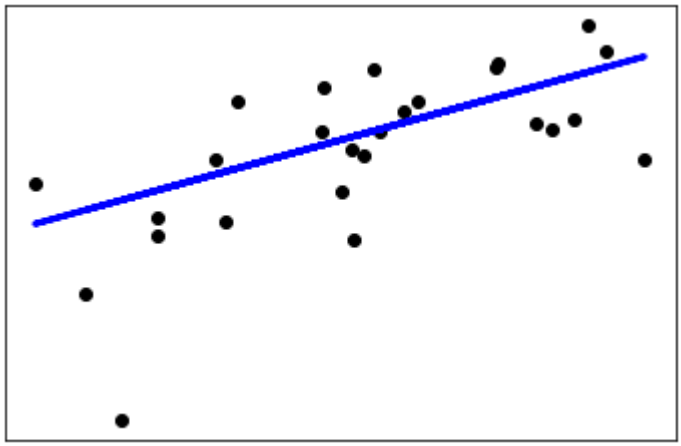


In [ ]:

Assignment-03

Download the dataset in CSV or XLSX format and perform all the operation that discussed in the class like boxplot countplot, do exploratory data analysis and and apply all the alorightma that discussed in class divide the same data set in train and test model.

In [2]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
```

In [3]:
```
dataset = pd.read_csv("FILES/titanic.csv")

dataset.head()
```

Out[3]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	S
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	S
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	S
3	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	C22 C26	S
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	C22 C26	S

In [4]:
```
dataset.columns
```

Out[4]:
```
Index(['pclass', 'survived', 'name', 'sex', 'age', 'sibsp', 'parch', 'ticket',
       'fare', 'cabin', 'embarked', 'boat', 'body', 'home.dest'],
      dtype='object')
```

In [5]:
```
dataset.isnull().sum()
```

Out[5]:
```
pclass          0
survived        0
name            0
sex             0
age           263
sibsp           0
parch           0
ticket          0
fare            1
cabin        1014
embarked        2
boat          823
body         1188
home.dest     564
dtype: int64
```

In [6]:
```
dataset.shape
```

Out[6]:
```
(1309, 14)
```

In [7]:
```
dataset['sex'] = dataset['sex'].map(lambda x: 1 if x=='male' else 0)
```

In [8]:
```
data = dataset.loc[:,['pclass','sex','age','survived']]
```

In [9]:
```
data.fillna(value=data.age.mean(),axis=0,inplace=True)
```

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

Out[10]:
```
pclass      0
sex         0
age         0
survived    0
dtype: int64
```

In [11]:
```
data.describe()
```

Out[11]:

	pclass	sex	age	survived
count	1309.000000	1309.000000	1309.000000	1309.000000
mean	2.294882	0.644003	29.881135	0.381971
std	0.837836	0.478997	12.883199	0.486055
min	1.000000	0.000000	0.166700	0.000000
25%	2.000000	0.000000	22.000000	0.000000
50%	3.000000	1.000000	29.881135	0.000000
75%	3.000000	1.000000	35.000000	1.000000
max	3.000000	1.000000	80.000000	1.000000

In [12]:
```
data.dtypes
```

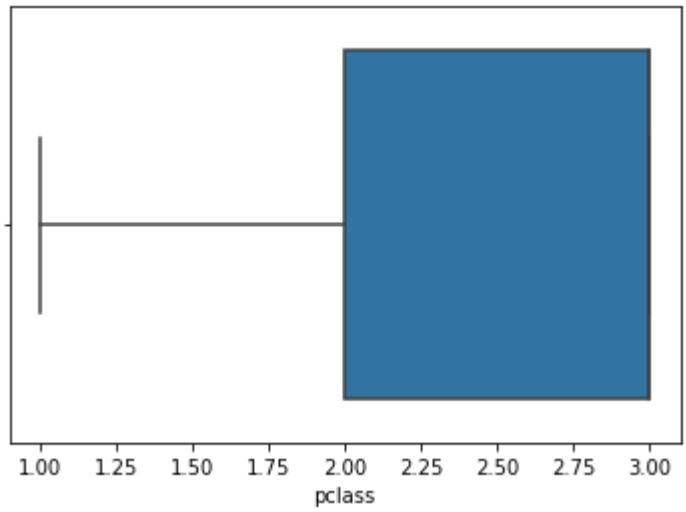
Out[12]:
```
pclass        int64
sex           int64
age         float64
survived      int64
dtype: object
```

In [13]:
```
data.age = data.age.apply(lambda x:round(x,2))
```

In [14]:
```
sns.boxplot(data['pclass'])
```

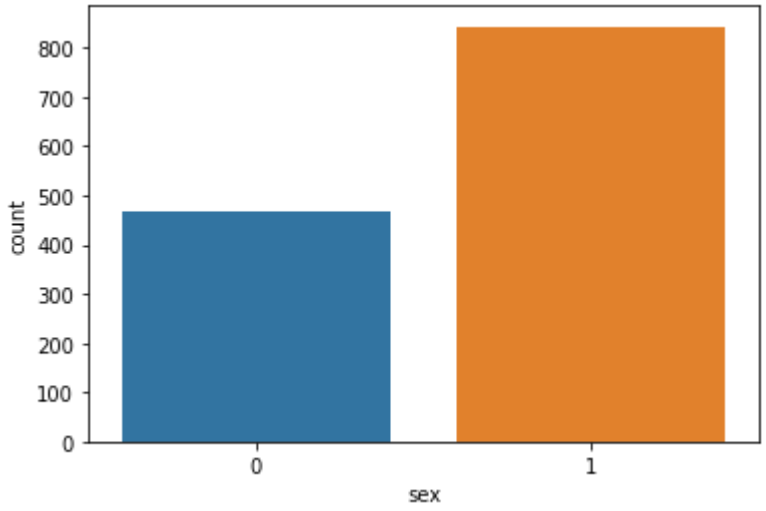
```
C:\Users\extrusion115\AppData\Local\Programs\Python\Python310\lib\site-packages
\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a key
word arg: x. From version 0.12, the only valid positional argument will be `dat
a`, and passing other arguments without an explicit keyword will result in an e
rror or misinterpretation.
  warnings.warn(
```

Out[14]:
```
<AxesSubplot:xlabel='pclass'>
```



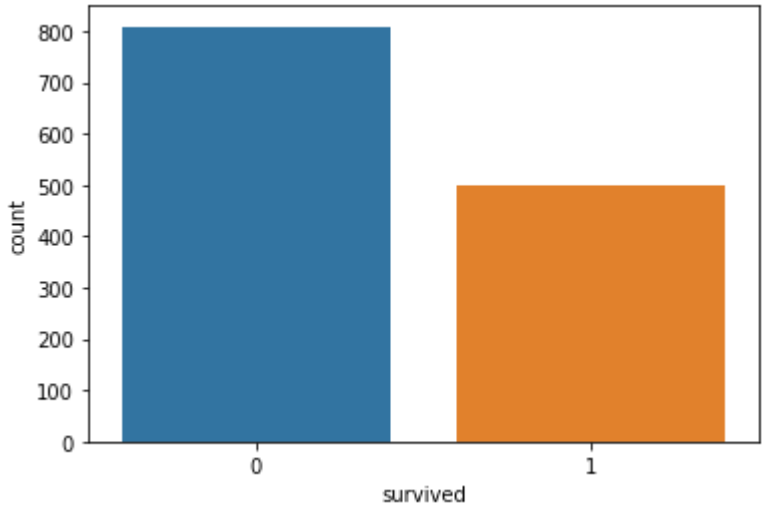
In [15]:
```
sns.countplot(x='sex',data=data)
```

Out[15]:
```
<AxesSubplot:xlabel='sex', ylabel='count'>
```



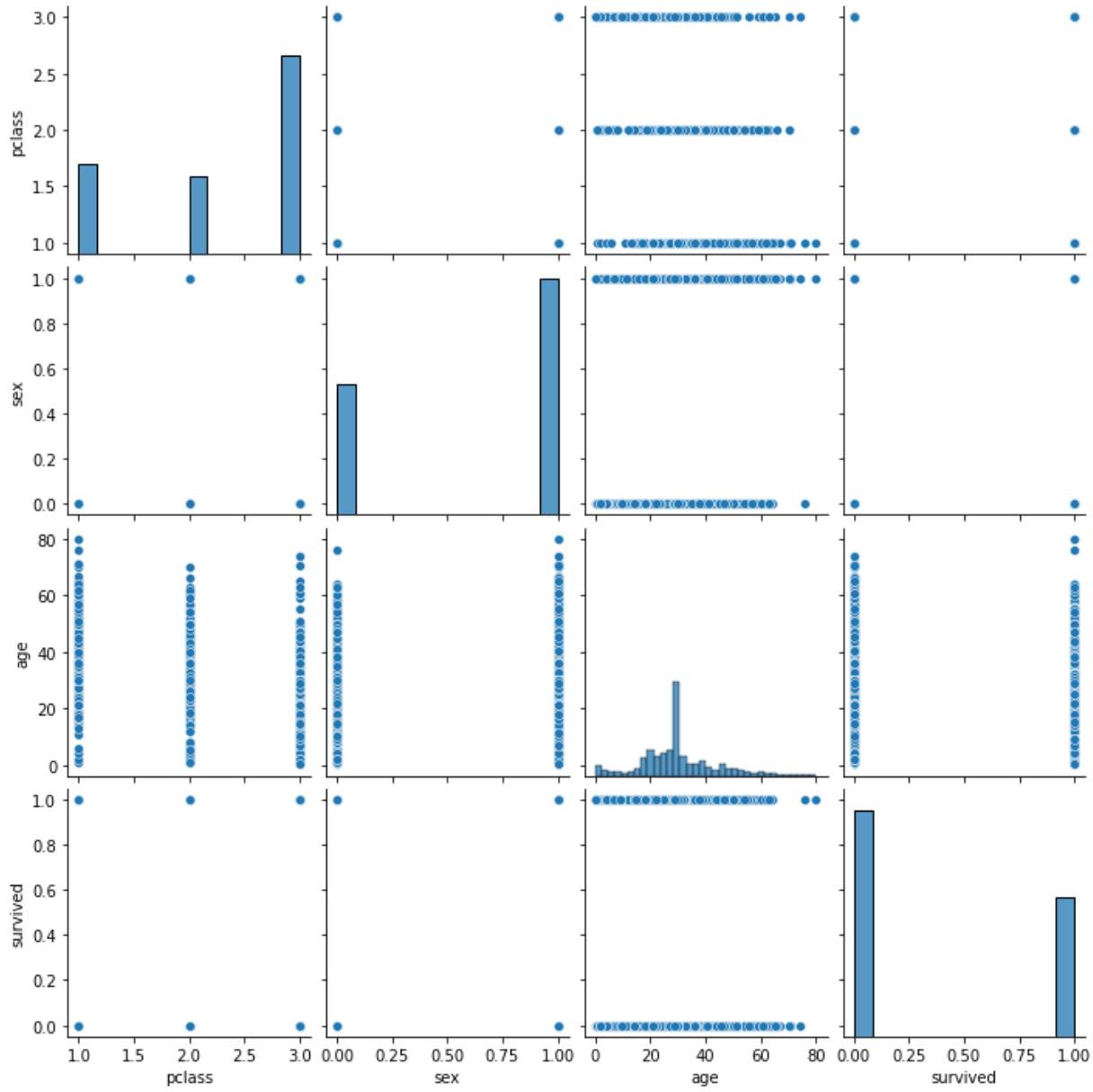
In [16]:
```
sns.countplot(x='survived',data=data)
```

Out[16]:
```
<AxesSubplot:xlabel='survived', ylabel='count'>
```



In [17]:
```
sns.pairplot(data)
```

Out[17]:
```
<seaborn.axisgrid.PairGrid at 0x23059f93580>
```

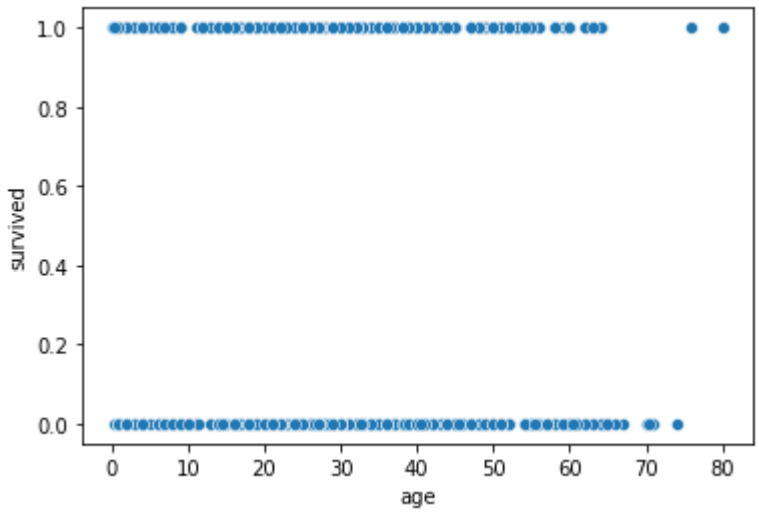


In [18]:
```
data.corrwith(data['survived'])
```

Out[18]:
```
pclass     -0.312469
sex        -0.528693
age        -0.050195
survived    1.000000
dtype: float64
```

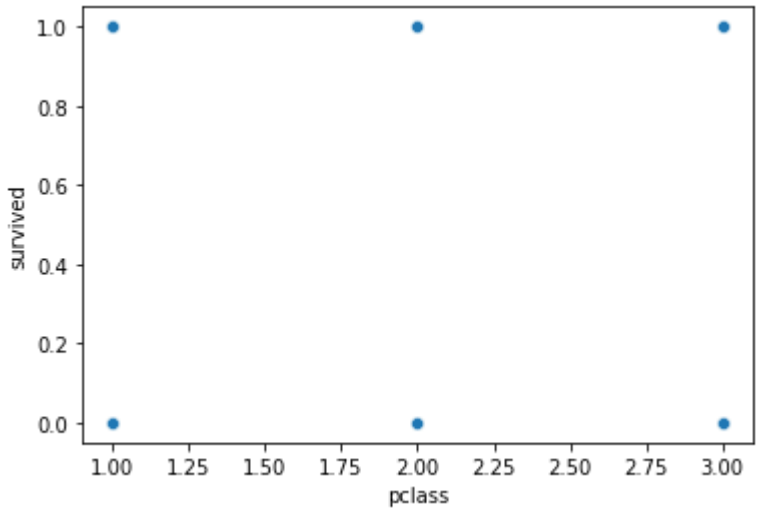
In [19]:
```
sns.scatterplot(data=data,x='age',y='survived')
```

Out[19]:
```
<AxesSubplot:xlabel='age', ylabel='survived'>
```



In [20]:
```
sns.scatterplot(data=data,x='pclass',y='survived')
```

Out[20]:
```
<AxesSubplot:xlabel='pclass', ylabel='survived'>
```



In [21]:
```
X = data.loc[:,['pclass','sex','age']]
y = data.loc[:,['survived']]
```

In [22]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=4
```

In [23]:
```
model = LinearRegression()
model.fit(X_train,y_train)
```

Out[23]:
```
▾ LinearRegression

LinearRegression()
```

In [24]:
```
print("The estimated intercept coeeficient is %.2f" %model.intercept_)
print("The number of coefficient used", len(model.coef_))
```

```
The estimated intercept coeeficient is 1.15
The number of coefficient used 1
```

In [25]:
```
coeff_df = pd.DataFrame(X.columns)
coeff_df.columns = ['Features']
coeff_df['Coefficient Estimate'] = pd.Series(model.coef_.flatten())
coeff_df
```

Out[25]:

	Features	Coefficient Estimate
0	pclass	-0.166264
1	sex	-0.476205
2	age	-0.003437

In [26]:
```
y_pred = model.predict(X_test)
```

In [27]:
```
from sklearn import metrics
```

In [28]:
```
df1 = pd.DataFrame({'Actual': y_test.to_numpy().flatten(), 'Predicted': y_pred.fl
df1
```

Out[28]:

	Actual	Predicted
0	0	0.057750
1	1	0.109302
2	0	0.075346
3	0	0.075346
4	0	0.075346
...	...	...
388	0	0.241198
389	0	0.114457
390	0	0.203393
391	1	0.551551
392	1	0.772392

393 rows × 2 columns

In [30]:
```
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pr
print('Root Squared Error:', metrics.r2_score(y_test, y_pred))

print("Coefficients(Slope): ", model.coef_)
```

```
Mean Absolute Error: 0.30623270000221997
Mean Squared Error: 0.15209260642461445
Root Mean Squared Error: 0.3899905209420024
Root Squared Error: 0.37947614725075873
Coefficients(Slope):  [[-0.16626413 -0.47620524 -0.00343678]]
```

In [ ]:

In [ ]: