1. Upload the Dataset

```
from google.colab import files
uploaded = files.upload()
```

Choose Files train.csv

- **train.csv**(text/csv) - 61194 bytes, last modified: 4/28/2025 - 100% done

```
Saving train.csv to train (3).csv
```

2. Import Required Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid")
plt.style.use("seaborn-v0_8-poster")
```

3. Load the Dataset

```
df = pd.read_csv('train.csv')
```

4. Basic Data Exploration

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
1	2	1	1	Cumings, Mrs. John Bradley (Florence	female	38.0	1	0	PC 17599	71.2833

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
print(f"Dataset contains {df.shape[0]} rows and {df.shape[1]} columns")
```

```
Dataset contains 891 rows and 12 columns
```

```
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
```

```
df.isnull().sum()
```

	0
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 Cleaning (Handling Missing Values)

```
# 1. Filling missing Age values with median
df['Age'].fillna(df['Age'].median(), inplace=True)

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

# 3. Droping Cabin column (Because too many missing values)
df.drop('Cabin', axis=1, inplace=True)

# 4. Rechecking if any missing values left
df.isnull().sum()
```

```
<ipython-input-42-63901d726beb>:2: FutureWarning: A value is trying to be set on a copy
The behavior will change in pandas 3.0. This inplace method will never work because the

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

  df['Age'].fillna(df['Age'].median(), inplace=True)
<ipython-input-42-63901d726beb>:5: FutureWarning: A value is trying to be set on a copy
The behavior will change in pandas 3.0. This inplace method will never work because the

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

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

	0
PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	0
SibSp	0
Parch	0
Ticket	0
Fare	0
Embarked	0

dtype: int64

6. Value Counts for Categorical Features

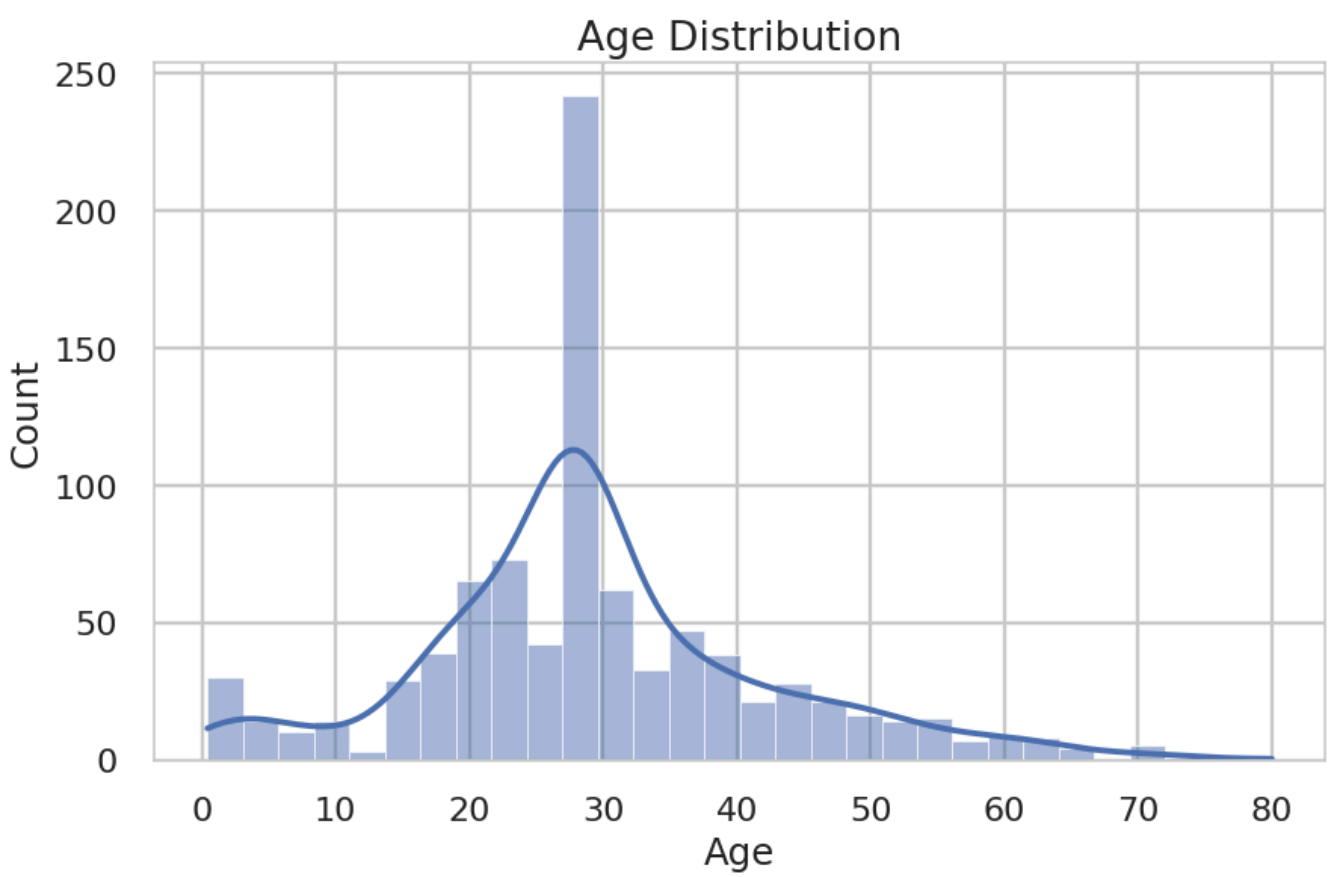
```
categorical_cols = ['Sex', 'Embarked', 'Pclass', 'Survived']
for col in categorical_cols:
    print(f"\nValue counts for {col}:")
    print(df[col].value_counts())
```

```
Value counts for Sex:
Sex
male      577
female    314
```

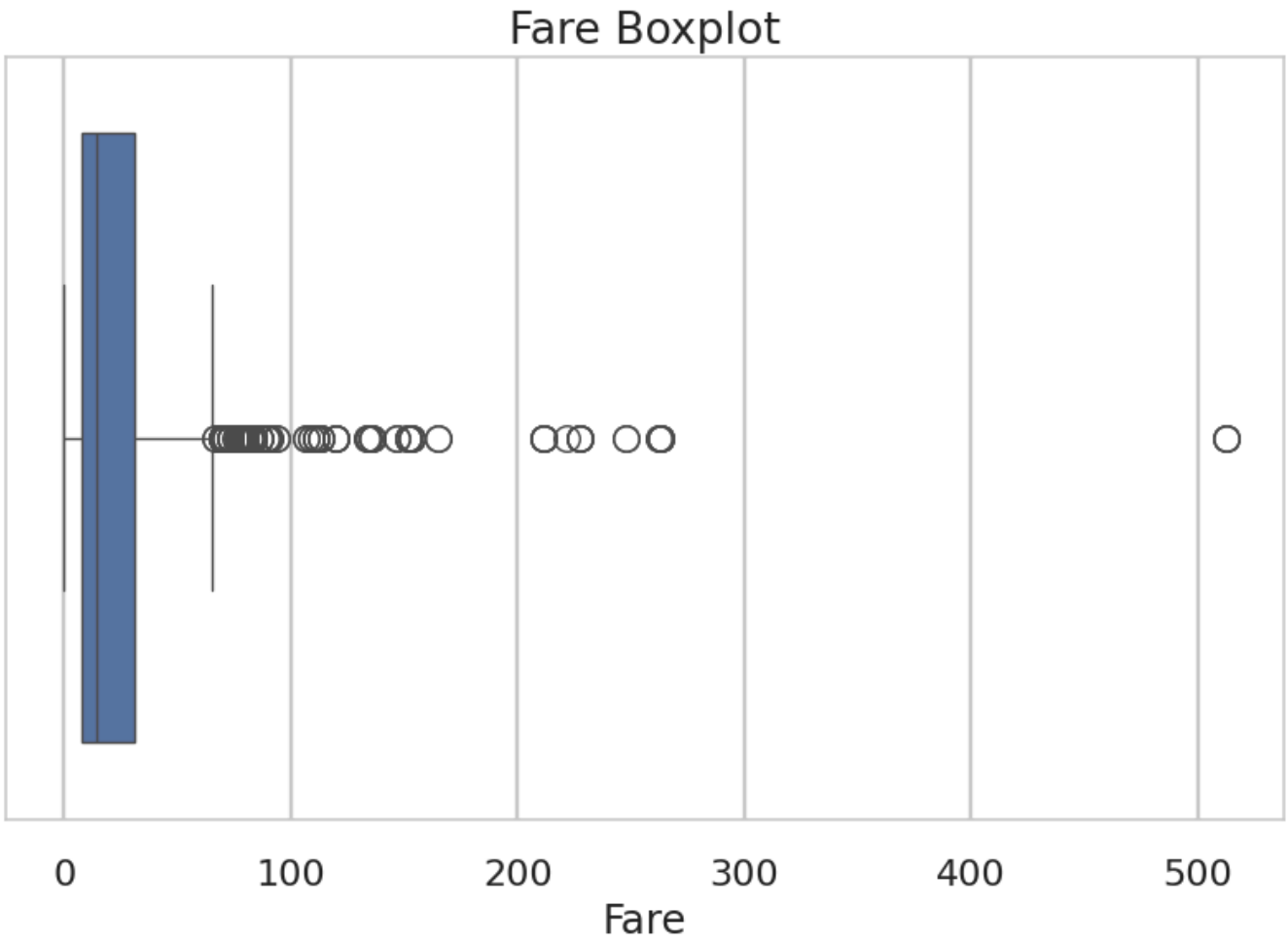
```
Name: count, dtype: int64

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

Value counts for Survived:
Survived
0    549
1    342
Name: count, dtype: int64
```

7. Univariate Analysis (One Variable at a Time)

```
# Distribution of Age
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=30, kde=True)
plt.title('Age Distribution')
plt.show()
```

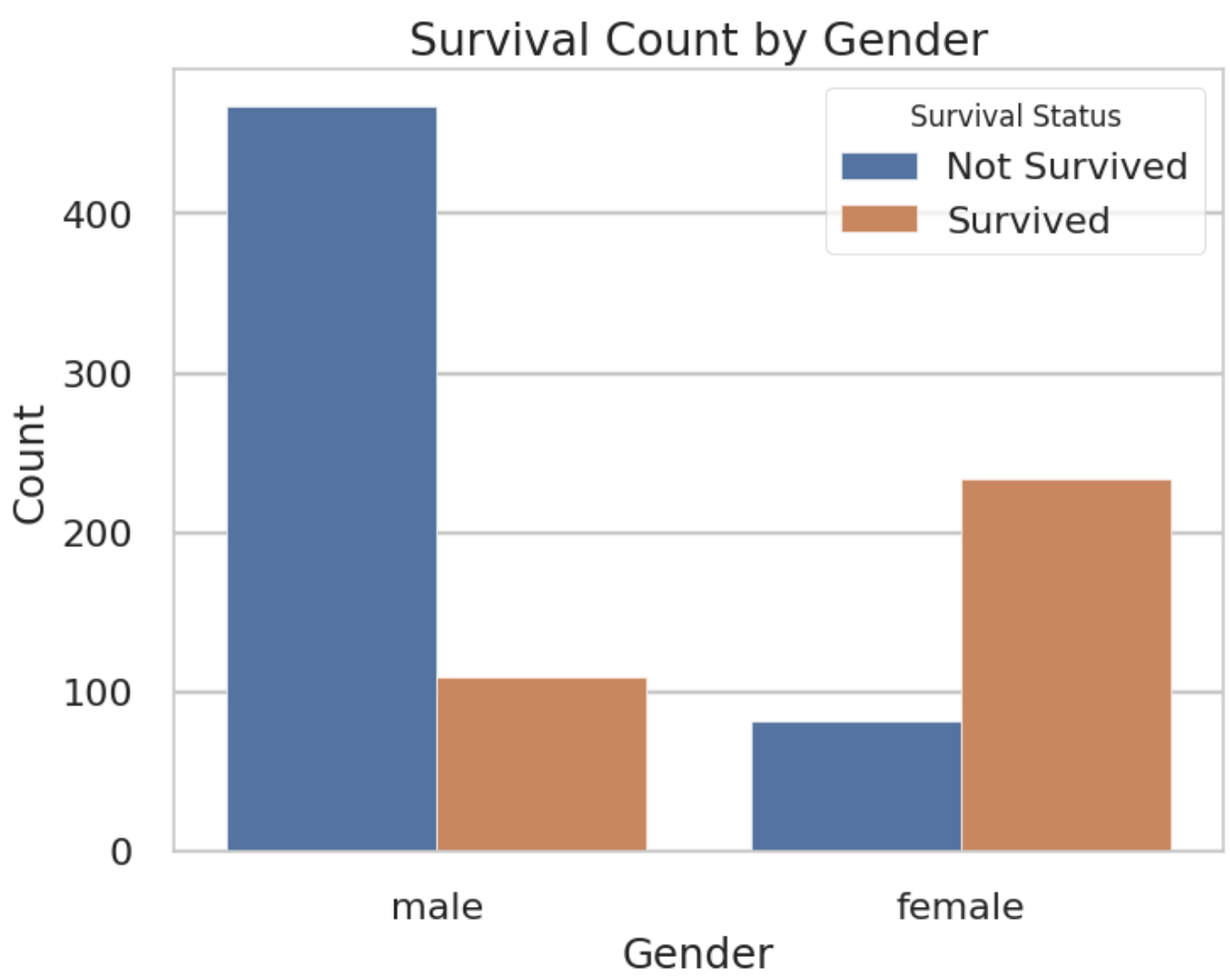



```
plt.figure(figsize=(10,6))
sns.boxplot(x=df['Fare'])
plt.title('Fare Boxplot')
plt.show()
```

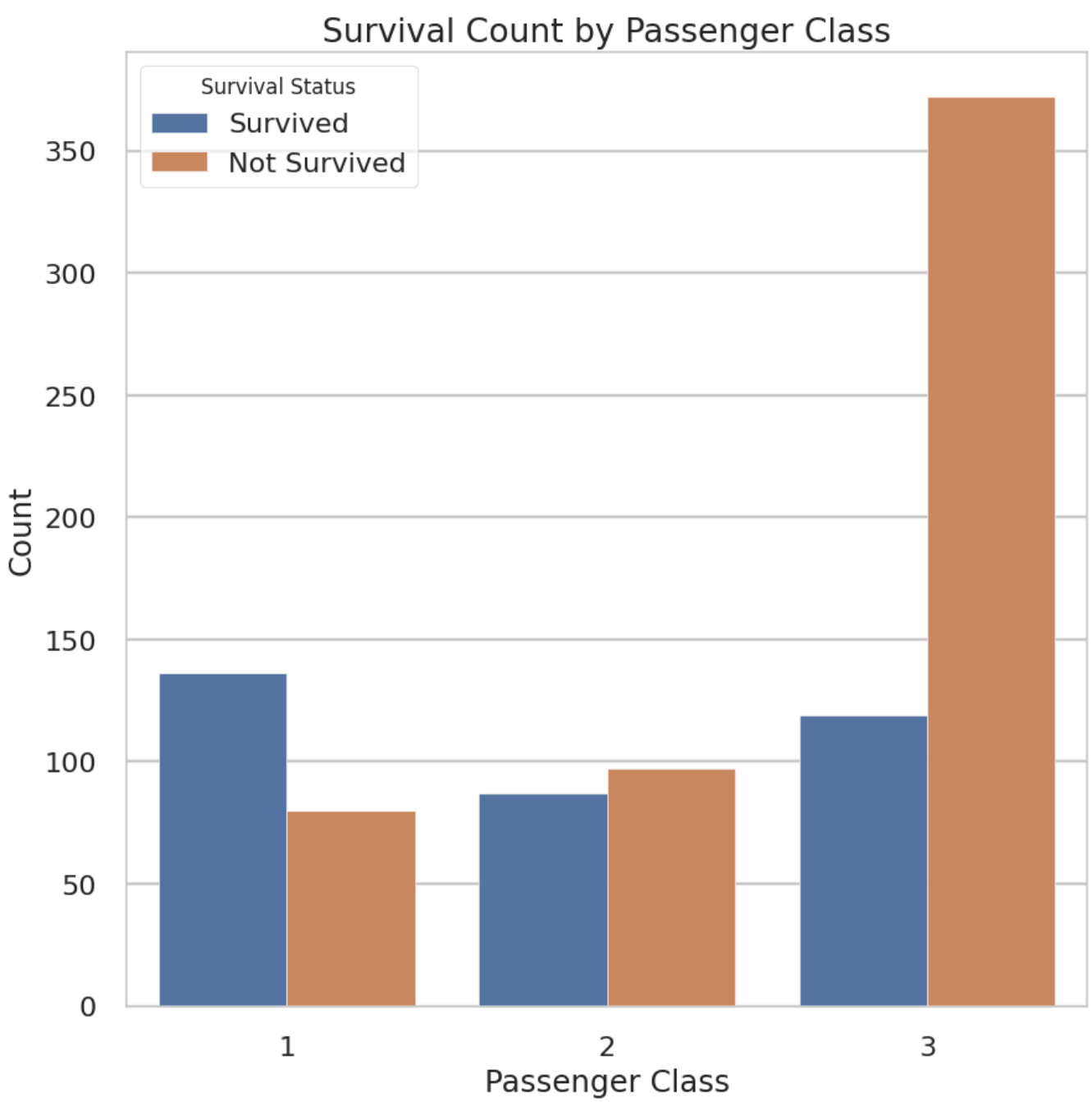


8. Bivariate Analysis (Relationship between Two Variables)

```
# 1. Map Survived values to readable easily
df['Survival_Status'] = df['Survived'].map({0: 'Not Survived', 1: 'Survived'})
# 2. Plot
plt.figure(figsize=(8,6))
sns.countplot(x='Sex', hue='Survival_Status', data=df)
plt.title('Survival Count by Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.legend(title='Survival Status')
plt.show()
```

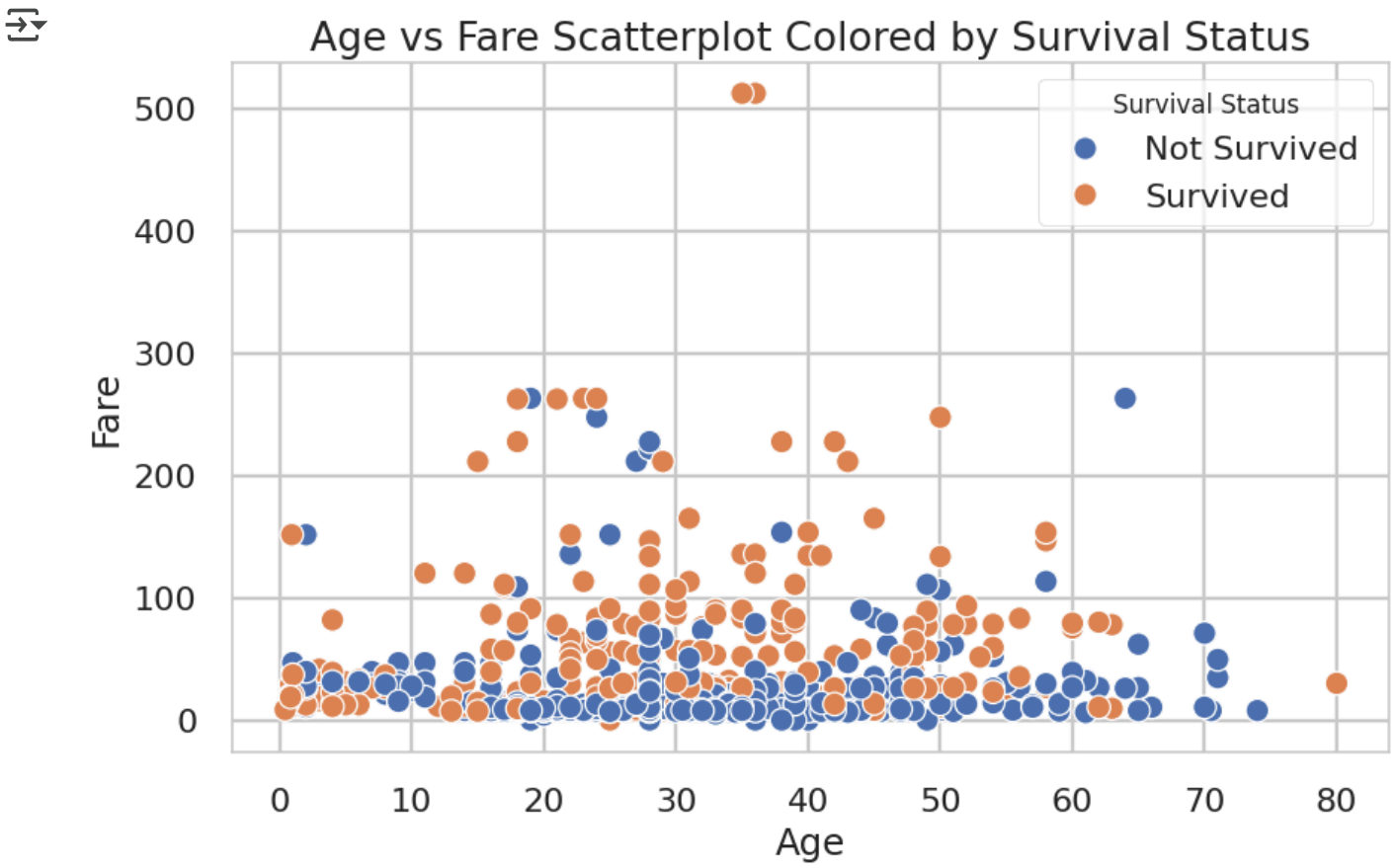



```
df['Survival_Status'] = df['Survived'].map({0: 'Not Survived', 1: 'Survived'})
plt.figure(figsize=(10,10))
sns.countplot(x='Pclass', hue='Survival_Status', data=df)
plt.title('Survival Count by Passenger Class')
plt.xlabel('Passenger Class')
plt.ylabel('Count')
plt.legend(title='Survival Status')
plt.show()
```



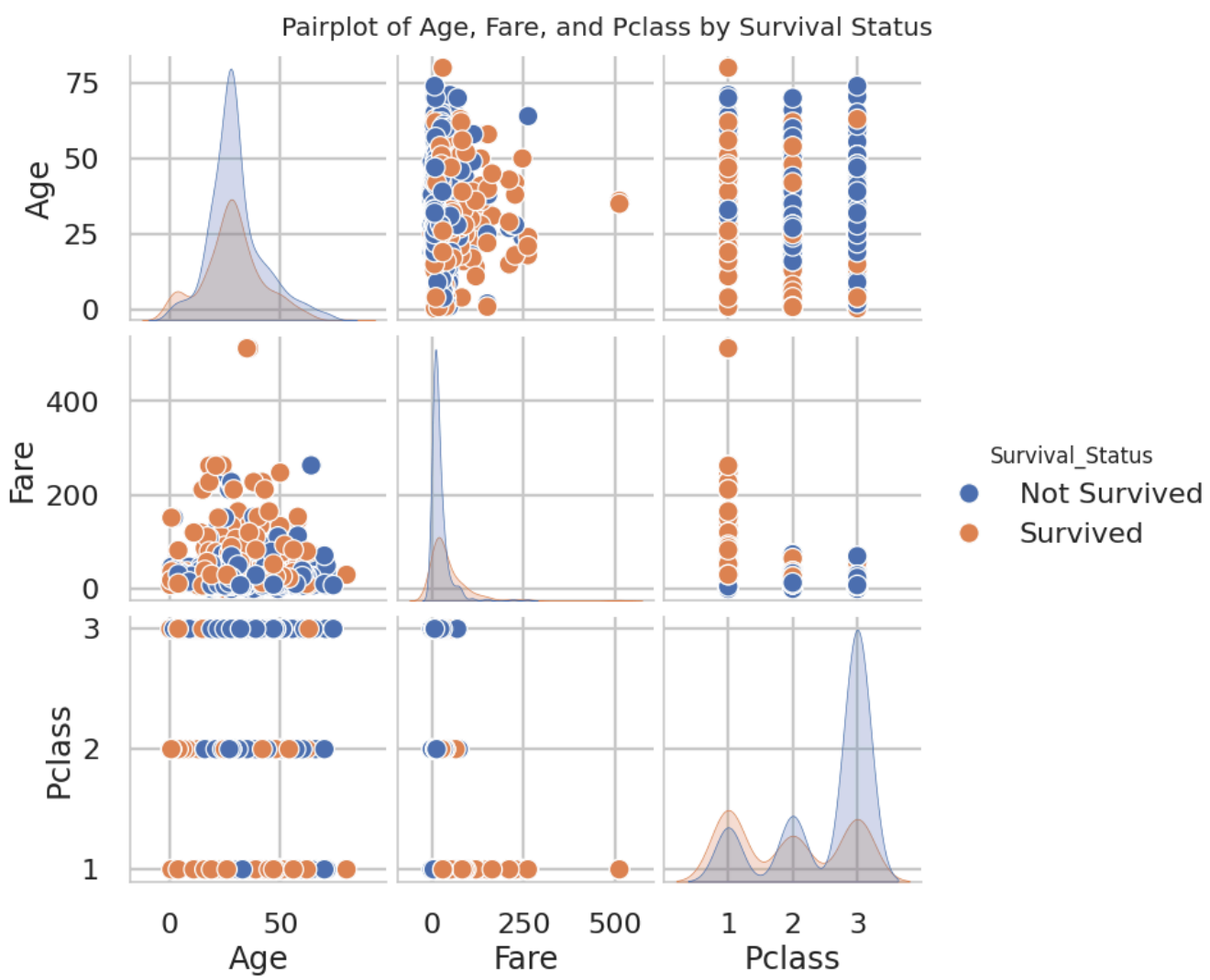

```
# Scatter plot with Survival_Status
plt.figure(figsize=(10,6))
```

```
sns.scatterplot(x='Age', y='Fare', hue='Survival_Status', data=df)
plt.title('Age vs Fare Scatterplot Colored by Survival Status')
plt.xlabel('Age')
plt.ylabel('Fare')
plt.legend(title='Survival Status')
plt.show()
```



9. Multivariate Analysis (Multiple Variables Together)

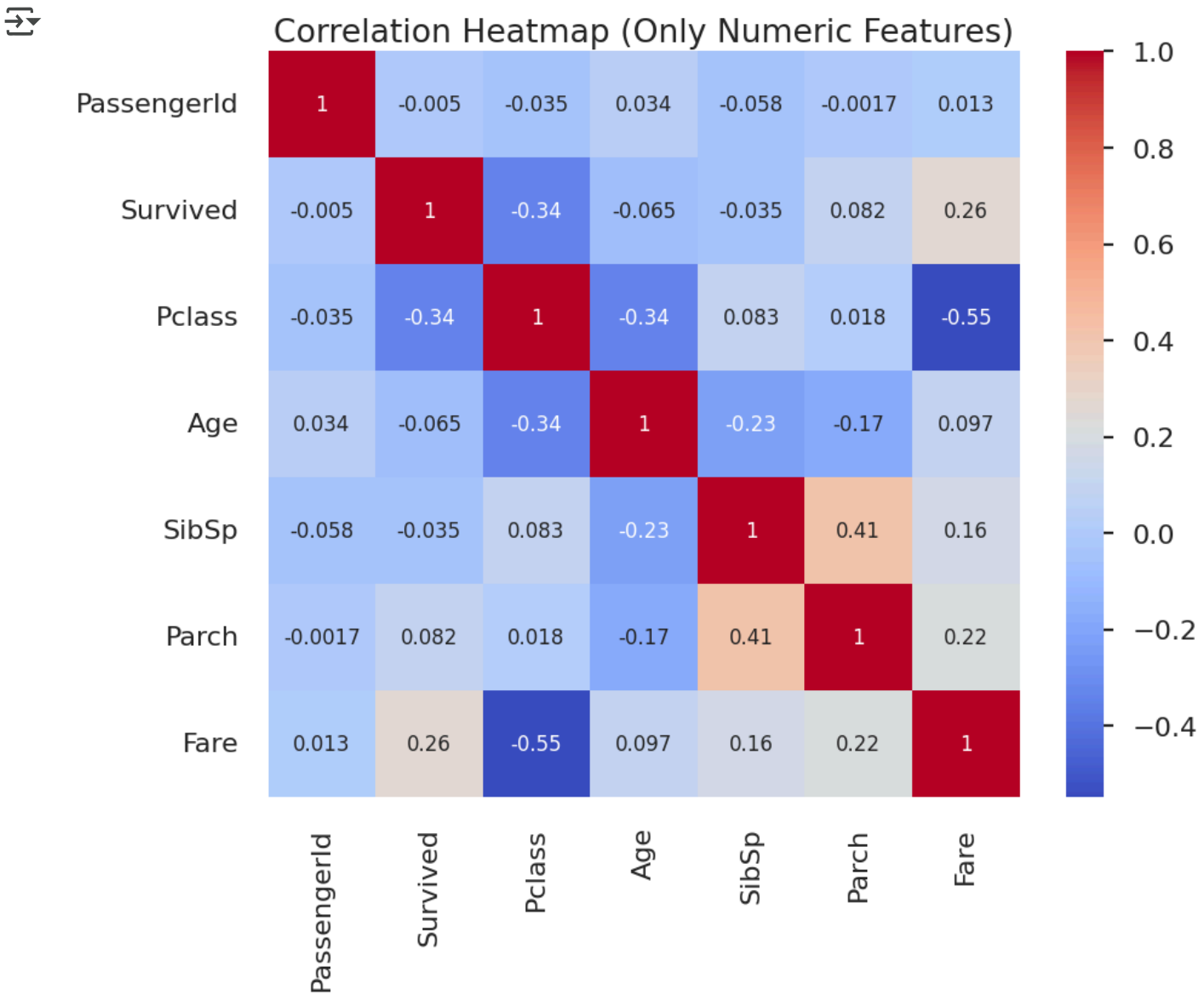
```
sns.pairplot(df[['Survival_Status', 'Age', 'Fare', 'Pclass']], hue='Survival_Status')
plt.suptitle('Pairplot of Age, Fare, and Pclass by Survival Status', y=1.02)
plt.show()
```



10. Correlation Matrix and Heatmap

```
# Correlation Matrix
plt.figure(figsize=(10,8))
numeric_df = df.select_dtypes(include=['int64', 'float64'])
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm')
```

```
plt.title('Correlation Heatmap (Only Numeric Features)')
plt.show()
```



11. Detect Skewness

```
print(df['Fare'].skew(), df['Age'].skew())
```

```
4.787316519674893 0.5102446555756495
```

```
df['Fare_log'] = np.log1p(df['Fare'])  # log(1 + Fare)
print(df['Fare_log'].skew())
```

```
0.3949280095189306
```

```
plt.figure(figsize=(12, 6))
sns.histplot(df['Fare'], kde=True, color='blue', label='Fare')
sns.histplot(df['Fare_log'], kde=True, color='red', label='Fare_log')
plt.legend()
plt.title('Distribution of Fare vs. Fare_log')
plt.show()
```