In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Ridge,RidgeCV
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
```

In [2]:

```
data=pd.read_csv(r"C:\Users\Prathyusha\Downloads\Advertising.csv")
data
```

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]:

```
data.head(10)
```

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
5	8.7	48.9	75.0	7.2
6	57.5	32.8	23.5	11.8
7	120.2	19.6	11.6	13.2
8	8.6	2.1	1.0	4.8
9	199.8	2.6	21.2	15.6

In [4]:

```
data.tail()
```

Out[4]:

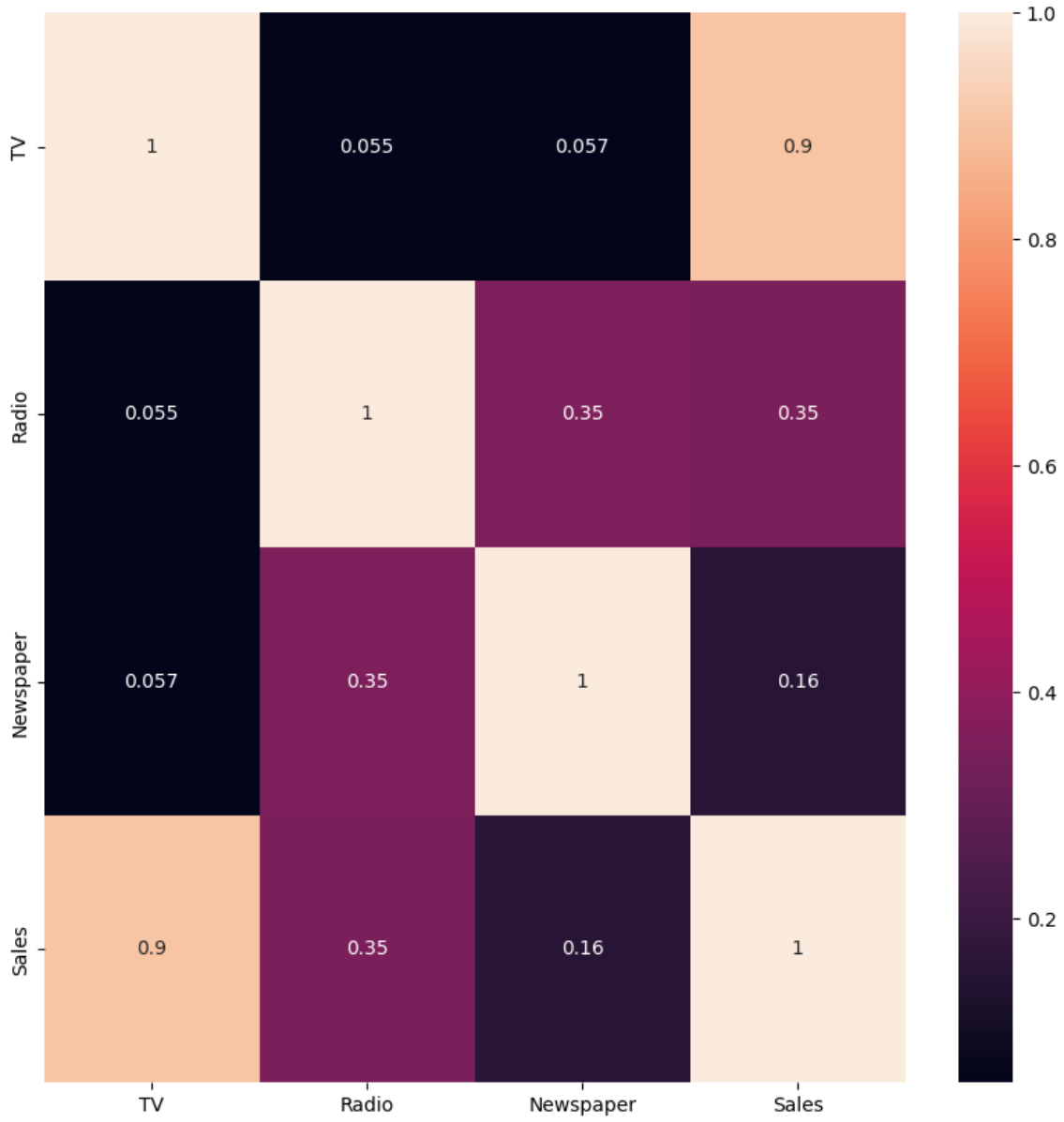
	TV	Radio	Newspaper	Sales
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

In [5]:

```
plt.figure(figsize=(10,10))
sns.heatmap(data.corr(),annot=True)
```

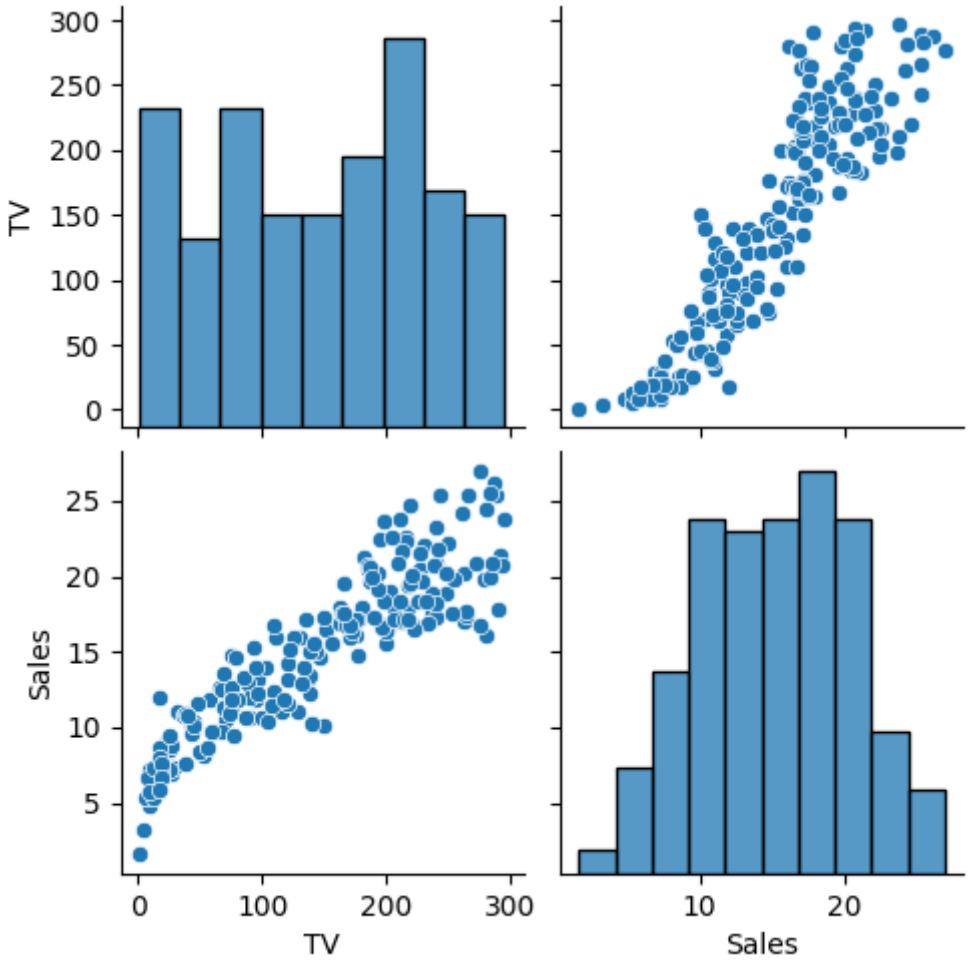
Out[5]:

<Axes: >



In [6]:

```
data.drop(columns=["Radio","Newspaper"],inplace=True)
sns.pairplot(data)
data.Sales=np.log(data.Sales)
```



In [7]:

```
features=data.columns[0:2]
target=data.columns[-1]
x=data[features].values
y=data[target].values
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=17)
print("The dimenision of x_train is {}".format(x_train.shape))
print("The dimenision of x_test is {}".format(x_test.shape))
```

```
The dimenision of x_train is (140, 2)
The dimenision of x_test is (60, 2)
```

In [8]:

```
lr=LinearRegression()
lr.fit(x_train,y_train)
actual=y_test
train_score_lr=lr.score(x_train,y_train)
test_score_lr=lr.score(x_test,y_test)
print("\n LinearRegression model \n")
print("The train score for lr model is {}".format(train_score_lr))
print("The test score for lr model is {}".format(test_score_lr))
```

```
 LinearRegression model

The train score for lr model is 1.0
The test score for lr model is 1.0
```

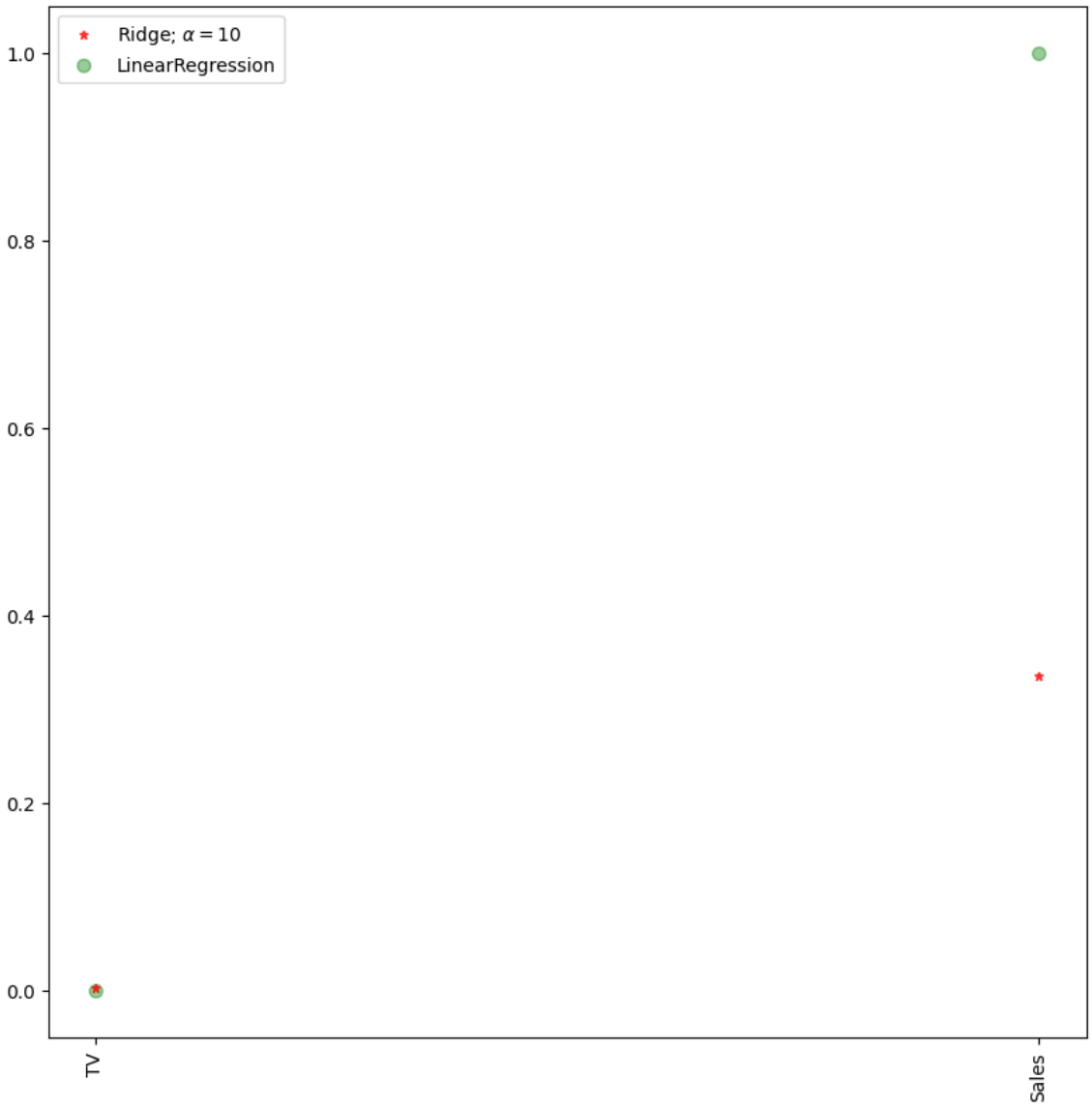
In [9]:

```
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("\nRidge Model:\n")
print("the train score for ridgemodel is {}".format(train_score_ridge))
print("the test score for ridgemodel is {}".format(train_score_ridge))
```

```
Ridge Model:

the train score for ridgemodel is 0.9053057322202976
the test score for ridgemodel is 0.9053057322202976
```

In [10]:

```
plt.figure(figsize=(10,10))
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,colo
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,color='gre
plt.xticks(rotation=90)
plt.legend()
plt.show()
```



In [11]:

```
print("\n Lasso 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 is {}".format(train_score_ls))
print("the test score for is {}".format(train_score_ls))
```

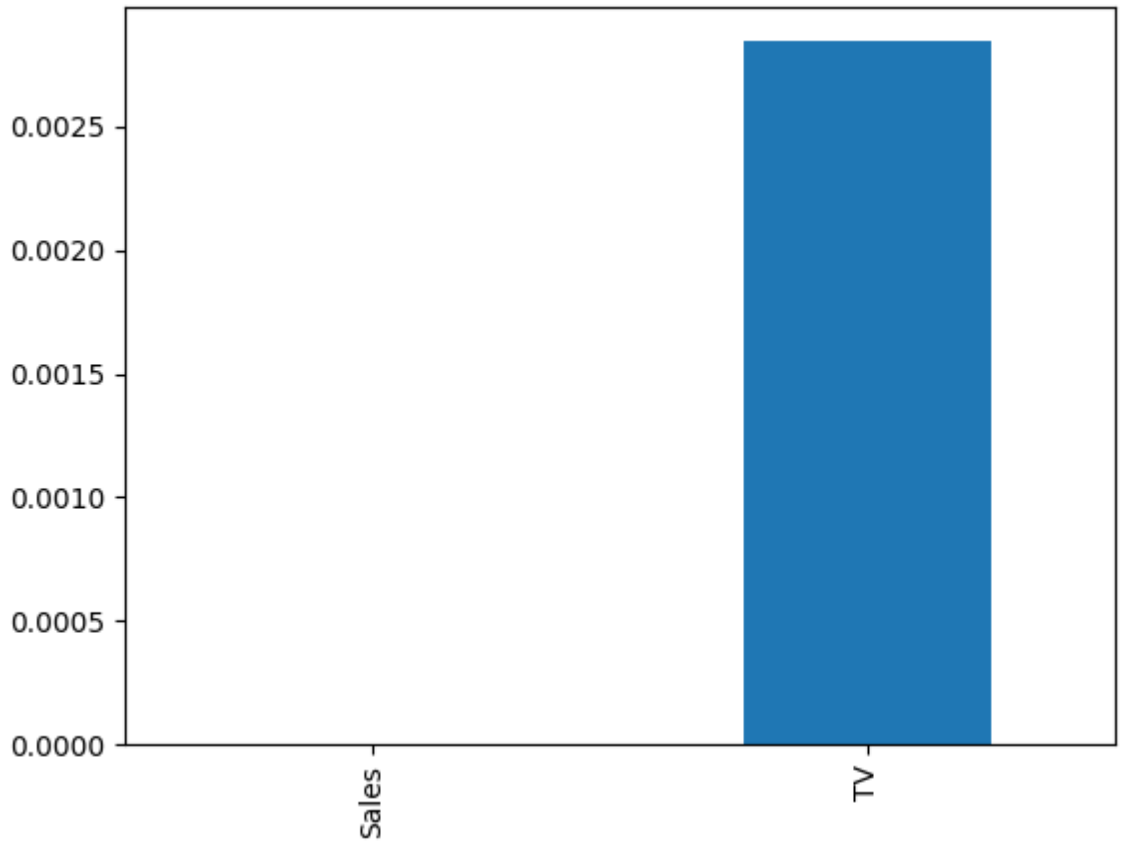
```
 Lasso model:

the train score for is 0.7082098077509238
the test score for is 0.7082098077509238
```

In [12]:

```
pd.Series(lasso.coef_,features).sort_values(ascending=True).plot(kind="bar")
```

Out[12]:

```
<Axes: >
```



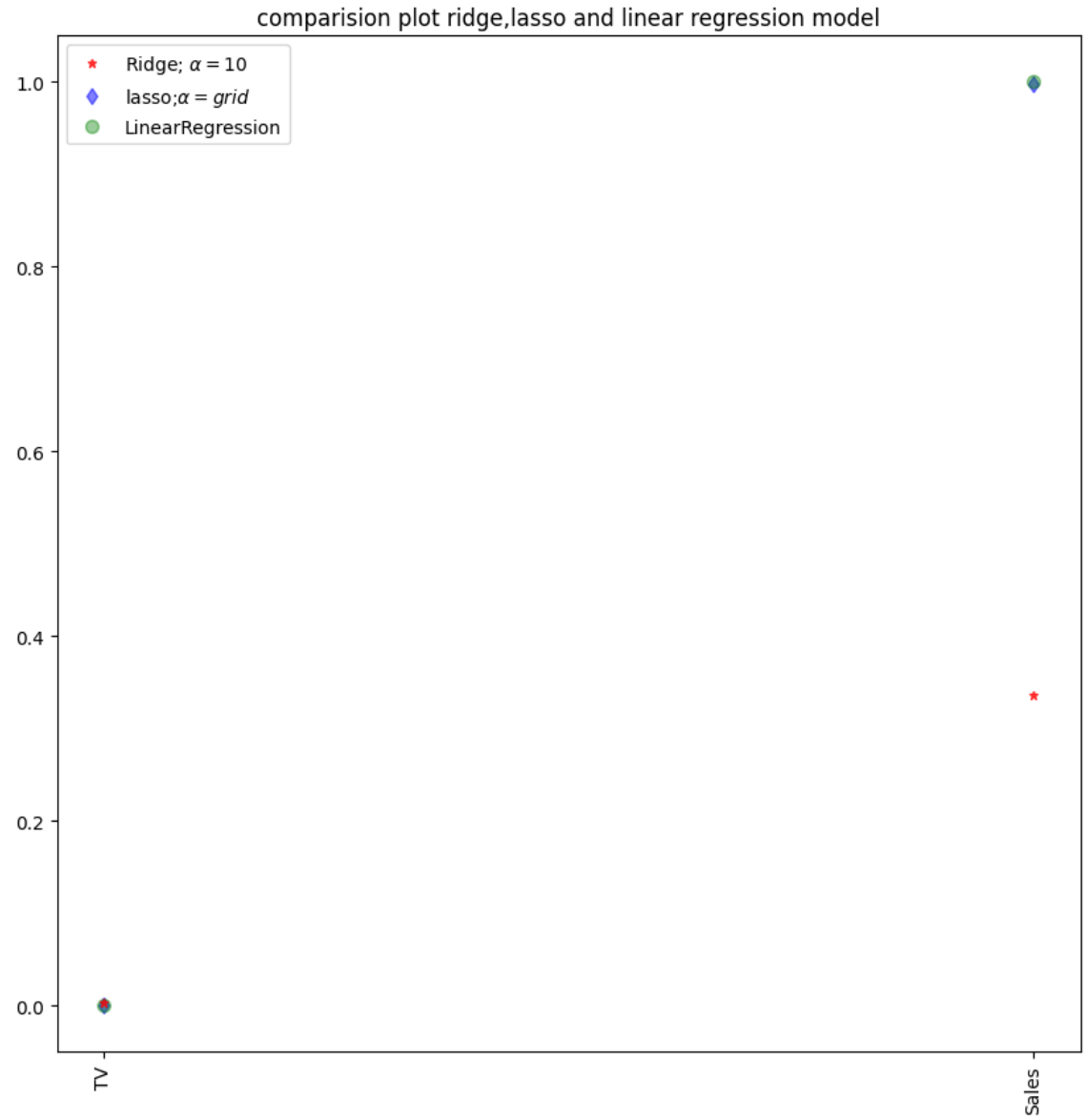
In [13]:

```
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_train,y_train
print(lasso_cv.score(x_train,y_train))
print(lasso_cv.score(x_test,y_test))
```

```
0.9999983462117727
0.9999973856137633
```

In [14]:

```
plt.figure(figsize=(10,10))
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,colo
plt.plot(lasso_cv.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,color='blue',
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,color='gre
plt.xticks(rotation=90)
plt.legend()
plt.title("comparision plot ridge,lasso and linear regression model")
plt.show()
```



In [15]:

```
from sklearn.linear_model import RidgeCV
ridge_cv=RidgeCV(alphas=[0.0001,0.001,0.01,0.1,1,10]).fit(x_train,y_train)
print("the train score for ridgemodel is {}".format(ridge_cv.score(x_train,y_train)))
print("the test score for ridgemodel is {}".format(ridge_cv.score(x_train,y_train)))
```

```
the train score for ridgemodel is 0.9999999998794061
the test score for ridgemodel is 0.9999999998794061
```

In [16]:

```
from sklearn.linear_model import ElasticNet
regr=ElasticNet()
regr.fit(x,y)
print(regr.coef_)
print(regr.intercept_)
```

```
[0.00417976 0.        ]
2.0263839193110043
```

In [17]:

```
y_pred_elastic=regr.predict(x_train)
```

In [18]:

```
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 0.03628705093551366
```

In [ ]:

```

```

In [ ]:

```

```