Import Libraries

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Load Dataset

```
# Load the Titanic dataset from seaborn
df = sns.load_dataset('titanic')
```

Data Cleaning

```
# Display the first few rows of the dataset
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 891,\n  \"fields\": [\
n    {\n      \"column\": \"survived\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0,\n        \"min\": 0,\n
\"max\": 1,\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          1,\n          0\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"pclass\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0,\n        \"min\": 1,\n
\"max\": 3,\n        \"num_unique_values\": 3,\n        \"samples\":
[\n          3,\n          1\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"sex\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"female\",\n          \"male\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 14.526497332334042,\n
\"min\": 0.42,\n        \"max\": 80.0,\n        \"num_unique_values\":
88,\n        \"samples\": [\n          0.75,\n          22.0\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"sibsp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 1,\n        \"min\": 0,\n        \"max\": 8,\n
\"num_unique_values\": 7,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

\"parch\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 0,\n   \"min\": 0,\n   \"max\": 6,\n
\"num_unique_values\": 7,\n   \"samples\": [\n   0,\n
1\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"fare\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 49.6934285971809,\n   \"min\": 0.0,\n   \"max\":
512.3292,\n   \"num_unique_values\": 248,\n   \"samples\":
[\n   11.2417,\n   51.8625\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"embarked\",\n   \"properties\":
{\n   \"dtype\": \"category\",\n   \"num_unique_values\":
3,\n   \"samples\": [\n   \"S\",\n   \"C\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"class\",\n   \"properties\":
{\n   \"dtype\": \"category\",\n   \"num_unique_values\":
3,\n   \"samples\": [\n   \"Third\",\n   \"First\"\
n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"who\",\n   \"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n   \"samples\": [\n   \"man\",\
n   \"woman\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"adult_male\",\n   \"properties\": {\n   \"dtype\":
\"boolean\",\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   false,\n   true\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"deck\",\n   \"properties\": {\n
\"dtype\": \"category\",\n   \"num_unique_values\": 7,\n
\"samples\": [\n   \"C\",\n   \"E\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"embark_town\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n   \"samples\": [\n
\"Southampton\",\n   \"Cherbourg\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"alive\",\n   \"properties\": {\
n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n
\"samples\": [\n   \"yes\",\n   \"no\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"alone\",\n   \"properties\": {\
n   \"dtype\": \"boolean\",\n   \"num_unique_values\": 2,\n
\"samples\": [\n   true,\n   false\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
# Data Cleaning
# Check for missing values
print(df.isnull().sum())
```

```
survived          0
pclass            0
sex               0
age             177
sibsp             0
parch             0
fare              0
embarked          2
class             0
who               0
adult_male        0
deck            688
embark_town       2
alive             0
alone             0
dtype: int64
```

```
# Fill missing values for 'age' with the median age
df['age'].fillna(df['age'].median(), inplace=True)

# Fill missing values for 'embarked' with the mode
df['embarked'].fillna(df['embarked'].mode()[0], inplace=True)

# Fill missing values for 'embark_town' with the mode
df['embark_town'].fillna(df['embark_town'].mode()[0], inplace=True)

# Convert 'deck' to string type to handle missing values properly
df['deck'] = df['deck'].astype(str)
# Fill missing values for 'deck' with a placeholder ('Unknown')
df['deck'].fillna('Unknown', inplace=True)

# Drop 'alive' column as it's redundant with 'survived'
df.drop(columns=['alive'], inplace=True)

# Check for any remaining missing values
print(df.isnull().sum())
```

```
survived        0
pclass          0
sex             0
age             0
sibsp           0
parch           0
fare            0
embarked        0
class           0
who             0
adult_male      0
deck            0
embark_town     0
```

```
alone          0
dtype: int64
```

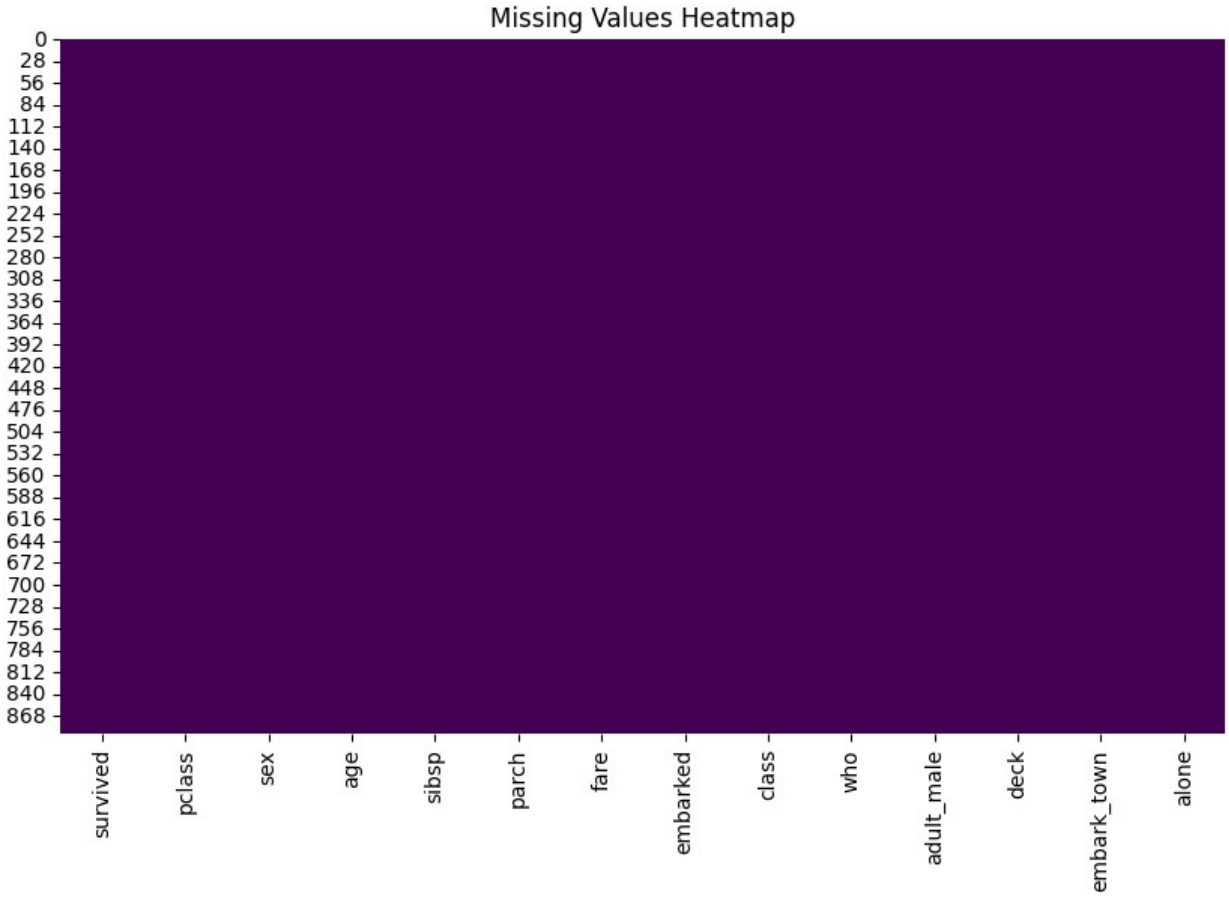
Exploratory Data Analysis

```
# Exploratory Data Analysis (EDA)
# Summary statistics
print(df.describe(include='all'))
```

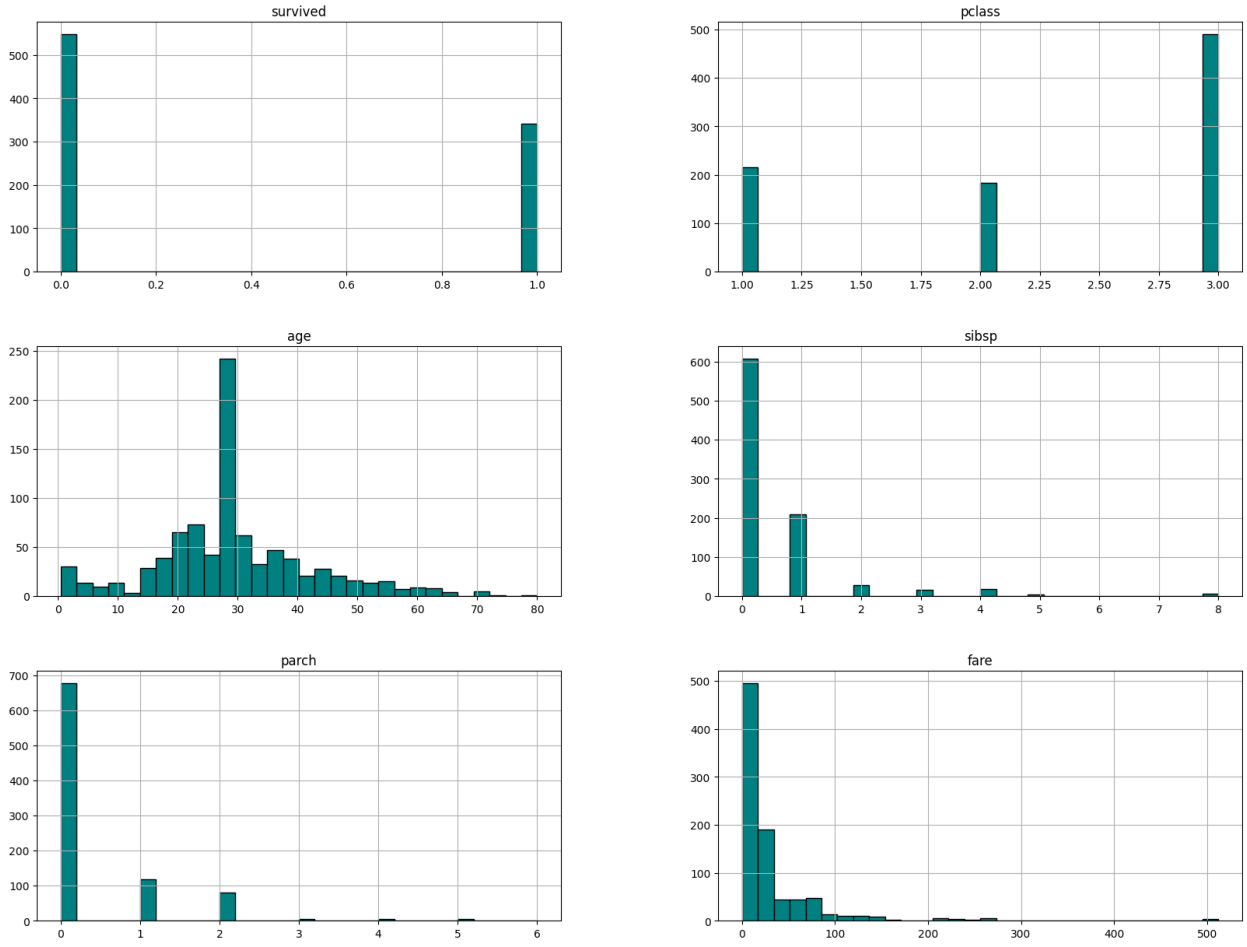
```
          survived      pclass   sex         age       sibsp
parch  \
count   891.000000  891.000000   891  891.000000  891.000000
891.000000
unique         NaN         NaN     2         NaN         NaN
NaN
top            NaN         NaN  male         NaN         NaN
NaN
freq           NaN         NaN   577         NaN         NaN
NaN
mean      0.383838    2.308642   NaN   29.361582    0.523008
0.381594
std       0.486592    0.836071   NaN   13.019697    1.102743
0.806057
min       0.000000    1.000000   NaN    0.420000    0.000000
0.000000
25%       0.000000    2.000000   NaN   22.000000    0.000000
0.000000
50%       0.000000    3.000000   NaN   28.000000    0.000000
0.000000
75%       1.000000    3.000000   NaN   35.000000    1.000000
0.000000
max       1.000000    3.000000   NaN   80.000000    8.000000
6.000000

              fare embarked  class  who adult_male deck  embark_town
alone
count   891.000000      891    891  891        891  891          891
891
unique         NaN        3      3    3          2    8            3
2
top            NaN        S  Third  man       True  nan  Southampton
True
freq           NaN      646    491  537        537  688          646
537
mean     32.204208      NaN    NaN  NaN        NaN  NaN          NaN
NaN
std      49.693429      NaN    NaN  NaN        NaN  NaN          NaN
NaN
```

```
min        0.000000       NaN     NaN  NaN         NaN  NaN           NaN
NaN
25%        7.910400       NaN     NaN  NaN         NaN  NaN           NaN
NaN
50%       14.454200       NaN     NaN  NaN         NaN  NaN           NaN
NaN
75%       31.000000       NaN     NaN  NaN         NaN  NaN           NaN
NaN
max      512.329200       NaN     NaN  NaN         NaN  NaN           NaN
NaN
```

```
# Visualizing missing values
plt.figure(figsize=(10, 6))
sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
plt.title('Missing Values Heatmap')
plt.show()
```

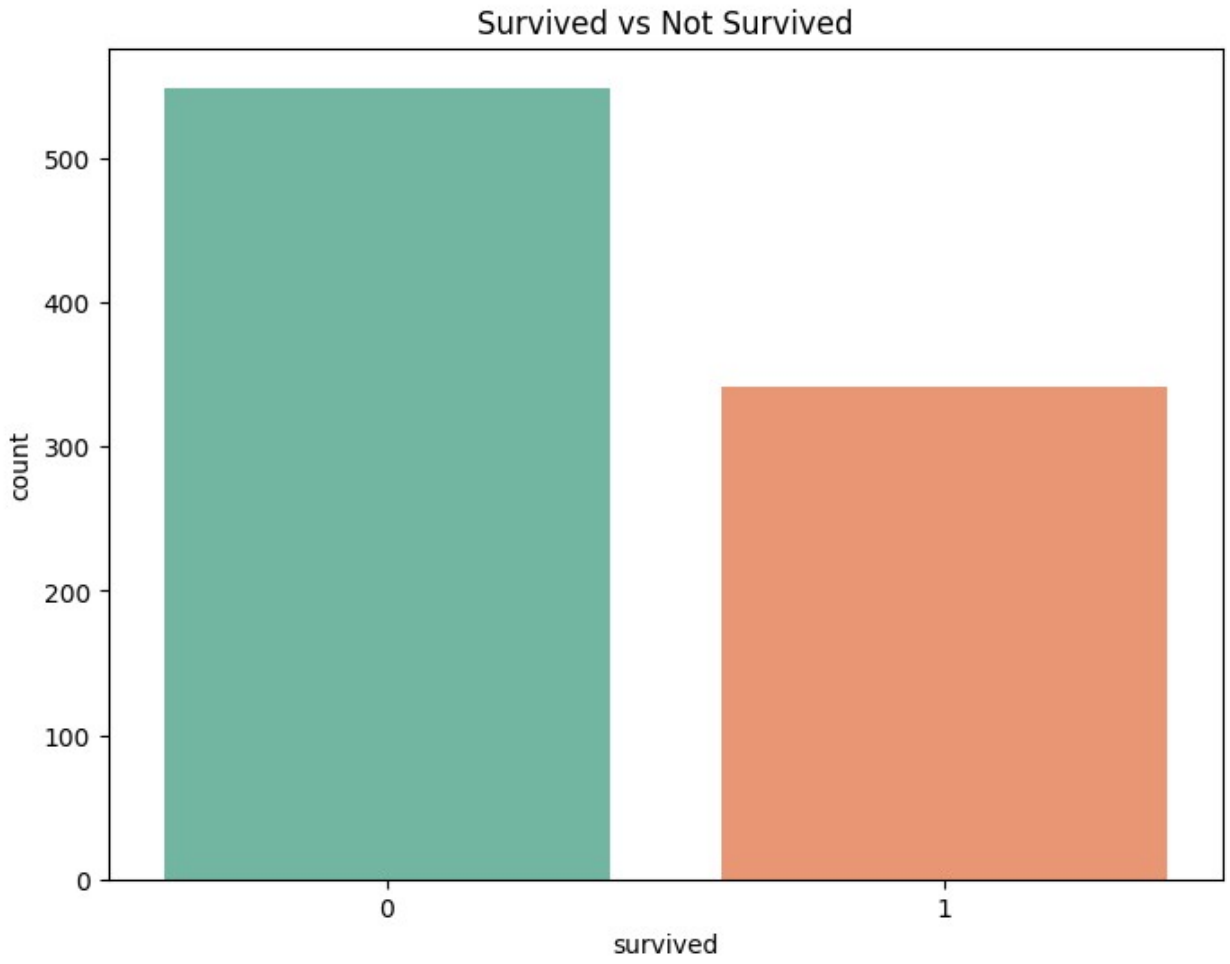



```
# Distribution of numerical features
df.hist(bins=30, figsize=(20, 15), color='teal', edgecolor='black')
plt.suptitle('Distribution of Numerical Features')
plt.show()
```



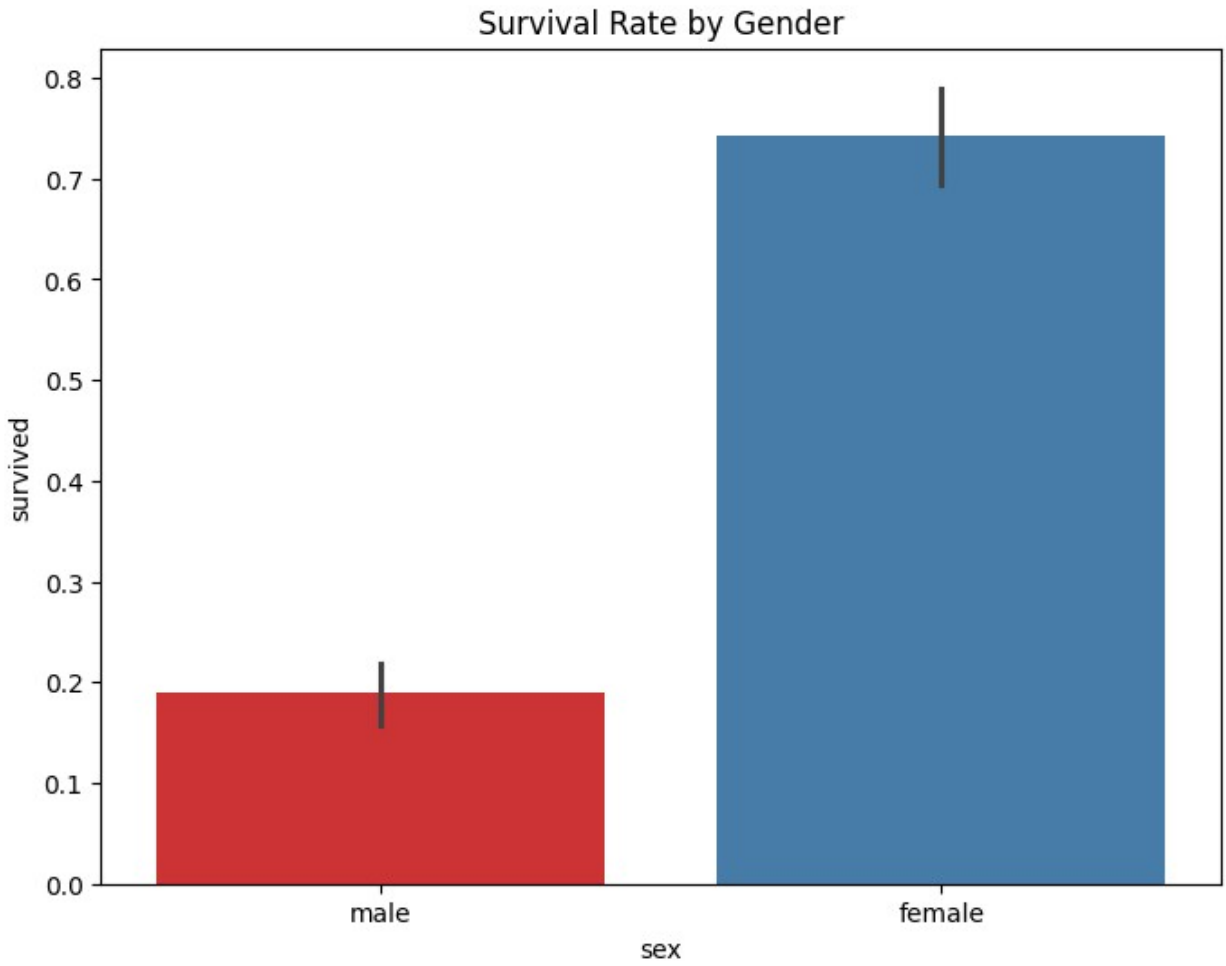

```
# Survived vs Not Survived
plt.figure(figsize=(8, 6))
sns.countplot(x='survived', data=df, palette='Set2')
plt.title('Survived vs Not Survived')
plt.show()
```

```
<ipython-input-14-dbae49908f4e>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.countplot(x='survived', data=df, palette='Set2')
```



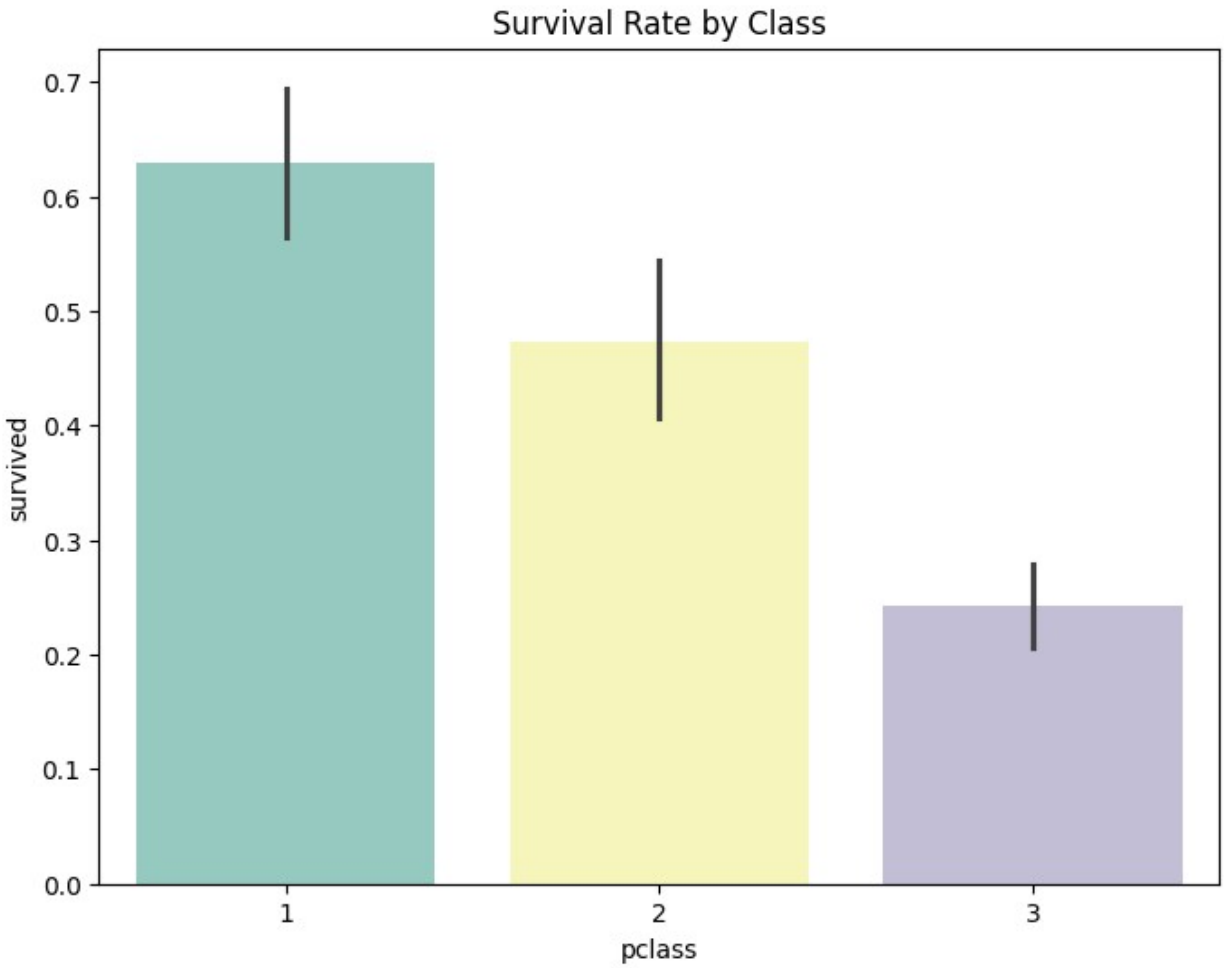

```
# Survival rate by gender
plt.figure(figsize=(8, 6))
sns.barplot(x='sex', y='survived', data=df, palette='Set1')
plt.title('Survival Rate by Gender')
plt.show()
```

```
<ipython-input-15-142366c60e8d>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.barplot(x='sex', y='survived', data=df, palette='Set1')
```




```
# Survival rate by class
plt.figure(figsize=(8, 6))
sns.barplot(x='pclass', y='survived', data=df, palette='Set3')
plt.title('Survival Rate by Class')
plt.show()
```

```
<ipython-input-16-1b7860bbf81b>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.barplot(x='pclass', y='survived', data=df, palette='Set3')
```

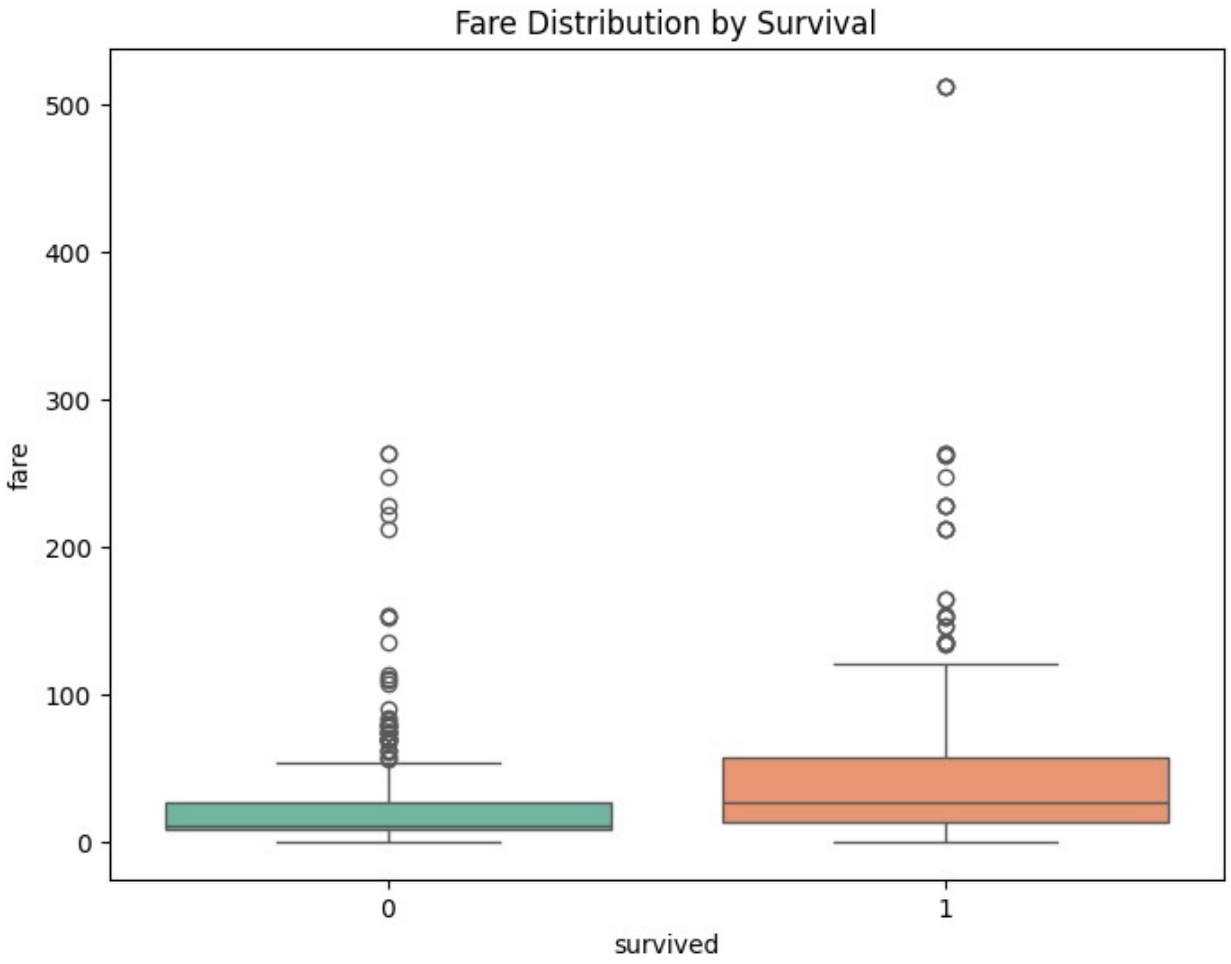



```
# Pairplot of the dataset
sns.pairplot(df, hue='survived', palette='Set1')
plt.suptitle('Pairplot of the Titanic Dataset', y=1.02)
plt.show()
```


```
# Correlation matrix (only numerical columns)
plt.figure(figsize=(10, 8))
numerical_df = df.select_dtypes(include=[np.number])
corr_matrix = numerical_df.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix')
plt.show()
```

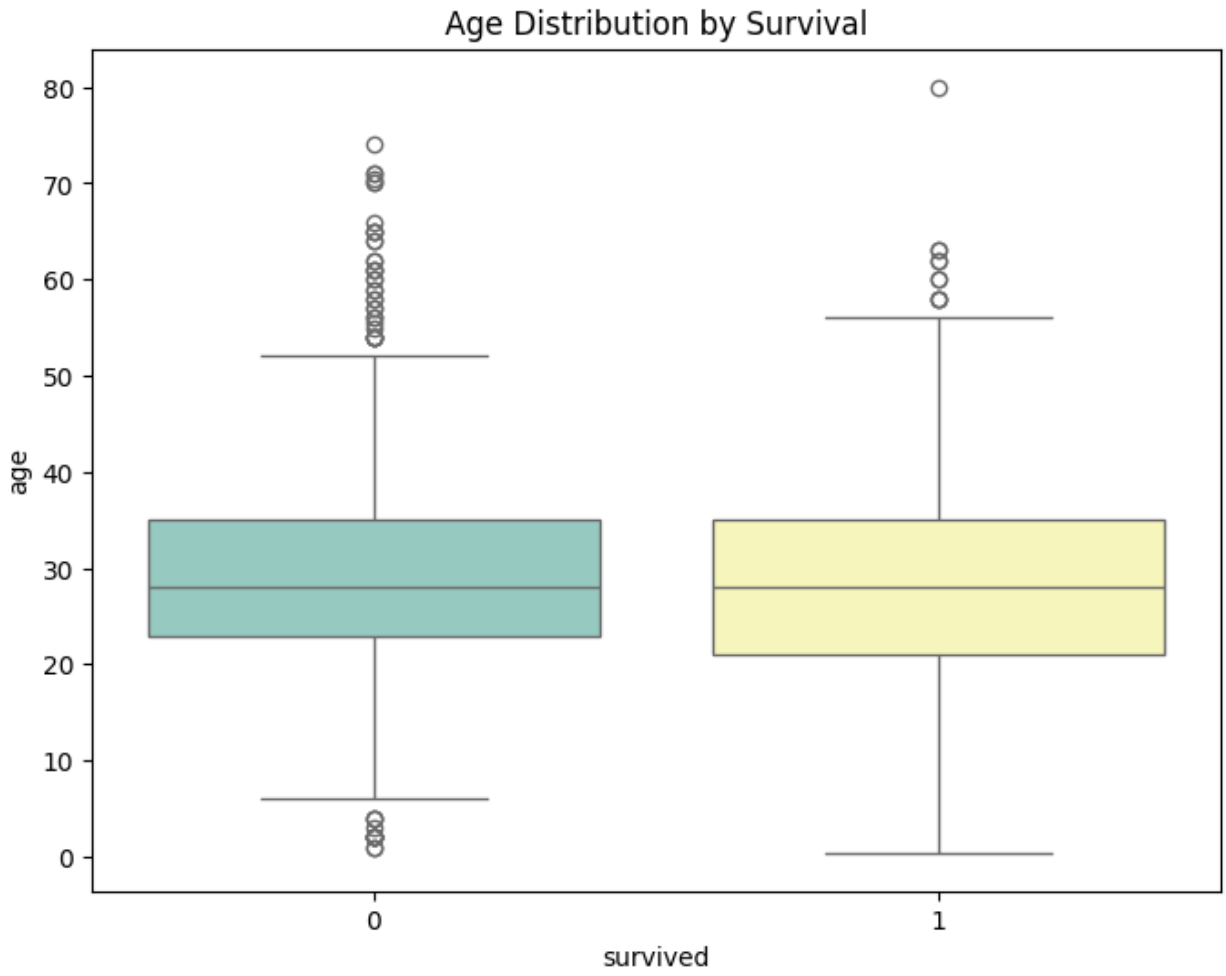

```
# Analyzing 'fare' distribution by survival
plt.figure(figsize=(8, 6))
sns.boxplot(x='survived', y='fare', data=df, palette='Set2')
plt.title('Fare Distribution by Survival')
plt.show()
```

```
<ipython-input-20-27724c38c14f>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.boxplot(x='survived', y='fare', data=df, palette='Set2')
```




```
# Analyzing 'age' distribution by survival
plt.figure(figsize=(8, 6))
sns.boxplot(x='survived', y='age', data=df, palette='Set3')
plt.title('Age Distribution by Survival')
plt.show()
```

```
<ipython-input-21-8b4927ef564a>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.boxplot(x='survived', y='age', data=df, palette='Set3')
```


Age Distribution by Survival
80
70
60
50
40
30
20
10
0
age
0
1
survived