Data Preprocessing on Titanic Dataset

.Data Preprocessing o Import the Libraries. o Importing the dataset. o Checking for Null Values. o Data Visualization. o Outlier Detection o Splitting Dependent and Independent variables o Perform Encoding o Feature Scaling. o Splitting Data into Train and Test

1.Import the Libraries

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

2.Importing the dataset

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

In [3]:
```
df
```

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
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

In [4]:
```
df.head()
```

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	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 [5]:
```
df.shape
```

Out[5]:
```
(891, 12)
```

In [6]:
```
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 [7]:
```
df.describe()
```

Out[7]:

	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

3.Handling Null Values.

In [8]:
```
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]:
```
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]:
```
#There are null values in Age,Cabin and Embarked columns
#handling null values in Age column--numerical type
df["Age"].fillna(df["Age"].mean(),inplace=True)
```

In [11]:
```
#handling null values in Cabin column--categorical type
df["Cabin"].fillna(df["Cabin"].mode()[0],inplace=True)
```

In [12]:
```
#handling null values in Embarked column--categorical type
df["Embarked"].fillna(df["Embarked"].mode()[0],inplace=True)
```

In [13]:
```
df.isnull().any()
```

Out[13]:
```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age            False
SibSp          False
Parch          False
Ticket         False
Fare           False
Cabin          False
Embarked       False
dtype: bool
```

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64
```

In [15]:
```
df.head()
```

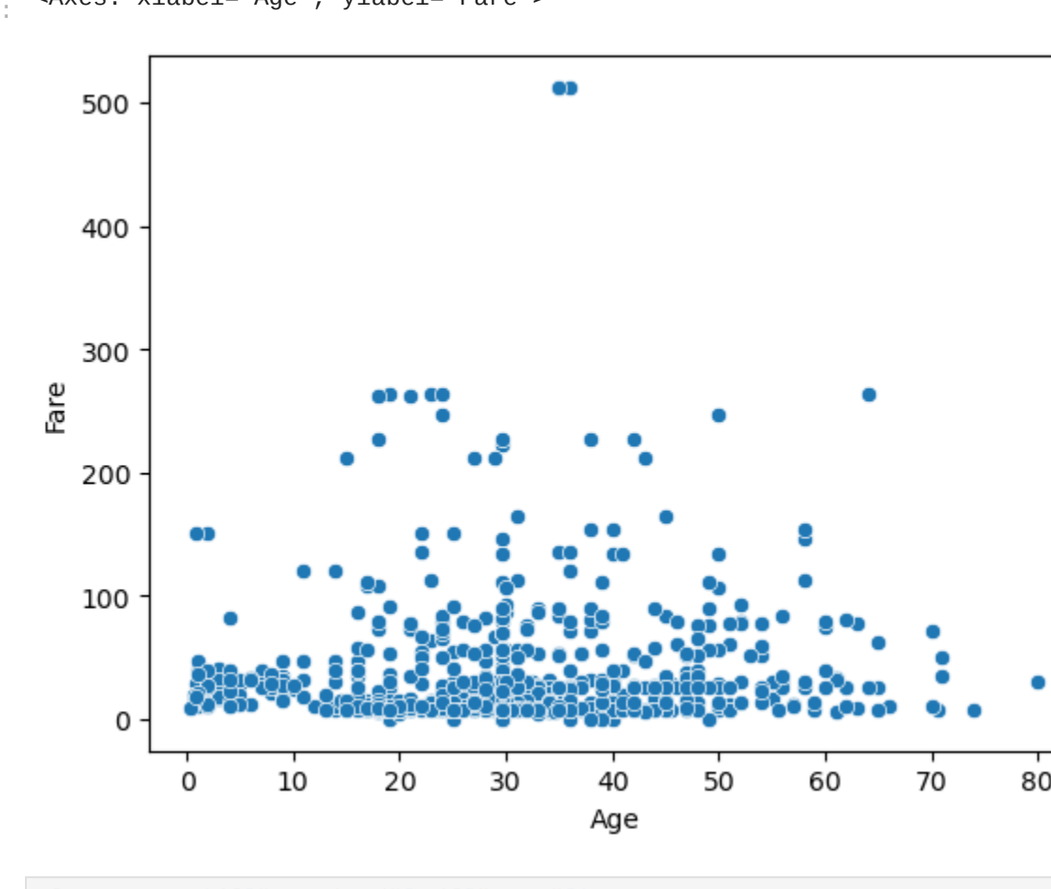
Out[15]:

	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	B96 B98	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	B96 B98	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	B96 B98	S

4.Data Visualization

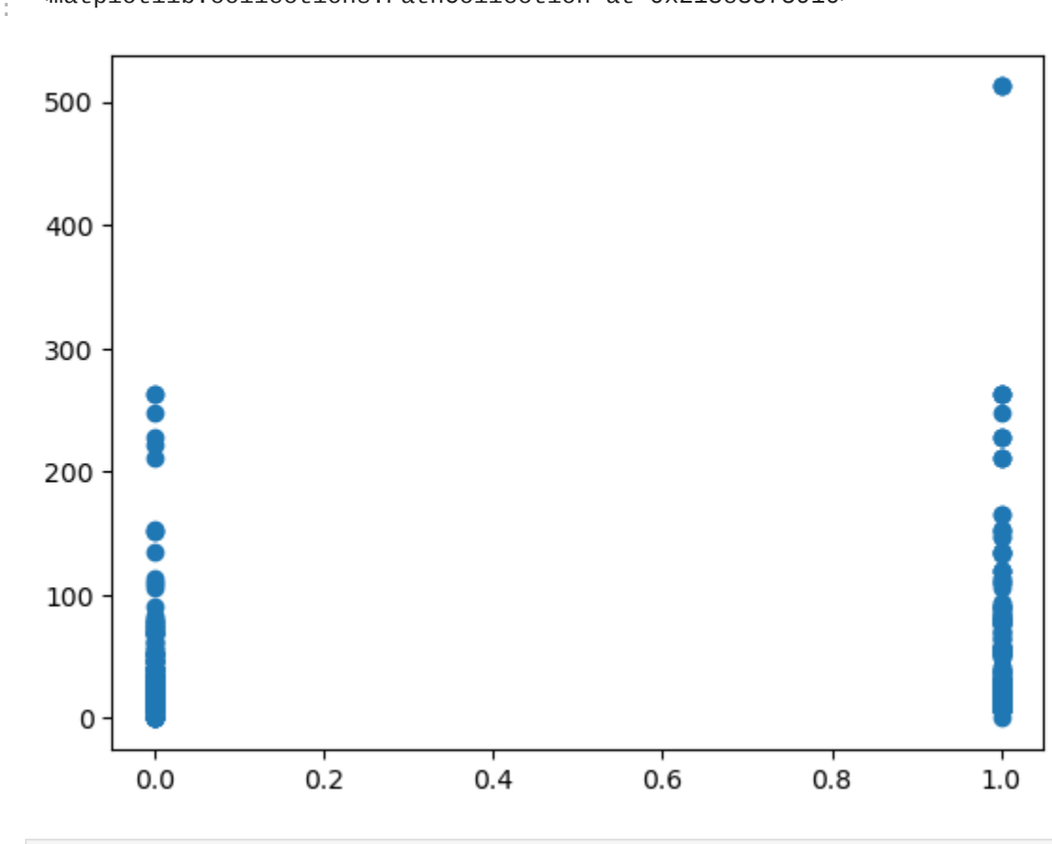
In [16]:
```
sns.scatterplot(x="Age",y="Fare",data=df)
```

Out[16]:
```
<Axes: xlabel='Age', ylabel='Fare'>
```



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

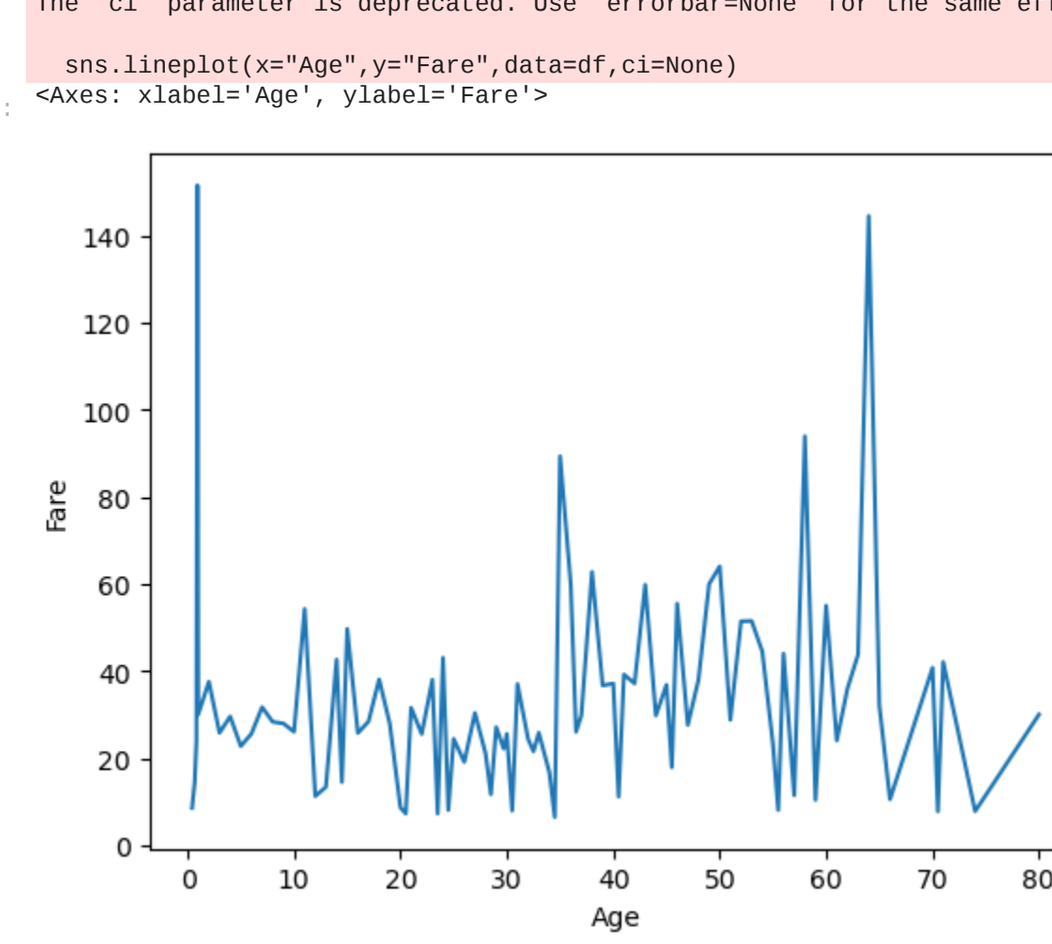
Out[17]:
```
<matplotlib.collections.PathCollection at 0x218e3875910>
```



In [18]:
```
sns.lineplot(x="Age",y="Fare",data=df,ci=None)
```

```
C:\Users\pigil\AppData\Local\Temp\ipykernel_13560\2712163228.py:1: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

  sns.lineplot(x="Age",y="Fare",data=df,ci=None)
```

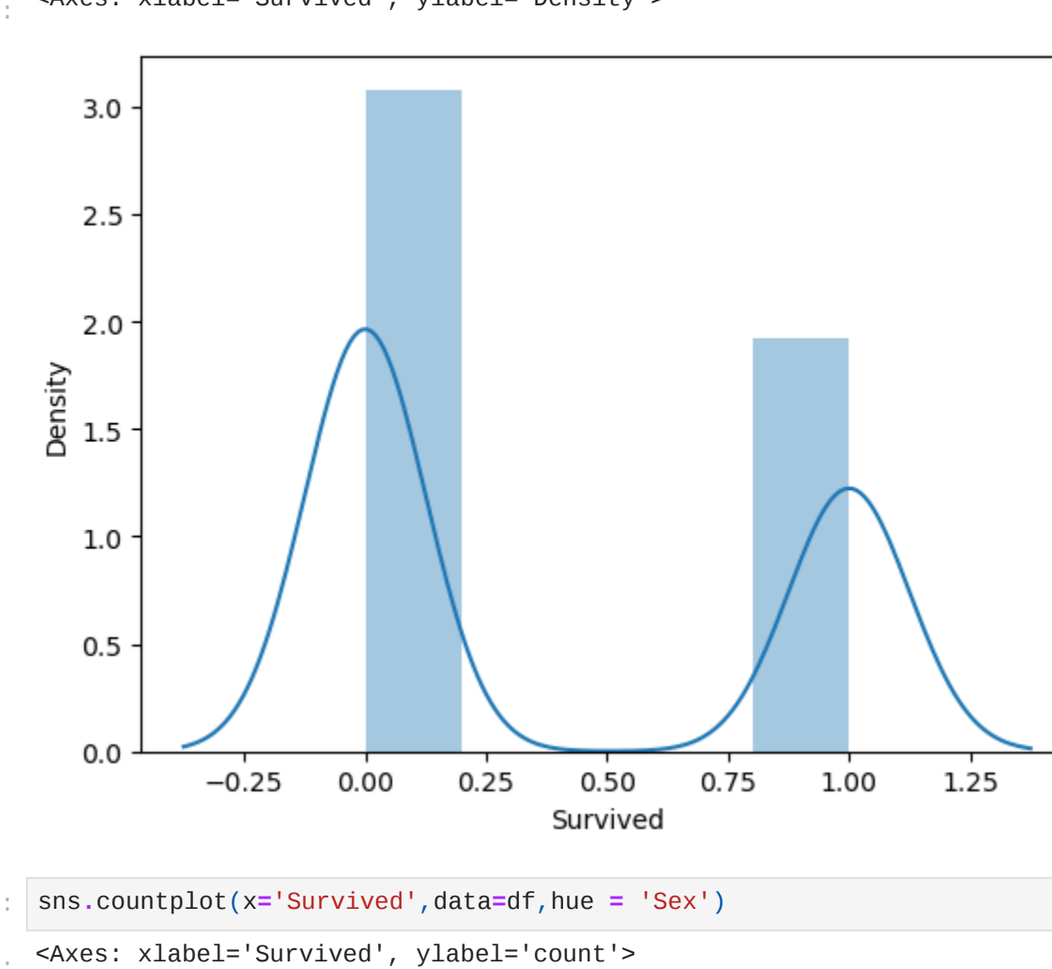
Out[18]:
```
<Axes: xlabel='Age', ylabel='Fare'>
```



In [19]:
```
sns.distplot(df["Survived"])
```

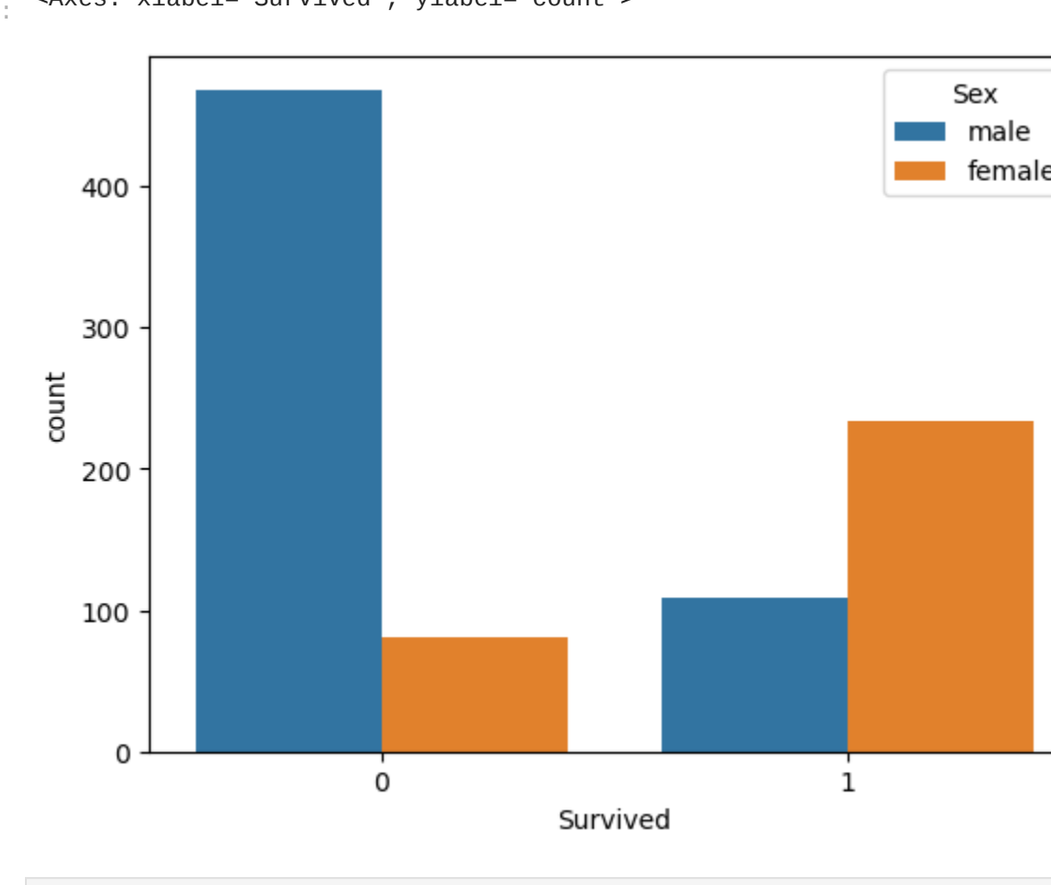
```
C:\Users\pigil\AppData\Local\Temp\ipykernel_13560\41727483.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df["Survived"])
```

Out[19]:
```
<Axes: xlabel='Survived', ylabel='Density'>
```



In [20]:
```
sns.countplot(x='Survived',data=df,hue = 'Sex')
```

Out[20]:
```
<Axes: xlabel='Survived', ylabel='count'>
```



In [22]:
```
df.corr()
```

```
C:\Users\pigil\AppData\Local\Temp\ipykernel_13560\1134722465.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.corr()
```

Out[22]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.033207	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.069809	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.331339	0.083081	0.018443	-0.549500
Age	0.033207	-0.069809	-0.331339	1.000000	-0.232625	-0.179191	0.091566
SibSp	-0.057527	-0.035322	0.083081	-0.232625	1.000000	0.414838	0.159651
Parch	-0.001652	0.081629	0.018443	-0.179191	0.414838	1.000000	0.216225
Fare	0.012658	0.257307	-0.549500	0.091566	0.159651	0.216225	1.000000

In [23]:
```
df.corr().Fare.sort_values(ascending=False)
```

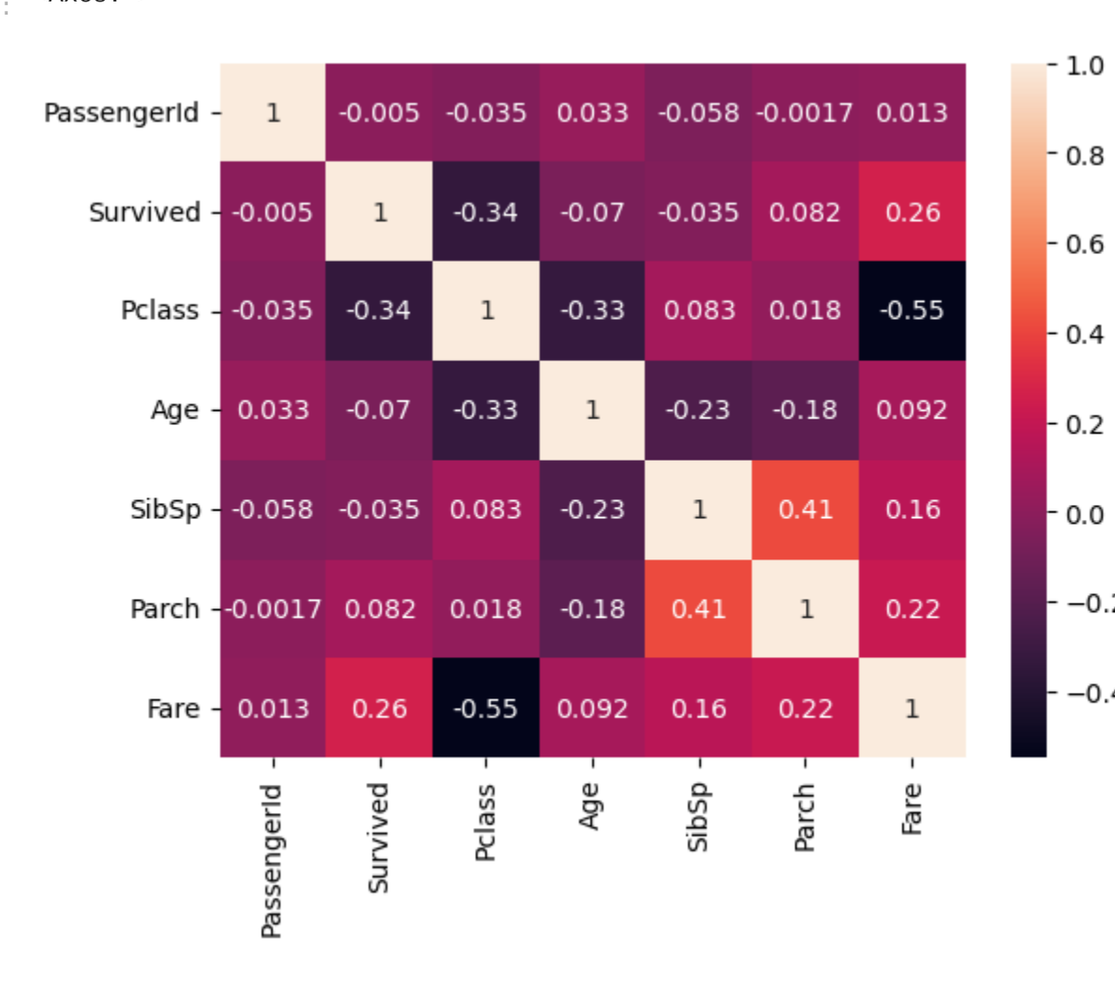
```
C:\Users\pigil\AppData\Local\Temp\ipykernel_13560\60082530.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.corr().Fare.sort_values(ascending=False)
```

Out[23]:
```
Fare           1.000000
Survived       0.257307
Parch          0.216225
SibSp          0.159651
Age            0.091566
PassengerId    0.012658
Pclass        -0.549500
Name: Fare, dtype: float64
```

In [24]:
```
sns.heatmap(df.corr(),annot=True)
```

```
C:\Users\pigil\AppData\Local\Temp\ipykernel_13560\4277794465.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(df.corr(),annot=True)
```

Out[24]:
```
<Axes: >
```



5.Outlier Detection

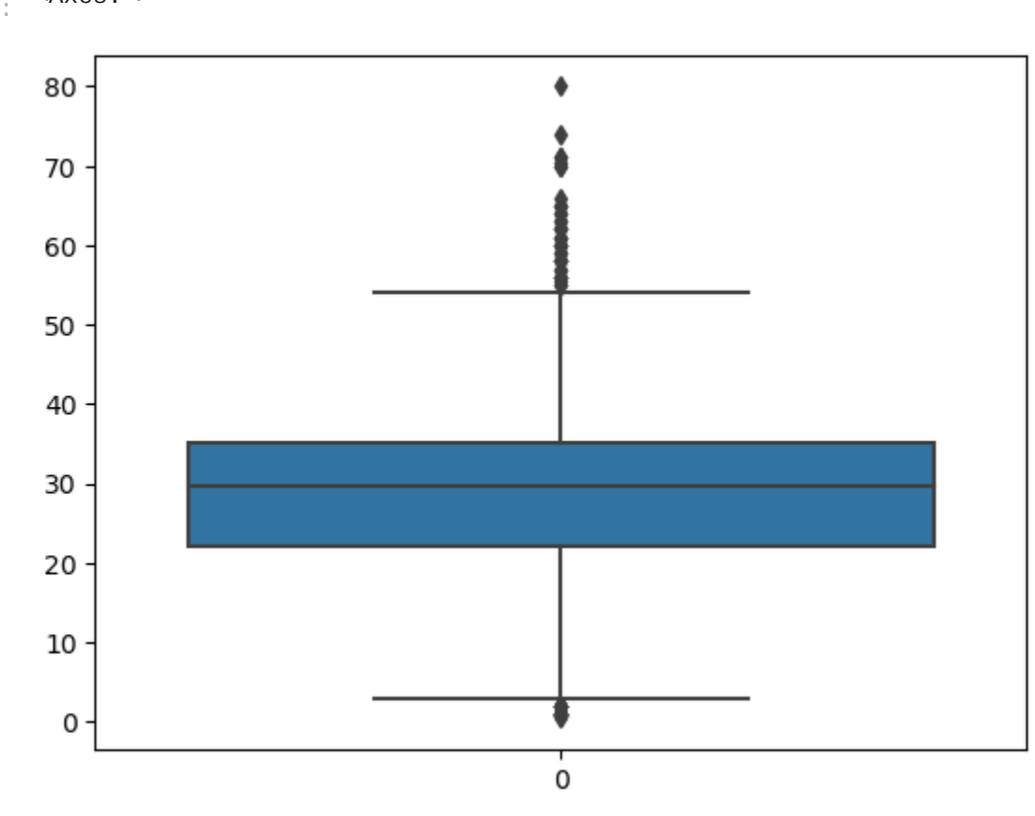
In [25]:
```
df.head()
```

Out[25]:

	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	B96 B98	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	B96 B98	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	B96 B98	S

In [26]:
```
sns.boxplot(df.Age)
```

Out[26]:
```
<Axes: >
```



In [27]:
```
q1=df.Age.quantile(0.25)
q3=df.Age.quantile(0.75)
print(q1)
print(q3)
```

```
22.0
35.0
```

In [28]:
```
IQR=q3-q1
IQR
```

Out[28]:
```
13.0
```

In [29]:
```
#upper limit
upper_limit=q3+1.5*IQR
upper_limit
```

Out[29]:
```
54.5
```

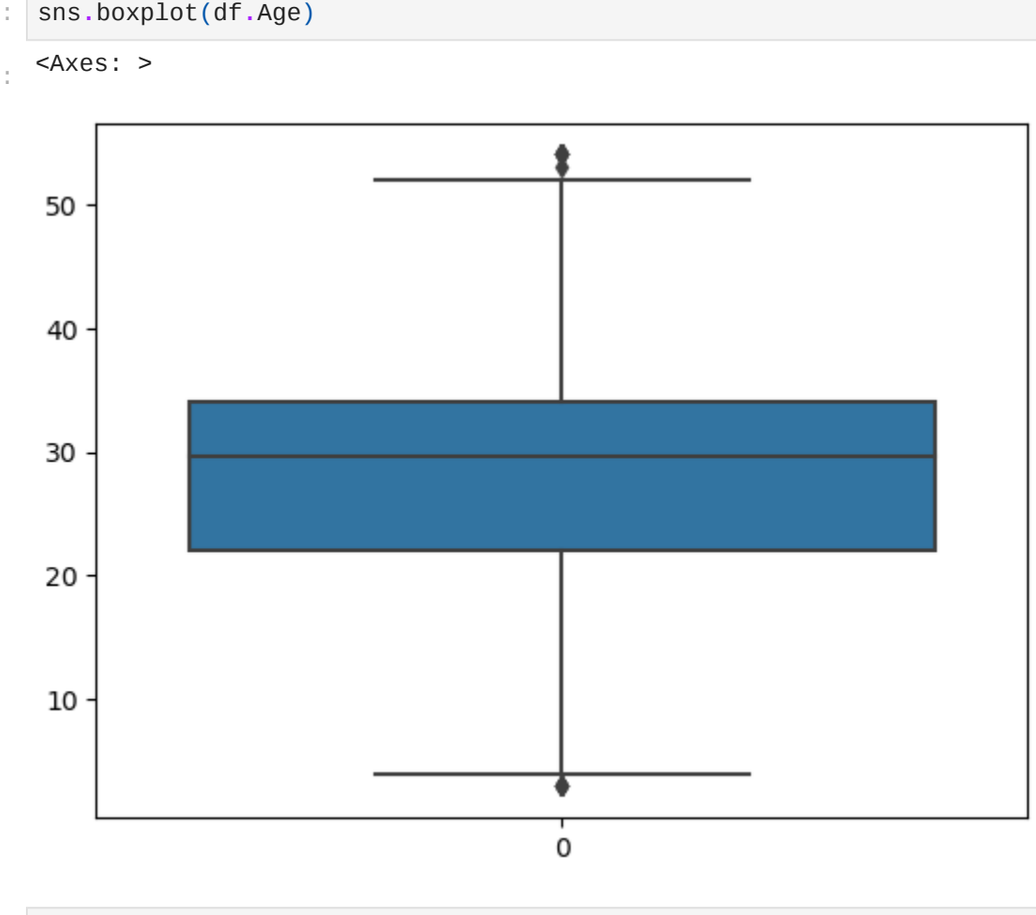
In [30]:
```
#lower limit
lower_limit=q1-1.5*IQR
lower_limit
```

Out[30]:
```
2.5
```

In [31]:
```
df=df[(df.Age>lower_limit)&(df.Age<upper_limit)]
```

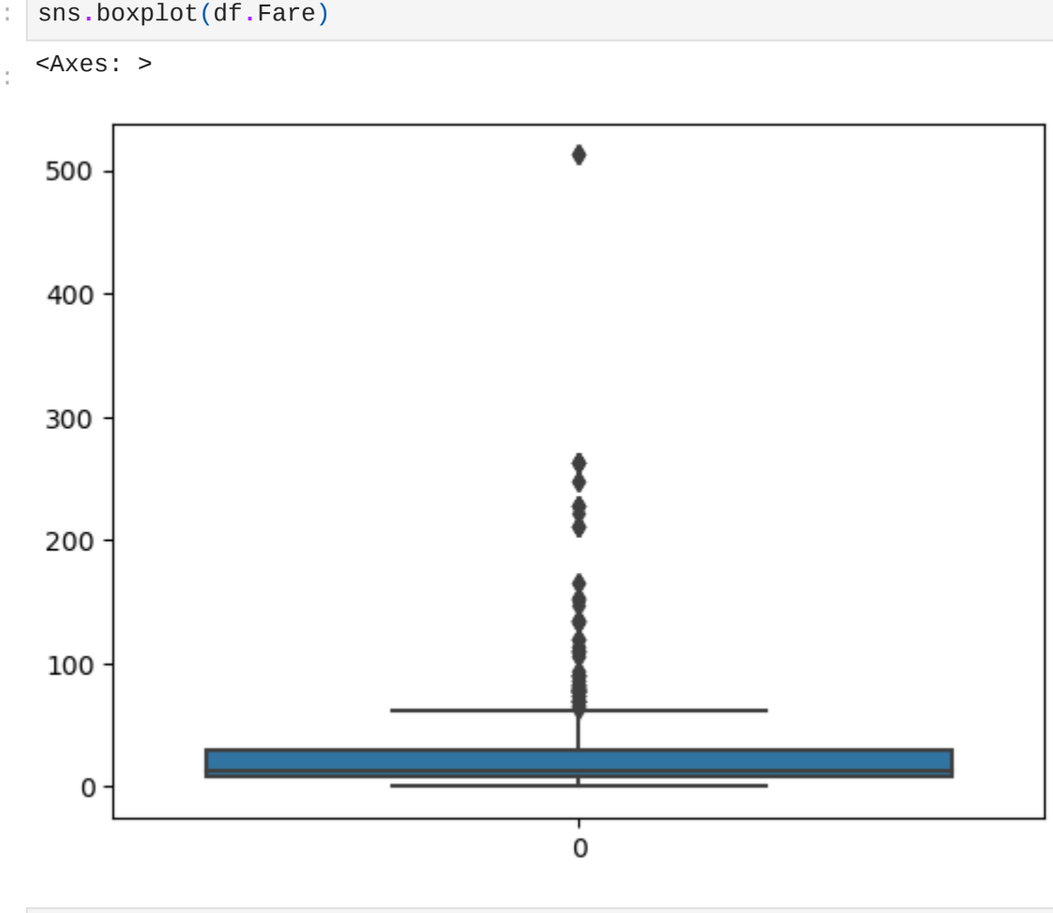
In [32]:
```
sns.boxplot(df.Age)
```

Out[32]:
```
<Axes: >
```



In [33]:
```
sns.boxplot(df.Fare)
```

Out[33]:
```
<Axes: >
```



In [34]:
```
q1=df.Fare.quantile(0.25)
q3=df.Fare.quantile(0.75)
print(q1)
print(q3)
```

```
7.8958
30.0708
```

In [35]:
```
IQR=q3-q1
IQR
```

Out[35]:
```
22.174999999999997
```

In [36]:
```
#upper limit
upper_limit=q3+1.5*IQR
upper_limit
```

Out[36]:
```
63.333299999999994
```

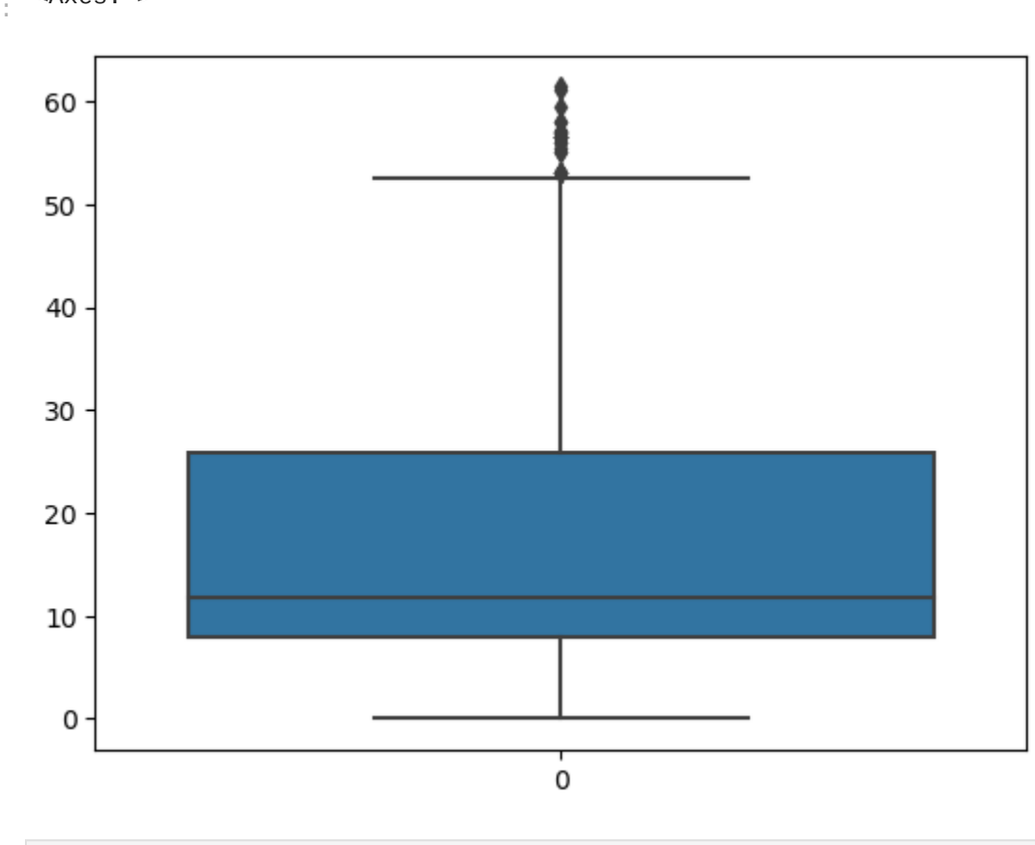
In [37]:
```
#lower limit
lower_limit=q1-1.5*IQR
lower_limit
```

Out[37]:
```
-25.366699999999994
```

In [38]:
```
df=df[(df.Fare>lower_limit)&(df.Fare<upper_limit)]
```

In [39]:
```
sns.boxplot(df.Fare)
```

Out[39]:
```
<Axes: >
```



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

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 718 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  718 non-null    int64
 1   Survived     718 non-null    int64
 2   Pclass       718 non-null    int64
 3   Name         718 non-null    object
 4   Sex          718 non-null    object
 5   Age          718 non-null    float64
 6   SibSp        718 non-null    int64
 7   Parch        718 non-null    int64
 8   Ticket       718 non-null    object
 9   Fare         718 non-null    float64
 10  Cabin        718 non-null    object
 11  Embarked     718 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 89.1+ KB
```

6.Splitting Dependent and independent Variables

In [41]:
```
df.head()
```

Out[41]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	B96 B98	S
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	B96 B98	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.000000	0	0	373450	8.0500	B96 B98	S
5	6	0	3	Moran, Mr. James	male	29.699118	0	0	330877	8.4583	B96 B98	Q

In [42]:
```
df.drop(['Name','Ticket'],axis=1,inplace=True)
```

In [43]:
```
df.head()
```

Out[43]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked
0	1	0	3	male	22.000000	1	0	7.2500	B96 B98	S
2	3	1	3	female	26.000000	0	0	7.9250	B96 B98	S
3	4	1	1	female	35.000000	1	0	53.1000	C123	S
4	5	0	3	male	35.000000	0	0	8.0500	B96 B98	S
5	6	0	3	male	29.699118	0	0	8.4583	B96 B98	Q

In [45]:
```
df.drop('Cabin',axis=1,inplace=True)
```

In [46]:
```
df.head()
```

Out[46]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	male	22.000000	1	0	7.2500	S
2	3	1	3	female	26.000000	0	0	7.9250	S
3	4	1	1	female	35.000000	1	0	53.1000	S
4	5	0	3	male	35.000000	0	0	8.0500	S
5	6	0	3	male	29.699118	0	0	8.4583	Q

In [48]:
```
#datset.iloc[rows,column]
x=df.iloc[:,2:]
y=df.iloc[:,1:2]
```

In [49]:
```
x.head()
```

Out[49]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	22.000000	1	0	7.2500	S
2	3	female	26.000000	0	0	7.9250	S
3	1	female	35.000000	1	0	53.1000	S
4	3	male	35.000000	0	0	8.0500	S
5	3	male	29.699118	0	0	8.4583	Q

In [50]:
```
y.head()
```

Out[50]:

	Survived
0	0
2	1
3	1
4	0
5	0

In [51]:
```
df.shape
```

Out[51]:
```
(718, 9)
```

In [52]:
```
x.shape
```

Out[52]:
```
(718, 7)
```

In [53]:
```
y.shape
```

Out[53]:
```
(718, 1)
```

7.Encoding

In [54]:
```
from sklearn.preprocessing import LabelEncoder
```

In [55]:
```
le=LabelEncoder()
```

In [56]:
```
x["Sex"]=le.fit_transform(x["Sex"])
```

In [57]:
```
x["Sex"]
```

Out[57]:
```
0      1
2      0
3      0
4      1
5      1
      ..
886    1
887    0
888    0
889    1
890    1
Name: Sex, Length: 718, dtype: int32
```

In [58]:
```
x["Sex"].value_counts()
```

Out[58]:
```
1    489
0    229
Name: Sex, dtype: int64
```

In [59]:
```
x.head()
```

Out[59]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.000000	1	0	7.2500	S
2	3	0	26.000000	0	0	7.9250	S
3	1	0	35.000000	1	0	53.1000	S
4	3	1	35.000000	0	0	8.0500	S
5	3	1	29.699118	0	0	8.4583	Q

In [62]:
```
x["Embarked"]=le.fit_transform(x["Embarked"])
```

In [63]:
```
x["Embarked"]
```

Out[63]:
```
0      2
2      2
3      2
4      2
5      1
      ..
886    2
887    2
888    2
889    0
890    1
Name: Embarked, Length: 718, dtype: int32
```

In [64]:
```
x["Embarked"].value_counts()
```

Out[64]:
```
2    543
0    104
1     71
Name: Embarked, dtype: int64
```

In [71]:
```
print(le.classes_)
```

```
['C' 'Q' 'S']
```

In [70]:
```
print(dict(zip(le.classes_,range(len(le.classes_)))))
```

```
{'C': 0, 'Q': 1, 'S': 2}
```

In [65]:
```
x.head()
```

Out[65]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	1	22.000000	1	0	7.2500	2
2	3	0	26.000000	0	0	7.9250	2
3	1	0	35.000000	1	0	53.1000	2
4	3	1	35.000000	0	0	8.0500	2
5	3	1	29.699118	0	0	8.4583	1

8.Feature Scaling

In [76]:
```
from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
```

In [77]:
```
x_scaled=pd.DataFrame(ms.fit_transform(x),columns=x.columns)
x_scaled.head()
```

Out[77]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1.0	1.0	0.372549	0.2	0.0	0.118118	1.0
1	1.0	0.0	0.450980	0.0	0.0	0.129115	1.0
2	0.0	0.0	0.627451	0.2	0.0	0.865114	1.0
3	1.0	1.0	0.627451	0.0	0.0	0.131152	1.0
4	1.0	1.0	0.523512	0.0	0.0	0.137804	0.5

9.Splitting the data into train and test

In [78]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [79]:
```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

Out[79]:
```
((574, 7), (144, 7), (574, 1), (144, 1))
```

In [80]:
```
x_train.head()
```

Out[80]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
593	3	0	29.699118	0	2	7.7500	1
289	3	0	22.000000	0	0	7.7500	1
22	3	0	15.000000	0	0	8.0292	1
338	3	1	45.000000	0	0	8.0500	2
451	3	1	29.699118	1	0	19.9667	2

In [81]:
```
y_train.head()
```

Out[81]:

	Survived
593	0
289	1
22	1
338	1
451	0

In [ ]:

In [ ]:

In [ ]: