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

In [3]:
```
#To load the dataset
adv = pd.read_csv(r"C:\Users\user\Downloads\advertising.csv")
adv
```

Out[3]:

	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

In [9]:
```
#To view the first few rows of the dataset
adv.head()
```

Out[9]:

	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

In [15]:
```
#To get columns names
adv.columns
```

Out[15]:
```
Index(['TV', 'Radio', 'Newspaper', 'Sales'], dtype='object')
```

In [17]:
```
#To get shape of dataset
adv.shape
```

Out[17]:
```
(200, 4)
```

In [19]:
```
#To get information of the dataset
adv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   TV         200 non-null    float64
 1   Radio      200 non-null    float64
 2   Newspaper  200 non-null    float64
 3   Sales      200 non-null    float64
dtypes: float64(4)
memory usage: 6.4 KB
```

In [23]:
```
#To check missing values
adv.isnull().sum()
```

Out[23]:
```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

In [27]:
```
#To check datatypes
adv.dtypes
```

Out[27]:
```
TV           float64
Radio        float64
Newspaper    float64
Sales        float64
dtype: object
```

In [29]:
```
#To check unique values in each columns
for i in adv.columns:
    print(i,':','\n',adv[i].unique(),'\n')
```

TV :
 [230.1 44.5 17.2 151.5 180.8   8.7 57.5 120.2   8.6 199.8 66.1 214.7
  23.8 97.5 204.1 195.4 67.8 281.4 69.2 147.3 218.4 237.4 13.2 228.3
  62.3 262.9 142.9 240.1 248.8 70.6 292.9 112.9 97.2 265.6 95.7 290.7
 266.9 74.7 43.1 228. 202.5 177. 293.6 206.9 25.1 175.1 89.7 239.9
 227.2 66.9 100.4 216.4 182.6 262.7 198.9   7.3 136.2 210.8 210.7 53.5
 261.3 239.3 102.7 131.1 69.   31.5 139.3 216.8 199.1 109.8 26.8 129.4
 213.4 16.9 27.5 120.5   5.4 116.   76.4 239.8 75.3 68.4 213.5 193.2
  76.3 110.7 88.3 134.3 28.6 217.7 250.9 107.4 163.3 197.6 184.9 289.7
 135.2 222.4 296.4 280.2 187.9 238.2 137.9 25.   90.4 13.1 255.4 225.8
 241.7 175.7 209.6 78.2 75.1 139.2 125.7 19.4 141.3 18.8 224. 123.1
 229.5 87.2   7.8 80.2 220.3 59.6   0.7 265.2   8.4 219.8 36.9 48.3
  25.6 273.7 43.   73.4 193.7 220.5 104.6 96.2 140.3 243.2 38.   44.7
 280.7 121. 171.3 187.8   4.1 93.9 149.8 11.7 131.7 172.5 85.7 188.4
 163.5 117.2 234.5 17.9 206.8 215.4 284.3 50. 164.5 19.6 168.4 276.9
 248.4 170.2 276.7 165.6 156.6 218.5 56.2 287.6 253.8 205. 139.5 191.1
 286.   18.7 39.5 75.5 166.8 149.7 38.2 94.2 283.6 232.1]

Radio :
 [37.8 39.3 45.9 41.3 10.8 48.9 32.8 19.6 2.1 2.6 5.8 24. 35.1 7.6
 32.9 47.7 36.6 39.6 20.5 23.9 27.7 5.1 15.9 16.9 12.6 3.5 29.3 16.7
 27.1 16. 28.3 17.4 1.5 20.   1.4 4.1 43.8 49.4 26.7 37.7 22.3 33.4
  8.4 25.7 22.5 9.9 41.5 15.8 11.7 3.1 9.6 41.7 46.2 28.8 28.1 19.2
 49.6 29.5 2. 42.7 15.5 29.6 42.8 9.3 24.6 14.5 27.5 43.9 30.6 14.3
 33.   5.7 43.7 1.6 28.5 29.9 7.7 20.3 44.5 43. 18.4 40.6 25.5 47.8
  4.9 33.5 36.5 14. 31.6 21. 42.3 4.3 36.3 10.1 17.2 34.3 46.4 11.
  0.3 0.4 26.9 8.2 38. 15.4 20.6 46.8 35.   0.8 36.9 26.8 21.7 2.4
 34.6 32.3 11.8 38.9 0. 49. 12.   2.9 27.2 38.6 47. 39. 28.9 25.9
 17. 35.4 33.2 14.8 1.9 7.3 40.3 25.8 13.9 23.3 39.7 21.1 11.6 43.5
  1.3 18.1 35.8 36.8 14.7 3.4 37.6 5.2 23.6 10.6 20.9 20.1 7.1 30.2
  7.8 2.3 10.   5.4 21.3 45.1 28.7 12.1 41.1 42. 35.6 3.7 8.6]

Newspaper :
 [69.2 45.1 69.3 58.5 58.4 75.   23.5 11.6   1.   21.2 24.2   4.
  65.9   7.2 46.   52.9 114.   55.8 18.3 19.1 53.4 49.6 26.2 19.5
  12.6 22.9 40.8 43.2 38.6 30.   0.3   7.4   8.5   5.   45.7 35.1
  32.   31.6 38.7   1.8 26.4 43.3 31.5 35.7 18.5 49.9 36.8 34.6
   3.6 39.6 58.7 15.9 60.   41.4 16.6 37.7   9.3 21.4 54.7 27.3
   8.4 28.9   0.9   2.2 10.2 11.   27.2 31.7 19.3 31.3 13.1 89.4
  20.7 14.2   9.4 23.1 22.3 36.9 32.5 35.6 33.8 65.7 16.   63.2
  73.4 51.4 33.   59.   72.3 10.9   5.9 22.   51.2 45.9 49.8 100.9
  17.9   5.3 29.7 23.2 25.6   5.5 56.5   2.4 10.7 34.5 52.7 14.8
  79.2 46.2 50.4 15.6 12.4 74.2 25.9 50.6   9.2   3.2 43.1   8.7
  43.   2.1 65.6 59.7 20.5   1.7 12.9 75.6 37.9 34.4 38.9   9.
  44.3 11.9 20.6 37.   48.7   9.5   5.7 50.5 24.3 45.2 30.7 49.3
   5.4 84.8 21.6 19.4 57.6   6.4 18.4 47.4 17.   12.8 41.8 20.3
  35.2 23.7 17.6   8.3 27.4 71.8 19.6 26.6 18.2   3.7 23.4   5.8
   6.   13.8   8.1 66.2]

Sales :
 [22.1 10.4 12. 16.5 17.9 7.2 11.8 13.2 4.8 15.6 12.6 17.4 9.2 13.7
 19. 22.4 12.5 24.4 11.3 14.6 18. 17.5 5.6 20.5 9.7 17. 15. 20.9
 18.9 10.5 21.4 11.9 17.8 25.4 14.7 10.1 21.5 16.6 17.1 20.7 8.5 16.1
 10.6 23.2 19.8 16.4 10.7 22.6 21.2 20.2 23.7 5.5 23.8 18.4 8.1 24.2
 14. 16. 11. 13.4 22.3 18.3 12.4 8.8 8.7 6.9 14.2 5.3 17.3 13.6
 21.7 12.9 16.7 7.3 19.4 22.2 11.5 16.9 17.2 19.7 21.8 12.2 9.4 15.9

```
 6.6 15.5  7.  15.2 24.7  1.6 17.7  5.7 19.6 10.8 11.6  9.5 20.8  9.6
10.9 19.2 20.1 12.3 10.3 18.2 20.6  3.2 15.3 13.3 19.9  8.  20.   8.4
 7.6 27.  16.8 17.6 26.2  6.7  5.9 14.8 25.5]
```

In [31]:
```
#To get statistical summary of the numerical columns
adv.describe()
```

Out[31]:

	TV	Radio	Newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000
mean	147.042500	23.264000	30.554000	15.130500
std	85.854236	14.846809	21.778621	5.283892
min	0.700000	0.000000	0.300000	1.600000
25%	74.375000	9.975000	12.750000	11.000000
50%	149.750000	22.900000	25.750000	16.000000
75%	218.825000	36.525000	45.100000	19.050000
max	296.400000	49.600000	114.000000	27.000000

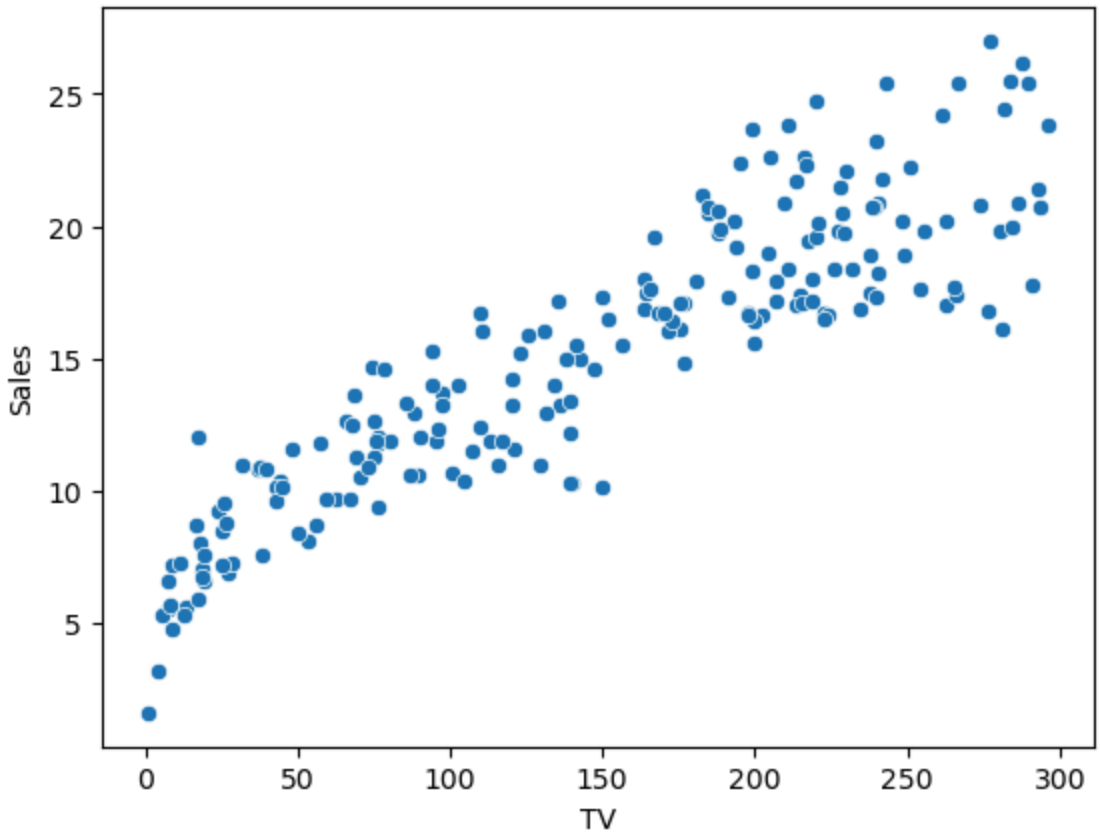
In [33]:
```
adv.describe().T
```

Out[33]:

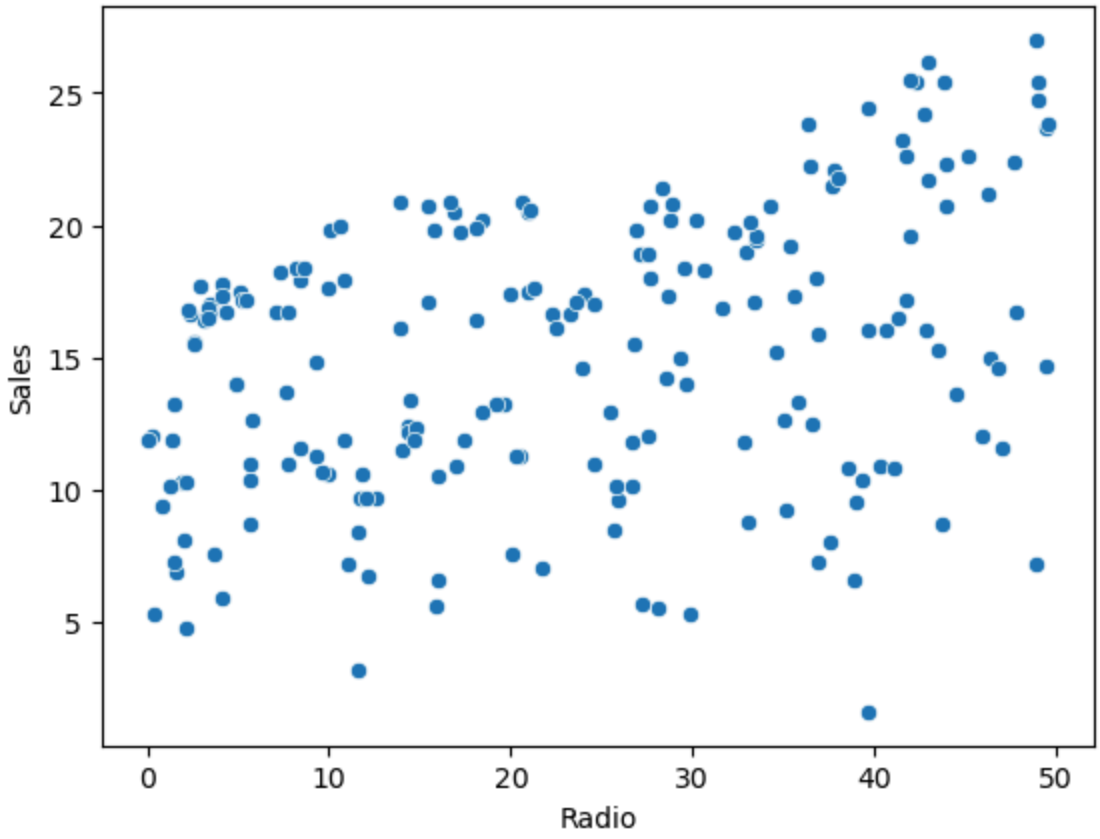
	count	mean	std	min	25%	50%	75%	max
TV	200.0	147.0425	85.854236	0.7	74.375	149.75	218.825	296.4
Radio	200.0	23.2640	14.846809	0.0	9.975	22.90	36.525	49.6
Newspaper	200.0	30.5540	21.778621	0.3	12.750	25.75	45.100	114.0
Sales	200.0	15.1305	5.283892	1.6	11.000	16.00	19.050	27.0

Data Visualization

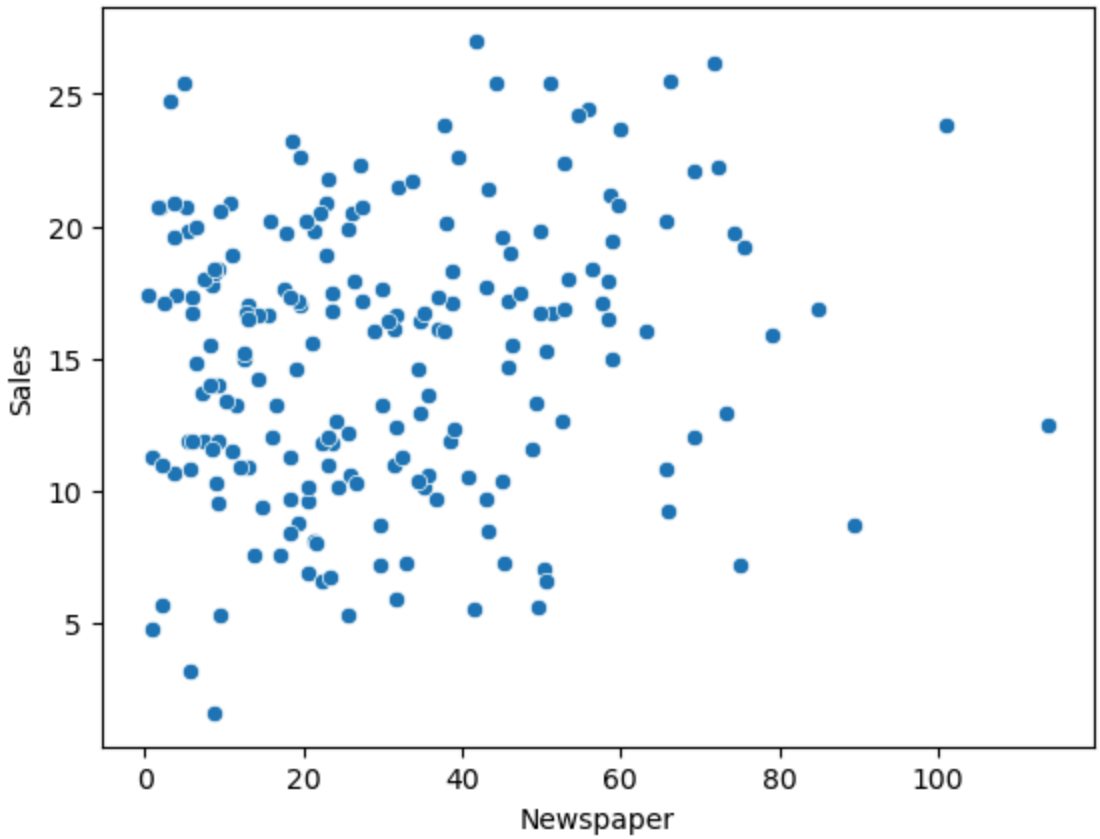
In [36]:
```
# Scatter plots to check the linearity assumption between each independent variable
sns.scatterplot(x=adv.TV,y=adv.Sales)
plt.show()
```



In [38]:
```
sns.scatterplot(x=adv.Radio,y=adv.Sales)
plt.show()
```

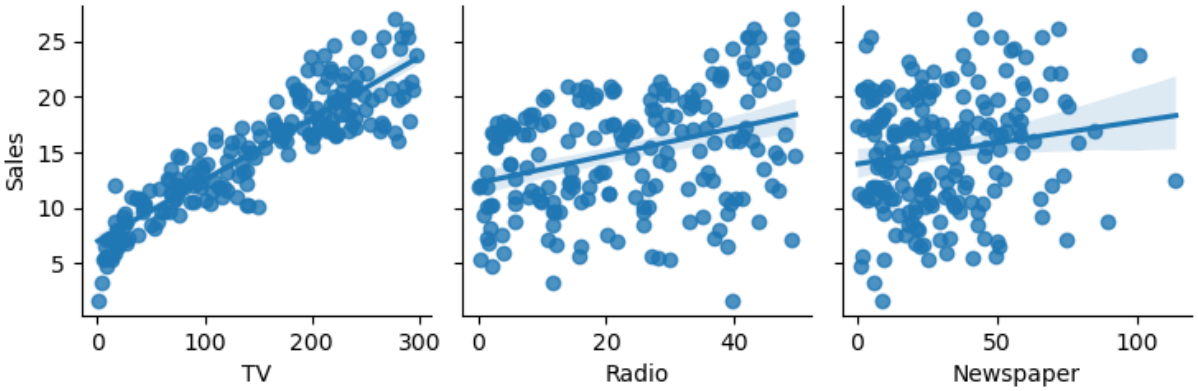


In [40]:
```
sns.scatterplot(x=adv.Newspaper,y=adv.Sales)
plt.show()
```



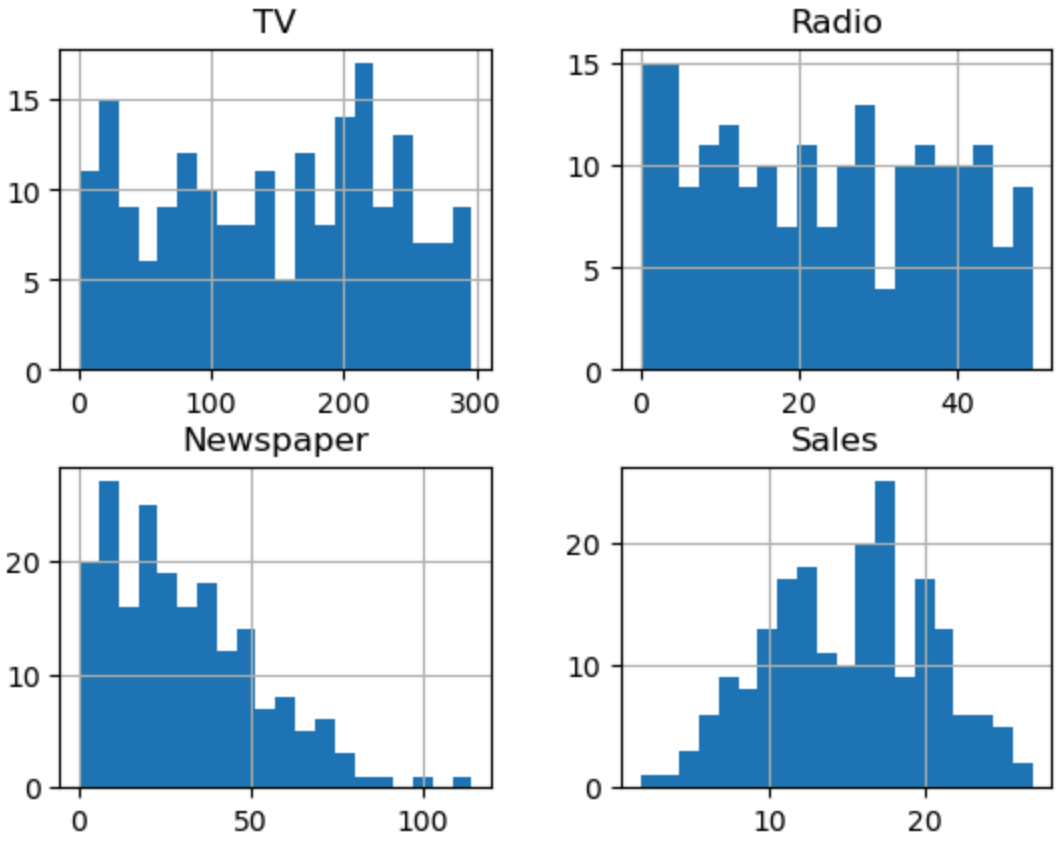
In [44]:
```
sns.pairplot(adv, x_vars=["TV", "Radio", "Newspaper"], y_vars="Sales", kind="reg")
```

Out[44]:
```
<seaborn.axisgrid.PairGrid at 0x15fa75b90d0>
```

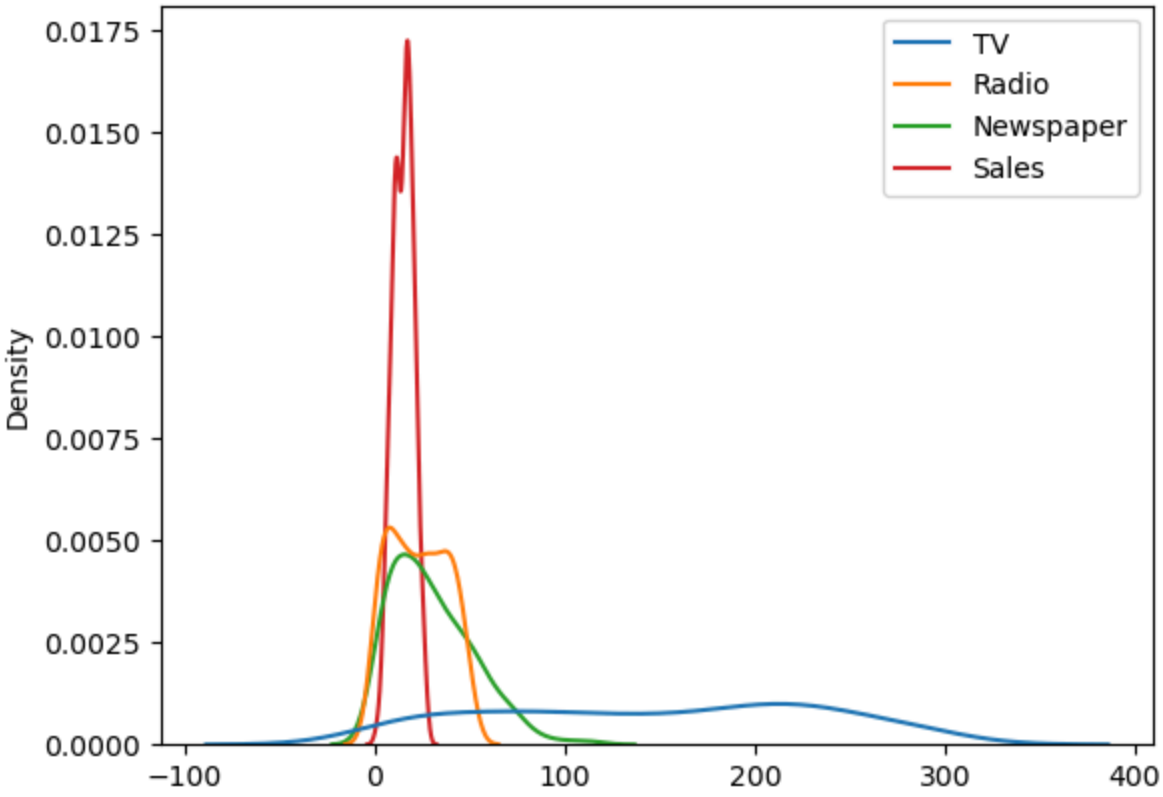


In [46]:
```
# Histograms to check the normality assumption of the dependent variable (Sales)
adv.hist(bins=20)
```

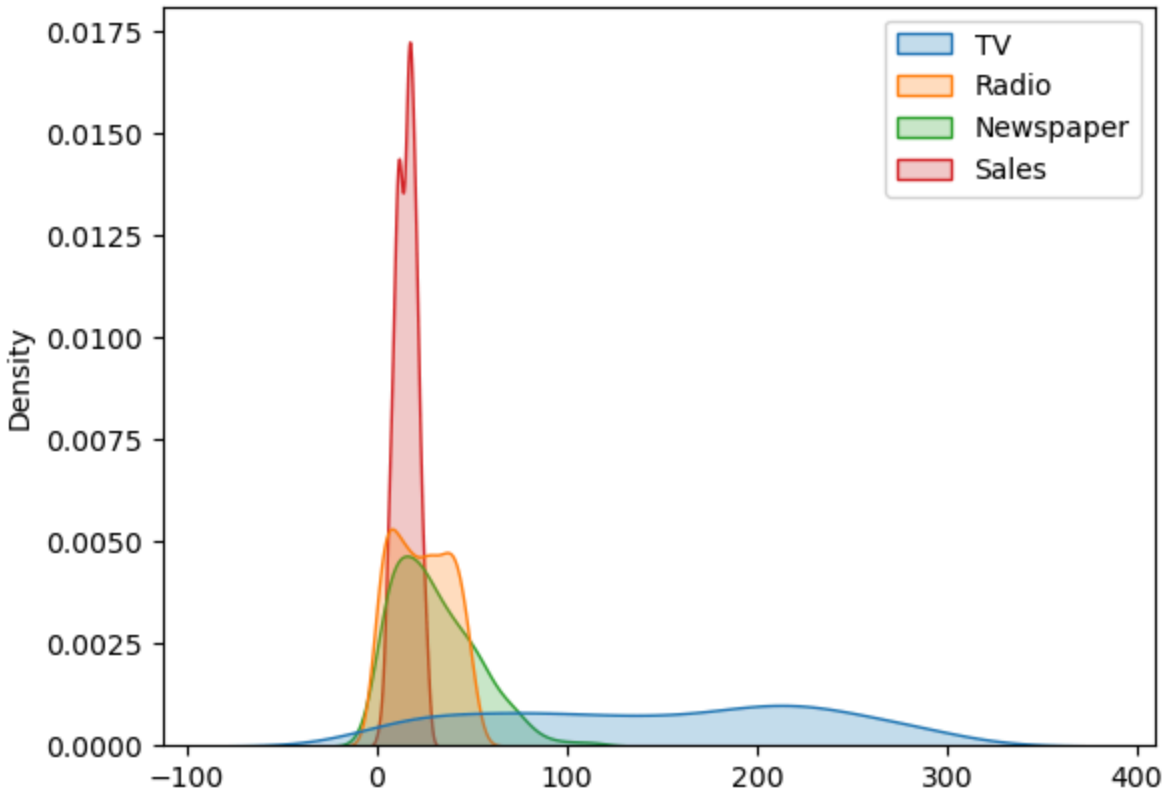
Out[46]:
```
array([[<Axes: title={'center': 'TV'}>,
        <Axes: title={'center': 'Radio'}>],
       [<Axes: title={'center': 'Newspaper'}>,
        <Axes: title={'center': 'Sales'}>]], dtype=object)
```



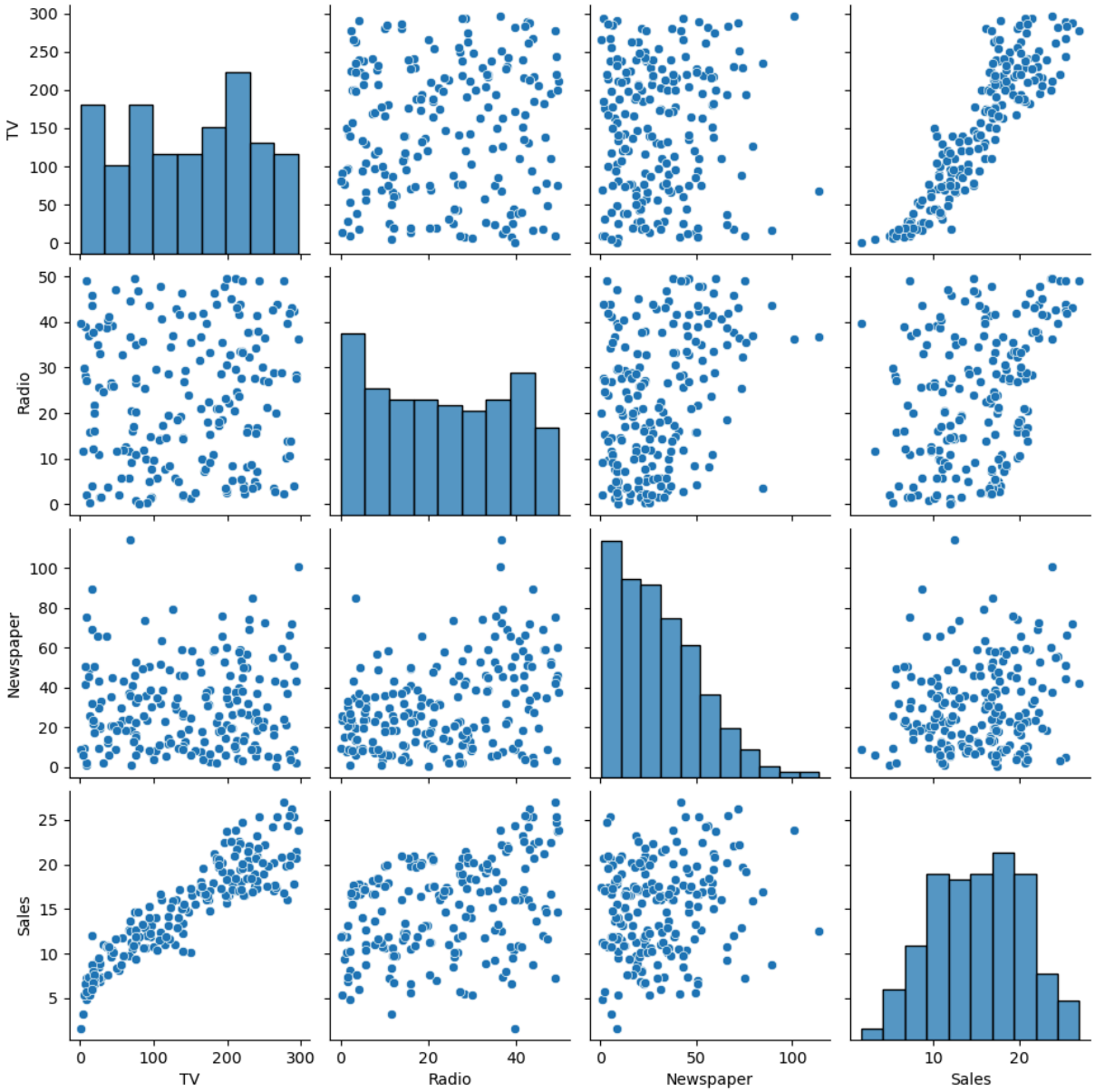
In [48]:
```
sns.kdeplot(data=adv)
plt.show()
```



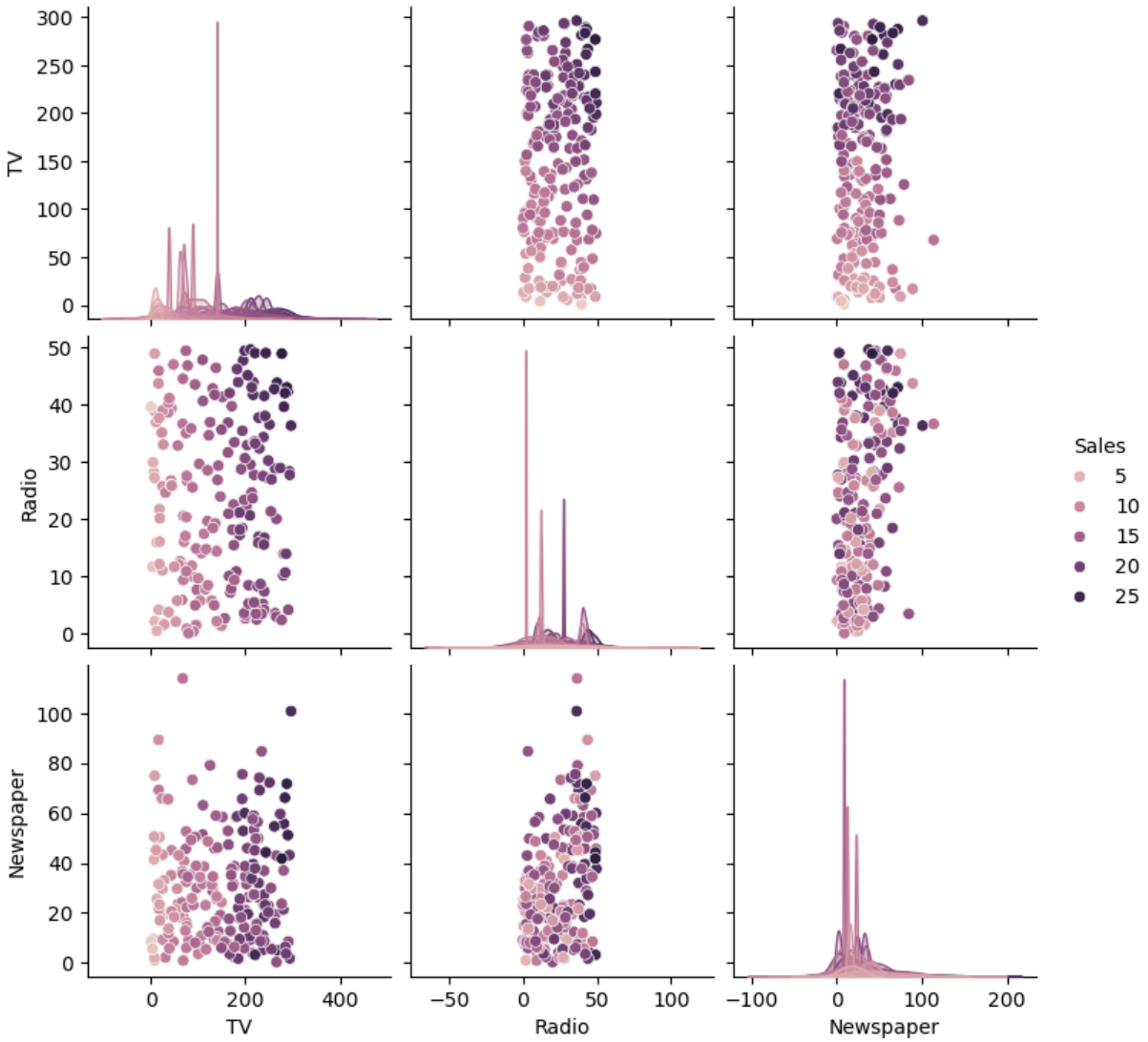
In [50]:
```
sns.kdeplot(data=adv,fill=True)
plt.show()
```



In [52]:
```
sns.pairplot(adv)
plt.show()
```

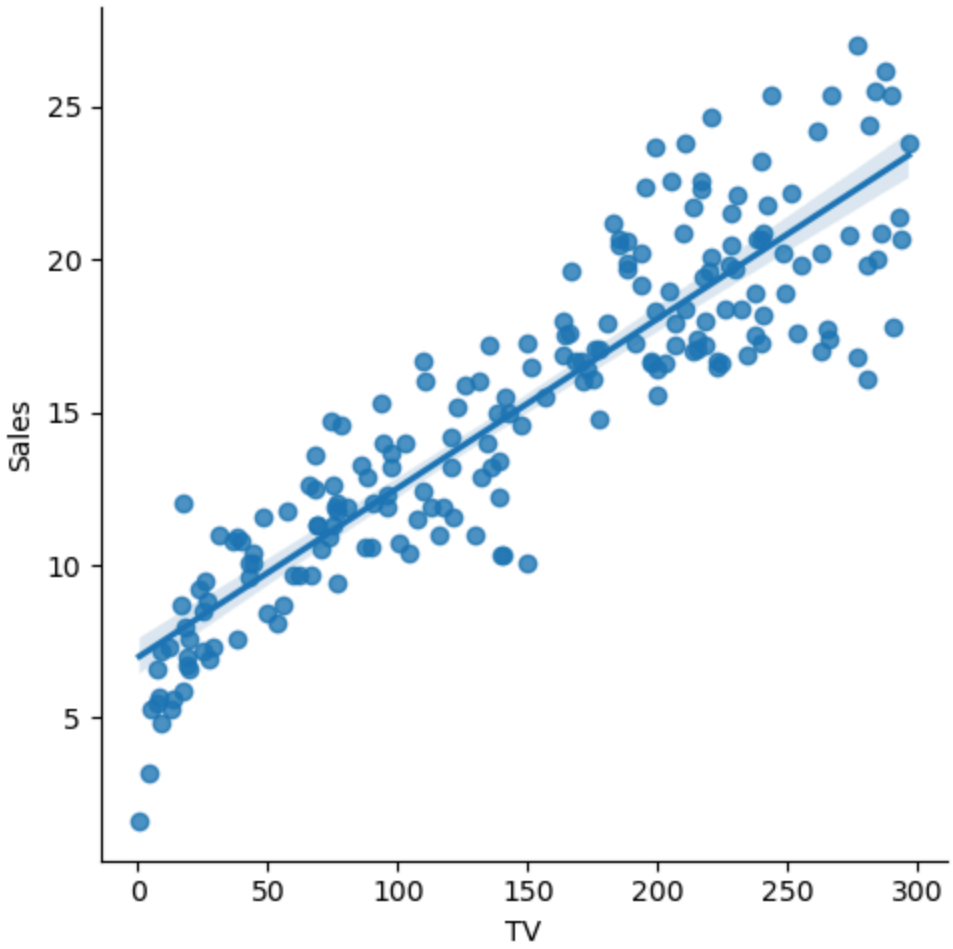


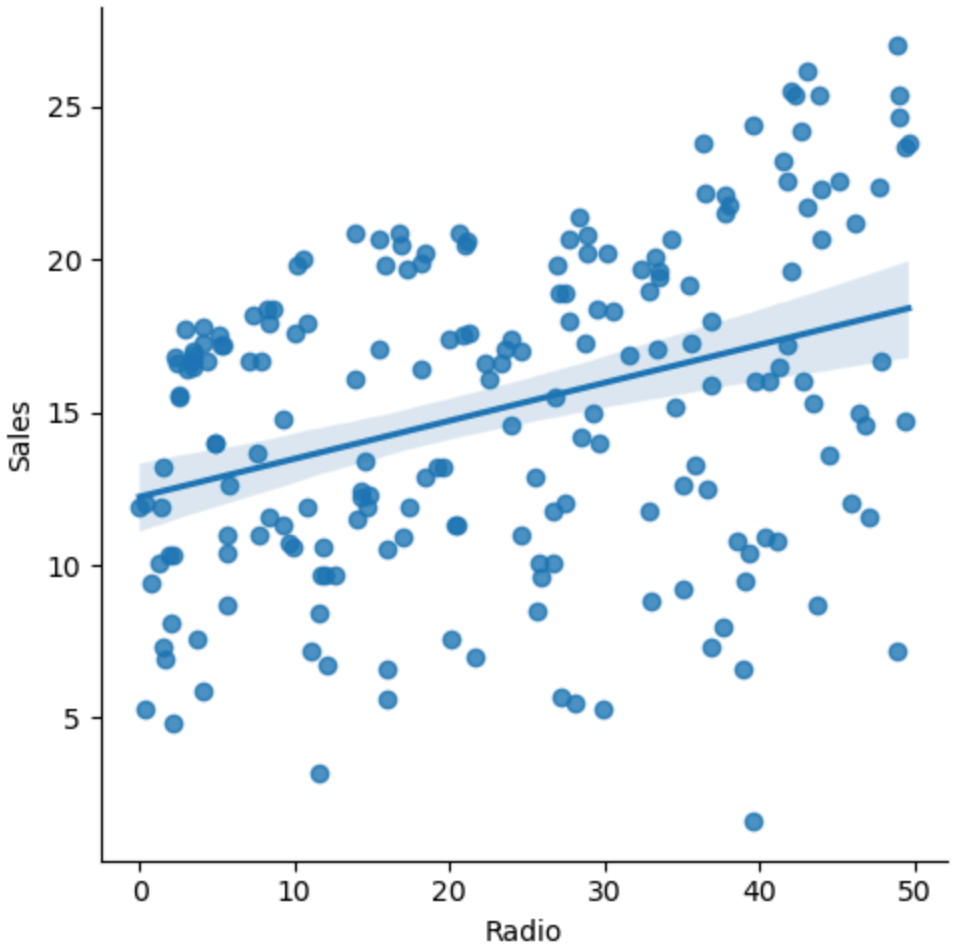
In [54]:
```
sns.pairplot(adv,hue='Sales')
plt.show()
```

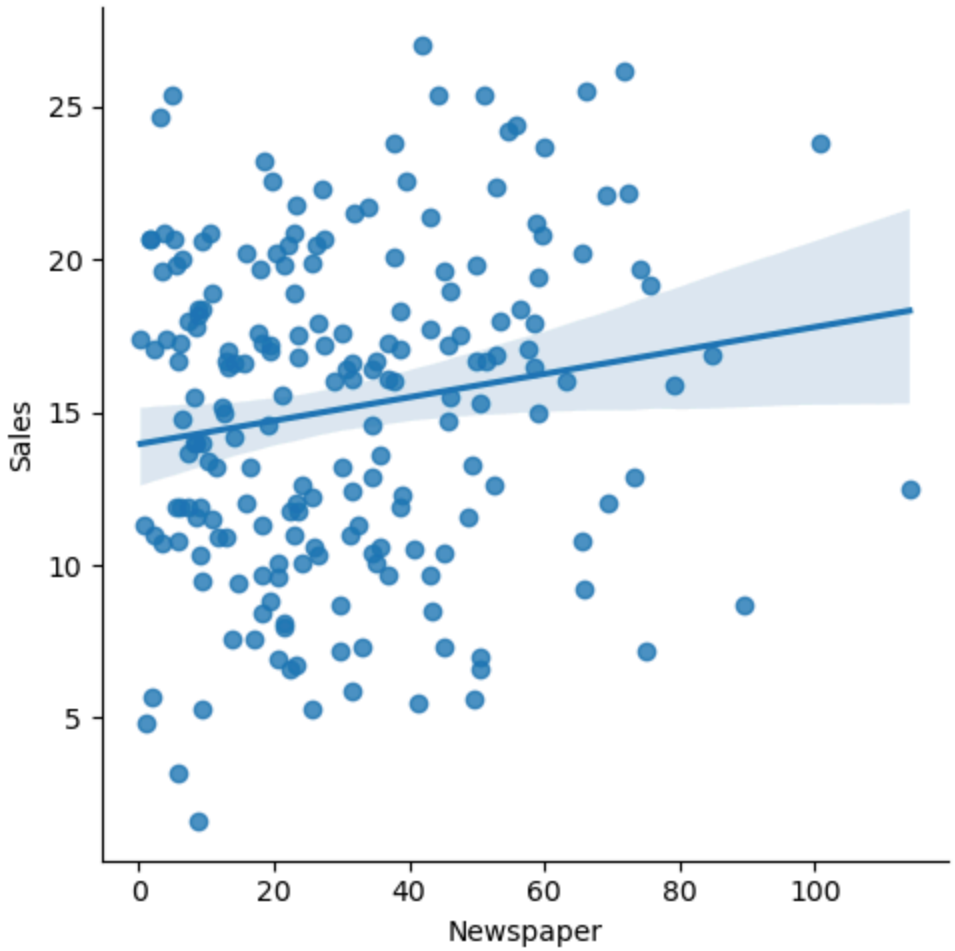


In [60]:
```
# Linear regression plots to visualize the relationship between each independent va
sns.lmplot(x='TV', y='Sales', data=adv)
sns.lmplot(x='Radio', y='Sales', data=adv)
sns.lmplot(x='Newspaper',y= 'Sales', data=adv)
```

Out[60]: <seaborn.axisgrid.FacetGrid at 0x15fab402350>


25
20
15
10
5
Sales
0
50
100
150
200
250
300
TV


Sales
25
20
15
10
5
0
10
20
30
40
50
Radio

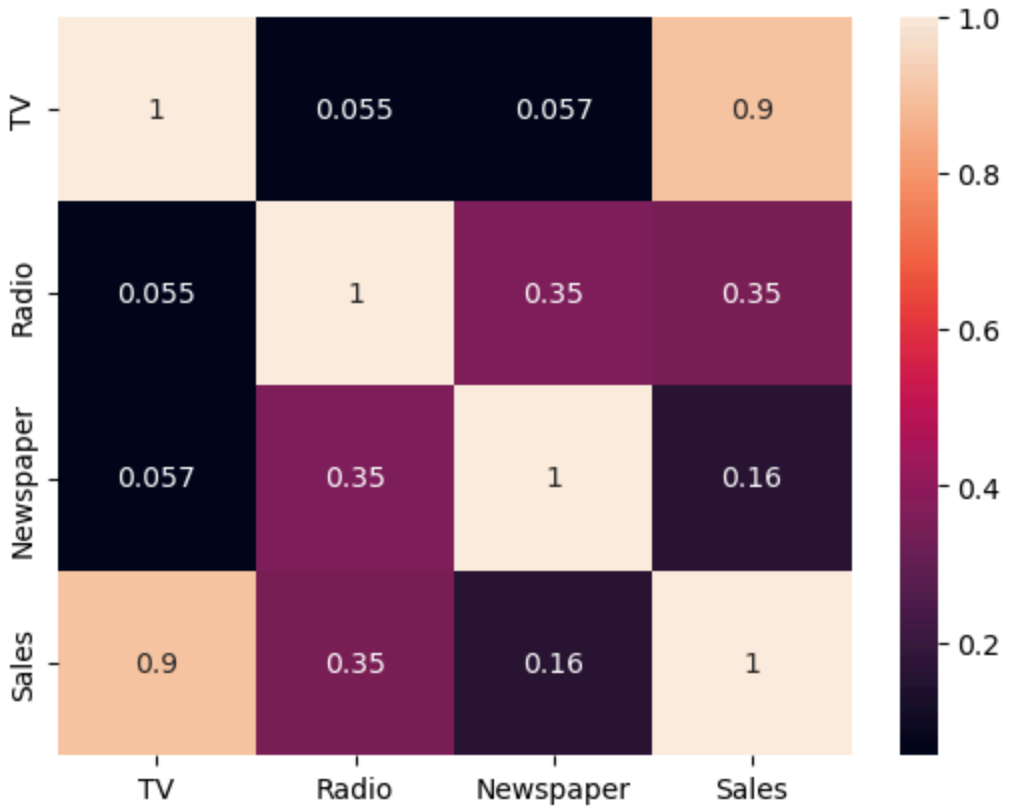


In [56]:
```
c = adv.corr()
c
```

Out[56]:

	TV	Radio	Newspaper	Sales
TV	1.000000	0.054809	0.056648	0.901208
Radio	0.054809	1.000000	0.354104	0.349631
Newspaper	0.056648	0.354104	1.000000	0.157960
Sales	0.901208	0.349631	0.157960	1.000000

In [58]:
```
sns.heatmap(c,annot=True)
plt.show()
```



In [62]:
```
#ip op creation
ip = adv.drop('Sales',axis=1)
op = adv.Sales
```

In [64]:
```
ip.head()
```

Out[64]:

	TV	Radio	Newspaper
0	230.1	37.8	69.2
1	44.5	39.3	45.1
2	17.2	45.9	69.3
3	151.5	41.3	58.5
4	180.8	10.8	58.4

In [66]:
```
op.head()
```

Out[66]:
```
0    22.1
1    10.4
2    12.0
3    16.5
4    17.9
Name: Sales, dtype: float64
```

In [68]:
```
#Train Test Split
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(ip,op,test_size=0.2,random_stat
```

In [70]:
```
x_train
```

Out[70]:

	TV	Radio	Newspaper
156	93.9	43.5	50.5
115	75.1	35.0	52.7
155	4.1	11.6	5.7
15	195.4	47.7	52.9
61	261.3	42.7	54.7
...	...	...	...
0	230.1	37.8	69.2
184	253.8	21.3	30.0
131	265.2	2.9	43.0
152	197.6	23.3	14.2
106	25.0	11.0	29.7

160 rows × 3 columns

In [72]:
```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(160, 3)
(160,)
(40, 3)
(40,)
```

In [74]:
```
#Standard Scaler Transform
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()

x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
```

In [76]:
```
x_train
```

Out[76]:
```
array([[-0.62579748,  1.39757261,  0.91683005],
       [-0.83959224,  0.81633695,  1.01775389],
       [-1.64700866, -0.78377065, -1.13834642],
       [ 0.52846684,  1.68477142,  1.02692879],
       [ 1.27788574,  1.34286808,  1.10950284],
       [ 1.45528991,  1.76682822,  0.51772212],
       [-0.08676173,  0.25561548,  0.71956981],
       [-1.68567367,  1.13088801, -1.00072299],
       [ 0.90260768, -0.42135312, -0.19791969],
       [ 0.25440014,  1.13772608,  0.32963677],
       [-0.40972829, -0.38716279,  0.3709238 ],
       [ 0.44317637, -0.40083892, -0.57867783],
       [-0.44498169,  1.69160948,  0.95811707],
       [-1.59810879,  0.28296775, -1.30349452],
       [ 1.21079057,  0.26245355, -1.14752131],
       [-1.60493203,  1.08302155,  0.92141749],
       [ 1.50646419,  1.13088801,  1.15996476],
       [ 1.6372429 ,  0.35818648,  0.58194638],
       [ 1.45301549, -1.41971086, -0.31260587],
       [ 0.4488624 , -0.33929632, -0.22544437],
       [-0.32330062,  0.37186261, -0.74841338],
       [-0.22208927, -1.18721659,  0.03604013],
       [-0.44498169, -0.59914285,  0.05438992],
       [-1.63222498,  0.46759555, -0.96861086],
       [ 1.64520335,  0.31715808, -1.31725687],
       [ 0.80594516,  0.71376595,  0.66910788],
       [-1.49803465, -1.29662566,  0.04980248],
       [-0.58485805, -1.05729332, -1.0695347 ],
       [ 1.03679802, -1.07780752, -1.00072299],
       [ 0.73316396,  0.10517801, -0.7988753 ],
       [-1.59469717,  1.76682822,  2.04075468],
       [-0.80433885,  1.62322882,  0.18283845],
       [ 1.49281771, -0.88634166, -0.41811716],
       [ 0.79116147, -1.20773079, -0.14287032],
       [-1.47983935, -0.09312592,  0.9122426 ],
       [ 0.75590808,  0.03679735,  1.24253882],
       [ 0.17707012, -0.14783045,  0.77461918],
       [-0.98515464, -0.71538999, -0.56032804],
       [-0.59964173, -0.56495252,  0.38468614],
       [ 0.78206382,  0.71376595,  1.30676308],
       [-1.59583438, -1.43338699, -1.35395645],
       [ 0.38290444,  1.58220042,  1.29300074],
       [ 1.03679802, -0.43502925, -0.34930545],
       [-0.12542674,  1.59587655,  1.30676308],
       [-0.2641659 ,  0.94626021,  2.23342747],
       [ 0.16569805,  0.93942215, -1.06035981],
       [ 0.65924555, -1.00258879, -0.18874479],
       [-0.14475924, -0.26407759, -0.63831464],
       [ 0.97311447, -1.34449212,  2.49032453],
       [-1.54352289, -0.48973379,  0.87554302],
       [ 0.08723082, -1.39919666, -1.01907278],
       [ 0.47956697,  0.38553875, -0.56491548],
       [ 0.26804661, -0.33929632,  0.00851545],
       [-0.43474683,  1.19926868,  1.49943587],
       [-1.27400502,  1.06250735,  1.60953461],
       [-0.52572333,  0.44708135, -1.01448533],
```

```
[-0.85892475, -0.41451505, -0.8080502 ],
[ 0.20322586,  1.29500161, -1.23468281],
[-1.14436351,  1.63690495, -1.00989789],
[ 0.29761398, -0.03842139,  0.04521503],
[ 1.61222436, -1.29662566, -1.00989789],
[-0.90896182, -0.94104619, -1.35854389],
[ 0.57850391, -1.36500632,  0.1874259 ],
[-0.70199029, -0.77009452, -0.21168203],
[ 1.6008523 ,  1.31551581,  0.94894218],
[-0.9226083 ,  0.92574601,  3.82985919],
[-0.01852935,  0.05731155, -0.52362846],
[-1.5446601 , -1.54963412, -0.22544437],
[-1.40819535,  0.18039675,  0.58653383],
[ 1.00609345,  0.30348195, -0.8952117 ],
[-1.3809024 , -1.46757732, -0.4502293 ],
[ 0.50913433,  0.84368921,  2.06827936],
[ 0.30443721, -0.52392412, -1.28973218],
[-0.9419408 , -1.18037852, -0.28966864],
[ 0.73430116,  1.36338228,  0.15072632],
[ 0.74794764,  0.06414961, -1.21633302],
[ 0.56826906,  1.80101855,  1.35263756],
[-0.68948102,  0.16672061,  1.96735552],
[ 1.13118613,  0.48810975, -0.46857909],
[ 1.05499332,  1.02147895, -0.33554311],
[-1.40250932,  1.08985961, -0.97319831],
[ 0.57850391, -1.39919666, -0.42729206],
[ 1.41889931,  0.39921488,  1.33887521],
[-0.58826967, -1.47441539, -0.02359668],
[-0.89076652, -0.48289572,  0.47184764],
[ 0.89919607,  1.00096475,  0.06815227],
[ 0.63763863,  1.50698168, -0.50069122],
[-1.61061806,  0.34451035,  0.49937233],
[-0.10723144, -1.43338699, -0.1795699 ],
[ 0.0087636 ,  0.85736535, -1.12458407],
[-1.36839314, -1.47441539,  0.11402674],
[ 0.70245939,  0.44024328, -0.97319831],
[ 0.83551252, -1.34449212, -0.7988753 ],
[ 1.34156928,  1.41808681, -1.17045855],
[ 0.24189087, -1.04361719,  0.21495059],
[ 0.76728014,  1.27448741,  0.41679827],
[-0.83504342, -0.83847519, -1.12458407],
[ 0.89009842, -0.49657185,  0.88930536],
[ 0.55348537, -1.33765406, -1.12917152],
[-1.50144626,  1.41124875,  2.70134711],
[-0.31761459, -1.00258879,  0.83425599],
[-0.82480856,  0.24877741, -0.37683014],
[-0.15613131,  1.27448741,  0.70580746],
[ 0.91625416,  0.63170915,  2.0040551 ],
[ 0.44203917, -0.13415432, -0.96402341],
[-0.82480856, -1.52228186, -0.7208887 ],
[ 0.5034483 , -0.31878212,  1.61412206],
[-1.18757735,  1.11037381,  0.66910788],
[ 0.68995012, -0.16834465, -0.90897404],
[-0.36082843, -0.57179059, -1.15210876],
[-0.11064306, -0.59914285, -0.22544437],
[-0.90668741, -0.17518272, -0.56032804],
```

```
       [-1.25922134, -1.32397792, -0.76676317],
       [ 1.29380662,  0.39237681, -0.67042678],
       [ 1.55877567, -0.62649512, -1.23009537],
       [ 1.13573496,  0.27612968, -0.34930545],
       [-0.78159473, -1.57698639, -0.97778575],
       [-1.48097655, -0.74958032, -0.32636822],
       [-0.93284315, -0.77693259,  0.28834974],
       [-0.47227464, -0.61965705, -0.89979915],
       [ 0.79002427,  0.31715808,  1.04986602],
       [-0.84414107,  1.80101855,  0.69663257],
       [-0.3744749 , -1.05045526, -0.34013056],
       [-0.29373326,  0.78898468, -0.83098744],
       [ 1.3267856 , -0.20937305, -1.38606858],
       [-0.32671224, -0.23672532, -0.86768702],
       [ 1.02770037, -0.51708605, -0.14745776],
       [ 1.53148272,  1.29500161,  1.6370593 ],
       [ 0.94582152, -0.98891266, -1.00072299],
       [ 0.0099008 , -1.48809152, -0.28508119],
       [ 0.40906018, -0.14099239, -0.39059248],
       [ 0.87417753, -1.01626492,  1.19207689],
       [ 1.0151911 ,  0.76847048, -1.15669621],
       [ 0.77182897,  1.42492488, -0.15204521],
       [-0.16636616, -1.24192112, -0.97319831],
       [-1.01585921, -0.75641839,  0.57735893],
       [-1.08522879, -1.44022506, -0.41811716],
       [-1.49803465,  1.56168622,  1.77927017],
       [-1.33541416,  0.10517801, -1.29890708],
       [ 1.49850374, -0.62649512,  0.29752464],
       [ 1.0333864 , -1.29662566,  0.29293719],
       [-0.06856643,  0.42656715, -0.82181254],
       [ 0.8139056 ,  0.69325175,  0.33881167],
       [ 0.65810835, -1.22140692, -0.50986611],
       [ 1.15961629,  0.91890795,  1.91689359],
       [-1.4707417 , -0.20253499, -0.61996485],
       [-1.26149575,  1.17875448, -0.85392467],
       [-1.4900742 ,  0.99412668, -0.40894227],
       [-1.47301611, -0.48289572, -0.37683014],
       [-1.18530293,  0.18723481, -0.45481674],
       [-1.20463544,  0.19407288, -0.45940419],
       [ 1.53944317, -0.85215132, -1.10623428],
       [-0.66559969, -1.55647219, -0.33554311],
       [ 1.07205141,  1.77366628,  0.6324083 ],
       [ 1.00609345, -1.22824499, -0.32178077],
       [ 0.9230774 ,  1.00780281,  1.77468272],
       [ 1.19259527, -0.12047819, -0.02359668],
       [ 1.32223678, -1.37868246,  0.57277149],
       [ 0.55348537,  0.01628315, -0.74841338],
       [-1.40933256, -0.82479905, -0.03735903]])
```

In [78]:

```
#ML Model
#Linear Regression(It finds the linear relationship between dependent and independe
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[78]:
```
▾ LinearRegression
LinearRegression()
```

In [80]:
```
pred = lr.predict(x_test)
pred
```

Out[80]:
```
array([18.98830462, 11.16190101, 19.8439492 , 15.99746555,  8.22867142,
       10.73975484, 26.56971668, 10.65406475, 19.6116695 , 18.57732873,
       15.39253339, 12.67095722, 20.93218621, 10.37865631, 14.13693181,
       15.26667862, 17.50339018, 17.72427339, 18.48411937, 22.19719483,
       20.18422486, 11.1835828 ,  9.06389194, 10.78895478,  8.09691355,
       12.9804461 , 22.5892956 , 18.34380189, 26.60289811, 11.43130388,
       16.44808924, 23.16826577,  8.38172636, 10.08295603,  9.33407908,
       10.33156099, 16.61222749,  8.84520254, 14.13187574, 12.96395562])
```

In [82]:
```
y_test
```

Out[82]:
```
40     16.6
51     10.7
139    20.7
197    14.8
170     8.4
82     11.3
183    26.2
46     10.6
70     18.3
100    16.7
179    17.6
83     13.6
25     17.0
190    10.8
159    12.9
173    16.7
95     16.9
3      16.5
41     17.1
58     23.8
14     19.0
143    10.4
12      9.2
6      11.8
182     8.7
161    13.3
128    24.7
122    16.6
101    23.8
86     12.0
64     16.0
47     23.2
158     7.3
34     11.9
38     10.1
196    14.0
4      17.9
72      8.8
67     13.4
145    10.3
Name: Sales, dtype: float64
```

In [84]:
```
#Accuracy
from sklearn.metrics import mean_squared_error,r2_score

mse = mean_squared_error(pred,y_test)
r2 = r2_score(pred,y_test)

print('MSE :',mse)
print('R2 Score :',r2)
```

```
MSE : 2.3082104039612523
R2 Score : 0.9115405118554368
```

In [86]:
```
#Linear model plot
df = pd.DataFrame({'Y_Test':list(y_test),'Prediction':pred})
```

```
df
```

Out[86]:

	Y_Test	Prediction
0	16.6	18.988305
1	10.7	11.161901
2	20.7	19.843949
3	14.8	15.997466
4	8.4	8.228671
5	11.3	10.739755
6	26.2	26.569717
7	10.6	10.654065
8	18.3	19.611670
9	16.7	18.577329
10	17.6	15.392533
11	13.6	12.670957
12	17.0	20.932186
13	10.8	10.378656
14	12.9	14.136932
15	16.7	15.266679
16	16.9	17.503390
17	16.5	17.724273
18	17.1	18.484119
19	23.8	22.197195
20	19.0	20.184225
21	10.4	11.183583
22	9.2	9.063892
23	11.8	10.788955
24	8.7	8.096914
25	13.3	12.980446
26	24.7	22.589296
27	16.6	18.343802
28	23.8	26.602898
29	12.0	11.431304

	Y_Test	Prediction
30	16.0	16.448089
31	23.2	23.168266
32	7.3	8.381726
33	11.9	10.082956
34	10.1	9.334079
35	14.0	10.331561
36	17.9	16.612227
37	8.8	8.845203
38	13.4	14.131876
39	10.3	12.963956

In [ ]: