```
#multiple linear regression
#multiple number of independent variables
#one dependet variable
#eg predicting the sales of a shope for diff kind of products
#equation is
#y=m1x1+m2x2+m3x3....mnxn+c
```

```
import numpy as np
import pandas as pd
df=pd.read_csv("/content/Advertising.csv")
df
```

	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	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

200 rows × 4 columns

```
df.head()
```

	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	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9

```
df.tail()
```

	TV	Radio	Newspaper	Sales
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

```
df.columns
```

```
Index(['TV', 'Radio', 'Newspaper', 'Sales'], dtype='object')
```

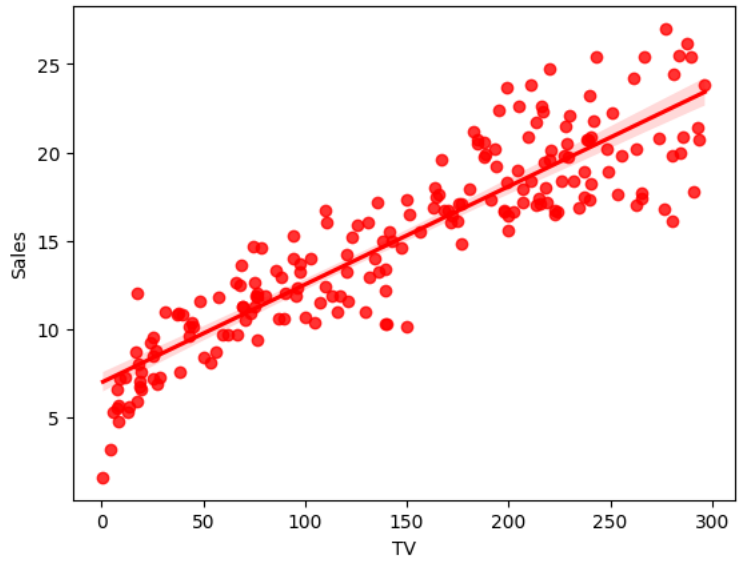
```
df.isna().sum()
```

```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

```
x=df.iloc[:,:-1]
y=df.iloc[:,-1]
```

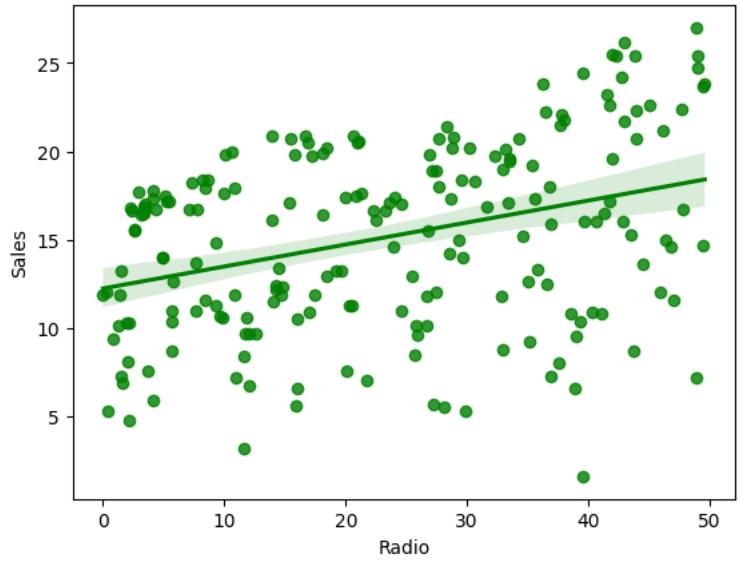
```
#seaborn ====>visualization tool
import seaborn as sns
sns.regplot(x=df['TV'],y=y,color='red')
```

```
<Axes: xlabel='TV', ylabel='Sales'>
```



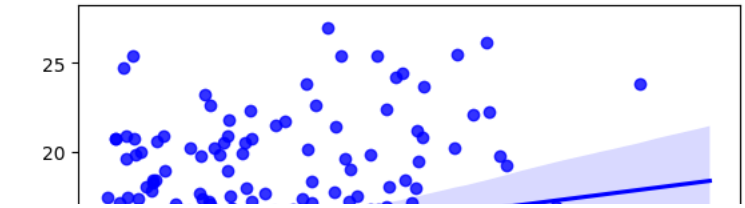

```
sns.regplot(x=df['Radio'],y=y,color='green')
```

```
<Axes: xlabel='Radio', ylabel='Sales'>
```




```
sns.regplot(x=df['Newspaper'],y=y,color='blue')
```

```
<Axes: xlabel='Newspaper', ylabel='Sales'>
```




```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
x_train
```

	TV	Radio	Newspaper
169	284.3	10.6	6.4
97	184.9	21.0	22.0
31	112.9	17.4	38.6
12	23.8	35.1	65.9
35	290.7	4.1	8.5
...	...	...	...
106	25.0	11.0	29.7
14	204.1	32.9	46.0
92	217.7	33.5	59.0
179	165.6	10.0	17.6
102	280.2	10.1	21.4

140 rows × 3 columns

```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
y_pred
```

```
array([17.15991908, 20.53369503, 23.68914396,  9.5191455 , 21.60736836,
       12.78101318, 21.08636345,  8.76054246, 17.11499951, 16.68789636,
        8.97584663,  8.57645026, 18.33212325,  8.17863567, 12.64605571,
       14.94486946,  8.34939536, 17.83858948, 11.12172174, 20.37740648,
       20.9483297 , 13.04035779, 11.01360656, 22.51142595,  9.40369784,
        7.98591291, 20.86943368, 13.77882255, 10.83407064,  8.00419229,
       15.88597618, 10.7027424 , 20.9521718 , 10.84679243, 21.50720813,
       21.07347295, 12.22673775, 22.85273767, 12.57698182,  6.54597206,
       11.93411853, 15.23490068, 10.07411153,  9.52159696, 17.11786382,
        7.28032677, 10.49404864, 15.24356754, 11.20742176, 11.78392665,
       14.01472163, 14.59884572, 10.82722434,  9.55839415,  9.03749681,
       12.51183313, 10.52551021, 25.01900824,  7.99334943, 15.73916263])
```

```
list(zip(x,model.coef_))
```

```
[('TV', 0.05358869132706914),
 ('Radio', 0.10270676778771287),
 ('Newspaper', 0.007931667677316324)]
```

```
print('intercept is',model.intercept_)
```

```
intercept is 4.743766701589685
```

```
df1=pd.DataFrame({'actual value':y_test,'predicted values':y_pred})
df1
```

	actual value	predicted values
95	16.9	17.159919
15	22.4	20.533695
30	21.4	23.689144
158	7.3	9.519146
128	24.7	21.607368
115	12.6	12.781013
69	22.3	21.086363
170	8.4	8.760542
174	16.5	17.115000
45	16.1	16.687896
66	11.0	8.975847
182	8.7	8.576450
165	16.9	18.332123
78	5.3	8.178636
186	10.3	12.646056
177	16.7	14.944869
56	5.5	8.349395
152	16.6	17.838589
82	11.3	11.121722
68	18.9	20.377406
124	19.7	20.948330
16	12.5	13.040358
148	10.9	11.013607
93	22.2	22.511426
65	11.3	9.403698
60	8.1	7.985913
84	21.7	20.869434
67	13.4	13.778823
125	10.6	10.834071
132	5.7	8.004192
9	15.6	15.885976
18	11.3	10.702742
55	23.7	20.952172
75	8.7	10.846792
150	16.1	21.507208
104	20.7	21.073473
135	11.6	12.226738
137	20.8	22.852738
164	11.9	12.576982
76	6.9	6.545972
79	11.0	11.934119
197	14.8	15.234901
38	10.1	10.074112
24	9.7	9.521597
122	16.6	17.117864
195	7.6	7.280327

29	10.5	10.494049
19	14.6	15.243568
143	10.4	11.207422
86	12.0	11.783927
114	14.6	14.014722
173	16.7	14.598846
5	7.2	10.827224
126	6.6	9.558394
117	9.4	9.037497
73	11.0	12.511833

```
from sklearn.metrics import mean_absolute_error
print('mean absolute error is',mean_absolute_error(y_test,y_pred))
```

```
mean absolute error is 1.1594875061090582
```

```
from sklearn.metrics import mean_absolute_percentage_error
print('percentage is',mean_absolute_percentage_error(y_test,y_pred))
```

```
percentage is 0.10536440823029307
```

```
from sklearn.metrics import mean_squared_error
print("mean squared error is",mean_squared_error(y_test,y_pred))
```

```
mean squared error is 2.541624036229147
```

```
z=mean_squared_error(y_test,y_pred)
print('root mean squared error is',np.sqrt(z))
```

```
root mean squared error is 1.5942471691143587
```

```
from sklearn.metrics import r2_score
print('r12 score is',r2_score(y_test,y_pred))
```

```
r12 score is 0.9091484341849799
```