In [1]:
```
import pandas as pd
```

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

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)#display top 10 rows
```

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
```

Out[5]:

	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 [6]:
```
data.info()#null values shows
```

```
<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]:
```
data.isna().sum()#no.of null values
```

Out[7]:
```
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 [8]:
```
data['Pclass'].unique()#unique values
```

Out[8]:
```
array([3, 1, 2])
```

In [9]:
```
data['Survived'].unique()
```

Out[9]:
```
array([0, 1])
```

In [10]:
```
data['SibSp'].unique()
```

Out[10]:
```
array([1, 0, 3, 4, 2, 5, 8])
```

In [11]:
```
data['Age'].unique()
```

Out[11]:
```
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 [12]:
```
data1=data.drop(['Cabin','Name','PassengerId','Ticket','SibSp','Parch'],axis=1)#deleting columns
```

In [13]:
```
data1
```

Out[13]:

	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 [14]:
```
data1.isna().sum()#finding null values

```

Out[14]:
```
Survived      0
Pclass        0
Sex           0
Age         177
Fare          0
Embarked      2
dtype: int64
```

In [15]:
```
data1.shape#no of rows and columns
```

Out[15]:
```
(891, 6)
```

In [16]:
```
data1['Sex']=data1['Sex'].map({'male':1,'female':0})#assinging 1 to male and 0 to female using map
```

In [17]:
```
data1
```

Out[17]:

	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 [43]:
```
data1=data1.fillna(data1.mean())#fill null values using mean

```

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

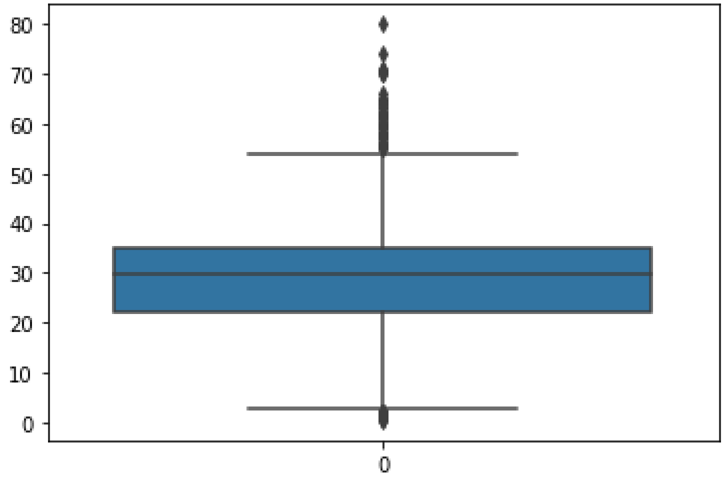
Out[19]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	1	22.000000	7.2500	S
1	1	1	0	38.000000	71.2833	C
2	1	3	0	26.000000	7.9250	S
3	1	1	0	35.000000	53.1000	S
4	0	3	1	35.000000	8.0500	S
...	...	...	...	...	...	...
886	0	2	1	27.000000	13.0000	S
887	1	1	0	19.000000	30.0000	S
888	0	3	0	29.699118	23.4500	S
889	1	1	1	26.000000	30.0000	C
890	0	3	1	32.000000	7.7500	Q

891 rows × 6 columns

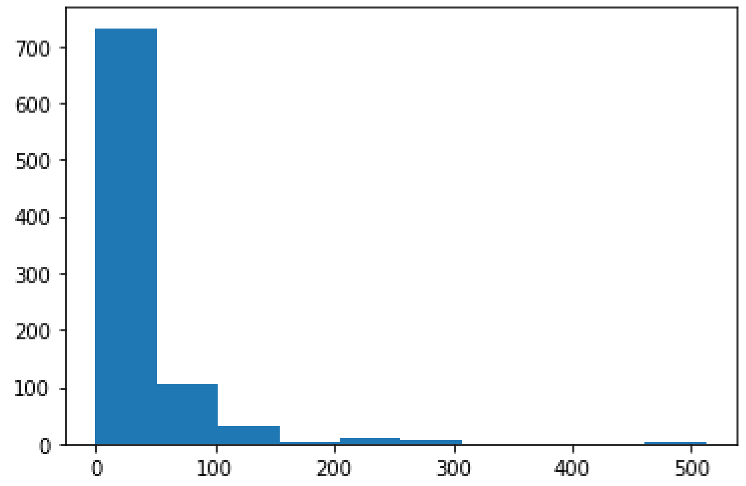
In [20]:
```
import seaborn as sns
import matplotlib.pyplot as mp
sns.boxplot(data1.Age)#plotting for age
```

Out[20]:
```
<AxesSubplot:>
```



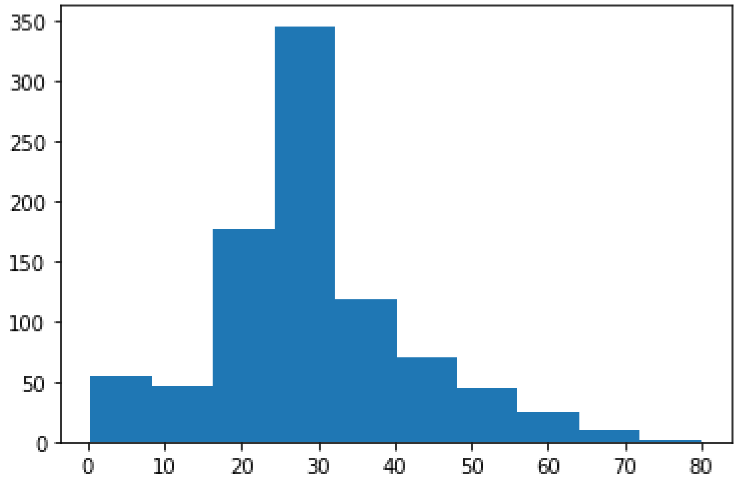
In [21]:
```
mp.hist(data1['Fare'])#histograam for fare
```

Out[21]:
```
(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>)
```



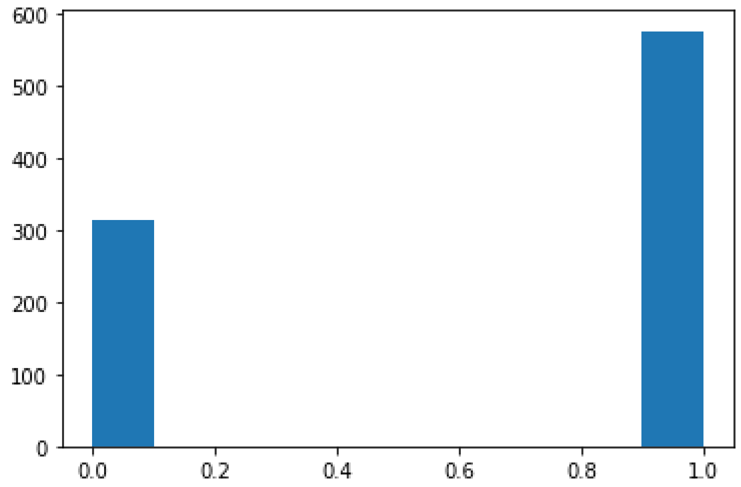
In [22]:
```
mp.hist(data1['Age'])
```

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

Out[23]:
```
(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 [24]:
```
data1['Age'].unique()
```

Out[24]:
```
array([22.        , 38.        , 26.        , 35.        , 29.69911765,
       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 [25]:
```
data1.groupby(['Age']).count()#no of ages can be printed
```

Out[25]:

Age	Survived	Pclass	Sex	Fare	Embarked
0.42	1	1	1	1	1
0.67	1	1	1	1	1
0.75	2	2	2	2	2
0.83	2	2	2	2	2
0.92	1	1	1	1	1
...	...	...	...	...	...
70.00	2	2	2	2	2
70.50	1	1	1	1	1
71.00	2	2	2	2	2
74.00	1	1	1	1	1
80.00	1	1	1	1	1

89 rows × 5 columns

In [26]:
```
data1['Pclass']=data1['Pclass'].map({1:'F',2:'S',3:'t'})#assigning values
```

In [27]:
```
data1
```

Out[27]:

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	t	1	22.000000	7.2500	S
1	1	F	0	38.000000	71.2833	C
2	1	t	0	26.000000	7.9250	S
3	1	F	0	35.000000	53.1000	S
4	0	t	1	35.000000	8.0500	S
...	...	...	...	...	...	...
886	0	S	1	27.000000	13.0000	S
887	1	F	0	19.000000	30.0000	S
888	0	t	0	29.699118	23.4500	S
889	1	F	1	26.000000	30.0000	C
890	0	t	1	32.000000	7.7500	Q

891 rows × 6 columns

In [28]:
```
data2=pd.get_dummies(data1)#where the pclass it shows"1" other pclass it shows "0"
data2
```

Out[28]:

	Survived	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_t	Embarked_C	Embarked_Q	Embarked_S
0	0	1	22.000000	7.2500	0	0	1	0	0	1
1	1	0	38.000000	71.2833	1	0	0	1	0	0
2	1	0	26.000000	7.9250	0	0	1	0	0	1
3	1	0	35.000000	53.1000	1	0	0	0	0	1
4	0	1	35.000000	8.0500	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...	...
886	0	1	27.000000	13.0000	0	1	0	0	0	1
887	1	0	19.000000	30.0000	1	0	0	0	0	1
888	0	0	29.699118	23.4500	0	0	1	0	0	1
889	1	1	26.000000	30.0000	1	0	0	1	0	0
890	0	1	32.000000	7.7500	0	0	1	0	1	0

891 rows × 10 columns

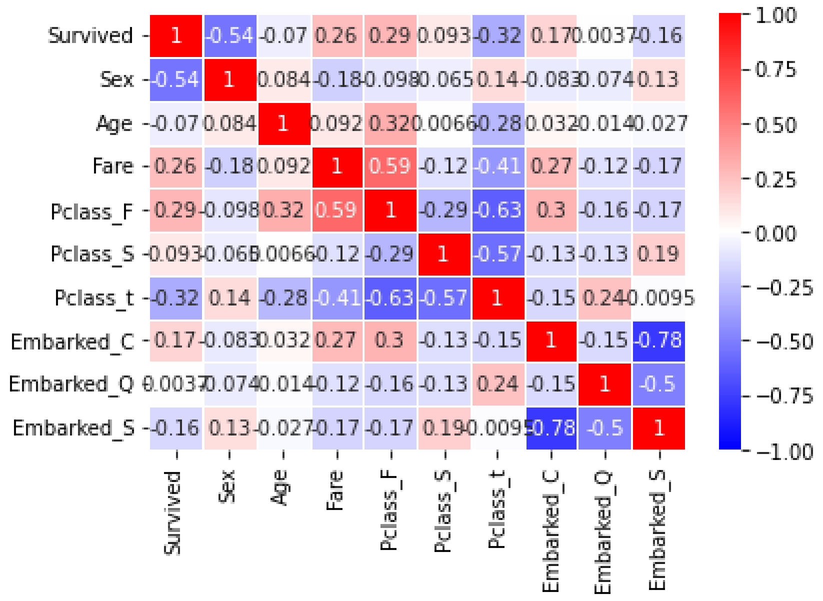
In [29]:
```
cor_mat=data2.corr()#correlation
cor_mat
```

Out[29]:

	Survived	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_t	Embarked_C	Embarked_Q	Embarked_S
Survived	1.000000	-0.543351	-0.069809	0.257307	0.285904	0.093349	-0.322308	0.168240	0.003650	-0.155660
Sex	-0.543351	1.000000	0.084153	-0.182333	-0.098013	-0.064746	0.137143	-0.082853	-0.074115	0.125722
Age	-0.069809	0.084153	1.000000	0.091566	0.319916	0.006589	-0.281004	0.032024	-0.013855	-0.027121
Fare	0.257307	-0.182333	0.091566	1.000000	0.591711	-0.118557	-0.413333	0.269335	-0.117216	-0.166603
Pclass_F	0.285904	-0.098013	0.319916	0.591711	1.000000	-0.288585	-0.626738	0.296423	-0.155342	-0.170379
Pclass_S	0.093349	-0.064746	0.006589	-0.118557	-0.288585	1.000000	-0.565210	-0.125416	-0.127301	0.192061
Pclass_t	-0.322308	0.137143	-0.281004	-0.413333	-0.626738	-0.565210	1.000000	-0.153329	0.237449	-0.009511
Embarked_C	0.168240	-0.082853	0.032024	0.269335	0.296423	-0.125416	-0.153329	1.000000	-0.148258	-0.778359
Embarked_Q	0.003650	-0.074115	-0.013855	-0.117216	-0.155342	-0.127301	0.237449	-0.148258	1.000000	-0.496624
Embarked_S	-0.155660	0.125722	-0.027121	-0.166603	-0.170379	0.192061	-0.009511	-0.778359	-0.496624	1.000000

In [30]:
```
import seaborn as reddy#plotting correlation
sns.heatmap(cor_mat,vmax=1,vmin=-1,annot=True,linewidth=.5,cmap='bwr')
```

Out[30]:
```
<AxesSubplot:>
```



In [31]:
```python
data2.groupby(['Survived']).count()
```

Out[31]:

	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_t	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 [32]:
```python
y=data2['Survived']
x=data2.drop('Survived',axis=1)
```

In [33]:
```python
x
```

Out[33]:

	Sex	Age	Fare	Pclass_F	Pclass_S	Pclass_t	Embarked_C	Embarked_Q	Embarked_S
0	1	22.000000	7.2500	0	0	1	0	0	1
1	0	38.000000	71.2833	1	0	0	1	0	0
2	0	26.000000	7.9250	0	0	1	0	0	1
3	0	35.000000	53.1000	1	0	0	0	0	1
4	1	35.000000	8.0500	0	0	1	0	0	1
...	...	...	...	...	...	...	...	...	...
886	1	27.000000	13.0000	0	1	0	0	0	1
887	0	19.000000	30.0000	1	0	0	0	0	1
888	0	29.699118	23.4500	0	0	1	0	0	1
889	1	26.000000	30.0000	1	0	0	1	0	0
890	1	32.000000	7.7500	0	0	1	0	1	0

891 rows × 9 columns

In [34]:
```
y
```

Out[34]:
```
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 [35]:
```
from sklearn.model_selection import train_test_split#training and testing data
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=42)
```

In [42]:
```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()#logistic regression
classifier.fit(x_train, y_train)

```

Out[42]:
```
LogisticRegression()
```

In [37]:
```
y_pred=classifier.predict(x_test)
```

```
In [38]: 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, 1, 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#confusing matrix
         confusion_matrix(y_test,y_pred)
```

```
Out[39]: array([[154,  21],
                [ 37,  83]])
```

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

```
Out[40]: 0.8033898305084746
```

```
In [ ]: 
```