

```
In [3]: import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import math
td = pd.read_csv(r"C:\Users\Saiko\OneDrive\Desktop\Data Analytics\Titanic\tra
```

```
In [4]: td.head(5)
```

```
Out[4]:
```

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

**Titanic Dataset Description

1. Data Overview

- Total Entries: 887
- Columns: 8

2. Types of Data:

- Numerical: Survived, Pclass, Age, Siblings/Spouses Aboard, Parents/Children Aboard, Fare
- Categorical: Name, Sex

3. Column Description

- Survived (0 = No, 1 = Yes) → Indicates whether the passenger survived.
- Pclass (1st, 2nd, 3rd) → Passenger class.
- Name → Passenger's full name.
- Sex → Gender of the passenger.
- Age → Age of the passenger.

- Siblings/Spouses Aboard → Number of siblings or spouses aboard.
- Parents/Children Aboard → Number of parents or children aboard.
- Fare → Ticket price paid by the passenger.

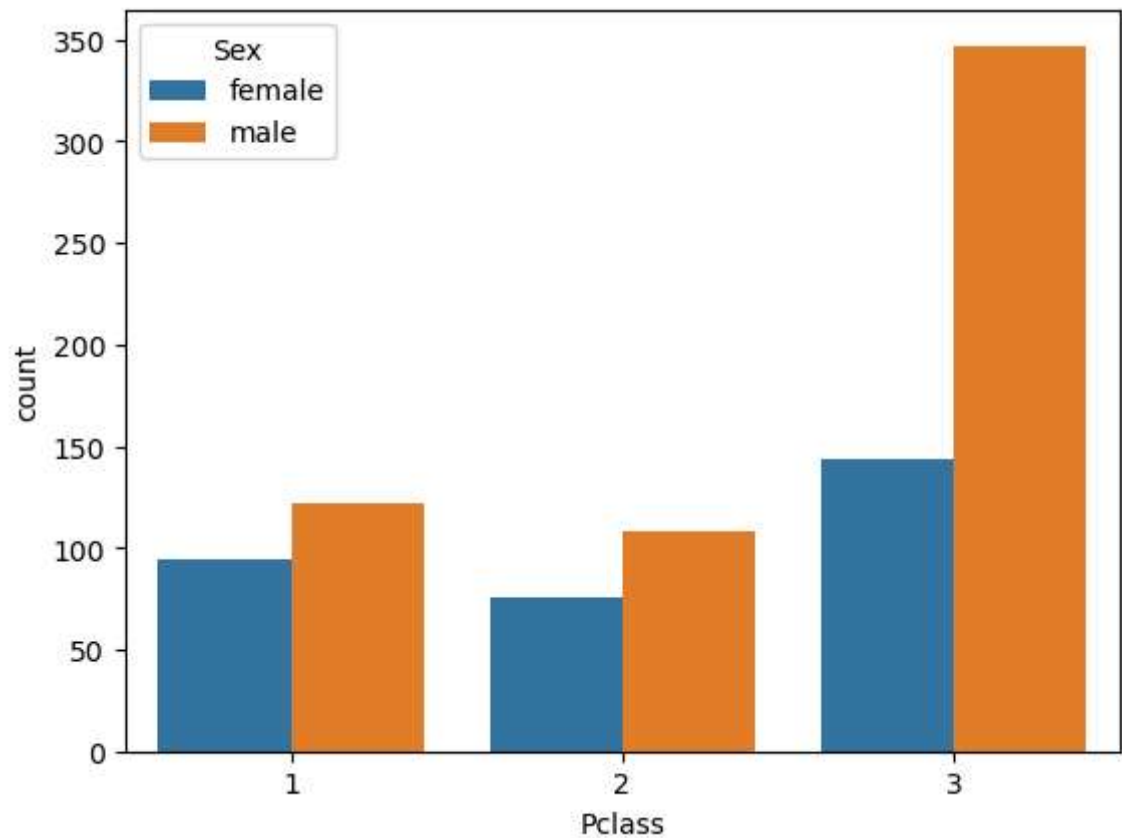
```
In [5]: ▶ plt.figure(figsize=(20, 25))
```

```
Out[5]: <Figure size 2000x2500 with 0 Axes>
```

```
<Figure size 2000x2500 with 0 Axes>
```

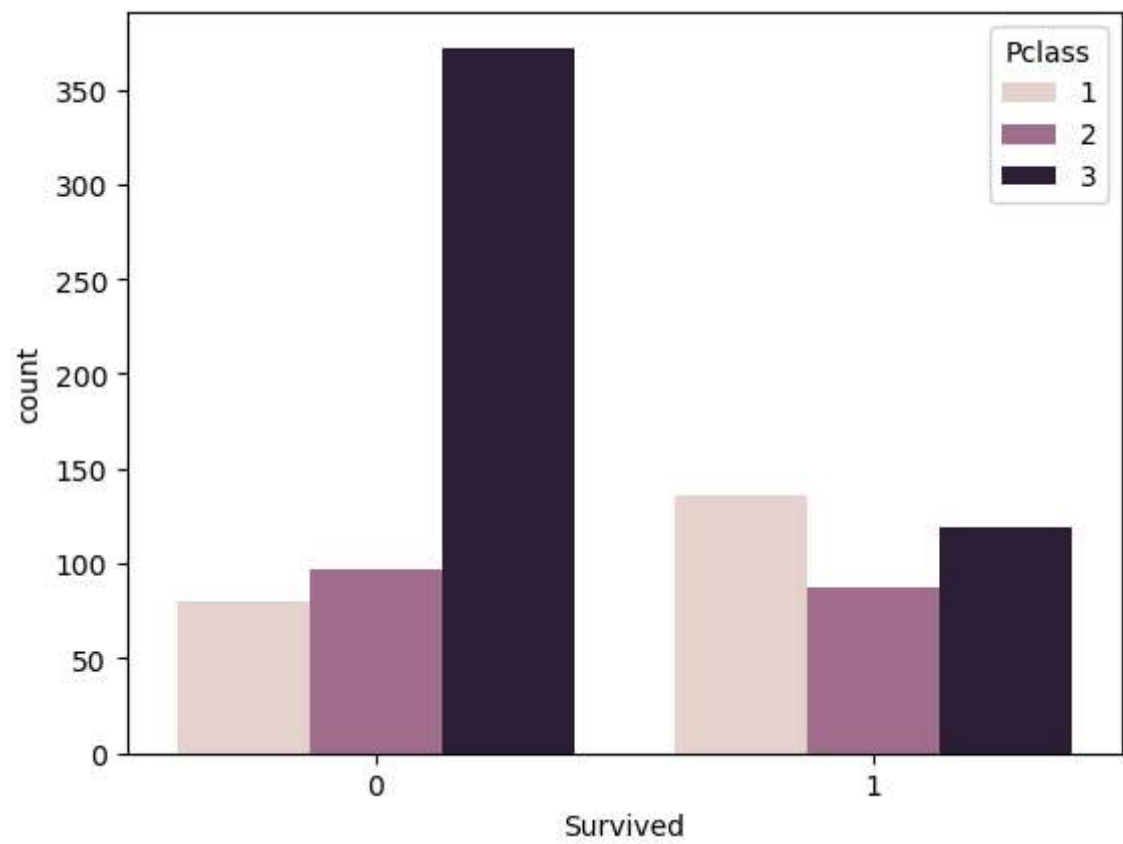
```
In [6]: ▶ sns.countplot(x="Pclass", hue="Sex", data=td)
```

```
Out[6]: <Axes: xlabel='Pclass', ylabel='count'>
```



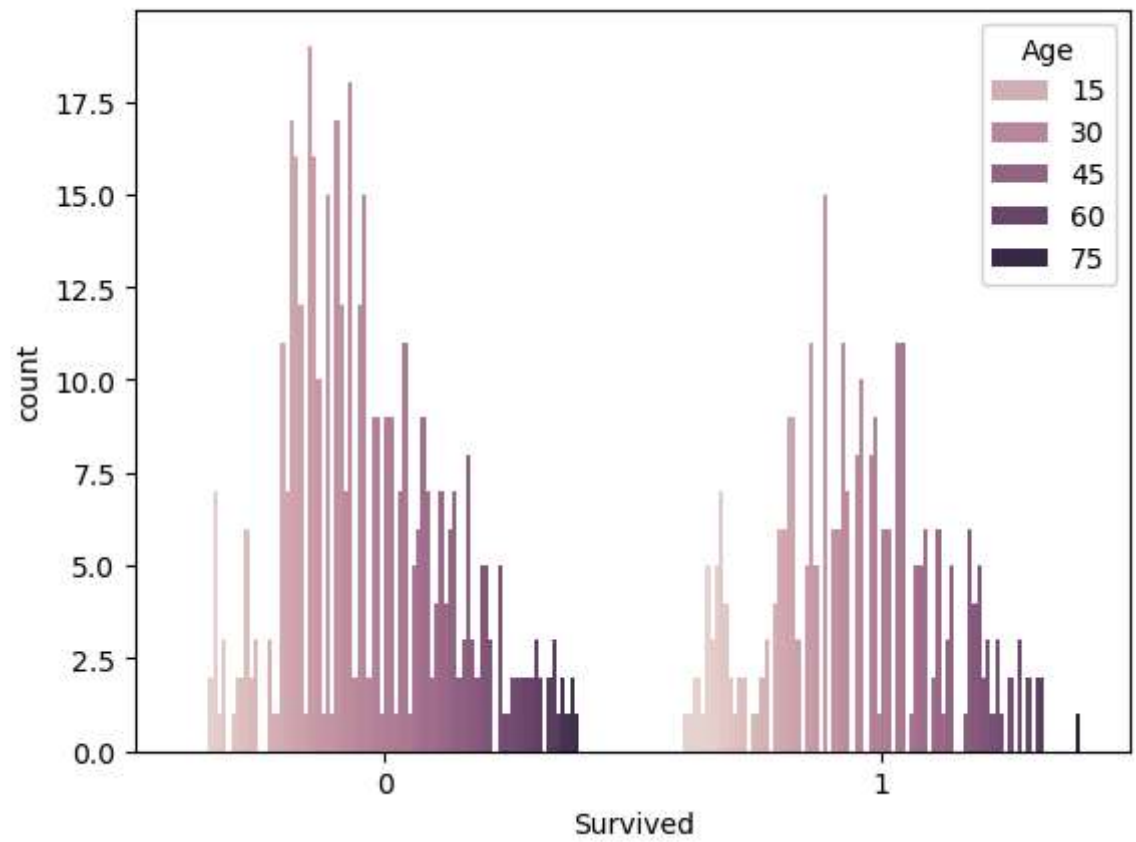
```
In [7]: sns.countplot(x="Survived", hue="Pclass", data=td)
```

```
Out[7]: <Axes: xlabel='Survived', ylabel='count'>
```



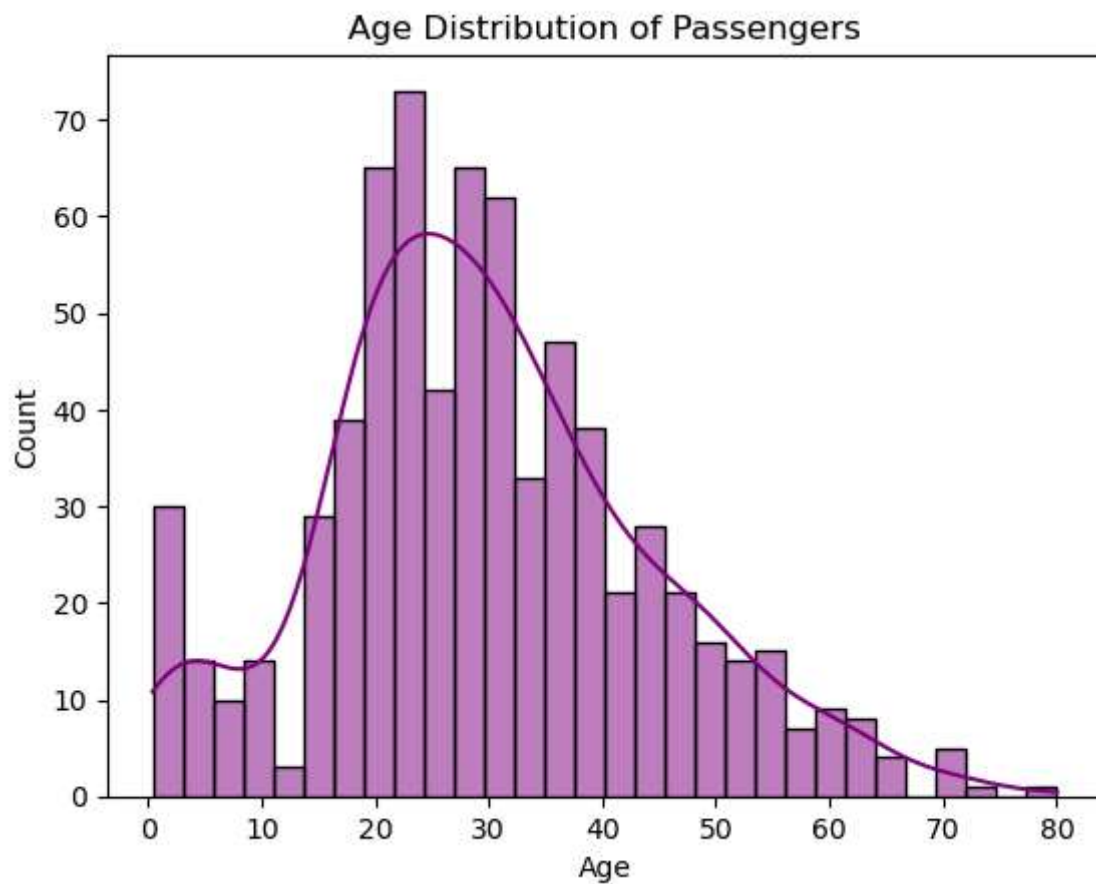
```
In [8]: sns.countplot(x="Survived", hue="Age", data=td)
```

```
Out[8]: <Axes: xlabel='Survived', ylabel='count'>
```



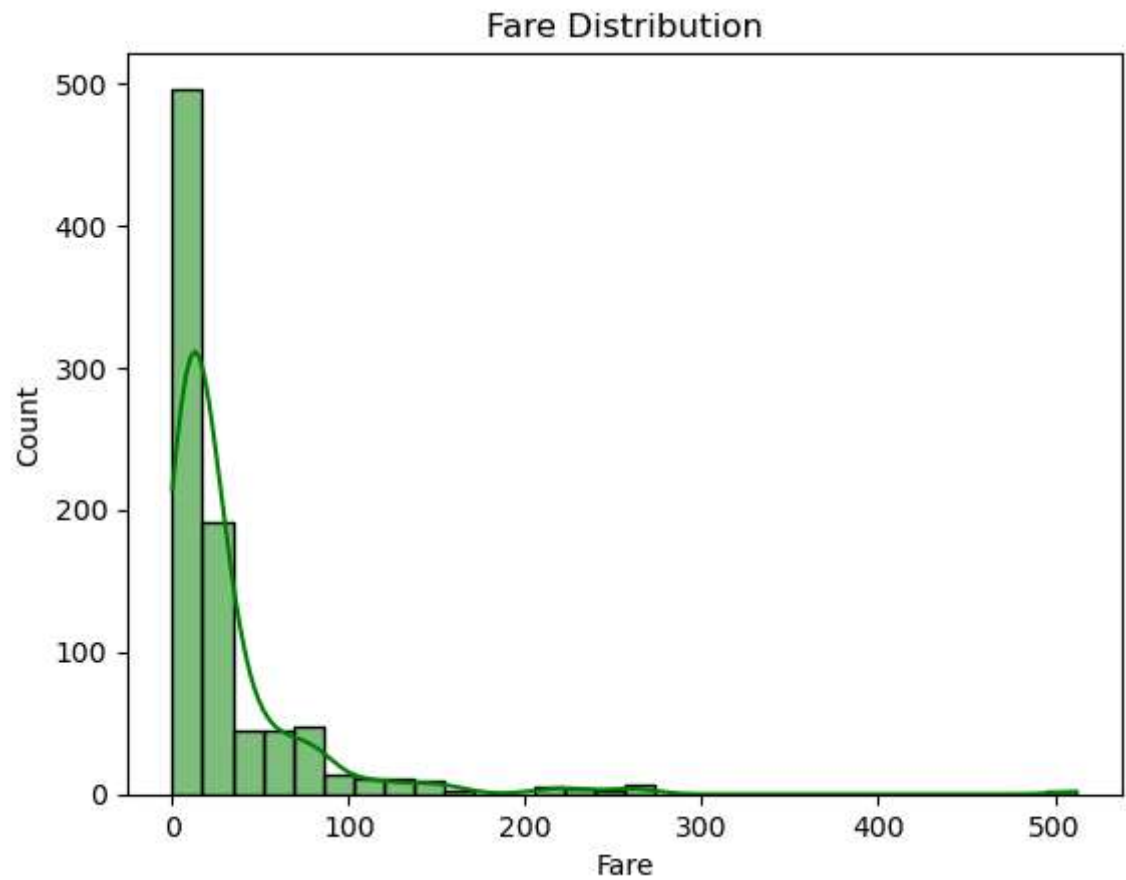
```
In [9]: ▶ sns.histplot(td["Age"], bins=30, kde=True, color="purple")  
plt.title("Age Distribution of Passengers")
```

Out[9]: Text(0.5, 1.0, 'Age Distribution of Passengers')



```
In [10]: sns.histplot(td["Fare"], bins=30, kde=True, color="green")  
plt.title("Fare Distribution")
```

```
Out[10]: Text(0.5, 1.0, 'Fare Distribution')
```



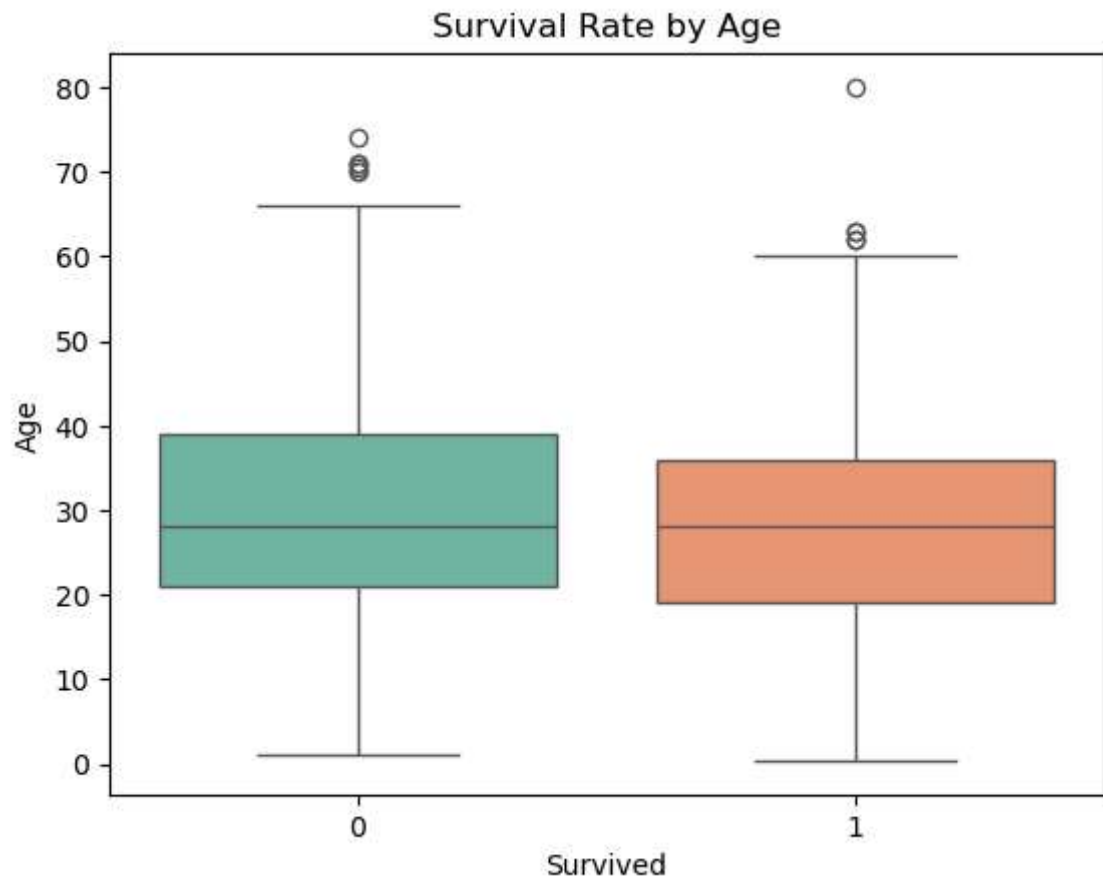
```
In [11]: sns.boxplot(x="Survived", y="Age", data=td, palette="Set2")  
plt.title("Survival Rate by Age")
```

C:\Users\Saiko\AppData\Local\Temp\ipykernel_2532\829218853.py:1: 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=td, palette="Set2")
```

Out[11]: Text(0.5, 1.0, 'Survival Rate by Age')



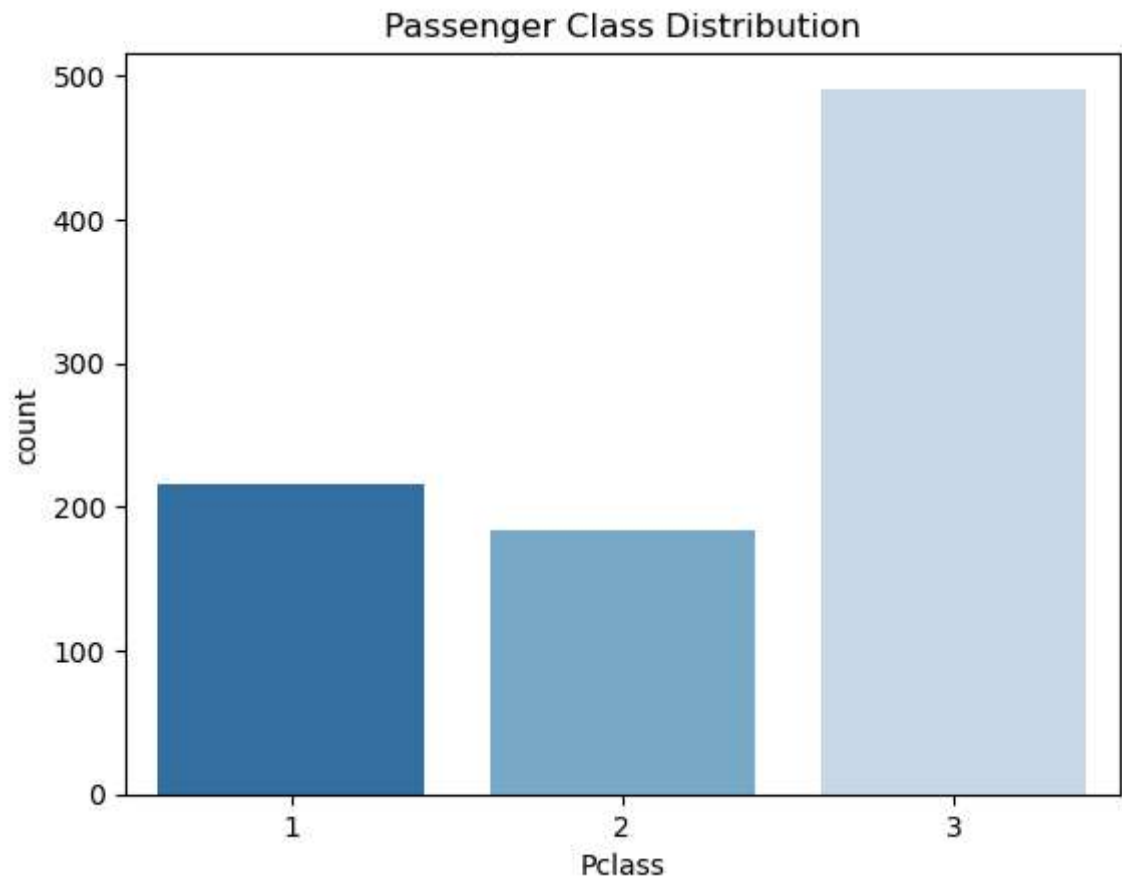
```
In [12]: sns.countplot(x="Pclass", data=td, palette="Blues_r")  
plt.title("Passenger Class Distribution")
```

C:\Users\Saiko\AppData\Local\Temp\ipykernel_2532\2069748178.py:1: 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.

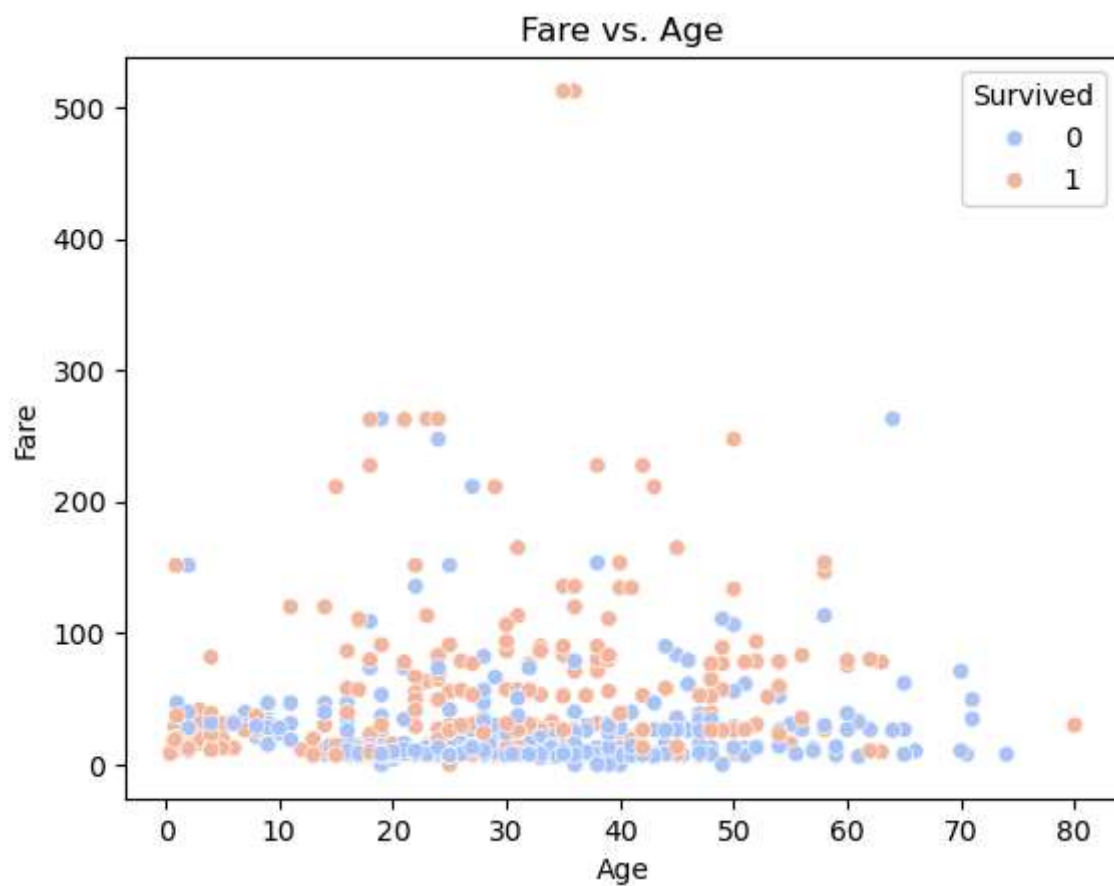
```
sns.countplot(x="Pclass", data=td, palette="Blues_r")
```

Out[12]: Text(0.5, 1.0, 'Passenger Class Distribution')




```
In [13]: sns.scatterplot(x="Age", y="Fare", hue="Survived", data=td, palette="coolwarm",  
plt.title("Fare vs. Age")
```

```
Out[13]: Text(0.5, 1.0, 'Fare vs. Age')
```



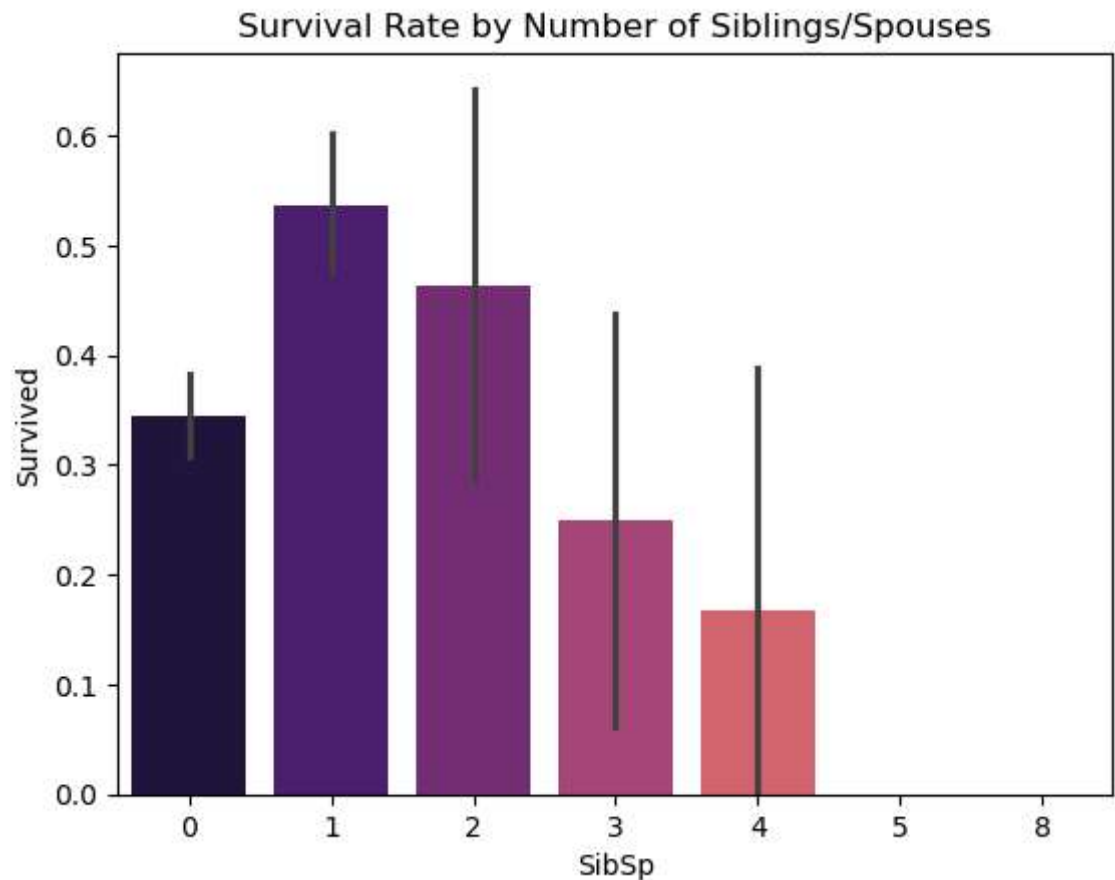
```
In [14]: ▶ sns.barplot(x="SibSp", y="Survived", data=td, palette="magma")  
plt.title("Survival Rate by Number of Siblings/Spouses")
```

C:\Users\Saiko\AppData\Local\Temp\ipykernel_2532\1592040016.py:1: 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="SibSp", y="Survived", data=td, palette="magma")
```

Out[14]: Text(0.5, 1.0, 'Survival Rate by Number of Siblings/Spouses')



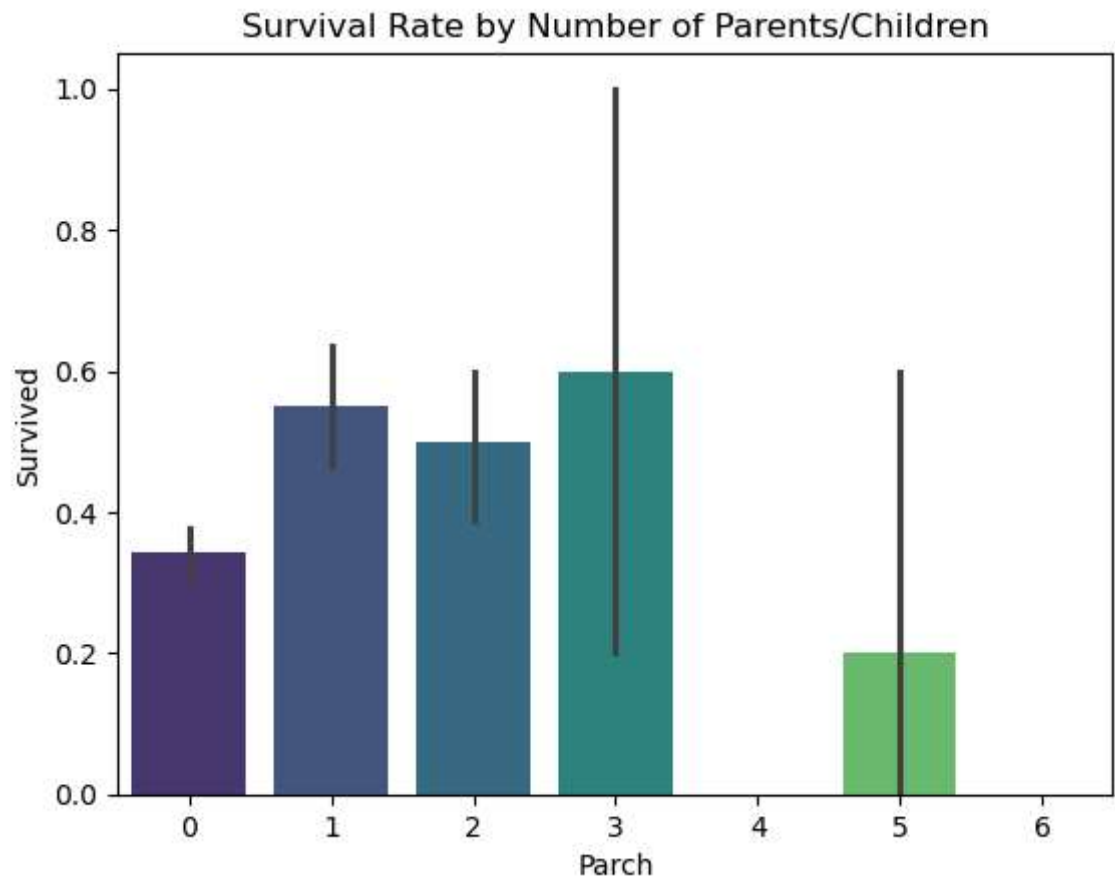
```
In [15]: ▶ sns.barplot(x="Parch", y="Survived", data=td, palette="viridis")  
plt.title("Survival Rate by Number of Parents/Children")
```

C:\Users\Saiko\AppData\Local\Temp\ipykernel_2532\3310087714.py:1: 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.

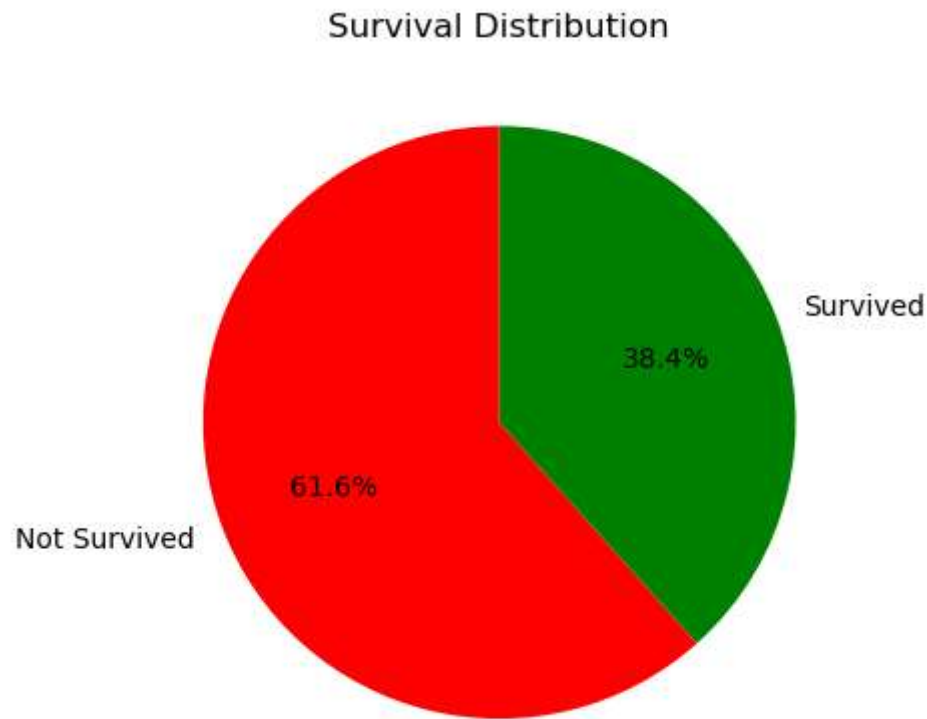
```
sns.barplot(x="Parch", y="Survived", data=td, palette="viridis")
```

Out[15]: Text(0.5, 1.0, 'Survival Rate by Number of Parents/Children')



```
In [16]: ▶ survival_counts = td["Survived"].value_counts()
plt.pie(survival_counts, labels=["Not Survived", "Survived"], autopct="%1.1f%")
plt.title("Survival Distribution")
```

```
Out[16]: Text(0.5, 1.0, 'Survival Distribution')
```



```
In [17]: ▶ td.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
```

In [18]: `td.isnull()`

Out[18]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	False	False	False	False	False	False	False	False	False	False	False	True
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	True
...
886	False	False	False	False	False	False	False	False	False	False	False	True
887	False	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	False	True
889	False	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	False	True

891 rows × 12 columns



In [19]: `td.isnull().sum()`

Out[19]:

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

In [20]: `td.drop(["Name", "PassengerId", "Ticket"], axis=1, inplace=True)`

In [21]: `td.Age.mean()`

Out[21]: 29.69911764705882

In [22]: `td['Age'] = td['Age'].fillna(td['Age'].mean())`

In [23]: `td.isnull().sum()`

```
Out[23]: Survived      0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Cabin      687
Embarked     2
dtype: int64
```

In [24]: `td.isnull().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   bool
1   Pclass      891 non-null   bool
2   Sex         891 non-null   bool
3   Age         891 non-null   bool
4   SibSp       891 non-null   bool
5   Parch       891 non-null   bool
6   Fare        891 non-null   bool
7   Cabin       891 non-null   bool
8   Embarked    891 non-null   bool
dtypes: bool(9)
memory usage: 8.0 KB
```

In [25]: `td.isnull().sum()`

```
Out[25]: Survived      0
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        0
Cabin      687
Embarked     2
dtype: int64
```

In [26]: `td.isnull().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    bool
1   Pclass      891 non-null    bool
2   Sex         891 non-null    bool
3   Age         891 non-null    bool
4   SibSp       891 non-null    bool
5   Parch       891 non-null    bool
6   Fare        891 non-null    bool
7   Cabin       891 non-null    bool
8   Embarked    891 non-null    bool
dtypes: bool(9)
memory usage: 8.0 KB
```

In [27]: `td.loc[(td["Survived"] == 1) & (td["Pclass"] == 1) & (td["Cabin"].isna()), "Cabin"] = "S"`
`td.loc[(td["Survived"] == 1) & (td["Pclass"] == 2) & (td["Cabin"].isna()), "Cabin"] = "S"`
`td.loc[(td["Survived"] == 1) & (td["Pclass"] == 3) & (td["Cabin"].isna()), "Cabin"] = "S"`
`td.loc[(td["Survived"] == 0) & (td["Pclass"] == 1) & (td["Cabin"].isna()), "Cabin"] = "S"`
`td.loc[(td["Survived"] == 0) & (td["Pclass"] == 2) & (td["Cabin"].isna()), "Cabin"] = "S"`
`td.loc[(td["Survived"] == 0) & (td["Pclass"] == 3) & (td["Cabin"].isna()), "Cabin"] = "S"`

In [28]: `td["Embarked"].fillna("S", inplace=True)`

C:\Users\Saiko\AppData\Local\Temp\ipykernel_2532\4252291910.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method. The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or 'df[col] = df[col].method(value)' instead, to perform the operation inplace on the original object.

```
td["Embarked"].fillna("S", inplace=True)
```

In [29]: `td.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   Cabin        891 non-null   object  
 8   Embarked     891 non-null   object  
dtypes: float64(2), int64(4), object(3)
memory usage: 62.8+ KB
```

In [30]: `print(td['Sex'])`

```
0      male
1     female
2     female
3     female
4      male
...
886     male
887     female
888     female
889      male
890      male
Name: Sex, Length: 891, dtype: object
```

In [31]: `print(td['Embarked'])`

```
0      S
1      C
2      S
3      S
4      S
..
886     S
887     S
888     S
889     C
890     Q
Name: Embarked, Length: 891, dtype: object
```


In [32]: `print(td['Pclass'])`

```
0      3
1      1
2      3
3      1
4      3
..
886    2
887    1
888    3
889    1
890    3
Name: Pclass, Length: 891, dtype: int64
```

In [33]: `cabin=pd.get_dummies(td['Cabin'])`

In [34]: `print(cabin)`

```

      A10   A14   A16   A19   A20   A23   A24   A26   A31   A32
\
0   False False False False False False False False False False
1   False False False False False False False False False False
2   False False False False False False False False False False
3   False False False False False False False False False False
4   False False False False False False False False False False
..   ...   ...   ...   ...   ...   ...   ...   ...   ...
886 False False False False False False False False False False
887 False False False False False False False False False False
888 False False False False False False False False False False
889 False False False False False False False False False False
890 False False False False False False False False False False

      ...   F E69   F G63   F G73   F2   F33   F38   F4   G   G6
T
0   ... False False False False False False False False True False Fa
lse
1   ... False False False False False False False False False False Fa
lse
2   ... False False False False False False False False False False Fa
lse
3   ... False False False False False False False False False False Fa
lse
4   ... False False False False False False False False True False Fa
lse
..   ...   ...   ...   ...   ...   ...   ...   ...   ...
...
886 ... False False False False False False False False False False Fa
lse
887 ... False False False False False False False False False False Fa
lse
888 ... False False False False False False False False True False Fa
lse
889 ... False False False False False False False False False False Fa
lse
890 ... False False False False False False False False True False Fa
lse

[891 rows x 152 columns]
```

In [35]: `td.drop(["Pclass", "Sex", "Embarked", "Cabin"], axis=1, inplace=True)`

In [36]: `td=pd.concat([td, cabin], axis=1)`

In [37]: `print(td.info())`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Columns: 157 entries, Survived to T
dtypes: bool(152), float64(2), int64(3)
memory usage: 167.2 KB
None
```

In [38]: `X=td.drop(["Survived"],axis=1)`

In [39]: `print(X)`

```

      Age  SibSp  Parch    Fare   A10   A14   A16   A19   A20  \
0  22.000000     1     0   7.2500  False  False  False  False  False
1  38.000000     1     0  71.2833  False  False  False  False  False
2  26.000000     0     0   7.9250  False  False  False  False  False
3  35.000000     1     0  53.1000  False  False  False  False  False
4  35.000000     0     0   8.0500  False  False  False  False  False
..      ...     ...     ...     ...     ...     ...     ...     ...
886  27.000000     0     0  13.0000  False  False  False  False  False
887  19.000000     0     0  30.0000  False  False  False  False  False
888  29.699118     1     2  23.4500  False  False  False  False  False
889  26.000000     0     0  30.0000  False  False  False  False  False
890  32.000000     0     0   7.7500  False  False  False  False  False

      A23  ...  F E69  F G63  F G73   F2   F33   F38   F4   G  \
0  False  ...  False  False  False  False  False  False  False  False  True
1  False  ...  False  False  False  False  False  False  False  False  False
2  False  ...  False  False  False  False  False  False  False  False  False
3  False  ...  False  False  False  False  False  False  False  False  False
4  False  ...  False  False  False  False  False  False  False  False  True
..      ...     ...     ...     ...     ...     ...     ...     ...     ...
886  False  ...  False  False  False  False  False  False  False  False  False
887  False  ...  False  False  False  False  False  False  False  False  False
888  False  ...  False  False  False  False  False  False  False  False  True
889  False  ...  False  False  False  False  False  False  False  False  False
890  False  ...  False  False  False  False  False  False  False  False  True

      G6      T
0  False  False
1  False  False
2  False  False
3  False  False
4  False  False
..      ...     ...
886  False  False
887  False  False
888  False  False
889  False  False
890  False  False

```

[891 rows x 156 columns]

In [40]: `X.info()`

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 891 entries, 0 to 890  
Columns: 156 entries, Age to T  
dtypes: bool(152), float64(2), int64(2)  
memory usage: 160.2 KB
```

In [41]: `y=td["Survived"]`

In [42]: `print(y)`

```
0      0  
1      1  
2      1  
3      1  
4      0  
..  
886    0  
887    1  
888    0  
889    1  
890    0  
Name: Survived, Length: 891, dtype: int64
```

In [43]: `from sklearn.model_selection import train_test_split`

In [44]: `X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state`

In [45]: `X_train.info()`

```
<class 'pandas.core.frame.DataFrame'>  
Index: 623 entries, 114 to 37  
Columns: 156 entries, Age to T  
dtypes: bool(152), float64(2), int64(2)  
memory usage: 116.8 KB
```

In [46]: `X_test.info()`

```
<class 'pandas.core.frame.DataFrame'>  
Index: 268 entries, 862 to 92  
Columns: 156 entries, Age to T  
dtypes: bool(152), float64(2), int64(2)  
memory usage: 50.2 KB
```

In [47]: `from sklearn.linear_model import LogisticRegression`

In [48]: `lm=LogisticRegression(max_iter=10000)`

```
In [49]: ▶ print(lm.fit(X_train,y_train))
```

```
LogisticRegression(max_iter=10000)
```

```
In [57]: ▶ Predictions=lm.predict(X_test)
```

```
In [58]: ▶ from sklearn.metrics import classification_report
```

```
In [52]: ▶ print(classification_report(y_test,Predictions))
```

	precision	recall	f1-score	support
0	0.97	0.90	0.93	153
1	0.87	0.97	0.92	115
accuracy			0.93	268
macro avg	0.92	0.93	0.92	268
weighted avg	0.93	0.93	0.93	268

```
In [53]: ▶ from sklearn.metrics import confusion_matrix
```

```
In [54]: ▶ print(confusion_matrix(y_test,Predictions))
```

```
[[137  16]
 [  4 111]]
```

```
In [55]: ▶ from sklearn.metrics import accuracy_score
```

```
In [56]: ▶ print(accuracy_score(y_test,Predictions))
```

```
0.9253731343283582
```

Dataset Observations:

1. Survival Rate Observations

- Only ~38% of passengers survived, while ~62% perished.
- Women had a much higher survival rate (74%) compared to men (18%).
- Children (age < 10) had a better chance of survival than adults.

2. Gender and Survival Insights

- ~74% of females survived, while only ~18% of males survived.
- "Women and children first" policy led to a higher survival rate for women.
- Male survival was much lower, especially in 3rd Class (~13%).

3. Summary:

- Higher class, higher fare, and being female or a child greatly increased survival chances.
- Men in 3rd class had the worst survival rate (~13%).

- Passengers with large families struggled to survive.

Dropping irrelevant columns (Name, PassengerId, Ticket) Handling missing values by filling missing Age values with the mean. However, I still haven't located the model building and evaluation code. Let me extract more cells to find the relevant information.

The extracted cells still show data cleaning and preprocessing steps, such as:

Filling missing Cabin values based on Survived and Pclass. Filling missing Embarked values with "S". Printing some of the dataset's categorical features (Sex, Embarked, Pclass).

One-hot encoding the Cabin column using `pd.get_dummies()`. Dropping categorical features (Pclass, Sex, Embarked, and Cabin) and replacing them with the encoded versions. Defining the features (X) and the target (y): $X \rightarrow$ Contains the independent variables (features). $y \rightarrow$ Contains the target variable (Survived).

In []: ▶