Assignment 3

Name : Bala Murugan V V

Reg No : 21BIT0725

Steps: 1 Begin by importing the necessary libraries. 2 Proceed with the dataset importation. 3 Conduct a thorough check for any missing values within the data. 4 Create visual representations of the data for better understanding. 5 Identify and address any outliers present in the dataset. 6 Differentiate between the independent and dependent variables. 7 Apply encoding techniques as required for data preparation. 8 Divide the dataset into training and testing subsets. 9 Normalize or standardize the features for consistent scaling.

Importing the necessary libraries

In [3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder,StandardScaler
```

Dataset Importation

In [4]:
```
data_set = pd.read_csv('titanic.csv')
```

In [5]:
```
data_set.head()
```

Out[5]:

	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 Briggs Th...	female	38.0	1	0	PC 17599	71.2833
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500

In [6]:
```
data_set.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 [7]:
```
data_set.shape
```

Out[7]: (891, 12)

In [8]:
```
data_set.describe()
```

Out[8]:

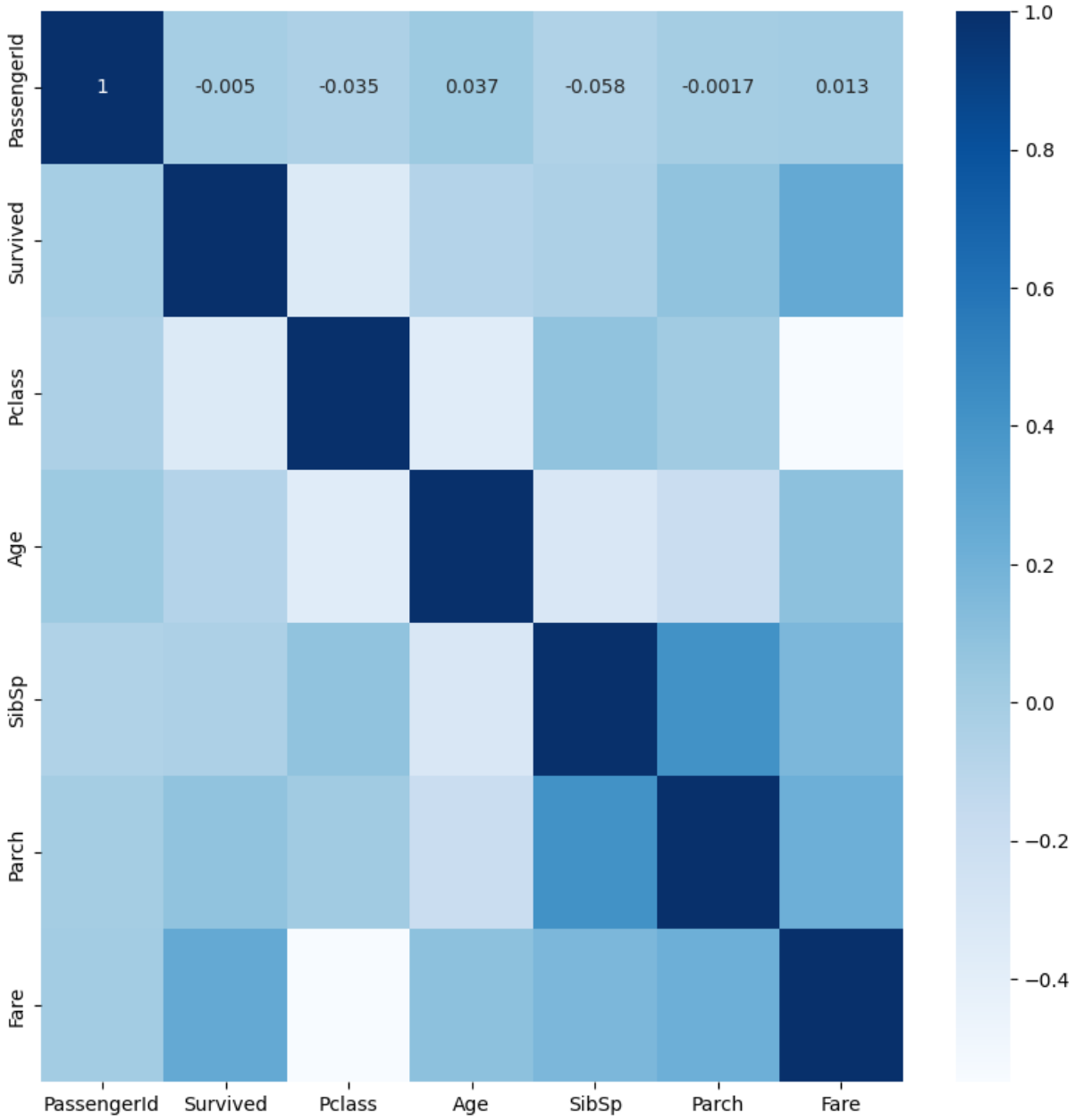
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [11]:
```
numeric_data = data_set.select_dtypes(include=[np.number])
corr = numeric_data.corr()
print(corr)
```

```
             PassengerId  Survived    Pclass       Age     SibSp     Parch  \
PassengerId     1.000000 -0.005007 -0.035144  0.036847 -0.057527 -0.001652
Survived       -0.005007  1.000000 -0.338481 -0.077221 -0.035322  0.081629
Pclass         -0.035144 -0.338481  1.000000 -0.369226  0.083081  0.018443
Age             0.036847 -0.077221 -0.369226  1.000000 -0.308247 -0.189119
SibSp          -0.057527 -0.035322  0.083081 -0.308247  1.000000  0.414838
Parch          -0.001652  0.081629  0.018443 -0.189119  0.414838  1.000000
Fare            0.012658  0.257307 -0.549500  0.096067  0.159651  0.216225

                 Fare
PassengerId  0.012658
Survived     0.257307
Pclass      -0.549500
Age          0.096067
SibSp        0.159651
Parch        0.216225
Fare         1.000000
```

In [14]:
```
plt.subplots(figsize=(10,10))
sns.heatmap(corr,annot=True,cmap = "Blues")
```

Out[14]: <Axes: >



Checking for Null values

In [15]:
```
data_set.isnull().any()
```

Out[15]:
```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age             True
SibSp          False
Parch          False
Ticket         False
Fare           False
Cabin           True
Embarked        True
dtype: bool
```

In [16]:
```
data_set.isnull().sum()
```

Out[16]:
```
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
```

Inference: Age , Cabin and Embarked has null values

In [18]:
```
#Since age is a numeric value we can use mean imputing
age_mean = data_set['Age'].mean()
age_mean
```

Out[18]:
```
29.69911764705882
```

In [19]:
```
data_set['Age'].fillna(age_mean,inplace=True)
```

Since Cabin is a categorical data and most of its values are null we drop this column. Embarked has less values so we use mode imputing

In [20]:
```
data_set.drop(columns = "Cabin",inplace = True)
embarked_mode=data_set['Embarked'].mode()[0]
embarked_mode
```

Out[20]:
```
'S'
```

In [22]:
```
data_set.isnull().any()
```

Out[22]:
```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age            False
SibSp          False
Parch          False
Ticket         False
Fare           False
Embarked        True
dtype: bool
```

In [24]:
```
data_set.isnull().sum()
```

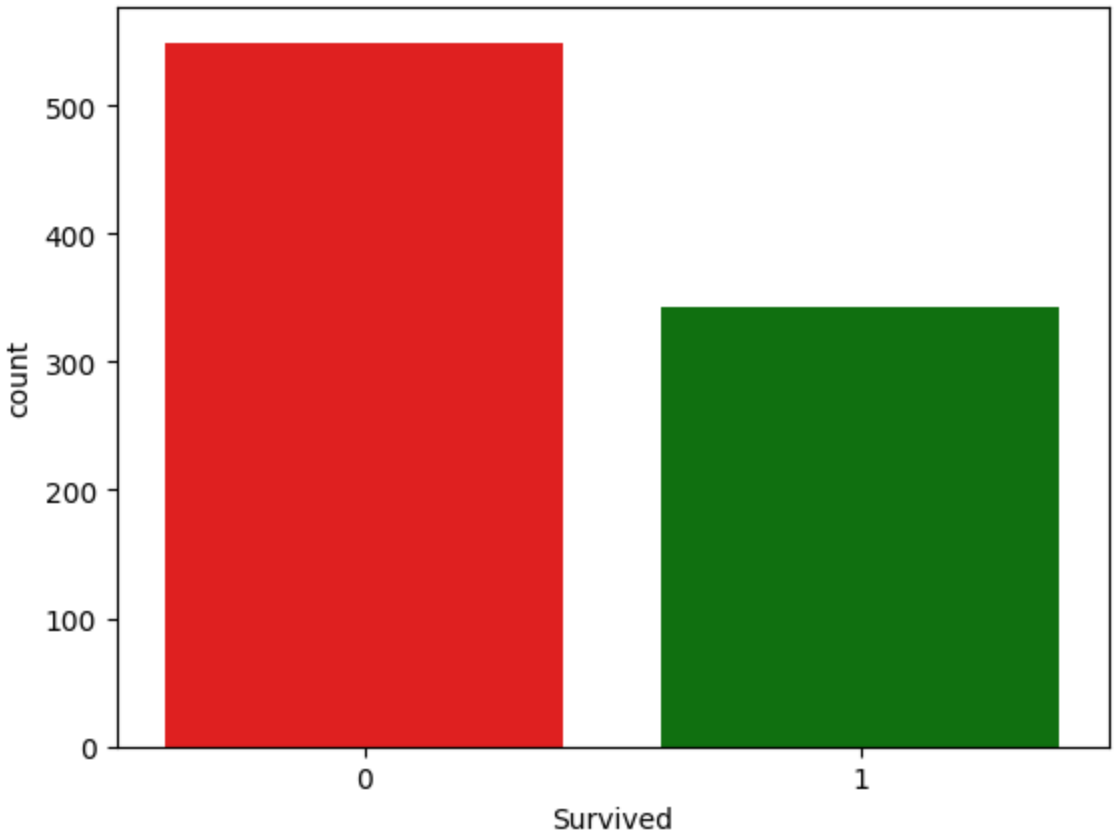
Out[24]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       2
dtype: int64
```

Data Visualisation

In [26]:
```
sns.countplot(x='Survived',data=data_set,palette = ['red','green'])
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```

Out[26]:
```
<Axes: xlabel='Survived', ylabel='count'>
```

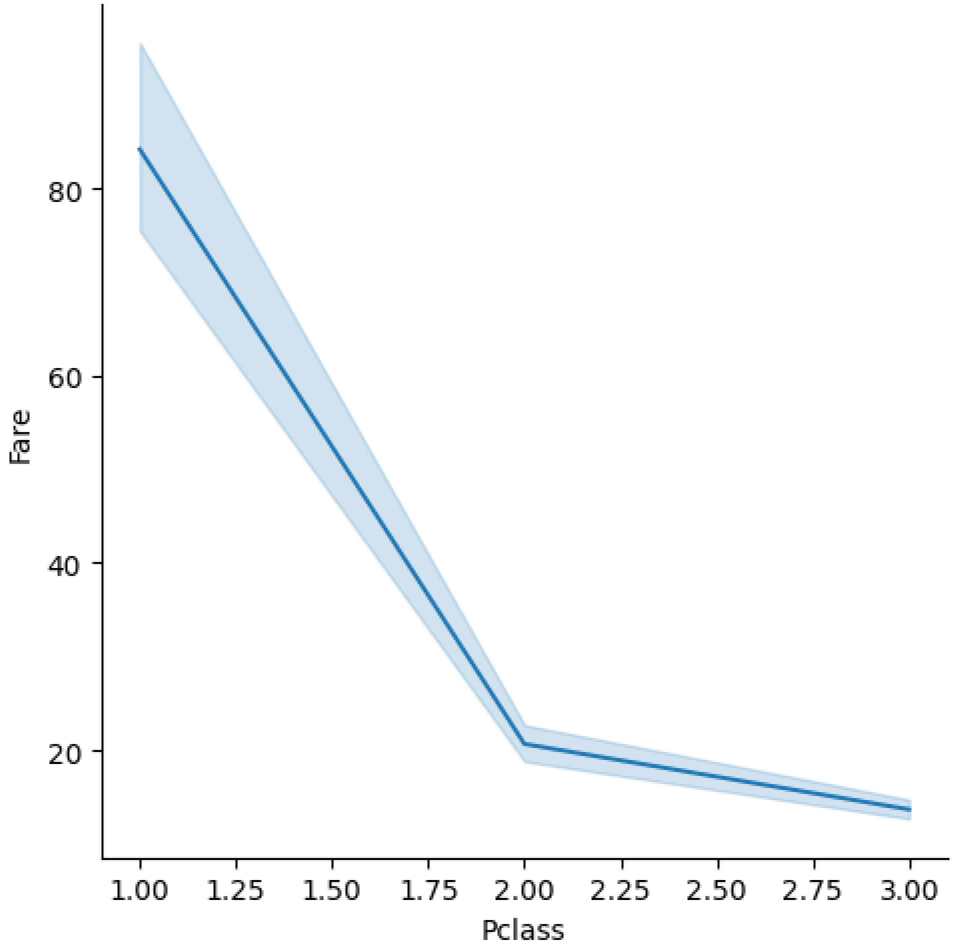


Inference : Majority didn't survive

In [28]:
```
sns.relplot(x='Pclass',y='Fare',data=data_set,kind='line')
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[28]: <seaborn.axisgrid.FacetGrid at 0x1c242b546d0>

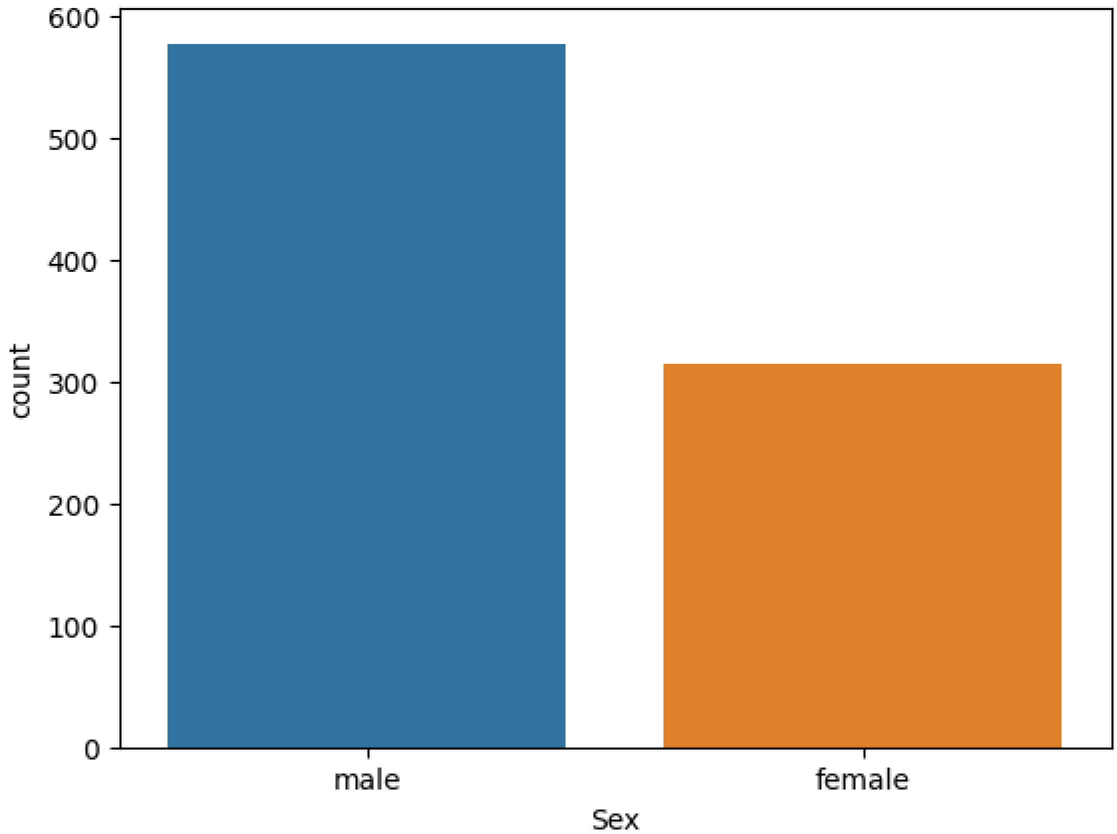


Inference : First class are expensive than second class tickets

In [30]:
```
sns.countplot(x='Sex',data=data_set)
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```

Out[30]:
```
<Axes: xlabel='Sex', ylabel='count'>
```



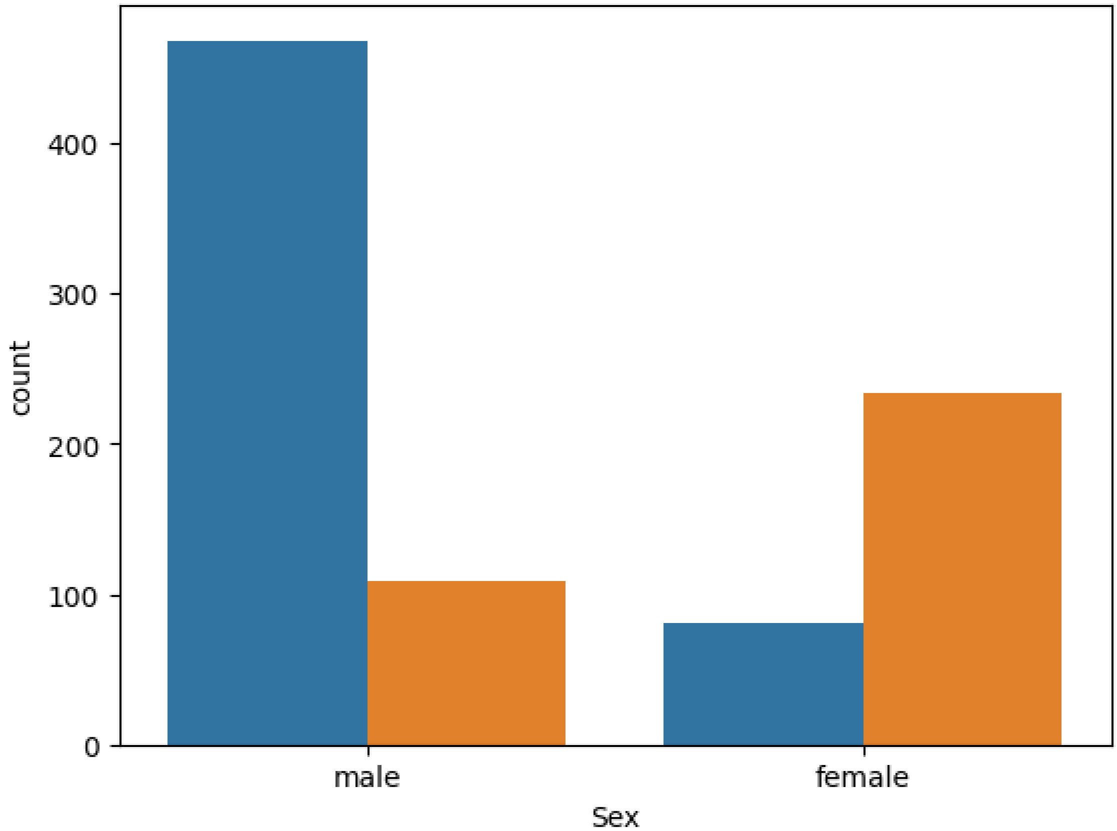
In [ ]:
```
Inference: More male passengers than female passengers
```

In [32]:
```
sns.countplot(x='Sex',hue='Survived',data=data_set)
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[32], line 1
----> 1 sns.countplot(x='Sex',hue='Survived',data=data_set)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:2955, in countplot(data, x, y, hue, order, hue_order, orient, color, palette, s
aturation, width, dodge, ax, **kwargs)
   2952 if ax is None:
   2953     ax = plt.gca()
-> 2955 plotter.plot(ax, kwargs)
   2956 return ax

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:1587, in _BarPlotter.plot(self, ax, bar_kws)
   1585 """Make the plot."""
   1586 self.draw_bars(ax, bar_kws)
-> 1587 self.annotate_axes(ax)
   1588 if self.orient == "h":
   1589     ax.invert_yaxis()

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:767, in _CategoricalPlotter.annotate_axes(self, ax)
    764     ax.set_ylim(-.5, len(self.plot_data) - .5, auto=None)
    766 if self.hue_names is not None:
--> 767     ax.legend(loc="best", title=self.hue_title)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\axes\_ax
es.py:322, in Axes.legend(self, *args, **kwargs)
    204 @_docstring.dedent_interpd
    205 def legend(self, *args, **kwargs):
    206     """
    207     Place a legend on the Axes.
    208
   (...)
    320     .. plot:: gallery/text_labels_and_annotations/legend.py
    321     """
--> 322     handles, labels, kwargs = mlegend._parse_legend_args([self], *args, **kw
args)
    323     self.legend_ = mlegend.Legend(self, handles, labels, **kwargs)
    324     self.legend_._remove_method = self._remove_legend

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\legend.p
y:1361, in _parse_legend_args(axs, handles, labels, *args, **kwargs)
   1357     handles = [handle for handle, label
   1358                in zip(_get_legend_handles(axs, handlers), labels)]
   1360 elif len(args) == 0:  # 0 args: automatically detect labels and handles.
-> 1361     handles, labels = _get_legend_handles_labels(axs, handlers)
   1362     if not handles:
   1363         log.warning(
   1364             "No artists with labels found to put in legend.  Note that "
   1365             "artists whose label start with an underscore are ignored "
   1366             "when legend() is called with no argument.")

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\legend.p
y:1291, in _get_legend_handles_labels(axs, legend_handler_map)
```

```
   1289 for handle in _get_legend_handles(axs, legend_handler_map):
   1290     label = handle.get_label()
-> 1291     if label and not label.startswith('_'):
   1292         handles.append(handle)
   1293         labels.append(label)

AttributeError: 'numpy.int64' object has no attribute 'startswith'
```



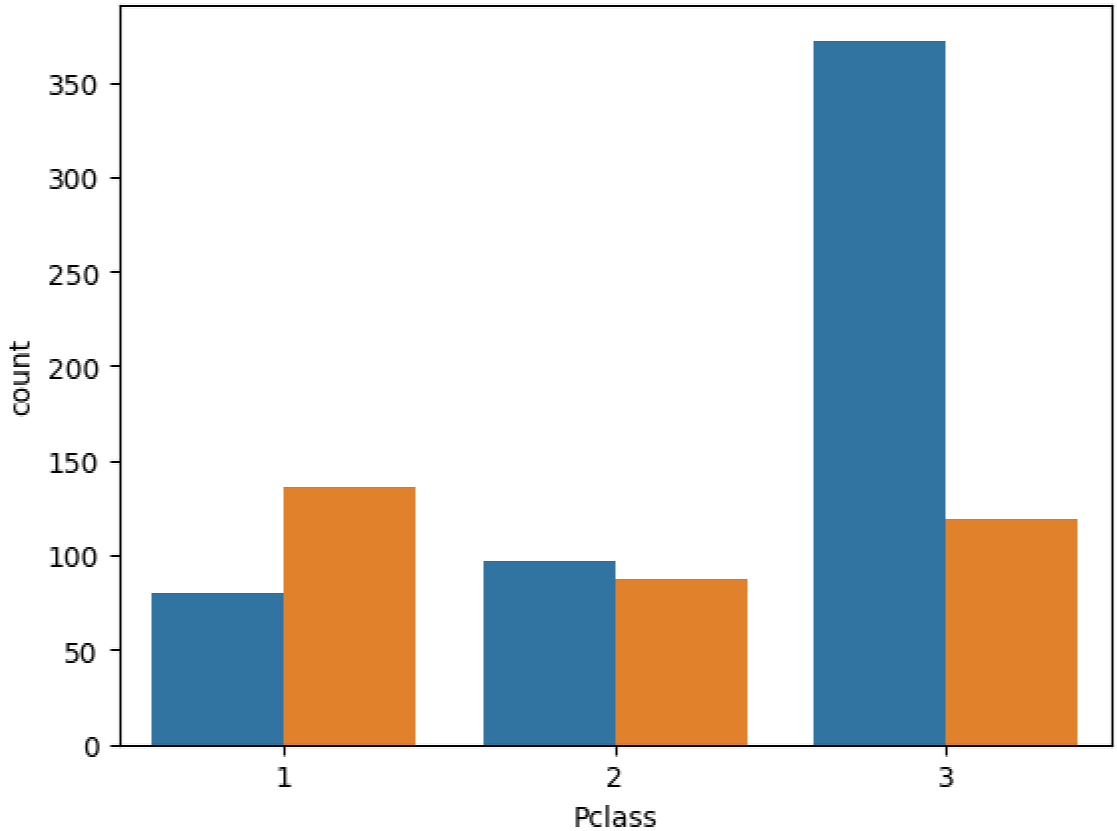
Most of the survivers were female

In [33]:
```
sns.countplot(x='Pclass',hue='Survived',data=data_set)
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[33], line 1
----> 1 sns.countplot(x='Pclass',hue='Survived',data=data_set)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:2955, in countplot(data, x, y, hue, order, hue_order, orient, color, palette, s
aturation, width, dodge, ax, **kwargs)
   2952 if ax is None:
   2953     ax = plt.gca()
-> 2955 plotter.plot(ax, kwargs)
   2956 return ax

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:1587, in _BarPlotter.plot(self, ax, bar_kws)
   1585 """Make the plot."""
   1586 self.draw_bars(ax, bar_kws)
-> 1587 self.annotate_axes(ax)
   1588 if self.orient == "h":
   1589     ax.invert_yaxis()

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:767, in _CategoricalPlotter.annotate_axes(self, ax)
    764     ax.set_ylim(-.5, len(self.plot_data) - .5, auto=None)
    766 if self.hue_names is not None:
--> 767     ax.legend(loc="best", title=self.hue_title)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\axes\_ax
es.py:322, in Axes.legend(self, *args, **kwargs)
    204 @_docstring.dedent_interpd
    205 def legend(self, *args, **kwargs):
    206     """
    207     Place a legend on the Axes.
    208
   (...)
    320     .. plot:: gallery/text_labels_and_annotations/legend.py
    321     """
--> 322     handles, labels, kwargs = mlegend._parse_legend_args([self], *args, **kw
args)
    323     self.legend_ = mlegend.Legend(self, handles, labels, **kwargs)
    324     self.legend_._remove_method = self._remove_legend

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\legend.p
y:1361, in _parse_legend_args(axs, handles, labels, *args, **kwargs)
   1357     handles = [handle for handle, label
   1358                in zip(_get_legend_handles(axs, handlers), labels)]
   1360 elif len(args) == 0:  # 0 args: automatically detect labels and handles.
-> 1361     handles, labels = _get_legend_handles_labels(axs, handlers)
   1362     if not handles:
   1363         log.warning(
   1364             "No artists with labels found to put in legend.  Note that "
   1365             "artists whose label start with an underscore are ignored "
   1366             "when legend() is called with no argument.")

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\legend.p
y:1291, in _get_legend_handles_labels(axs, legend_handler_map)
```

```
   1289 for handle in _get_legend_handles(axs, legend_handler_map):
   1290     label = handle.get_label()
-> 1291     if label and not label.startswith('_'):
   1292         handles.append(handle)
   1293         labels.append(label)

AttributeError: 'numpy.int64' object has no attribute 'startswith'
```

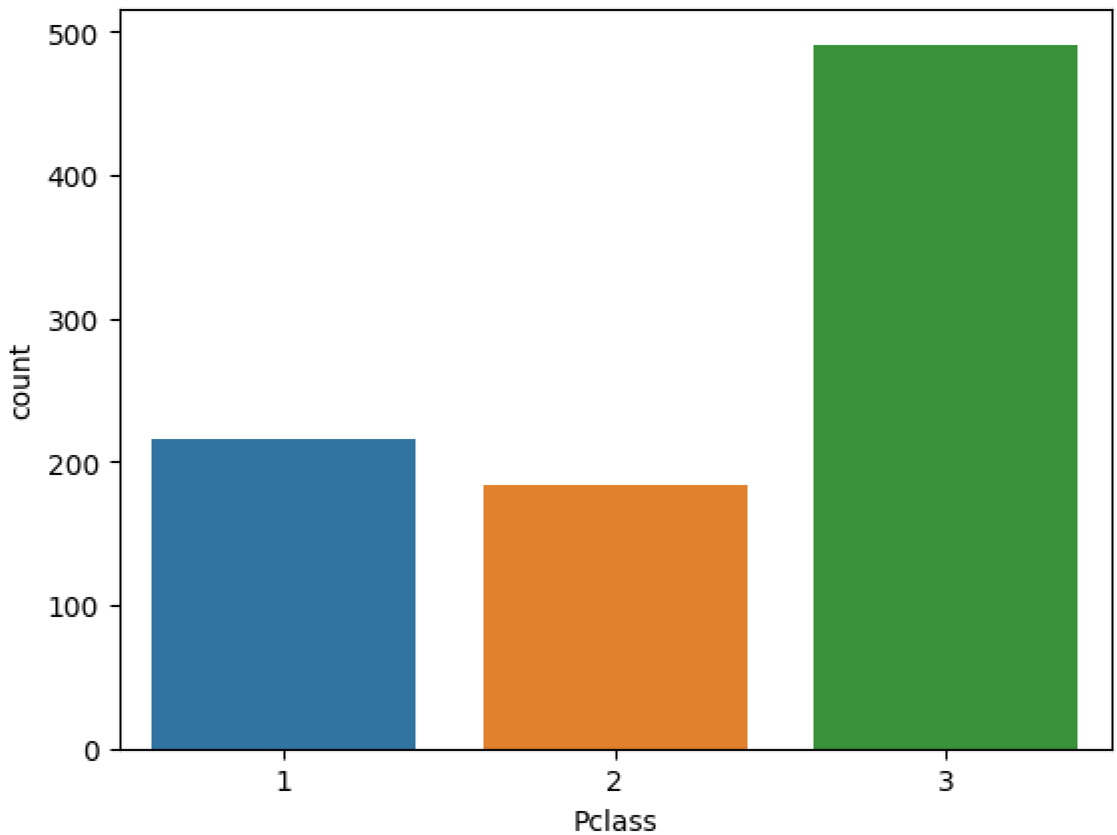


In [35]:

```
sns.countplot(x='Pclass',data=data_set)
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```

Out[35]: <Axes: xlabel='Pclass', ylabel='count'>

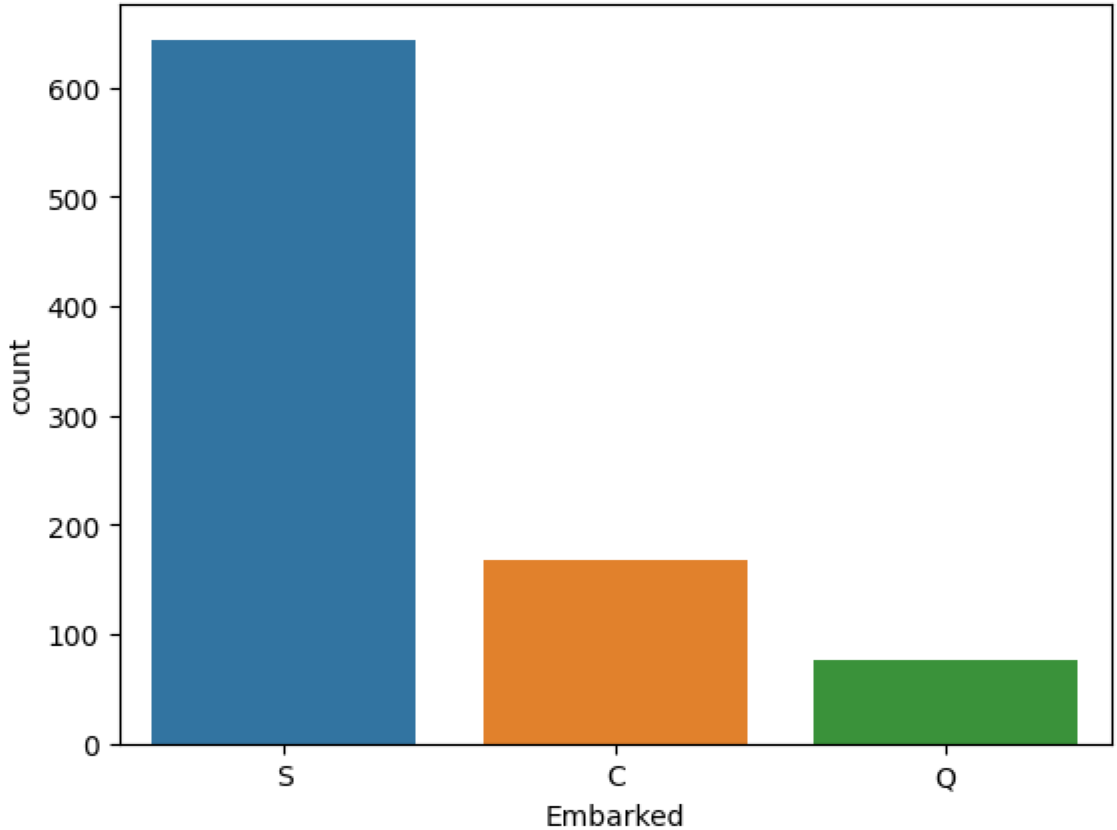


Majority travelled in third class

In [36]:
```
sns.countplot(x='Embarked',data=data_set)
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```

Out[36]:
```
<Axes: xlabel='Embarked', ylabel='count'>
```

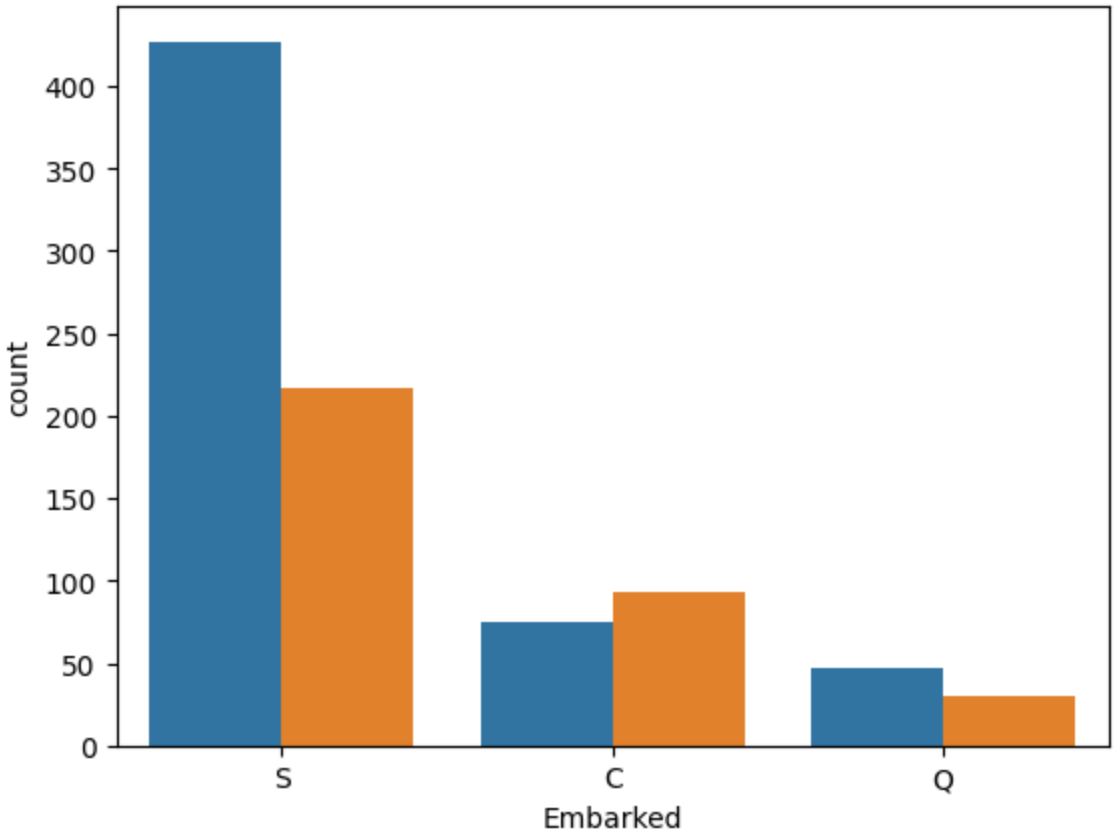


In [37]:
```
sns.countplot(x='Embarked',hue='Survived',data=data_set)
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_old
core.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed
in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[37], line 1
----> 1 sns.countplot(x='Embarked',hue='Survived',data=data_set)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:2955, in countplot(data, x, y, hue, order, hue_order, orient, color, palette, s
aturation, width, dodge, ax, **kwargs)
   2952 if ax is None:
   2953     ax = plt.gca()
-> 2955 plotter.plot(ax, kwargs)
   2956 return ax

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:1587, in _BarPlotter.plot(self, ax, bar_kws)
   1585 """Make the plot."""
   1586 self.draw_bars(ax, bar_kws)
-> 1587 self.annotate_axes(ax)
   1588 if self.orient == "h":
   1589     ax.invert_yaxis()

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\categorica
l.py:767, in _CategoricalPlotter.annotate_axes(self, ax)
    764     ax.set_ylim(-.5, len(self.plot_data) - .5, auto=None)
    766 if self.hue_names is not None:
--> 767     ax.legend(loc="best", title=self.hue_title)

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\axes\_ax
es.py:322, in Axes.legend(self, *args, **kwargs)
    204 @_docstring.dedent_interpd
    205 def legend(self, *args, **kwargs):
    206     """
    207     Place a legend on the Axes.
    208
   (...)
    320     .. plot:: gallery/text_labels_and_annotations/legend.py
    321     """
--> 322     handles, labels, kwargs = mlegend._parse_legend_args([self], *args, **kw
args)
    323     self.legend_ = mlegend.Legend(self, handles, labels, **kwargs)
    324     self.legend_._remove_method = self._remove_legend

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\legend.p
y:1361, in _parse_legend_args(axs, handles, labels, *args, **kwargs)
   1357     handles = [handle for handle, label
   1358                in zip(_get_legend_handles(axs, handlers), labels)]
   1360 elif len(args) == 0:  # 0 args: automatically detect labels and handles.
-> 1361     handles, labels = _get_legend_handles_labels(axs, handlers)
   1362     if not handles:
   1363         log.warning(
   1364             "No artists with labels found to put in legend.  Note that "
   1365             "artists whose label start with an underscore are ignored "
   1366             "when legend() is called with no argument.")

File ~\AppData\Local\Programs\Python\Python311\Lib\site-packages\matplotlib\legend.p
y:1291, in _get_legend_handles_labels(axs, legend_handler_map)
```

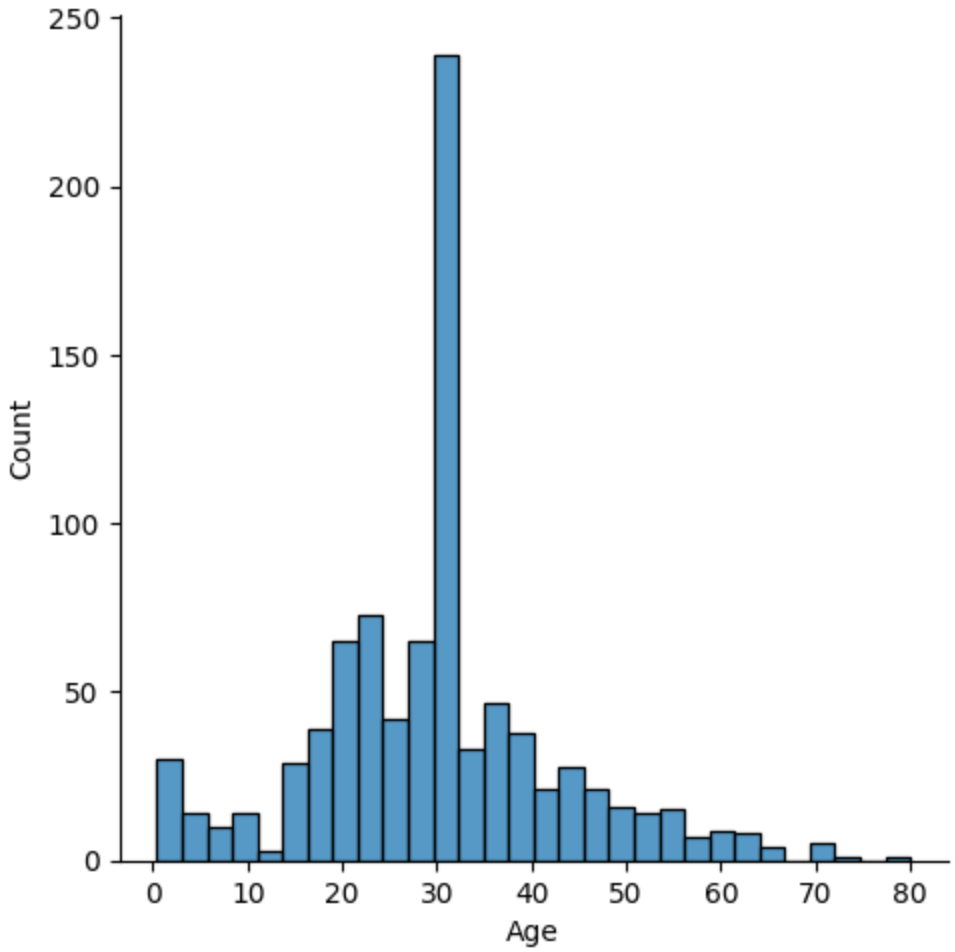
```
   1289 for handle in _get_legend_handles(axs, legend_handler_map):
   1290     label = handle.get_label()
-> 1291     if label and not label.startswith('_'):
   1292         handles.append(handle)
   1293         labels.append(label)

AttributeError: 'numpy.int64' object has no attribute 'startswith'
```



In [39]:
```
sns.displot(data_set['Age'])
```

```
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Asus\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[39]:
```
<seaborn.axisgrid.FacetGrid at 0x1c2454f6350>
```

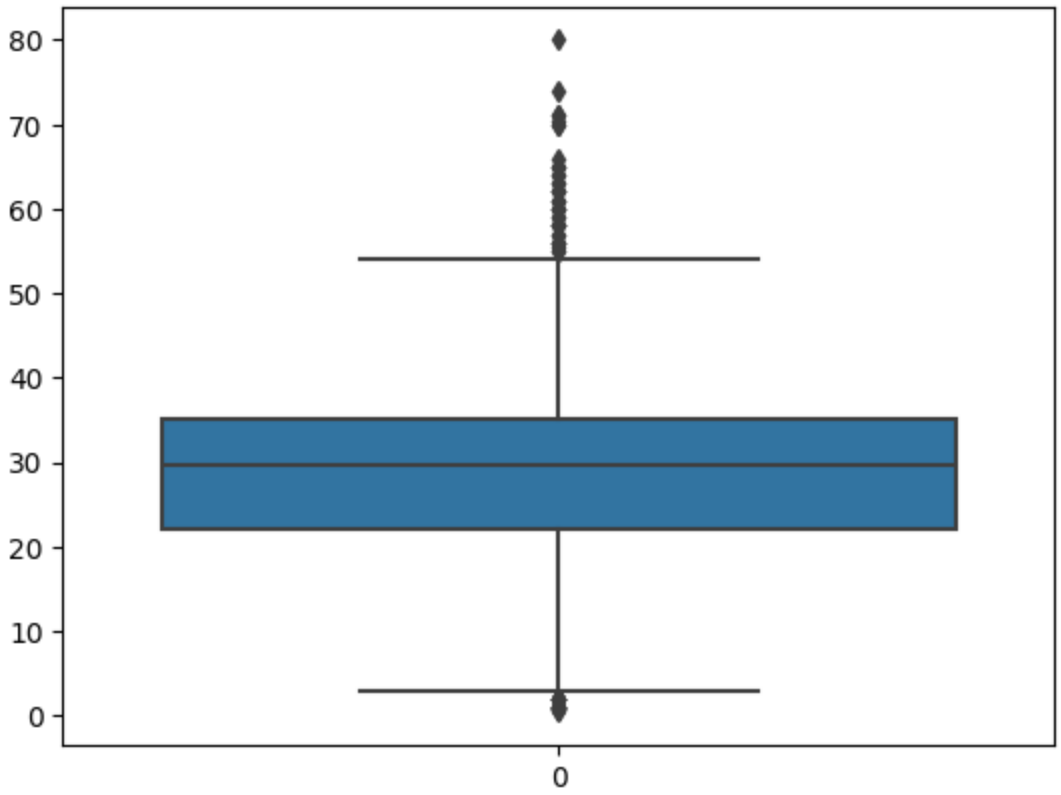


From above distribution plot we can see 1/4th of passengers were aged 30 to 33

Identifying and Addressing Outliers

In [40]:
```
sns.boxplot(data_set.Age)
```

Out[40]: <Axes: >

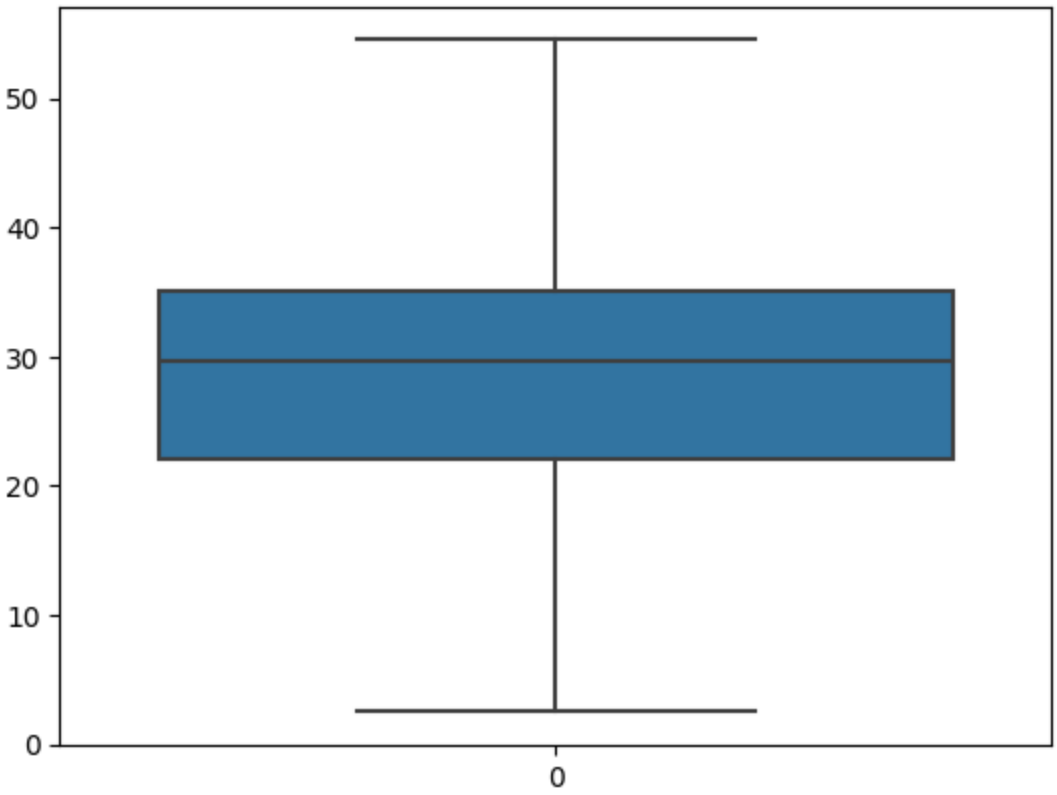


In [43]:
```
q1=data_set.Age.quantile(0.25)
q3=data_set.Age.quantile(0.75)
iqr=q3-q1
upperlimit=q3+(1.5*iqr)
lowerlimit=q1-(1.5*iqr)
```

In [46]:
```
data_set['Age']=np.where(data_set['Age']>upperlimit,upperlimit,data_set['Age'])
data_set['Age']=np.where(data_set['Age']<lowerlimit,lowerlimit,data_set['Age'])
```

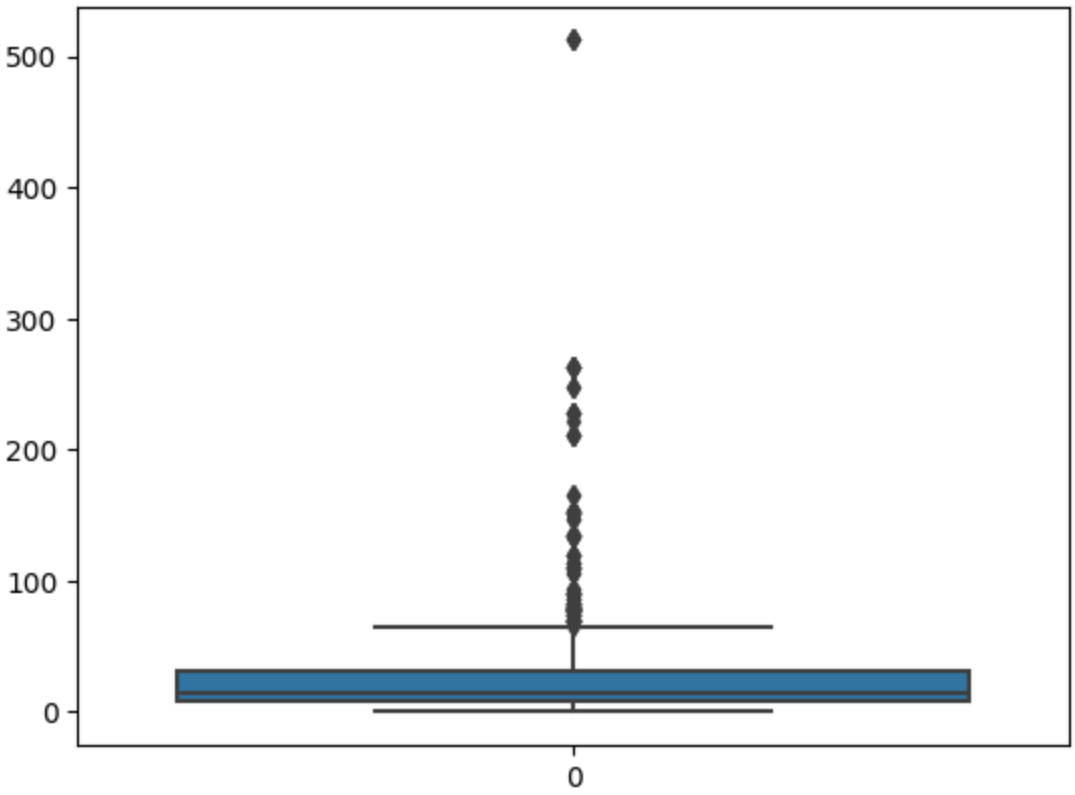
In [47]:
```
sns.boxplot(data_set['Age'])
```

Out[47]: <Axes: >



In [49]:
```
sns.boxplot(data_set['Fare'])
```

Out[49]: <Axes: >



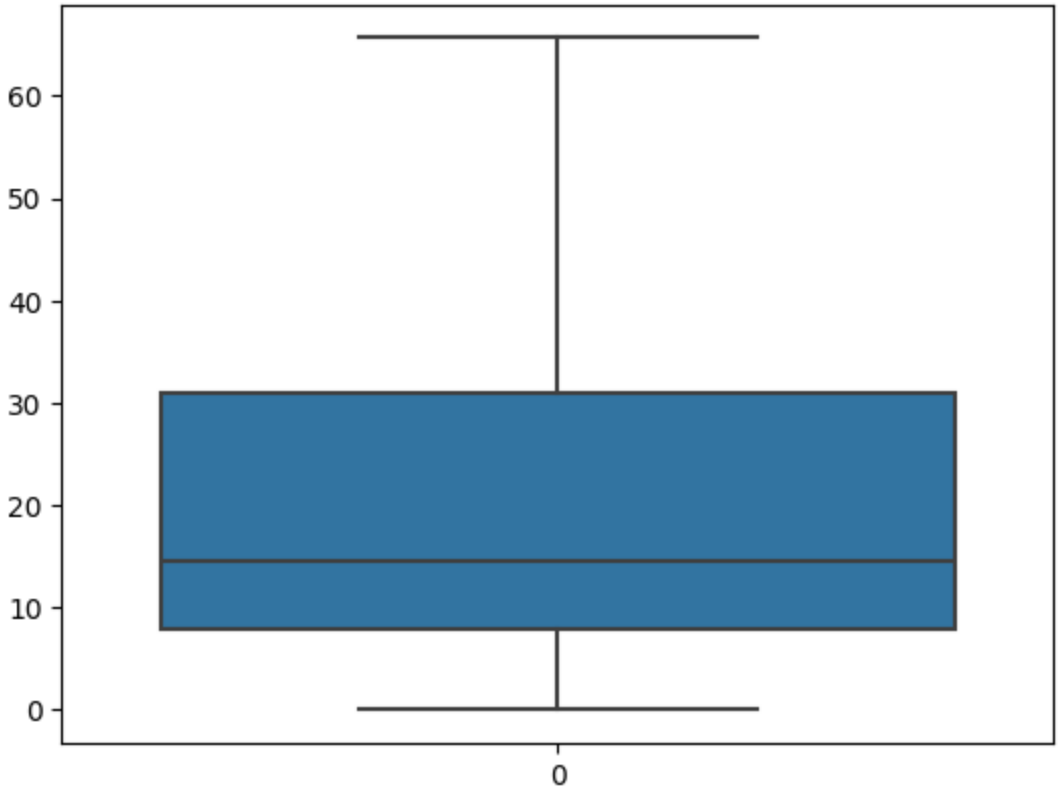
In [50]:
```
q1=data_set.Fare.quantile(0.25)
q3=data_set.Fare.quantile(0.75)
```

```
iqr=q3-q1
upperlimit=q3+(1.5*iqr)
lowerlimit=q1-(1.5*iqr)
```

In [51]:
```
data_set['Fare']=np.where(data_set['Fare']>upperlimit,upperlimit,data_set['Fare'])
data_set['Fare']=np.where(data_set['Fare']<lowerlimit,lowerlimit,data_set['Fare'])
```

In [52]:
```
sns.boxplot(data_set['Fare'])
```

Out[52]: <Axes: >



Differentiate between the independent and dependent variables

In [53]:
```
data_set.head()
```

Out[53]:

	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 Briggs Th...	female	38.0	1	0	PC 17599	65.6344
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500

In [55]:
```
x = data_set.iloc[:,2:]
x.drop(columns=["Name","Ticket"],inplace=True)
y = data_set.iloc[:,1:2]
```

Apply encoding techniques as required for data preparation.

In [57]:
```
le = LabelEncoder()
```

In [58]:
```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Pclass    891 non-null    int64  
 1   Sex       891 non-null    object 
 2   Age       891 non-null    float64
 3   SibSp     891 non-null    int64  
 4   Parch     891 non-null    int64  
 5   Fare      891 non-null    float64
 6   Embarked  889 non-null    object 
dtypes: float64(2), int64(3), object(2)
memory usage: 48.9+ KB
```

In [59]:
```
x['Sex']=le.fit_transform(x['Sex'])
```

In [60]:
```
x['Embarked']=le.fit_transform(x['Embarked'])
```

In [61]:
```
x.head()
```

Out[61]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.0	1	0	7.2500	2
1	1	0	38.0	1	0	65.6344	0
2	3	0	26.0	0	0	7.9250	2
3	1	0	35.0	1	0	53.1000	2
4	3	1	35.0	0	0	8.0500	2

Split data for Testing and Training

In [63]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [64]:
```
x_train.shape
```

Out[64]: (712, 7)

In [66]:
```
x_test.shape
```

Out[66]: (179, 7)

In [68]:
```
y_train.shape
```

Out[68]: (712, 1)

In [69]:
```
y_test.shape
```

Out[69]: (179, 1)

Feature Scaling

In [71]:
```
sc = StandardScaler()
```

In [72]:
```
x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
```

In [ ]:
```

```