In [3]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
```

In [4]:
```
data= pd.read_csv('/home/placement/Desktop/TitanicDataset.csv')
data
```

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
...	...	...	...	...	...	...	...	...	...	...	...	...
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 [5]:
```
data.describe()
```

Out[5]:

	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 [6]:
```
data.columns
```

Out[6]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

```
In [7]: data.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 [8]:
```
data.head(10)
```

Out[8]:

	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
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C

In [9]:
```
data.shape
```

Out[9]:
```
(891, 12)
```

In [10]:
```
data.isna().sum()
```

Out[10]:
```
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 [11]:
```
data['Pclass'].unique()
```

Out[11]:
```
array([3, 1, 2])
```

In [12]:
```
data['Survived'].unique()
```

Out[12]:
```
array([0, 1])
```

In [13]:
```
data['SibSp'].unique()
```

Out[13]:
```
array([1, 0, 3, 4, 2, 5, 8])
```

In [14]:
```
data['Age'].unique()
```

Out[14]:
```
array([22.  , 38.  , 26.  , 35.  ,   nan, 54.  ,  2.  , 27.  , 14.  ,
        4.  , 58.  , 20.  , 39.  , 55.  , 31.  , 34.  , 15.  , 28.  ,
        8.  , 19.  , 40.  , 66.  , 42.  , 21.  , 18.  ,  3.  ,  7.  ,
       49.  , 29.  , 65.  , 28.5 ,  5.  , 11.  , 45.  , 17.  , 32.  ,
       16.  , 25.  ,  0.83, 30.  , 33.  , 23.  , 24.  , 46.  , 59.  ,
       71.  , 37.  , 47.  , 14.5 , 70.5 , 32.5 , 12.  ,  9.  , 36.5 ,
       51.  , 55.5 , 40.5 , 44.  ,  1.  , 61.  , 56.  , 50.  , 36.  ,
       45.5 , 20.5 , 62.  , 41.  , 52.  , 63.  , 23.5 ,  0.92, 43.  ,
       60.  , 10.  , 64.  , 13.  , 48.  ,  0.75, 53.  , 57.  , 80.  ,
       70.  , 24.5 ,  6.  ,  0.67, 30.5 ,  0.42, 34.5 , 74.  ])
```

In [15]:
```
data1=data.drop(columns=['PassengerId','Ticket','Cabin','Fare','SibSp','Name'])#remove unwanted columns
```

In [16]:
```
data1
```

Out[16]:

	Survived	Pclass	Sex	Age	Parch	Embarked
0	0	3	male	22.0	0	S
1	1	1	female	38.0	0	C
2	1	3	female	26.0	0	S
3	1	1	female	35.0	0	S
4	0	3	male	35.0	0	S
...	...	...	...	...	...	...
886	0	2	male	27.0	0	S
887	1	1	female	19.0	0	S
888	0	3	female	NaN	2	S
889	1	1	male	26.0	0	C
890	0	3	male	32.0	0	Q

891 rows × 6 columns

In [17]:
```
data2=pd.get_dummies(data1) #convert the strings into integers for each column using getdummies()
```

In [18]:
```
data2
```

Out[18]:

	Survived	Pclass	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	0	3	22.0	0	0	1	0	0	1
1	1	1	38.0	0	1	0	1	0	0
2	1	3	26.0	0	1	0	0	0	1
3	1	1	35.0	0	1	0	0	0	1
4	0	3	35.0	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...
886	0	2	27.0	0	0	1	0	0	1
887	1	1	19.0	0	1	0	0	0	1
888	0	3	NaN	2	1	0	0	0	1
889	1	1	26.0	0	0	1	1	0	0
890	0	3	32.0	0	0	1	0	1	0

891 rows × 9 columns

In [19]:
```
data2=data2.fillna(data1.median()) #fill the null values with median of data
```

In [20]:
```
data2
```

Out[20]:

	Survived	Pclass	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	0	3	22.0	0	0	1	0	0	1
1	1	1	38.0	0	1	0	1	0	0
2	1	3	26.0	0	1	0	0	0	1
3	1	1	35.0	0	1	0	0	0	1
4	0	3	35.0	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...
886	0	2	27.0	0	0	1	0	0	1
887	1	1	19.0	0	1	0	0	0	1
888	0	3	28.0	2	1	0	0	0	1
889	1	1	26.0	0	0	1	1	0	0
890	0	3	32.0	0	0	1	0	1	0

891 rows × 9 columns

In [21]:
```
data2['Pclass']=data2['Pclass'].map({1:'First',2:'Second',3:'Third'})
```

In [22]:
```
data2
```

Out[22]:

	Survived	Pclass	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	0	Third	22.0	0	0	1	0	0	1
1	1	First	38.0	0	1	0	1	0	0
2	1	Third	26.0	0	1	0	0	0	1
3	1	First	35.0	0	1	0	0	0	1
4	0	Third	35.0	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...
886	0	Second	27.0	0	0	1	0	0	1
887	1	First	19.0	0	1	0	0	0	1
888	0	Third	28.0	2	1	0	0	0	1
889	1	First	26.0	0	0	1	1	0	0
890	0	Third	32.0	0	0	1	0	1	0

891 rows × 9 columns

In [23]:
```
data2=pd.get_dummies(data2) #convert the strings into integers for each column using getdummies()
```

In [24]:
```
data2
```

Out[24]:

	Survived	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pclass_Third
0	0	22.0	0	0	1	0	0	1	0	0	1
1	1	38.0	0	1	0	1	0	0	1	0	0
2	1	26.0	0	1	0	0	0	1	0	0	1
3	1	35.0	0	1	0	0	0	1	1	0	0
4	0	35.0	0	0	1	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...
886	0	27.0	0	0	1	0	0	1	0	1	0
887	1	19.0	0	1	0	0	0	1	1	0	0
888	0	28.0	2	1	0	0	0	1	0	0	1
889	1	26.0	0	0	1	1	0	0	1	0	0
890	0	32.0	0	0	1	0	1	0	0	0	1

891 rows × 11 columns

In [25]:
```
data2=data2.fillna(data1.median()) #fill the null values with median of data
```

In [26]:
```
data2
```

Out[26]:

	Survived	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pclass_Third
0	0	22.0	0	0	1	0	0	1	0	0	1
1	1	38.0	0	1	0	1	0	0	1	0	0
2	1	26.0	0	1	0	0	0	1	0	0	1
3	1	35.0	0	1	0	0	0	1	1	0	0
4	0	35.0	0	0	1	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...
886	0	27.0	0	0	1	0	0	1	0	1	0
887	1	19.0	0	1	0	0	0	1	1	0	0
888	0	28.0	2	1	0	0	0	1	0	0	1
889	1	26.0	0	0	1	1	0	0	1	0	0
890	0	32.0	0	0	1	0	1	0	0	0	1

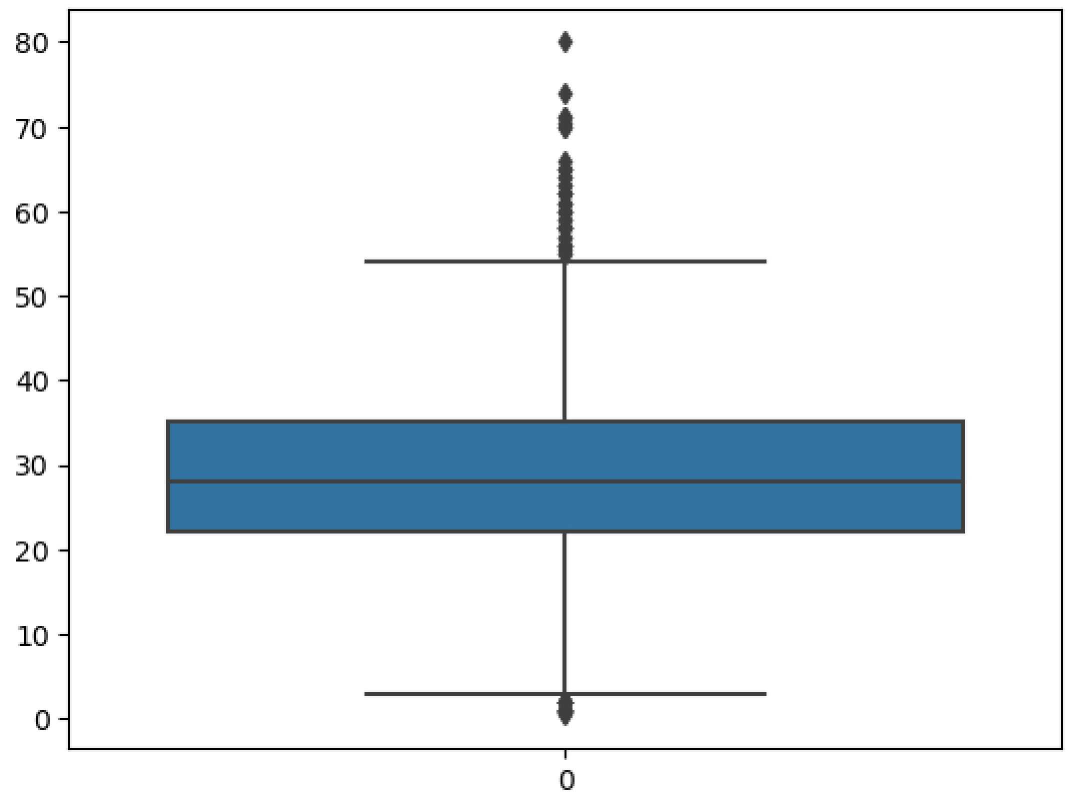
891 rows × 11 columns

In [27]:
```
data2.isna().sum()   #checking if data have any null values with isna()
```

Out[27]:
```
Survived         0
Age              0
Parch            0
Sex_female       0
Sex_male         0
Embarked_C       0
Embarked_Q       0
Embarked_S       0
Pclass_First     0
Pclass_Second    0
Pclass_Third     0
dtype: int64
```

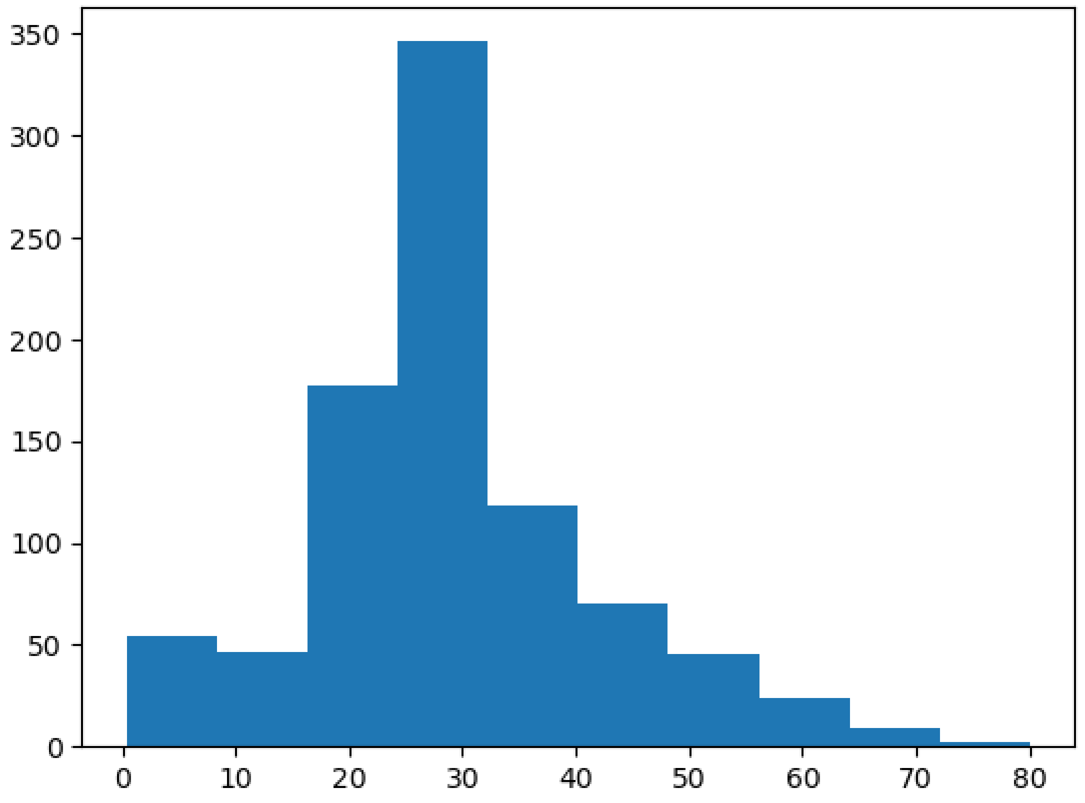
In [28]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(data2.Age)
```

Out[28]:
```
<Axes: >
```



In [29]:
```
plt.hist(data2['Age'])
```

Out[29]:
```
(array([ 54.,  46., 177., 346., 118.,  70.,  45.,  24.,   9.,   2.]),
 array([ 0.42 ,  8.378, 16.336, 24.294, 32.252, 40.21 , 48.168, 56.126,
        64.084, 72.042, 80.   ]),
 <BarContainer object of 10 artists>)
```



In [30]:
```
dataage=data2.groupby([data2.Age]).sum() #get the data based on age using groupby()
```

In [31]:
```
dataage.tail(10)
```

Out[31]:

	Survived	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pclass_Third
Age										
62.0	2	0	1	3	0	0	3	3	1	0
63.0	2	0	2	0	0	0	2	1	0	1
64.0	0	4	0	2	0	0	2	2	0	0
65.0	0	1	0	3	1	1	1	2	0	1
66.0	0	0	0	1	0	0	1	0	1	0
70.0	0	1	0	2	0	0	2	1	1	0
70.5	0	0	0	1	0	1	0	0	0	1
71.0	0	0	0	2	2	0	0	2	0	0
74.0	0	0	0	1	0	0	1	0	0	1
80.0	1	0	0	1	0	0	1	1	0	0

In [32]:
```
data2.describe()
```

Out[32]:

	Survived	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pcla
count	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	89
mean	0.383838	29.361582	0.381594	0.352413	0.647587	0.188552	0.086420	0.722783	0.242424	0.206510	
std	0.486592	13.019697	0.806057	0.477990	0.477990	0.391372	0.281141	0.447876	0.428790	0.405028	
min	0.000000	0.420000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	22.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
50%	0.000000	28.000000	0.000000	0.000000	1.000000	0.000000	0.000000	1.000000	0.000000	0.000000	
75%	1.000000	35.000000	0.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000	0.000000	
max	1.000000	80.000000	6.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	

In [33]:
```
data2['Age'].unique()
```

Out[33]:
```
array([22.  , 38.  , 26.  , 35.  , 28.  , 54.  ,  2.  , 27.  , 14.  ,
        4.  , 58.  , 20.  , 39.  , 55.  , 31.  , 34.  , 15.  ,  8.  ,
       19.  , 40.  , 66.  , 42.  , 21.  , 18.  ,  3.  ,  7.  , 49.  ,
       29.  , 65.  , 28.5 ,  5.  , 11.  , 45.  , 17.  , 32.  , 16.  ,
       25.  ,  0.83, 30.  , 33.  , 23.  , 24.  , 46.  , 59.  , 71.  ,
       37.  , 47.  , 14.5 , 70.5 , 32.5 , 12.  ,  9.  , 36.5 , 51.  ,
       55.5 , 40.5 , 44.  ,  1.  , 61.  , 56.  , 50.  , 36.  , 45.5 ,
       20.5 , 62.  , 41.  , 52.  , 63.  , 23.5 ,  0.92, 43.  , 60.  ,
       10.  , 64.  , 13.  , 48.  ,  0.75, 53.  , 57.  , 80.  , 70.  ,
       24.5 ,  6.  ,  0.67, 30.5 ,  0.42, 34.5 , 74.  ])
```

In [34]:
```
data2.describe()
```

Out[34]:

	Survived	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pcla
count	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000	89
mean	0.383838	29.361582	0.381594	0.352413	0.647587	0.188552	0.086420	0.722783	0.242424	0.206510	
std	0.486592	13.019697	0.806057	0.477990	0.477990	0.391372	0.281141	0.447876	0.428790	0.405028	
min	0.000000	0.420000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	22.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
50%	0.000000	28.000000	0.000000	0.000000	1.000000	0.000000	0.000000	1.000000	0.000000	0.000000	
75%	1.000000	35.000000	0.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000	0.000000	
max	1.000000	80.000000	6.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	

In [35]:
```
cor=data2.corr()    #correlation of data
```

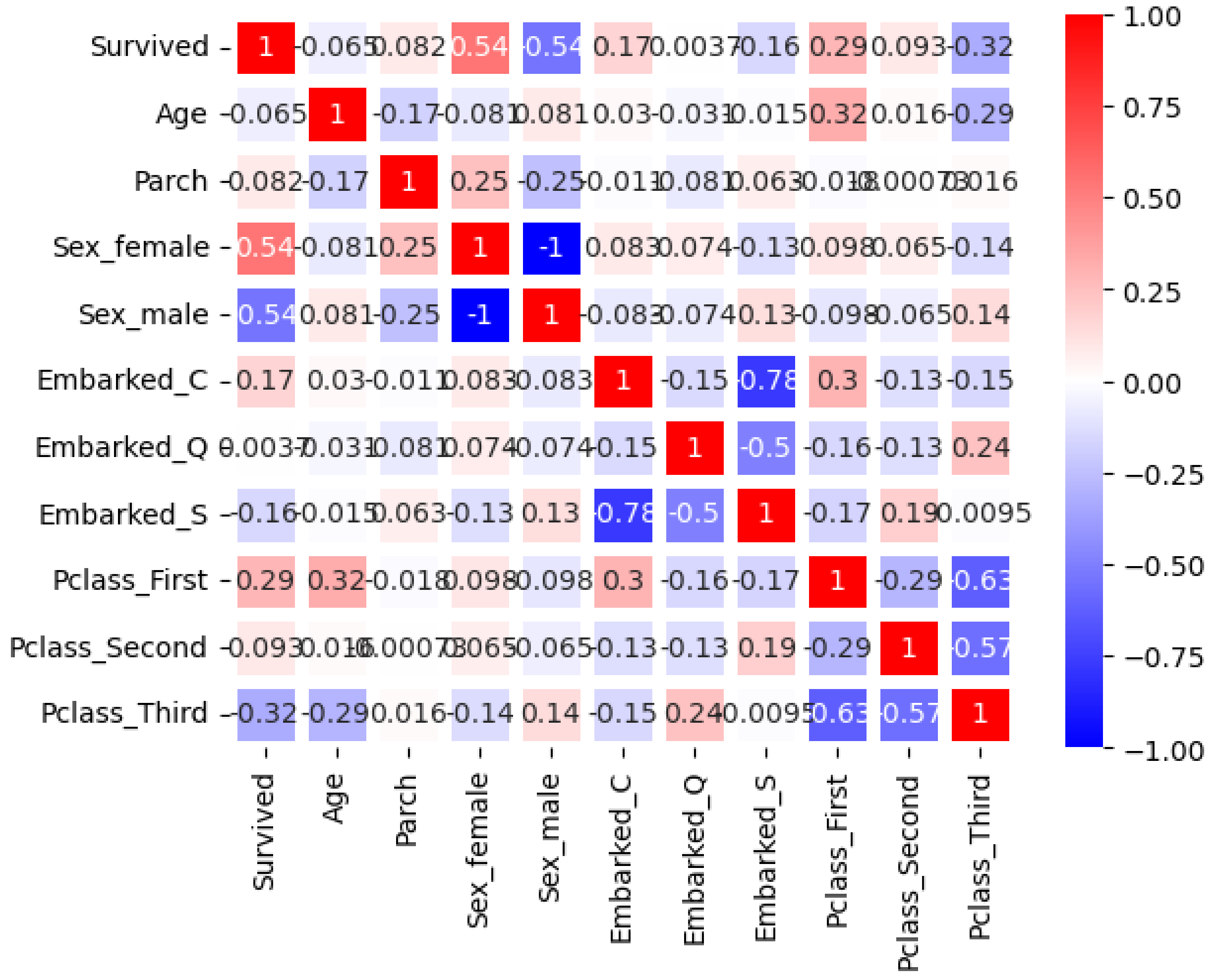
In [36]:
```
cor
```

Out[36]:

	Survived	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second
Survived	1.000000	-0.064910	0.081629	0.543351	-0.543351	0.168240	0.003650	-0.155660	0.285904	0.093349
Age	-0.064910	1.000000	-0.172482	-0.081163	0.081163	0.030248	-0.031415	-0.014665	0.323896	0.015831
Parch	0.081629	-0.172482	1.000000	0.245489	-0.245489	-0.011069	-0.081228	0.063036	-0.017633	-0.000734
Sex_female	0.543351	-0.081163	0.245489	1.000000	-1.000000	0.082853	0.074115	-0.125722	0.098013	0.064746
Sex_male	-0.543351	0.081163	-0.245489	-1.000000	1.000000	-0.082853	-0.074115	0.125722	-0.098013	-0.064746
Embarked_C	0.168240	0.030248	-0.011069	0.082853	-0.082853	1.000000	-0.148258	-0.778359	0.296423	-0.125416
Embarked_Q	0.003650	-0.031415	-0.081228	0.074115	-0.074115	-0.148258	1.000000	-0.496624	-0.155342	-0.127301
Embarked_S	-0.155660	-0.014665	0.063036	-0.125722	0.125722	-0.778359	-0.496624	1.000000	-0.170379	0.192061
Pclass_First	0.285904	0.323896	-0.017633	0.098013	-0.098013	0.296423	-0.155342	-0.170379	1.000000	-0.288585
Pclass_Second	0.093349	0.015831	-0.000734	0.064746	-0.064746	-0.125416	-0.127301	0.192061	-0.288585	1.000000
Pclass_Third	-0.322308	-0.291955	0.015790	-0.137143	0.137143	-0.153329	0.237449	-0.009511	-0.626738	-0.565210

In [37]:
```
sns.heatmap(cor,vmax=1,vmin=-1,annot=True,linewidth=5,cmap='bwr')
#correlation matrix using heatmap
```

Out[37]:
```
<Axes: >
```



In [38]:
```
data2.groupby(["Survived"]).count()
```

Out[38]:

	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pclass_Third
Survived										
0	549	549	549	549	549	549	549	549	549	549
1	342	342	342	342	342	342	342	342	342	342

In [39]:
```
y=data2['Survived']
x=data2.drop(columns='Survived')
```

In [40]:
```
x
```

Out[40]:

	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pclass_Third
0	22.0	0	0	1	0	0	1	0	0	1
1	38.0	0	1	0	1	0	0	1	0	0
2	26.0	0	1	0	0	0	1	0	0	1
3	35.0	0	1	0	0	0	1	1	0	0
4	35.0	0	0	1	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...
886	27.0	0	0	1	0	0	1	0	1	0
887	19.0	0	1	0	0	0	1	1	0	0
888	28.0	2	1	0	0	0	1	0	0	1
889	26.0	0	0	1	1	0	0	1	0	0
890	32.0	0	0	1	0	1	0	0	0	1

891 rows × 10 columns

In [41]:
```
y
```

Out[41]:
```
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
```

split the data into training set and testing set

In [42]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=42)
```

In [43]:
```
x_train
```

Out[43]:

	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pclass_Third
6	54.0	0	0	1	0	0	1	1	0	0
718	28.0	0	0	1	0	1	0	0	0	1
685	25.0	2	0	1	1	0	0	0	1	0
73	26.0	0	0	1	1	0	0	0	0	1
882	22.0	0	1	0	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...
106	21.0	0	1	0	0	0	1	0	0	1
270	28.0	0	0	1	0	0	1	1	0	0
860	41.0	0	0	1	0	0	1	0	0	1
435	14.0	2	1	0	0	0	1	1	0	0
102	21.0	1	0	1	0	0	1	1	0	0

596 rows × 10 columns

In [44]:
```
x_test
```

Out[44]:

	Age	Parch	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S	Pclass_First	Pclass_Second	Pclass_Third
709	28.0	1	0	1	1	0	0	0	0	1
439	31.0	0	0	1	0	0	1	0	1	0
840	20.0	0	0	1	0	0	1	0	0	1
720	6.0	1	1	0	0	0	1	0	1	0
39	14.0	0	1	0	1	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...
715	19.0	0	0	1	0	0	1	0	0	1
525	40.5	0	0	1	0	1	0	0	0	1
381	1.0	2	1	0	1	0	0	0	0	1
140	28.0	2	1	0	1	0	0	0	0	1
173	21.0	0	0	1	0	0	1	0	0	1

295 rows × 10 columns

Logistic Regression

In [45]:
```
from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression()
classifier.fit(x_train,y_train)
```

Out[45]:
```
▾ LogisticRegression
LogisticRegression()
```

In [46]:
```
y_pred=classifier.predict(x_test)    #multiply x_test with classifier
```

```
In [47]: y_pred

Out[47]: array([0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
                1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
                1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1,
                0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1,
                0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
                1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0,
                0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1,
                0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0,
                0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0,
                1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
                0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1,
                0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0,
                0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
                1, 0, 0, 0, 0, 0, 1, 1, 0])
```

```
In [48]: from sklearn.metrics import confusion_matrix     #confusion matrix
         confusion_matrix(y_test,y_pred)

Out[48]: array([[152,  23],
                [ 35,  85]])
```

```
In [49]: from sklearn.metrics import accuracy_score   ##accuracy of test data and predicted data
         accuracy_score(y_test,y_pred)

Out[49]: 0.8033898305084746
```

In [50]:
```
y
```

Out[50]:
```
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 [ ]:

In [ ]: