In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score,explained_variance_score,mean_absolute_er
from math import sqrt
import seaborn as sns
%matplotlib inline
```

In [2]:
```
student=pd.read_csv("C:\\Users\\HP\\Desktop\\LINEAR PROGRAMMING CENTRALS\\Stud
student
```

Out[2]:

	gender	race/ ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/ reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75
...	...	...	...	...	...	...	...	...
94	female	group B	some college	standard	none	79	86	92
95	male	group C	associate's degree	free/ reduced	completed	78	81	82
96	male	group B	some high school	standard	completed	65	66	62
97	female	group E	some college	standard	completed	63	72	70
98	female	group D	some college	free/ reduced	none	58	67	62

99 rows × 8 columns

In [3]:
```
print(student.shape)
student.head()
```

```
(99, 8)
```

Out[3]:

	gender	race/ ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/ reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

In [4]:
```
student.isnull()
```

Out[4]:

	gender	race/ ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
94	False	False	False	False	False	False	False	False
95	False	False	False	False	False	False	False	False
96	False	False	False	False	False	False	False	False
97	False	False	False	False	False	False	False	False
98	False	False	False	False	False	False	False	False

99 rows × 8 columns

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

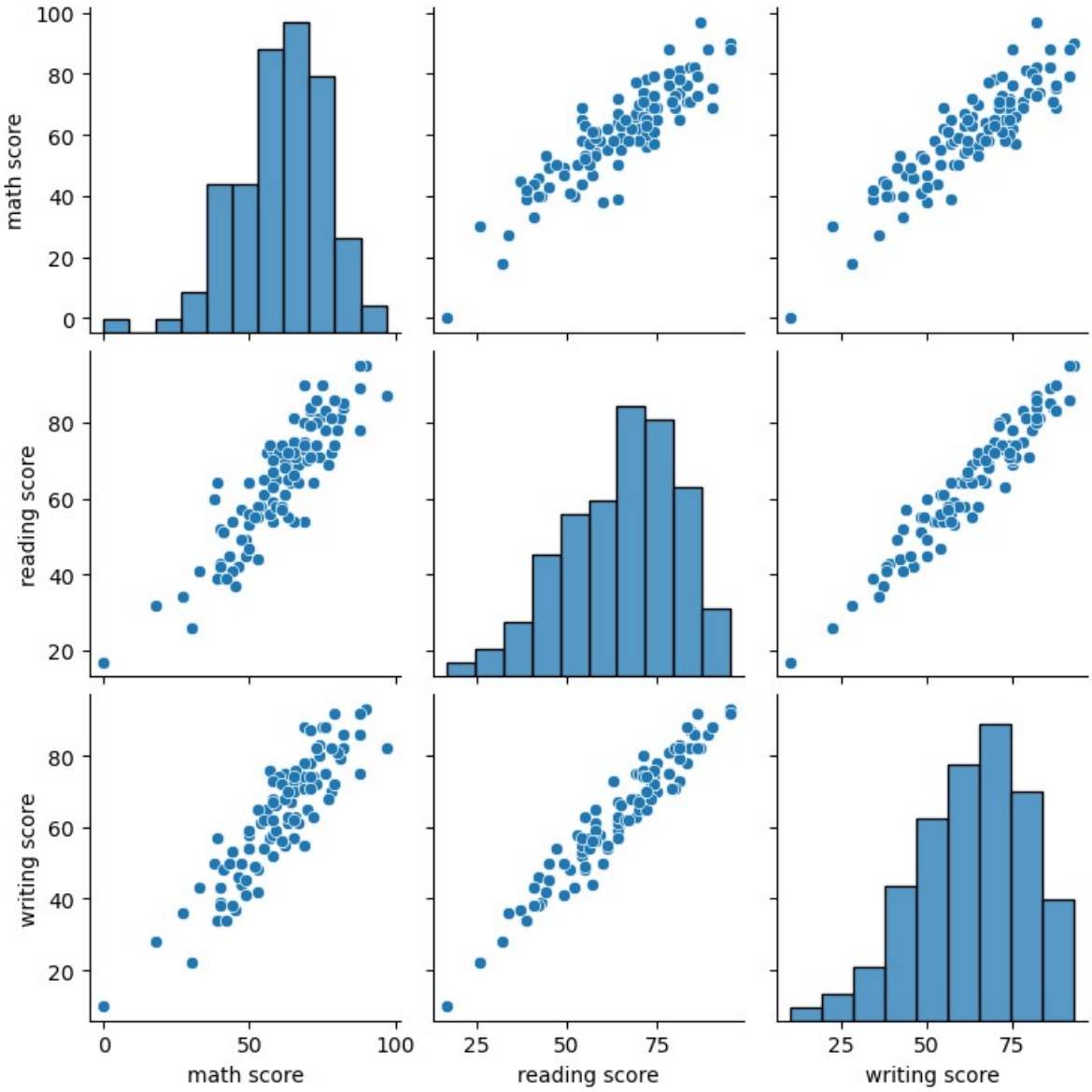
Out[5]:
```
gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
dtype: int64
```

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

Out[6]:
```
(99, 8)
```

In [7]:
```
#visualizing data using seaborn pairplot
EDWARD=sns.pairplot(student)
```

```
C:\Users\HP\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [8]:
```
student.columns
```

Out[8]:
```
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'test preparation course', 'math score', 'reading score',
       'writing score'],
      dtype='object')
```

In [9]:
```
X=np.array(student[['math score','reading score']])
Y=np.array(student['writing score'])
X
```

Out[9]:
```
array([[72, 72],
       [69, 90],
       [90, 95],
       [47, 57],
       [76, 78],
       [71, 83],
       [88, 95],
       [40, 43],
       [64, 64],
       [38, 60],
       [58, 54],
       [40, 52],
       [65, 81],
       [78, 72],
       [50, 53],
       [69, 75],
       [88, 89],
       [18, 32],
       [46, 42],
       [54, 58],
       [66, 69],
       [65, 75],
       [44, 54],
       [69, 73],
       [74, 71],
       [73, 74],
       [69, 54],
       [67, 69],
       [70, 70],
       [62, 70],
       [69, 74],
       [63, 65],
       [56, 72],
       [40, 42],
       [97, 87],
       [81, 81],
       [74, 81],
       [50, 64],
       [75, 90],
       [57, 56],
       [55, 61],
       [58, 73],
       [53, 58],
       [59, 65],
       [50, 56],
       [65, 54],
       [55, 65],
       [66, 71],
       [57, 74],
       [82, 84],
       [53, 55],
       [77, 69],
```

```
[53, 44],
[88, 78],
[71, 84],
[33, 41],
[82, 85],
[52, 55],
[58, 59],
[ 0, 17],
[79, 74],
[39, 39],
[62, 61],
[69, 80],
[59, 58],
[67, 64],
[45, 37],
[60, 72],
[61, 58],
[39, 64],
[58, 63],
[63, 55],
[41, 51],
[61, 57],
[49, 49],
[44, 41],
[30, 26],
[80, 78],
[61, 74],
[62, 68],
[47, 49],
[49, 45],
[50, 47],
[72, 64],
[42, 39],
[73, 80],
[76, 83],
[71, 71],
[58, 70],
[73, 86],
[65, 72],
[27, 34],
[71, 79],
[43, 45],
[79, 86],
[78, 81],
[65, 66],
[63, 72],
[58, 67]], dtype=int64)
```

In [36]:
```
Y
```

Out[36]:
```
array([74, 88, 93, 44, 75, 78, 92, 39, 67, 50, 52, 43, 73, 70, 58, 78, 86,
       28, 46, 61, 63, 70, 53, 73, 80, 72, 55, 75, 65, 75, 74, 61, 65, 38,
       82, 79, 83, 59, 88, 57, 54, 68, 65, 66, 54, 57, 62, 76, 76, 82, 48,
       68, 42, 75, 87, 43, 86, 49, 58, 10, 72, 34, 55, 71, 59, 61, 37, 74,
       56, 57, 73, 63, 48, 56, 41, 38, 22, 81, 72, 68, 50, 45, 54, 63, 34,
       82, 88, 74, 67, 82, 74, 36, 71, 50, 92, 82, 62, 70, 62],
      dtype=int64)
```

In [37]:
```
print(X.shape)
```

```
(99, 2)
```

In [38]:
```
print(Y.shape)
```

```
(99,)
```

In [40]:
```
dt=DecisionTreeClassifier(random_state=15,criterion='entropy', max_depth=10)
dt.fit(X,Y)
```

Out[40]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=10, random_state=15)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [14]:
```
X_train, X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state
print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(79, 2)
(20, 2)
(79,)
(20,)
```

In [15]:
```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
```

In [16]:
```
model.fit(X_train,Y_train)
```

Out[16]:
```
▾ LinearRegression
LinearRegression()
```

In [17]:
```
Y_pred=model.predict(X_test)
Y_pred
```

Out[17]:
```
array([72.23452555, 86.15351459, 61.26809246, 26.33709266, 61.19684632,
       65.28399609, 71.56326874, 87.40103335, 94.55948261, 39.33522627,
       75.30582505, 57.06801775, 58.90972858, 41.34317809, 70.85033312,
       77.91378754, 77.68211903, 42.15110856, 54.78090001, 48.86996951])
```

In [21]:
```
from sklearn.metrics import accuracy_score,mean_squared_error,r2_score
mean_absolute_error(Y_test,Y_pred)
```

Out[21]:
```
3.2251680709052346
```

In [22]:
```
mean_absolute_error(Y_test,Y_pred)
```

Out[22]:
```
3.2251680709052346
```

In [23]:
```
r2_score(Y_test,Y_pred)
```

Out[23]:
```
0.9600528276799257
```

In [24]:
```
student=model.score(X,Y)
student
```

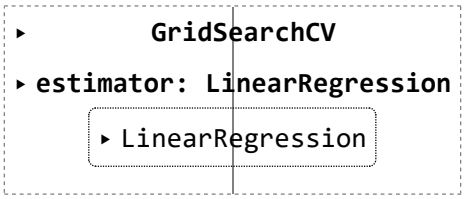
Out[24]:
```
0.9339528080790578
```

In [25]:
```
from sklearn.model_selection import GridSearchCV
model=LinearRegression()
param_grid={
    'fit_intercept':[True,False],
    'copy_X':[True,False],
    'n_jobs':[True,False],
}
param_grid
```

Out[25]:
```
{'fit_intercept': [True, False],
 'copy_X': [True, False],
 'n_jobs': [True, False]}
```

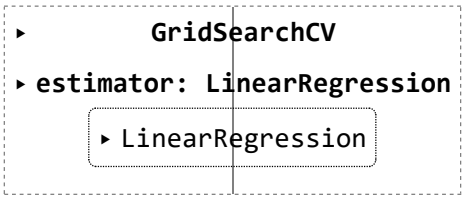
In [26]:
```
grid_search=GridSearchCV(model,param_grid,cv=5)
grid_search
```

Out[26]:



In [35]:
```
grid_search.fit(X_train,Y_train)
```

Out[35]:



In [37]:
```
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
```

In [39]:
```
mse=mean_squared_error(Y_test,Y_pred)
mse
```

Out[39]:
```
14.128815509956274
```

In [ ]:
```

```