Assignment 3

Data Preprocessing

UMMALETI KUMAR

kumar.21bce9309@vitapstudent.ac.in

Data Preprocessing.

```
1.      Import the Libraries.
2.      Importing the dataset.
3.      Checking for Null Values.
4.      Data Visualization.
5.      Outlier Detection
6.      Splitting Dependent and Independent variables
7.      Encoding
8.      Feature Scaling.
9.      Splitting Data into Train and Test.
```

1.Import the Libraries

In [1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2.Importing the dataset.

In [2]:
```python
df=pd.read_csv("Titanic-Dataset.csv")
```

In [3]:
```python
df.head()
```

Out[3]:

	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	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [4]:
```python
df.describe()
```

Out[4]:

	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 [5]:
```python
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
```

In [6]:
```python
df.corr(numeric_only=True)
```

Out[6]:

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

In [7]:
```python
df.corr(numeric_only=True).Fare.sort_values(ascending=False)
```

Out[7]:
```
Fare           1.000000
Survived       0.257307
Parch          0.216225
SibSp          0.159651
Age            0.096067
PassengerId    0.012658
Pclass        -0.549500
Name: Fare, dtype: float64
```

3.Checking for Null Values

In [8]:
```python
df.isnull().any()
```

Out[8]:
```
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 [9]:
```python
df.isnull().sum()
```

Out[9]:
```
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 [10]:
```python
df.Sex.nunique()
```

Out[10]: 2

In [11]:
```python
df.Sex.unique()
```

Out[11]: array(['male', 'female'], dtype=object)

In [12]:
```python
df.Sex.value_counts()
```

Out[12]:
```
Sex
male      577
female    314
Name: count, dtype: int64
```

In [13]:
```python
df.Embarked.nunique()
```

Out[13]: 3

In [14]:
```python
df.Embarked.unique()
```

Out[14]: array(['S', 'C', 'Q', nan], dtype=object)

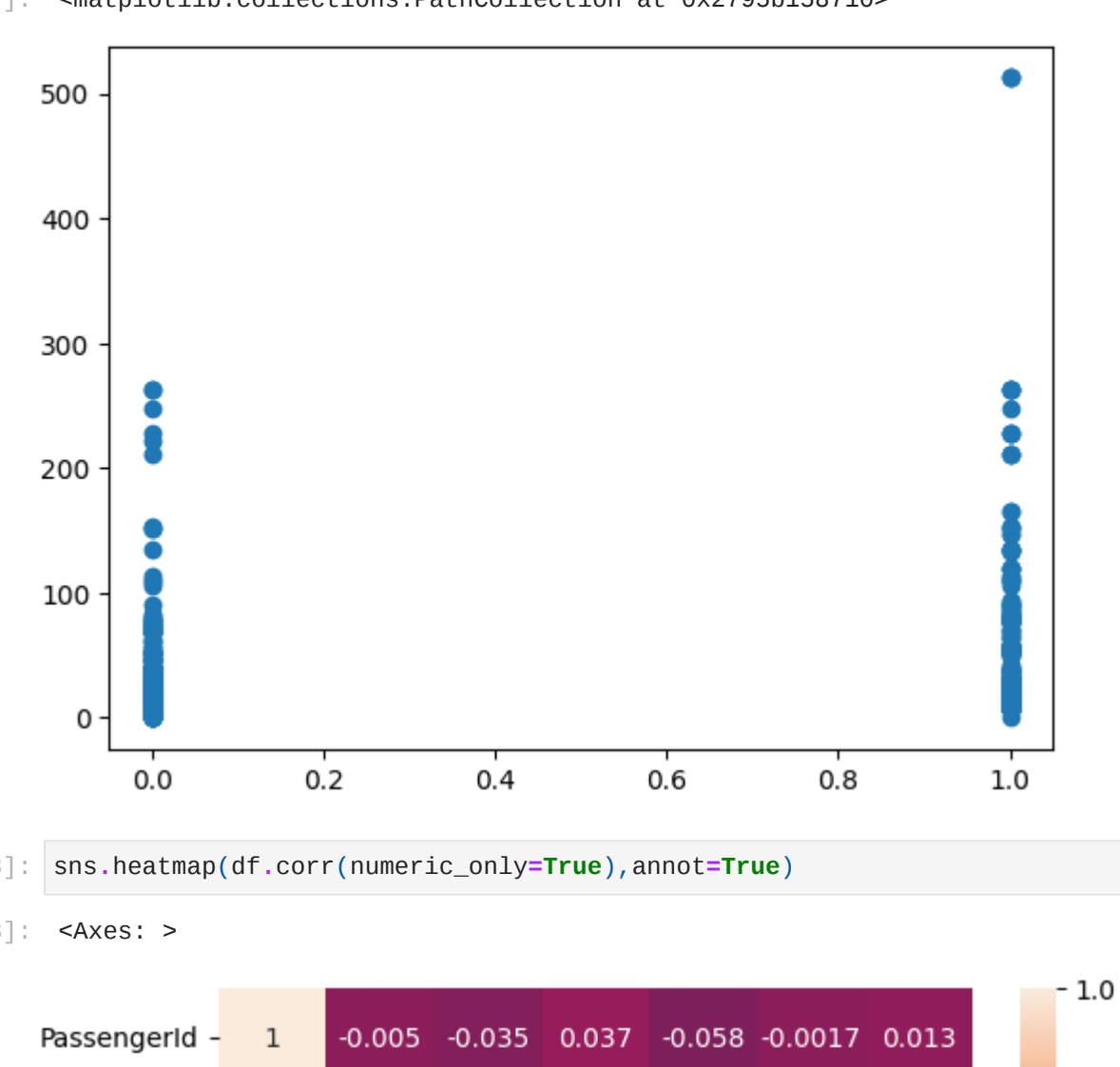
In [15]:
```python
df.Embarked.value_counts()
```

Out[15]:
```
Embarked
S    644
C    168
Q     77
Name: count, dtype: int64
```

4.Data Visualization

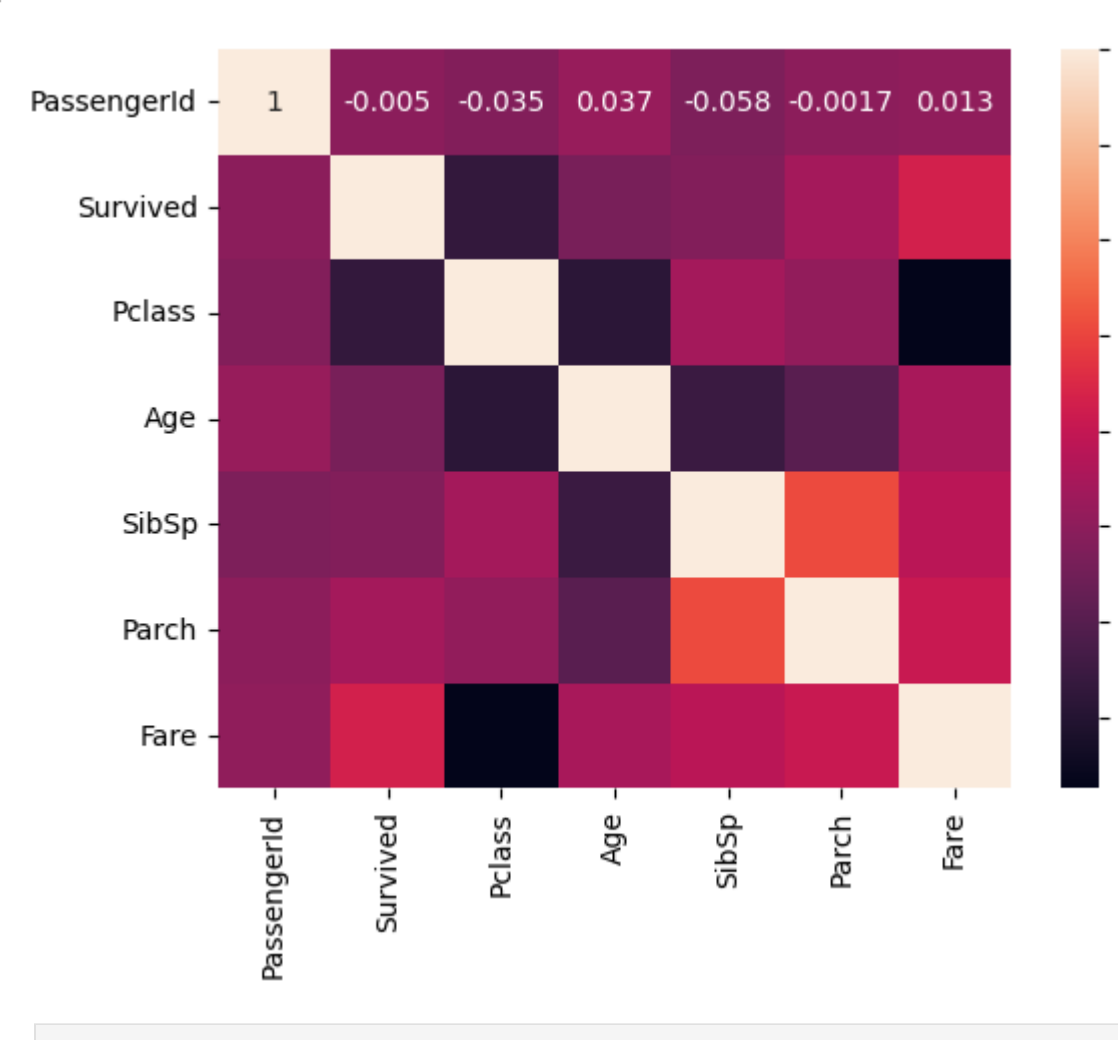
In [16]:
```python
sns.countplot(x='Survived', data=df)
plt.show()
```

```
C:\Users\kumar\AppData\Roaming\Python\Python311\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\kumar\AppData\Roaming\Python\Python311\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\kumar\AppData\Roaming\Python\Python311\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):
```



In [17]:
```python
plt.scatter(df["Survived"],df["Fare"])
```

Out[17]: <matplotlib.collections.PathCollection at 0x2795b158710>



In [18]:
```python
sns.heatmap(df.corr(numeric_only=True),annot=True)
```

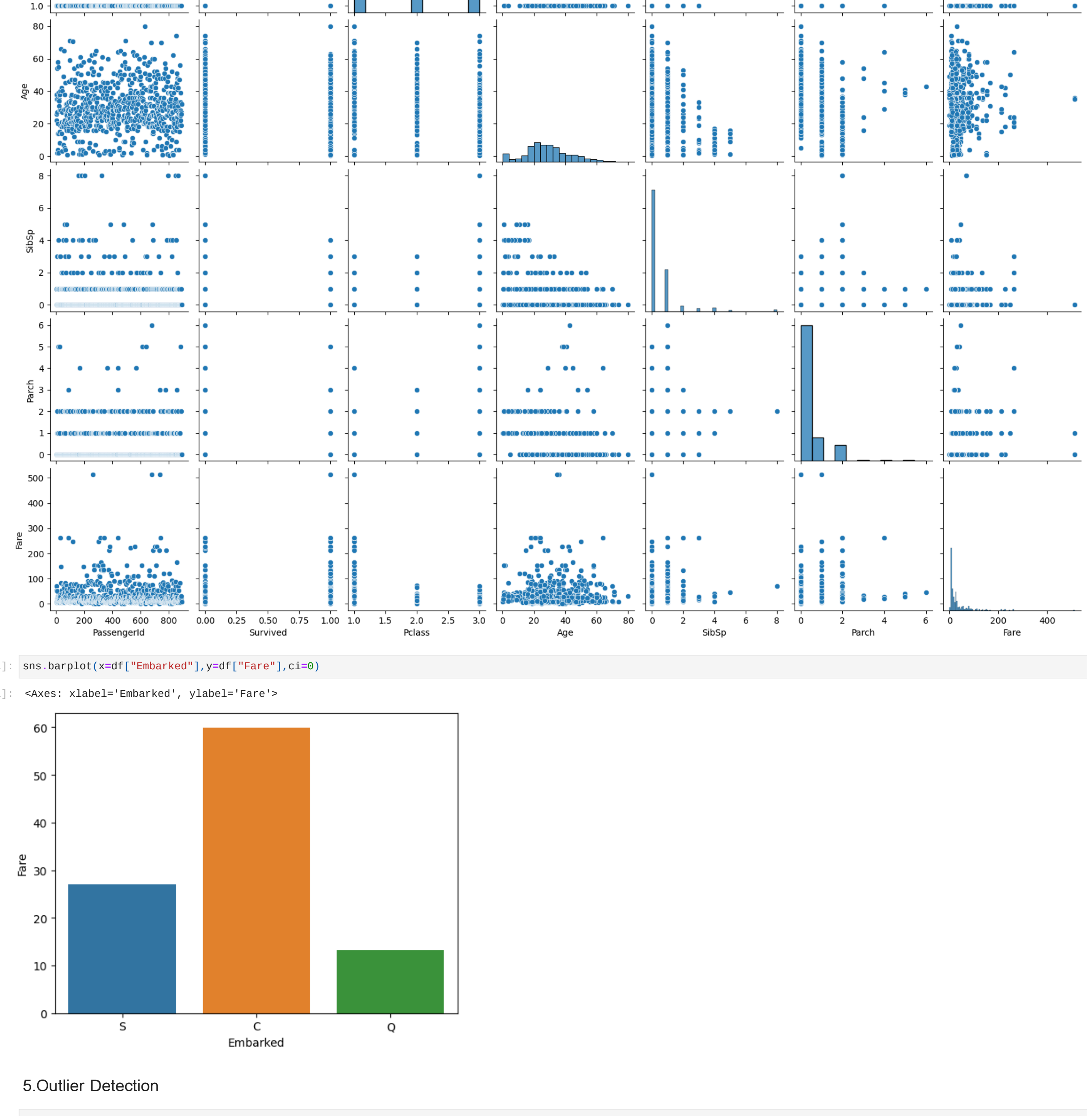
Out[18]: <Axes: >

In [19]:
```python
import warnings

# Ignore the specified warnings globally
warnings.filterwarnings("ignore", category=FutureWarning)
warnings.filterwarnings("ignore", category=UserWarning)
```

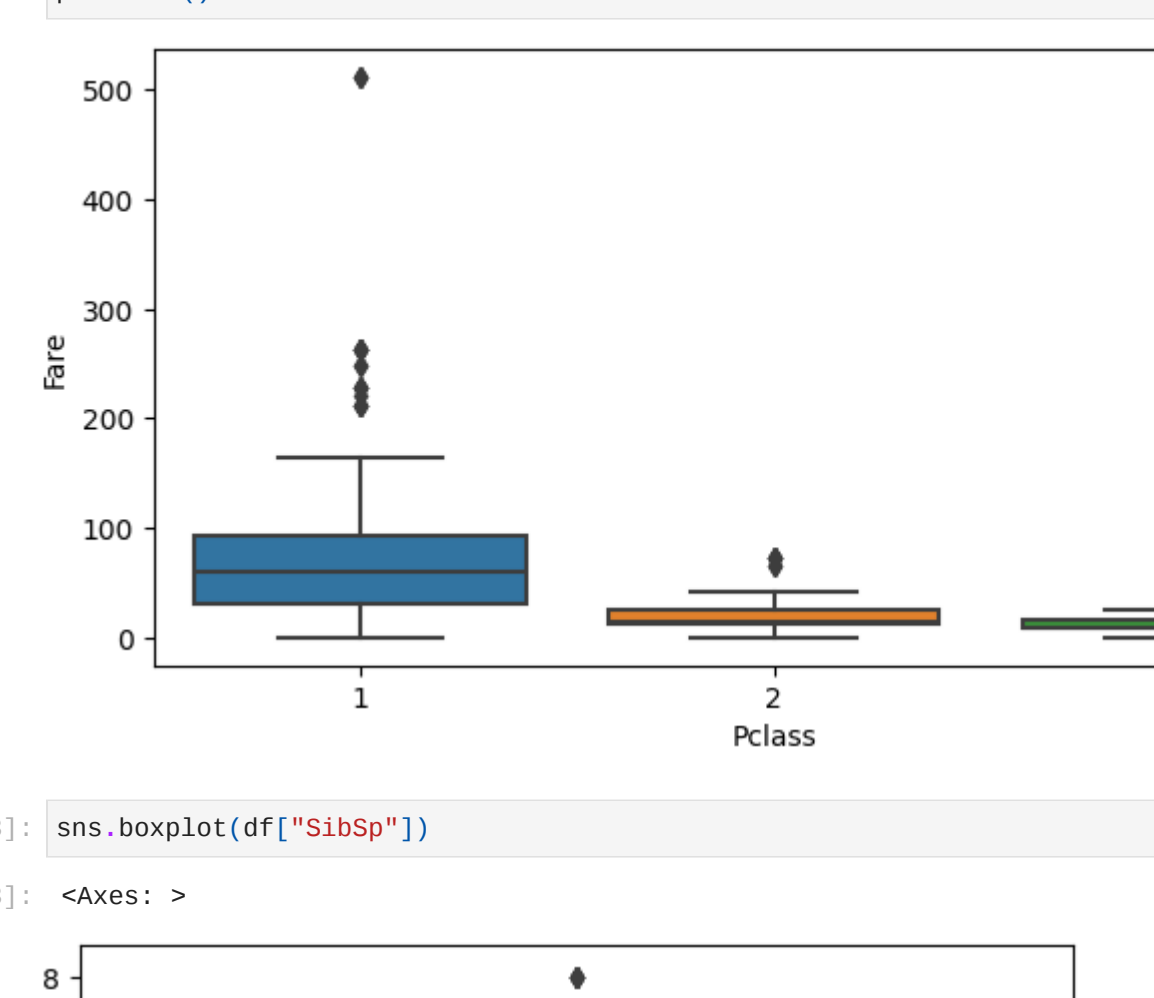
In [20]:
```python
sns.pairplot(df)
```

Out[20]: <seaborn.axisgrid.PairGrid at 0x2795b1c1d10>



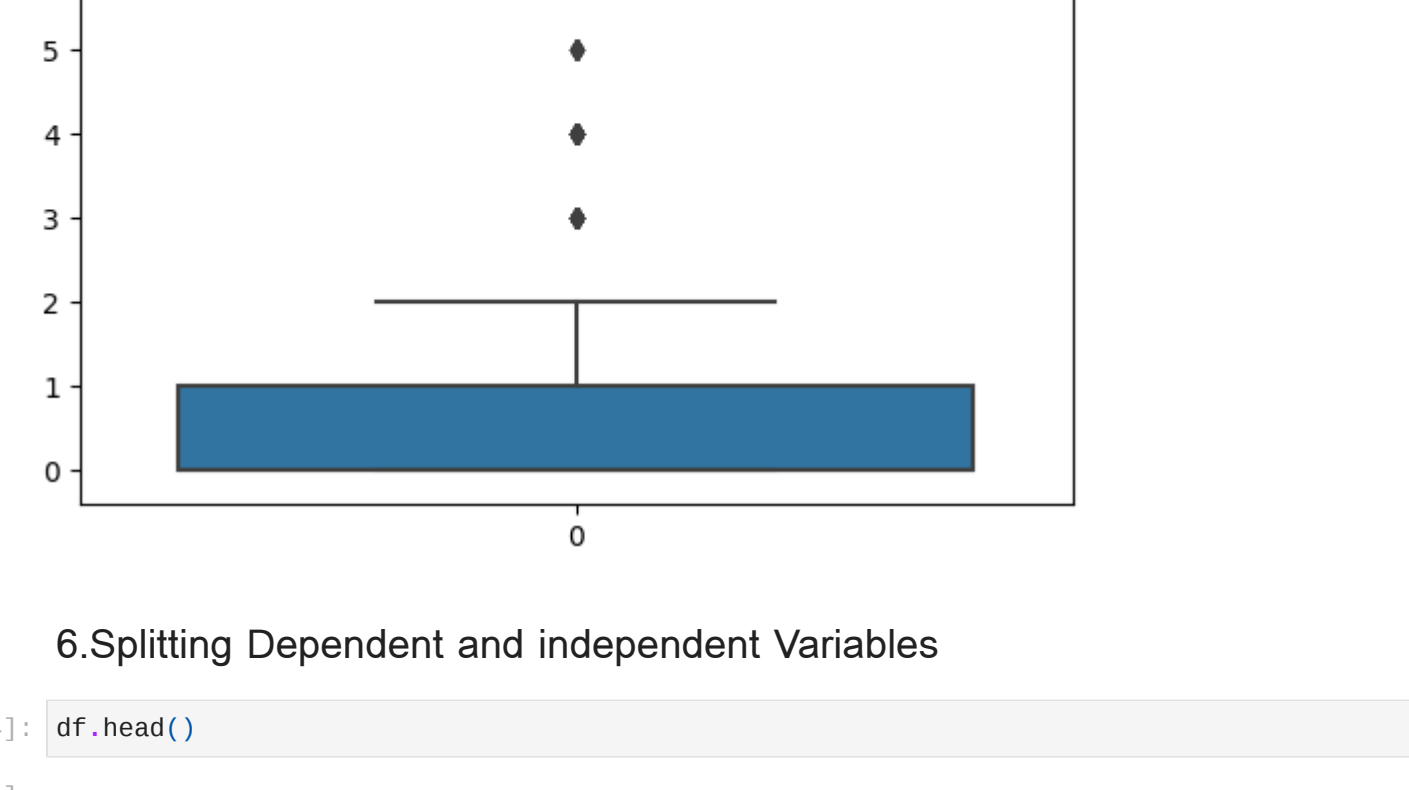
In [21]:
```python
sns.barplot(x=df["Embarked"],y=df["Fare"],ci=0)
```

Out[21]: <Axes: xlabel='Embarked', ylabel='Fare'>



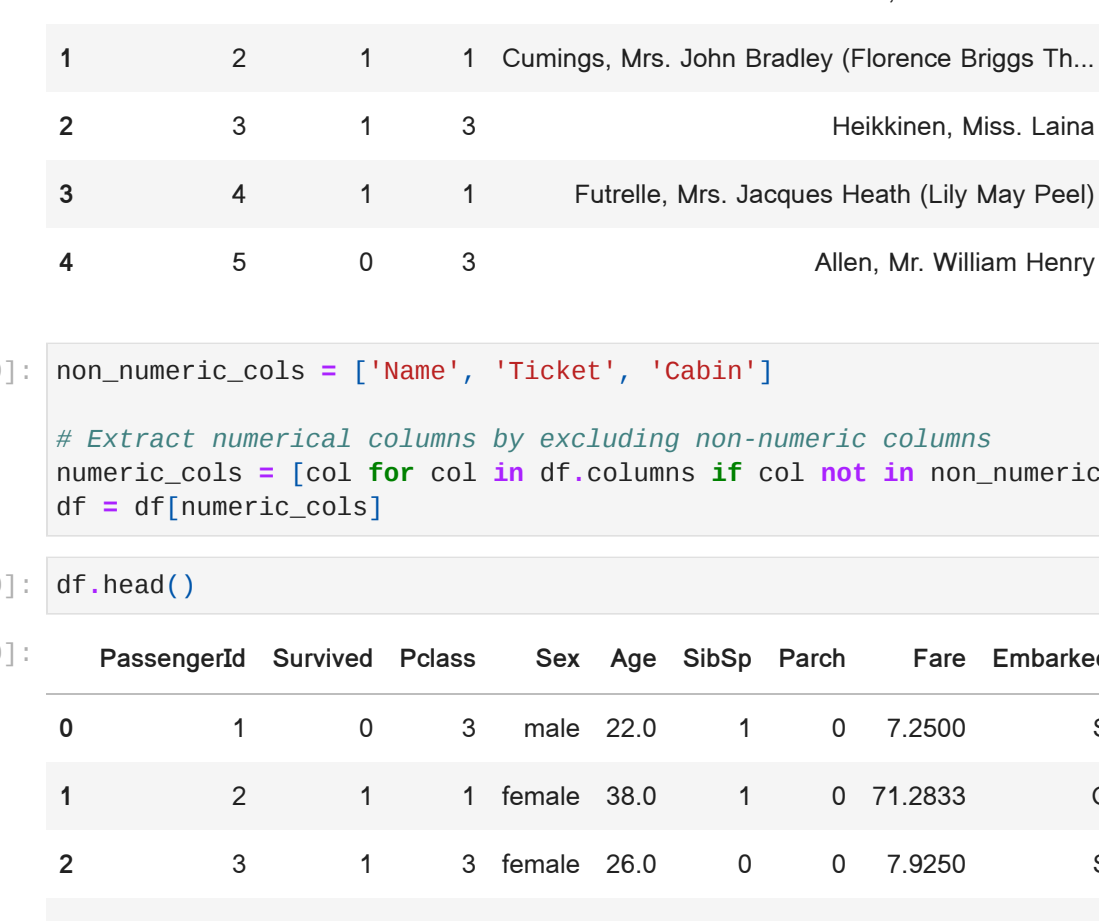
5.Outlier Detection

In [22]:
```python
plt.figure(figsize=(8, 4))
sns.boxplot(x='Pclass', y='Fare', data=df)
plt.show()
```



In [23]:
```python
sns.boxplot(df["SibSp"])
```

Out[23]: <Axes: >



6.Splitting Dependent and independent Variables

In [24]:
```python
df.head()
```

Out[24]:

	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	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [49]:
```python
non_numeric_cols = ['Name', 'Ticket', 'Cabin']

# Extract numerical columns by excluding non-numeric columns
numeric_cols = [col for col in df.columns if col not in non_numeric_cols]
df = df[numeric_cols]
```

In [50]:
```python
df.head()
```

Out[50]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	male	22.0	1	0	7.2500	S
1	2	1	1	female	38.0	1	0	71.2833	C
2	3	1	3	female	26.0	0	0	7.9250	S
3	4	1	1	female	35.0	1	0	53.1000	S
4	5	0	3	male	35.0	0	0	8.0500	S

In [51]:
```python
#indenpendent variables hould be 2 d array or dataframe
X=df.drop(columns=["Survived"],axis=1)
X.head()
```

Out[51]:

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	male	22.0	1	0	7.2500	S
1	2	1	female	38.0	1	0	71.2833	C
2	3	3	female	26.0	0	0	7.9250	S
3	4	1	female	35.0	1	0	53.1000	S
4	5	3	male	35.0	0	0	8.0500	S

In [52]:
```python
X.shape
```

Out[52]: (891, 8)

In [53]:
```python
type(X)
```

Out[53]: pandas.core.frame.DataFrame

In [54]:
```python
y=df["Survived"]
y.head()
```

Out[54]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

7.Encoding

In [55]:
```python
X.head()
```

Out[55]:

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	male	22.0	1	0	7.2500	S
1	2	1	female	38.0	1	0	71.2833	C
2	3	3	female	26.0	0	0	7.9250	S
3	4	1	female	35.0	1	0	53.1000	S
4	5	3	male	35.0	0	0	8.0500	S

In [56]:
```python
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

In [57]:
```python
X["Sex"]=le.fit_transform(X["Sex"])
```

In [58]:
```python
X.head()
```

Out[58]:

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	1	22.0	1	0	7.2500	S
1	2	1	0	38.0	1	0	71.2833	C
2	3	3	0	26.0	0	0	7.9250	S
3	4	1	0	35.0	1	0	53.1000	S
4	5	3	1	35.0	0	0	8.0500	S

In [59]:
```python
print(le.classes_)
```

['female' 'male']

In [60]:
```python
mapping=dict(zip(le.classes_,range(len(le.classes_))))
mapping
```

Out[60]: {'female': 0, 'male': 1}

In [61]:
```python
X["Embarked"]=le.fit_transform(X["Embarked"])
```

In [62]:
```python
X.head()
```

Out[62]:

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	1	22.0	1	0	7.2500	2
1	2	1	0	38.0	1	0	71.2833	0
2	3	3	0	26.0	0	0	7.9250	2
3	4	1	0	35.0	1	0	53.1000	2
4	5	3	1	35.0	0	0	8.0500	2

In [63]:
```python
print(le.classes_)
```

['C' 'Q' 'S' nan]

In [64]:
```python
mapping=dict(zip(le.classes_,range(len(le.classes_))))
mapping
```

Out[64]: {'C': 0, 'Q': 1, 'S': 2, nan: 3}

8.Feature Scaling

In [66]:
```python
from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
```

In [69]:
```python
X_scaled = ms.fit_transform(X)
```

In [70]:
```python
X_Scaled=pd.DataFrame(ms.fit_transform(X),columns=X.columns)
```

In [71]:
```python
X_Scaled.head()
```

Out[71]:

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0.000000	1.0	1.0	0.271174	0.125	0.0	0.014151	0.666667
1	0.001124	0.0	0.0	0.472229	0.125	0.0	0.139136	0.000000
2	0.002247	1.0	0.0	0.321438	0.000	0.0	0.015469	0.666667
3	0.003371	0.0	0.0	0.434531	0.125	0.0	0.103644	0.666667
4	0.004494	1.0	1.0	0.434531	0.000	0.0	0.015713	0.666667

In [73]:
```python
# Splitting Data into Train and Test
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_Scaled, y, test_size=0.2, random_state=42)

# Displaying the preprocessed data
print(X_train.head())
print(X_test.head())
print(y_train.head())
print(y_test.head())
```

```
     PassengerId  Pclass  Sex       Age  SibSp     Parch      Fare  Embarked
331     0.371910     0.0  1.0  0.566474  0.000  0.000000  0.055628  0.666667
733     0.823596     0.5  1.0  0.283740  0.000  0.000000  0.025374  0.666667
382     0.429213     1.0  1.0  0.396833  0.000  0.000000  0.015469  0.666667
704     0.791011     1.0  1.0  0.321438  0.125  0.000000  0.015330  0.666667
813     0.913483     1.0  0.0  0.070118  0.500  0.333333  0.061045  0.666667
     PassengerId  Pclass  Sex       Age  SibSp     Parch      Fare  Embarked
709     0.796629     1.0  1.0       NaN  0.125  0.166667  0.029758  0.000000
439     0.493258     0.5  1.0  0.384267  0.000  0.000000  0.020495  0.666667
840     0.943820     1.0  1.0  0.246042  0.000  0.000000  0.015469  0.666667
720     0.808989     0.5  0.0  0.070118  0.000  0.166667  0.064412  0.666667
39      0.043820     1.0  0.0  0.170646  0.125  0.000000  0.021942  0.000000
331    0
733    0
382    0
704    0
813    0
Name: Survived, dtype: int64
709    1
439    0
840    0
720    1
39     1
Name: Survived, dtype: int64
```

In [75]:
```python
print(X_train.shape,X_test.shape,y_train.shape,y_test.shape)
```

(712, 8) (179, 8) (712,) (179,)

In [65]: