In [11]:

```
#Importing the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [9]:

```
#reading the data set
dataset=pd.read_csv("advertising.csv")
```

In [10]:

```
dataset.head()
```

Out[10]:

	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 [12]:

```
#checking for missing values
dataset.isna().sum()
```

Out[12]:

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

In [14]:

```
#checking for duplicate rows
dataset.duplicated().any()
```

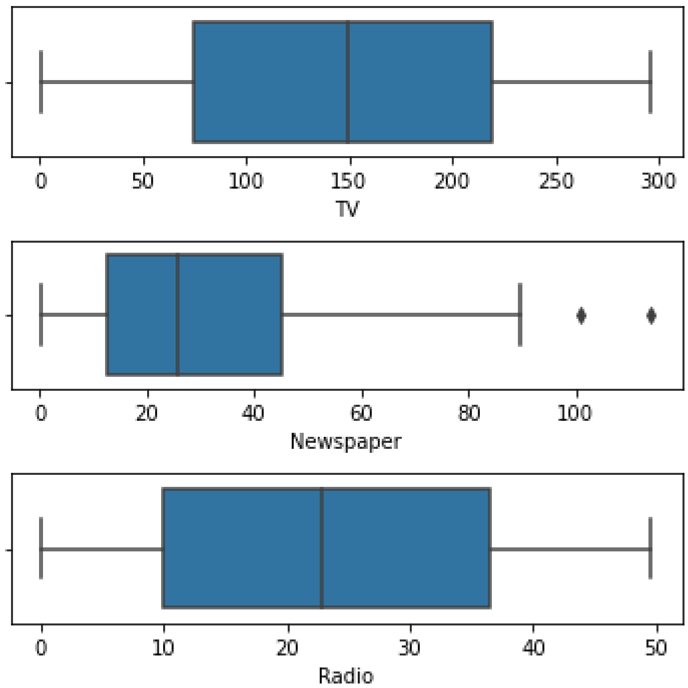
Out[14]:

```
False
```

In [15]:

```
#checking for outliers
fig, axs=plt.subplots(3, figsize=(5,5))
plt1=sns.boxplot(dataset['TV'],ax=axs[0])
plt2=sns.boxplot(dataset['Newspaper'],ax=axs[1])
plt3=sns.boxplot(dataset['Radio'],ax=axs[2])
plt.tight_layout()
```

```
C:\Users\SAHYADRI\anaconda3\lib\site-packages\seaborn\_decorators.py:36: F
utureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing oth
er arguments without an explicit keyword will result in an error or misint
erpretation.
  warnings.warn(
C:\Users\SAHYADRI\anaconda3\lib\site-packages\seaborn\_decorators.py:36: F
utureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing oth
er arguments without an explicit keyword will result in an error or misint
erpretation.
  warnings.warn(
C:\Users\SAHYADRI\anaconda3\lib\site-packages\seaborn\_decorators.py:36: F
utureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing oth
er arguments without an explicit keyword will result in an error or misint
erpretation.
  warnings.warn(
```



In [16]:

```
#Data Pre-Processing
dataset.shape
```

Out[16]:

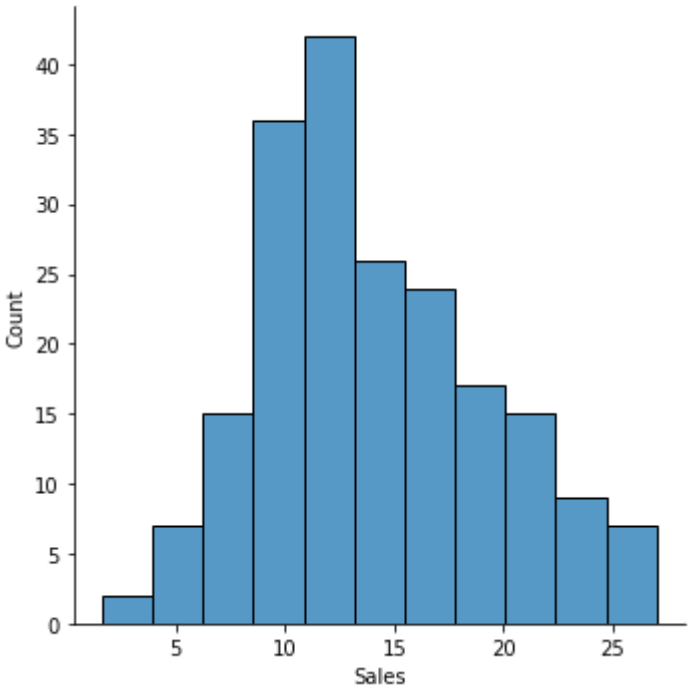
```
(200, 5)
```

In [17]:

```
sns.displot(dataset['Sales'])
```

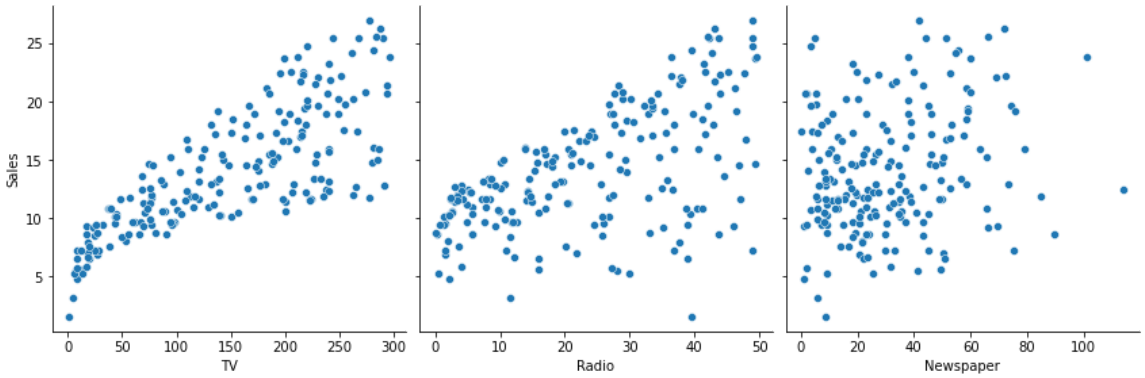
Out[17]:

```
<seaborn.axisgrid.FacetGrid at 0x214e9090a90>
```



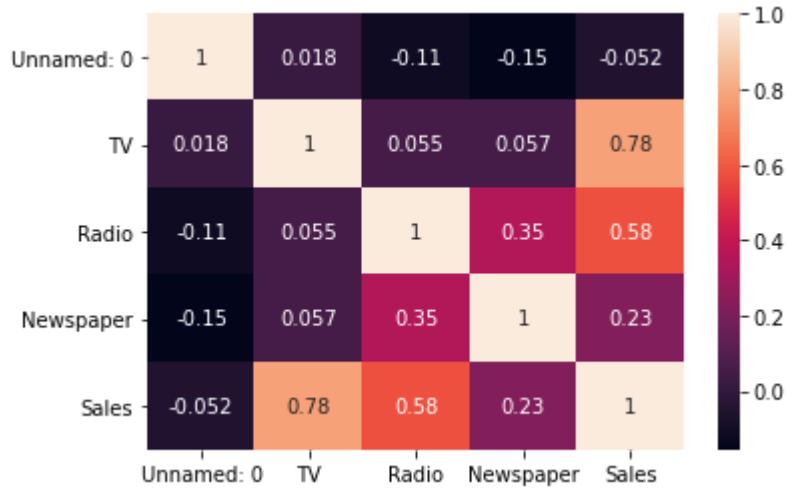
In [19]:

```
sns.pairplot(dataset, x_vars=['TV', 'Radio', 'Newspaper'], y_vars='Sales', height=4, aspe
plt.show()
```



In [20]:

```
#HeatMap
sns.heatmap(dataset.corr(),annot=True)
plt.show()
```



In [22]:

```
'''Model Building

Prediction using:
    1.Simple Linear Regression
    2.Multiple Linear Regression'''
```

Out[22]:

```
'Model Building\n\nPrediction using:\n    1.Simple Linear Regression\n
2.Multiple Linear Regression'
```

In [23]:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

In [24]:

```
#Setting the value for X & Y
x=dataset[['TV']]
y=dataset[['Sales']]
```

In [26]:

```
x_train,x_test,y_train,y_test= train_test_split(x,y,test_size=0.3,random_state=100)
```

In [27]:

```
slr=LinearRegression()
slr.fit(x_train,y_train)
```

Out[27]:

```
LinearRegression()
```

In [28]:

```
# Printing model coefficients
print('Intercept: ', slr.intercept_)
print('Coefficent: ',slr.coef_)
```

```
Intercept:  [6.98966586]
Coefficent:  [[0.04649736]]
```

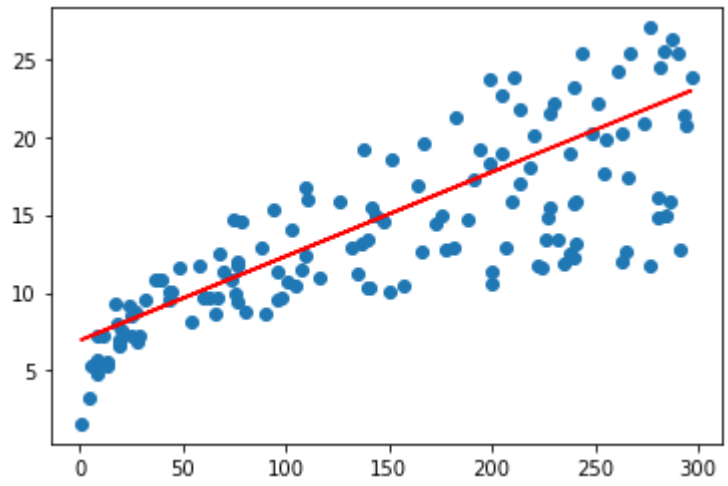
In [29]:

```
print('Regression Equation: Sales= 6.948+0.054*TV')
```

```
Regression Equation: Sales= 6.948+0.054*TV
```

In [30]:

```
#LINE OF BEST FIT
plt.scatter(x_train,y_train)
plt.plot(x_train,6.948+0.054*x_train, 'r')
plt.show()
```



In [42]:

```
#Prediction of Test & Training Set result
y_pred_slr=slr.predict(x_test)
x_pred_slr=slr.predict(x_train)
```

In [49]:

```
print("Prediction for test set: {}".format(y_pred_slr))
```

```
Prediction for test set: [[ 7.35234526]
 [18.06533671]
 [13.27610876]
 [17.11214086]
 [18.22807747]
 [16.60531965]
 [13.4620982 ]
 [16.17754395]
 [17.05169429]
 [17.07029323]
 [12.4391563 ]
 [17.66080969]
 [ 9.60281742]
 [15.72186983]
 [11.04423554]
 [11.36971705]
 [13.95032046]
 [14.90351632]
 [14.59198401]
 [12.23921766]
 [16.97264878]
 [13.00642408]
 [16.07524976]
 [15.21969836]
 [15.58702749]
 [17.23303399]
 [17.20978531]
 [10.49091697]
 [15.58702749]
 [12.71349072]
 [10.1700852 ]
 [10.19798361]
 [12.61584627]
 [15.74976825]
 [ 9.31453379]
 [12.59259759]
 [11.50920913]
 [14.81982107]
 [17.33067844]
 [15.97295557]
 [17.00519693]
 [15.15925179]
 [14.63848137]
 [17.14933874]
 [12.57864838]
 [11.16047894]
 [ 7.77547122]
 [18.55820871]
 [10.27237939]
 [ 8.76586496]
 [16.405381  ]
 [14.95466341]
 [10.4816175 ]
 [13.08546959]
 [16.78665935]
 [ 9.05879832]
 [ 7.78942043]
 [ 8.17999824]
 [16.17754395]
 [10.9744895 ]]
```

In [ ]:

```
slr_diff = pd.DataFrame({'Actual value': y_test, 'Predicted value': y_pred_slr})
slr_diff
```

In [51]:

```
#Predict for any value
slr.predict([[56]])
```

Out[51]:

```
array([[9.59351795]])
```

In [53]:

```
#predict the R-squared value for the model
from sklearn.metrics import accuracy_score
print('R-squared value of the model: {:.2f}'.format(slr.score(x,y)*100))
```

```
R-squared value of the model: 61.02
```