

```
In [6]: import pandas as pd
```

```
In [7]: data=pd.read_csv("/home/placement/Downloads/Titanic Dataset.csv")
```

```
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.describe()
```

```
Out[9]:
```

	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 [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(['PassengerId', 'Cabin', 'Name', 'Ticket', 'SibSp', 'Parch'],axis=1)
```

In [16]: data1

Out[16]:

	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 [17]: data1['Sex']=data1['Sex'].map({'male':1,'female':0})

In [18]: data2=data1.fillna(data1.median())

/tmp/ipykernel\_3816/1290514040.py:1: FutureWarning: The default value of numeric\_only in DataFrame.median is deprecated. In a future version, it will default to False. In addition, specifying 'numeric\_only=None' is deprecated. Select only valid columns or specify the value of numeric\_only to silence this warning.  
 data2=data1.fillna(data1.median())

```
In [19]: data1
```

```
Out[19]:
```

	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

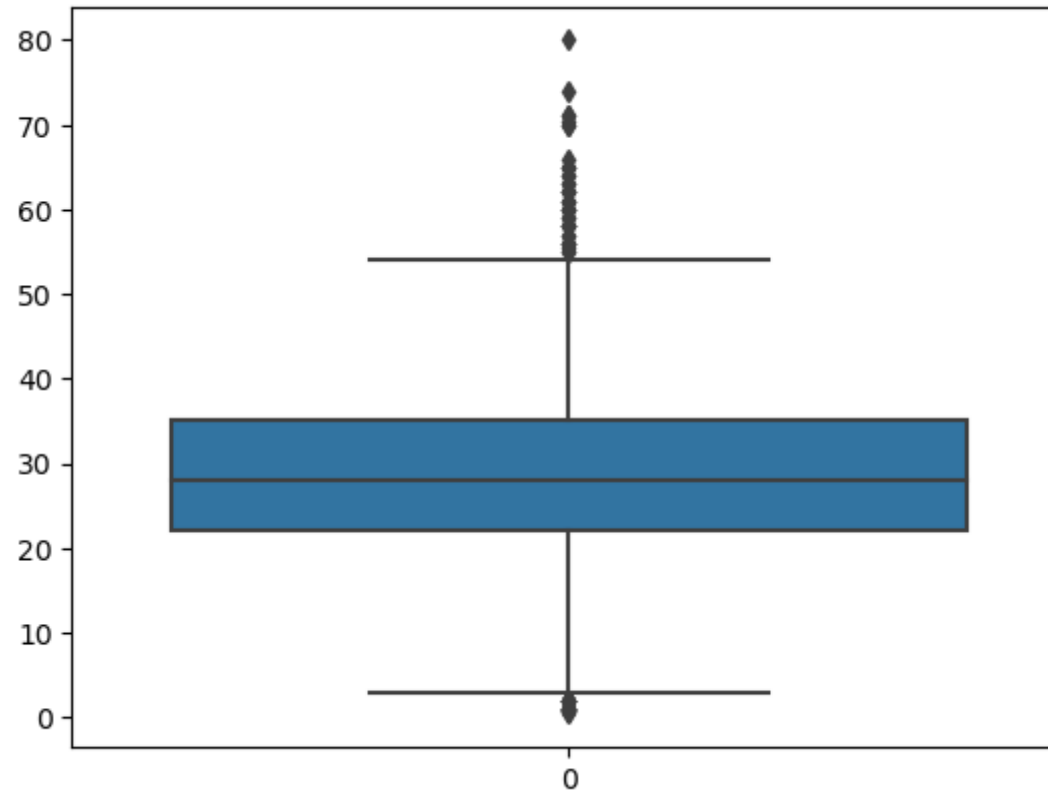
891 rows × 6 columns

```
In [20]: data2.isna().sum()
```

```
Out[20]: Survived    0
Pclass    0
Sex        0
Age        0
Fare       0
Embarked   2
dtype: int64
```

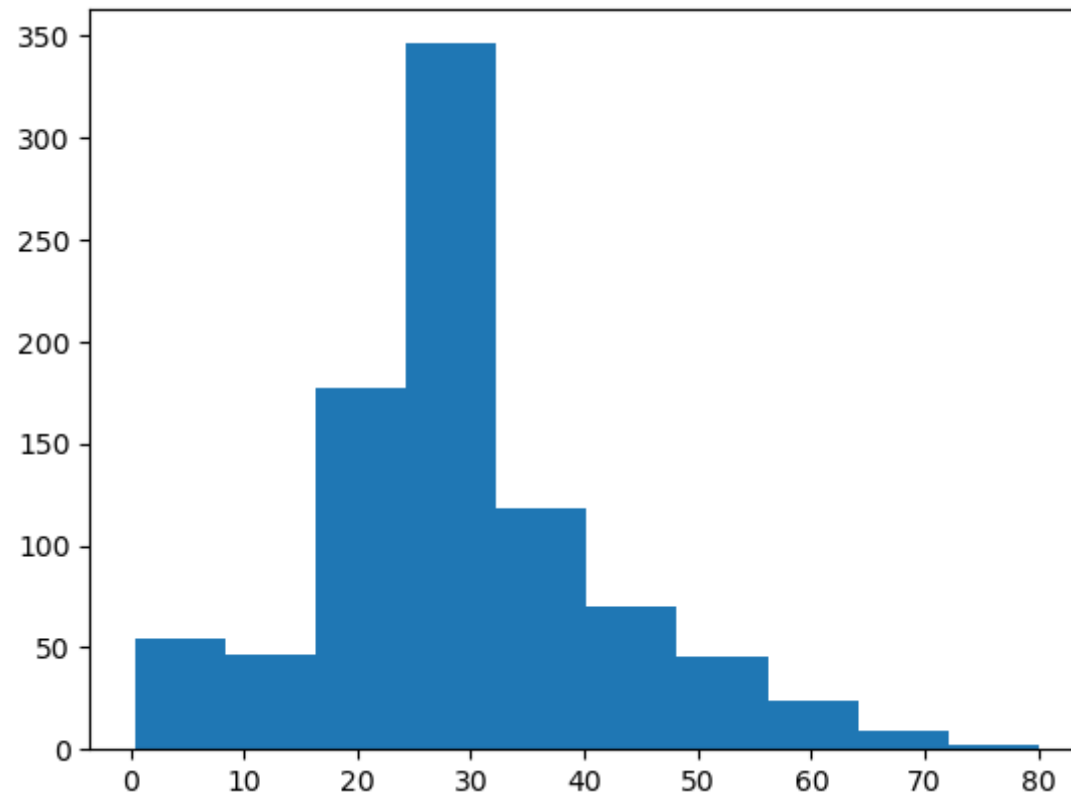
```
In [21]: import seaborn as sns  
import matplotlib.pyplot as plt  
sns.boxplot(data2.Age)
```

Out[21]: <Axes: >



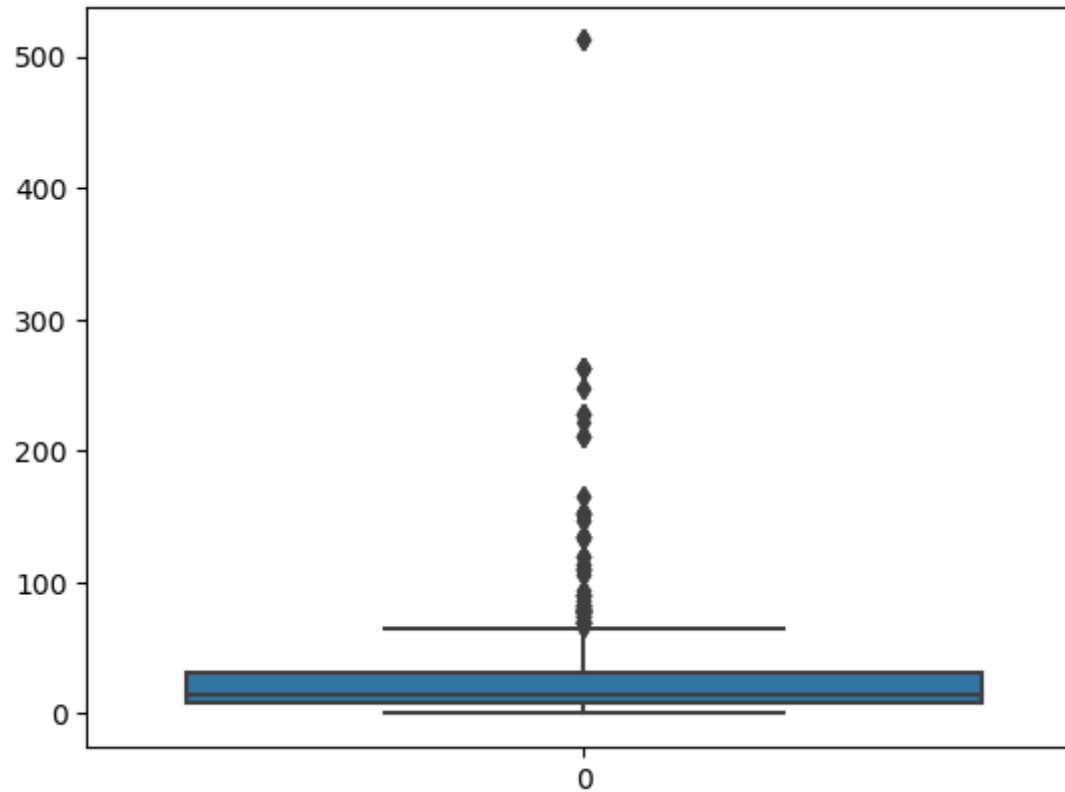
```
In [22]: plt.hist(data2['Age'])
```

```
Out[22]: (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 [23]: import seaborn as sns  
import matplotlib.pyplot as plt  
sns.boxplot(data2.Fare)
```

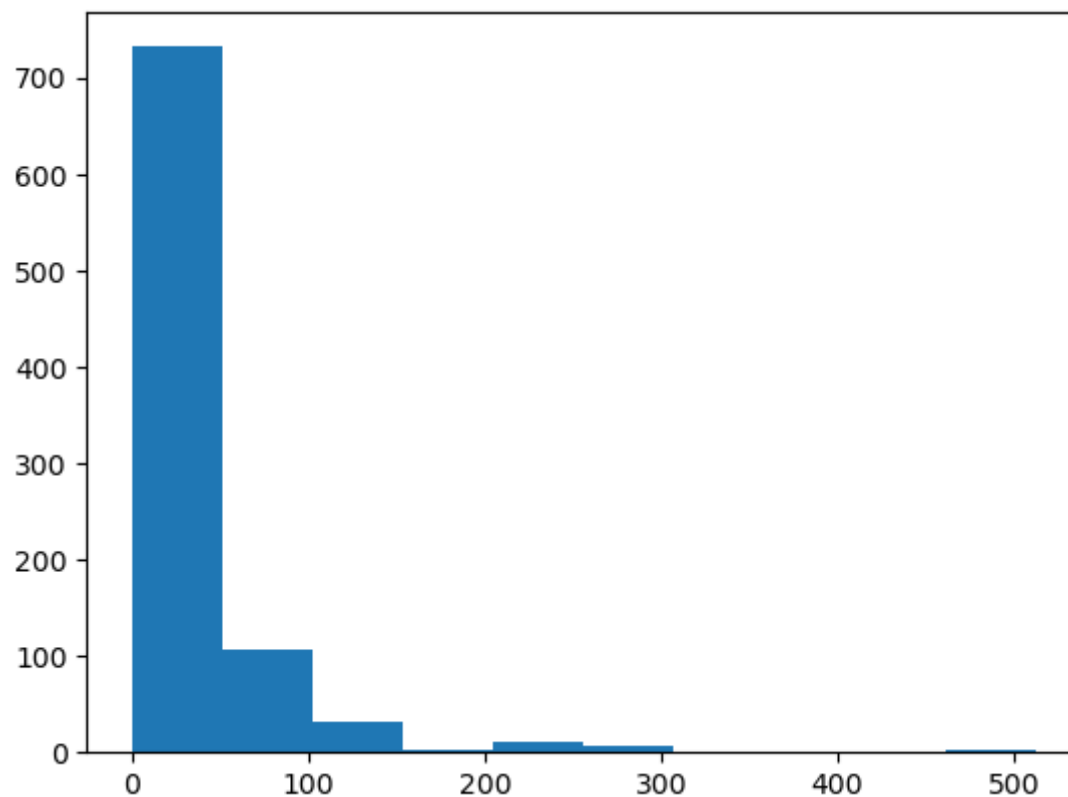
Out[23]: <Axes: >





```
In [24]: plt.hist(data2['Fare'])
```

```
Out[24]: (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 [25]: data.describe()
```

```
Out[25]:
```

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

```
In [26]: data['Age'].unique()
```

```
Out[26]: 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 [27]: data.groupby(['Age']).count()
```

```
Out[27]:
```

	PassengerId	Survived	Pclass	Name	Sex	SibSp	Parch	Ticket	Fare	Cabin	Embarked
Age											
0.42	1	1	1	1	1	1	1	1	1	0	1
0.67	1	1	1	1	1	1	1	1	1	0	1
0.75	2	2	2	2	2	2	2	2	2	0	2
0.83	2	2	2	2	2	2	2	2	2	0	2
0.92	1	1	1	1	1	1	1	1	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...
70.00	2	2	2	2	2	2	2	2	2	1	2
70.50	1	1	1	1	1	1	1	1	1	0	1
71.00	2	2	2	2	2	2	2	2	2	1	2
74.00	1	1	1	1	1	1	1	1	1	0	1
80.00	1	1	1	1	1	1	1	1	1	1	1

88 rows × 11 columns

```
In [28]: data2['Pclass']=data['Pclass'].map({1:'F',2:'S',3:'Third'})
data2
```

Out[28]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	Third	1	22.0	7.2500	S
1	1	F	0	38.0	71.2833	C
2	1	Third	0	26.0	7.9250	S
3	1	F	0	35.0	53.1000	S
4	0	Third	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	Third	0	28.0	23.4500	S
889	1	F	1	26.0	30.0000	C
890	0	Third	1	32.0	7.7500	Q

891 rows × 6 columns

```
In [29]: data2=pd.get_dummies(data2)
data2
```

Out[29]:

	Survived	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_Third	Embarked_C	Embarked_Q	Embarked_S
0	0	1	22.0	7.2500	0	0	1	0	0	1
1	1	0	38.0	71.2833	1	0	0	1	0	0
2	1	0	26.0	7.9250	0	0	1	0	0	1
3	1	0	35.0	53.1000	1	0	0	0	0	1
4	0	1	35.0	8.0500	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...
886	0	1	27.0	13.0000	0	1	0	0	0	1
887	1	0	19.0	30.0000	1	0	0	0	0	1
888	0	0	28.0	23.4500	0	0	1	0	0	1
889	1	1	26.0	30.0000	1	0	0	1	0	0
890	0	1	32.0	7.7500	0	0	1	0	1	0

891 rows × 10 columns

```
In [30]: cor=data2.corr()
```

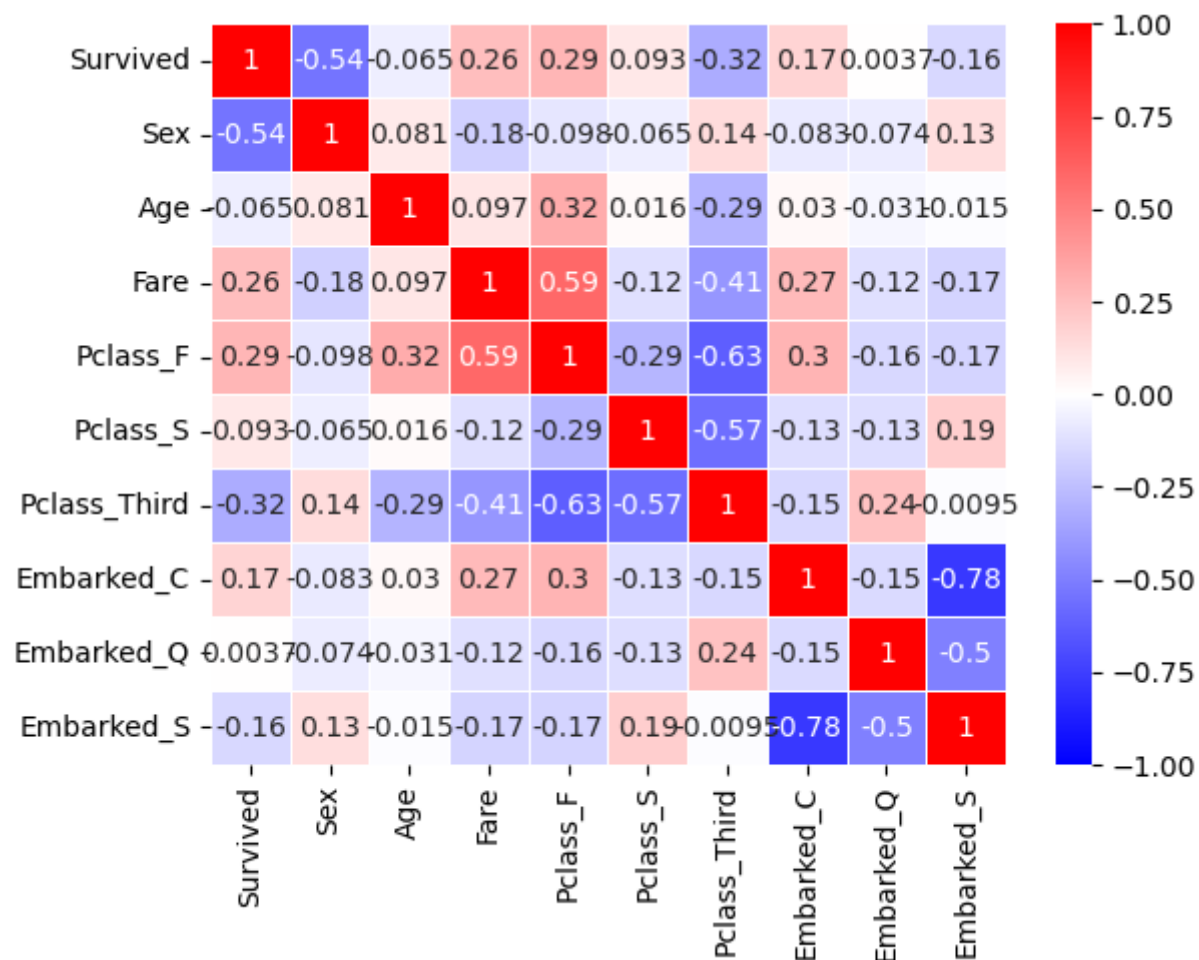
In [31]: cor

Out[31]:

	Survived	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_Third	Embarked_C	Embarked_Q	Embarked_S
<b>Survived</b>	1.000000	-0.543351	-0.064910	0.257307	0.285904	0.093349	-0.322308	0.168240	0.003650	-0.155660
<b>Sex</b>	-0.543351	1.000000	0.081163	-0.182333	-0.098013	-0.064746	0.137143	-0.082853	-0.074115	0.125722
<b>Age</b>	-0.064910	0.081163	1.000000	0.096688	0.323896	0.015831	-0.291955	0.030248	-0.031415	-0.014665
<b>Fare</b>	0.257307	-0.182333	0.096688	1.000000	0.591711	-0.118557	-0.413333	0.269335	-0.117216	-0.166603
<b>Pclass_F</b>	0.285904	-0.098013	0.323896	0.591711	1.000000	-0.288585	-0.626738	0.296423	-0.155342	-0.170379
<b>Pclass_S</b>	0.093349	-0.064746	0.015831	-0.118557	-0.288585	1.000000	-0.565210	-0.125416	-0.127301	0.192061
<b>Pclass_Third</b>	-0.322308	0.137143	-0.291955	-0.413333	-0.626738	-0.565210	1.000000	-0.153329	0.237449	-0.009511
<b>Embarked_C</b>	0.168240	-0.082853	0.030248	0.269335	0.296423	-0.125416	-0.153329	1.000000	-0.148258	-0.778359
<b>Embarked_Q</b>	0.003650	-0.074115	-0.031415	-0.117216	-0.155342	-0.127301	0.237449	-0.148258	1.000000	-0.496624
<b>Embarked_S</b>	-0.155660	0.125722	-0.014665	-0.166603	-0.170379	0.192061	-0.009511	-0.778359	-0.496624	1.000000

```
In [32]: import seaborn as sns
sns.heatmap(cor, vmax=1, vmin=-1, annot=True, linewidth=.5, cmap='bwr')
```

Out[32]: <Axes: >



```
In [33]: data2.groupby('Survived').count()
```

```
Out[33]:
```

	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_Third	Embarked_C	Embarked_Q	Embarked_S
Survived									
0	549	549	549	549	549	549	549	549	549
1	342	342	342	342	342	342	342	342	342

```
In [35]: y=data2['Survived']  
x=data2.drop('Survived',axis=1)
```

```
In [36]: 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 [38]: from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression()
classifier.fit(x_train,y_train)
```

/home/placement/anaconda3/lib/python3.10/site-packages/sklearn/linear\_model/\_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):  
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in:

<https://scikit-learn.org/stable/modules/preprocessing.html> (<https://scikit-learn.org/stable/modules/preprocessing.html>)

Please also refer to the documentation for alternative solver options:

[https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression) ([https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression))

```
n_iter_i = _check_optimize_result(
```

```
Out[38]: ▾ LogisticRegression
LogisticRegression()
```

```
In [45]: y_pred=classifier.predict(x_test)
```

```
In [46]: y_pred
```

```
Out[46]: 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, 0, 1, 1, 0, 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, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
1, 0, 1, 0, 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, 0, 1, 0, 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, 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 [47]: from sklearn.metrics import confusion_matrix  
confusion_matrix(y_test,y_pred)
```

```
Out[47]: array([[154,  21],  
               [ 37,  83]])
```

```
In [48]: from sklearn.metrics import accuracy_score  
accuracy_score(y_test,y_pred)
```

```
Out[48]: 0.8033898305084746
```

```
In [49]: y
```

```
Out[49]: 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 [ ]:
```