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

In [2]:
```
df=pd.read_csv(r"C:\Users\pappu\Downloads\fiat500_VehicleSelection_Dataset.csv
df
```

Out[2]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon
0	1	lounge	51	882	25000	1	44.907242	8.611560
1	2	pop	51	1186	32500	1	45.666359	12.241890
2	3	sport	74	4658	142228	1	45.503300	11.417840
3	4	lounge	51	2739	160000	1	40.633171	17.634609
4	5	pop	73	3074	106880	1	41.903221	12.495650
...	...	...	...	...	...	...	...	...
1533	1534	sport	51	3712	115280	1	45.069679	7.704920
1534	1535	lounge	74	3835	112000	1	45.845692	8.666870
1535	1536	pop	51	2223	60457	1	45.481541	9.413480
1536	1537	lounge	51	2557	80750	1	45.000702	7.682270
1537	1538	pop	51	1766	54276	1	40.323410	17.568270

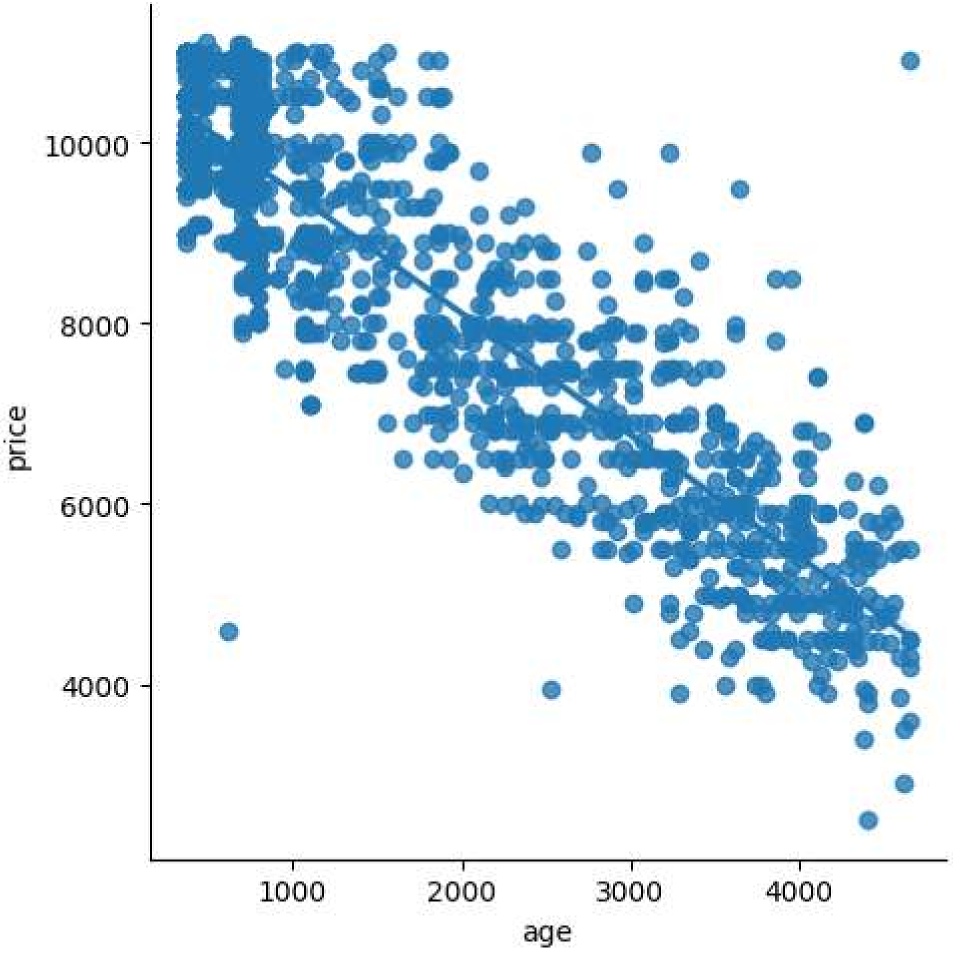
1538 rows × 9 columns

```
In [3]: df=df[['age_in_days','price']]
        df.columns=['age','price']
        df.head(10)
```

```
Out[3]:
           age   price
        0   882   8900
        1  1186   8800
        2  4658   4200
        3  2739   6000
        4  3074   5700
        5  3623   7900
        6   731  10750
        7  1521   9190
        8  4049   5600
        9  3653   6000
```

```
In [4]: sns.lmplot(x='age',y='price',data=df)
```

```
Out[4]: <seaborn.axisgrid.FacetGrid at 0x219f8ff90f0>
```



In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1538 entries, 0 to 1537
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   age     1538 non-null   int64
 1   price   1538 non-null   int64
dtypes: int64(2)
memory usage: 24.2 KB
```

In [6]:
```
x=df[['age']]
y=df['price']
```

In [7]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [8]:
```
lr=LinearRegression()
```

In [9]:
```
lr.fit(x_train,y_train)
```

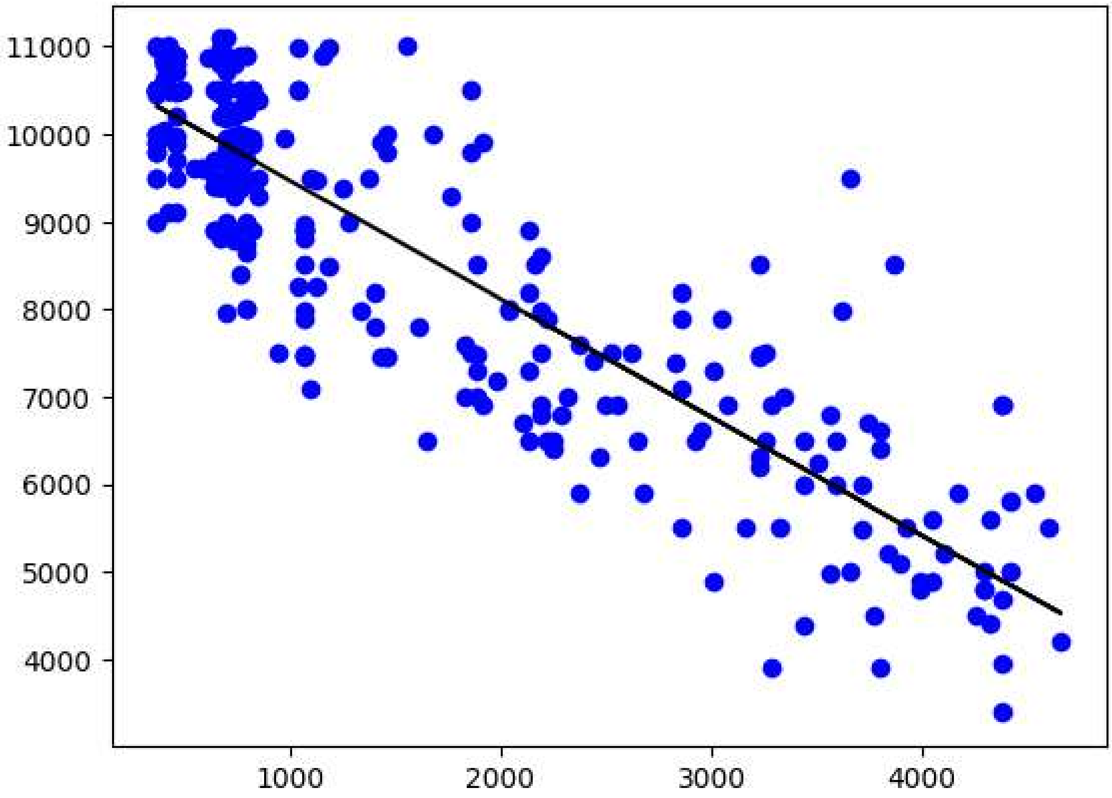
Out[9]:
```
▾ LinearRegression
LinearRegression()
```

In [10]:
```
lr.score(x_test,y_test)
```

Out[10]:
```
0.7682404327577492
```

In [11]:

```
y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



Ridge Regression

In [12]:

```
from sklearn.linear_model import Ridge,RidgeCV
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
```

In [13]:

```
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("Train score of Ridge model is {}".format(train_score_ridge))
print("Test score of Ridge model is {}".format(test_score_ridge))
```

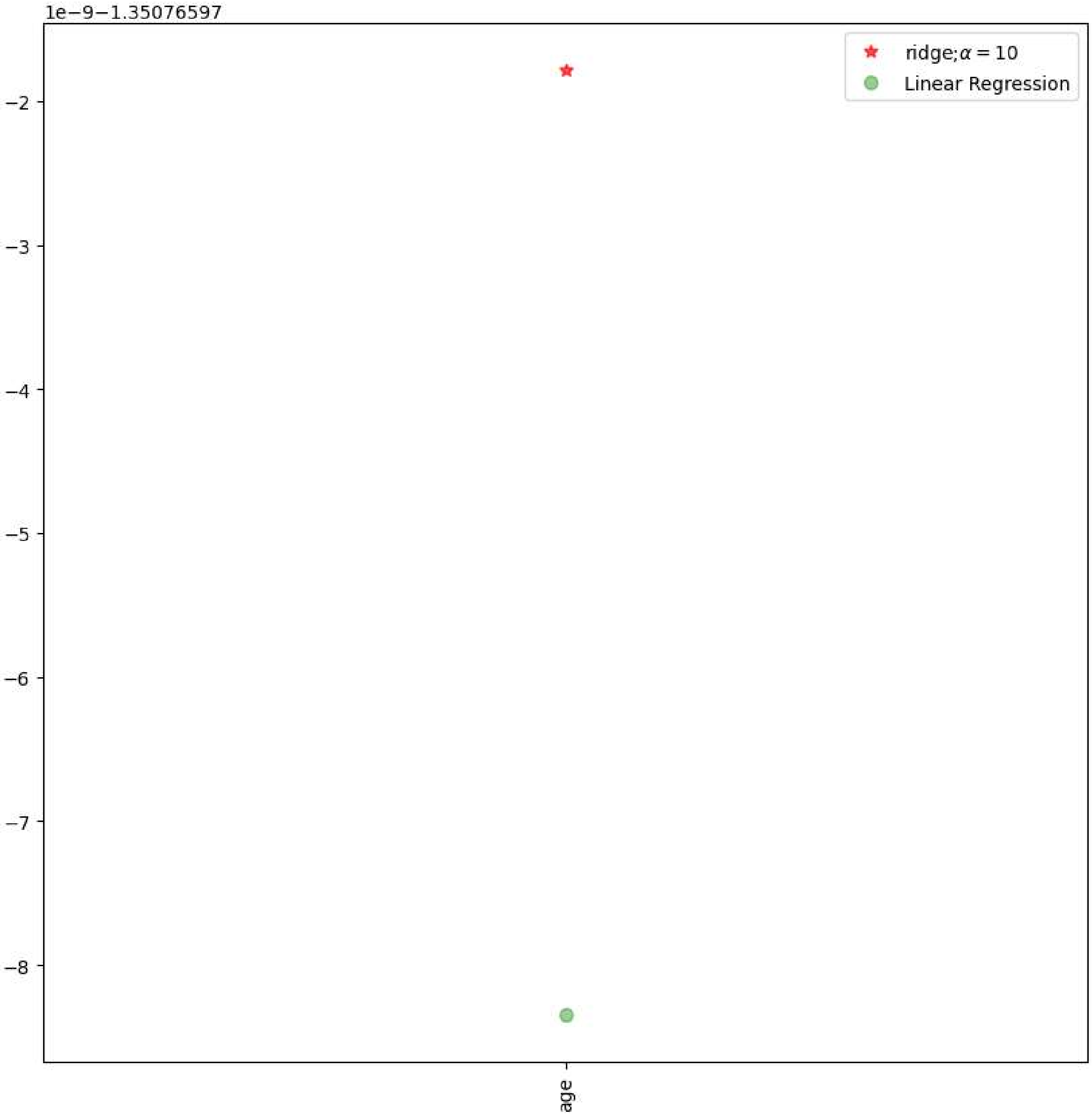
```

RidgeModel:
Train score of Ridge model is 0.8050178332327796
Test score of Ridge model is 0.7682404329678096
```

In [14]:

```
features=['age']
target=['price']
```

In [15]:

```
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

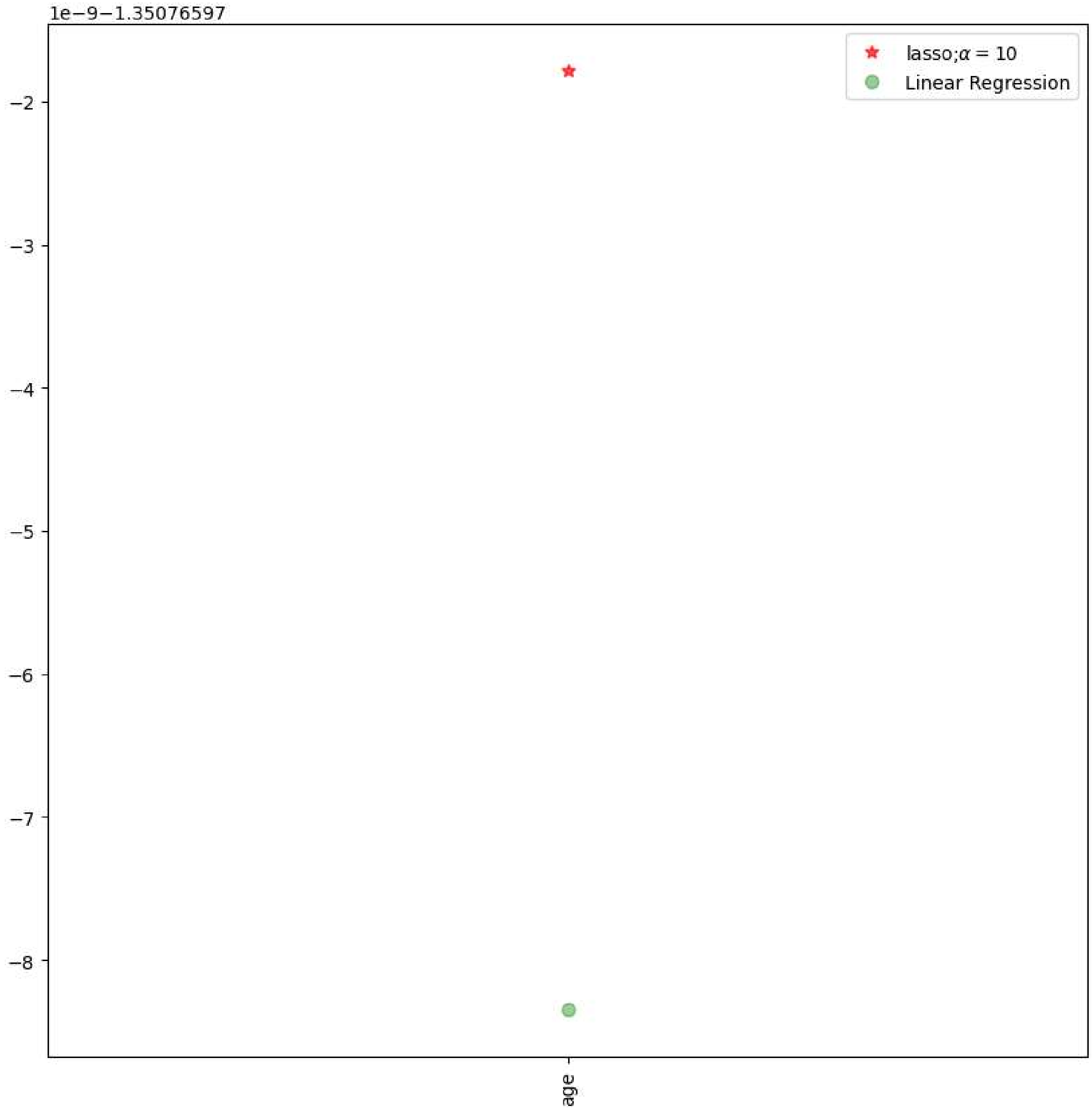
In [16]:

```
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("Train score of lasso model is {}".format(train_score_lasso))
print("Test score of lasso model is {}".format(test_score_lasso))
```

```

LassoModel:
Train score of lasso model is 0.8050178332327796
Test score of lasso model is 0.7682404329678096
```

In [17]:

```
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()
```



In [18]:

```
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))
```

```
0.8050178332170441
0.7682406240217069
```

Elastic Net

In [19]:

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

```
[-1.34392217]
10794.7931859617
```

In [20]:

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

In [21]:

```
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 740216.9493362805
```

In [ ]: