In [1]:
```
import pandas as pd #importing pandas
import numpy as np #importing numpy
import seaborn as sb #importing seaborn
import matplotlib.pyplot as mp #importing matplotlib
```

In [2]:
```
data=pd.read_csv("/home/placement/Downloads/Titanic Dataset.csv") #reading the titanic data
```

In [3]:
```
data.describe()
```

Out[3]:

	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 [4]:
```
data.head(10)
```

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
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 [5]:
```
data.isna().sum() #getting the nul values count
```

Out[5]:
```
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 [6]:
```
data.Pclass.unique() #getting the unique data of pclass
```

Out[6]:
```
array([3, 1, 2])
```

In [7]:
```
data.Survived.unique()
```

Out[7]:
```
array([0, 1])
```

In [8]:
```
data['Age'].unique()
```

Out[8]:
```
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 [9]:
```
data1=data.drop(['PassengerId','Name','Cabin','Ticket','Parch','SibSp'],axis=1) #droping columns
data1
```

Out[9]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	male	22.0	7.2500	S
1	1	1	female	38.0	71.2833	C
2	1	3	female	26.0	7.9250	S
3	1	1	female	35.0	53.1000	S
4	0	3	male	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	male	27.0	13.0000	S
887	1	1	female	19.0	30.0000	S
888	0	3	female	NaN	23.4500	S
889	1	1	male	26.0	30.0000	C
890	0	3	male	32.0	7.7500	Q

891 rows × 6 columns

In [10]:
```
list(data1)
```

Out[10]:
```
['Survived', 'Pclass', 'Sex', 'Age', 'Fare', 'Embarked']
```

In [11]:
```
data1["Sex"]=data1["Sex"].map({"male":1,"female":0}) #maping the male and female data
data1
```

Out[11]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	1	22.0	7.2500	S
1	1	1	0	38.0	71.2833	C
2	1	3	0	26.0	7.9250	S
3	1	1	0	35.0	53.1000	S
4	0	3	1	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	S
887	1	1	0	19.0	30.0000	S
888	0	3	0	NaN	23.4500	S
889	1	1	1	26.0	30.0000	C
890	0	3	1	32.0	7.7500	Q

891 rows × 6 columns

In [12]:
```
data1
```

Out[12]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	1	22.0	7.2500	S
1	1	1	0	38.0	71.2833	C
2	1	3	0	26.0	7.9250	S
3	1	1	0	35.0	53.1000	S
4	0	3	1	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	S
887	1	1	0	19.0	30.0000	S
888	0	3	0	NaN	23.4500	S
889	1	1	1	26.0	30.0000	C
890	0	3	1	32.0	7.7500	Q

891 rows × 6 columns

In [13]:
```
data1.fillna(data1.mode) #fill null values with mode of the data
```

Out[13]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	1	22.0	7.2500	S
1	1	1	0	38.0	71.2833	C
2	1	3	0	26.0	7.9250	S
3	1	1	0	35.0	53.1000	S
4	0	3	1	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	S
887	1	1	0	19.0	30.0000	S
888	0	3	0	<bound method DataFrame.mode of Survived ...	23.4500	S
889	1	1	1	26.0	30.0000	C
890	0	3	1	32.0	7.7500	Q

891 rows × 6 columns

In [14]:
```
data2=data1.fillna(data1.mean)#fill null values with maen of the data
data2
```

Out[14]:

	Survived	Pclass	Sex		Age	Fare	Embarked
0	0	3	1		22.0	7.2500	S
1	1	1	0		38.0	71.2833	C
2	1	3	0		26.0	7.9250	S
3	1	1	0		35.0	53.1000	S
4	0	3	1		35.0	8.0500	S
...	...	...	...		...	...	...
886	0	2	1		27.0	13.0000	S
887	1	1	0		19.0	30.0000	S
888	0	3	0	<bound method NDFrame._add_numeric_operations....		23.4500	S
889	1	1	1		26.0	30.0000	C
890	0	3	1		32.0	7.7500	Q

891 rows × 6 columns

In [16]:
```
mo=data1.fillna(data1.mode)
mo
```

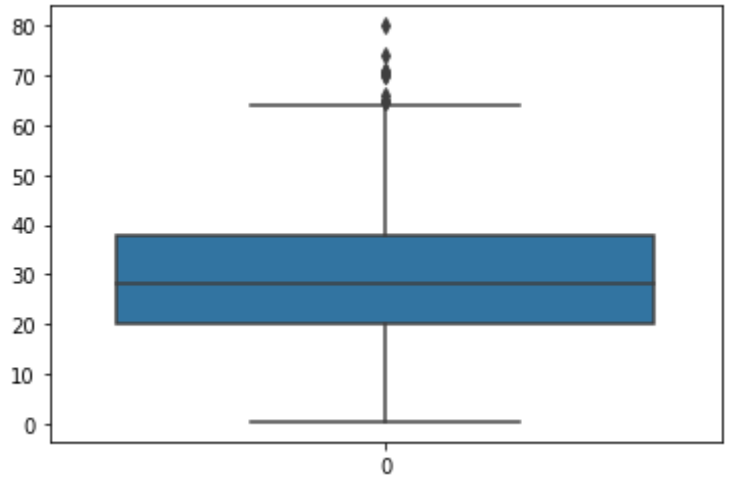
Out[16]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	1	22.0	7.2500	S
1	1	1	0	38.0	71.2833	C
2	1	3	0	26.0	7.9250	S
3	1	1	0	35.0	53.1000	S
4	0	3	1	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	S
887	1	1	0	19.0	30.0000	S
888	0	3	0	<bound method DataFrame.mode of Survived ...	23.4500	S
889	1	1	1	26.0	30.0000	C
890	0	3	1	32.0	7.7500	Q

891 rows × 6 columns

In [17]:
```
sb.boxplot(data1['Age']) #ploting the age column
```

Out[17]: <Axes: >



In [18]:
```
data1
```

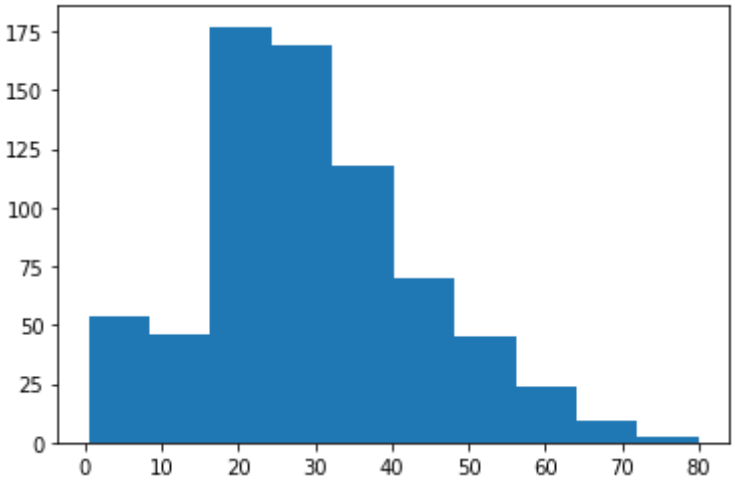
Out[18]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	1	22.0	7.2500	S
1	1	1	0	38.0	71.2833	C
2	1	3	0	26.0	7.9250	S
3	1	1	0	35.0	53.1000	S
4	0	3	1	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	S
887	1	1	0	19.0	30.0000	S
888	0	3	0	NaN	23.4500	S
889	1	1	1	26.0	30.0000	C
890	0	3	1	32.0	7.7500	Q

891 rows × 6 columns

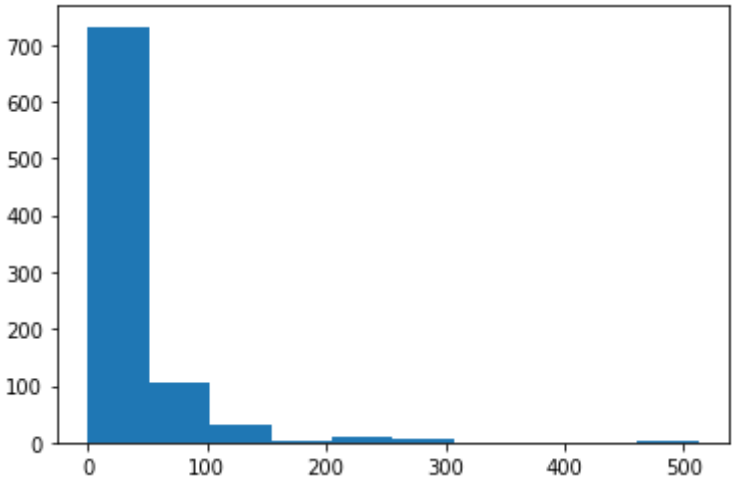
In [19]:
```
mp.hist(data1['Age']) #histograph
```

Out[19]:
```
(array([ 54.,  46., 177., 169., 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>)
```



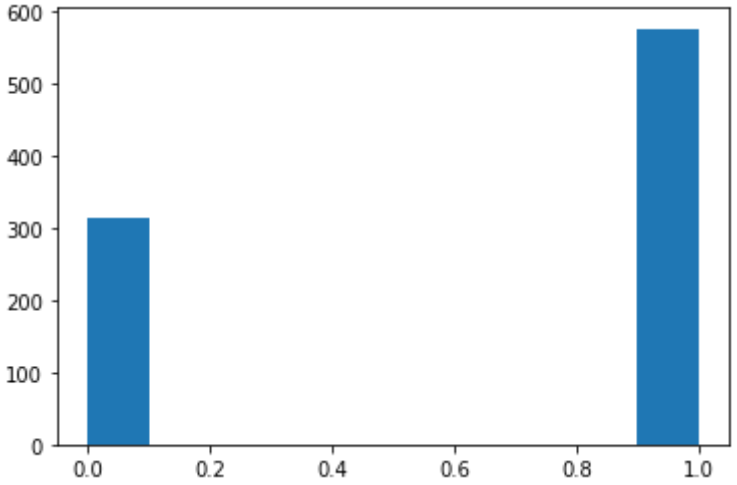
In [20]:
```
mp.hist(data1['Fare'])
```

Out[20]:
```
(array([732., 106.,  31.,   2.,  11.,   6.,   0.,   0.,   0.,   3.]),
 array([  0.     ,  51.23292, 102.46584, 153.69876, 204.93168, 256.1646 ,
        307.39752, 358.63044, 409.86336, 461.09628, 512.3292 ]),
 <BarContainer object of 10 artists>)
```



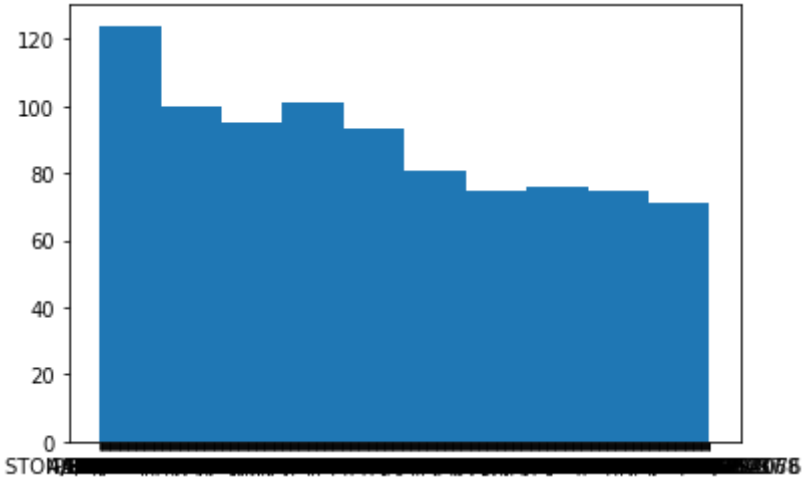
In [21]:
```
mp.hist(data1['Sex'])
```

Out[21]:
```
(array([314.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., 577.]),
 array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ]),
 <BarContainer object of 10 artists>)
```



In [22]:
```
mp.hist(data['Ticket'])
```

Out[22]:
```
(array([124., 100.,  95., 101.,  93.,  81.,  75.,  76.,  75.,  71.]),
 array([  0.,  68., 136., 204., 272., 340., 408., 476., 544., 612., 680.]),
 <BarContainer object of 10 artists>)
```



In [23]:
```
md.isna().sum()
```

Out[23]:
```
Survived    0
Pclass      0
Sex         0
Age         0
Fare        0
Embarked    0
dtype: int64
```

In [24]:
```
data1.fillna(35,inplace=True) #replacing null values with 35 because most of age people in 35
```

In [25]:
```
data1.isnull().sum() #counting the null values
```

Out[25]:
```
Survived    0
Pclass      0
Sex         0
Age         0
Fare        0
Embarked    0
dtype: int64
```

In [26]:
```
data1.describe()
```

Out[26]:

	Survived	Pclass	Sex	Age	Fare
count	891.000000	891.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	0.647587	30.752155	32.204208
std	0.486592	0.836071	0.477990	13.173100	49.693429
min	0.000000	1.000000	0.000000	0.420000	0.000000
25%	0.000000	2.000000	0.000000	22.000000	7.910400
50%	0.000000	3.000000	1.000000	32.000000	14.454200
75%	1.000000	3.000000	1.000000	35.000000	31.000000
max	1.000000	3.000000	1.000000	80.000000	512.329200

In [27]:
```
data1['Age'].unique()
```

Out[27]:
```
array([22.  , 38.  , 26.  , 35.  , 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 [28]:
```
data1
```

Out[28]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	1	22.0	7.2500	S
1	1	1	0	38.0	71.2833	C
2	1	3	0	26.0	7.9250	S
3	1	1	0	35.0	53.1000	S
4	0	3	1	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	1	27.0	13.0000	S
887	1	1	0	19.0	30.0000	S
888	0	3	0	35.0	23.4500	S
889	1	1	1	26.0	30.0000	C
890	0	3	1	32.0	7.7500	Q

891 rows × 6 columns

In [29]:
```
data1['Pclass']=data1['Pclass'].map({1:'F',2:'S',3:'T'})
data1
```

Out[29]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	T	1	22.0	7.2500	S
1	1	F	0	38.0	71.2833	C
2	1	T	0	26.0	7.9250	S
3	1	F	0	35.0	53.1000	S
4	0	T	1	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	S	1	27.0	13.0000	S
887	1	F	0	19.0	30.0000	S
888	0	T	0	35.0	23.4500	S
889	1	F	1	26.0	30.0000	C
890	0	T	1	32.0	7.7500	Q

891 rows × 6 columns

In [30]:
```
data2=pd.get_dummies(data1,dtype=int) #creating dummies
data2
```

Out[30]:

	Survived	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_T	Embarked_35	Embarked_C	Embarked_Q	Embarked_S
0	0	1	22.0	7.2500	0	0	1	0	0	0	1
1	1	0	38.0	71.2833	1	0	0	0	1	0	0
2	1	0	26.0	7.9250	0	0	1	0	0	0	1
3	1	0	35.0	53.1000	1	0	0	0	0	0	1
4	0	1	35.0	8.0500	0	0	1	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...
886	0	1	27.0	13.0000	0	1	0	0	0	0	1
887	1	0	19.0	30.0000	1	0	0	0	0	0	1
888	0	0	35.0	23.4500	0	0	1	0	0	0	1
889	1	1	26.0	30.0000	1	0	0	0	1	0	0
890	0	1	32.0	7.7500	0	0	1	0	0	1	0

891 rows × 11 columns

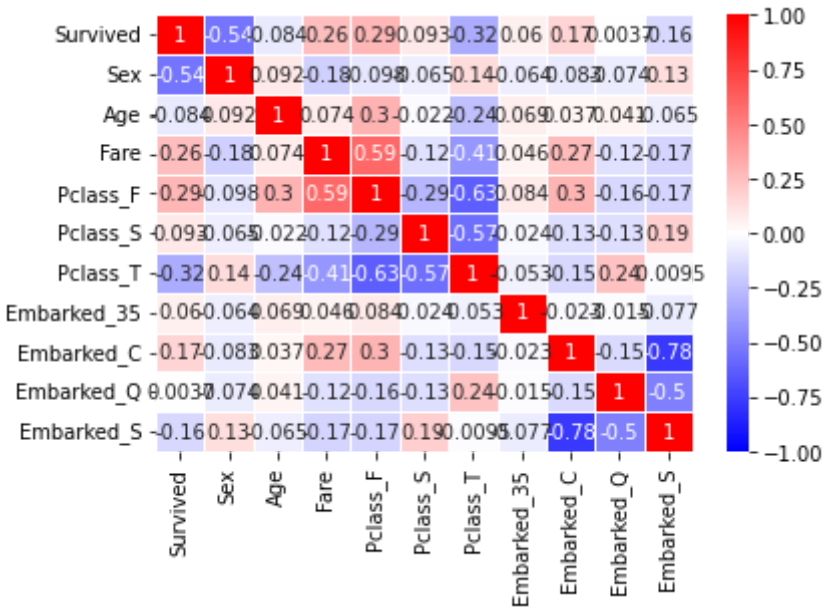
In [31]:
```
cor=data2.corr() #correlation for the data
cor
```

Out[31]:

	Survived	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_T	Embarked_35	Embarked_C	Embarked_Q	Embarked_S
Survived	1.000000	-0.543351	-0.083713	0.257307	0.285904	0.093349	-0.322308	0.060095	0.168240	0.003650	-0.155660
Sex	-0.543351	1.000000	0.091930	-0.182333	-0.098013	-0.064746	0.137143	-0.064296	-0.082853	-0.074115	0.125722
Age	-0.083713	0.091930	1.000000	0.074199	0.302149	-0.022021	-0.242412	0.069343	0.036953	0.040528	-0.065062
Fare	0.257307	-0.182333	0.074199	1.000000	0.591711	-0.118557	-0.413333	0.045646	0.269335	-0.117216	-0.166603
Pclass_F	0.285904	-0.098013	0.302149	0.591711	1.000000	-0.288585	-0.626738	0.083847	0.296423	-0.155342	-0.170379
Pclass_S	0.093349	-0.064746	-0.022021	-0.118557	-0.288585	1.000000	-0.565210	-0.024197	-0.125416	-0.127301	0.192061
Pclass_T	-0.322308	0.137143	-0.242412	-0.413333	-0.626738	-0.565210	1.000000	-0.052550	-0.153329	0.237449	-0.009511
Embarked_35	0.060095	-0.064296	0.069343	0.045646	0.083847	-0.024197	-0.052550	1.000000	-0.022864	-0.014588	-0.076588
Embarked_C	0.168240	-0.082853	0.036953	0.269335	0.296423	-0.125416	-0.153329	-0.022864	1.000000	-0.148258	-0.778359
Embarked_Q	0.003650	-0.074115	0.040528	-0.117216	-0.155342	-0.127301	0.237449	-0.014588	-0.148258	1.000000	-0.496624
Embarked_S	-0.155660	0.125722	-0.065062	-0.166603	-0.170379	0.192061	-0.009511	-0.076588	-0.778359	-0.496624	1.000000

In [32]:
```
sb.heatmap(cor,vmax=1,vmin=-1,annot=True,linewidths=.5,cmap='bwr') #correlation graph
```

Out[32]: <Axes: >



In [33]:
```
data.groupby('Survived').count() #count value for survived
```

Out[33]:

Survived	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	549	549	549	549	424	549	549	549	549	68	549
1	342	342	342	342	290	342	342	342	342	136	340

In [34]:
```
y=data2['Survived']
x=data2.drop('Survived',axis=1)
```

In [35]:
```
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 [36]:
```
import warnings #ignoring warnings
warnings.filterwarnings('ignore')
```

In [37]:
```
from sklearn.linear_model import LogisticRegression #importing logistic regression
classifier=LogisticRegression()
classifier.fit(x_train,y_train)
```

Out[37]:

▾ LogisticRegression

LogisticRegression()

In [38]:
```
y_pred=classifier.predict(x_test) #array values for y_pred
y_pred
```

Out[38]:
```
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, 0, 0, 1, 0, 0, 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, 0, 1, 1, 1, 0, 1, 0,
       0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 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, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0,
       0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 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, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 1, 0])
```

In [39]:
```
from sklearn.metrics import confusion_matrix #confusion matrix importing
confusion_matrix(y_test,y_pred)
```

Out[39]:
```
array([[155,  20],
       [ 37,  83]])
```

In [40]:
```
from sklearn.metrics import accuracy_score #counting the accuracy
accuracy_score(y_test,y_pred)
```

Out[40]:
```
0.8067796610169492
```

In [ ]:

In [ ]: