In [45]:

```
import numpy as np
import pandas as pd
```

In [43]:

```
pip install autoviz --upgrade
```

```
Requirement already satisfied: autoviz in c:\users\91755\appdata\local
\programs\python\python39\lib\site-packages (0.1.58)
Requirement already satisfied: pandas in c:\users\91755\appdata\local\p
rograms\python\python39\lib\site-packages (from autoviz) (1.5.3)
Requirement already satisfied: pyamg in c:\users\91755\appdata\local\pr
ograms\python\python39\lib\site-packages (from autoviz) (4.2.3)
Requirement already satisfied: jupyter in c:\users\91755\appdata\local
\programs\python\python39\lib\site-packages (from autoviz) (1.0.0)
Requirement already satisfied: nltk in c:\users\91755\appdata\local\pro
grams\python\python39\lib\site-packages (from autoviz) (3.8.1)
Requirement already satisfied: hvplot>=0.7.3 in c:\users\91755\appdata
\local\programs\python\python39\lib\site-packages (from autoviz) (0.8.
3)
Requirement already satisfied: panel~=0.12.6 in c:\users\91755\appdata
\local\programs\python\python39\lib\site-packages (from autoviz) (0.12.
7)
Requirement already satisfied: holoviews>=1.14.6 in c:\users\91755\appd
ata\local\programs\python\python39\lib\site-packages (from autoviz) (1.
14.9)
Requirement already satisfied: bokeh>=2.4.2 in c:\users\91755\appdata\l
```

In [10]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegressio
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from autoviz.classify_method import data_cleaning_suggestions,data_suggestions
from xgboost import XGBRegressor
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn import metrics
%matplotlib inline
```

In [118]:

```
df=pd.read_csv("Advertising.csv")
```

In [119]:

```
df.head()
```

Out[119]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9

In [120]:

```
df.sample(5)
```

Out[120]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
199	200	232.1	8.6	8.7	13.4
80	81	76.4	26.7	22.3	11.8
104	105	238.2	34.3	5.3	20.7
7	8	120.2	19.6	11.6	13.2
95	96	163.3	31.6	52.9	16.9

In [121]:

```
df.shape
```

Out[121]:

```
(200, 5)
```

In [122]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  200 non-null    int64  
 1   TV          200 non-null    float64
 2   Radio       200 non-null    float64
 3   Newspaper   200 non-null    float64
 4   Sales       200 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 7.9 KB
```

In [123]:

```
df.describe()
```

Out[123]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000	200.000000
mean	100.500000	147.042500	23.264000	30.554000	14.022500
std	57.879185	85.854236	14.846809	21.778621	5.217457
min	1.000000	0.700000	0.000000	0.300000	1.600000
25%	50.750000	74.375000	9.975000	12.750000	10.375000
50%	100.500000	149.750000	22.900000	25.750000	12.900000
75%	150.250000	218.825000	36.525000	45.100000	17.400000
max	200.000000	296.400000	49.600000	114.000000	27.000000

In [124]:

```
df.isnull()
```

Out[124]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
195	False	False	False	False	False
196	False	False	False	False	False
197	False	False	False	False	False
198	False	False	False	False	False
199	False	False	False	False	False

200 rows × 5 columns

In [125]:

```
df.tail()
```

Out[125]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
195	196	38.2	3.7	13.8	7.6
196	197	94.2	4.9	8.1	9.7
197	198	177.0	9.3	6.4	12.8
198	199	283.6	42.0	66.2	25.5
199	200	232.1	8.6	8.7	13.4

In [126]:

```
df=df.drop(columns=['Unnamed: 0'])
```

In [127]:

```
df.head()
```

Out[127]:

	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

In [128]:

```
pip install plotly
```

```
Requirement already satisfied: plotly in c:\users\91755\appdata\local\prog
rams\python\python39\lib\site-packages (5.14.1)Note: you may need to resta
rt the kernel to use updated packages.

Requirement already satisfied: packaging in c:\users\91755\appdata\local\p
rograms\python\python39\lib\site-packages (from plotly) (23.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\91755\appdata\l
ocal\programs\python\python39\lib\site-packages (from plotly) (8.2.2)

WARNING: You are using pip version 21.2.3; however, version 23.0.1 is avai
lable.
You should consider upgrading via the 'C:\Users\91755\AppData\Local\Progra
ms\Python\Python39\python.exe -m pip install --upgrade pip' command.
```

In [129]:

```
import plotly.express as px
```

In [130]:

```
fix=px.scatter(df,x="TV",y="Sales",color="Newspaper",hover_name="Radio",size="Sales",tit
fix.show()
```

In [131]:

```
import plotly.graph_objects as go
```

In [132]:

```
fig=go.Figure()
fig.add_trace(go.Bar(x=df.index,y=df['TV'],name='TV'))
fig.add_trace(go.Bar(x=df.index,y=df['Radio'],name='Radio'))
fig.add_trace(go.Bar(x=df.index,y=df['Newspaper'],name='Newspaper'))
fig.update_layout(barmode='stack',xaxis_title='observations',yaxis_title='Sales')
fig.show()
```

In [133]:

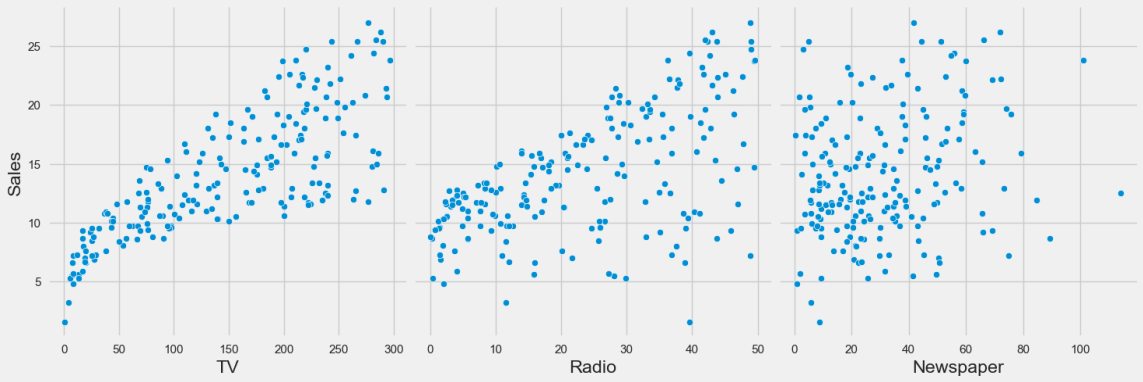
```
import seaborn as sns
```

In [134]:

```
sns.pairplot(data=df,x_vars=['TV','Radio','Newspaper'],y_vars=['Sales'],height=5)
```

Out[134]:

```
<seaborn.axisgrid.PairGrid at 0x23941262880>
```



In [135]:

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

Out[135]:

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

In [136]:

```
df.duplicated().sum()
```

Out[136]:

```
0
```

In [137]:

```
df.corr()
```

Out[137]:

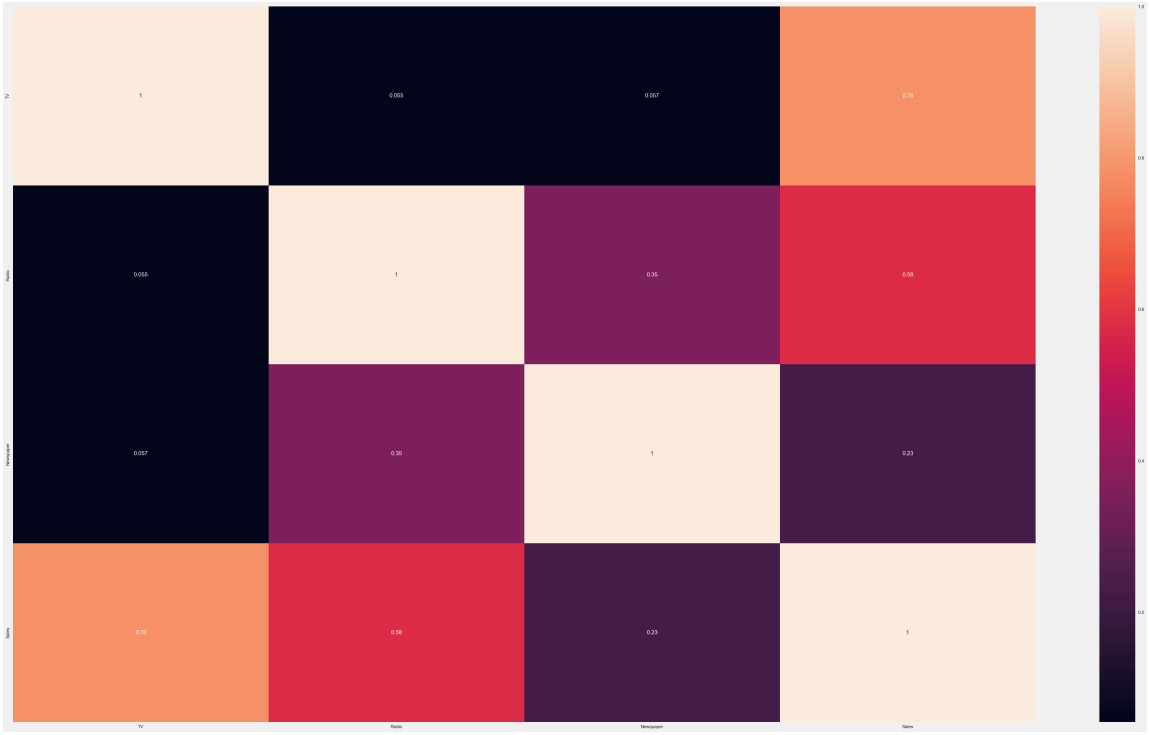
	TV	Radio	Newspaper	Sales
TV	1.000000	0.054809	0.056648	0.782224
Radio	0.054809	1.000000	0.354104	0.576223
Newspaper	0.056648	0.354104	1.000000	0.228299
Sales	0.782224	0.576223	0.228299	1.000000

In [138]:

```
plt.figure(figsize=(50,30))
sns.heatmap(df.corr(),annot=True)
```

Out[138]:

<Axes: >



In [139]:

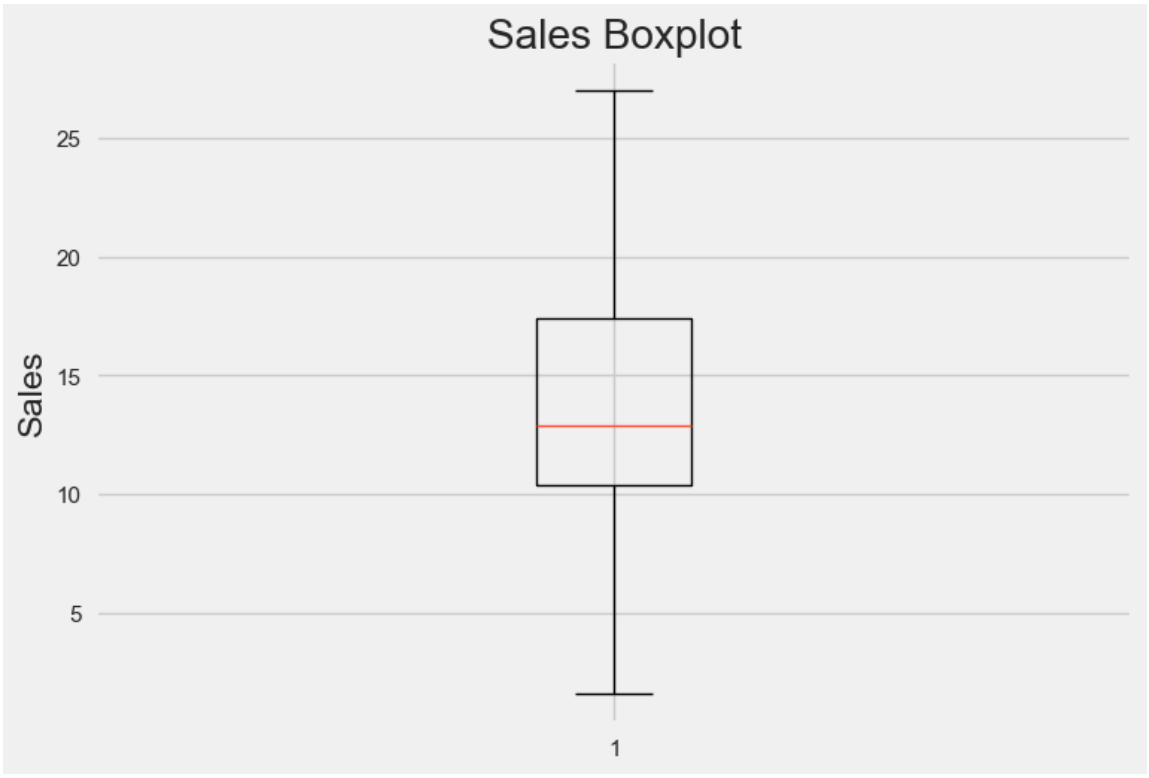
```
fig= px.imshow(df.corr(), text_auto=True, color_continuous_scale='Viridis')
```

In [140]:

```
fig.show()
```

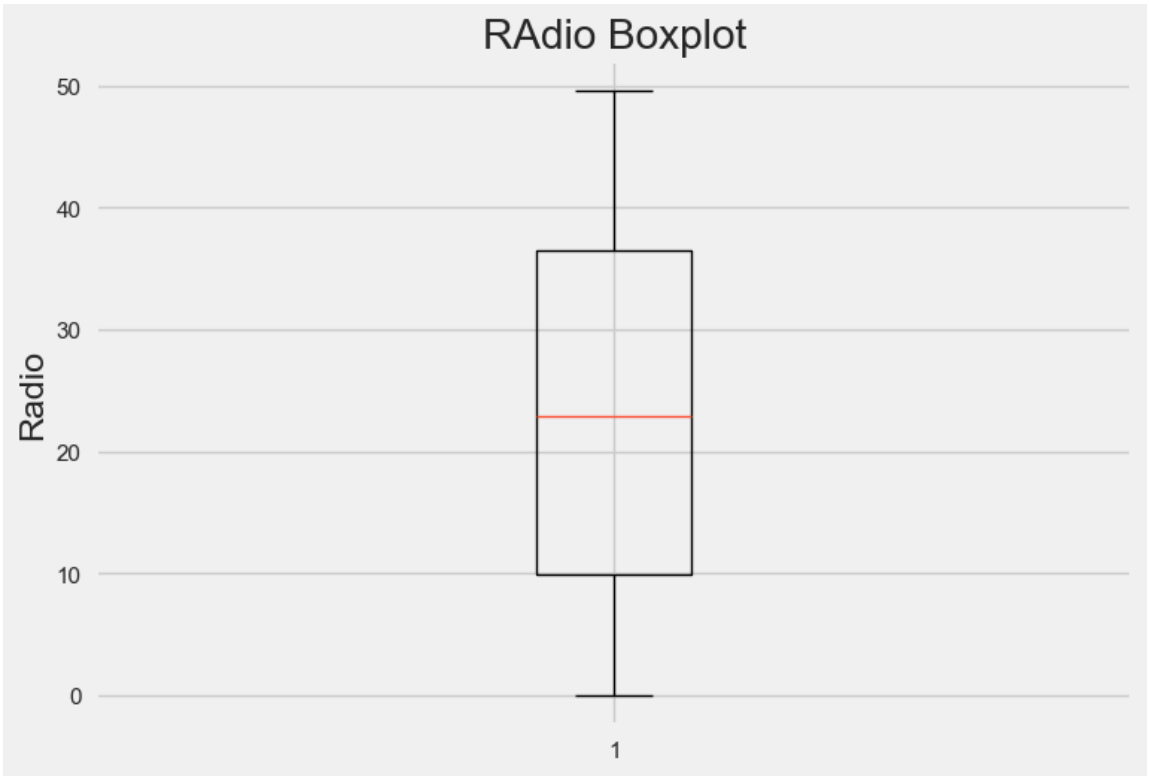
In [141]:

```
plt.boxplot(df['Sales'])
plt.title("Sales Boxplot")
plt.ylabel("Sales")
plt.show()
```



In [142]:

```
plt.boxplot(df['Radio'])
plt.title("RAdio Boxplot")
plt.ylabel("Radio")
plt.show()
```



In [143]:

```
def find_outliers(dataframe):
    outlier_percentages=[]
    for column in dataframe.columns:
        if dataframe[column].dtype != object :
            q1,q3=dataframe[column].quantile([0.25,0.75])
            iqr=q3 - q1
            upper_bound=q3 + 1.5*iqr
            lower_bound=q3 - 1.5*iqr
            num_outliers=len(dataframe[(dataframe[column]<lower_bound)|(dataframe[column
            outlier_percentage=round(num_outliers/len(dataframe[column])*100,2)
            outlier_percentages.append((column,outlier_percentage))
            outlier_df=pd.DataFrame(outlier_percentages,columns=['Column','Outlier Perce
            outlier_df=outlier_df.sort_values(by='Outlier Percentage',ascending=False).r
            return outlier_df
```

In [144]:

```
outlier_df=find_outliers(df)
outlier_df
```

Out[144]:

	Column	Outlier Percentage
0	TV	0.5

In [145]:

```
def remove_outliers(dataframe):
    df=dataframe.copy()
    for column in dataframe.columns:
         if df[column].dtype != object :
            q1,q3=df[column].quantile([0.25,0.75])
            iqr=q3 - q1
            upper_bound=q3 + 1.5*iqr
            lower_bound=q3 - 1.5*iqr
            df=df[(df[column]>=lower_bound)&(df[column]<=upper_bound)]
    return df
```

In [146]:

```
df_without_outliers=remove_outliers(df)
```

In [147]:

```
outlier_df=find_outliers(df_without_outliers)
outlier_df
```

Out[147]:

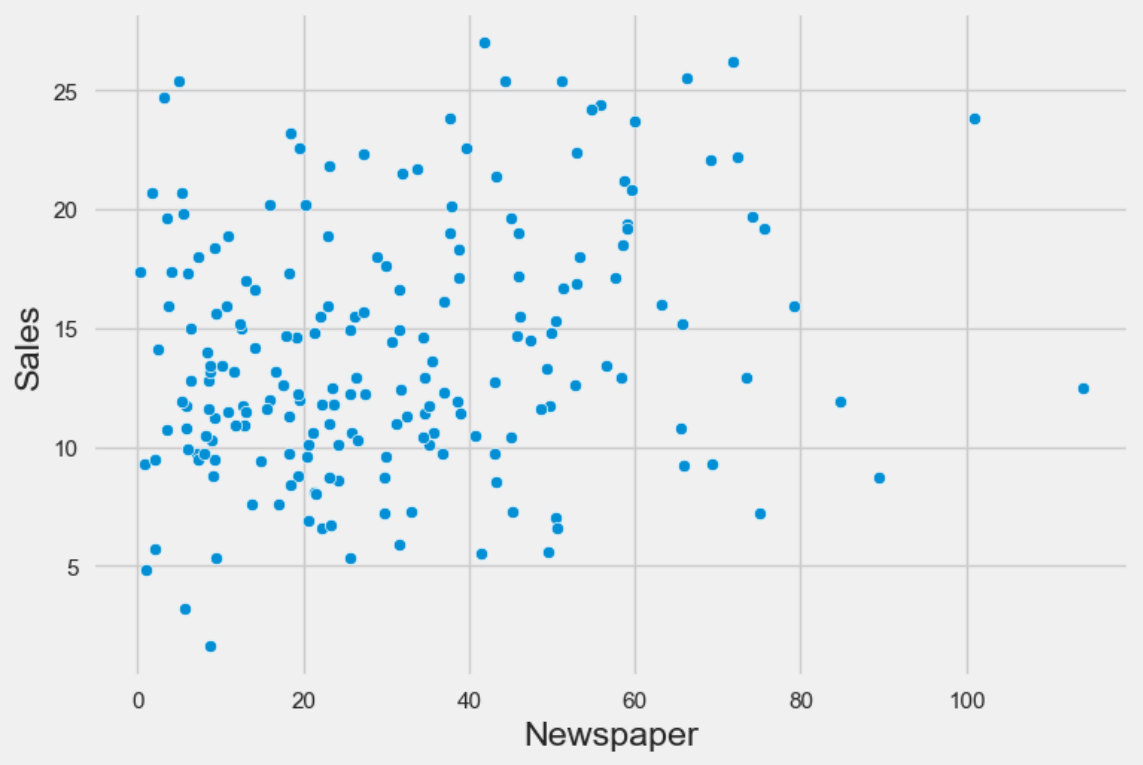
	Column	Outlier Percentage
0	TV	3.78

In [148]:

```
sns.scatterplot(x='Newspaper',y='Sales',data=df)
```

Out[148]:

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

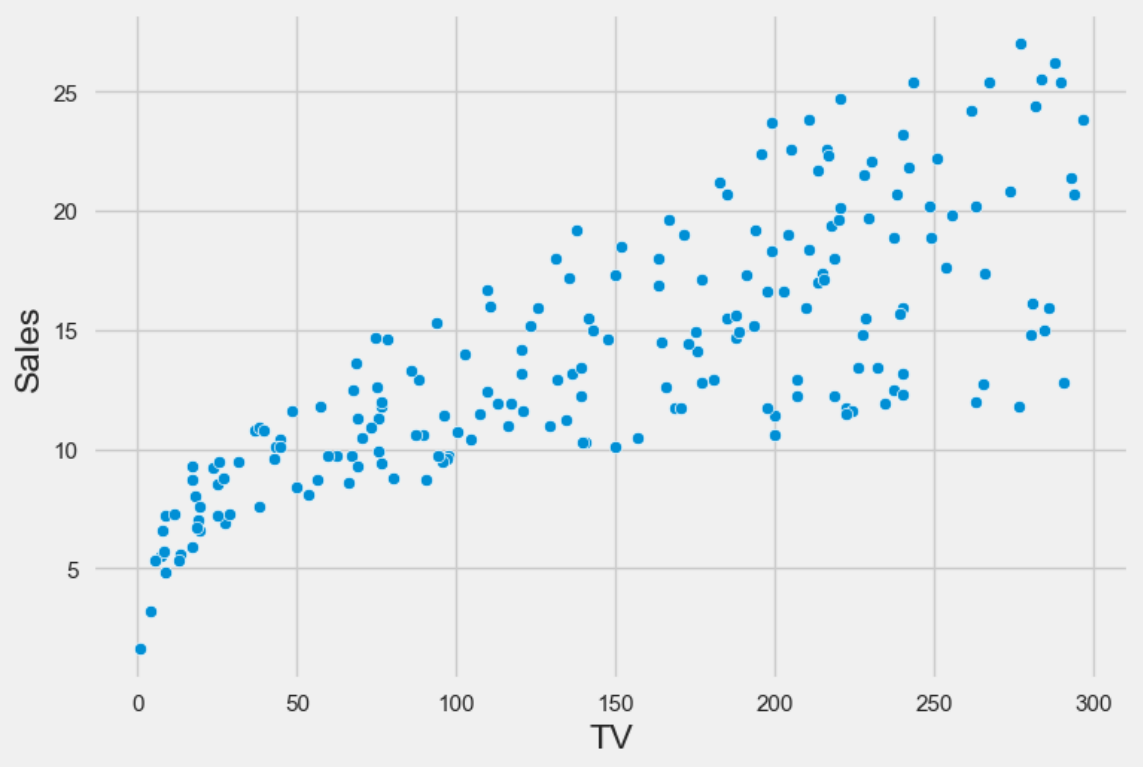


In [149]:

```
sns.scatterplot(x='TV',y='Sales',data=df)
```

Out[149]:

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

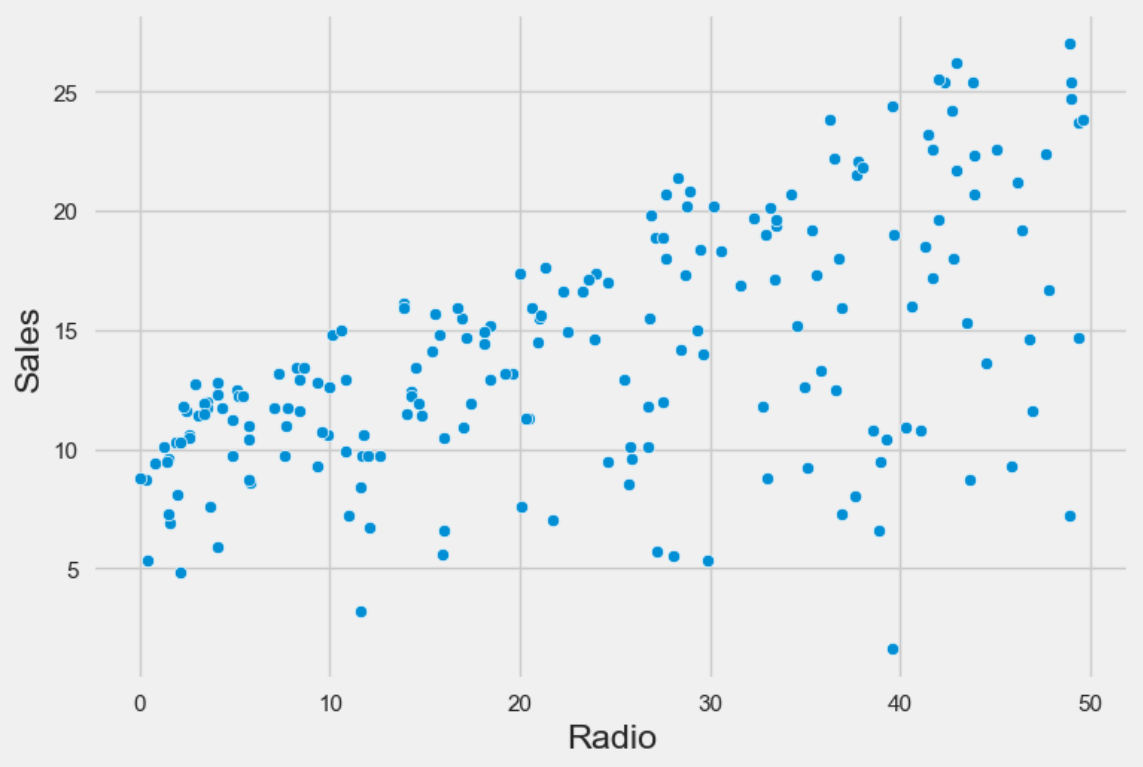


In [150]:

```
sns.scatterplot(x='Radio',y='Sales',data=df)
```

Out[150]:

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



In [151]:

```
x=df.iloc[:,0:-1]
x
```

Out[151]:

	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

200 rows × 3 columns

In [152]:

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

Out[152]:

```
0      22.1
1      10.4
2       9.3
3      18.5
4      12.9
       ...
195     7.6
196     9.7
197    12.8
198    25.5
199    13.4
Name: Sales, Length: 200, dtype: float64
```

In [153]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

In [154]:

```
sc=StandardScaler()
```

In [155]:

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

In [156]:

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

In [157]:

```
lr=LinearRegression()
```

In [158]:

```
lr.fit(x_train_scaled,y_train)
```

Out[158]:

▾ LinearRegression

LinearRegression()

In [159]:

```
y_pred=lr.predict(x_test_scaled)
```

r2_score(y_test,y_pred)

In [160]:

```
r2_score(y_test,y_pred)
```

Out[160]:

```
0.8863195562008254
```

In [ ]:

```

```