sales prediction

In [37]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

load dataset

In [38]:
```
df=pd.read_csv("sales.csv")
```

In [39]:
```
df
```

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]:
```
df.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 [41]:
```
df.describe()
```

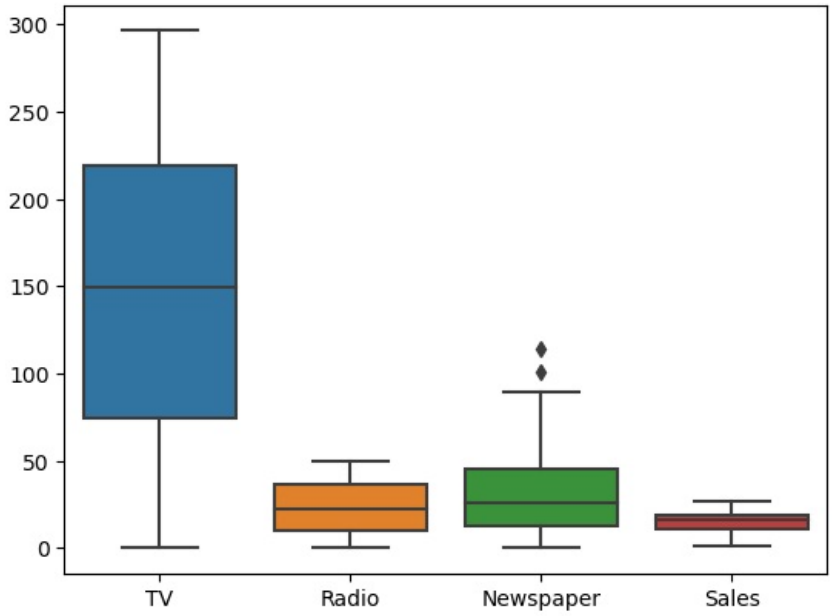
Out[41]:

	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

outlier detection

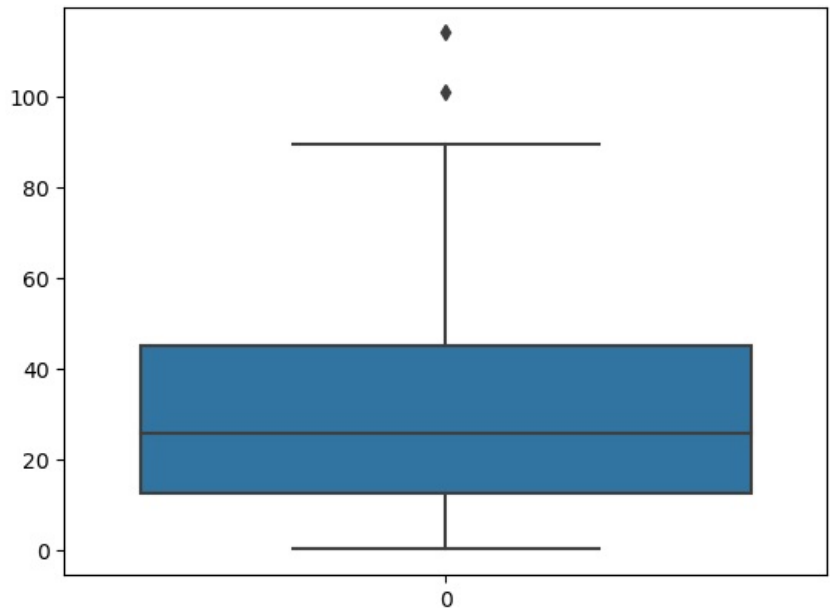
In [42]:
```
sns.boxplot(data=df)
```

Out[42]:
```
<AxesSubplot:>
```



In [43]:
```
sns.boxplot(data=df["Newspaper"])
```

Out[43]:
```
<AxesSubplot:>
```



In [44]:
```
df[df["Newspaper"]>90]
```

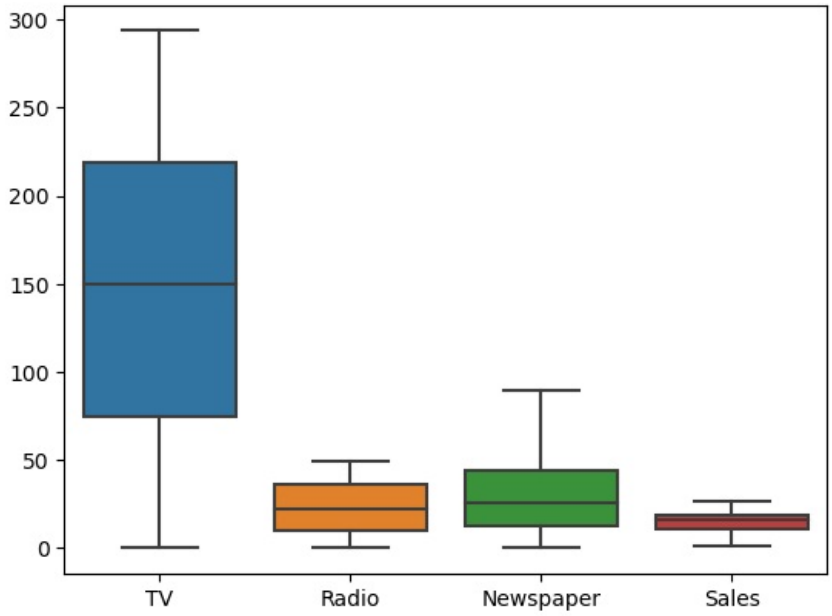
Out[44]:

	TV	Radio	Newspaper	Sales
16	67.8	36.6	114.0	12.5
101	296.4	36.3	100.9	23.8

In [45]:
```
df.drop([16,101],axis=0,inplace=True)
```

In [46]:
```
sns.boxplot(data=df)
```

Out[46]:
```
<AxesSubplot:>
```



In [47]:
```
df.head()
```

Out[47]:

	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

split into features and target

In [48]:
```
features=df.iloc[:,:-1]
target=df.iloc[:,-1]
```

In [49]:
```
features
```

Out[49]:

	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
...	...	...	...
195	38.2	3.7	13.8
196	94.2	4.9	8.1
197	177.0	9.3	6.4
198	283.6	42.0	66.2
199	232.1	8.6	8.7

198 rows × 3 columns

In [50]:
```
target
```

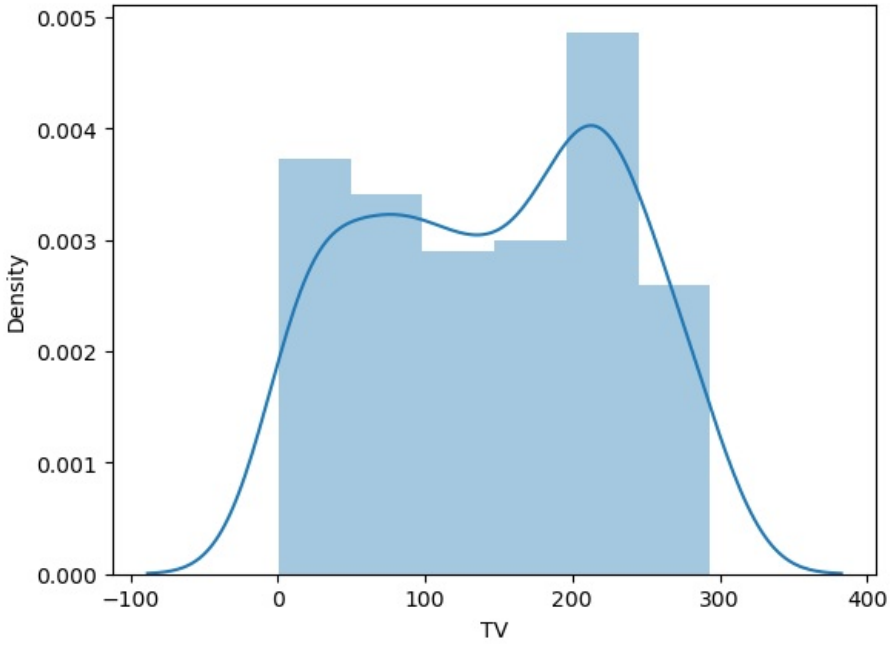
Out[50]:
```
0      22.1
1      10.4
2      12.0
3      16.5
4      17.9
       ...
195     7.6
196    14.0
197    14.8
198    25.5
199    18.4
Name: Sales, Length: 198, dtype: float64
```

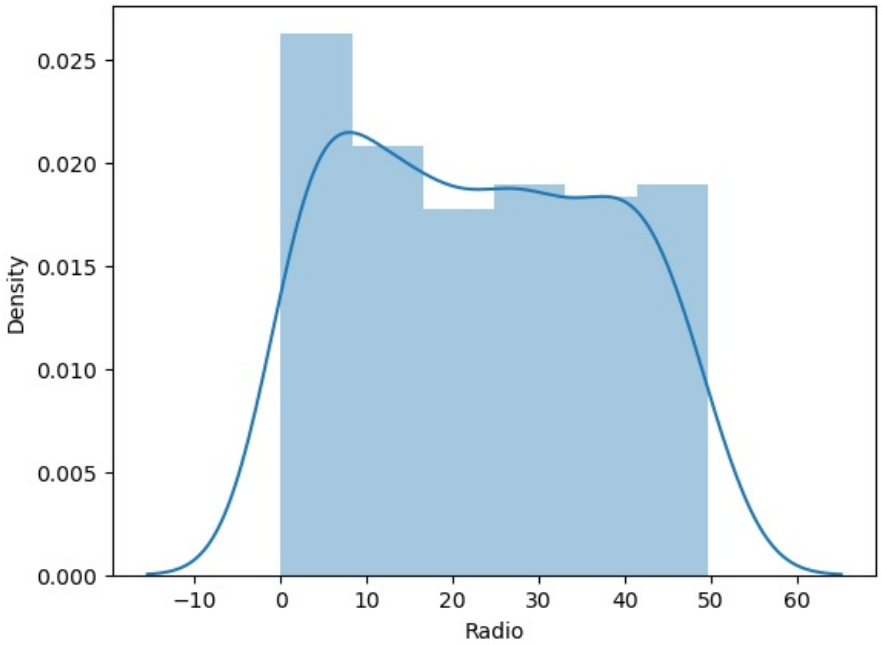
skew checking

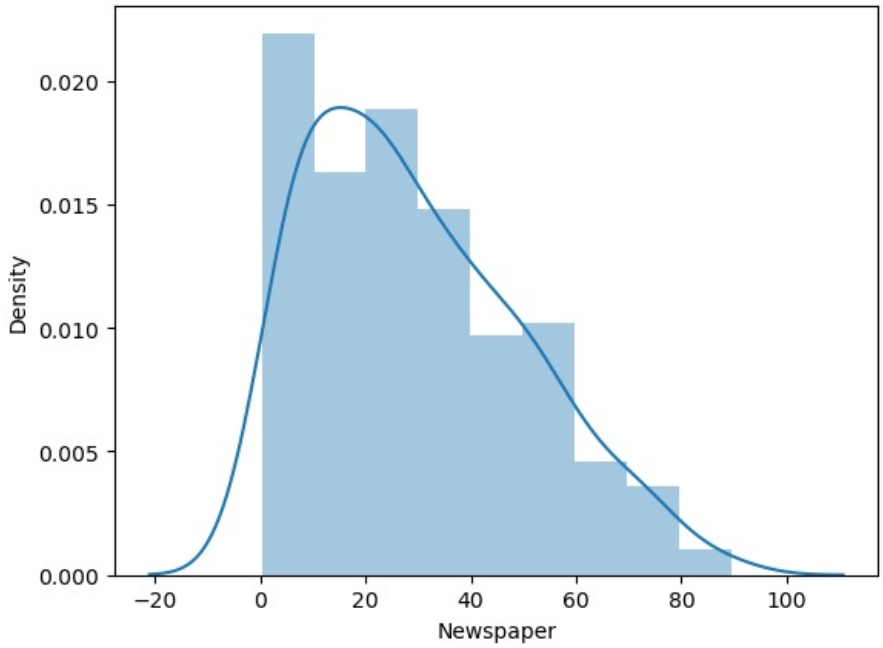
In [51]:
```
from scipy.stats import skew
```

In [52]:
```
for i in features:
    print(i)
    print(skew(features[i]))
    plt.figure()
    sns.distplot(features[i]);
```

```
TV
-0.08170703953189966
Radio
0.1139702341142612
Newspaper
0.6451760631790179
```







checking correlation

In [53]:
```
pd.concat([features,target],axis=1).corr().style.background_gradient()
```

Out[53]:

	TV	Radio	Newspaper	Sales
TV	1.000000	0.051978	0.049771	0.899974
Radio	0.051978	1.000000	0.346364	0.348566
Newspaper	0.049771	0.346364	1.000000	0.151764
Sales	0.899974	0.348566	0.151764	1.000000

train_test_split

In [54]:
```
from sklearn.model_selection import train_test_split
```

In [55]:
```
xtrain,xtest,ytrain,ytest=train_test_split(features,target,test_size=0.3)
```

LinearRegression

In [56]:
```
from sklearn.linear_model import LinearRegression
```

In [57]:
```
lr= LinearRegression()
lr.fit(xtrain,ytrain)
ypred=lr.predict(xtest)
```

In [58]:
```
ypred
```

Out[58]:
```
array([17.28845308, 21.68439349, 11.11198635, 23.19473266, 10.67281322,
        5.47257701,  8.20367585, 22.35315155, 19.73962075, 18.81344238,
       10.53191279,  9.85787084, 16.93161914, 10.41841   ,  9.60710963,
       21.51091337, 21.33856307, 15.76608346, 22.26459119,  9.44652164,
        8.26736708,  9.74372803, 10.67854986, 19.58907299, 18.71942483,
       13.84680392, 21.6703537 , 15.06499574, 25.00706911, 12.60282167,
       21.70214629, 24.073405  , 16.92956157, 22.42345908, 19.07658178,
       18.25508701, 10.07327216, 18.15430315, 14.42009541, 20.64016825,
       12.43191902, 18.54518965, 13.59582741, 12.97124911, 24.79252579,
        5.33463931, 21.32635151,  8.98758697, 19.62242874, 17.37196535,
        9.16416759, 16.6711436 , 10.47784458, 18.25116853,  9.14875188,
       13.05100992, 11.80998841, 12.16446014, 21.66777964, 16.97269797])
```

check the manual input

In [59]:
```
check=lr.predict([[197.6,23.3,14.2]])
```

In [60]:
```
check
```

In [60]:
```
check
```

Out[60]:
```
array([18.15430315])
```

accuracy

In [61]:
```
trainacc=lr.score(xtrain,ytrain)
testacc=lr.score(xtest,ytest)
```

In [62]:
```
print(f"trainacc {trainacc}\ntestacc {testacc}")
```

```
trainacc 0.9138698437359909
testacc 0.8691057695450339
```