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

In [2]:
```
df=pd.read_csv("https://raw.githubusercontent.com/dsrscientist/DSData/master/Advertising.csv")
```

In [3]:
```
df
```

Out[3]:

	Unnamed: 0	TV	radio	newspaper	sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9
...	...	...	...	...	...
195	196	38.2	3.7	13.8	7.6
196	197	94.2	4.9	8.1	9.7
197	198	177.0	9.3	6.4	12.8
198	199	283.6	42.0	66.2	25.5
199	200	232.1	8.6	8.7	13.4

200 rows × 5 columns

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  200 non-null    int64  
 1   TV          200 non-null    float64
 2   radio       200 non-null    float64
 3   newspaper   200 non-null    float64
 4   sales       200 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 7.9 KB
```

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

Out[5]:
```
Unnamed: 0    0
TV            0
radio         0
newspaper     0
sales         0
dtype: int64
```

In [6]:
```
df.describe()
```

Out[6]:

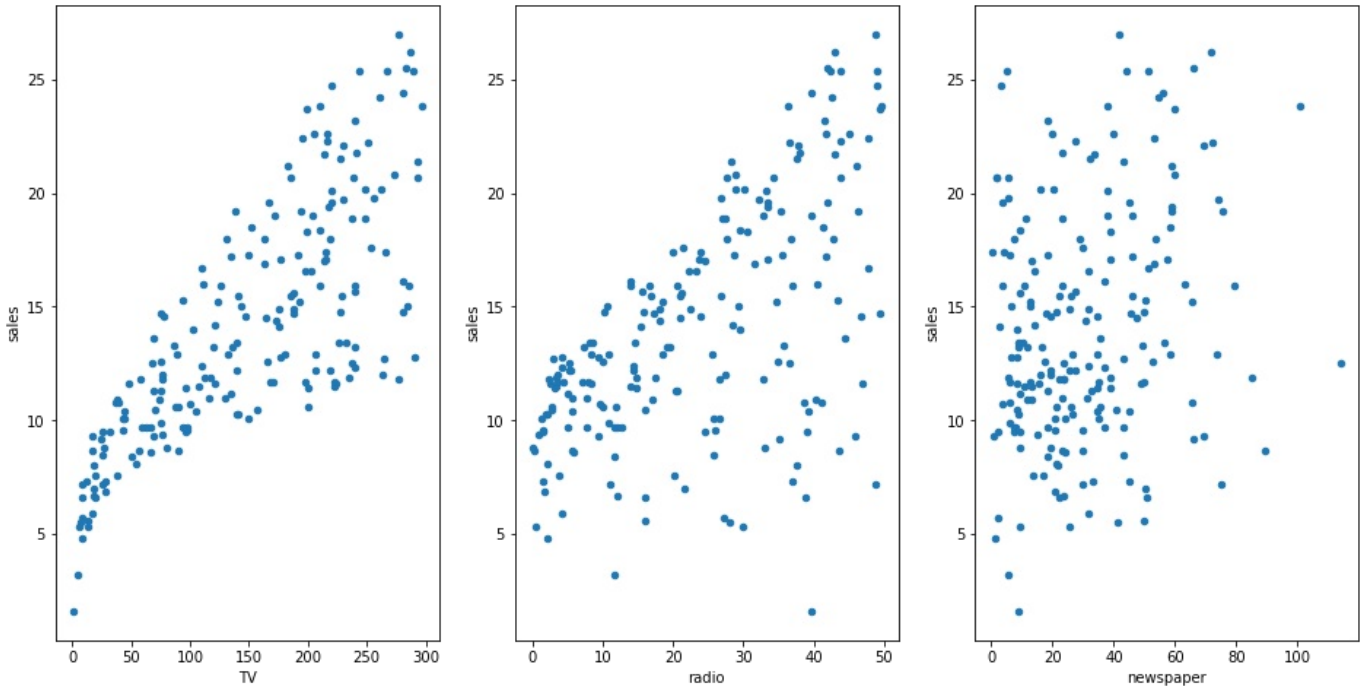
	Unnamed: 0	TV	radio	newspaper	sales
count	200.000000	200.000000	200.000000	200.000000	200.000000
mean	100.500000	147.042500	23.264000	30.554000	14.022500
std	57.879185	85.854236	14.846809	21.778621	5.217457
min	1.000000	0.700000	0.000000	0.300000	1.600000
25%	50.750000	74.375000	9.975000	12.750000	10.375000
50%	100.500000	149.750000	22.900000	25.750000	12.900000
75%	150.250000	218.825000	36.525000	45.100000	17.400000
max	200.000000	296.400000	49.600000	114.000000	27.000000

In [7]:
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [8]:
```
fig,axs=plt.subplots(1,3)

df.plot(kind="scatter",x="TV",y="sales",ax=axs[0],figsize=(16,8))
df.plot(kind="scatter",x="radio",y="sales",ax=axs[1])
df.plot(kind="scatter",x="newspaper",y="sales",ax=axs[2])
```

Out[8]:
```
<AxesSubplot:xlabel='newspaper', ylabel='sales'>
```



In [9]:
```
df=df.drop(["Unnamed: 0"],axis=1)
```

In [10]:
```
df.head()
```

Out[10]:

	TV	radio	newspaper	sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9

In [11]:
```
df.shape
```

Out[11]:
```
(200, 4)
```

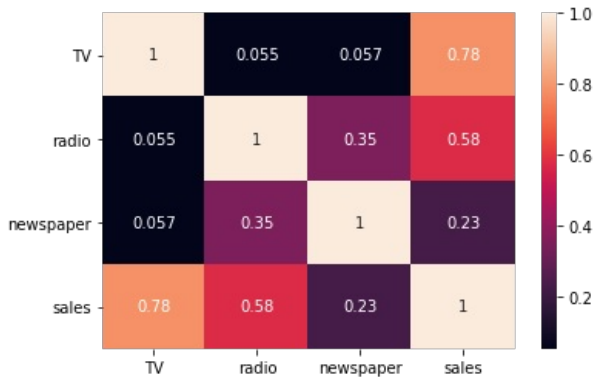
In [12]:
```
df.corr()
```

Out[12]:

	TV	radio	newspaper	sales
TV	1.000000	0.054809	0.056648	0.782224
radio	0.054809	1.000000	0.354104	0.576223
newspaper	0.056648	0.354104	1.000000	0.228299
sales	0.782224	0.576223	0.228299	1.000000

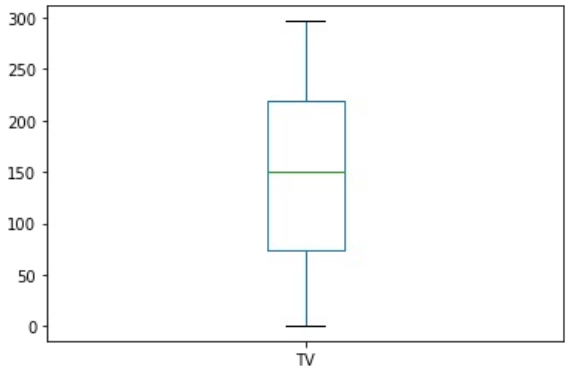
In [13]:
```
sns.heatmap(df.corr(),annot=True)
```

Out[13]: <AxesSubplot:>



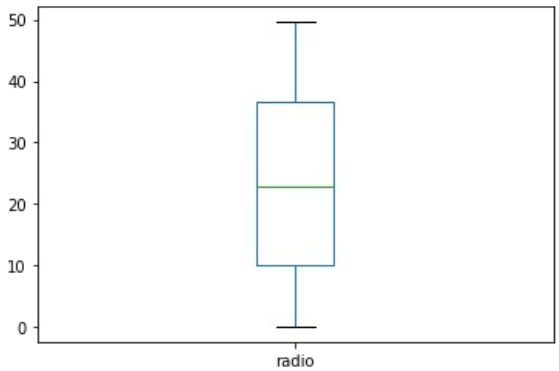
In [14]:
```
df["TV"].plot.box()
```

Out[14]: <AxesSubplot:>



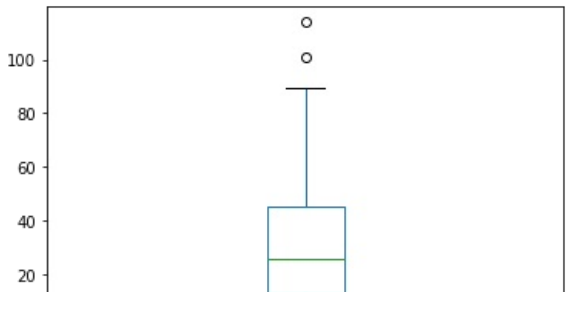
In [15]:
```
df["radio"].plot.box()
```

Out[15]: <AxesSubplot:>



In [16]:
```
df["newspaper"].plot.box()
```

Out[16]: <AxesSubplot:>



In [17]:
```
df.skew()
```

Out[17]:
```
TV          -0.069853
radio        0.094175
newspaper    0.894720
sales        0.407571
dtype: float64
```

In [18]:
```
x=df.drop(columns=["sales"])
y=df["sales"]
```

In [19]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x=sc.fit_transform(x)
x
```

Out[19]:
```
array([[ 9.69852266e-01,  9.81522472e-01,  1.77894547e+00],
       [-1.19737623e+00,  1.08280781e+00,  6.69578760e-01],
       [-1.51615499e+00,  1.52846331e+00,  1.78354865e+00],
       [ 5.20496822e-02,  1.21785493e+00,  1.28640506e+00],
       [ 3.94182198e-01, -8.41613655e-01,  1.28180188e+00],
       [-1.61540845e+00,  1.73103399e+00,  2.04592999e+00],
       [-1.04557682e+00,  6.43904671e-01, -3.24708413e-01],
       [-3.13436589e-01, -2.47406325e-01, -8.72486994e-01],
       [-1.61657614e+00, -1.42906863e+00, -1.36042422e+00],
       [ 6.16042873e-01, -1.39530685e+00, -4.30581584e-01],
       [-9.45155670e-01, -1.17923146e+00, -2.92486143e-01],
       [ 7.90028350e-01,  4.96973404e-02, -1.22232878e+00],
       [-1.43908760e+00,  7.99208859e-01,  1.62704048e+00],
       [-5.78501712e-01, -1.05768905e+00, -1.07502697e+00],
       [ 6.66253447e-01,  6.50657027e-01,  7.11007392e-01],
       [ 5.64664612e-01,  1.65000572e+00,  1.02862691e+00],
       [-9.25304978e-01,  9.00494200e-01,  3.84117072e+00],
       [ 1.56887609e+00,  1.10306488e+00,  1.16211917e+00],
       [-9.08957349e-01, -1.86635121e-01, -5.64073843e-01],
       [ 3.00679600e-03,  4.29449843e-02, -5.27248393e-01],
       [ 8.33232798e-01,  2.99534513e-01,  1.05164281e+00],
       [ 1.05509347e+00, -1.22649795e+00, -3.24708413e-01],
       [-1.56286250e+00, -4.97243498e-01,  8.76721921e-01],
       [ 9.48833887e-01, -4.29719938e-01, -2.00422516e-01],
       [-9.89527805e-01, -7.20071247e-01, -5.64073843e-01],
       [ 1.35285385e+00, -1.33453565e+00, -5.08835667e-01],
       [-4.83714657e-02,  4.07572210e-01, -8.26455181e-01],
       [ 1.08662104e+00, -4.43224650e-01, -3.52327501e-01],
       [ 1.18820988e+00,  2.59020377e-01, -3.52327501e-01],
       [-8.92609721e-01, -4.90491142e-01,  4.71641962e-01],
       [ 1.70316018e+00,  3.40048650e-01,  5.82118314e-01],
       [-3.98677796e-01, -3.95958157e-01,  3.70371972e-01],
       [-5.82004775e-01, -1.46958277e+00, -2.55016247e-02],
       [ 1.38438142e+00, -2.20396901e-01, -1.39264649e+00],
       [-5.99520091e-01, -1.47633512e+00, -1.06582061e+00],
       [ 1.67747105e+00, -1.29402151e+00, -1.01518562e+00],
       [ 1.39956136e+00,  1.38666383e+00, -1.17629696e+00],
       [-8.44734522e-01,  1.76479577e+00,  6.97197848e-01],
