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

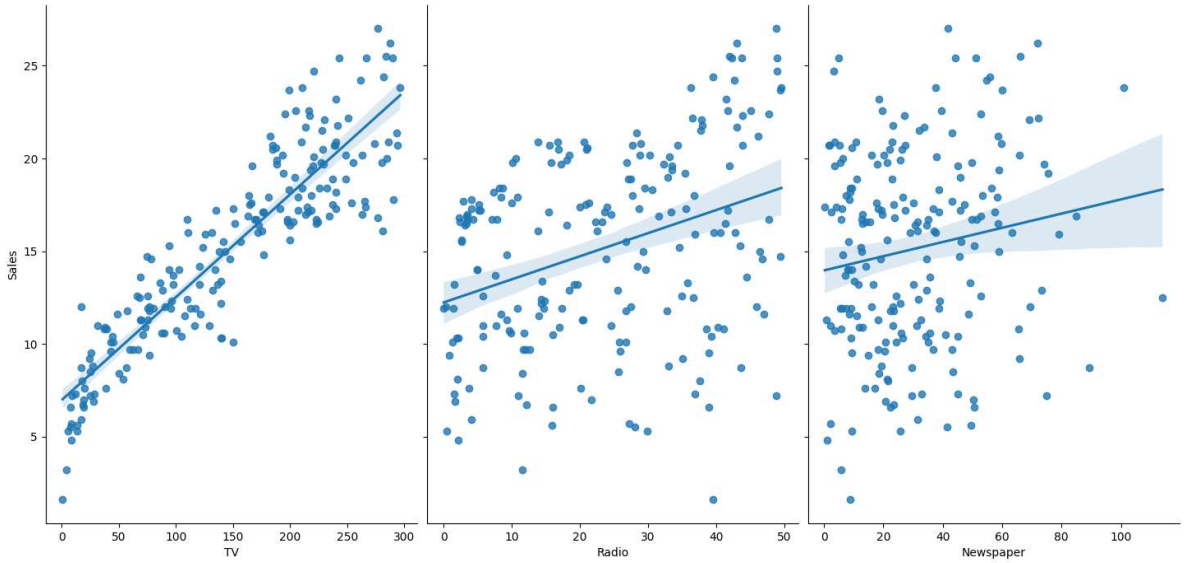
In [3]:
```
df.head()
```

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
```

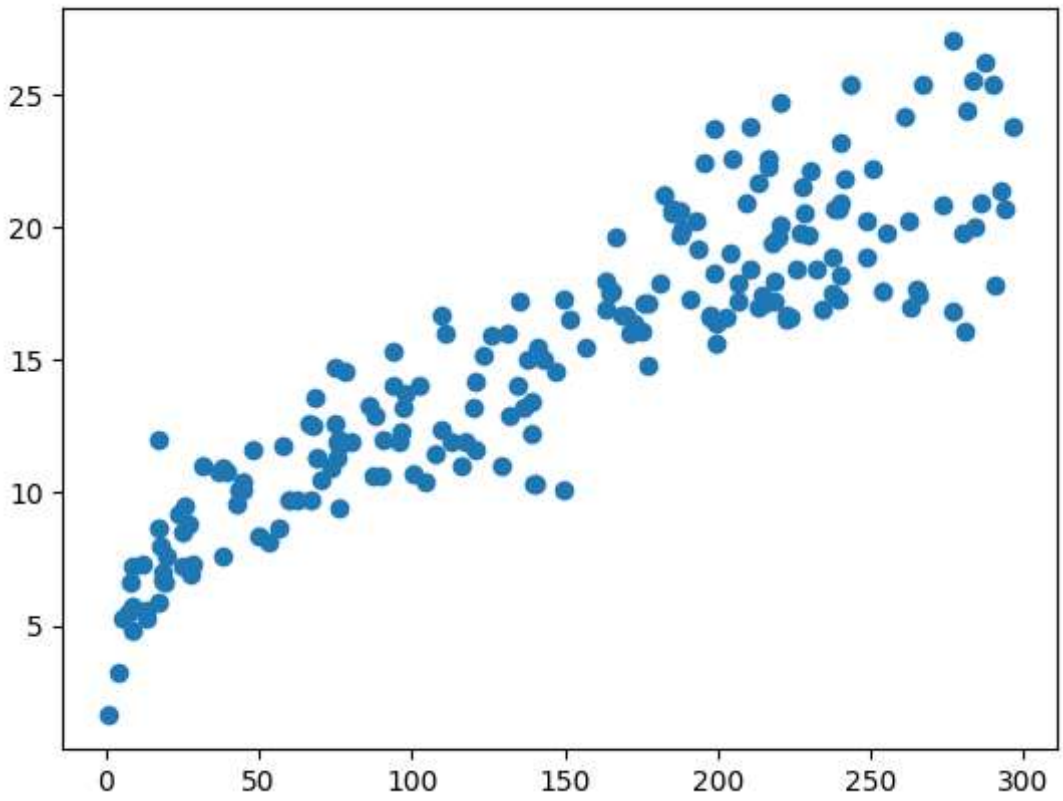
Out[4]: <seaborn.axisgrid.PairGrid at 0x1bfcf5a1ff0>



Step3

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

Out[5]: <matplotlib.collections.PathCollection at 0x1bfd1c8beb0>



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
128	220.3
89	109.8
145	140.3
44	25.1
103	187.9
...	...
131	265.2
115	75.1
26	142.9
144	96.2
49	66.9

140 rows × 1 columns

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

Out[8]:

	TV
94	107.4
175	276.9
73	129.4
192	17.2
156	93.9
40	202.5
32	97.2
69	216.8
187	191.1
23	228.3
60	53.5
65	69.0
8	8.6
114	78.2
20	218.4
160	172.5
136	25.6
37	74.7
189	18.7
135	48.3
171	164.5
168	215.4
46	89.7
56	7.3
130	0.7
190	39.5
70	199.1
146	240.1
66	31.5
59	210.7
109	255.4
83	68.4
170	50.0
112	175.7
181	218.5

	TV
43	206.9
100	222.4
54	262.7
52	216.4
154	187.8
58	210.8
25	262.9
30	292.9
167	206.8
79	116.0
116	139.2
142	220.5
71	109.8
50	199.8
113	209.6
137	273.7
132	8.4
182	56.2
68	237.4
101	296.4
153	171.3
108	13.1
123	123.1
1	44.5
31	112.9

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

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

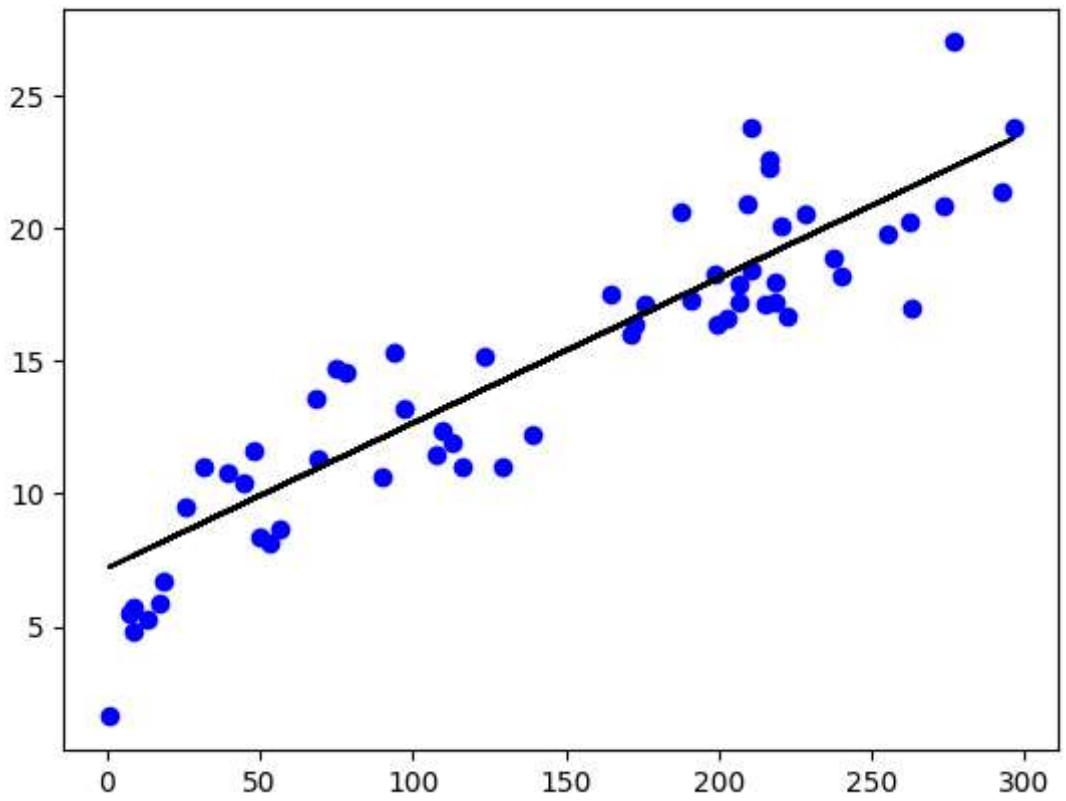
Out[10]:
```
▾ LinearRegression
LinearRegression()
```

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

Out[11]:
```
array([13.05440324, 22.31082222, 14.25582635,  8.12856848, 12.31716633,
       18.2478277 , 12.4973798 , 19.02875272, 17.62527208, 19.65676935,
       10.11091662, 10.95737381,  7.65892126, 11.45978711, 19.11612895,
       16.60952345,  8.58729367, 11.26865162,  8.21048369,  9.82694388,
       16.17264232, 18.95229852, 12.08780374,  7.58792808,  7.22750115,
        9.34637464, 18.06215322, 20.30116902,  8.9094935 , 18.69563086,
       21.13670418, 10.92460773,  9.91978112, 16.78427591, 19.12158996,
       18.48811232, 19.33456951, 21.53535821, 19.00690866, 17.44505862,
       18.70109187, 21.54628024, 23.18458449, 18.48265131, 13.52405046,
       14.79100574, 19.23081024, 13.18546758, 18.10038031, 18.6355597 ,
       22.13606977,  7.64799924, 10.258364  , 20.15372163, 23.37571998,
       16.54399128,  7.9046669 , 13.91178246,  9.61942534, 13.35475902])
```

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: 0.8328377086114365
```



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
3	151.5
47	239.9
68	237.4
88	88.3
95	163.3
...	...
50	199.8
77	120.5
62	239.3
71	109.8
60	53.5

70 rows × 1 columns

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

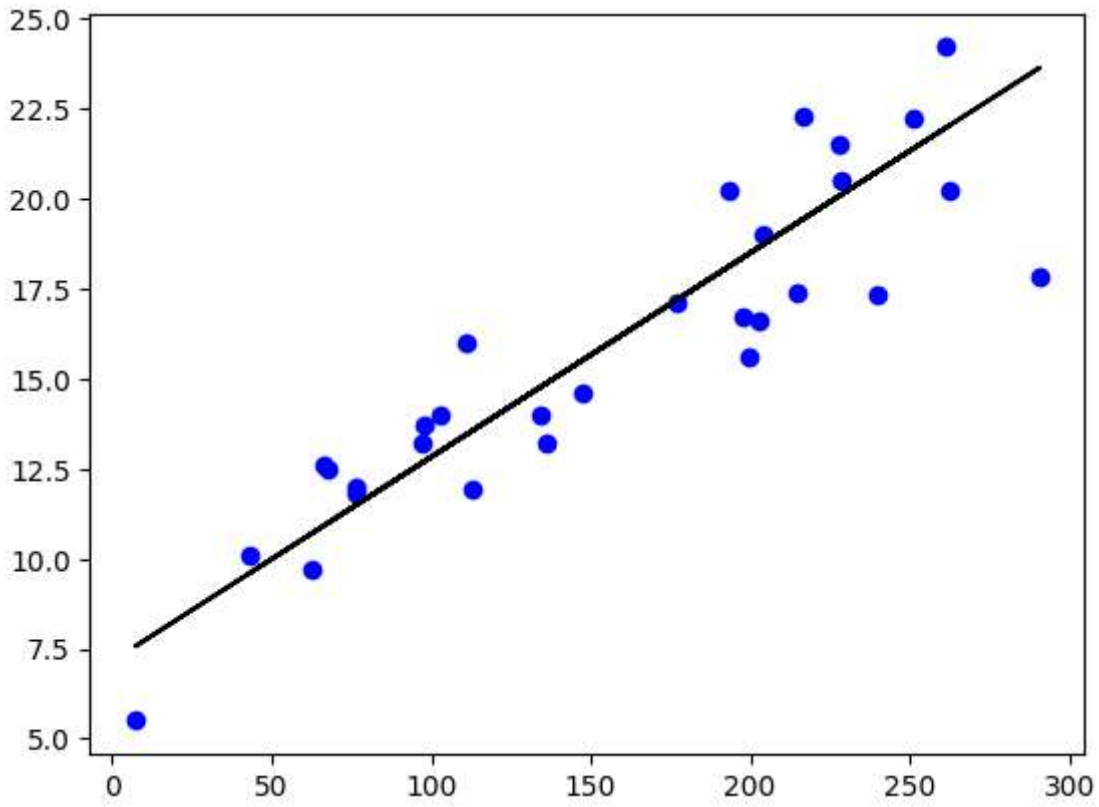
In [18]:
```
lr.fit(x_train,y_train)
```

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: 0.7851849402642901
```



Ridge Regression Model

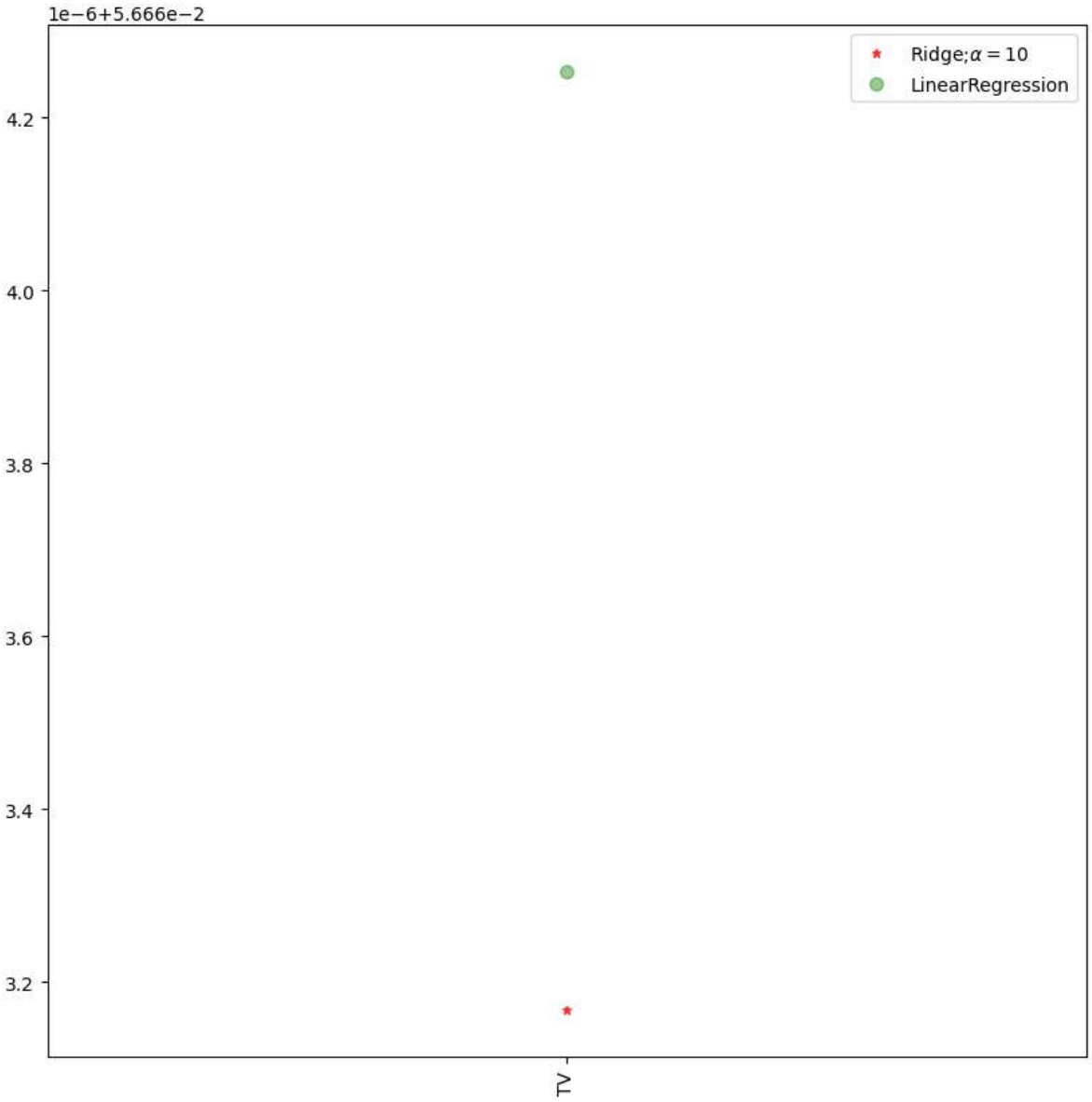
In [20]:
```
from sklearn.linear_model import Ridge,RidgeCV
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
```

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 0.8268181077960175
The test score for ridge model is 0.7851898097399101
```

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()
```

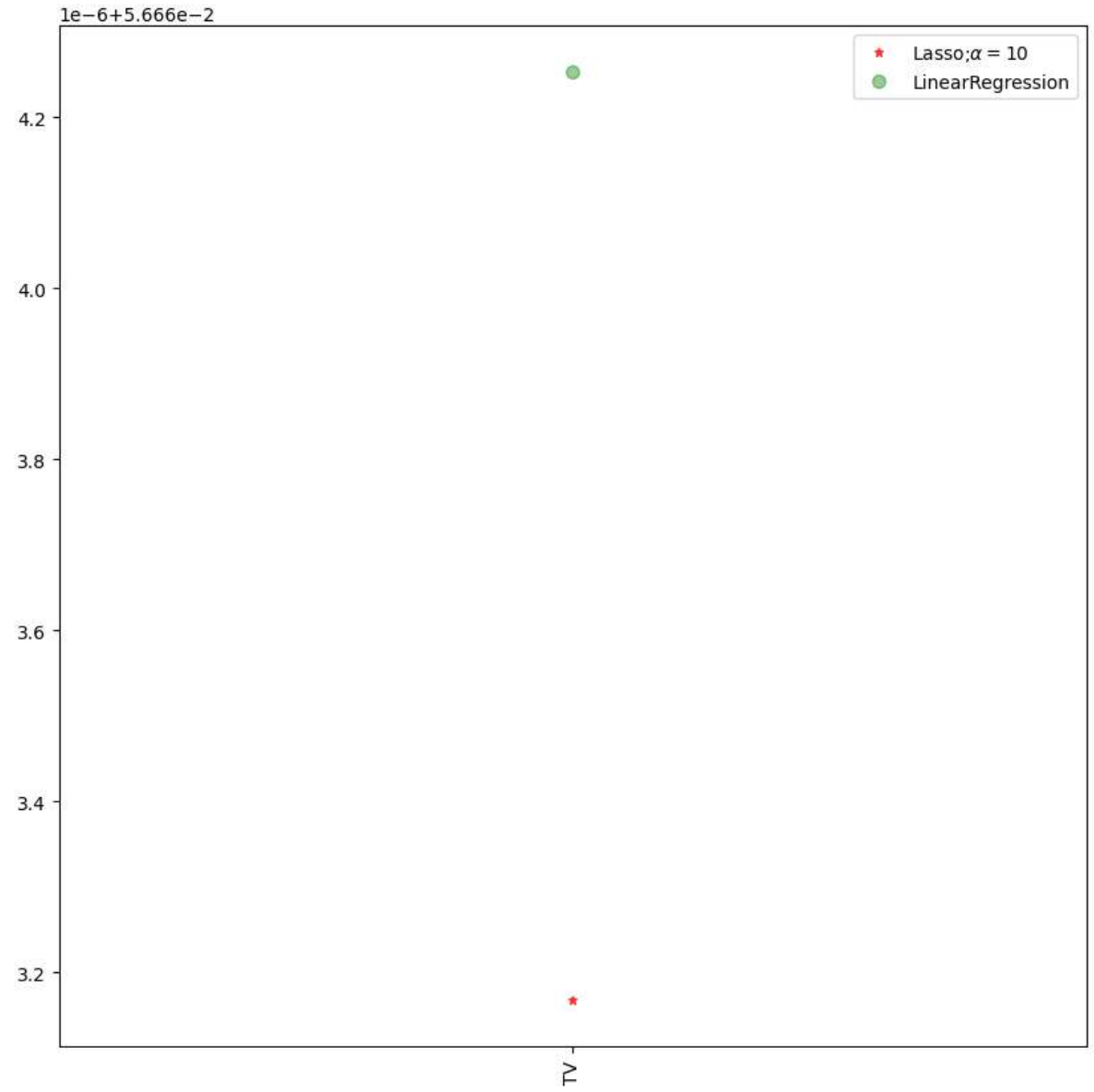


Lasso Regression Model

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 0.8268181077960175
The test score for Lasso model is 0.7851898097399101
```

In [25]:
```
plt.figure(figsize=(10,10))
plt.plot(features,lassoReg.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()
```



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))
```

```
0.8268181080992734
0.7851850004250708
```

Elastic Net

In [27]:
```
from sklearn.linear_model import ElasticNet
regr=ElasticNet()
regr.fit(x,y)
print(regr.coef_)
print(regr.intercept_)
```

```
[0.05490453]
7.334974110731961
```

In [28]:
```
y_pred_elastic=regr.predict(x_train)
```

In [30]:
```
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print("Mean Squared Error on test set",mean_squared_error)
```

```
Mean Squared Error on test set 5.044367375264402
```

In [ ]:
```

```