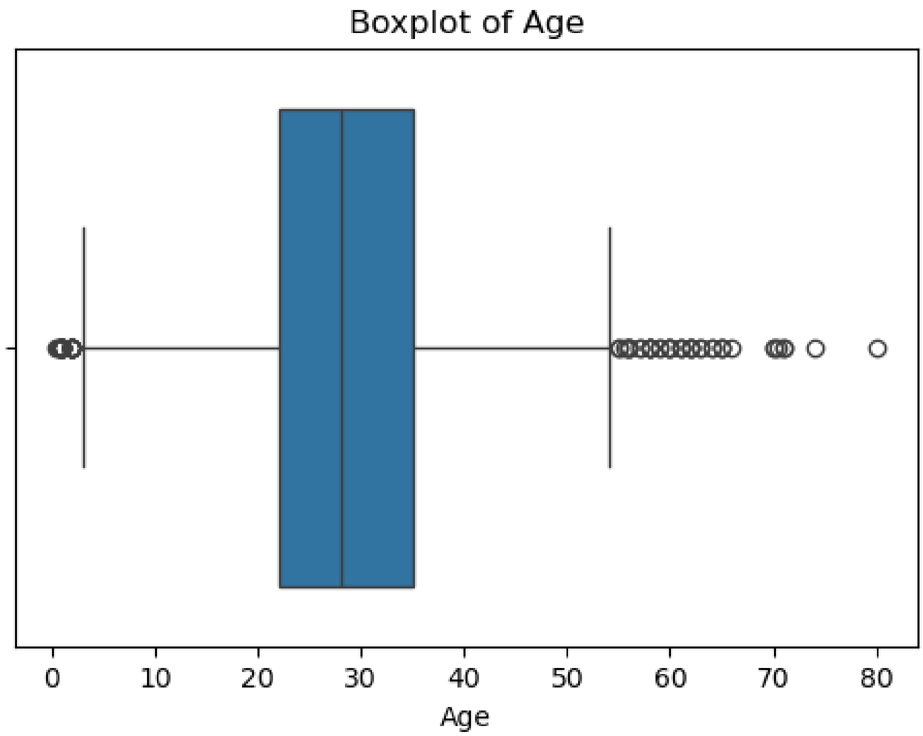
Distribution of SibSp
Count
0
100
200
300
400
500
600
0
1
2
3
4
5
6
7
8
SibSp

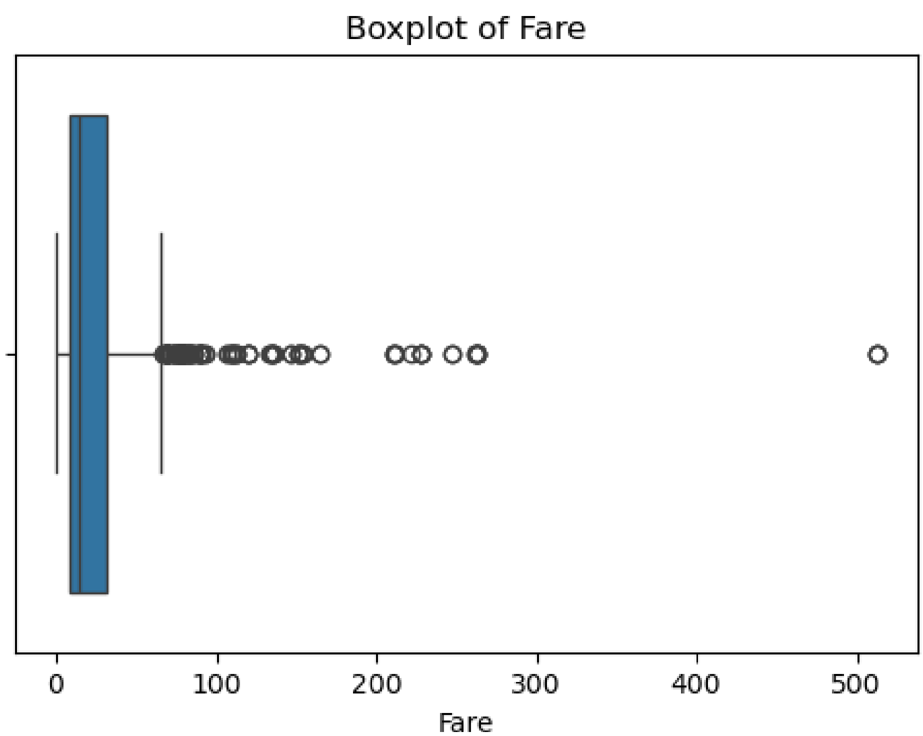


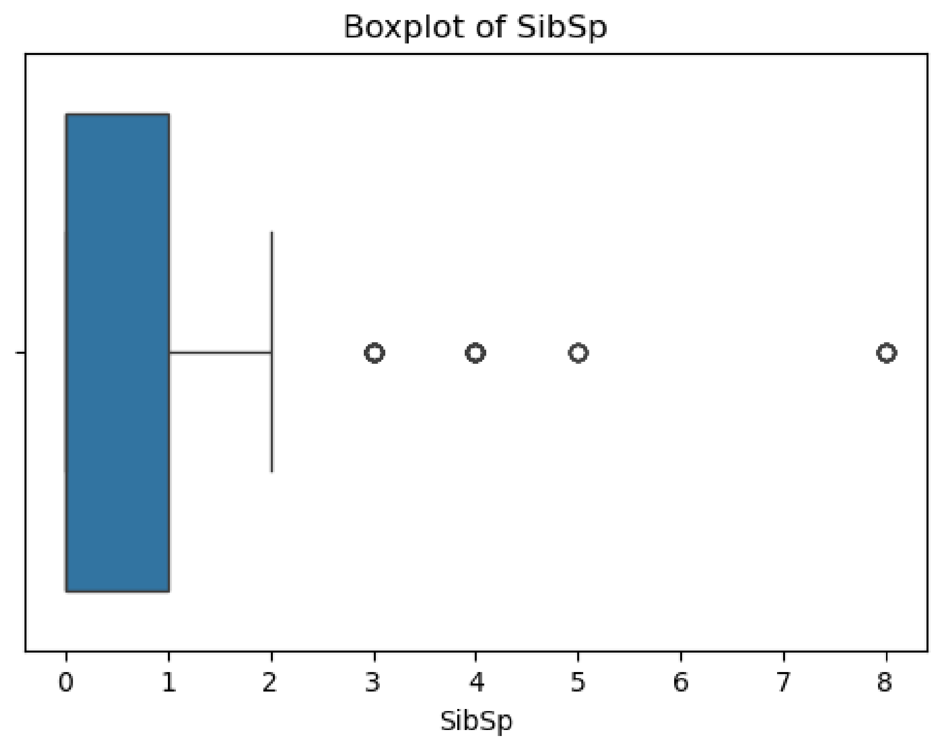
- **Age**: Most passengers are between 20–40 years old. A smaller peak is seen for children under 10.
- **Fare**: Highly skewed towards lower fares, with a few extreme high values.
- **SibSp**: Majority have 0 siblings/spouse aboard, small peaks at 1 and 2.
- **Parch**: Most passengers have 0 parents/children aboard, small peaks at 1–3.

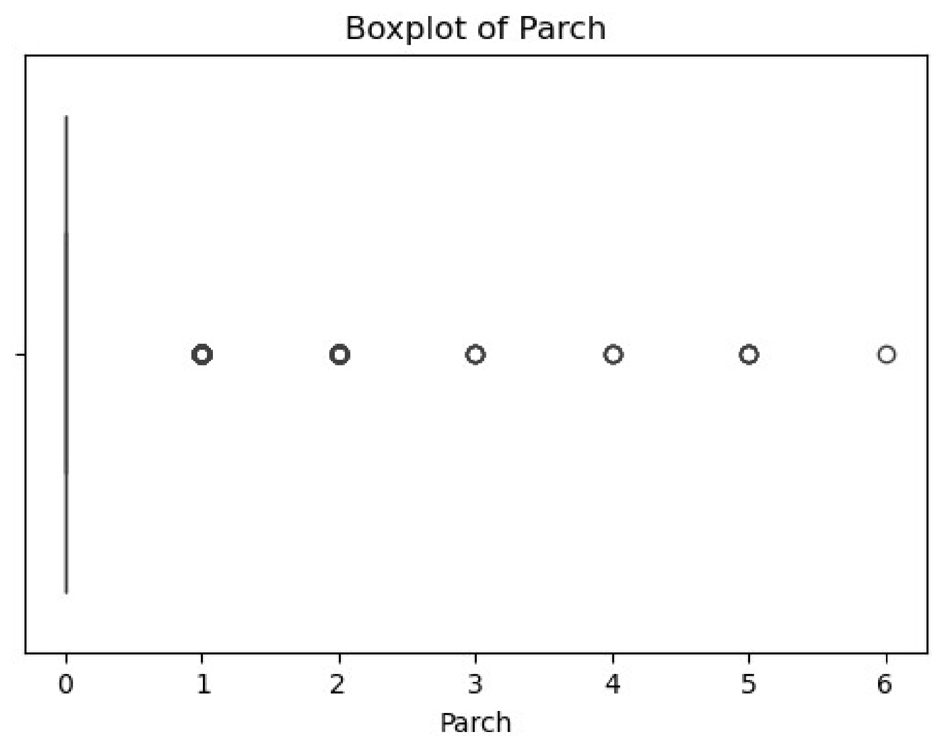
5.2 Numeric Columns – Boxplots (for Outliers)

```
for col in numeric_cols:
    plt.figure(figsize=(6,4))
    sns.boxplot(x=train_df[col])
    plt.title(f'Boxplot of {col}')
    plt.show()
```


Boxplot of Age
0
10
20
30
40
50
60
70
80
Age


Boxplot of Fare
0
100
200
300
400
500
Fare



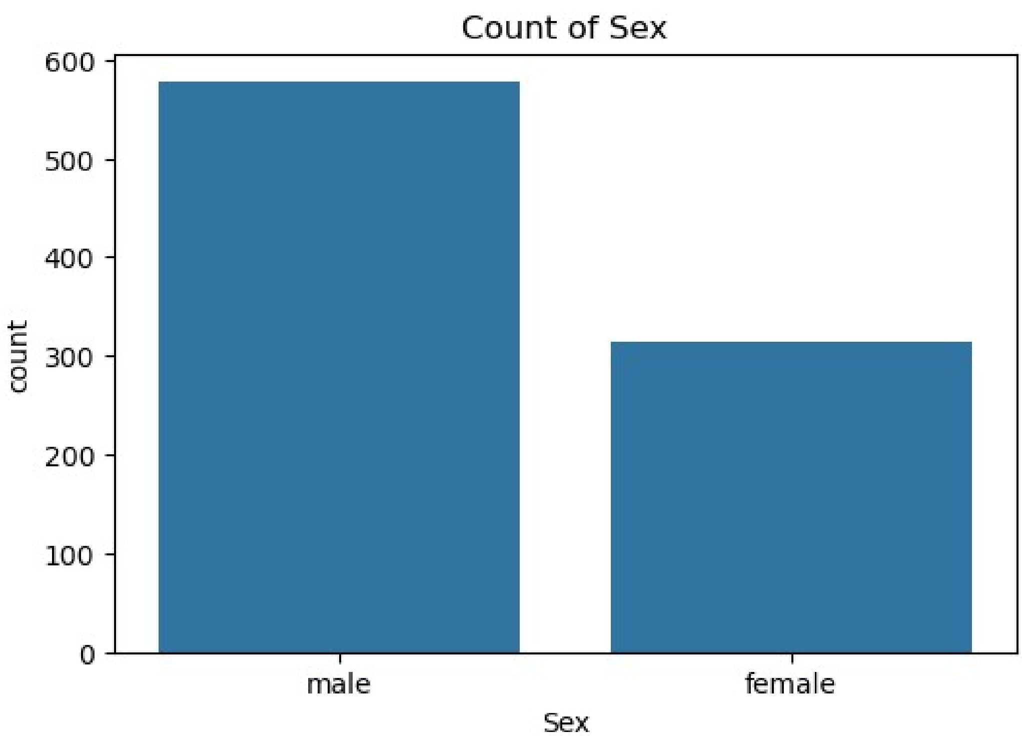


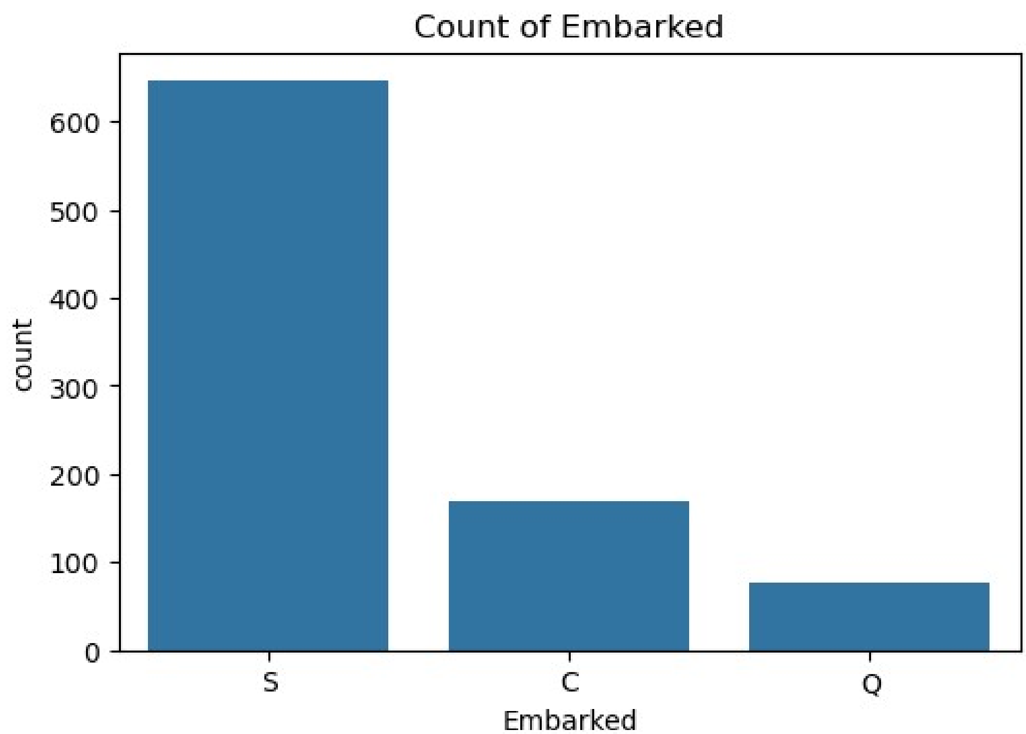
- **Age**: Outliers present, especially older passengers above 60.
- **Fare**: Strong right skew, with several high-fare outliers above 200.

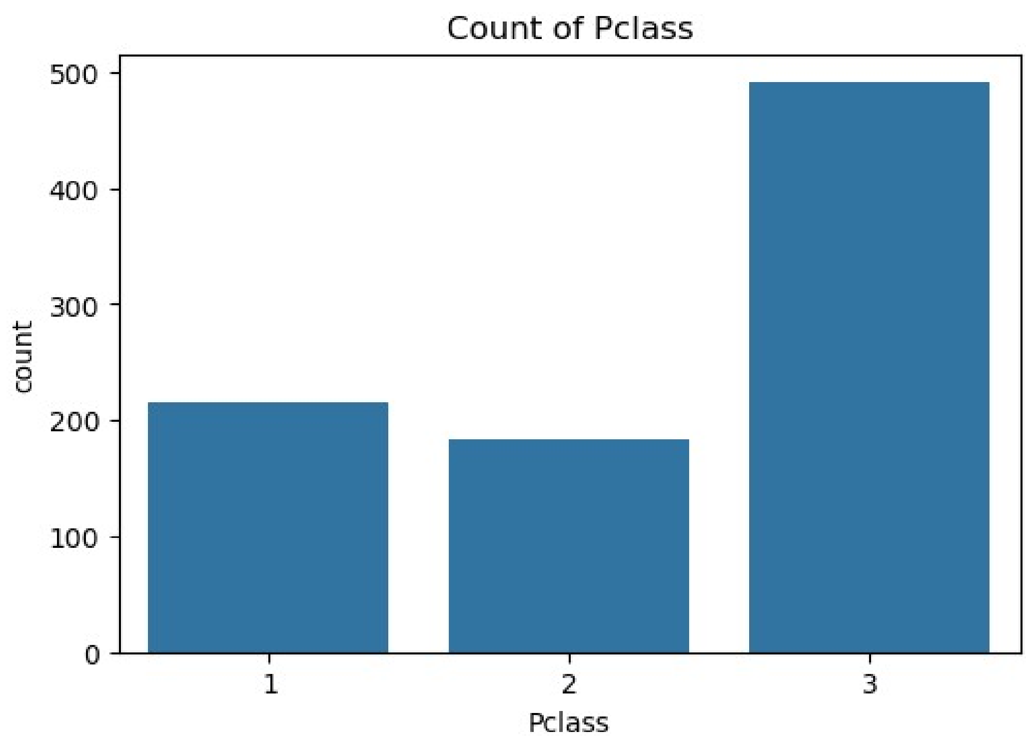
- **SibSp**: Outliers exist for very large sibling/spouse counts.
- **Parch**: A few outliers with large family sizes.

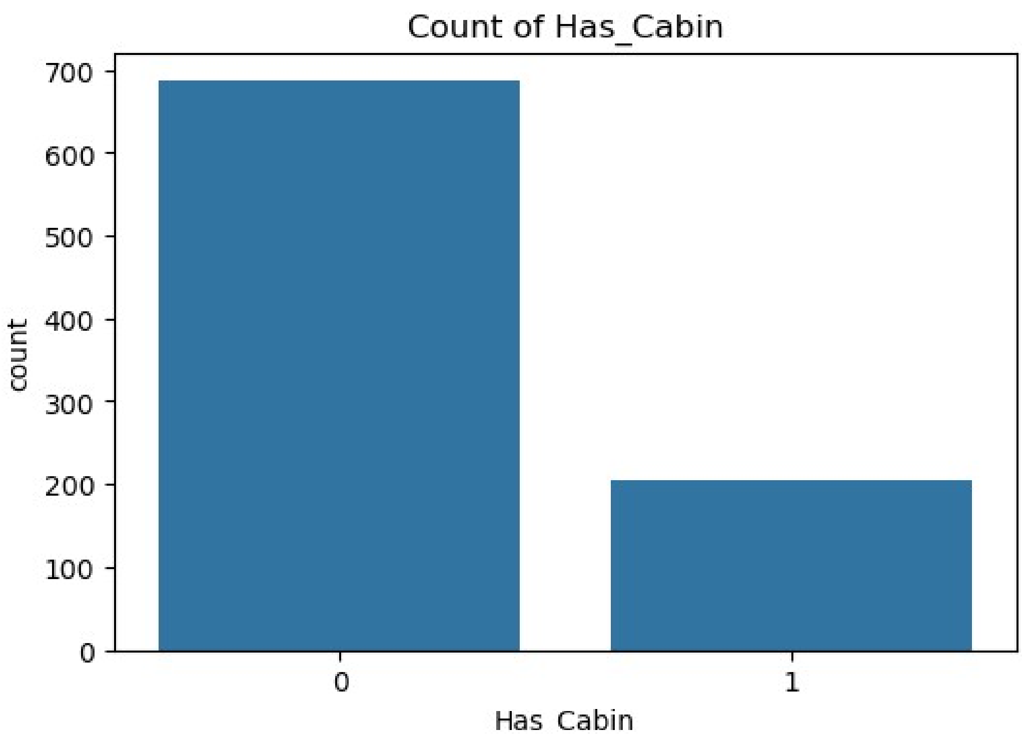
5.3 Categorical Columns – Count Plots

```
categorical_cols = ['Sex', 'Embarked', 'Pclass', 'Has_Cabin']

for col in categorical_cols:
    plt.figure(figsize=(6,4))
    sns.countplot(data=train_df, x=col)
    plt.title(f'Count of {col}')
    plt.show()
```




Count of Embarked
count
600
500
400
300
200
100
0
S
C
Q
Embarked


Count of Pclass
count
500
400
300
200
100
0
1
2
3
Pclass

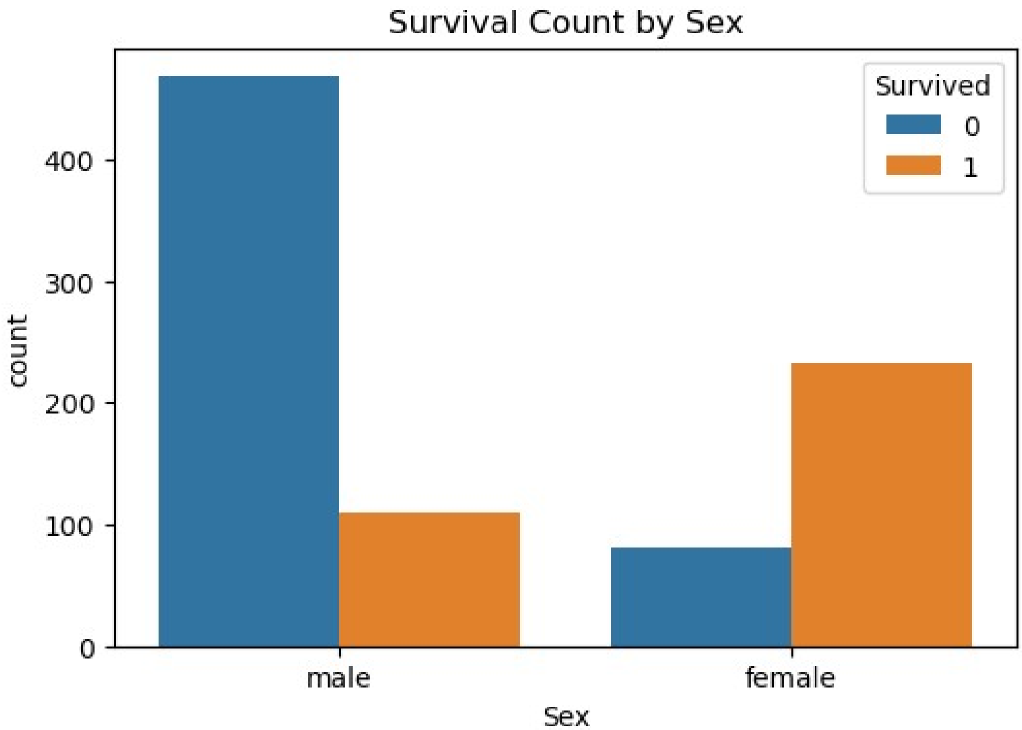


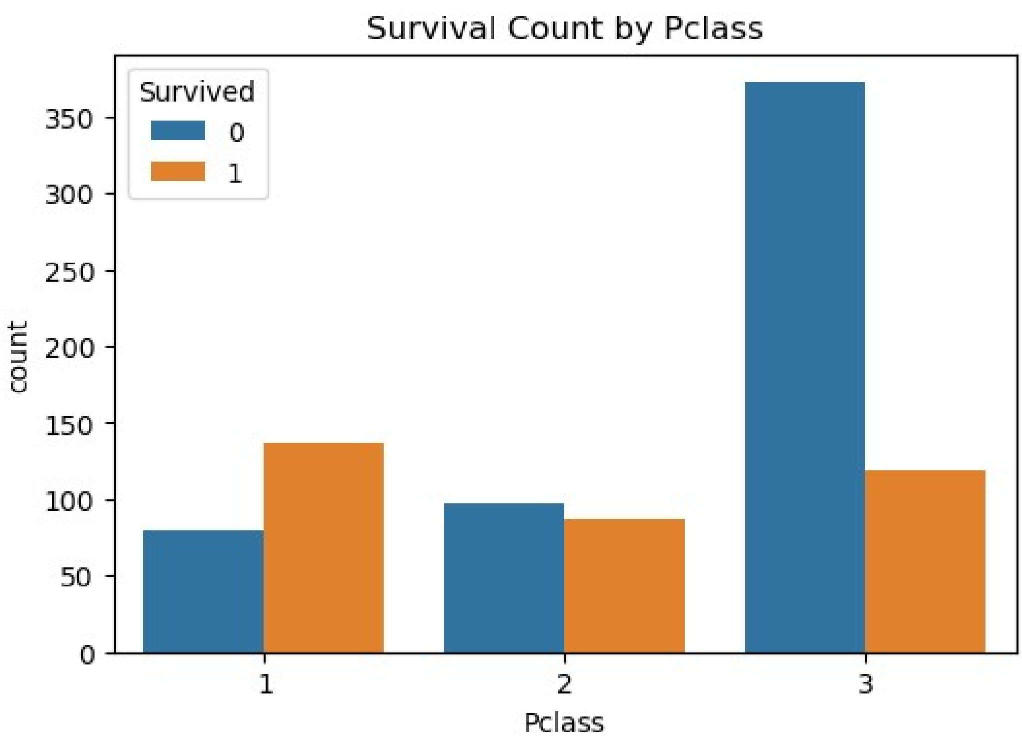
- **Sex**: More male passengers than female.
- **Embarked**: Southampton is the most common embarkation point, followed by Cherbourg and Queenstown.
- **Pclass**: Most passengers traveled in 3rd class, then 1st, then 2nd.
- **Has_Cabin**: Majority do not have cabin information.

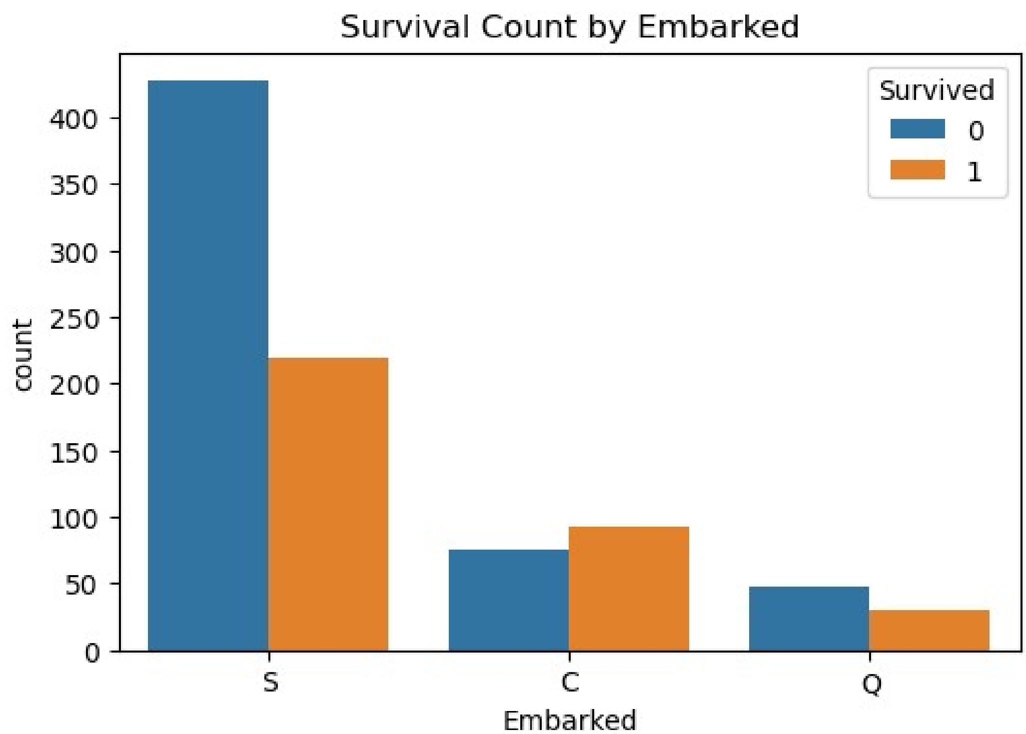
Step 6: Bivariate Analysis

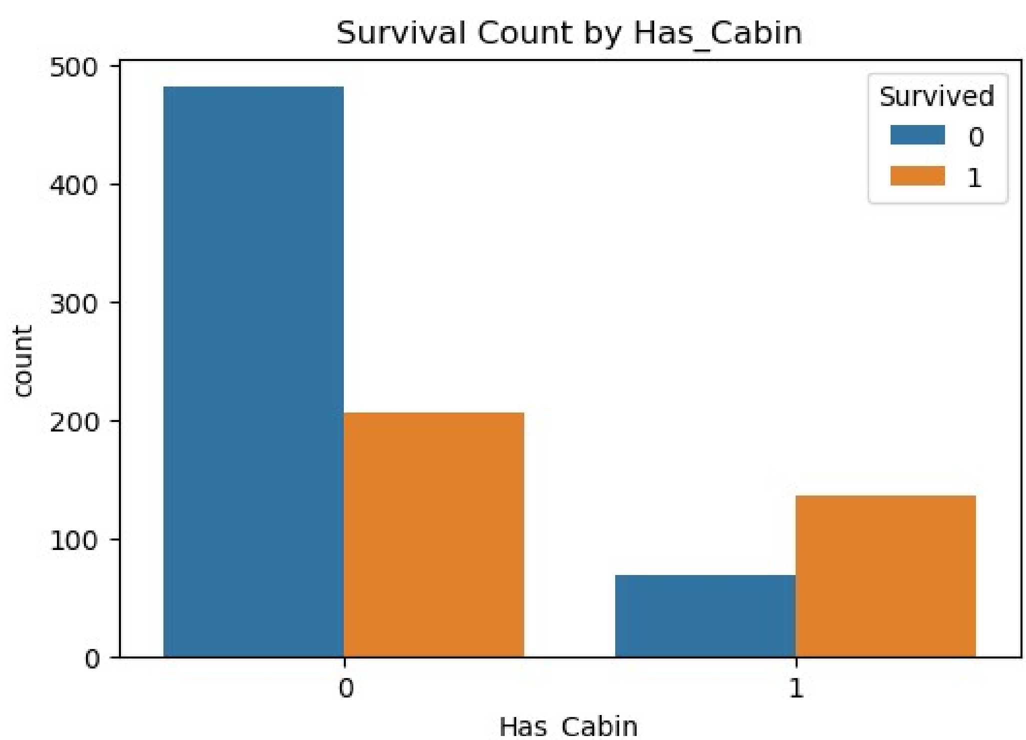
6.1 Survived vs Categorical Variables (Count Plots)

```
categorical_cols = ['Sex', 'Pclass', 'Embarked', 'Has_Cabin']

for col in categorical_cols:
    plt.figure(figsize=(6,4))
    sns.countplot(data=train_df, x=col, hue='Survived')
    plt.title(f'Survival Count by {col}')
    plt.show()
```


Survival Count by Sex
Survived
0
1
count
0
100
200
300
400
male
female
Sex


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



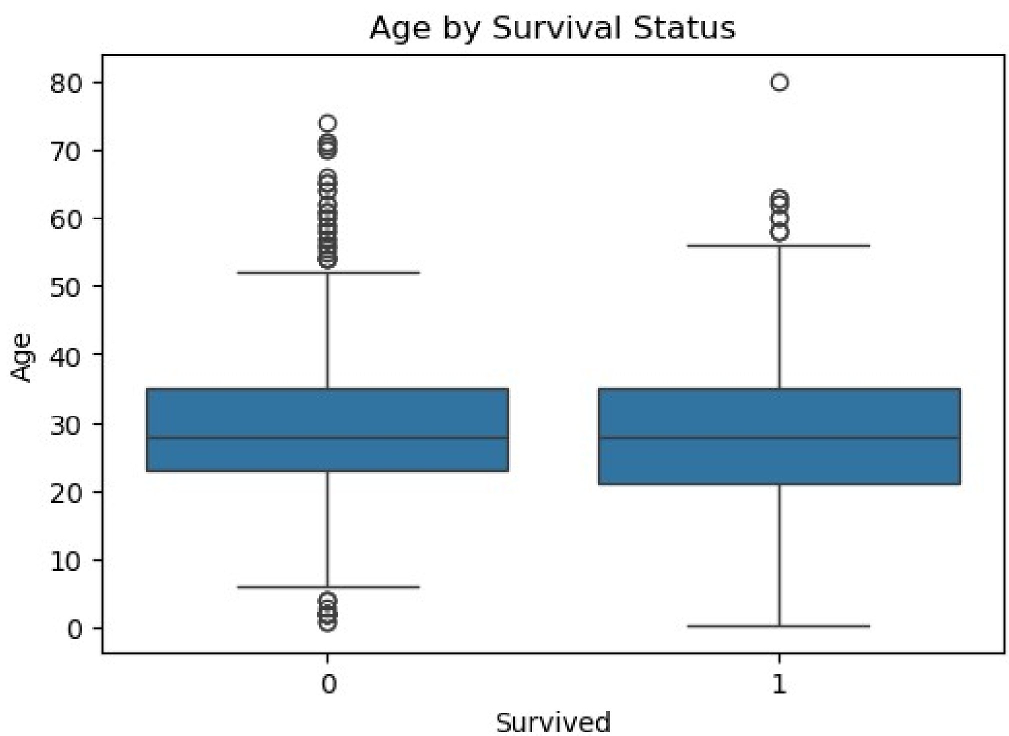


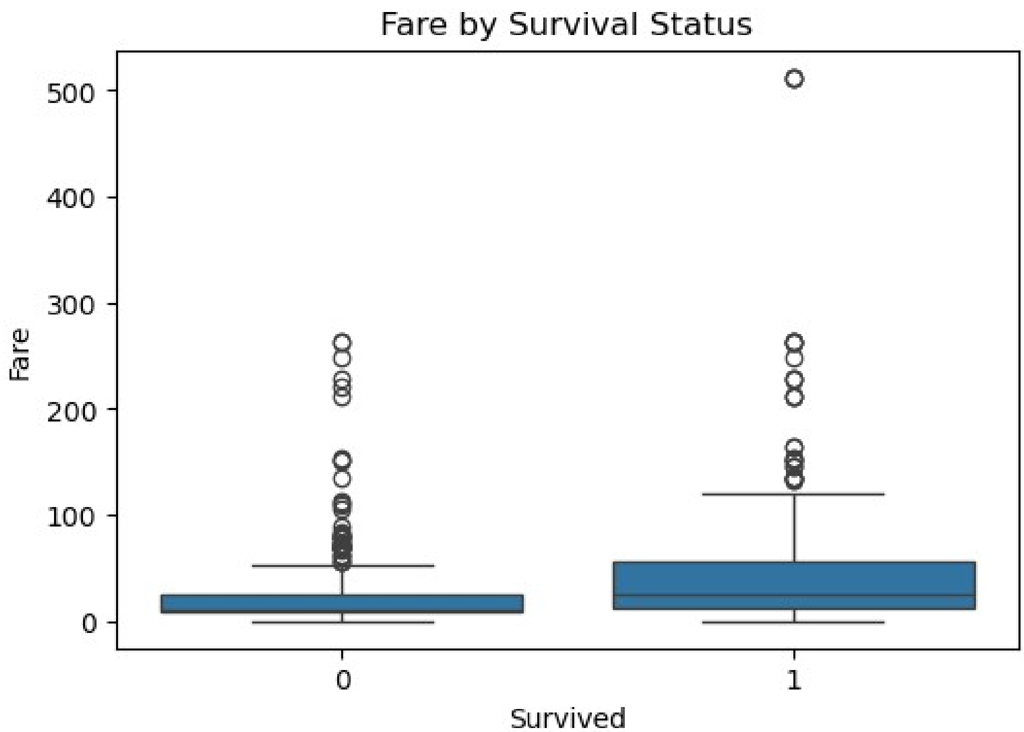
- **Sex**: Females have a higher survival rate than males.
- **Pclass**: 1st class passengers had the highest survival rate, 3rd class the lowest.

- **Embarked**: Passengers from Cherbourg show higher survival rates.
- **Has_Cabin**: Those with cabins had higher survival chances.

6.2 Survived vs Numerical Variables (Boxplots)

```
numeric_cols = ['Age', 'Fare']

for col in numeric_cols:
    plt.figure(figsize=(6,4))
    sns.boxplot(data=train_df, x='Survived', y=col)
    plt.title(f'{col} by Survival Status')
    plt.show()
```





- **Age**: Younger survivors are slightly more common, but survival occurs across all ages.
- **Fare**: Higher fares generally link to higher survival rates.

6.3 Correlation Heatmap (Numerical Features)

```
num_cols = train_df.select_dtypes(include=['int64', 'float64'])
print(num_cols.columns)
plt.figure(figsize=(8,6))
sns.heatmap(num_cols.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation between numeric features")
plt.show()
```

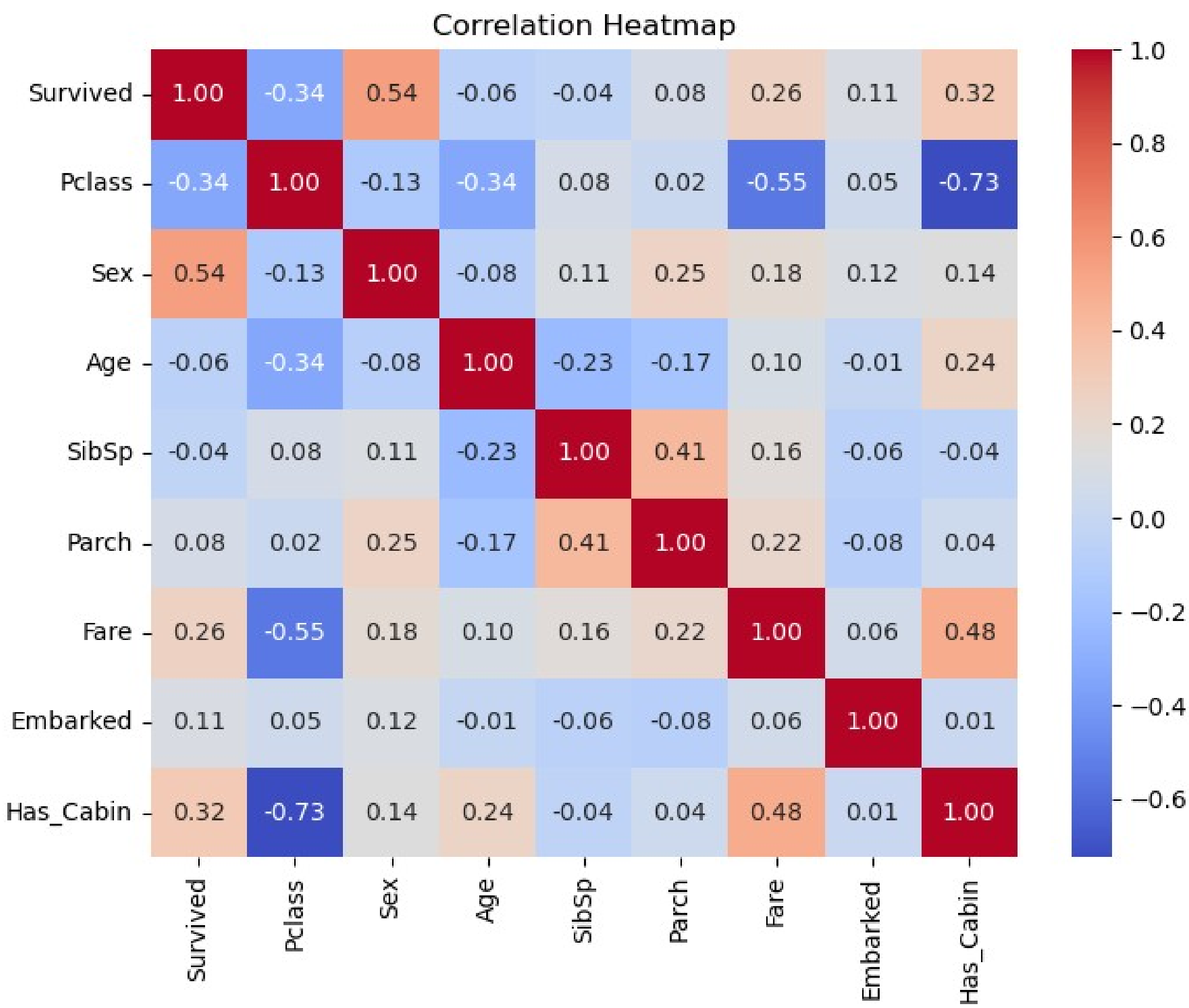
```
Index(['Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare',
'Has_Cabin'], dtype='object')
```

- Strong negative correlation between `Pclass` and `Fare` (higher class → higher fare).
- `Has_Cabin` is positively correlated with survival.

```
df_encoded = train_df.copy()
df_encoded['Sex'] = df_encoded['Sex'].map({'male': 0, 'female': 1})
df_encoded['Embarked'] = df_encoded['Embarked'].map({'S': 0, 'C': 1,
'Q': 2})
print(df_encoded.head())
```

```
   Survived  Pclass  Sex   Age  SibSp  Parch     Fare  Embarked
Has_Cabin
0         0       3    0  22.0      1      0   7.2500         0
0
1         1       1    1  38.0      1      0  71.2833         1
1
2         1       3    1  26.0      0      0   7.9250         0
0
3         1       1    1  35.0      1      0  53.1000         0
1
```

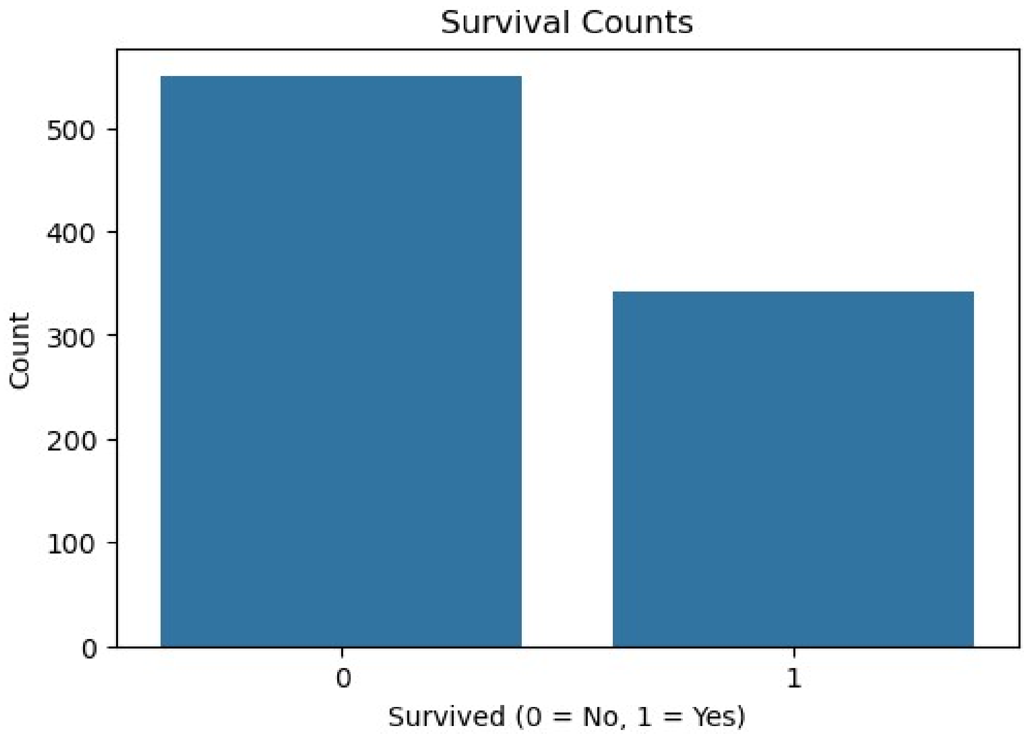
```
4           0         3     0   35.0        0       0    8.0500          0
0

# correlation heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(df_encoded.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```



Step 7: Encoding Categorical Variables

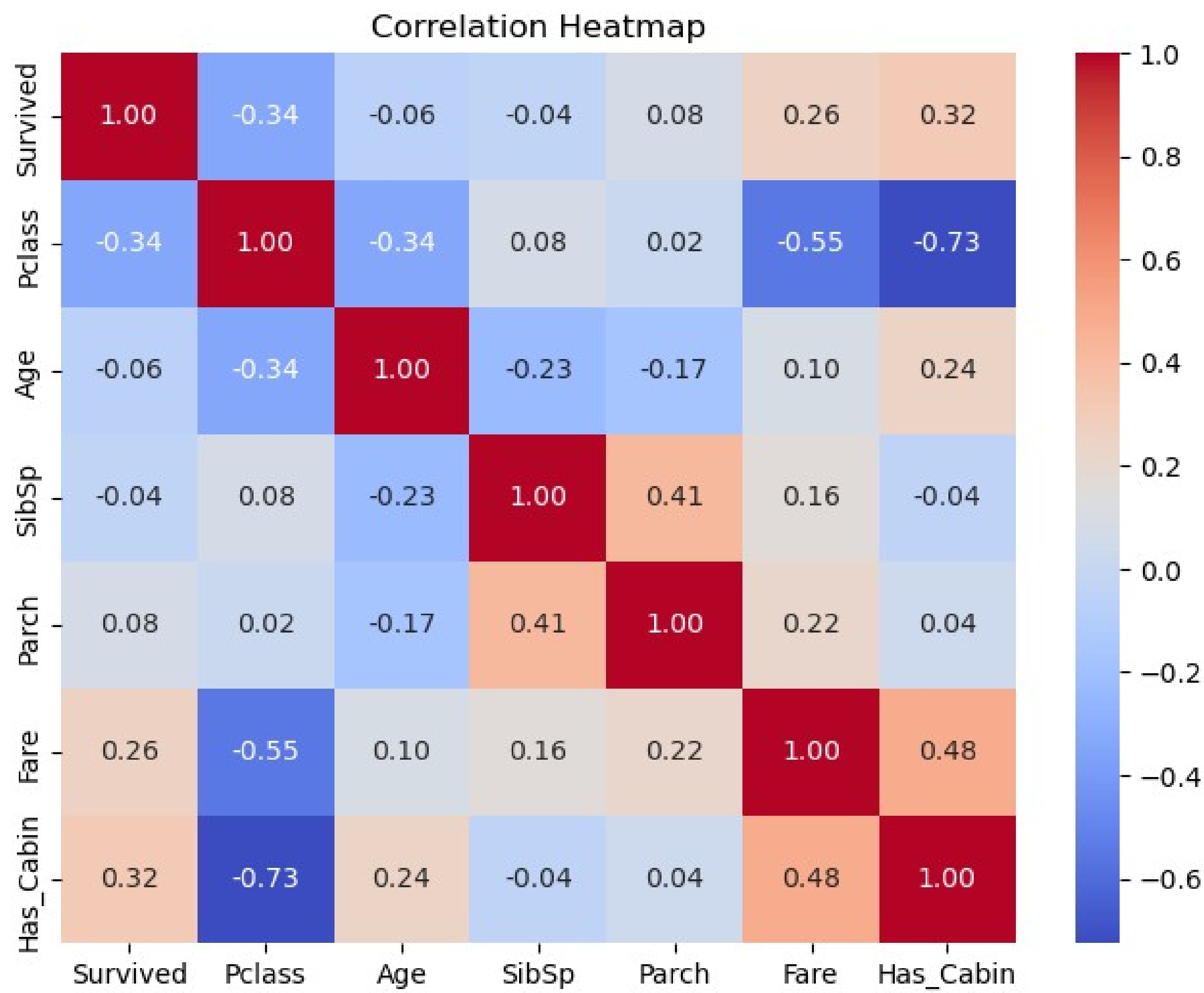
```
plt.figure(figsize=(6,4))
sns.countplot(data=df_encoded, x='Survived')
plt.title("Survival Counts")
plt.xlabel("Survived (0 = No, 1 = Yes)")
```

```
plt.ylabel("Count")
plt.show()
```



Step 8: Correlation Heatmap (after encoding)

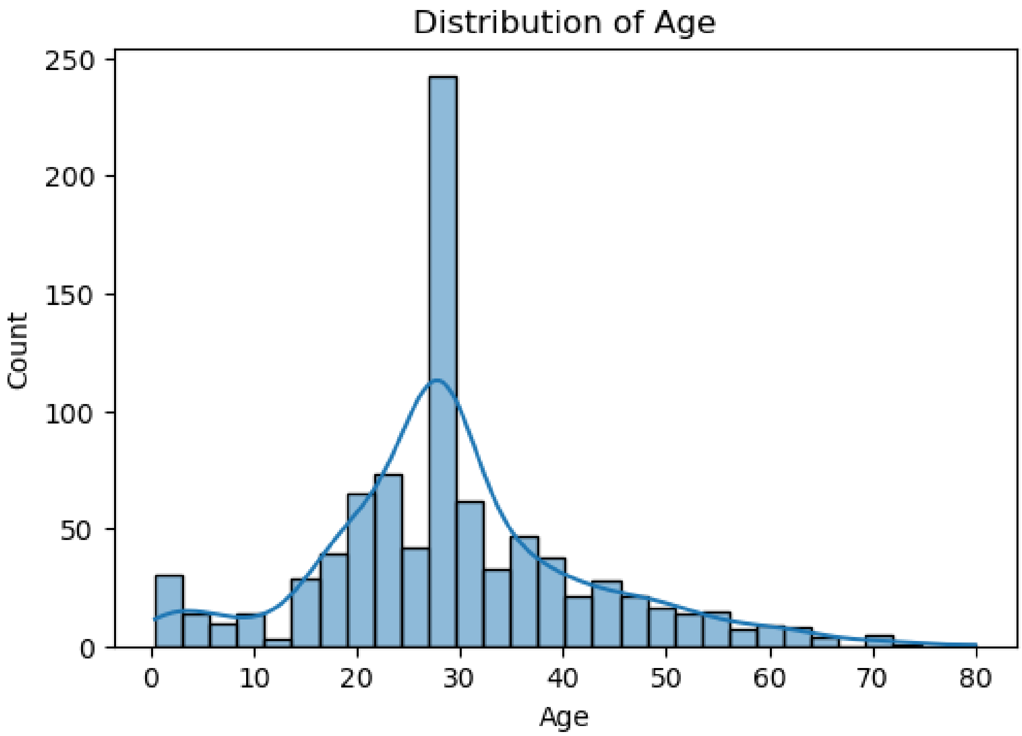
```
numeric_df = train_df.select_dtypes(include=['number'])
plt.figure(figsize=(8,6))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```

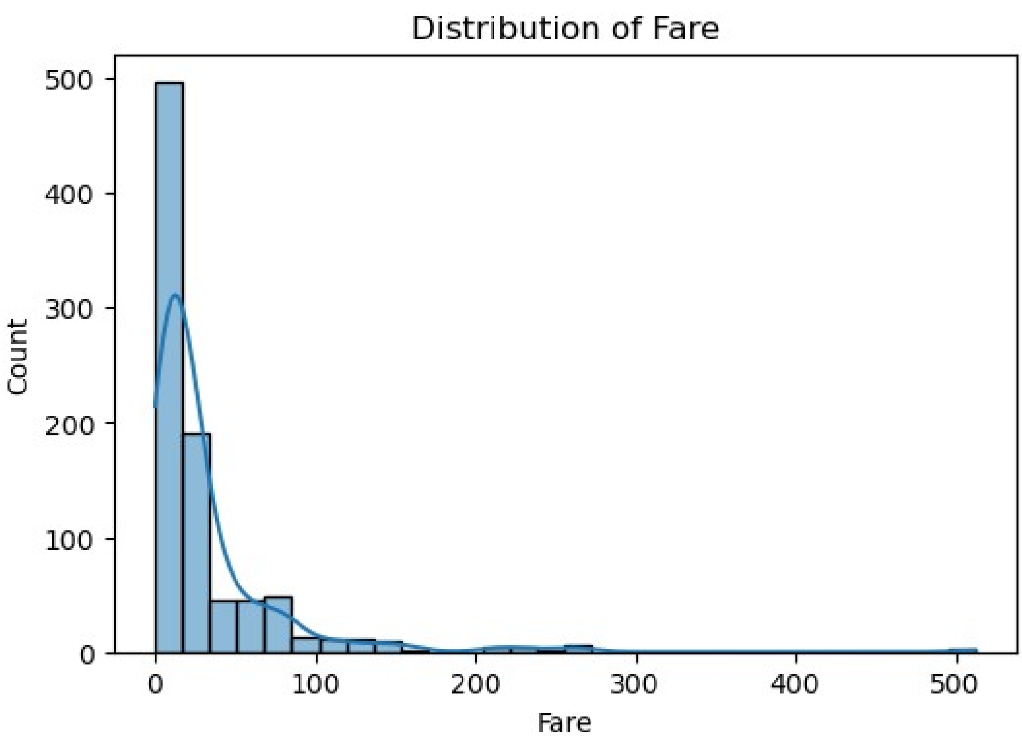


- `Sex` (encoded) shows negative correlation with survival (0 = male, 1 = female → females survive more).
- `Embarked` shows weak correlation with survival.
- `Fare` shows positive correlation with survival.

Step 9: Univariate Visualizations

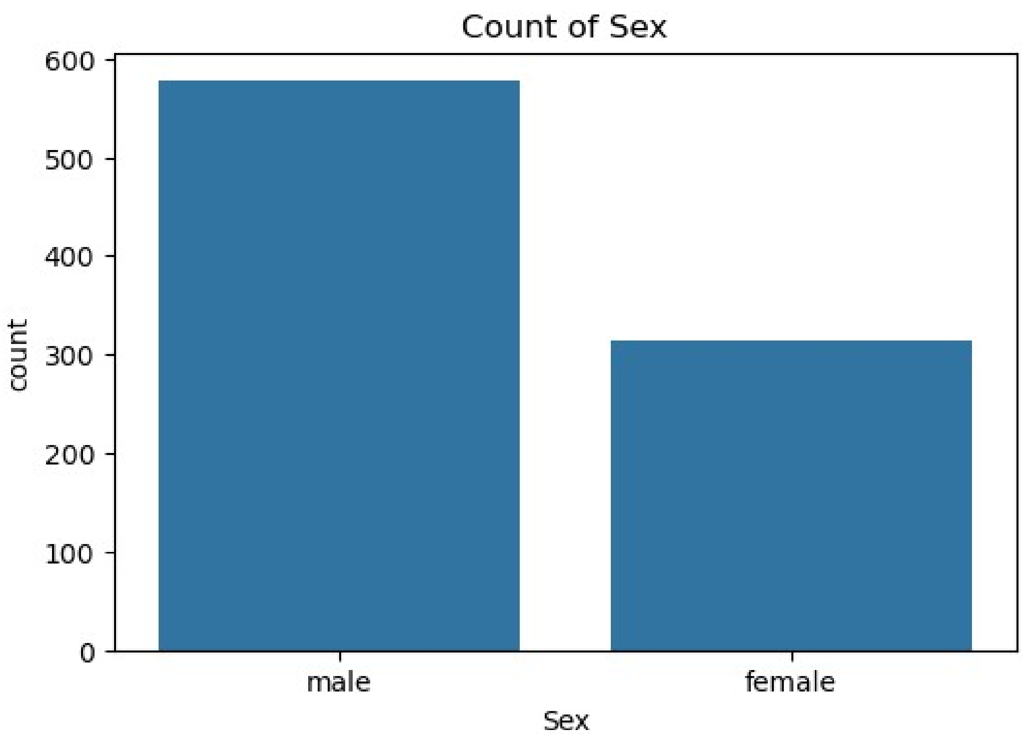
```
# Numerical features
num_cols = ['Age', 'Fare']
for col in num_cols:
    plt.figure(figsize=(6,4))
    sns.histplot(train_df[col], kde=True, bins=30)
    plt.title(f"Distribution of {col}")
    plt.show()
```

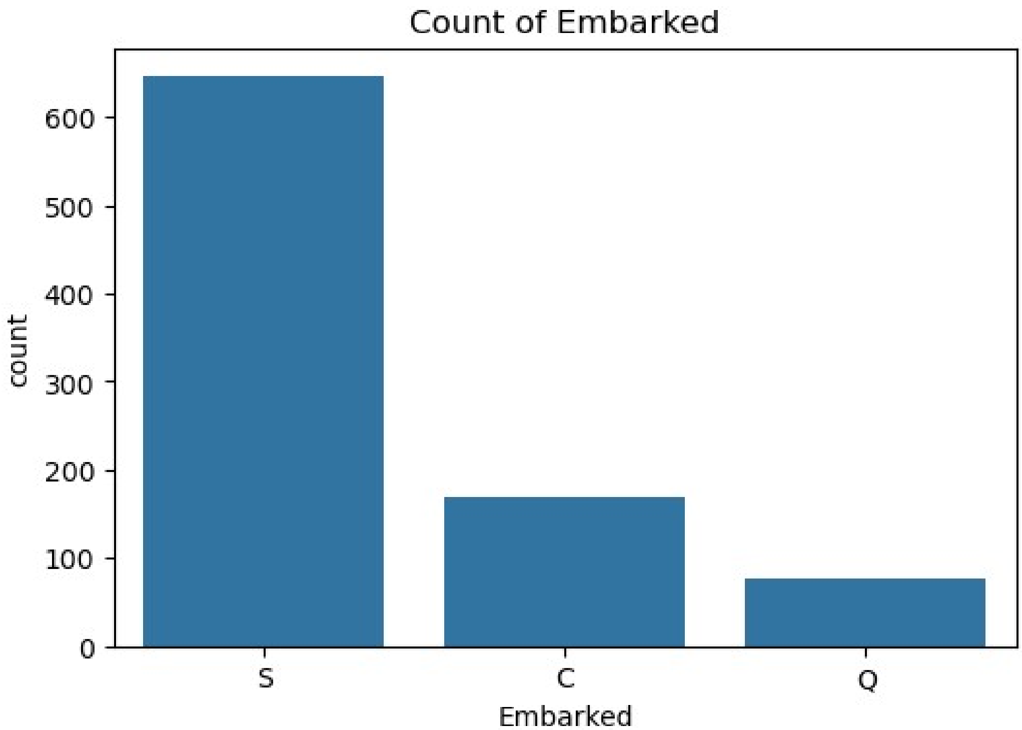


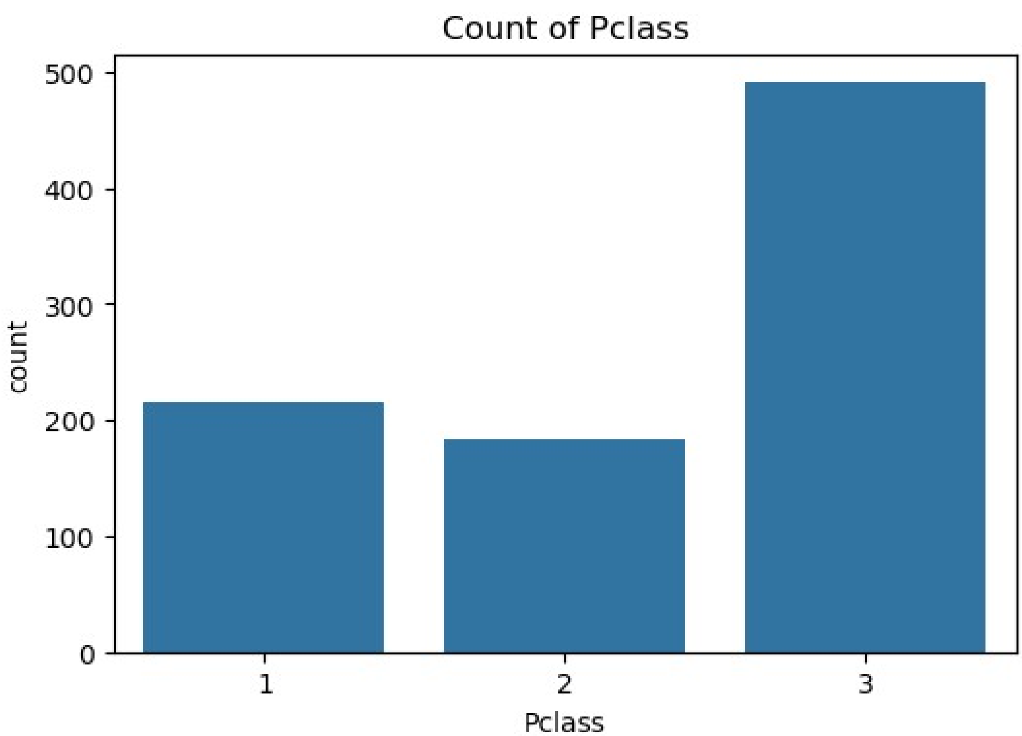



```
# Categorical features
cat_cols = ['Sex', 'Embarked', 'Pclass']
for col in cat_cols:
```

```
    plt.figure(figsize=(6,4))
    sns.countplot(x=col, data=train_df)
    plt.title(f"Count of {col}")
    plt.show()
```

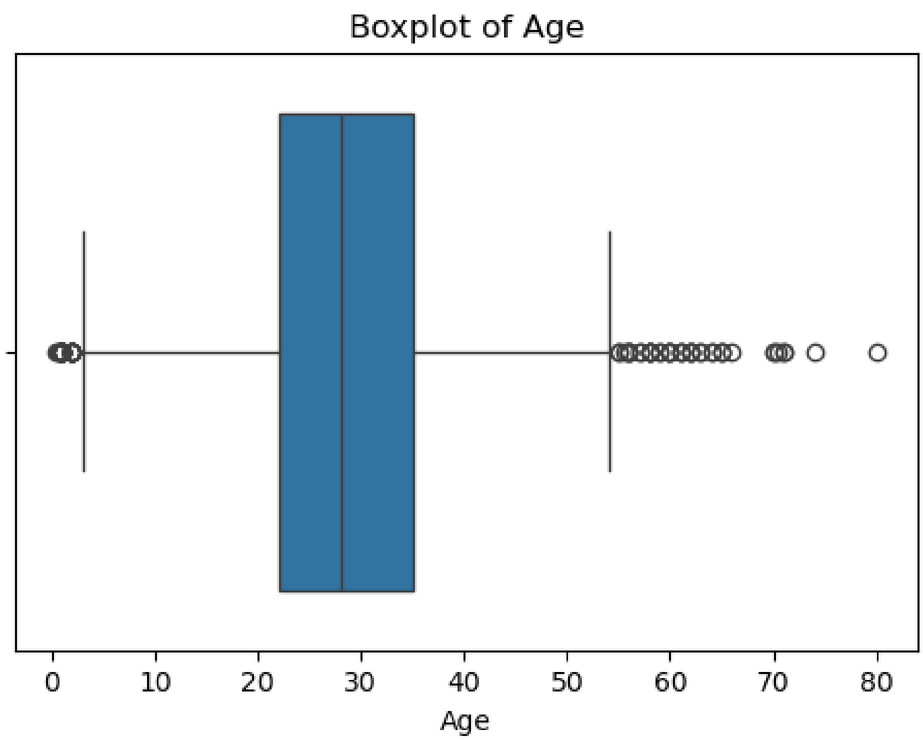


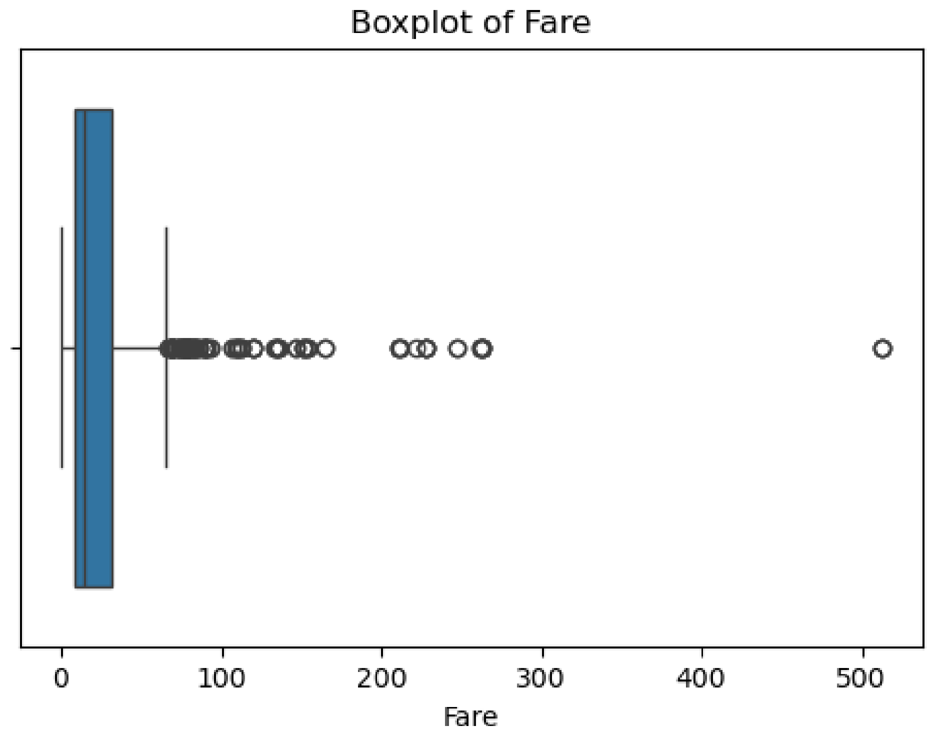




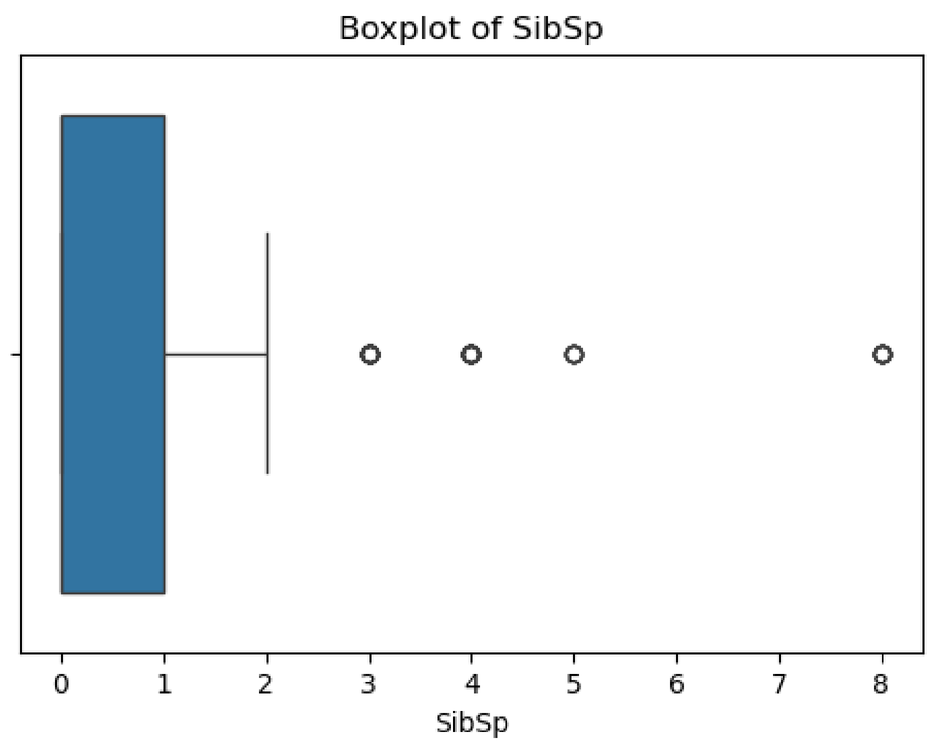

```
# Boxplots for Numeric Columns
numeric_cols = ['Age', 'Fare', 'SibSp', 'Parch']

for col in numeric_cols:
```

```python
plt.figure(figsize=(6,4))
sns.boxplot(x=train_df[col])
plt.title(f'Boxplot of {col}')
plt.show()
```





Boxplot of Fare
0
100
200
300
400
500
Fare

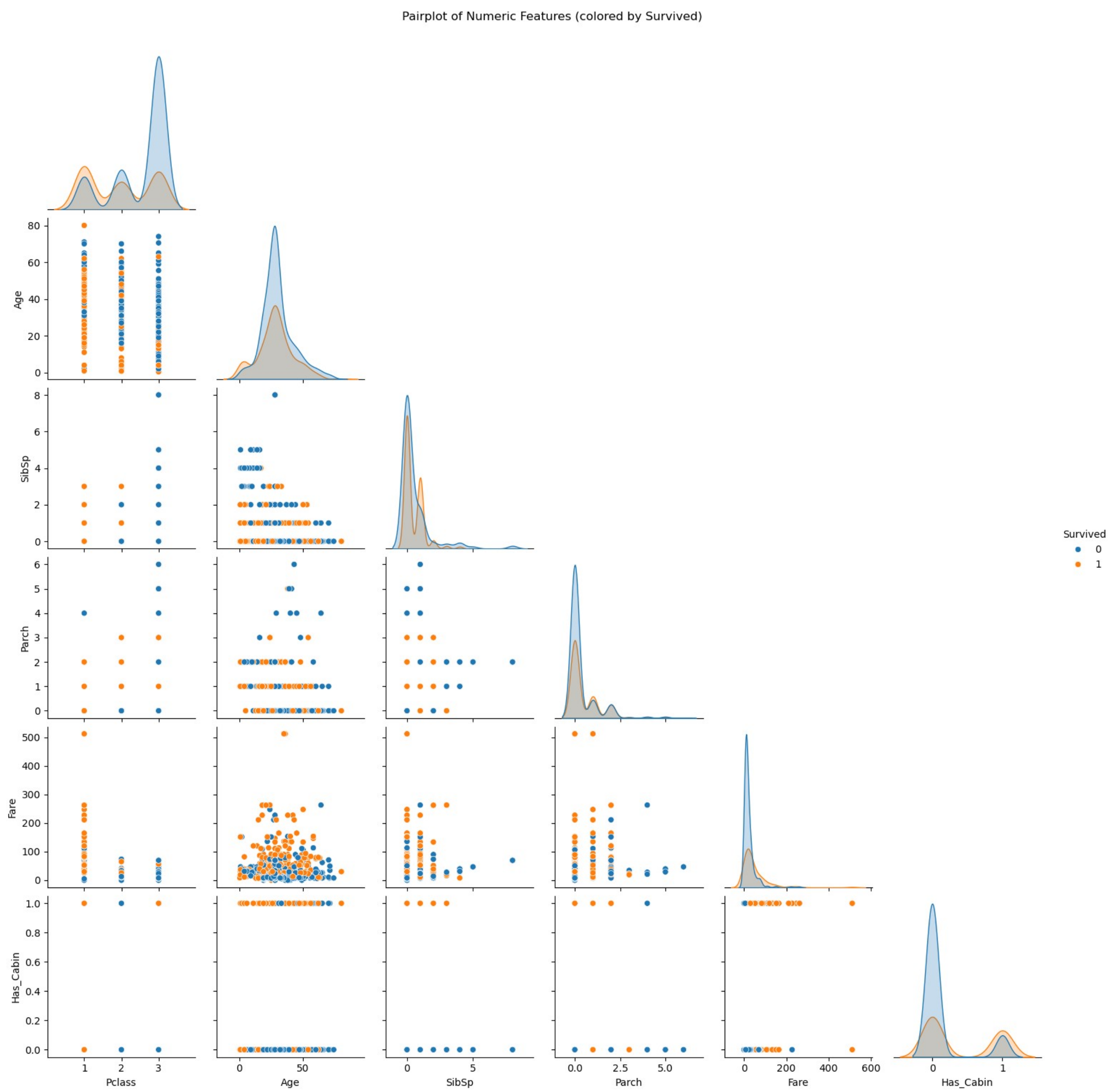


Boxplot of SibSp
0
1
2
3
4
5
6
7
8
SibSp

- Matches earlier Step 5 trends:
  - Age distribution still peaks in 20–40 range.
  - Fare distribution remains skewed.
  - Class imbalance in categorical features is visible.

Step 10: Pairplot for Relationships and Trends

```
numeric_cols = ['Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare', 
'Has_Cabin']
sns.pairplot(train_df[numeric_cols], hue='Survived', diag_kind='kde', 
corner=True)
plt.suptitle('Pairplot of Numeric Features (colored by Survived)', 
y=1.02)
plt.show()
```



- Survivors are more frequent in higher-fare, lower-class (Pclass=1) areas.
- Clusters visible where Fare is high and Pclass is 1.
- `Has_Cabin` and `Pclass` separate survivors and non-survivors slightly.

Step 12: Summary of Findings

- **Survival Rate**: Around 38% of passengers survived. Females, 1st class passengers, and those with cabins had noticeably higher survival rates.
- **Passenger Class & Fare**: Higher fares are strongly linked to higher survival, and 1st class passengers paid significantly more on average.

- **Age Distribution**: Most passengers were between 20–40 years old. Children had slightly better survival chances in certain classes.
- **Family Size**: Majority of passengers traveled alone. Those with small families had better chances of survival compared to those traveling alone or with large families.
- **Embarkation Point**: Most passengers boarded at Southampton, but those from Cherbourg had the highest survival rates.
- **Cabin Information**: Presence of cabin data (`Has_Cabin`) is strongly associated with higher survival, indicating a link with passenger class and ticket price.