In [57]:
```python
import pandas as pd
import numpy as np
import seaborn  as sns
from sklearn  .model_selection import train_test_split
from matplotlib import  pyplot as plt
```

In [58]:
```python
df= pd.read_csv(r"C:\Users\teppa\Downloads\Advertising.csv")
df
```

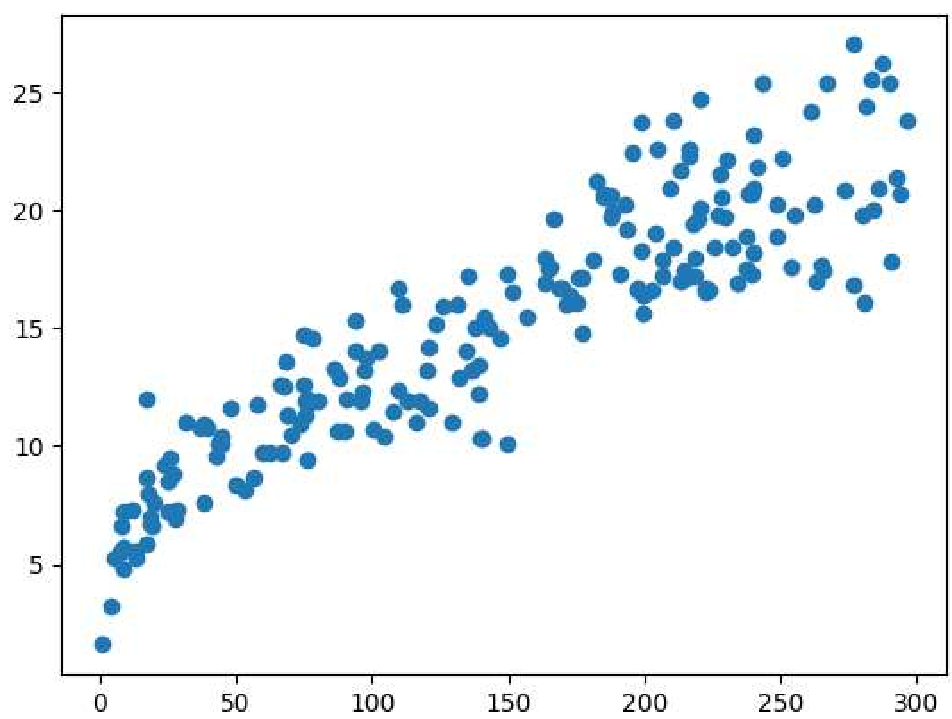
Out[58]:

	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 [59]:
```python
plt.scatter(df['TV'],df['Sales'])
```

Out[59]: <matplotlib.collections.PathCollection at 0x201e070aaa0>



In [60]:
```
x=df[['TV']]
y=np.array(df['Sales']).reshape(-1,1)
```

In [61]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
x_train
```

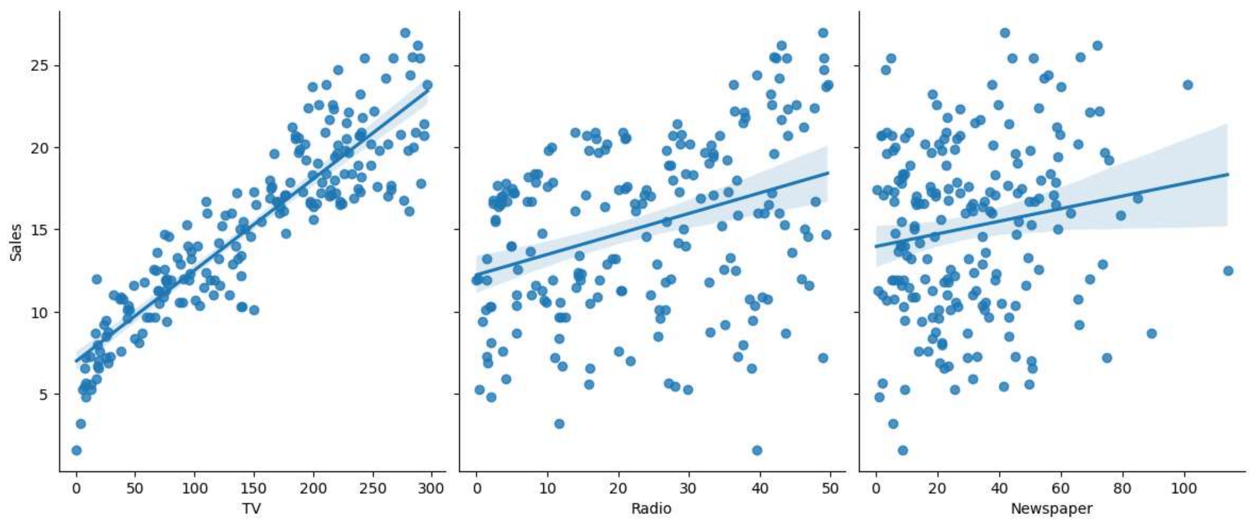
Out[61]:

	TV
195	38.2
29	70.6
88	88.3
14	204.1
79	116.0
...	...
44	25.1
140	73.4
149	44.7
73	129.4
45	175.1

140 rows × 1 columns

In [62]:
```
sns.pairplot(df,x_vars=["TV","Radio","Newspaper"],y_vars='Sales',height=5,aspect=0.8,kind='r
```

Out[62]:
```
<seaborn.axisgrid.PairGrid at 0x201da7a46d0>
```



In [63]:
```
len(x_train)
```

Out[63]:
```
140
```

In [64]:
```
len(x_test)
```

Out[64]:
```
60
```

In [65]:
```
len(y_train)
```
Out[65]:
```
140
```

In [66]:
```
len(y_test)
```
Out[66]:
```
60
```

In [67]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

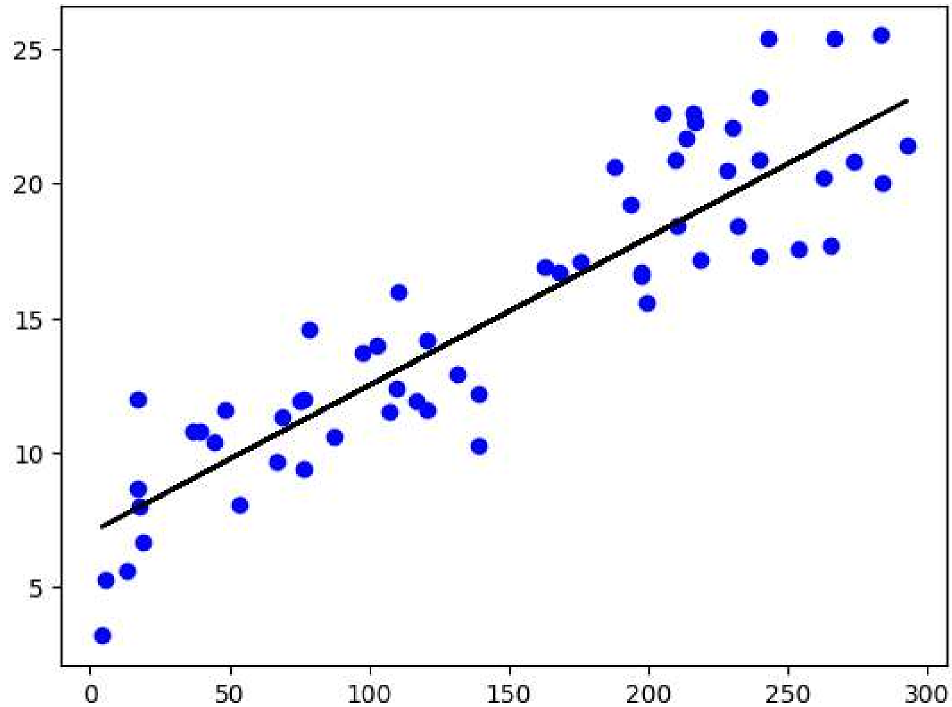
In [68]:
```
lr.fit(x_train,y_train)
```
Out[68]:
```
▾ LinearRegression
LinearRegression()
```

In [69]:
```
lr.predict(x_test)
```

Out[69]:
```
array([[13.03303058],
       [19.52785079],
       [ 9.94730588],
       [20.16363066],
       [17.84522648],
       [ 9.66230111],
       [ 7.73851892],
       [22.59713291],
       [10.79683932],
       [ 7.23976057],
       [17.30810211],
       [18.25081019],
       [20.34449907],
       [11.20242303],
       [22.55876688],
       [ 9.17998535],
       [ 7.99611938],
       [23.06848695],
       [18.50292979],
       [14.23333913],
       [18.99072642],
       [16.24481509],
       [13.6194827 ],
       [20.92547032],
       [17.84522648],
       [ 9.03748296],
       [21.64346311],
       [12.64388945],
       [11.30107853],
       [21.55028847],
       [20.17459238],
       [20.15814979],
       [16.64491794],
       [12.35888469],
       [11.79435601],
       [13.64688701],
       [ 9.45402839],
       [ 8.03996627],
       [11.15309528],
       [17.6314729 ],
       [15.96529118],
       [21.41326695],
       [10.68174124],
       [13.43861429],
       [17.96580542],
       [18.56321926],
       [22.01616165],
       [14.6444037 ],
       [ 7.95775336],
       [13.08235833],
       [18.71668337],
       [14.66084628],
       [19.7361235 ],
       [18.87562834],
       [ 7.94131077],
       [19.62650628],
       [12.90148992],
       [ 7.31101177],
       [18.89755178],
       [11.19694217]])
```

In [70]:
```
print("Rgression:",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()
```

```
Rgression: 0.8191900194075661
```



In [71]:
```
lr.score(x_test,y_test)
```

Out[71]:
```
0.8191900194075661
```

In [72]:
```
df100=df[:][:180]
```

In [73]:
```
x=df100[["TV"]]
y=df100["Sales"]
```

In [74]:
```python
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
x_train
```

Out[74]:

	TV
100	222.4
132	8.4
70	199.1
3	151.5
144	96.2
...	...
148	38.0
142	220.5
15	195.4
66	31.5
74	213.4

126 rows × 1 columns

In [75]:
```python
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

In [76]:
```python
feature=["TV"]
target=["Sales"]
```

In [77]:
```python
lr.fit(x_train,y_train)
```

Out[77]:
```
▾ LinearRegression
LinearRegression()
```

In [78]:
```python
lr.predict(x_test)
```

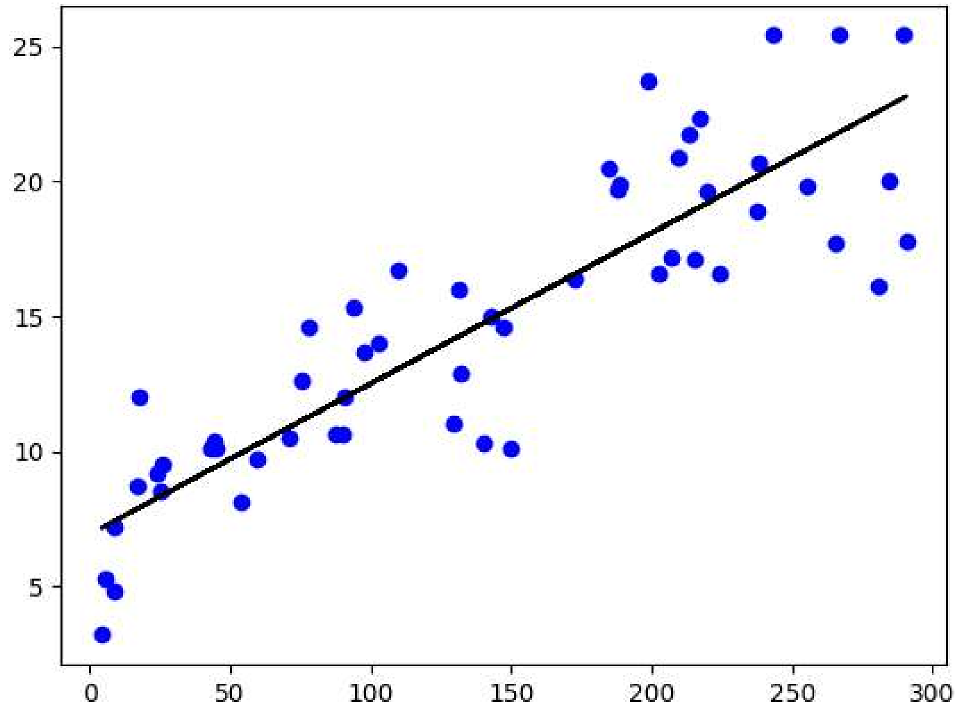
Out[78]:
```
array([23.08838931, 16.55080001, 21.81657159, 17.43772552, 11.29060656,
       19.02191951,  9.41077073, 23.14417079, 15.14510674, 18.83784063,
        8.32861005, 10.86666733, 18.46410472,  7.22971491, 14.24144679,
       17.40983478, 21.72174308, 15.28456044, 19.18926395,  7.41379379,
       12.16637578, 20.49455055, 20.17101797,  7.87120192,  7.40821565,
       11.9711406 , 20.21564316, 12.3671891 , 19.42354616, 11.79263987,
        8.25609412, 14.75463639, 13.05330129, 22.586356  ,  7.15719899,
        8.35650078,  9.91280404,  9.42192703, 10.25307106, 18.62029287,
       11.93209357, 14.14661827, 17.24249034, 18.22424437,  9.33267666,
       18.94382544, 14.89966824, 14.27491567, 18.02343104,  7.88793636,
       22.78716932, 11.11768398, 21.17508459, 12.65725279])
```

In [79]:
```python
lr.score(x_test,y_test)
```

Out[79]:
```
0.7555145093619565
```

In [80]:
```
print("Rgression:",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()
```

```
Rgression: 0.7555145093619565
```



In [81]:
```
from sklearn.linear_model import Ridge,RidgeCV,Lasso
from sklearn.preprocessing import StandardScaler
```

In [82]:
```
ridge=Ridge(alpha=10)
ridge.fit(x_train,y_train)
```

Out[82]:
```
▾ Ridge
Ridge(alpha=10)
```

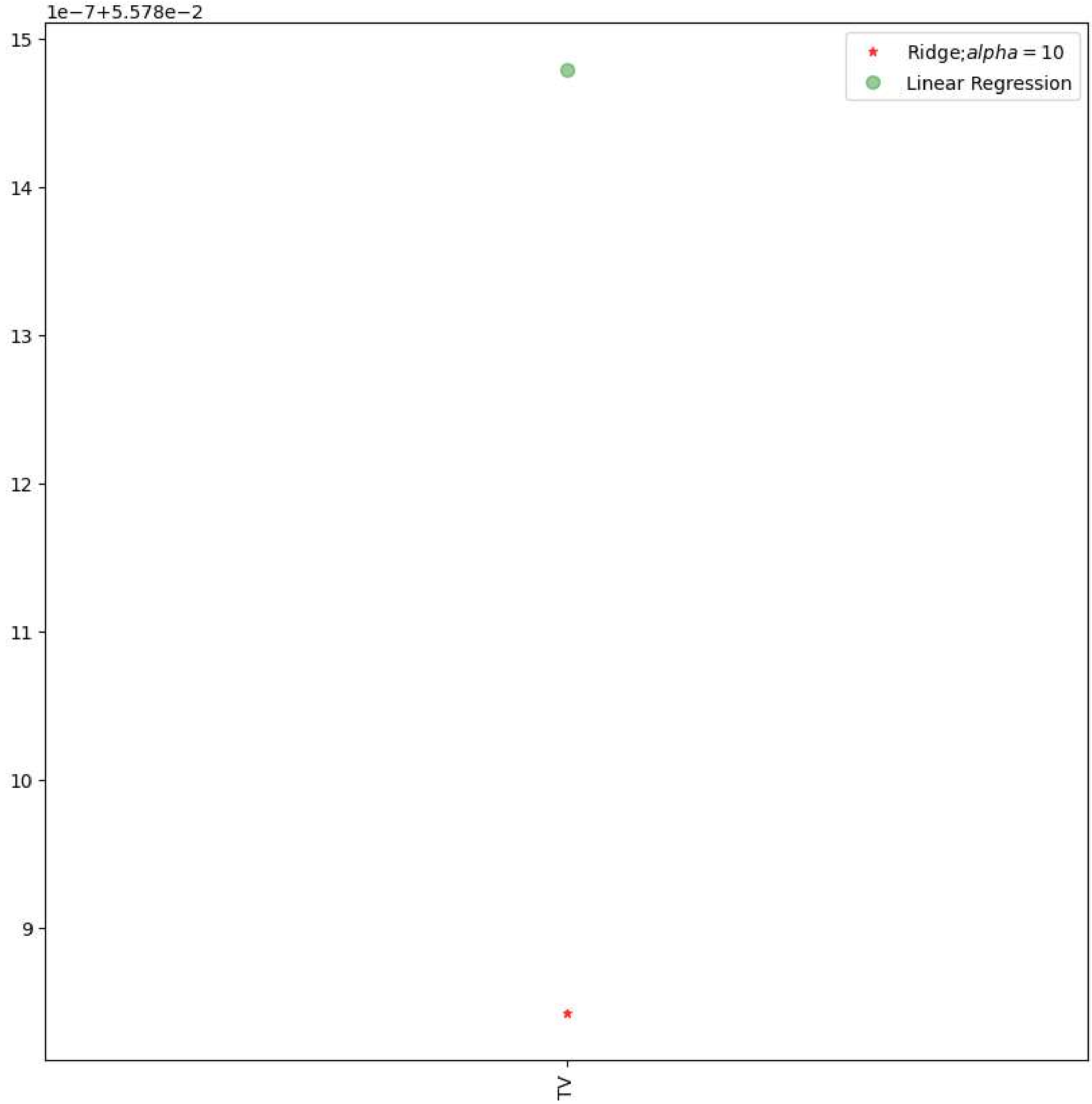
In [83]:
```
train_score_ridge=ridge.score(x_train,y_train)
test_score_ridge=ridge.score(x_test,y_test)
print("\nRidge Model:\n")
print("The train score for ridge model is {}".format(train_score_ridge))
print("The test score for ridge model is {}".format(test_score_ridge))
```

```

Ridge Model:

The train score for ridge model is 0.8349938083289384
The test score for ridge model is 0.7555153346434035
```

In [84]:
```
plt.figure(figsize=(10,10))
plt.plot(feature,ridge.coef_,alpha=0.7,linestyle='None',marker='*',markersize=5,color='red',l
plt.plot(feature,lr.coef_,alpha=0.4,linestyle="none",marker='o',markersize=7,color='green',la
plt.xticks(rotation=90)
plt.legend()
plt.show()
```



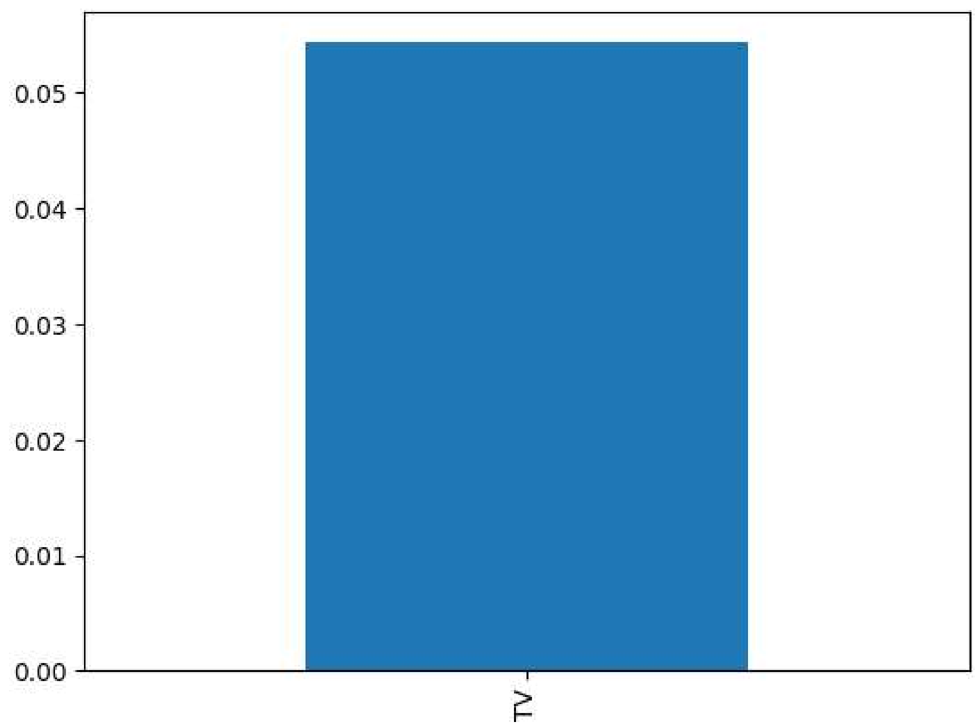
In [85]:
```
print("\nLasso Model:\n")
lasso=Lasso(alpha=10)
lasso.fit(x_train,y_train)
train_score_ls=lasso.score(x_train,y_train)
test_score_ls=lasso.score(x_test,y_test)
print("The train score for ridge model is {}".format(train_score_ridge))
print("The test score for ridge model is {}".format(test_score_ridge))
```

```
Lasso Model:

The train score for ridge model is 0.8349938083289384
The test score for ridge model is 0.7555153346434035
```

In [86]:
```
pd.Series(lasso.coef_,feature).sort_values(ascending=True).plot(kind="bar")
```

Out[86]: <Axes: >



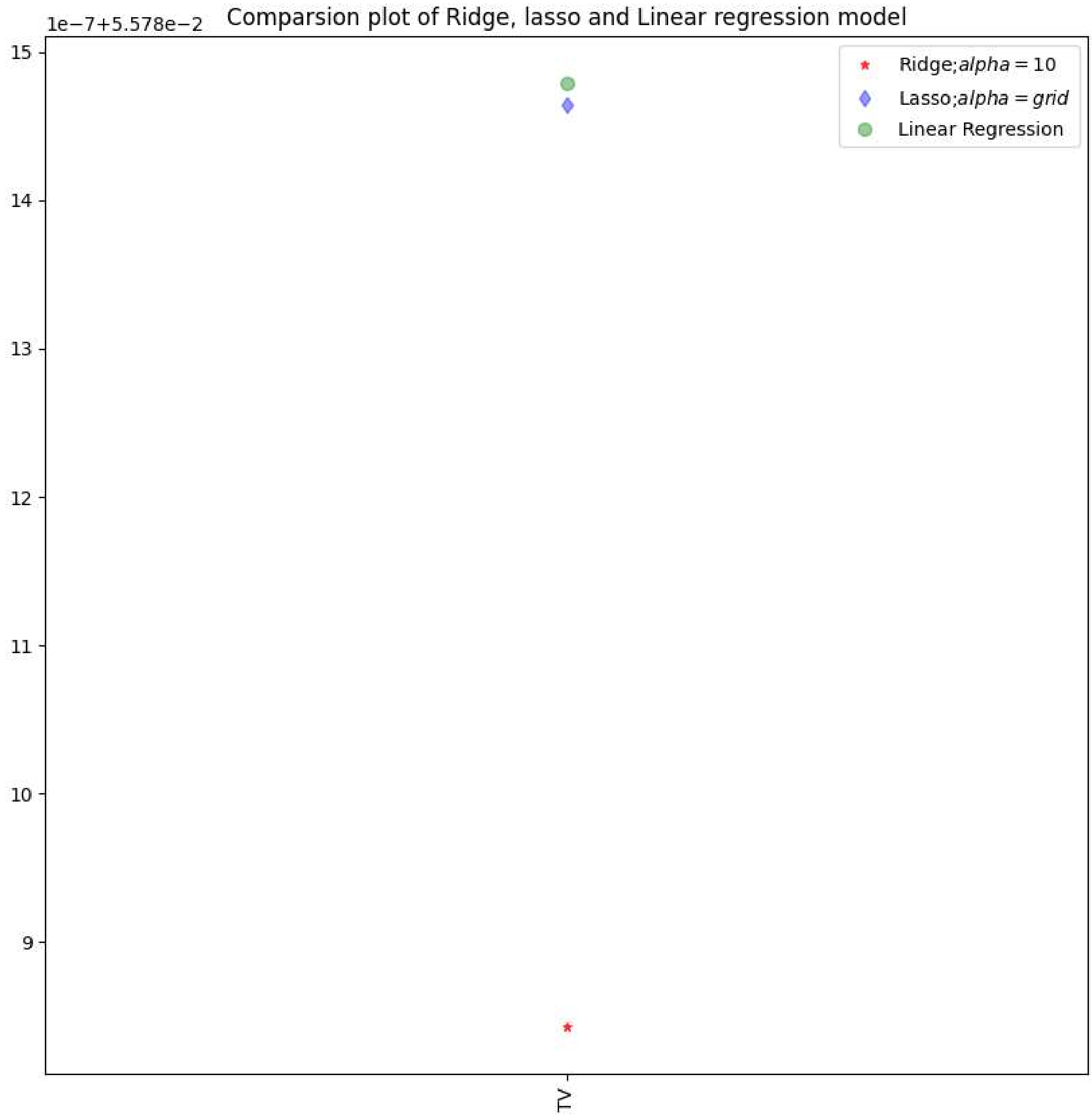
In [87]:
```
from sklearn.linear_model import LassoCV
```

In [88]:
```
lasso_cv=LassoCV(alphas=[0.0001,0.001,0.01,1,10],random_state=0).fit(x_train,y_train)
```

In [89]:
```
print(lasso_cv.score(x_train,y_train))
print(lasso_cv.score(x_test,y_test))
```

```
0.8349938084374653
0.7555145280061444
```

In [90]:
```
plt.figure(figsize=(10,10))
plt.plot(feature,ridge.coef_,alpha=0.7,linestyle='None',marker='*',markersize=5,color='red',l
plt.plot(lasso_cv.coef_,alpha=0.4,linestyle='none',marker='d',markersize=6,color='blue',label
plt.plot(feature,lr.coef_,alpha=0.4,linestyle="none",marker='o',markersize=7,color='green',la
plt.xticks(rotation=90)
plt.legend()
plt.title("Comparsion plot of Ridge, lasso and Linear regression model")
plt.show()
```



In [91]:
```
from sklearn.linear_model import RidgeCV
```

In [92]:
```
ridge_cv=RidgeCV(alphas=[0.0001,0.001,0.01,1,10]).fit(x_train,y_train)
```

In [93]:
```
print("The train score for ridge for ridge model is {}".format(ridge_cv.score(x_train,y_trai
print("The test score for ridge for ridge model is {}".format(ridge_cv.score(x_test,y_test))
```

```
The train score for ridge for ridge model is 0.8349938083289381
The test score for ridge for ridge model is 0.7555153346448448
```

In [94]:
```
from sklearn.linear_model import ElasticNet
```

In [95]:
```
regr=ElasticNet()
regr.fit(x,y)
print(regr.coef_)
print(regr.intercept_)
```

```
[0.05491733]
7.065459085981752
```

In [102]:
```
y_predt_elastic=regr.predict(x_train)
```

In [104]:
```

mean_squared_error=np.mean((y_predt_elastic-y_train)**2)
print("Mean squared Error on test set",mean_squared_error)
```

```
Mean squared Error on test set 4.284671287369669
```

In [ ]:

In [ ]: