Machine Learning:

Machine Learning Procedure:-

- Dataset Reading and Studying
- Data Cleaning and Analysis
- Data Visualization or EDA (Exploratory Data Analysis)
- Encoding (converting of string columns to integer columns)
- ip/op Creation (separating input data and output/target data)
- Train Test Split (separate the training data and testing data)
- Standard Scaler Transform (standardizing all the input datas)
- Machine Learning Algorithm
- Prediction
- Accuracy

Regression Model:-

Linear Regression:-

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

In [2]:
```
# Reading the Dataset:-
adv = pd.read_csv(r"C:\Users\lab25\Downloads\archive (9)\advertising.csv")
adv
```

Out[2]:

	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 [3]:
```
# Data Cleaning:-
# Checking null values:
adv.isnull().sum()
```

Out[3]:
```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

In [5]:
```
# checking the datatypes:
adv.dtypes
```

Out[5]:
```
TV           float64
Radio        float64
Newspaper    float64
Sales        float64
dtype: object
```

In [6]:
```
# checking the unique values:
for i in adv.columns:
    print(f"{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 [7]:
```
# to check the information of the dataset:
adv.info()
```

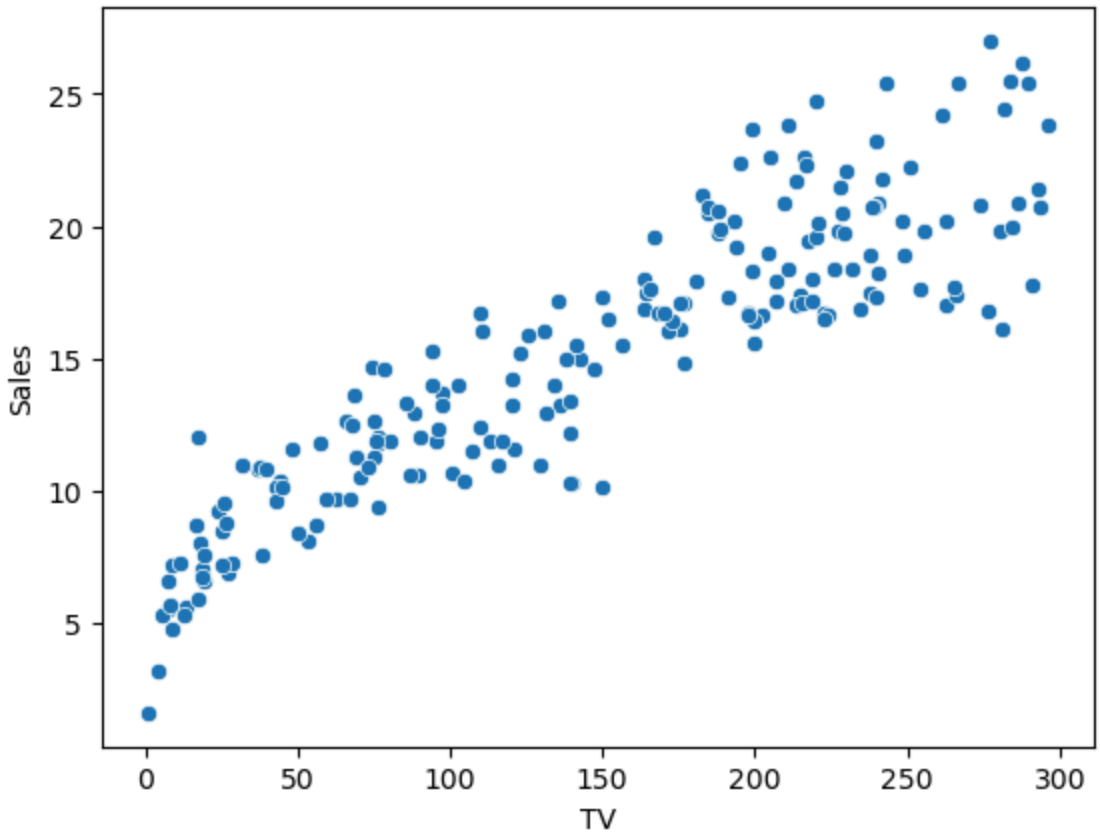
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   TV         200 non-null    float64
 1   Radio      200 non-null    float64
 2   Newspaper  200 non-null    float64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

In [8]:
```
# to check the statistical value of all columns:
adv.describe()
```

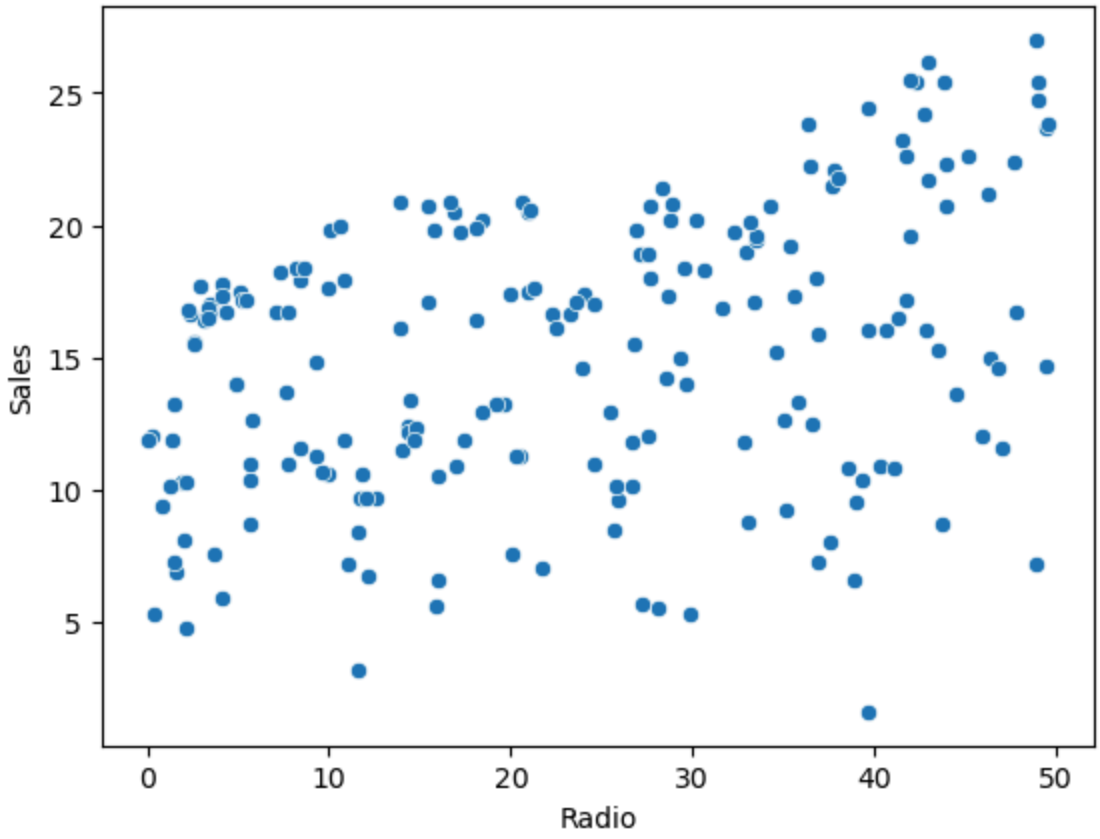
Out[8]:

	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

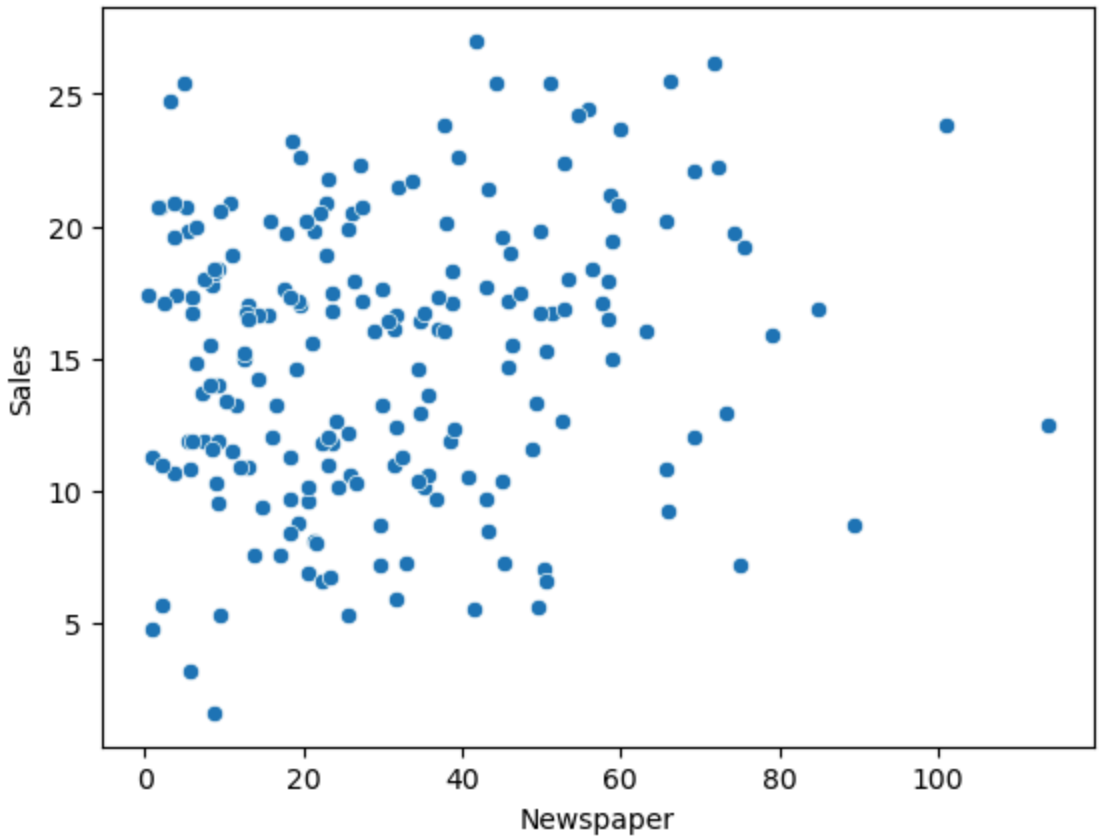
In [9]:
```
# Data Visualization:
sns.scatterplot(x=adv.TV, y=adv.Sales)
plt.show()
```



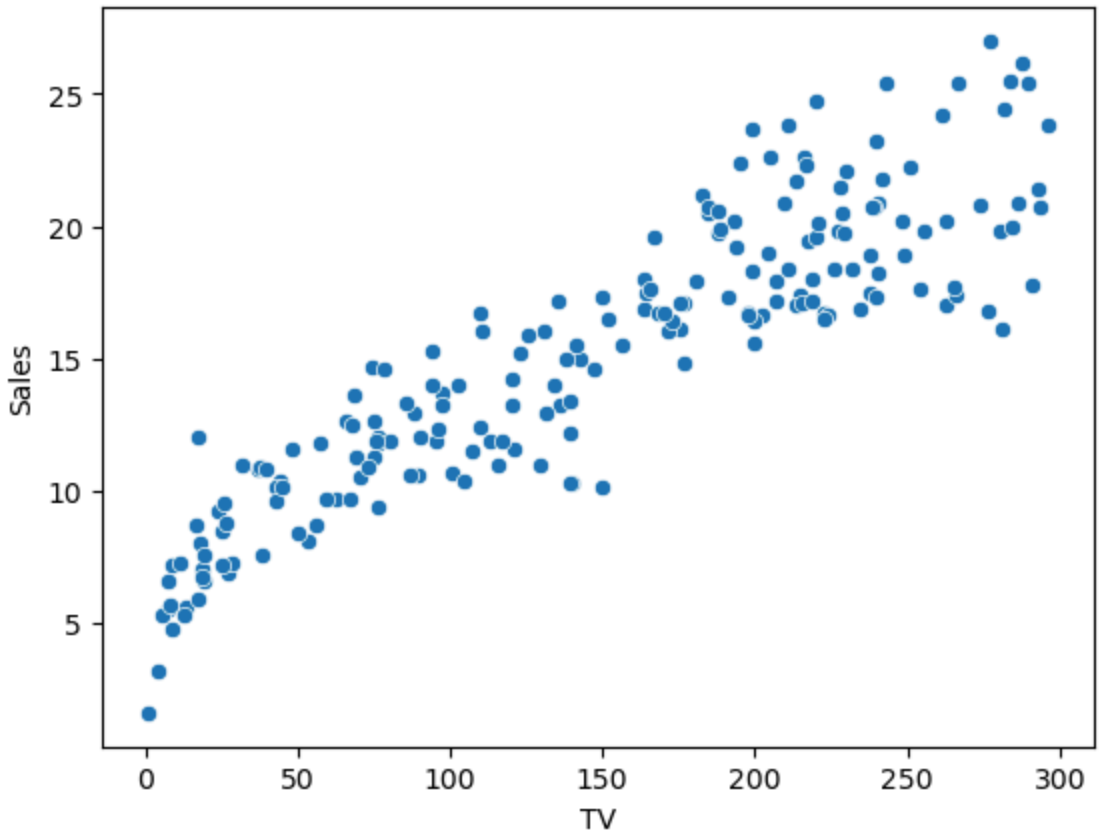
In [27]:
```
sns.scatterplot(x=adv.Radio, y=adv.Sales)
plt.show()
```

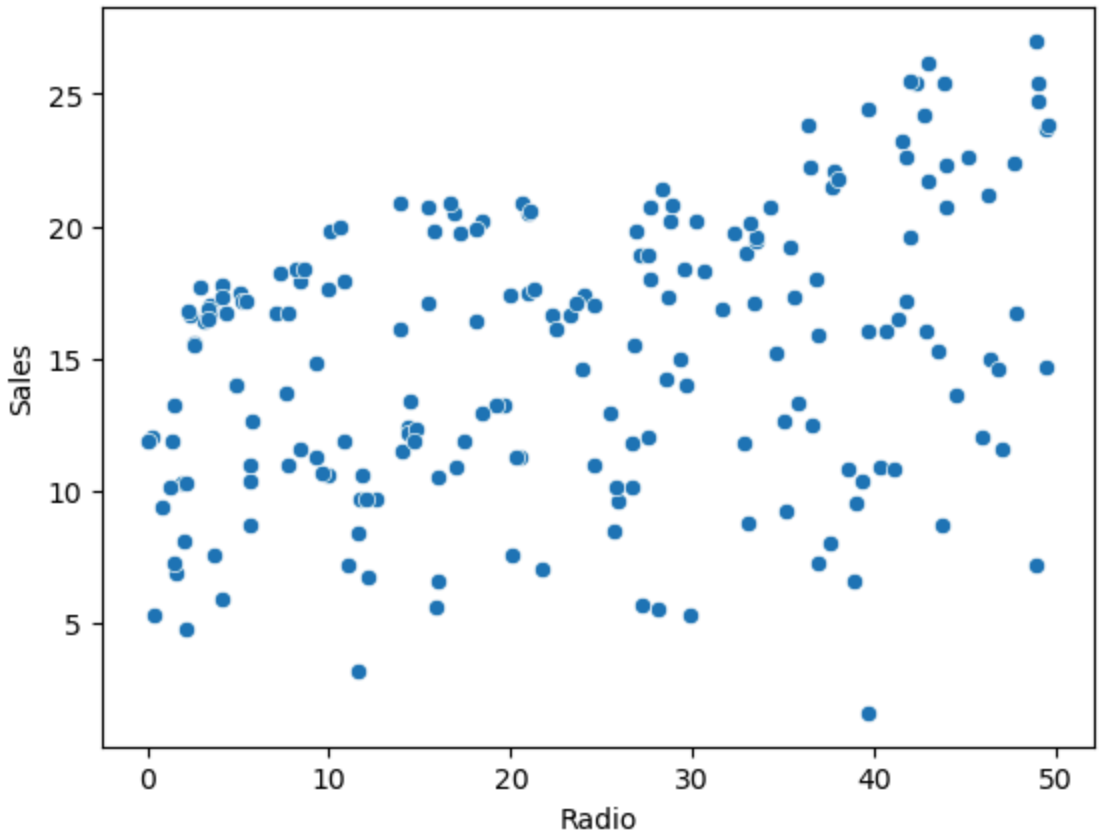


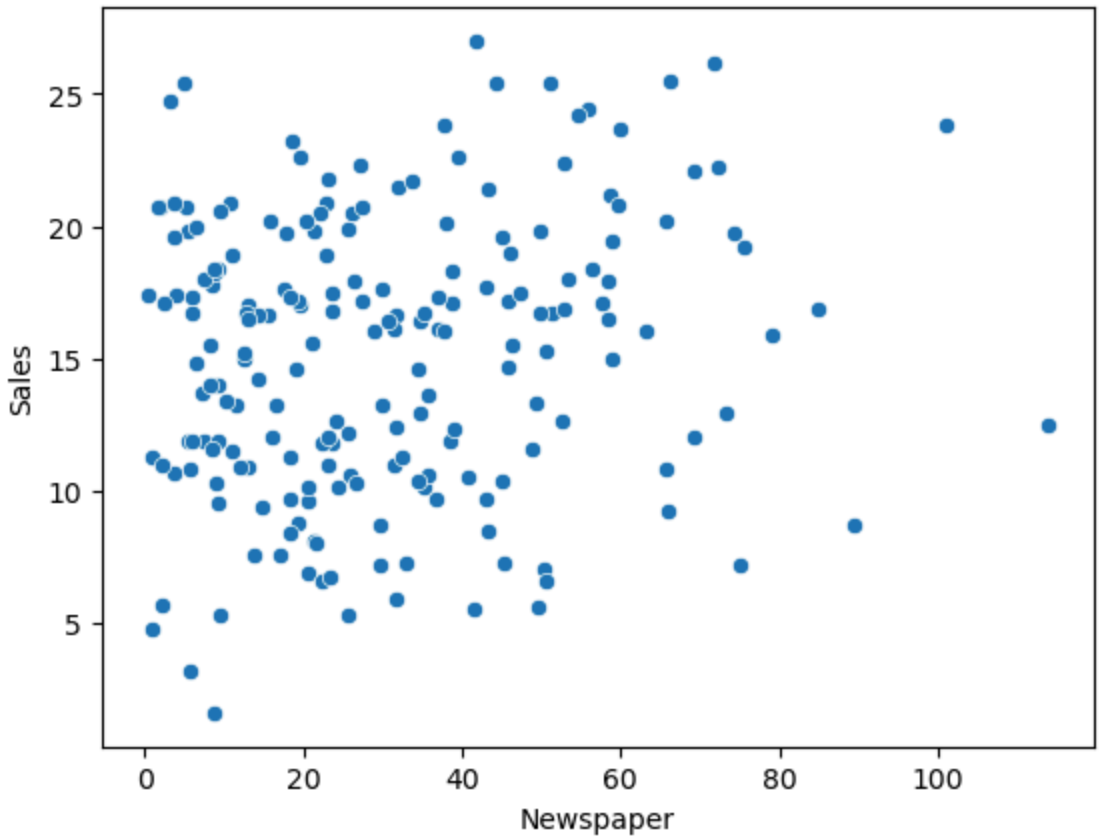
In [28]:
```
sns.scatterplot(x=adv.Newspaper, y=adv.Sales)
plt.show()
```



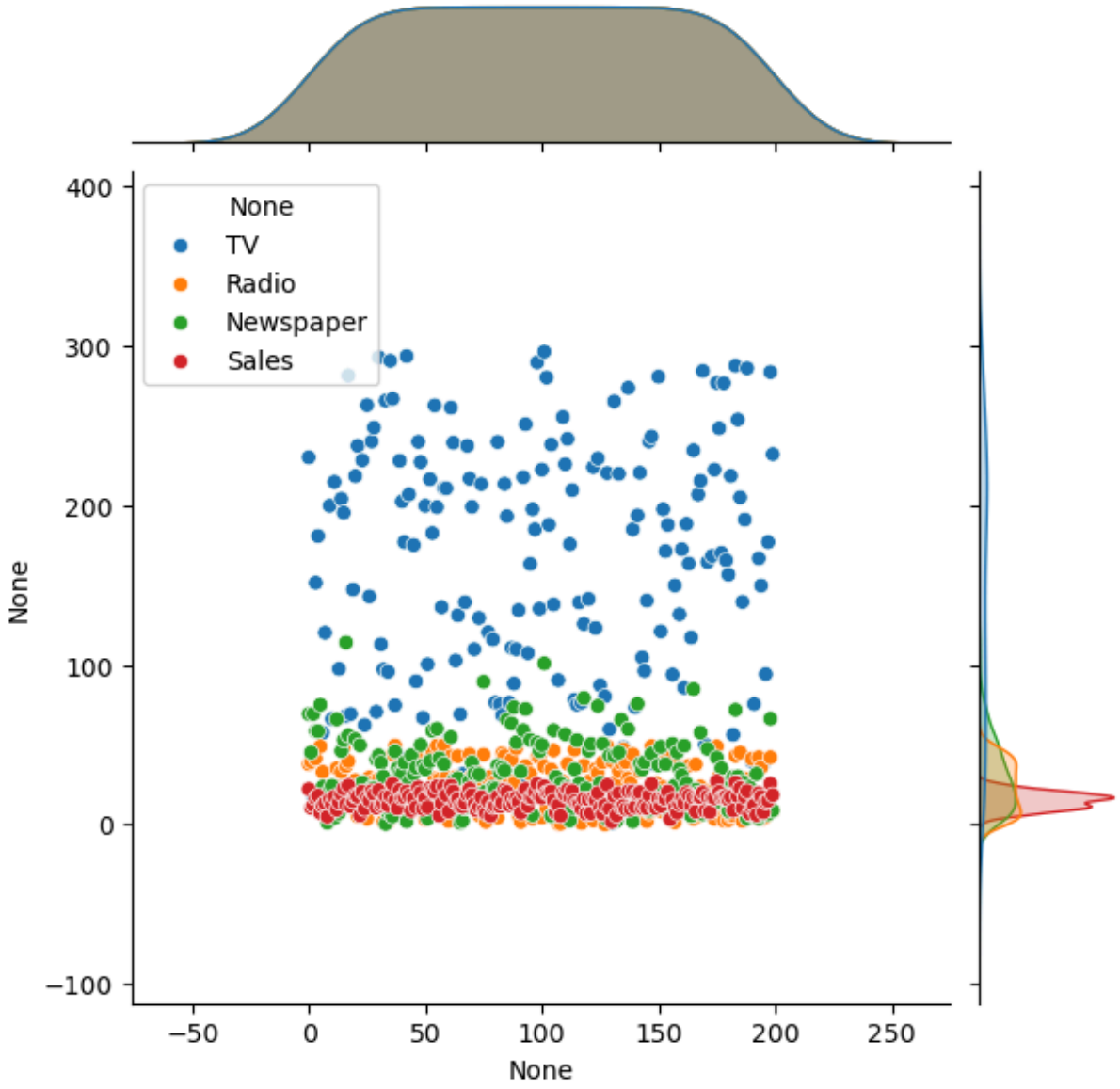
In [30]:
```
# Plotting the scatterplot for all the continuous columns at a time.
for i in ['TV','Radio','Newspaper']:
    sns.scatterplot(x=adv[i], y=adv.Sales)
    plt.show()
```


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

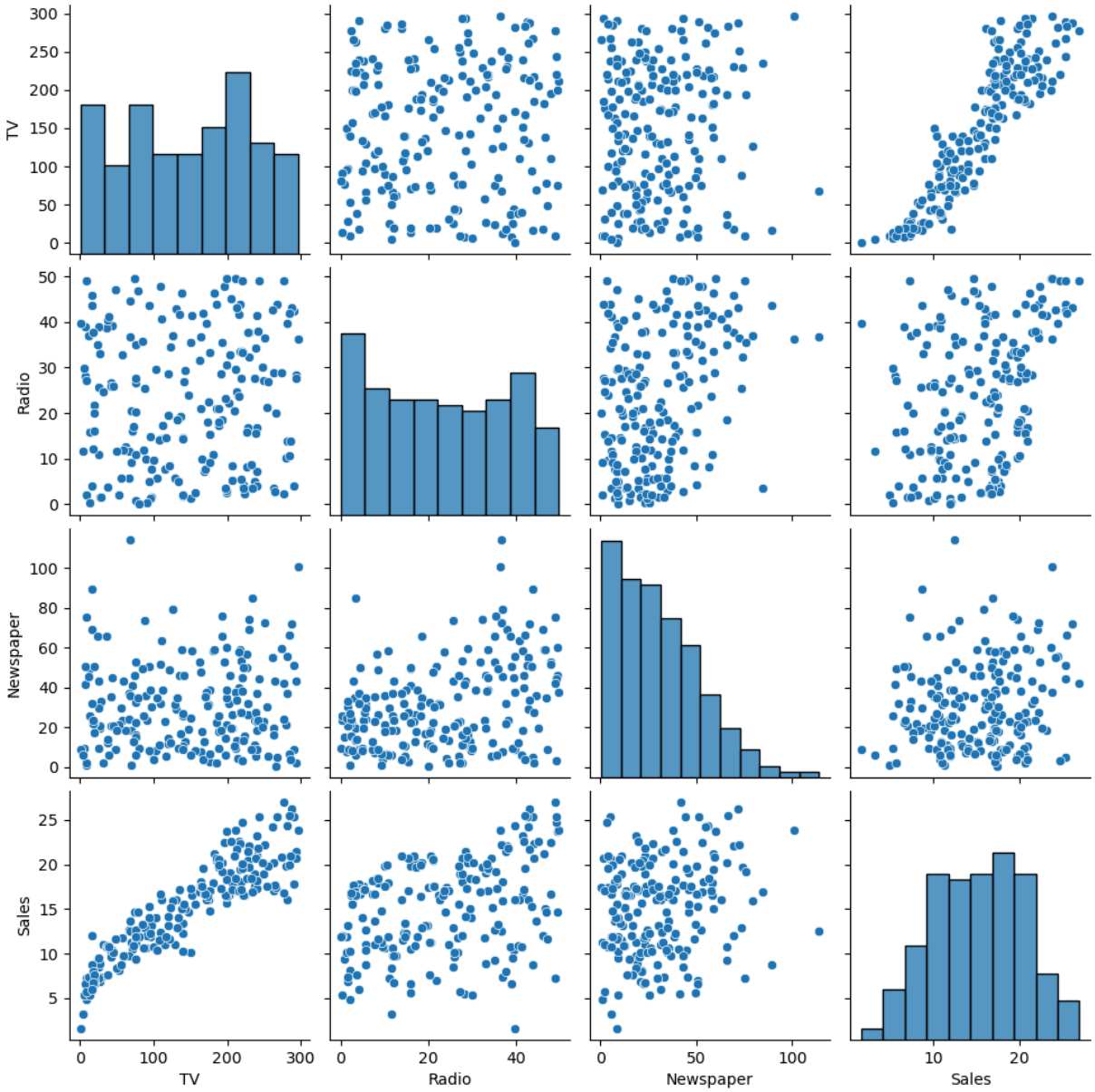

Sales
Radio
0
10
20
30
40
50
5
10
15
20
25



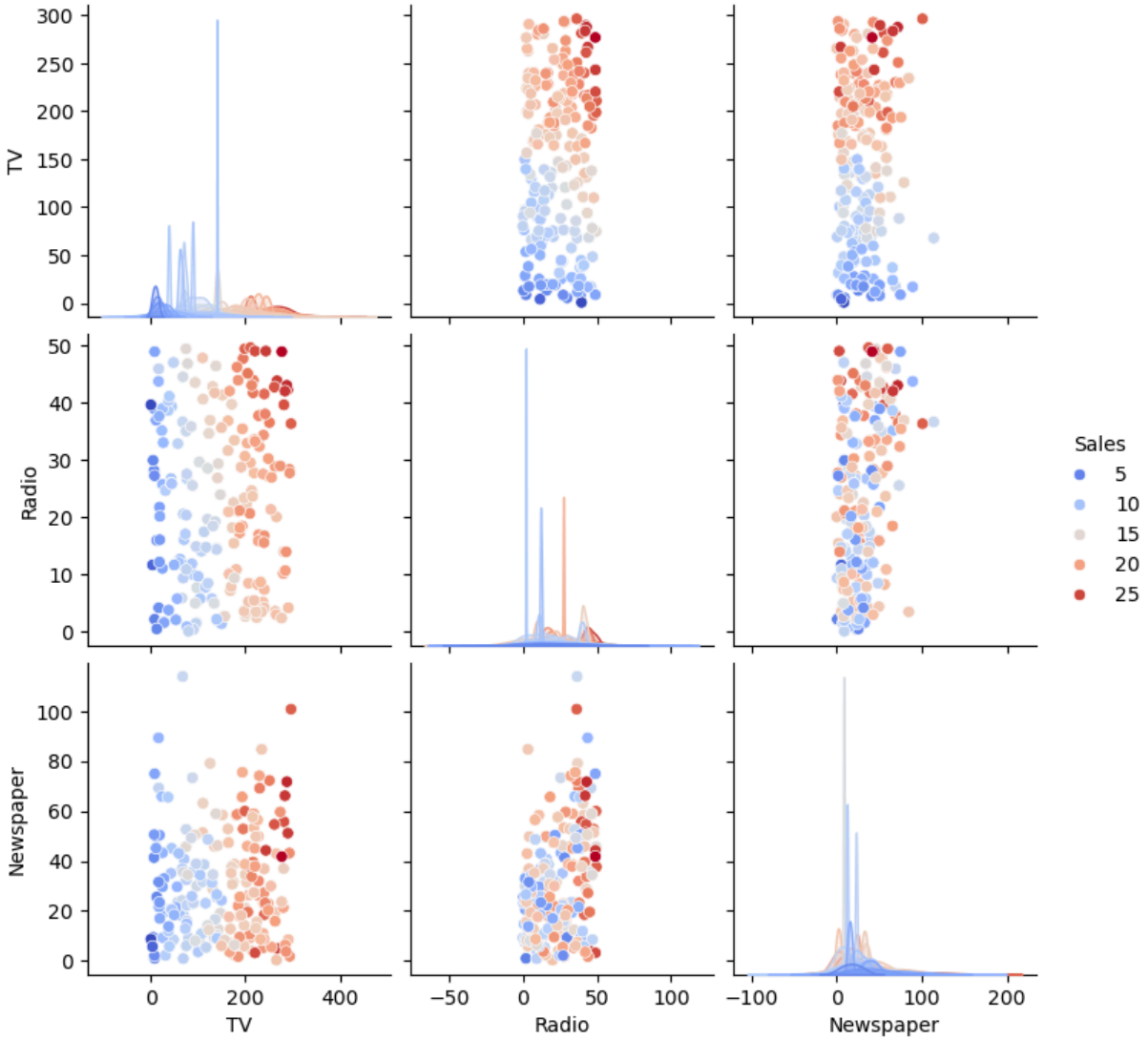
In [31]:
```
sns.jointplot(data=adv)
plt.show()
```



In [32]:
```
sns.pairplot(adv)
plt.show()
```



In [37]:
```
sns.pairplot(adv,hue='Sales',palette='coolwarm')
plt.show()
```

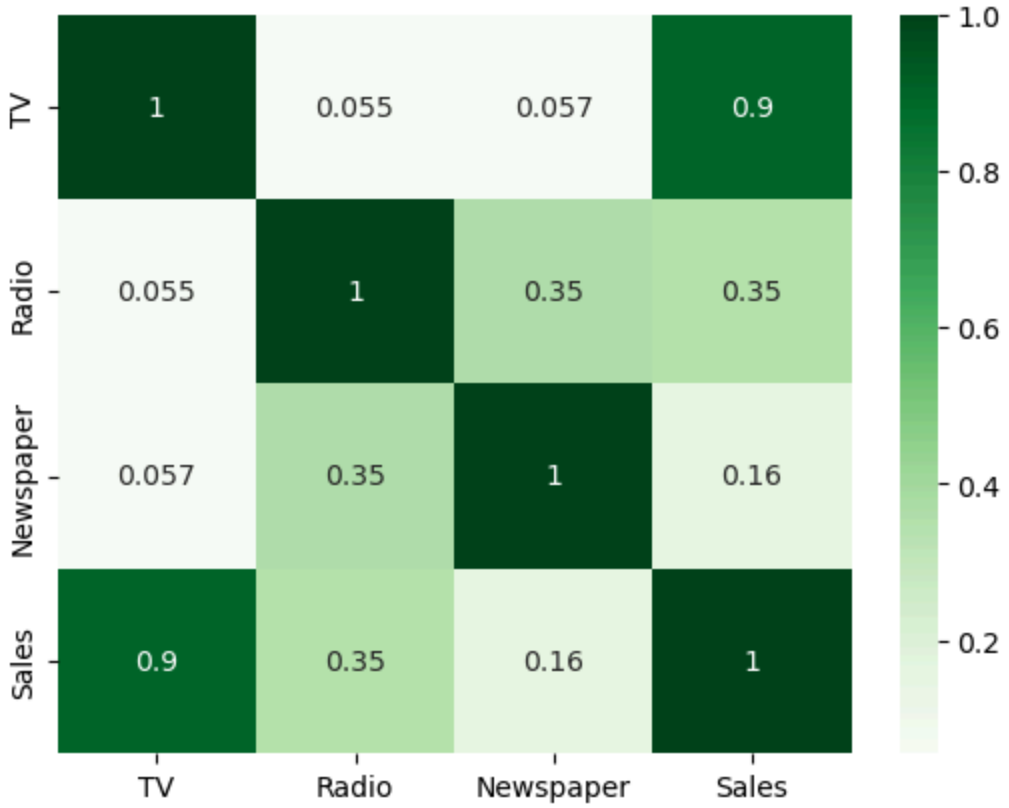


In [11]:
```
# Correlation: finds the statistical relationship between 2 varibales.
c = adv.corr()
c
```

Out[11]:

	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 [16]:
```
# plotting the correlation using heatmap:
sns.heatmap(c,annot=True,cmap='Greens')
plt.show()
```



In [38]:
```
# Encoding:- As the dataset doesnot contain any string columns therefore this step
```

In [39]:
```
# Creation of ip/op:- separating the input columns from output columns
adv
```

Out[39]:

	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 [40]:
```
# ip will store all the input columns except the target one.
ip = adv.drop('Sales',axis=1)
ip.head()
```

Out[40]:

	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 [41]:
```
op = adv.Sales
op.head()
```

Out[41]:
```
0    22.1
1    10.4
2    12.0
3    16.5
4    17.9
Name: Sales, dtype: float64
```

In [42]:
```
# Train Test Split: splitting of the 100% datas into training and testing datas.
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(ip,op,test_size=0.2)
```

In [43]:
```
xtrain.head()
```

Out[43]:

	TV	Radio	Newspaper
192	17.2	4.1	31.6
125	87.2	11.8	25.9
63	102.7	29.6	8.4
160	172.5	18.1	30.7
44	25.1	25.7	43.3

In [44]:
```
xtest.head()
```

Out[44]:

	TV	Radio	Newspaper
194	149.7	35.6	6.0
139	184.9	43.9	1.7
75	16.9	43.7	89.4
93	250.9	36.5	72.3
21	237.4	5.1	23.5

In [45]:
```
ytrain.head()
```

Out[45]:
```
192     5.9
125    10.6
63     14.0
160    16.4
44      8.5
Name: Sales, dtype: float64
```

In [46]:
```
ytest.head()
```

Out[46]:
```
194    17.3
139    20.7
75      8.7
93     22.2
21     17.5
Name: Sales, dtype: float64
```

In [ ]:

In [ ]: