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

In [2]:
```
data=pd.read_csv("C:/Users/Admin/Downloads/titanic.csv")
```

```
In [3]: data
```

Out[3]:

	pclass	survived	name	sex	age	sibsp	parch	fare	embarked
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0	0	0	211.3375	S
1	1	0	Allison, Miss. Helen Loraine	female	2.0	1	2	151.5500	S
2	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0	1	2	151.5500	S
3	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0	1	2	151.5500	S
4	1	1	Anderson, Mr. Harry	male	48.0	0	0	26.5500	S
5	1	0	Andrews, Mr. Thomas Jr	male	39.0	0	0	0.0000	S
6	1	1	Appleton, Mrs. Edward Dale (Charlotte Lamson)	female	53.0	2	0	51.4792	S
7	1	0	Astor, Col. John Jacob	male	47.0	1	0	227.5250	C
8	1	1	Astor, Mrs. John Jacob (Madeleine Talmadge Force)	female	18.0	1	0	227.5250	C
9	1	1	Aubart, Mme. Leontine Pauline	female	24.0	0	0	69.3000	C
10	1	1	Barber, Miss. Ellen "Nellie"	female	26.0	0	0	78.8500	S
11	1	0	Baumann, Mr. John D	male	NaN	0	0	25.9250	S
12	1	0	Baxter, Mr. Quigg Edmond	male	24.0	0	1	247.5208	C
13	1	1	Baxter, Mrs. James (Helene DeLaudeniere Chaput)	female	50.0	0	1	247.5208	C
14	1	1	Bazzani, Miss. Albina	female	32.0	0	0	76.2917	C
15	1	0	Beattie, Mr. Thomson	male	36.0	0	0	75.2417	C
16	1	1	Beckwith, Mr. Richard Leonard	male	37.0	1	1	52.5542	S
17	1	1	Beckwith, Mrs. Richard Leonard (Sallie Monypeny)	female	47.0	1	1	52.5542	S
18	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	30.0000	C
19	1	1	Bidois, Miss. Rosalie	female	42.0	0	0	227.5250	C
20	1	1	Bird, Miss. Ellen	female	29.0	0	0	221.7792	S
21	1	0	Birnbaum, Mr. Jakob	male	25.0	0	0	26.0000	C
22	1	1	Bishop, Mr. Dickinson H	male	25.0	1	0	91.0792	C
23	1	1	Bishop, Mrs. Dickinson H (Helen Walton)	female	19.0	1	0	91.0792	C
24	1	1	Bissette, Miss. Amelia	female	35.0	0	0	135.6333	S
25	1	1	Bjornstrom-Steffansson, Mr. Mauritz Hakan	male	28.0	0	0	26.5500	S
26	1	0	Blackwell, Mr. Stephen Weart	male	45.0	0	0	35.5000	S
27	1	1	Blank, Mr. Henry	male	40.0	0	0	31.0000	C
28	1	1	Bonnell, Miss. Caroline	female	30.0	0	0	164.8667	S
29	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	26.5500	S
...	...	...	...	...	...	...	...	...	...
1227	3	0	Vander Planke, Miss. Augusta Maria	female	18.0	2	0	18.0000	S
1228	3	0	Vander Planke, Mr. Julius	male	31.0	3	0	18.0000	S
1229	3	0	Vander Planke, Mr. Leo Edmondus	male	16.0	2	0	18.0000	S

	pclass	survived	name	sex	age	sibsp	parch	fare	embarked
1230	3	0	Vander Planke, Mrs. Julius (Emelia Maria Vande...	female	31.0	1	0	18.0000	S
1231	3	1	Vartanian, Mr. David	male	22.0	0	0	7.2250	C
1232	3	0	Vendel, Mr. Olof Edvin	male	20.0	0	0	7.8542	S
1233	3	0	Vestrom, Miss. Hulda Amanda Adolfina	female	14.0	0	0	7.8542	S
1234	3	0	Vovk, Mr. Janko	male	22.0	0	0	7.8958	S
1235	3	0	Waelens, Mr. Achille	male	22.0	0	0	9.0000	S
1236	3	0	Ware, Mr. Frederick	male	NaN	0	0	8.0500	S
1237	3	0	Warren, Mr. Charles William	male	NaN	0	0	7.5500	S
1238	3	0	Webber, Mr. James	male	NaN	0	0	8.0500	S
1239	3	0	Wenzel, Mr. Linhart	male	32.5	0	0	9.5000	S
1240	3	1	Whabee, Mrs. George Joseph (Shawneene Abi-Saab)	female	38.0	0	0	7.2292	C
1241	3	0	Widegren, Mr. Carl/Charles Peter	male	51.0	0	0	7.7500	S
1242	3	0	Wiklund, Mr. Jakob Alfred	male	18.0	1	0	6.4958	S
1243	3	0	Wiklund, Mr. Karl Johan	male	21.0	1	0	6.4958	S
1244	3	1	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	7.0000	S
1245	3	0	Willer, Mr. Aaron ("Abi Weller")	male	NaN	0	0	8.7125	S
1246	3	0	Willey, Mr. Edward	male	NaN	0	0	7.5500	S
1247	3	0	Williams, Mr. Howard Hugh "Harry"	male	NaN	0	0	8.0500	S
1248	3	0	Williams, Mr. Leslie	male	28.5	0	0	16.1000	S
1249	3	0	Windelov, Mr. Einar	male	21.0	0	0	7.2500	S
1250	3	0	Wirz, Mr. Albert	male	27.0	0	0	8.6625	S
1251	3	0	Wiseman, Mr. Phillippe	male	NaN	0	0	7.2500	S
1252	3	0	Wittevrongel, Mr. Camille	male	36.0	0	0	9.5000	S
1253	3	0	Yasbeck, Mr. Antoni	male	27.0	1	0	14.4542	C
1254	3	1	Yasbeck, Mrs. Antoni (Selini Alexander)	female	15.0	1	0	14.4542	C
1255	3	0	Youseff, Mr. Gerious	male	45.5	0	0	7.2250	C
1256	3	0	Yousif, Mr. Wazli	male	NaN	0	0	7.2250	C

1257 rows × 9 columns

In [4]:
```
data.shape
```

Out[4]:
```
(1257, 9)
```

In [5]:
```
data.head()
```

Out[5]:

	pclass	survived	name	sex	age	sibsp	parch	fare	embarked
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0	0	0	211.3375	S
1	1	0	Allison, Miss. Helen Loraine	female	2.0	1	2	151.5500	S
2	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0	1	2	151.5500	S
3	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0	1	2	151.5500	S
4	1	1	Anderson, Mr. Harry	male	48.0	0	0	26.5500	S