       [-1.21372386e+00,  2.32010953e-01,  2.09260624e-01],
       [ 9.45330823e-01,  9.74770116e-01,  6.65620024e-02],
       [ 6.47570443e-01, -6.50927121e-02,  4.81492770e-02],
       [ 3.49810063e-01,  6.84418807e-01,  3.74975153e-01],
       [ 1.71133400e+00,  2.99534513e-01, -1.32359877e+00],
       [ 6.98948705e-01, -1.00367020e+00, -1.91216154e-01],
       [-1.42390765e+00,  1.64487393e-01,  5.86721496e-01],
       [ 3.27623995e-01, -5.15880000e-02,  4.35460956e-02],
       [-6.69581357e-01, -9.02384859e-01,  2.36879713e-01],
       [ 1.08428567e+00,  1.23135965e+00, -5.54867481e-01],
       [ 9.35989321e-01, -5.03995854e-01,  8.90531465e-01],
       [-9.35814168e-01, -7.80842451e-01,  2.87514708e-01],
       [ 6.16042873e-01, -1.36154507e+00,  1.86244718e-01],
       [-5.44638766e-01, -9.22641928e-01, -1.24074150e+00],
       [ 8.09879042e-01,  1.24486436e+00,  4.16403786e-01],
       [ 4.15200577e-01,  1.54872038e+00,  1.29561142e+00],
       [ 1.35051848e+00,  3.73810430e-01, -6.74550196e-01],
```

```
[ 6.05533683e-01,  1.76479577e+00,  1.35545278e+00],
[-1.63175608e+00,  3.26543937e-01,  4.99261050e-01],
[-1.26606546e-01, -2.74415749e-01, -6.42327927e-01],
[ 7.44488528e-01,  1.77830048e+00,  3.28943340e-01],
[ 7.43320840e-01,  4.21076922e-01, -9.78360166e-01],
[-1.09228433e+00, -1.43582099e+00, -4.21375221e-01],
[ 1.33417085e+00,  1.31238792e+00,  1.11148417e+00],
[ 1.07727954e+00, -5.24252922e-01, -1.49787521e-01],
[-5.17781948e-01,  4.27829278e-01, -1.01978880e+00],
[-1.86158622e-01,  1.31914027e+00, -7.61366196e-02],
[-9.11292725e-01, -9.42898996e-01, -1.36502740e+00],
[-1.34917564e+00,  9.02114765e-02, -1.30518604e+00],
[-9.04082253e-02, -5.91776482e-01, -9.36931533e-01],
[ 1.05509347e+00,  2.86029801e-01, -9.00106083e-01],
[ 8.14549794e-01,  1.39341619e+00, -1.54390703e-01],
[ 6.07869059e-01,  4.95352838e-01,  3.74975153e-01],
[-4.34876116e-01, -6.05281194e-01,  5.27524584e-02],
[-1.40405696e+00,  6.57409383e-01, -5.18042030e-01],
[-2.06009314e-01, -1.18598381e+00,  3.43397329e-02],
[ 7.74848409e-01,  9.02114765e-02, -8.03439274e-01],
[-1.51965805e+00,  1.37991148e+00,  2.70878810e+00],
[-1.39588315e+00, -1.46283041e+00, -4.53597491e-01],
[-3.09933525e-01,  3.53553362e-01, -7.52804279e-01],
[-1.65394214e+00,  4.48086346e-01, -9.73756984e-01],
[-3.62479475e-01, -1.05093669e+00, -3.43121138e-01],
[-8.24883830e-01,  2.32010953e-01, -3.79946589e-01],
[ 1.08311798e+00, -1.29402151e+00,  2.92117889e-01],
[-8.37728396e-01, -2.00139833e-01,  8.95779092e-02],
[-9.18298852e-01,  1.43393033e+00,  2.32276531e-01],
[ 7.76016097e-01,  1.33264499e+00,  1.49419267e-01],
[ 5.38975481e-01, -3.28434597e-01,  1.61783412e+00],
[-8.26051518e-01,  2.86029801e-01, -6.69947015e-01],
[-4.24366926e-01,  1.17058844e+00,  1.50275459e+00],
[-6.85928986e-01,  1.50982681e-01,  1.97227908e+00],
[-4.34876116e-01,  1.65675807e+00,  9.59579186e-01],
[-1.48792614e-01, -1.24000266e+00, -9.78360166e-01],
[-1.38303858e+00, -1.46958277e+00,  1.12593816e-01],
[ 8.25058983e-01,  6.91171163e-01,  1.30942097e+00],
[ 1.21273132e+00,  8.93741844e-01,  1.92164409e+00],
[-4.62900623e-01, -6.25538262e-01, -9.04709264e-01],
[ 1.89836839e-01,  5.62876398e-01,  1.02862691e+00],
[ 5.90353742e-01, -1.33453565e+00, -1.13486833e+00],
[ 4.42057396e-01, -1.52873340e-01, -3.93756133e-01],
[ 1.66579418e+00,  1.28537849e+00,  9.50372823e-01],
[-1.38283424e-01,  1.24486436e+00,  7.06404211e-01],
[ 8.79940308e-01, -1.28051680e+00,  8.85928284e-01],
[ 1.74402926e+00,  8.80237132e-01,  3.23815396e+00],
[ 1.55486384e+00, -8.88880147e-01, -4.21375221e-01],
[ 4.77088029e-01, -4.09462869e-01, -5.82486569e-01],
[ 1.06443498e+00,  7.45190011e-01, -1.16248742e+00],
[-1.06755854e-01,  1.56222509e+00,  1.30942097e+00],
[-1.42507534e+00, -8.28108943e-01, -3.93111688e-02],
[-6.61407543e-01, -1.55061104e+00, -3.38517957e-01],
[-1.56403019e+00, -1.54385868e+00, -2.28041604e-01],
[ 1.26527727e+00,  2.45515665e-01, -1.15328106e+00],
[ 9.19641692e-01, -1.01717491e+00,  1.19434143e+00],
[ 1.10530405e+00,  9.95027184e-01, -3.38517957e-01],
[ 3.34630122e-01, -5.31005278e-01, -1.29597968e+00],
[ 7.30476274e-01, -1.79882765e-01, -9.13915627e-01],
[-8.03865450e-01,  1.58923451e+00,  1.81641536e-01],
[-8.40063771e-01,  7.92456503e-01,  1.01942054e+00],
[-9.15759131e-02, -6.05281194e-01, -2.28041604e-01],
[-8.24883830e-01, -1.51684926e+00, -7.25185191e-01],
[-2.49213762e-01,  9.20751268e-01,  2.23926360e+00],
[-1.49046586e+00, -4.90491142e-01, -3.79946589e-01],
[-6.70544700e-02,  2.38763309e-01,  7.20213755e-01],
[-1.49747198e+00, -1.05606848e-01,  9.13547372e-01],
[ 8.98623313e-01, -1.40881156e+00, -6.88359740e-01],
[-2.79573643e-01,  7.65447079e-01, -8.35661544e-01],
[ 9.62846140e-01,  6.10142891e-01,  2.00910454e+00],
[-6.98773552e-01, -7.74090095e-01, -2.14232060e-01],
[-1.62591764e+00,  1.05579839e+00,  9.22753735e-01],
[-7.80511695e-01, -1.57086811e+00, -9.82963347e-01],
[ 8.55418865e-01,  1.73778635e+00, -1.25915423e+00],
[-1.02105537e+00, -7.60585383e-01,  5.77515133e-01],
[-1.70882347e+00,  1.10306488e+00, -1.00597925e+00],
[ 1.37971067e+00, -1.37504978e+00,  5.72911952e-01],
[-1.61891151e+00,  2.65772733e-01, -1.30978922e+00],
[ 8.49580427e-01,  6.91171163e-01,  6.69578760e-01],
[-1.28612050e+00,  1.03554132e+00,  1.61323094e+00],
[-1.15300409e+00,  1.60273923e+00, -1.01518562e+00],
[-1.41806922e+00,  1.06255074e+00, -9.78360166e-01],
[ 1.47896413e+00,  3.80562786e-01,  1.34164324e+00],
```

```
       [-1.21489154e+00,  1.77992105e-01, -4.62803854e-01],
       [ 4.42057396e-01,  1.39341619e+00, -1.32820195e+00],
       [-8.59914463e-01, -4.22967582e-01, -8.12645637e-01],
       [ 5.44813920e-01,  8.19465927e-01,  2.07354907e+00],
       [ 8.57754241e-01,  6.70914095e-01,  3.38149702e-01],
       [-4.95595880e-01, -1.18598381e+00,  1.77038355e-01],
       [-5.93681653e-01, -5.71519414e-01,  3.84181516e-01],
       [-7.87313476e-02, -1.44257334e+00, -9.92169710e-01],
       [ 1.08662104e+00, -1.07794612e+00, -1.00597925e+00],
       [ 1.12281936e+00,  1.73778635e+00,  6.32753309e-01],
       [-1.27327593e+00,  1.15033137e+00, -8.58677450e-01],
       [-1.19504085e+00,  1.71239749e-01, -4.58200672e-01],
       [ 1.56070228e+00, -6.32290618e-01,  2.96721070e-01],
       [-3.04095087e-01, -1.00367020e+00,  8.35293289e-01],
       [ 5.90353742e-01,  2.43084817e-03, -7.52804279e-01],
       [ 2.83251860e-01,  1.10981724e+00,  3.28943340e-01],
       [ 4.75920341e-01, -1.46120984e-01, -9.69153803e-01],
       [-1.66912209e+00, -7.87594807e-01, -1.14407469e+00],
       [-6.20538471e-01,  1.36640677e+00,  9.18150553e-01],
       [ 3.21989902e-02, -1.48308748e+00, -2.87882962e-01],
       [-1.58037782e+00,  9.20751268e-01,  6.74181942e-01],
       [-1.79152496e-01, -3.28434597e-01,  1.86244718e-01],
       [ 2.97264113e-01, -3.48691665e-01,  6.72064478e-03],
       [-7.16288868e-01,  8.46475352e-01,  8.62912377e-01],
       [ 4.82926468e-01, -3.48691665e-01, -2.28041604e-01],
       [ 1.92172214e-01,  9.13998912e-01, -1.06582061e+00],
       [-3.48467222e-01, -5.78271770e-01, -1.15788424e+00],
       [ 1.02123053e+00, -1.34128800e+00,  2.49704176e+00],
       [-1.50798117e+00,  9.68017760e-01, -4.12168859e-01],
       [ 6.97781017e-01, -1.21974559e+00, -5.13438849e-01],
       [ 7.98202165e-01,  2.26879163e-02,  1.24497643e+00],
       [ 1.60273904e+00, -8.55118367e-01, -1.11185242e+00],
       [-1.13315340e+00, -7.87594807e-01, -5.59470662e-01],
       [ 2.03849092e-01, -1.59625696e-01,  7.75451931e-01],
       [-1.48813048e+00, -2.13644545e-01, -6.23915201e-01],
       [ 2.49388915e-01, -1.09145083e+00, -8.17248818e-01],
       [ 8.79940308e-01, -1.34128800e+00, -8.03439274e-01],
       [ 1.51633014e+00,  1.73103399e+00,  5.17673775e-01],
       [ 1.18353913e+00,  4.68343414e-01, -4.72010216e-01],
       [ 2.70407294e-01, -1.04418434e+00,  2.13863806e-01],
       [ 1.51399477e+00, -1.41556392e+00, -3.15502050e-01],
       [ 2.16693657e-01, -8.95632503e-01, -5.96296113e-01],
       [ 1.11601758e-01, -1.39530685e+00, -1.02439198e+00],
       [ 8.34400486e-01, -1.20624088e+00, -1.45184340e-01],
       [-1.06075676e+00, -1.18598381e+00, -3.93111688e-02],
       [ 1.64127273e+00,  1.33264499e+00,  1.89862818e+00],
       [ 1.24659427e+00, -1.32616272e-01, -2.55016247e-02],
       [ 6.76762637e-01,  1.47444446e+00, -5.04232486e-01],
       [-8.80728498e-02, -1.42906863e+00, -1.82009791e-01],
       [ 5.14454038e-01,  3.67058074e-01, -5.68677025e-01],
       [ 1.62258973e+00, -6.32290618e-01, -1.23613832e+00],
       [-1.49863967e+00, -7.53833027e-01, -3.29311594e-01],
       [-1.25576062e+00,  1.20435022e+00, -1.13947151e+00],
       [-8.35393020e-01, -8.41613655e-01, -1.13026515e+00],
       [-1.51615499e+00, -1.29402151e+00,  4.81492770e-02],
       [ 2.30705910e-01,  1.26512143e+00, -1.24074150e+00],
       [ 3.10313024e-02,  8.32970639e-01, -1.13026515e+00],
       [-1.27094056e+00, -1.32103093e+00, -7.71217005e-01],
       [-6.17035408e-01, -1.24000266e+00, -1.03359834e+00],
       [ 3.49810063e-01, -9.42898996e-01, -1.11185242e+00],
       [ 1.59456522e+00,  1.26512143e+00,  1.64085003e+00],
       [ 9.93206022e-01, -9.90165488e-01, -1.00597925e+00]])
```

In [20]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [21]:

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

lr=LinearRegression()
model=lr.fit(x_train,y_train)
y_pred=model.predict(x_test)
rmse=mean_squared_error(y_test, y_pred)**0.5
print("rmse=",rmse)
r2=r2_score(y_test, y_pred)
print("r2=",r2)
score=cross_val_score(model,x,y,cv=5,scoring="r2")
print("cv_score=",score)
```

```
rmse= 1.824823850448188
```

```
rmse= 1.0248253650440186
r2= 0.8838571885601488
cv_score= [0.87865198 0.91763212 0.92933032 0.81443904 0.89547829]
```

In [22]:

```
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.model_selection import cross_val_score
svr=SVR()
model=svr.fit(x_train,y_train)
y_pred=model.predict(x_test)
rmse=mean_squared_error(y_test, y_pred)**0.5
print("rmse=",rmse)
r2=r2_score(y_test, y_pred)
print("r2=",r2)
score=cross_val_score(model,x,y,cv=5,scoring="r2")
print("cv_score=",score)
```

```
rmse= 1.9492344860960387
r2= 0.8674808595299256
cv_score= [0.89985983 0.9464921  0.90928964 0.82899857 0.92371706]
```

In [23]:

```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.model_selection import cross_val_score
neigh = KNeighborsRegressor(n_neighbors=5)
model = neigh.fit(x_train,y_train)
y_pred=model.predict(x_test)
rmse=mean_squared_error(y_test, y_pred)**0.5
print("rmse=",rmse)
r2=r2_score(y_test, y_pred)
print("r2=",r2)
score=cross_val_score(model,x,y,cv=5,scoring="r2")
print("cv_score=",score)
```

```
rmse= 1.469091556030461
r2= 0.9247254509114142
cv_score= [0.93979721 0.95178272 0.96532907 0.89134222 0.95581669]
```

In [24]:

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.model_selection import cross_val_score
rf=RandomForestRegressor(max_depth=2, random_state=0)
model=rf.fit(x_train,y_train)
y_pred=model.predict(x_test)
rmse=mean_squared_error(y_test, y_pred)**0.5
print("rmse=",rmse)
r2=r2_score(y_test, y_pred)
print("r2=",r2)
score=cross_val_score(model,x,y,cv=5,scoring="r2")
print("cv_score=",score)
```

```
rmse= 1.782930419039334
r2= 0.8891286777584831
cv_score= [0.85944441 0.89932237 0.78180718 0.82561401 0.78039484]
```

In [25]:

```
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.model_selection import cross_val_score
gbr = GradientBoostingRegressor(random_state=0)
model=gbr.fit(x_train,y_train)
y_pred=model.predict(x_test)
rmse=mean_squared_error(y_test, y_pred)**0.5
print("rmse=",rmse)
r2=r2_score(y_test, y_pred)
print("r2=",r2)
score=cross_val_score(model,x,y,cv=5,scoring="r2")
print("cv_score=",score)
```

```
rmse= 0.742559954434567
r2= 0.980768485509912
cv_score= [0.98166668 0.98692979 0.96637465 0.96691346 0.98703714]
```

In [26]:

```
from sklearn.model_selection import GridSearchCV
```

```
from sklearn.model_selection import GridSearchCV
param_test1 = {'n_estimators':range(20,81,10)}
gsearch1 = GridSearchCV(estimator = GradientBoostingRegressor(learning_rate=0.1, min_samples_split=50,min_samples
param_grid = param_test1, scoring='r2', cv=10)
gsearch1.fit(x_train,y_train)
```

Out[26]:
```
GridSearchCV(cv=10,
             estimator=GradientBoostingRegressor(max_depth=4,
                                                 min_samples_leaf=5,
                                                 min_samples_split=50,
                                                 random_state=10),
             param_grid={'n_estimators': range(20, 81, 10)}, scoring='r2')
```

In [27]:
```
tuned_model = gsearch1.best_estimator_
```

In [28]:
```
gsearch1.best_score_
```

Out[28]:
```
0.9608834734464752
```

In [29]:
```
tuned_model.fit(x_train,y_train)
y_pred=tuned_model.predict(x_test)
```

In [30]:
```
y_pred
```

Out[30]:
```
array([12.16042115,  6.84525392,  9.98849457, 14.3573196 , 12.08081181,
        9.87561097, 12.26135347,  4.79432056, 10.90081364, 19.93960889,
       12.82407769, 13.45591568, 14.06306802,  9.89160066,  5.31836835,
       17.74983088, 23.62551705, 15.66598114, 21.52746405, 19.20127874,
       21.31251125, 15.35582961, 10.2067217 , 19.45350149,  9.81513801,
        7.77851175,  8.89799929, 13.20200216, 23.52190737, 20.92729843,
        7.13165879, 10.11861334, 22.0761659 , 14.64637418, 18.46717738,
       12.28687081, 16.14031805, 12.36063915, 11.92208718, 19.30590136])
```

In [ ]: