

In [1]:

```

1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 from sklearn.linear_model import LinearRegression
6 from sklearn.linear_model import Lasso,RidgeCV,Ridge
7 from sklearn.preprocessing import StandardScaler
8 from sklearn.model_selection import train_test_split

```

In [2]:

```

1 df=pd.read_csv(r"C:\Users\HP\OneDrive\Desktop\revathi\Advertising.csv")
2 df

```

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]:

```

1 df.head()

```

Out[3]:

	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

In [4]:

```
1 df.tail()
```

Out[4]:

	TV	Radio	Newspaper	Sales
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

In [5]:

```
1 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 [6]:

```
1 df.describe()
```

Out[6]:

	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 [7]:

```
1 df.isnull().sum()
```

Out[7]:

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

LINEAR REGRESSION

In [8]:

```
1 feature = df.columns[0:3]
2 target = df.columns[-1]
3 x=df[feature].values
4 y=df[target].values
5 x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=1)
6 regr=LinearRegression()
7 regr.fit(x_train,y_train)
8 print("Regresson: ",regr.score(x_test,y_test))
```

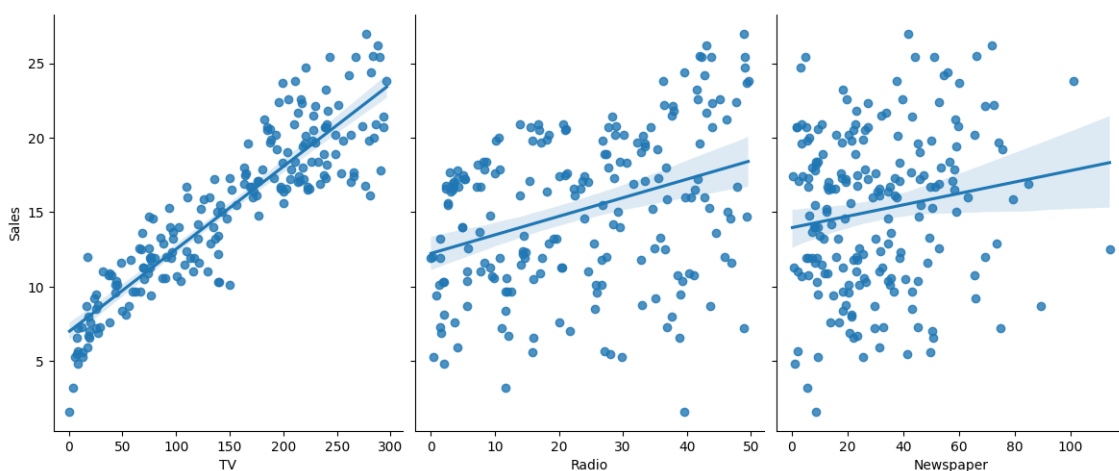
Regresson: 0.9071151423684273

In [9]:

```
1 y_pred=regr.predict(x_test)
2 sns.pairplot(df,x_vars=['TV','Radio','Newspaper'],y_vars=['Sales'],height=5,aspect=)
```

Out[9]:

<seaborn.axisgrid.PairGrid at 0x1876413b790>



RIDGE REGRESSION

In [10]:

```
1 ridgeReg = Ridge(alpha=10)
2 ridgeReg.fit(x_train,y_train)
3 train_score_ridge = ridgeReg.score(x_train, y_train)
4 test_score_ridge = ridgeReg.score(x_test, y_test)
5 print('\nRidge Model\n')
6 print("the score of ridge method is {}".format(train_score_ridge))
7 print("the score of ridge method is {}".format(test_score_ridge))
```

Ridge Model

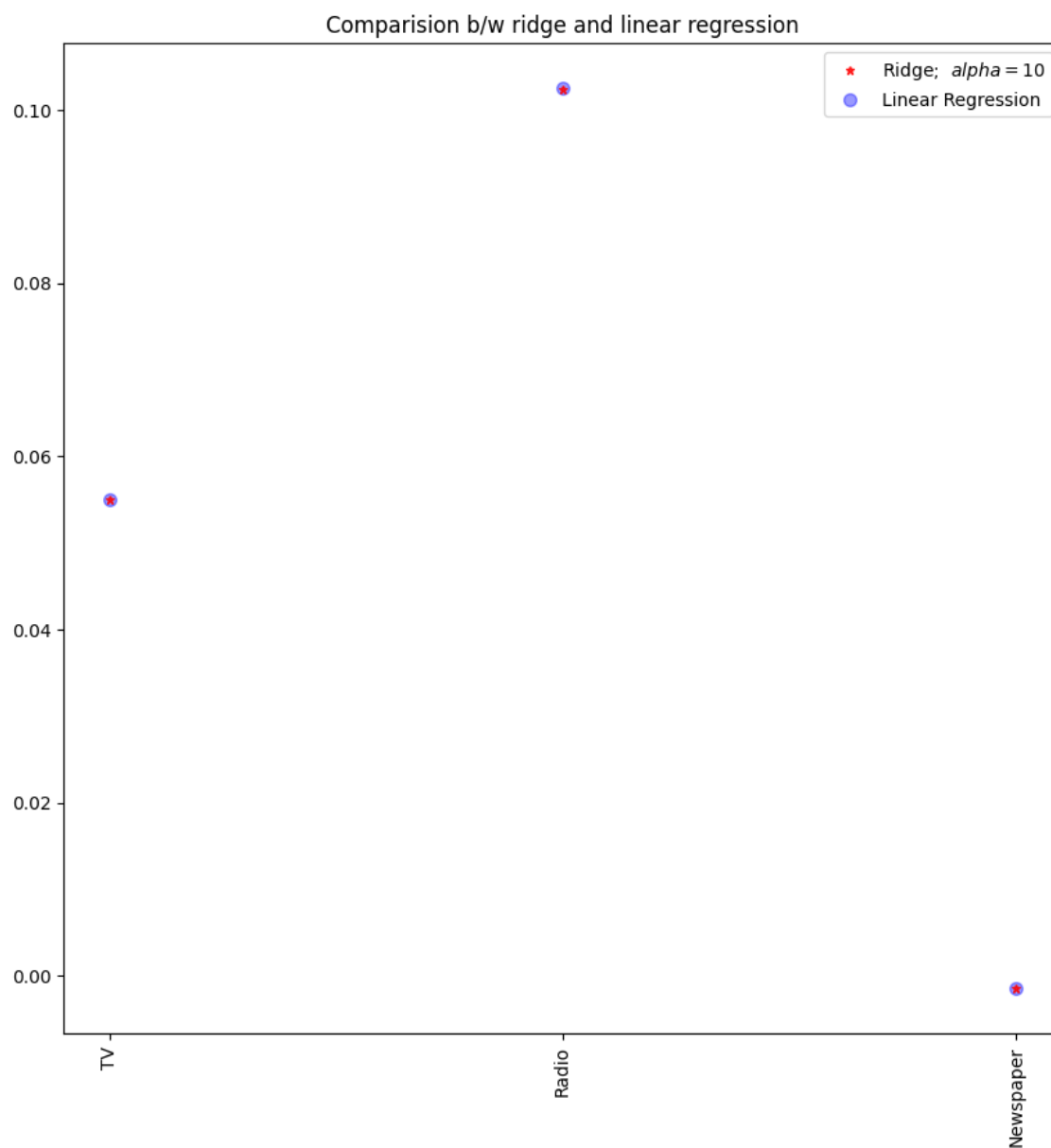
the score of ridge method is 0.8993745741098564

the score of ridge method is 0.9071080215617562

Comparission between ridge and linear regression

In [11]:

```
1 plt.figure(figsize=(10,10))
2 plt.plot(feature,ridgeReg.coef_,alpha=0.8,linestyle='none',marker='*',markersize=5,
3 plt.plot(feature,regr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,colo
4 plt.xticks(rotation=90)
5 plt.legend()
6 plt.title("Comparision b/w ridge and linear regression")
7 plt.show()
```



Lasso regression

In [12]:

```
1 lasso = Lasso(alpha=10)
2 lasso.fit(x_train,y_train)
3 train_score_lasso=lasso.score(x_train,y_train)
4 test_score_lasso=lasso.score(x_test,y_test)
5 print('\n Lasso method\n')
6 print("the score of lasso method is {}".format(train_score_lasso))
7 print("the score of lasso method is {}".format(test_score_lasso))
```

Lasso method

the score of lasso method is 0.8838659530099751
the score of lasso method is 0.8804750072676752

In [13]:

```
1 from sklearn.linear_model import LassoCV
2 lasso_cv = LassoCV(alphas=[0.001,0.001,0.01,0.1,1,10]).fit(x_train,y_train)
3 lasso_cv.fit(x_train,y_train)
4 print('\n Lasso method\n')
5 print("the score of lasso method is {}".format(lasso_cv.score(x_train,y_train)))
6 print("the score of lasso method is {}".format(lasso_cv.score(x_test,y_test)))
```

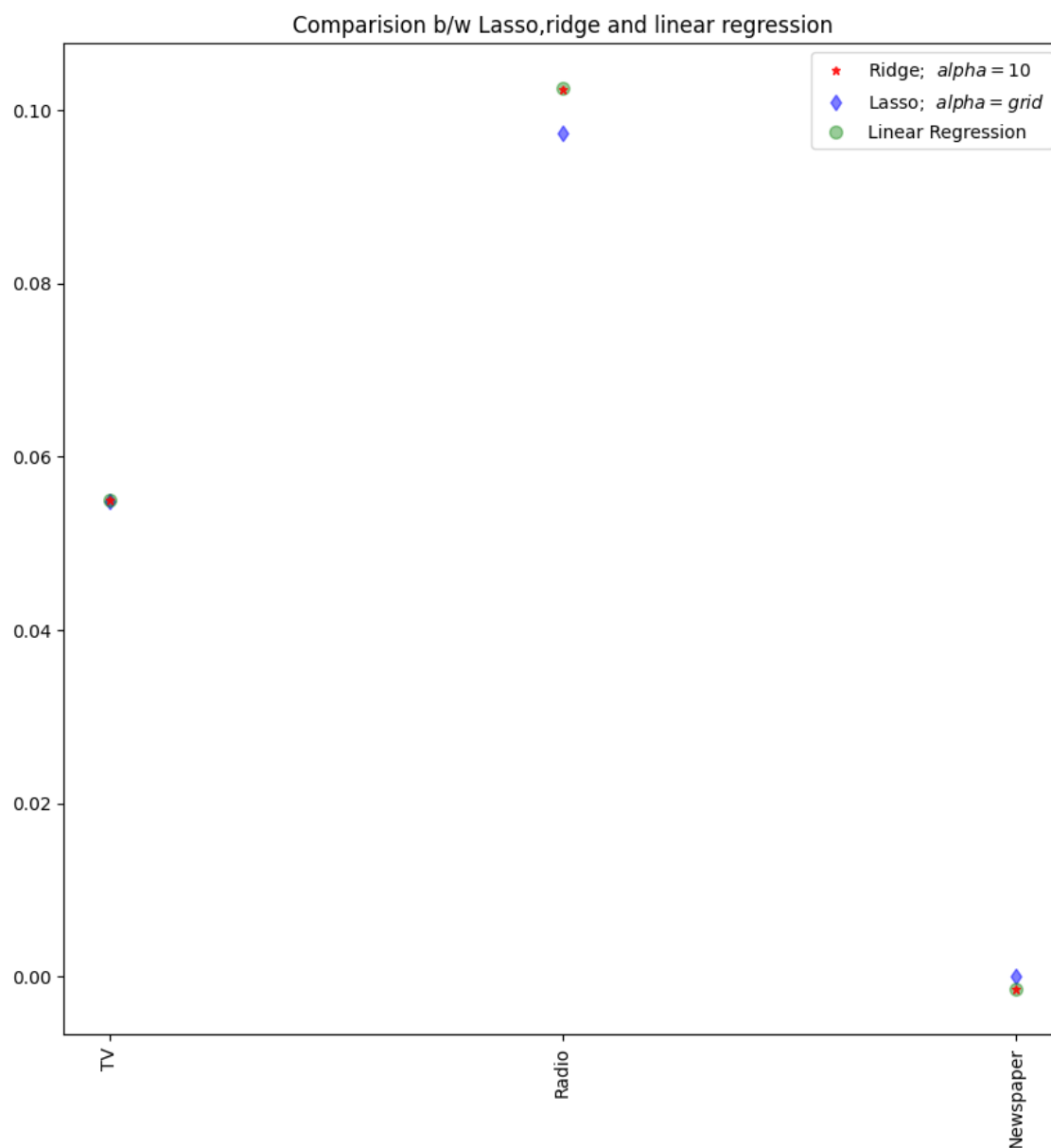
Lasso method

the score of lasso method is 0.8991891680141245
the score of lasso method is 0.9059846692924887

Comparition between Lasso,Ridge and Linear Regression

In [14]:

```
1 plt.figure(figsize=(10,10))
2 plt.plot(feature,ridgeReg.coef_,alpha=0.8,linestyle='none',marker='*',markersize=5,
3 plt.plot(lasso_cv.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,color='b'
4 plt.plot(feature,regr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,colo
5 plt.xticks(rotation=90)
6 plt.legend()
7 plt.title("Comparision b/w Lasso,ridge and linear regression")
8 plt.show()
```



In [23]:

```
1 ridge_cv = RidgeCV(alphas=[0.001,0.001,0.01,0.1,1,10]).fit(x_train,y_train)
2 print('\n Ridge method\n')
3 print("the score of ridge method is {}".format(ridge_cv.score(x_train,y_train)))
4 print("the score of ridge method is {}".format(ridge_cv.score(x_test,y_test)))
```

Ridge method

the score of ridge method is 0.8993745741098558

the score of ridge method is 0.907108021563779

In [16]:

```
1 from sklearn.linear_model import ElasticNet
2 lr = ElasticNet()
3 lr.fit(x_train,y_train)
4 print("The ElasticNet score is {}".format(lr.score(x_train,y_train)))
5 print("The elasticnet score is {}".format(lr.score(x_test,y_test)))
```

The ElasticNet score is 0.899297276466209

The elasticnet score is 0.9066357943437607

In [17]:

```
1 lr.fit(x,y)
2 print(lr.coef_)
3 print(lr.intercept_)
```

```
[0.05440081 0.1046715  0.          ]
4.696191158087224
```

In [18]:

```
1 y_pred_elastic = lr.predict(x_train)
```

In [19]:

```
1 mean_squared_error=np.mean(((y_pred_elastic)-y_train)**2)
2 print(mean_squared_error)
```

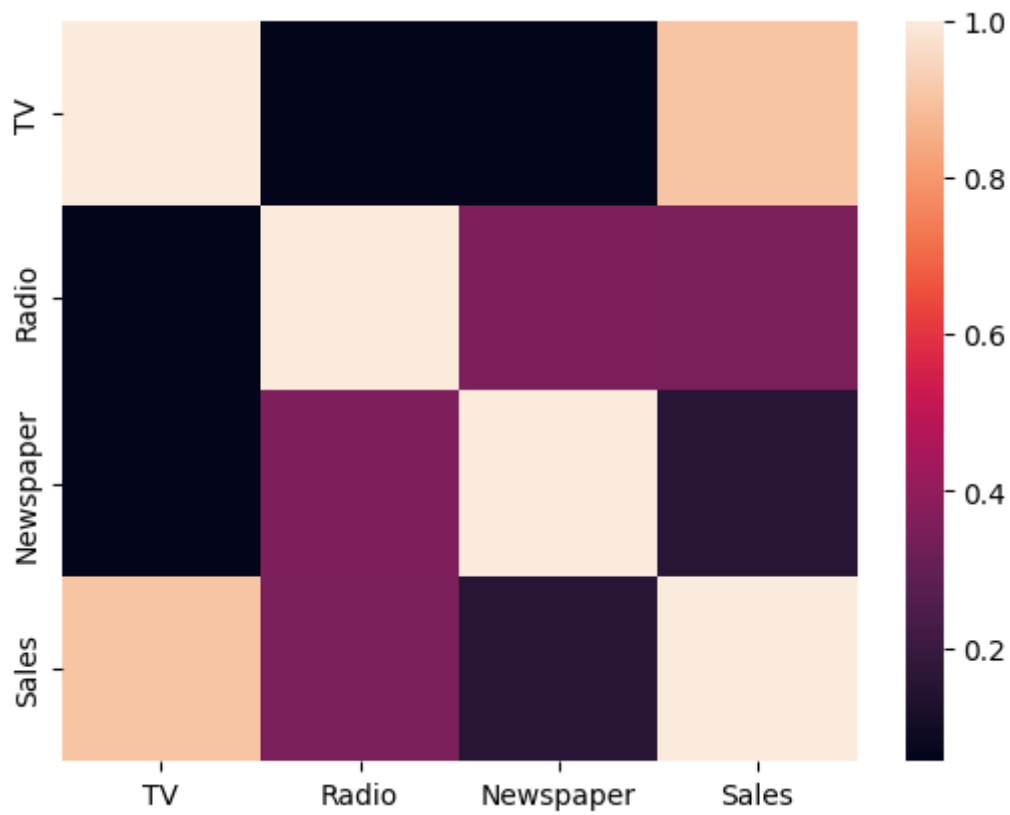
2.873170425561577

In [28]:

```
1 sns.heatmap(df.corr())
```

Out[28]:

<Axes: >



In []:

```
1
```