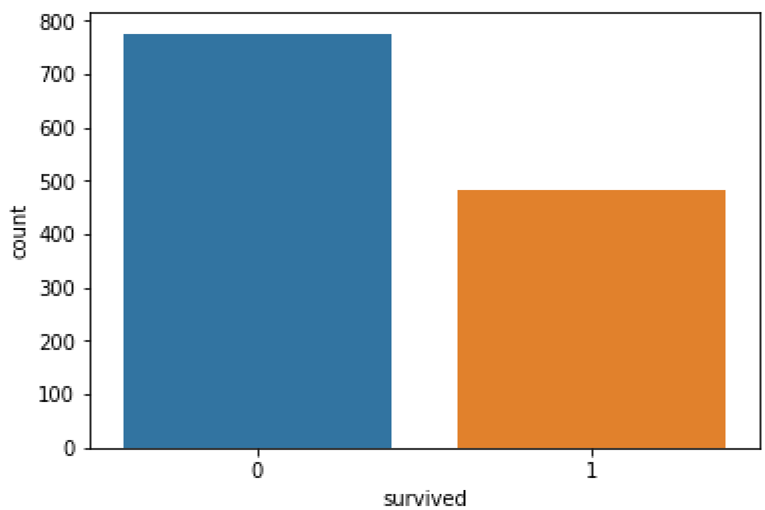
In [6]:
```
data.info()
```

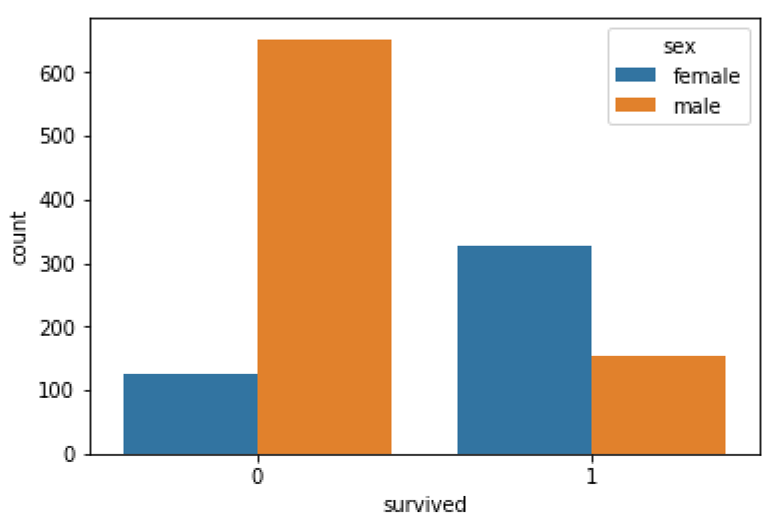
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1257 entries, 0 to 1256
Data columns (total 9 columns):
pclass      1257 non-null int64
survived    1257 non-null int64
name        1257 non-null object
sex         1257 non-null object
age         996 non-null float64
sibsp       1257 non-null int64
parch       1257 non-null int64
fare        1257 non-null float64
embarked    1257 non-null object
dtypes: float64(2), int64(4), object(3)
memory usage: 88.5+ KB
```

In [7]:
```
data.describe()
```

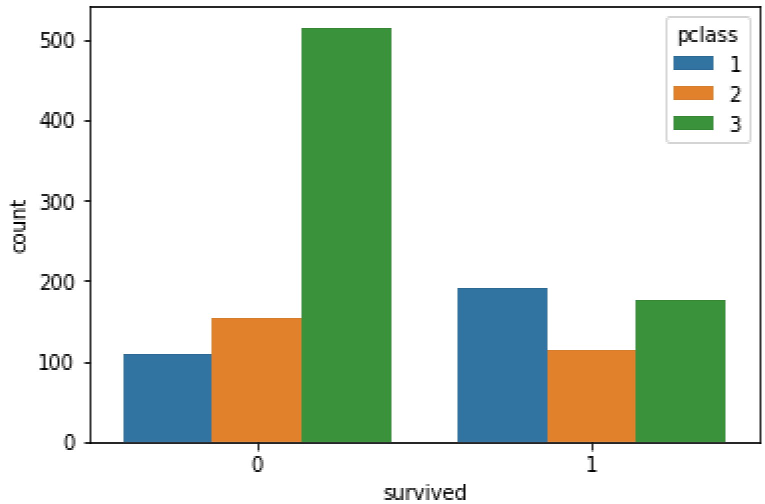
Out[7]:

	pclass	survived	age	sibsp	parch	fare
count	1257.000000	1257.000000	996.000000	1257.000000	1257.000000	1257.000000
mean	2.310263	0.382657	29.070783	0.501989	0.377884	32.720896
std	0.831791	0.486229	12.819750	1.056616	0.863035	51.127788
min	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000
25%	2.000000	0.000000	21.000000	0.000000	0.000000	7.895800
50%	3.000000	0.000000	28.000000	0.000000	0.000000	14.400000
75%	3.000000	1.000000	37.000000	1.000000	0.000000	31.000000
max	3.000000	1.000000	60.000000	8.000000	9.000000	512.329200

In [8]:
```
data.isnull().sum()
```
Out[8]:
```
pclass        0
survived      0
name          0
sex           0
age         261
sibsp         0
parch         0
fare          0
embarked      0
dtype: int64
```
In [9]:
```
import seaborn as sns
```
In [10]:
```
sns.countplot(x="survived",data=data)
```
Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x9b81780>
```


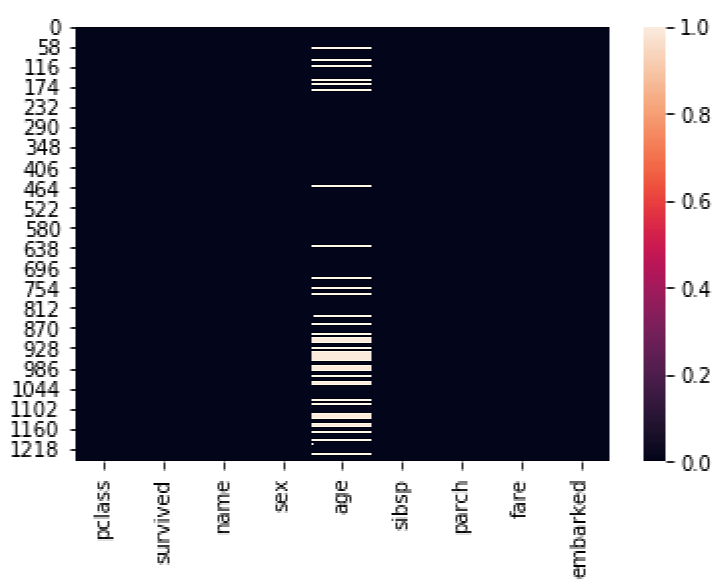
In [11]:
```
sns.countplot(x="survived",hue="sex",data=data)
```
Out[11]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xb2da0f0>
```


In [12]:
```
sns.countplot(x="survived",hue="pclass",data=data)
```
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0xb0715c0>



In [13]:
```
data.isnull().sum()
```
Out[13]:
```
pclass        0
survived      0
name          0
sex           0
age         261
sibsp         0
parch         0
fare          0
embarked      0
dtype: int64
```

In [14]:
```
sns.heatmap(data.isnull())
```
Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0xb263470>

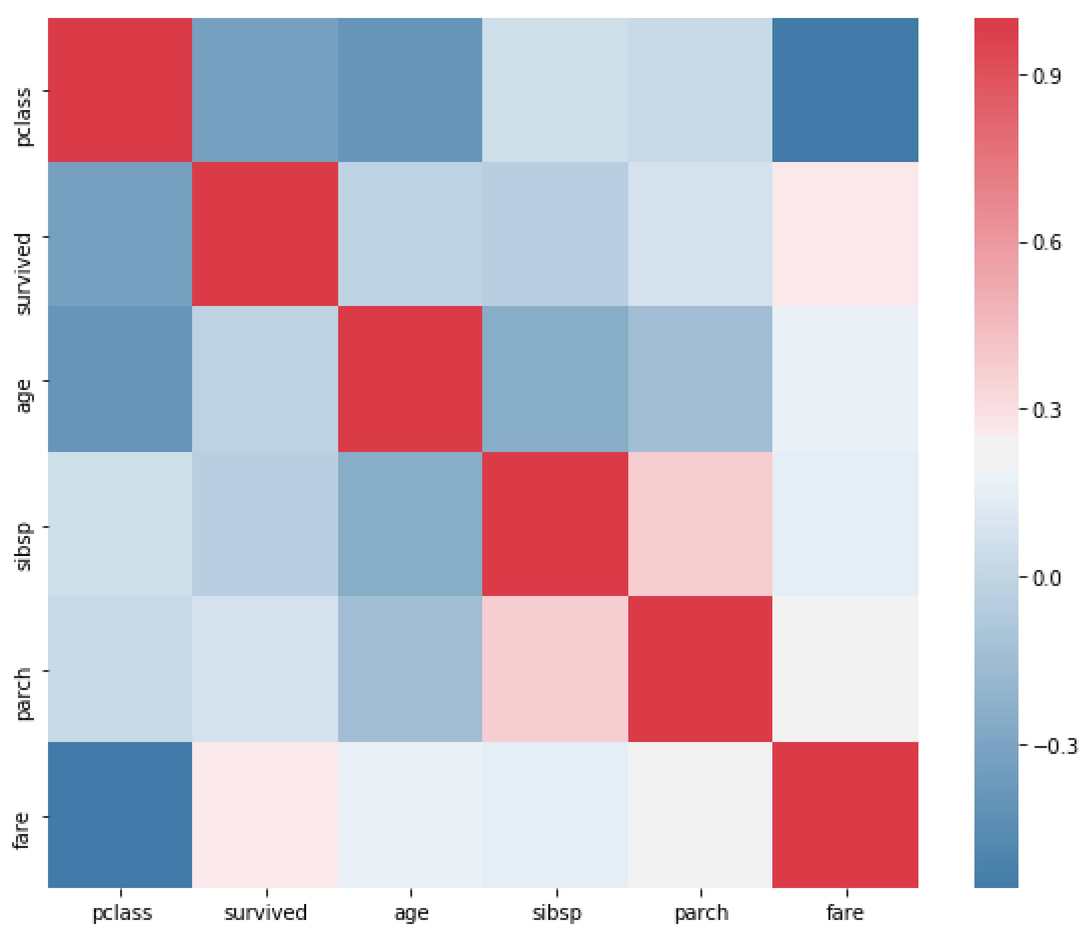


In [15]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1257 entries, 0 to 1256
Data columns (total 9 columns):
pclass      1257 non-null int64
survived    1257 non-null int64
name        1257 non-null object
sex         1257 non-null object
age         996 non-null float64
sibsp       1257 non-null int64
parch       1257 non-null int64
fare        1257 non-null float64
embarked    1257 non-null object
dtypes: float64(2), int64(4), object(3)
memory usage: 88.5+ KB
```

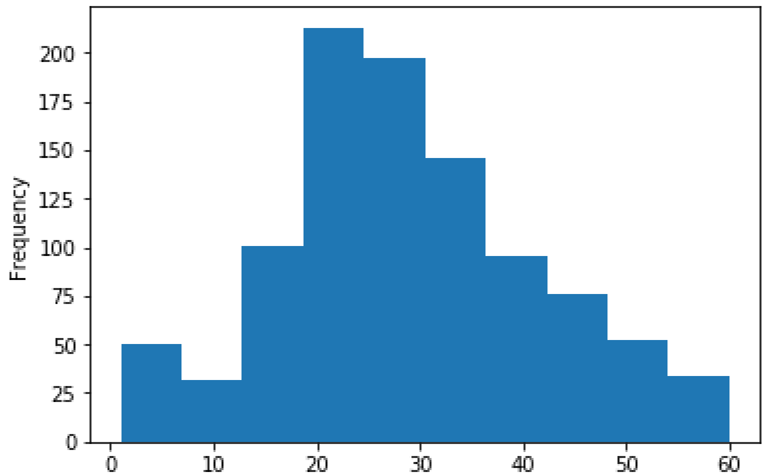
In [16]:
```
f, ax=pl.subplots(figsize=(10,8))
corr=data.corr()
sns.heatmap(corr,mask=np.zeros_like(corr,dtype=np.bool),cmap=sns.diverging_palette(240,10,
```

Out[16]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xb40f898>
```



In [17]:
```python
data['age'].plot.hist()
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0xb51fb70>

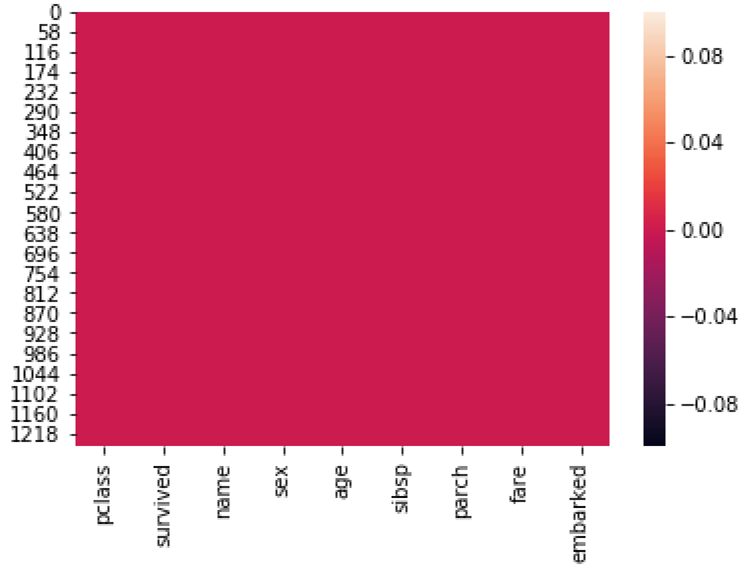


In [18]:
```python
data['age'].fillna(data['age'].mean(),inplace=True)
```

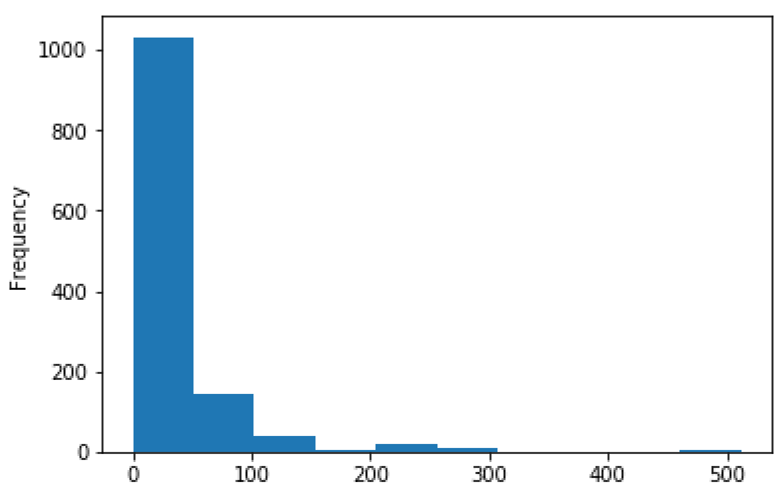
In [19]:
```python
data.isnull().sum()
```

Out[19]:
```
pclass      0
survived    0
name        0
sex         0
age         0
sibsp       0
parch       0
fare        0
embarked    0
dtype: int64
```

In [20]:
```
sns.heatmap(data.isnull())
```
Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0xb511be0>

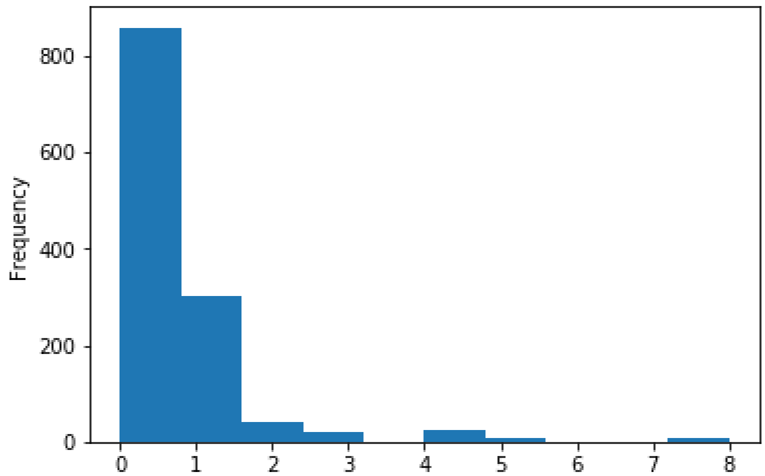


In [21]:
```
data['fare'].plot.hist()
```
Out[21]: <matplotlib.axes._subplots.AxesSubplot at 0xb84e748>



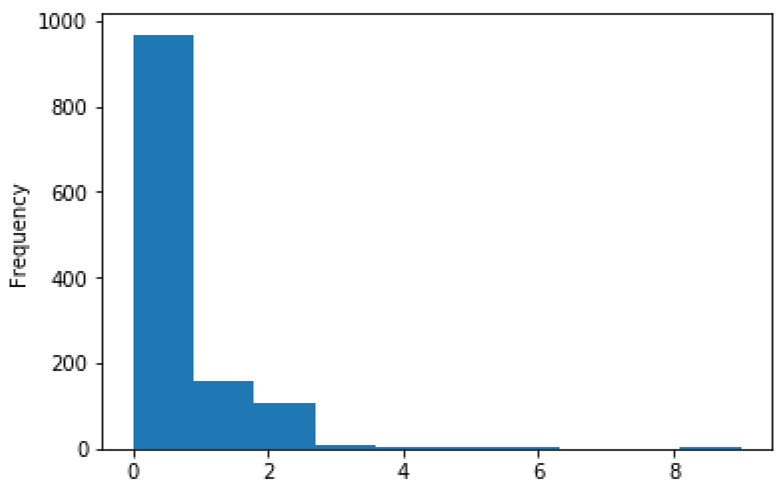
In [22]:
```
data['sibsp'].plot.hist()
```

Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0xba530b8>



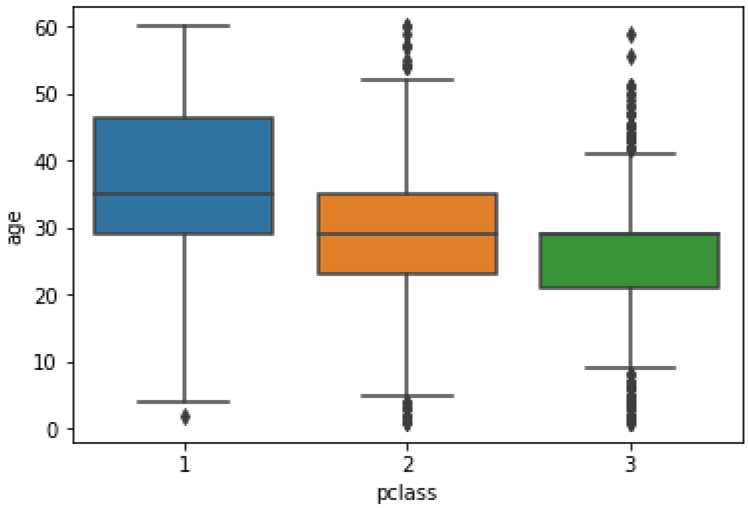
In [23]:
```
data['parch'].plot.hist()
```

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0xbad4978>



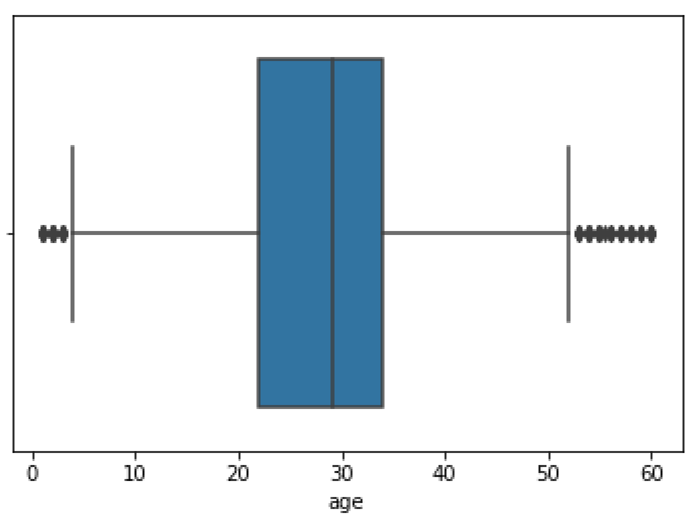
```
In [24]: sns.boxplot(x="pclass",y="age",data=data)
```

Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0xbb9d0f0>



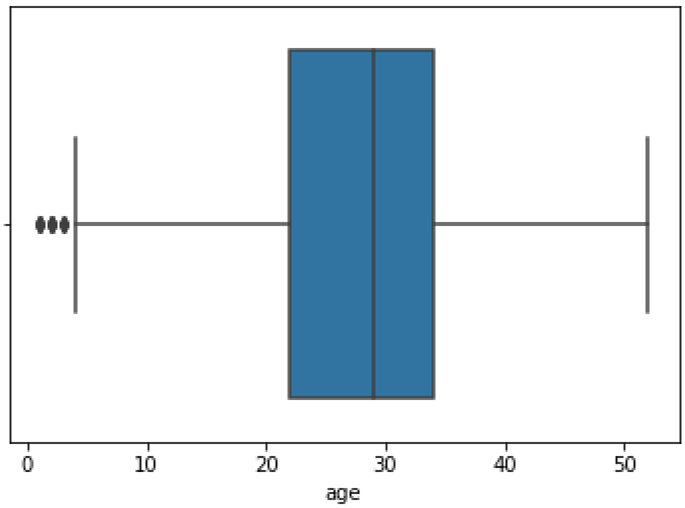

```
In [25]: sns.boxplot(x="age",data=data)
```

Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0xbb3ecf8>



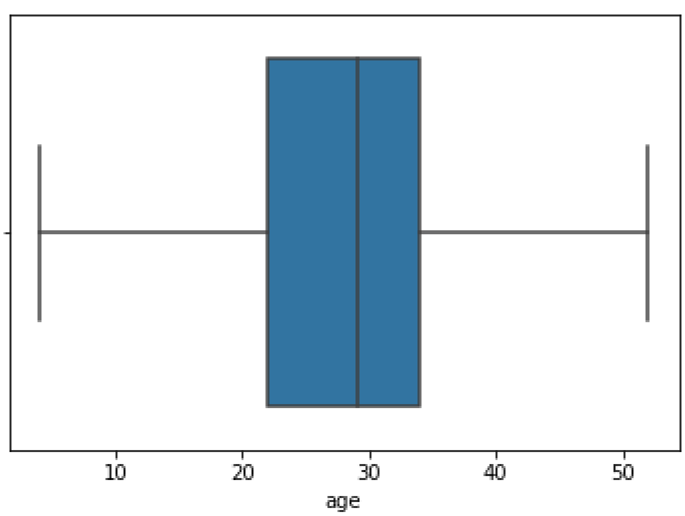

```
In [26]: data['age']=np.where(data['age']>52,51,data['age'])
```

In [27]:
```
sns.boxplot(data['age'])
```
Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0xbc84080>



In [28]:
```
data['age']=np.where(data['age']<4,4,data['age'])
```

In [29]:
```
sns.boxplot(data['age'])
```
Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0xbc0c390>



```
In [30]: data
```

Out[30]:

	pclass	survived	name	sex	age	sibsp	parch	fare	embarked
0	1	1	Allen, Miss. Elisabeth Walton	female	29.000000	0	0	211.3375	S
1	1	0	Allison, Miss. Helen Loraine	female	4.000000	1	2	151.5500	S
2	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.000000	1	2	151.5500	S
3	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.000000	1	2	151.5500	S
4	1	1	Anderson, Mr. Harry	male	48.000000	0	0	26.5500	S
5	1	0	Andrews, Mr. Thomas Jr	male	39.000000	0	0	0.0000	S
6	1	1	Appleton, Mrs. Edward Dale (Charlotte Lamson)	female	51.000000	2	0	51.4792	S
7	1	0	Astor, Col. John Jacob	male	47.000000	1	0	227.5250	C
8	1	1	Astor, Mrs. John Jacob (Madeleine Talmadge Force)	female	18.000000	1	0	227.5250	C
9	1	1	Aubart, Mme. Leontine Pauline	female	24.000000	0	0	69.3000	C
10	1	1	Barber, Miss. Ellen "Nellie"	female	26.000000	0	0	78.8500	S
11	1	0	Baumann, Mr. John D	male	29.070783	0	0	25.9250	S
12	1	0	Baxter, Mr. Quigg Edmond	male	24.000000	0	1	247.5208	C
13	1	1	Baxter, Mrs. James (Helene DeLaudeniere Chaput)	female	50.000000	0	1	247.5208	C
14	1	1	Bazzani, Miss. Albina	female	32.000000	0	0	76.2917	C
15	1	0	Beattie, Mr. Thomson	male	36.000000	0	0	75.2417	C
16	1	1	Beckwith, Mr. Richard Leonard	male	37.000000	1	1	52.5542	S
17	1	1	Beckwith, Mrs. Richard Leonard (Sallie Monypeny)	female	47.000000	1	1	52.5542	S
18	1	1	Behr, Mr. Karl Howell	male	26.000000	0	0	30.0000	C
19	1	1	Bidois, Miss. Rosalie	female	42.000000	0	0	227.5250	C
20	1	1	Bird, Miss. Ellen	female	29.000000	0	0	221.7792	S
21	1	0	Birnbaum, Mr. Jakob	male	25.000000	0	0	26.0000	C
22	1	1	Bishop, Mr. Dickinson H	male	25.000000	1	0	91.0792	C
23	1	1	Bishop, Mrs. Dickinson H (Helen Walton)	female	19.000000	1	0	91.0792	C
24	1	1	Bissette, Miss. Amelia	female	35.000000	0	0	135.6333	S
25	1	1	Bjornstrom-Steffansson, Mr. Mauritz Hakan	male	28.000000	0	0	26.5500	S
26	1	0	Blackwell, Mr. Stephen Weart	male	45.000000	0	0	35.5000	S
27	1	1	Blank, Mr. Henry	male	40.000000	0	0	31.0000	C
28	1	1	Bonnell, Miss. Caroline	female	30.000000	0	0	164.8667	S
29	1	1	Bonnell, Miss. Elizabeth	female	51.000000	0	0	26.5500	S
...	...	...	...	...	...	...	...	...	...
1227	3	0	Vander Planke, Miss. Augusta Maria	female	18.000000	2	0	18.0000	S

	pclass	survived	name	sex	age	sibsp	parch	fare	embarked
1228	3	0	Vander Planke, Mr. Julius	male	31.000000	3	0	18.0000	S
1229	3	0	Vander Planke, Mr. Leo Edmondus	male	16.000000	2	0	18.0000	S
1230	3	0	Vander Planke, Mrs. Julius (Emelia Maria Vande...	female	31.000000	1	0	18.0000	S
1231	3	1	Vartanian, Mr. David	male	22.000000	0	0	7.2250	C
1232	3	0	Vendel, Mr. Olof Edvin	male	20.000000	0	0	7.8542	S
1233	3	0	Vestrom, Miss. Hulda Amanda Adolfina	female	14.000000	0	0	7.8542	S
1234	3	0	Vovk, Mr. Janko	male	22.000000	0	0	7.8958	S
1235	3	0	Waelens, Mr. Achille	male	22.000000	0	0	9.0000	S
1236	3	0	Ware, Mr. Frederick	male	29.070783	0	0	8.0500	S
1237	3	0	Warren, Mr. Charles William	male	29.070783	0	0	7.5500	S
1238	3	0	Webber, Mr. James	male	29.070783	0	0	8.0500	S
1239	3	0	Wenzel, Mr. Linhart	male	32.500000	0	0	9.5000	S
1240	3	1	Whabee, Mrs. George Joseph (Shawneene Abi-Saab)	female	38.000000	0	0	7.2292	C
1241	3	0	Widegren, Mr. Carl/Charles Peter	male	51.000000	0	0	7.7500	S
1242	3	0	Wiklund, Mr. Jakob Alfred	male	18.000000	1	0	6.4958	S
1243	3	0	Wiklund, Mr. Karl Johan	male	21.000000	1	0	6.4958	S
1244	3	1	Wilkes, Mrs. James (Ellen Needs)	female	47.000000	1	0	7.0000	S
1245	3	0	Willer, Mr. Aaron ("Abi Weller")	male	29.070783	0	0	8.7125	S
1246	3	0	Willey, Mr. Edward	male	29.070783	0	0	7.5500	S
1247	3	0	Williams, Mr. Howard Hugh "Harry"	male	29.070783	0	0	8.0500	S
1248	3	0	Williams, Mr. Leslie	male	28.500000	0	0	16.1000	S
1249	3	0	Windelov, Mr. Einar	male	21.000000	0	0	7.2500	S
1250	3	0	Wirz, Mr. Albert	male	27.000000	0	0	8.6625	S
1251	3	0	Wiseman, Mr. Phillippe	male	29.070783	0	0	7.2500	S
1252	3	0	Wittevrongel, Mr. Camille	male	36.000000	0	0	9.5000	S
1253	3	0	Yasbeck, Mr. Antoni	male	27.000000	1	0	14.4542	C
1254	3	1	Yasbeck, Mrs. Antoni (Selini Alexander)	female	15.000000	1	0	14.4542	C
1255	3	0	Youseff, Mr. Gerious	male	45.500000	0	0	7.2250	C
1256	3	0	Yousif, Mr. Wazli	male	29.070783	0	0	7.2250	C

1257 rows × 9 columns

In [31]:
```
data=data.drop(['name'],axis=1)
```

In [32]:
```
data
```
Out[32]:

	pclass	survived	sex	age	sibsp	parch	fare	embarked
0	1	1	female	29.000000	0	0	211.3375	S
1	1	0	female	4.000000	1	2	151.5500	S
2	1	0	male	30.000000	1	2	151.5500	S
3	1	0	female	25.000000	1	2	151.5500	S
4	1	1	male	48.000000	0	0	26.5500	S
5	1	0	male	39.000000	0	0	0.0000	S
6	1	1	female	51.000000	2	0	51.4792	S
7	1	0	male	47.000000	1	0	227.5250	C
8	1	1	female	18.000000	1	0	227.5250	C
9	1	1	female	24.000000	0	0	69.3000	C
10	1	1	female	26.000000	0	0	78.8500	S
11	1	0	male	29.070783	0	0	25.9250	S
12	1	0	male	24.000000	0	1	247.5208	C
13	1	1	female	50.000000	0	1	247.5208	C
14	1	1	female	32.000000	0	0	76.2917	C
15	1	0	male	36.000000	0	0	75.2417	C
16	1	1	male	37.000000	1	1	52.5542	S
17	1	1	female	47.000000	1	1	52.5542	S
18	1	1	male	26.000000	0	0	30.0000	C
19	1	1	female	42.000000	0	0	227.5250	C
20	1	1	female	29.000000	0	0	221.7792	S
21	1	0	male	25.000000	0	0	26.0000	C
22	1	1	male	25.000000	1	0	91.0792	C
23	1	1	female	19.000000	1	0	91.0792	C
24	1	1	female	35.000000	0	0	135.6333	S
25	1	1	male	28.000000	0	0	26.5500	S
26	1	0	male	45.000000	0	0	35.5000	S
27	1	1	male	40.000000	0	0	31.0000	C
28	1	1	female	30.000000	0	0	164.8667	S
29	1	1	female	51.000000	0	0	26.5500	S
...	...	...	...	...	...	...	...	...
1227	3	0	female	18.000000	2	0	18.0000	S
1228	3	0	male	31.000000	3	0	18.0000	S
1229	3	0	male	16.000000	2	0	18.0000	S
1230	3	0	female	31.000000	1	0	18.0000	S
1231	3	1	male	22.000000	0	0	7.2250	C
1232	3	0	male	20.000000	0	0	7.8542	S

	pclass	survived	sex	age	sibsp	parch	fare	embarked
1233	3	0	female	14.000000	0	0	7.8542	S
1234	3	0	male	22.000000	0	0	7.8958	S
1235	3	0	male	22.000000	0	0	9.0000	S
1236	3	0	male	29.070783	0	0	8.0500	S
1237	3	0	male	29.070783	0	0	7.5500	S
1238	3	0	male	29.070783	0	0	8.0500	S
1239	3	0	male	32.500000	0	0	9.5000	S
1240	3	1	female	38.000000	0	0	7.2292	C
1241	3	0	male	51.000000	0	0	7.7500	S
1242	3	0	male	18.000000	1	0	6.4958	S
1243	3	0	male	21.000000	1	0	6.4958	S
1244	3	1	female	47.000000	1	0	7.0000	S
1245	3	0	male	29.070783	0	0	8.7125	S
1246	3	0	male	29.070783	0	0	7.5500	S
1247	3	0	male	29.070783	0	0	8.0500	S
1248	3	0	male	28.500000	0	0	16.1000	S
1249	3	0	male	21.000000	0	0	7.2500	S
1250	3	0	male	27.000000	0	0	8.6625	S
1251	3	0	male	29.070783	0	0	7.2500	S
1252	3	0	male	36.000000	0	0	9.5000	S
1253	3	0	male	27.000000	1	0	14.4542	C
1254	3	1	female	15.000000	1	0	14.4542	C
1255	3	0	male	45.500000	0	0	7.2250	C
1256	3	0	male	29.070783	0	0	7.2250	C

1257 rows × 8 columns

In [33]:
```
sex_dum=data['sex']=pd.get_dummies(data['sex'],drop_first=True)
```

In [34]:
```
sex_dum
```

Out[34]:

	male
0	0
1	0
2	1
3	0
4	1
5	1
6	0
7	1
8	0
9	0
10	0
11	1
12	1
13	0
14	0
15	1
16	1
17	0
18	1
19	0
20	0
21	1
22	1
23	0
24	0
25	1
26	1
27	1
28	0
29	0
...	...
1227	0
1228	1
1229	1
1230	0
1231	1
1232	1

	male
1233	0
1234	1
1235	1
1236	1
1237	1
1238	1
1239	1
1240	0
1241	1
1242	1
1243	1
1244	0
1245	1
1246	1
1247	1
1248	1
1249	1
1250	1
1251	1
1252	1
1253	1
1254	0
1255	1
1256	1

1257 rows × 1 columns

In [35]:
```
embarked_dum=data['embarked']=pd.get_dummies(data['embarked'],drop_first=True)
```

In [36]:
```
embarked_dum
```

Out[36]:

	Q	S
0	0	1
1	0	1
2	0	1
3	0	1
4	0	1
5	0	1
6	0	1
7	0	0
8	0	0
9	0	0
10	0	1
11	0	1
12	0	0
13	0	0
14	0	0
15	0	0
16	0	1
17	0	1
18	0	0
19	0	0
20	0	1
21	0	0
22	0	0
23	0	0
24	0	1
25	0	1
26	0	1
27	0	0
28	0	1
29	0	1
...	...	...
1227	0	1
1228	0	1
1229	0	1
1230	0	1
1231	0	0
1232	0	1

	Q	S
1233	0	1
1234	0	1
1235	0	1
1236	0	1
1237	0	1
1238	0	1
1239	0	1
1240	0	0
1241	0	1
1242	0	1
1243	0	1
1244	0	1
1245	0	1
1246	0	1
1247	0	1
1248	0	1
1249	0	1
1250	0	1
1251	0	1
1252	0	1
1253	0	0
1254	0	0
1255	0	0
1256	0	0

1257 rows × 2 columns

In [37]:
```
pclass_dum=data['pclass']=pd.get_dummies(data['pclass'],drop_first=True)
```

In [38]:
```
pclass_dum
```

Out[38]:

	2	3
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	0	0
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0
13	0	0
14	0	0
15	0	0
16	0	0
17	0	0
18	0	0
19	0	0
20	0	0
21	0	0
22	0	0
23	0	0
24	0	0
25	0	0
26	0	0
27	0	0
28	0	0
29	0	0
...	...	...
1227	0	1
1228	0	1
1229	0	1
1230	0	1
1231	0	1
1232	0	1

	2	3
1233	0	1
1234	0	1
1235	0	1
1236	0	1
1237	0	1
1238	0	1
1239	0	1
1240	0	1
1241	0	1
1242	0	1
1243	0	1
1244	0	1
1245	0	1
1246	0	1
1247	0	1
1248	0	1
1249	0	1
1250	0	1
1251	0	1
1252	0	1
1253	0	1
1254	0	1
1255	0	1
1256	0	1

1257 rows × 2 columns

In [39]:
```
data=pd.concat([data,sex_dum,embarked_dum,pclass_dum],axis=1)
```

In [40]:
```
data
```
Out[40]:

	pclass	survived	sex	age	sibsp	parch	fare	embarked	male	Q	S	2	3
0	0	1	0	29.000000	0	0	211.3375	0	0	0	1	0	0
1	0	0	0	4.000000	1	2	151.5500	0	0	0	1	0	0
2	0	0	1	30.000000	1	2	151.5500	0	1	0	1	0	0
3	0	0	0	25.000000	1	2	151.5500	0	0	0	1	0	0
4	0	1	1	48.000000	0	0	26.5500	0	1	0	1	0	0
5	0	0	1	39.000000	0	0	0.0000	0	1	0	1	0	0
6	0	1	0	51.000000	2	0	51.4792	0	0	0	1	0	0
7	0	0	1	47.000000	1	0	227.5250	0	1	0	0	0	0
8	0	1	0	18.000000	1	0	227.5250	0	0	0	0	0	0
9	0	1	0	24.000000	0	0	69.3000	0	0	0	0	0	0
10	0	1	0	26.000000	0	0	78.8500	0	0	0	1	0	0
11	0	0	1	29.070783	0	0	25.9250	0	1	0	1	0	0
12	0	0	1	24.000000	0	1	247.5208	0	1	0	0	0	0
13	0	1	0	50.000000	0	1	247.5208	0	0	0	0	0	0
14	0	1	0	32.000000	0	0	76.2917	0	0	0	0	0	0
15	0	0	1	36.000000	0	0	75.2417	0	1	0	0	0	0
16	0	1	1	37.000000	1	1	52.5542	0	1	0	1	0	0
17	0	1	0	47.000000	1	1	52.5542	0	0	0	1	0	0
18	0	1	1	26.000000	0	0	30.0000	0	1	0	0	0	0
19	0	1	0	42.000000	0	0	227.5250	0	0	0	0	0	0
20	0	1	0	29.000000	0	0	221.7792	0	0	0	1	0	0
21	0	0	1	25.000000	0	0	26.0000	0	1	0	0	0	0
22	0	1	1	25.000000	1	0	91.0792	0	1	0	0	0	0
23	0	1	0	19.000000	1	0	91.0792	0	0	0	0	0	0
24	0	1	0	35.000000	0	0	135.6333	0	0	0	1	0	0
25	0	1	1	28.000000	0	0	26.5500	0	1	0	1	0	0
26	0	0	1	45.000000	0	0	35.5000	0	1	0	1	0	0
27	0	1	1	40.000000	0	0	31.0000	0	1	0	0	0	0
28	0	1	0	30.000000	0	0	164.8667	0	0	0	1	0	0
29	0	1	0	51.000000	0	0	26.5500	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1227	0	0	0	18.000000	2	0	18.0000	0	0	0	1	0	1
1228	0	0	1	31.000000	3	0	18.0000	0	1	0	1	0	1
1229	0	0	1	16.000000	2	0	18.0000	0	1	0	1	0	1
1230	0	0	0	31.000000	1	0	18.0000	0	0	0	1	0	1
1231	0	1	1	22.000000	0	0	7.2250	0	1	0	0	0	1
1232	0	0	1	20.000000	0	0	7.8542	0	1	0	1	0	1

	pclass	survived	sex	age	sibsp	parch	fare	embarked	male	Q	S	2	3
1233	0	0	0	14.000000	0	0	7.8542	0	0	0	1	0	1
1234	0	0	1	22.000000	0	0	7.8958	0	1	0	1	0	1
1235	0	0	1	22.000000	0	0	9.0000	0	1	0	1	0	1
1236	0	0	1	29.070783	0	0	8.0500	0	1	0	1	0	1
1237	0	0	1	29.070783	0	0	7.5500	0	1	0	1	0	1
1238	0	0	1	29.070783	0	0	8.0500	0	1	0	1	0	1
1239	0	0	1	32.500000	0	0	9.5000	0	1	0	1	0	1
1240	0	1	0	38.000000	0	0	7.2292	0	0	0	0	0	1
1241	0	0	1	51.000000	0	0	7.7500	0	1	0	1	0	1
1242	0	0	1	18.000000	1	0	6.4958	0	1	0	1	0	1
1243	0	0	1	21.000000	1	0	6.4958	0	1	0	1	0	1
1244	0	1	0	47.000000	1	0	7.0000	0	0	0	1	0	1
1245	0	0	1	29.070783	0	0	8.7125	0	1	0	1	0	1
1246	0	0	1	29.070783	0	0	7.5500	0	1	0	1	0	1
1247	0	0	1	29.070783	0	0	8.0500	0	1	0	1	0	1
1248	0	0	1	28.500000	0	0	16.1000	0	1	0	1	0	1
1249	0	0	1	21.000000	0	0	7.2500	0	1	0	1	0	1
1250	0	0	1	27.000000	0	0	8.6625	0	1	0	1	0	1
1251	0	0	1	29.070783	0	0	7.2500	0	1	0	1	0	1
1252	0	0	1	36.000000	0	0	9.5000	0	1	0	1	0	1
1253	0	0	1	27.000000	1	0	14.4542	0	1	0	0	0	1
1254	0	1	0	15.000000	1	0	14.4542	0	0	0	0	0	1
1255	0	0	1	45.500000	0	0	7.2250	0	1	0	0	0	1
1256	0	0	1	29.070783	0	0	7.2250	0	1	0	0	0	1

1257 rows × 13 columns

In [41]:
```
data=data.drop(['sex','embarked','pclass'],axis=1)
```

```
In [42]: data
```

Out[42]:

	survived	age	sibsp	parch	fare	male	Q	S	2	3
0	1	29.000000	0	0	211.3375	0	0	1	0	0
1	0	4.000000	1	2	151.5500	0	0	1	0	0
2	0	30.000000	1	2	151.5500	1	0	1	0	0
3	0	25.000000	1	2	151.5500	0	0	1	0	0
4	1	48.000000	0	0	26.5500	1	0	1	0	0
5	0	39.000000	0	0	0.0000	1	0	1	0	0
6	1	51.000000	2	0	51.4792	0	0	1	0	0
7	0	47.000000	1	0	227.5250	1	0	0	0	0
8	1	18.000000	1	0	227.5250	0	0	0	0	0
9	1	24.000000	0	0	69.3000	0	0	0	0	0
10	1	26.000000	0	0	78.8500	0	0	1	0	0
11	0	29.070783	0	0	25.9250	1	0	1	0	0
12	0	24.000000	0	1	247.5208	1	0	0	0	0
13	1	50.000000	0	1	247.5208	0	0	0	0	0
14	1	32.000000	0	0	76.2917	0	0	0	0	0
15	0	36.000000	0	0	75.2417	1	0	0	0	0
16	1	37.000000	1	1	52.5542	1	0	1	0	0
17	1	47.000000	1	1	52.5542	0	0	1	0	0
18	1	26.000000	0	0	30.0000	1	0	0	0	0
19	1	42.000000	0	0	227.5250	0	0	0	0	0
20	1	29.000000	0	0	221.7792	0	0	1	0	0
21	0	25.000000	0	0	26.0000	1	0	0	0	0
22	1	25.000000	1	0	91.0792	1	0	0	0	0
23	1	19.000000	1	0	91.0792	0	0	0	0	0
24	1	35.000000	0	0	135.6333	0	0	1	0	0
25	1	28.000000	0	0	26.5500	1	0	1	0	0
26	0	45.000000	0	0	35.5000	1	0	1	0	0
27	1	40.000000	0	0	31.0000	1	0	0	0	0
28	1	30.000000	0	0	164.8667	0	0	1	0	0
29	1	51.000000	0	0	26.5500	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...
1227	0	18.000000	2	0	18.0000	0	0	1	0	1
1228	0	31.000000	3	0	18.0000	1	0	1	0	1
1229	0	16.000000	2	0	18.0000	1	0	1	0	1
1230	0	31.000000	1	0	18.0000	0	0	1	0	1
1231	1	22.000000	0	0	7.2250	1	0	0	0	1
1232	0	20.000000	0	0	7.8542	1	0	1	0	1

	survived	age	sibsp	parch	fare	male	Q	S	2	3
1233	0	14.000000	0	0	7.8542	0	0	1	0	1
1234	0	22.000000	0	0	7.8958	1	0	1	0	1
1235	0	22.000000	0	0	9.0000	1	0	1	0	1
1236	0	29.070783	0	0	8.0500	1	0	1	0	1
1237	0	29.070783	0	0	7.5500	1	0	1	0	1
1238	0	29.070783	0	0	8.0500	1	0	1	0	1
1239	0	32.500000	0	0	9.5000	1	0	1	0	1
1240	1	38.000000	0	0	7.2292	0	0	0	0	1
1241	0	51.000000	0	0	7.7500	1	0	1	0	1
1242	0	18.000000	1	0	6.4958	1	0	1	0	1
1243	0	21.000000	1	0	6.4958	1	0	1	0	1
1244	1	47.000000	1	0	7.0000	0	0	1	0	1
1245	0	29.070783	0	0	8.7125	1	0	1	0	1
1246	0	29.070783	0	0	7.5500	1	0	1	0	1
1247	0	29.070783	0	0	8.0500	1	0	1	0	1
1248	0	28.500000	0	0	16.1000	1	0	1	0	1
1249	0	21.000000	0	0	7.2500	1	0	1	0	1
1250	0	27.000000	0	0	8.6625	1	0	1	0	1
1251	0	29.070783	0	0	7.2500	1	0	1	0	1
1252	0	36.000000	0	0	9.5000	1	0	1	0	1
1253	0	27.000000	1	0	14.4542	1	0	0	0	1
1254	1	15.000000	1	0	14.4542	0	0	0	0	1
1255	0	45.500000	0	0	7.2250	1	0	0	0	1
1256	0	29.070783	0	0	7.2250	1	0	0	0	1

1257 rows × 10 columns

In [43]:
```
x=data.drop(['survived'],axis=1)
```

In [44]:
```
y=data.survived
```

In [45]:
```
from sklearn import preprocessing
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import cohen_kappa_score as kappa
from sklearn.metrics import confusion_matrix
from sklearn import metrics
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

In [46]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=10)
```

In [47]:
```
from sklearn.linear_model import LinearRegression
```

In [48]:
```
 classifier=(LogisticRegression())
    #fitting training data to the model
```

In [49]:
```
classifier.fit(x_train,y_train)
```

Out[49]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

In [50]:
```
y_pred=classifier.predict(x_test)
```

In [51]:
```python
print(list(zip(y_test,y_pred)))
```

```
[(1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0),
(0, 0), (0, 1), (0, 1), (0, 0), (0, 1), (1, 1), (1, 1), (0, 0), (0, 0), (1, 1), (0, 0),
(0, 0), (1, 0), (0, 0), (0, 0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 1), (0, 0), (1, 0),
(0, 1), (1, 0), (0, 1), (1, 1), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0),
(0, 0), (0, 0), (1, 0), (1, 1), (1, 0), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0),
(0, 1), (1, 1), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (1, 1), (1, 1),
(1, 1), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 1), (0, 0), (0, 0), (1, 1), (1, 1),
(1, 1), (0, 0), (0, 0), (0, 1), (1, 1), (1, 1), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1),
(0, 0), (1, 1), (0, 0), (1, 1), (1, 0), (0, 0), (0, 0), (1, 1), (0, 1), (1, 1), (0, 0),
(0, 0), (1, 1), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 1), (1, 0), (0, 0), (0, 1),
(0, 0), (1, 1), (0, 1), (0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (0, 1), (1, 1), (0, 0),
(1, 1), (0, 0), (1, 1), (0, 1), (1, 1), (0, 0), (0, 0), (1, 0), (1, 1), (1, 1), (1, 1),
(0, 0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 1), (1, 1), (1, 1),
(1, 0), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (1, 0), (0, 0), (1, 1), (0, 1),
(1, 0), (0, 0), (0, 0), (0, 0), (0, 1), (0, 0), (0, 0), (1, 1), (1, 1), (1, 1), (0, 0),
(1, 1), (1, 1), (1, 1), (0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (0, 0), (1, 1), (0, 0),
(0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1),
(1, 1), (0, 0), (1, 1), (1, 0), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 1), (0, 0),
(1, 1), (1, 1), (1, 0), (1, 1), (0, 0), (0, 1), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0),
(0, 0), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (1, 1), (1, 1), (0, 1), (1, 0),
(0, 0), (1, 1), (1, 1), (0, 0), (0, 0), (0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 0),
(0, 0), (0, 0), (0, 0), (1, 0), (1, 1), (0, 0), (0, 1), (1, 0), (1, 1), (1, 0), (1, 1),
(1, 1), (0, 0), (1, 1), (0, 0), (1, 0), (0, 0), (0, 0), (0, 1), (1, 0), (0, 0), (0, 1),
(0, 0), (1, 1), (0, 0), (0, 0), (0, 1), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0),
(0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (1, 1),
(0, 0), (0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (0, 0), (0, 1), (0, 0),
(0, 0), (0, 1), (0, 0), (0, 1), (1, 1), (0, 0), (1, 1), (1, 1), (0, 1), (1, 0), (1, 0),
(0, 0), (0, 0), (0, 0), (1, 0), (0, 0), (1, 1), (0, 0), (1, 1), (0, 0), (0, 0), (1, 1),
(1, 1), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (0, 0), (0, 1), (1, 1), (1, 1), (0, 0),
(0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1), (1, 1),
(1, 0), (1, 1), (0, 0), (1, 1), (0, 0), (1, 1), (1, 0), (0, 0), (0, 0), (0, 0), (1, 1),
(1, 1), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (1, 1),
(0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (1, 1), (1, 0), (0, 0), (0, 1), (0, 0), (1, 1),
(1, 1), (1, 0), (0, 0), (0, 1), (1, 1), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0),
(0, 0), (0, 0), (1, 0), (0, 0)]
```

In [52]:
```python
from sklearn.metrics import confusion_matrix,accuracy_score
```

In [53]:
```python
confusion_matrix=confusion_matrix(y_test,y_pred)
```

In [54]:
```python
print(confusion_matrix)
```

```
[[202  35]
 [ 36 105]]
```

In [55]:
```python
accuracy_score=accuracy_score(y_test,y_pred)
```

In [56]:
```python
print("Accuracy of the model:",accuracy_score)
```

```
Accuracy of the model: 0.8121693121693122
```

In [57]:
```python
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
```

In [58]:
```python
cfm=confusion_matrix(y_test,y_pred)
```

In [59]:
```
print(cfm)
```
```
[[202  35]
 [ 36 105]]
```

In [60]:
```
print("classification_report:")
```
```
classification_report:
```

In [61]:
```
print(classification_report(y_test,y_pred))
```
```
              precision    recall  f1-score   support

           0       0.85      0.85      0.85       237
           1       0.75      0.74      0.75       141

   micro avg       0.81      0.81      0.81       378
   macro avg       0.80      0.80      0.80       378
weighted avg       0.81      0.81      0.81       378
```

In [62]:
```
acc=accuracy_score(y_test,y_pred)
```

In [63]:
```
print('acc',acc)
```
```
acc 0.8121693121693122
```

In [64]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [65]:
```
model_RandomForest=RandomForestClassifier(501)
```

In [66]:
```
model_RandomForest.fit(x_train,y_train)
```

Out[66]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=501, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

In [67]:
```
y_pred=model_RandomForest.predict(x_test)
```

In [68]:
```
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
```

In [69]:
```
print(confusion_matrix(y_test,y_pred))
```
```
[[195  42]
 [ 44  97]]
```

In [70]:
```
print(accuracy_score(y_test,y_pred))
```
```
0.7724867724867724
```

In [71]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.82      0.82      0.82       237
           1       0.70      0.69      0.69       141

   micro avg       0.77      0.77      0.77       378
   macro avg       0.76      0.76      0.76       378
weighted avg       0.77      0.77      0.77       378
```

In [72]:
```
print('acc',acc)
```

```
acc 0.8121693121693122
```

In [73]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [74]:
```
model_DecisionTree=DecisionTreeClassifier()
```

In [75]:
```
model_DecisionTree.fit(x_train,y_train)
```

Out[75]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

In [76]:
```
y_pred=model_DecisionTree.predict(x_test)
```

In [77]:
```
print(list(zip(y_test,y_pred)))
```

```
[(1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 1), (0, 1), (0, 1), (0, 0), (0, 0), (1, 1),
(0, 0), (0, 1), (0, 1), (0, 0), (0, 1), (1, 1), (1, 1), (0, 0), (0, 0), (1, 1), (0, 0),
(0, 0), (1, 1), (0, 0), (0, 0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 1), (0, 0), (1, 0),
(0, 0), (1, 0), (0, 1), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1),
(0, 0), (0, 0), (1, 1), (1, 1), (1, 0), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1),
(0, 1), (1, 1), (1, 0), (0, 0), (0, 1), (0, 0), (1, 1), (1, 1), (0, 1), (1, 1), (1, 1),
(1, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 1), (0, 0), (0, 0), (1, 1), (1, 0),
(1, 0), (0, 0), (0, 0), (0, 0), (1, 1), (1, 1), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1),
(0, 0), (1, 1), (0, 0), (1, 0), (1, 1), (0, 0), (0, 0), (1, 0), (0, 1), (1, 1), (0, 0),
(0, 0), (1, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 1), (1, 0), (0, 0), (0, 1),
(0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (0, 1), (1, 1), (0, 1),
(1, 1), (0, 0), (1, 0), (0, 1), (1, 1), (0, 0), (0, 0), (1, 1), (1, 1), (1, 1), (1, 1),
(0, 0), (0, 1), (0, 1), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 1), (1, 1), (1, 1),
(1, 0), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (1, 0), (0, 0), (1, 1), (0, 1),
(1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (1, 1), (1, 1), (0, 0),
(1, 1), (1, 0), (1, 1), (0, 0), (0, 0), (1, 1), (1, 0), (0, 0), (0, 0), (1, 1), (0, 0),
(0, 0), (0, 0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 1), (0, 0), (0, 0), (1, 0),
(1, 1), (0, 1), (1, 1), (1, 0), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0),
(1, 1), (1, 1), (1, 0), (1, 1), (0, 0), (0, 1), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0),
(0, 0), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (1, 0), (1, 1), (0, 1), (1, 0),
(0, 0), (1, 1), (1, 1), (0, 0), (0, 0), (0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 1),
(0, 0), (0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (0, 1), (1, 0), (1, 0), (1, 0), (1, 0),
(1, 1), (0, 0), (1, 1), (0, 0), (1, 1), (0, 0), (0, 0), (0, 1), (1, 0), (0, 1), (0, 0),
(0, 0), (1, 0), (0, 0), (0, 1), (0, 1), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0),
(0, 1), (1, 0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1), (0, 1), (1, 1), (1, 1),
(0, 1), (0, 0), (1, 0), (0, 0), (0, 1), (0, 0), (0, 1), (1, 0), (0, 0), (0, 1), (0, 0),
(0, 1), (0, 1), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (1, 1), (0, 1), (1, 0), (1, 1),
(0, 0), (0, 0), (0, 1), (1, 0), (0, 1), (1, 1), (0, 0), (1, 1), (0, 0), (0, 0), (1, 1),
(1, 1), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (0, 0), (0, 0), (1, 1), (1, 1), (0, 0),
(0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1), (1, 1),
(1, 0), (1, 1), (0, 0), (1, 1), (0, 0), (1, 1), (1, 1), (0, 0), (0, 0), (0, 0), (1, 0),
(1, 1), (0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (1, 0),
(0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (1, 1), (1, 0), (0, 0), (0, 1), (0, 0), (1, 1),
(1, 1), (1, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (1, 1), (0, 0), (0, 1), (0, 1),
(0, 1), (0, 0), (1, 0), (0, 0)]
```

In [78]:
```
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
```

In [79]:
```
#confusion matrix
print(confusion_matrix(y_test,y_pred))
```

```
[[187  50]
 [ 44  97]]
```

In [80]:
```
print(accuracy_score(y_test,y_pred))
```

```
0.7513227513227513
```

In [81]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.81      0.79      0.80       237
           1       0.66      0.69      0.67       141

   micro avg       0.75      0.75      0.75       378
   macro avg       0.73      0.74      0.74       378
weighted avg       0.75      0.75      0.75       378
```

In [82]:
```
from sklearn import tree
```

In [83]:
```
with open("model_DecisionTree.txt","w")as f:
    f=tree.export_graphviz(model_DecisionTree,out_file=f)
```

In [84]:
```
#http://www.webgraphviz.com
#go to C drive->Users->Admin->open model_DecisionTree(txt doc)->copy and paste the text on
#DecisionTree will be formed
```

In [85]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [86]:
```
model_RandomForest=RandomForestClassifier(501)
```

In [87]:
```
model_RandomForest.fit(x_train,y_train)
```

Out[87]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=501, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

In [88]:
```
y_pred=model_RandomForest.predict(x_test)
```

In [89]:
```
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
```

In [90]:
```
print(confusion_matrix(y_test,y_pred))
```

```
[[194  43]
 [ 44  97]]
```

In [91]:
```
print(accuracy_score(y_test,y_pred))
```

```
0.7698412698412699
```

In [92]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.82      0.82      0.82       237
           1       0.69      0.69      0.69       141

   micro avg       0.77      0.77      0.77       378
   macro avg       0.75      0.75      0.75       378
weighted avg       0.77      0.77      0.77       378
```

In [93]:
```
from sklearn.ensemble import ExtraTreesClassifier
model=(ExtraTreesClassifier(10))
model=model.fit(x_train,y_train)
y_pred=model.predict(x_test)
#confusion matrix
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
confusion_matrix=confusion_matrix(y_test,y_pred)
print(confusion_matrix)
print(accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
[[203  34]
 [ 47  94]]
0.7857142857142857
              precision    recall  f1-score   support

           0       0.81      0.86      0.83       237
           1       0.73      0.67      0.70       141

   micro avg       0.79      0.79      0.79       378
   macro avg       0.77      0.76      0.77       378
weighted avg       0.78      0.79      0.78       378
```

In [94]:
```
from sklearn.ensemble import GradientBoostingClassifier
model_GradientBoosting=(GradientBoostingClassifier())
model_GradientBoosting.fit(x_train,y_train)
y_pred=model_GradientBoosting.predict(x_test)
#confusion matrix
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
confusion_matrix=confusion_matrix(y_test,y_pred)
print(confusion_matrix)
print(accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
[[210  27]
 [ 43  98]]
0.8148148148148148
              precision    recall  f1-score   support

           0       0.83      0.89      0.86       237
           1       0.78      0.70      0.74       141

   micro avg       0.81      0.81      0.81       378
   macro avg       0.81      0.79      0.80       378
weighted avg       0.81      0.81      0.81       378
```

In [96]:

```python
from sklearn import svm
svc_model=svm.SVC(kernel='rbf',C=1.0,gamma=0.1)
svc_model.fit(x_train,y_train)
```

Out[96]:

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

In [98]:

```python
y_pred=svc_model.predict(x_test)
print(list(zip(y_test,y_pred)))
```

```
[(1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 1), (0, 0), (1, 0),
(0, 0), (0, 1), (0, 1), (0, 0), (0, 0), (1, 0), (1, 0), (0, 0), (0, 0), (1, 1), (0, 0),
(0, 0), (1, 0), (0, 1), (0, 0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0),
(0, 0), (1, 0), (0, 1), (1, 0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1),
(0, 0), (0, 1), (1, 1), (1, 1), (1, 0), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1),
(0, 0), (1, 0), (1, 1), (0, 0), (0, 1), (0, 0), (1, 1), (1, 1), (0, 0), (1, 1), (1, 0),
(1, 0), (0, 0), (0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1), (1, 0),
(1, 1), (0, 1), (0, 0), (0, 1), (1, 1), (1, 1), (1, 0), (0, 0), (0, 1), (0, 0), (1, 0),
(0, 0), (1, 0), (0, 1), (1, 0), (1, 1), (0, 0), (0, 0), (1, 1), (0, 1), (1, 0), (0, 0),
(0, 0), (1, 0), (0, 0), (0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (1, 0), (0, 0), (0, 1),
(0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (0, 0), (1, 1), (0, 0),
(1, 1), (0, 0), (1, 0), (0, 1), (1, 0), (0, 0), (0, 0), (1, 1), (1, 1), (1, 1), (1, 0),
(0, 0), (0, 0), (0, 0), (0, 1), (0, 0), (0, 1), (1, 0), (0, 0), (0, 0), (1, 0), (1, 0),
(1, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (1, 0), (0, 0), (1, 1), (0, 0),
(1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 1), (0, 0), (1, 0), (1, 0), (1, 1), (0, 0),
(1, 0), (1, 0), (1, 1), (0, 0), (0, 0), (1, 1), (1, 0), (0, 0), (0, 0), (1, 0), (0, 0),
(0, 0), (0, 0), (0, 1), (0, 0), (0, 1), (0, 0), (0, 0), (0, 1), (0, 0), (0, 0), (1, 0),
(1, 1), (0, 0), (1, 1), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0),
(1, 1), (1, 1), (1, 0), (1, 1), (0, 0), (0, 0), (0, 1), (1, 1), (0, 0), (0, 0), (0, 1),
(0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (0, 0), (1, 1), (1, 0), (1, 0), (0, 0), (1, 0),
(0, 0), (1, 0), (1, 1), (0, 1), (0, 0), (0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 1),
(0, 0), (0, 0), (0, 0), (1, 0), (1, 0), (0, 1), (0, 1), (1, 0), (1, 1), (1, 0), (1, 0),
(1, 1), (0, 0), (1, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0),
(0, 1), (1, 0), (0, 1), (0, 1), (0, 0), (1, 0), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0),
(0, 1), (1, 1), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 1), (0, 0), (1, 0), (1, 0),
(0, 1), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (0, 0), (0, 1), (0, 0),
(0, 1), (0, 0), (0, 0), (0, 1), (1, 1), (0, 0), (1, 1), (1, 0), (0, 1), (1, 0), (1, 0),
(0, 0), (0, 0), (0, 1), (1, 0), (0, 0), (1, 1), (0, 0), (1, 1), (0, 0), (0, 0), (1, 0),
(1, 0), (0, 0), (0, 0), (1, 1), (0, 0), (1, 1), (0, 0), (0, 1), (1, 1), (1, 1), (0, 1),
(0, 0), (0, 1), (0, 0), (1, 0), (0, 0), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1), (1, 1),
(1, 0), (1, 0), (0, 0), (1, 1), (0, 0), (1, 1), (1, 1), (0, 0), (0, 0), (0, 0), (1, 1),
(1, 0), (0, 1), (1, 0), (0, 1), (0, 0), (0, 0), (0, 0), (0, 0), (0, 0), (1, 0), (1, 0),
(0, 0), (0, 0), (1, 1), (1, 1), (0, 0), (1, 1), (1, 0), (0, 0), (0, 0), (0, 0), (1, 1),
(1, 1), (1, 0), (0, 0), (0, 0), (1, 1), (0, 0), (0, 0), (1, 1), (0, 1), (0, 1), (0, 0),
(0, 1), (0, 0), (1, 1), (0, 0)]
```

In [99]:

```python
from sklearn .metrics import confusion_matrix,accuracy_score
confusion_matrix=confusion_matrix(y_test,y_pred)
print(confusion_matrix)
accuracy_score=accuracy_score(y_test,y_pred)
print("Accuracy of the model:",accuracy_score)
```

```
[[190  47]
 [ 72  69]]
Accuracy of the model: 0.6851851851851852
```

```
In [ ]:
```