Linear Regerssion Model

Step 1:Importing functions

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

Step 2: Import the csv file into jupyter notebook

In [2]:
```
df=pd.read_csv(r"C:\Users\Sushma sree\Downloads\Advertising.csv")
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

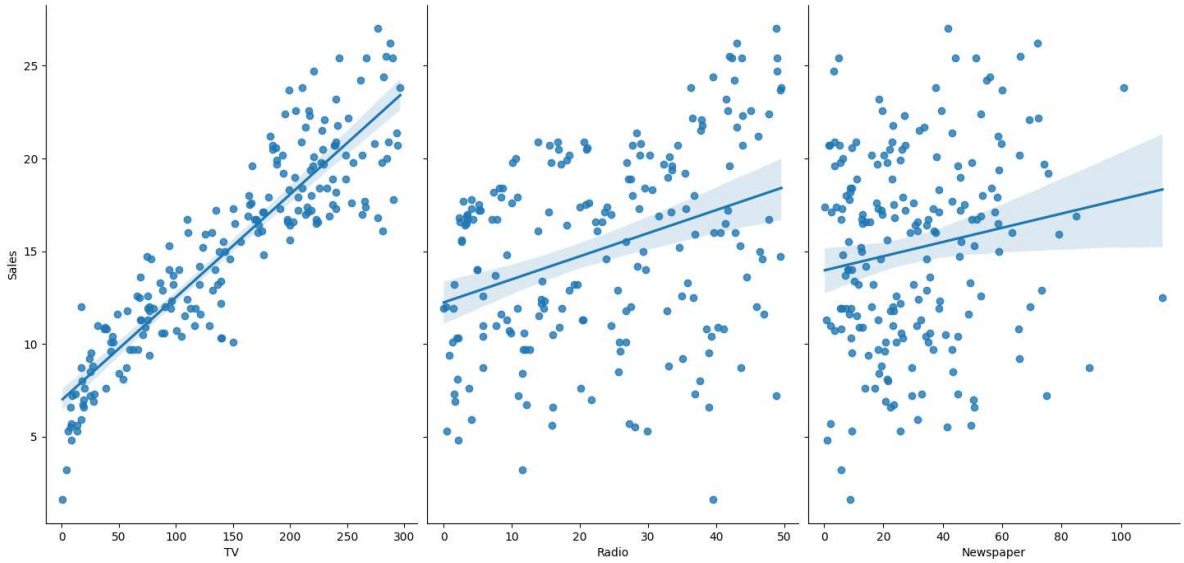
In [3]:
```
df.head()
```

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]:
```
sns.pairplot(df,x_vars=['TV','Radio','Newspaper'],y_vars='Sales',height=7,aspe
```

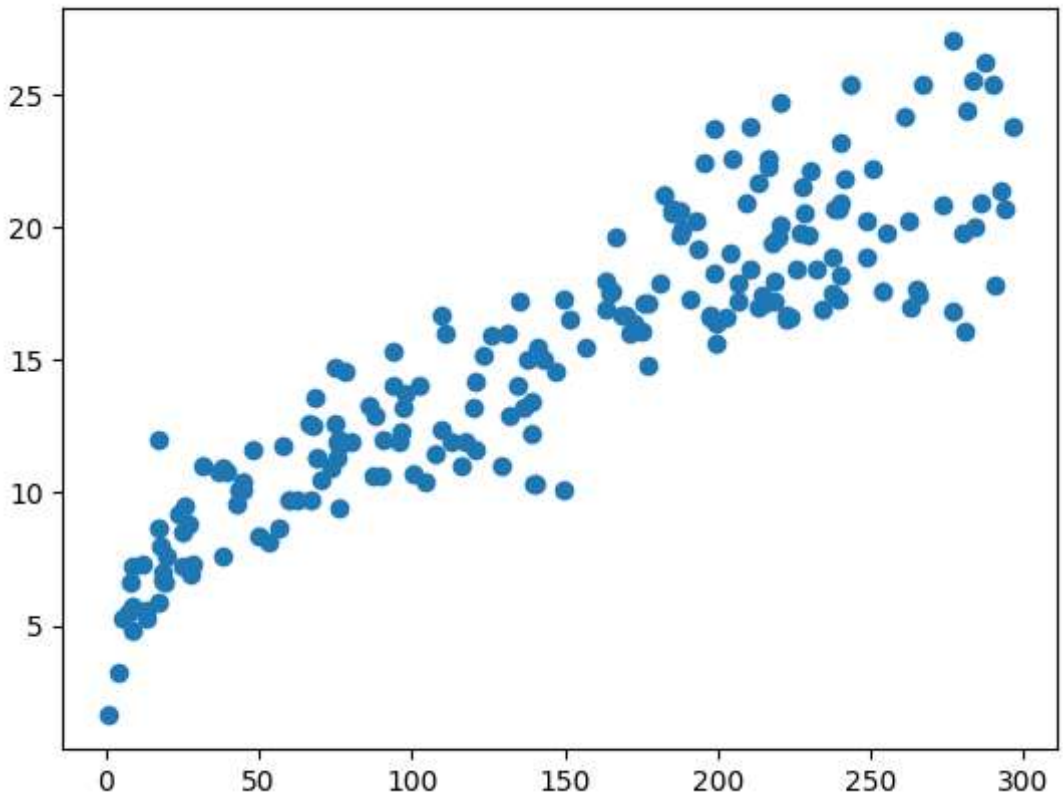
Out[4]:
```
<seaborn.axisgrid.PairGrid at 0x21c276f1bd0>
```



Step3

In [5]:
```
plt.scatter(df['TV'],df['Sales'])
```

Out[5]:
```
<matplotlib.collections.PathCollection at 0x21c29dd7e80>
```



In [6]:
```
x=df[['TV']]
y=df['Sales']
x.head()
```

Out[6]:

	TV
0	230.1
1	44.5
2	17.2
3	151.5
4	180.8

In [7]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
x_train
```

Out[7]:

	TV
129	59.6
156	93.9
56	7.3
96	197.6
46	89.7
...	...
148	38.0
83	68.4
135	48.3
183	287.6
95	163.3

140 rows × 1 columns

In [8]:
```
x_test
```

Out[8]:

	TV
187	191.1
169	284.3
47	239.9
143	104.6
134	36.9
127	80.2
180	156.6
26	142.9
89	109.8
124	229.5
12	23.8
152	197.6
53	182.6
104	238.2
92	217.7
82	75.3
70	199.1
22	13.2
5	8.7
35	290.7
99	135.2
57	136.2
188	286.0
39	228.0
173	168.4
113	209.6
151	121.0
168	215.4
144	96.2
196	94.2
190	39.5
2	17.2
75	16.9
118	125.7
86	76.3

	TV
105	137.9
63	102.7
109	255.4
21	237.4
177	170.2
140	73.4
160	172.5
17	281.4
111	241.7
108	13.1
145	140.3
87	110.7
125	87.2
69	216.8
76	27.5
137	273.7
126	7.8
64	131.1
128	220.3
9	199.8
119	19.4
191	75.5
78	5.4
97	184.9
106	25.0

In [9]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

In [10]:
```
lr.fit(x_train,y_train)
```

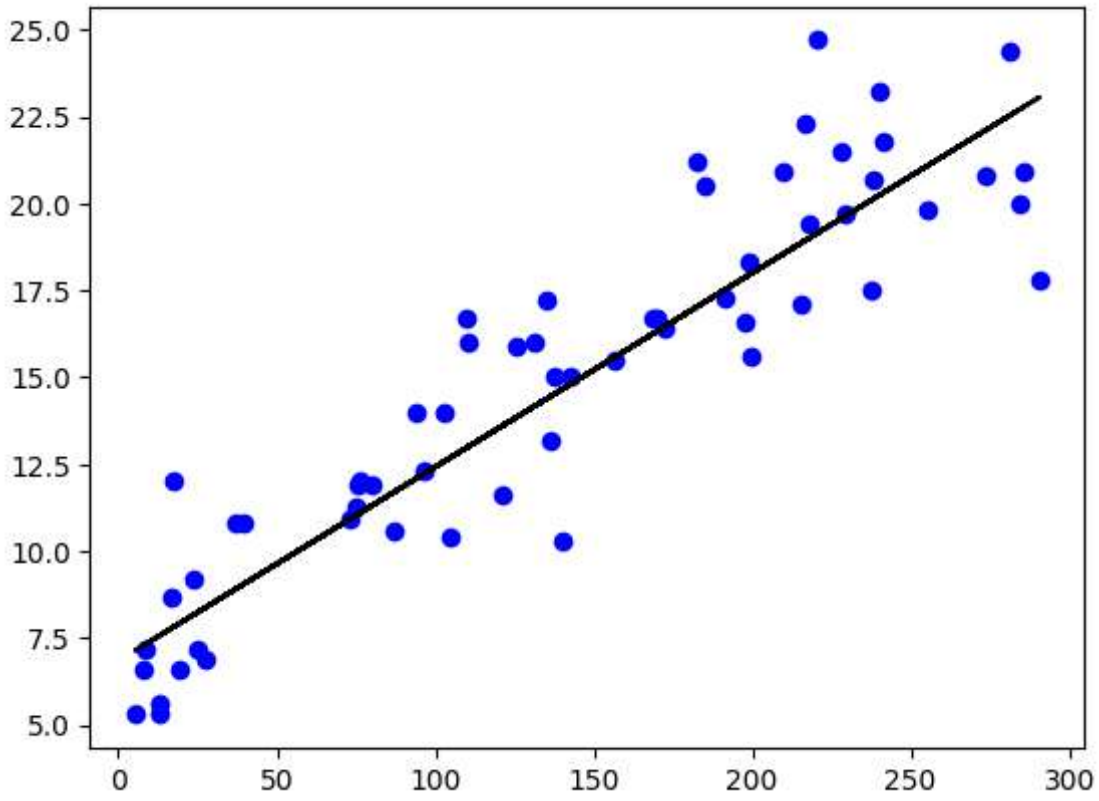
Out[10]:
```
▾ LinearRegression
LinearRegression()
```

In [11]:
```
lr.predict(x_test)
```

Out[11]:
```
array([17.50605788, 22.70453365, 20.22800657, 12.68129227,  8.90514624,
       11.32031793, 15.5817294 , 14.81757577, 12.97133598, 19.64791914,
        8.1744592 , 17.86861252, 17.03194796, 20.13318458, 18.98974302,
       11.0470075 , 17.95227897,  7.58321624,  7.33221688, 23.06151053,
       14.38808796, 14.4438656 , 22.79935563, 19.56425268, 16.23990552,
       18.53794416, 13.59604552, 18.86145446, 12.21276012, 12.10120484,
        9.0501681 ,  7.80632679,  7.7895935 , 13.85820041, 11.10278514,
       14.53868758, 12.57531476, 21.09255994, 20.08856247, 16.34030526,
       10.94102999, 16.46859383, 22.5427785 , 20.32840631,  7.57763848,
       14.67255391, 13.02153586, 11.71076139, 18.93954315,  8.38083645,
       22.1132907 ,  7.282017  , 14.15939965, 19.13476488, 17.99132332,
        7.92903759, 11.05816303,  7.14815067, 17.16023653,  8.24139236])
```

In [12]:
```
print('Regression:',lr.score(x_test,y_test))
y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```

```
Regression: 0.8284746469697153
```



In [13]:
```
df100=df[:][:100]
```

In [14]:
```
df100.isna().any()
```

Out[14]:
```
TV           False
Radio        False
Newspaper    False
Sales        False
dtype: bool
```

In [15]:
```
x=df100[['TV']]
y=df100['Sales']
x.head()
```

Out[15]:

	TV
0	230.1
1	44.5
2	17.2
3	151.5
4	180.8

In [16]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
x_train
```

Out[16]:

	TV
94	107.4
2	17.2
95	163.3
15	195.4
42	293.6
...	...
65	69.0
24	62.3
84	213.5
74	213.4
40	202.5

70 rows × 1 columns

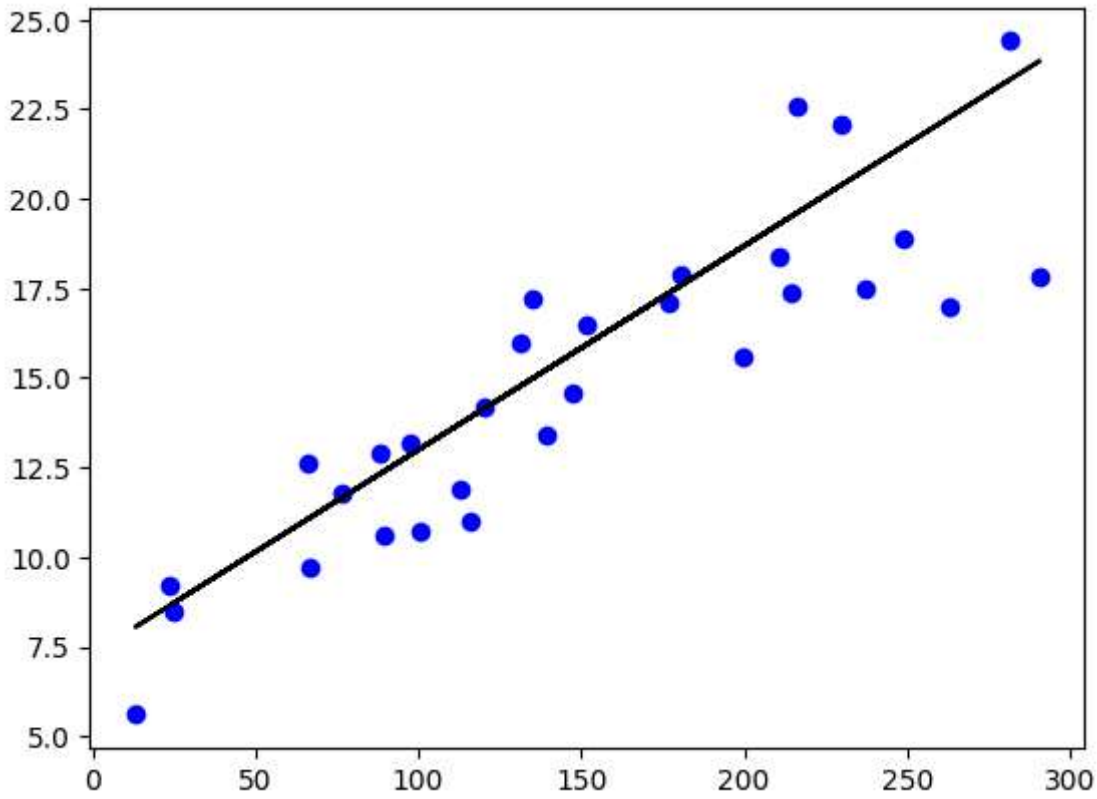
In [17]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

In [18]:
```
lr.fit(x_train,y_train)
```

Out[18]:
```
LinearRegression
LinearRegression()
```

In [19]:
```
print('Regression:',lr.score(x_test,y_test))
y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```

```
Regression: 0.7231975176500547
```



Ridge Regression Model

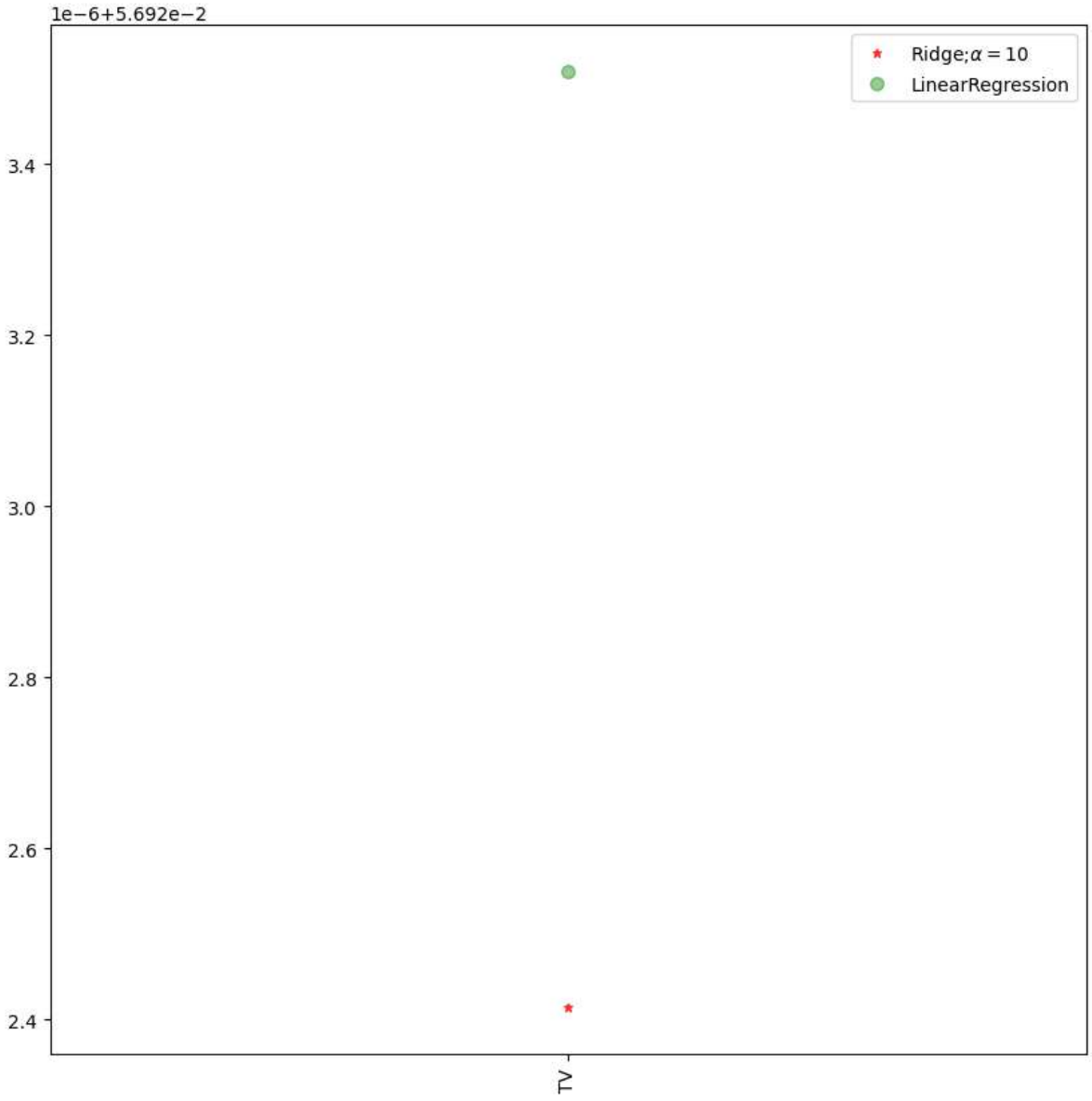
In [20]:
```
from sklearn.linear_model import Ridge,RidgeCV
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
```

In [21]:
```
ridgeReg=Ridge(alpha=10)
ridgeReg.fit(x_train,y_train)
train_score_ridge=ridgeReg.score(x_train,y_train)
test_score_ridge=ridgeReg.score(x_test,y_test)
print('\nRidgeModel:')
print("The train score for ridge model is {}".format(train_score_ridge))
print("The test score for ridge model is {}".format(test_score_ridge))
```

```
RidgeModel:
The train score for ridge model is 0.8401159033915412
The test score for ridge model is 0.7232029497084462
```

In [22]:
```
features=['TV']
target=['Sales']
```

In [23]:
```
plt.figure(figsize=(10,10))
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markers
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker="o",markersize=7,
plt.xticks(rotation=90)
plt.legend()
plt.show()
```

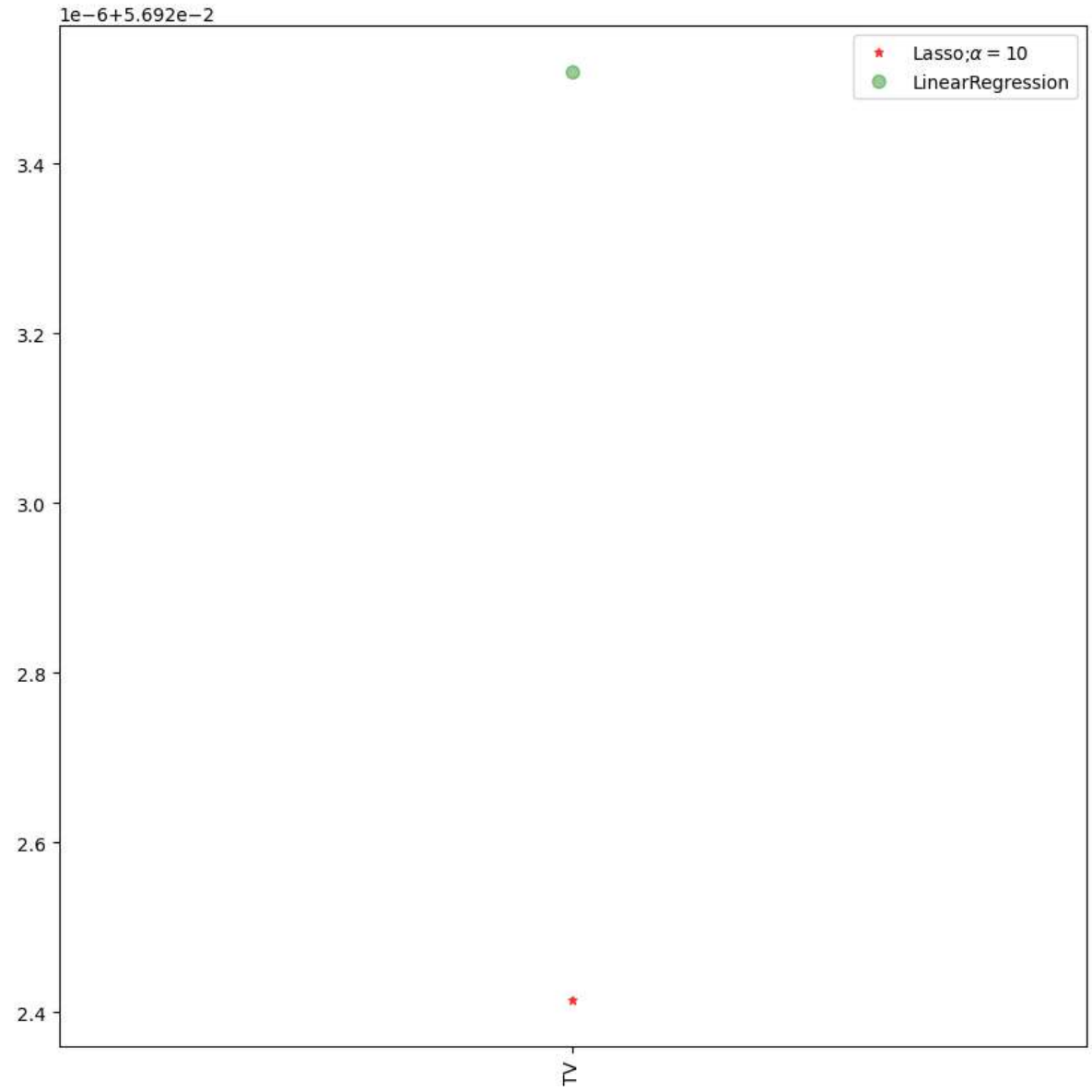


Lasso Regression Model

In [24]:
```
lassoReg=Ridge(alpha=10)
lassoReg.fit(x_train,y_train)
train_score_lasso=lassoReg.score(x_train,y_train)
test_score_lasso=lassoReg.score(x_test,y_test)
print('\nLassoModel:')
print("The train score for Lasso model is {}".format(train_score_lasso))
print("The test score for Lasso model is {}".format(test_score_lasso))
```

```
LassoModel:
The train score for Lasso model is 0.8401159033915412
The test score for Lasso model is 0.7232029497084462
```

In [25]:
```
.figure(figsize=(10,10))
.plot(features,lassoReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=
.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker="o",markersize=7,colo
.xticks(rotation=90)
.legend()
.show()
```



In [26]:
```
from sklearn.linear_model import LassoCV
lasso_cv=LassoCV(alphas=[0.0001,0.001,0.01,0.1,1,10],random_state=0).fit(x_tra
print(lasso_cv.score(x_train,y_train))
print(lasso_cv.score(x_test,y_test))
```

```
0.840115903701502
0.723197584455123
```

In [ ]: