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

In [2]:
```
df=pd.read_csv("Advertising.csv")
```

In [3]:
```
df=df.iloc[:,1:]
```

In [4]:
```
df
```

Out[4]:

	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
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	9.7
197	177.0	9.3	6.4	12.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	13.4

200 rows × 4 columns

In [5]:
```
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
```

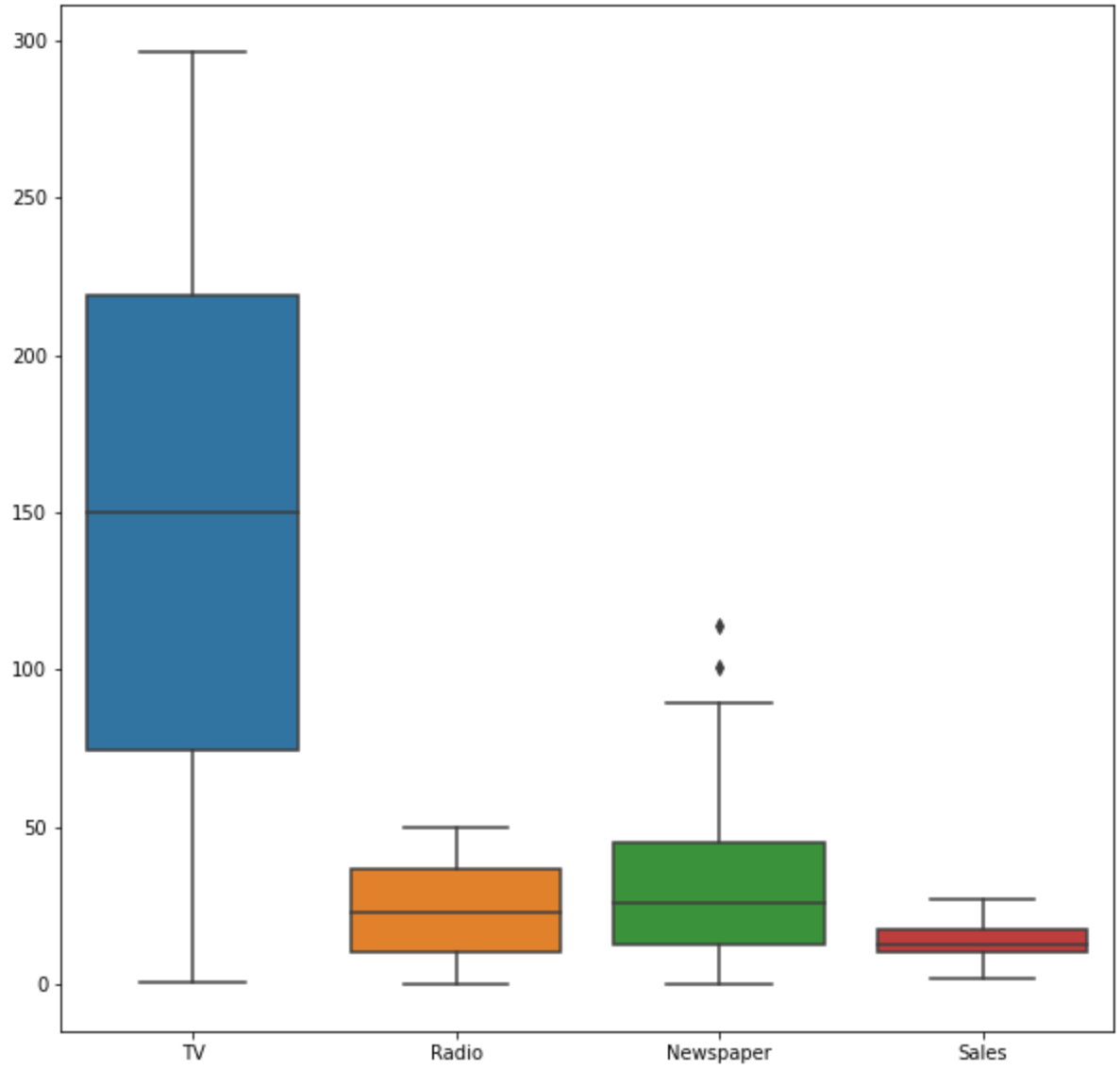
checking the missing values

In [6]:
```
df.isnull().sum()
```

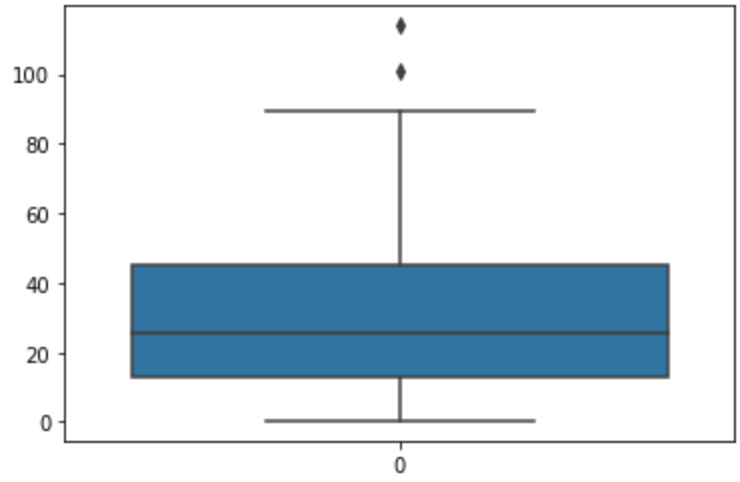
Out[6]:
```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

IdentifY the outliers

In [7]:
```
plt.figure(figsize=(10,10))
sns.boxplot(data=df);
```



In [8]:
```
sns.boxplot(data=df.Newspaper);
```



handling the outliers using quantile function

In [9]:
```
q1=np.quantile(df["Newspaper"],0.25)
q3=np.quantile(df["Newspaper"],0.75)
iqr=q3-q1
uw=q3+1.5*iqr
lw=q1-1.5*iqr
print(iqr)
print(uw,lw)
```

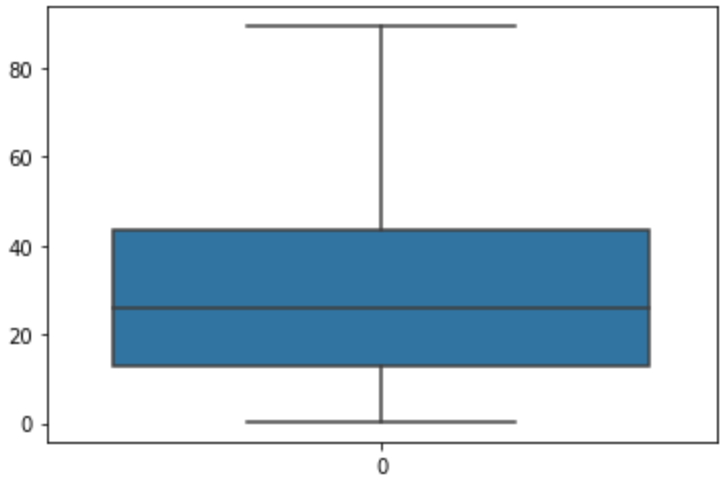
```
32.35
93.625 -35.775000000000006
```

In [10]:
```
x=df.Newspaper.mean()
x
```

Out[10]:
```
30.553999999999995
```

In [11]:
```
for i in df["Newspaper"]:
    if i>uw:
        df["Newspaper"]= df["Newspaper"].replace(i,x)
```

In [12]:
```
sns.boxplot(data=df.Newspaper);
```



Checking Skew for all the variables

In [13]:
```
from scipy.stats import skew
```

In [14]:
```
print(skew(df))
```

```
[-0.06932837  0.09346685  0.6472645   0.40450825]
```

Remove the skew using log transformation

In [15]:
```
df["Newspaper"]=np.log(df["Newspaper"])
```

In [16]:
```
print(skew(df["Newspaper"]))
```

```
-1.2268218346427877
```

Separate the input and output

In [17]:
```
x=df[['TV','Newspaper','Radio']]
```

```
y=df["Sales"]
```

In [18]:
```
pd.concat([x,y],axis=1).corr().style.background_gradient()
```

Out[18]:

	TV	Newspaper	Radio	Sales
TV	1.000000	0.019994	0.054809	0.782224
Newspaper	0.019994	1.000000	0.228636	0.144252
Radio	0.054809	0.228636	1.000000	0.576223
Sales	0.782224	0.144252	0.576223	1.000000

In [ ]:
```
```

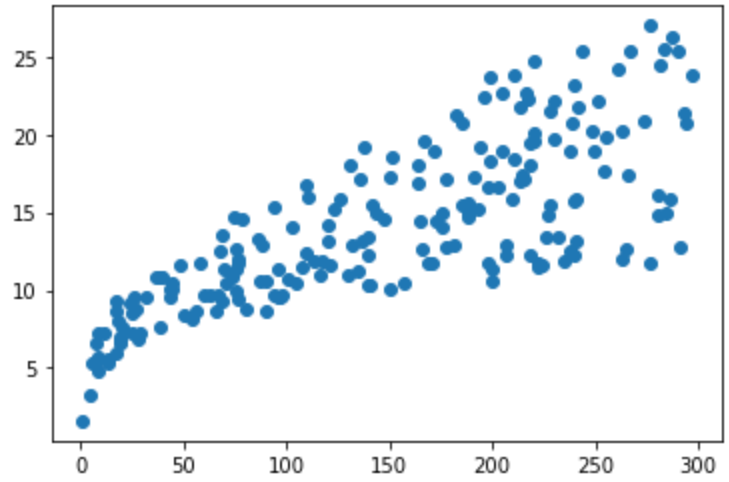
In [32]:
```
from sklearn.model_selection import train_test_split
Xtrain, Xtest, ytrain, ytest = train_test_split(x,y, train_size = 0.7, test_size = 0.3,
```

In [33]:
```
from sklearn.linear_model import LinearRegression
```

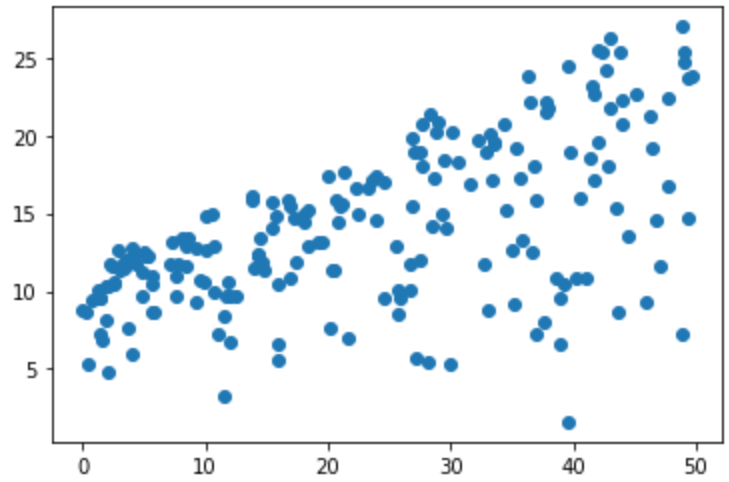
In [34]:
```
reg=LinearRegression()
reg.fit(x,y)
```

Out[34]:
```
LinearRegression()
```

In [22]:
```
plt.scatter(x[["TV"]],y);
```

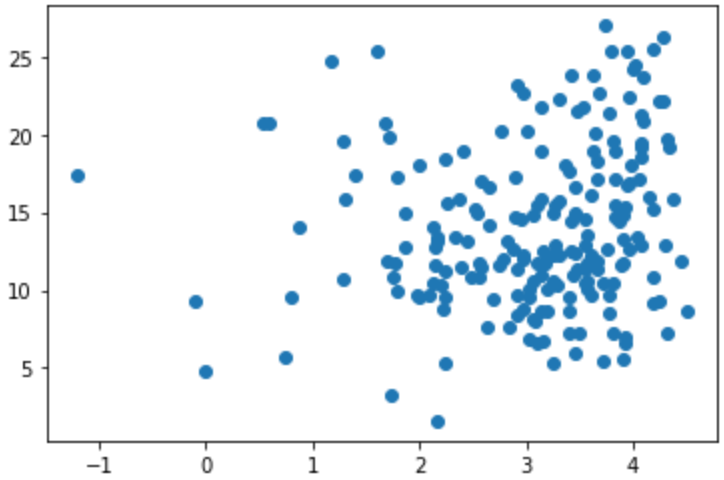


In [23]:
```
plt.scatter(x[["Radio"]],y);
```



In [ ]:
```
```

In [24]:
```
plt.scatter(x[["Newspaper"]],y);
```



R2SCORE of training set

In [35]:
```
accu=reg.score(Xtrain,ytrain)
```

In [36]:
```
accu
```

Out[36]:
```
0.883756310048439
```

R2SCORE FOR TESTING SET

In [37]:
```
pd=reg.predict(Xtest)
```

In [38]:
```
pd
```

Out[38]:
```
array([21.89753761, 16.39450002,  7.53399643, 17.64757751, 18.55208854,
       23.8261634 , 16.36431993, 13.2707245 ,  9.18927934, 17.25204597,
       14.40958443,  9.93690434, 17.1896182 , 16.95154034, 14.8409665 ,
       15.65328565, 12.42797216, 16.96702573, 11.3825072 , 18.07070444,
        9.25733641, 12.91042311,  8.92782427, 10.44597022, 11.38475422,
       14.95395829,  9.93046877, 19.39964785, 18.36736151, 17.05253258,
       21.68275279, 14.35547136, 16.40575829, 12.20352503, 19.95756129,
       15.33133385, 13.72809711,  9.93866094, 21.09090664,  7.58975405,
        3.61653947,  7.13050118,  6.06191001, 18.46909496,  8.56146184,
       14.14898573, 15.30263199, 20.44779386, 20.76298739, 19.53762429,
       24.15132078, 15.03003433,  6.8229379 , 19.98763834, 18.66639899,
       12.37799077, 13.95363998,  6.15341032, 15.13025675,  9.75472028])
```

In [41]:
```
from sklearn.metrics import mean_squared_error,r2_score
mse=mean_squared_error(ytest,pd)
rmse=np.sqrt(mse)
print(rmse)
```

```
1.312347819976752
```

In [45]:
```
r2_score(ytest,pd)
```

Out[45]:
```
0.9307682693882993
```

In [39]:
```
accu1=reg.score(Xtest,ytest)
```

In [40]:
```
accu1
```

Out[40]:
```
0.9307682693882993
```

In [ ]:
```
```