In [1]:

```python
import pandas as pd
import numpy as np
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]:

```python
df=pd.read_csv(r"C:\Users\HP\Downloads\USA_Housing.csv")
df
```

Out[2]:

| | Avg. Area Income | Avg. Area House Age | Avg. Area Number of Rooms | Avg. Area Number of Bedrooms | Area Population | Price | |
|---|---|---|---|---|---|---|---|
| **0** | 79545.458574 | 5.682861 | 7.009188 | 4.09 | 23086.800503 | 1.059034e+06 | 208 Michael 674\nLaur |
| **1** | 79248.642455 | 6.002900 | 6.730821 | 3.09 | 40173.072174 | 1.505891e+06 | 188 Johns Suite C Kathl |
| **2** | 61287.067179 | 5.865890 | 8.512727 | 5.13 | 36882.159400 | 1.058988e+06 | 9127 Stravenue\nDa W |
| **3** | 63345.240046 | 7.188236 | 5.586729 | 3.26 | 34310.242831 | 1.260617e+06 | USS Barnett\ |
| **4** | 59982.197226 | 5.040555 | 7.839388 | 4.23 | 26354.109472 | 6.309435e+05 | USNS Raymc |
| **...** | ... | ... | ... | ... | ... | ... | |
| **4995** | 60567.944140 | 7.830362 | 6.137356 | 3.46 | 22837.361035 | 1.060194e+06 | USNS Willia AP 30 |
| **4996** | 78491.275435 | 6.999135 | 6.576763 | 4.02 | 25616.115489 | 1.482618e+06 | PSC 8489\nAPO A |
| **4997** | 63390.686886 | 7.250591 | 4.805081 | 2.13 | 33266.145490 | 1.030730e+06 | 4215 Trac Suite 076\nJo |
| **4998** | 68001.331235 | 5.534388 | 7.130144 | 5.44 | 42625.620156 | 1.198657e+06 | USS Wallace\ |
| **4999** | 65510.581804 | 5.992305 | 6.792336 | 4.07 | 46501.283803 | 1.298950e+06 | 37778 Georg Apt. 509\nE |

5000 rows × 7 columns

In [3]:

```
df.head()
```

Out[3]:

| | Avg. Area Income | Avg. Area House Age | Avg. Area Number of Rooms | Avg. Area Number of Bedrooms | Area Population | Price | Ad |
|---|---|---|---|---|---|---|---|
| **0** | 79545.458574 | 5.682861 | 7.009188 | 4.09 | 23086.800503 | 1.059034e+06 | 208 Michael Ferr 674\nLaurabu 3 |
| **1** | 79248.642455 | 6.002900 | 6.730821 | 3.09 | 40173.072174 | 1.505891e+06 | 188 Johnson Suite 079\ Kathleen, |
| **2** | 61287.067179 | 5.865890 | 8.512727 | 5.13 | 36882.159400 | 1.058988e+06 | 9127 Eliz Stravenue\nDanie WI 06 |
| **3** | 63345.240046 | 7.188236 | 5.586729 | 3.26 | 34310.242831 | 1.260617e+06 | USS Barnett\nFF |
| **4** | 59982.197226 | 5.040555 | 7.839388 | 4.23 | 26354.109472 | 6.309435e+05 | USNS Raymond\ AE ( |

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Avg. Area Income              5000 non-null   float64
 1   Avg. Area House Age           5000 non-null   float64
 2   Avg. Area Number of Rooms     5000 non-null   float64
 3   Avg. Area Number of Bedrooms  5000 non-null   float64
 4   Area Population               5000 non-null   float64
 5   Price                         5000 non-null   float64
 6   Address                       5000 non-null   object 
dtypes: float64(6), object(1)
memory usage: 273.6+ KB
```

In [5]:

```
1 df.describe()
```

Out[5]:

| | Avg. Area Income | Avg. Area House Age | Avg. Area Number of Rooms | Avg. Area Number of Bedrooms | Area Population | Price |
|---|---|---|---|---|---|---|
| **count** | 5000.000000 | 5000.000000 | 5000.000000 | 5000.000000 | 5000.000000 | 5.000000e+03 |
| **mean** | 68583.108984 | 5.977222 | 6.987792 | 3.981330 | 36163.516039 | 1.232073e+06 |
| **std** | 10657.991214 | 0.991456 | 1.005833 | 1.234137 | 9925.650114 | 3.531176e+05 |
| **min** | 17796.631190 | 2.644304 | 3.236194 | 2.000000 | 172.610686 | 1.593866e+04 |
| **25%** | 61480.562388 | 5.322283 | 6.299250 | 3.140000 | 29403.928702 | 9.975771e+05 |
| **50%** | 68804.286404 | 5.970429 | 7.002902 | 4.050000 | 36199.406689 | 1.232669e+06 |
| **75%** | 75783.338666 | 6.650808 | 7.665871 | 4.490000 | 42861.290769 | 1.471210e+06 |
| **max** | 107701.748378 | 9.519088 | 10.759588 | 6.500000 | 69621.713378 | 2.469066e+06 |

In [6]:

```
1 df.columns
```

Out[6]:

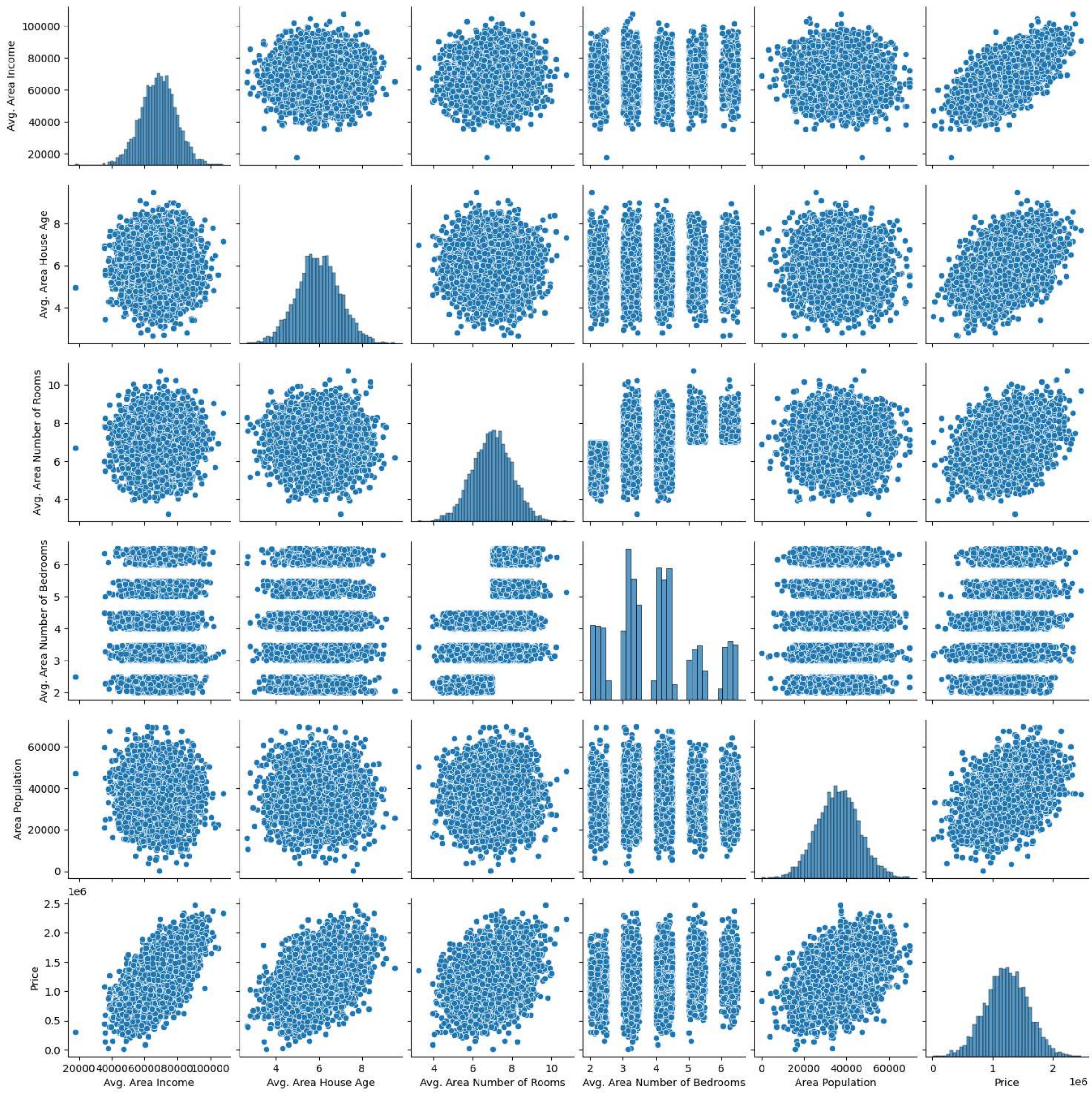
```
Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Roo
ms',
       'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Addres
s'],
      dtype='object')
```

In [7]:

```
sns.pairplot(df)
```

Out[7]:

<seaborn.axisgrid.PairGrid at 0x27622b66e50>

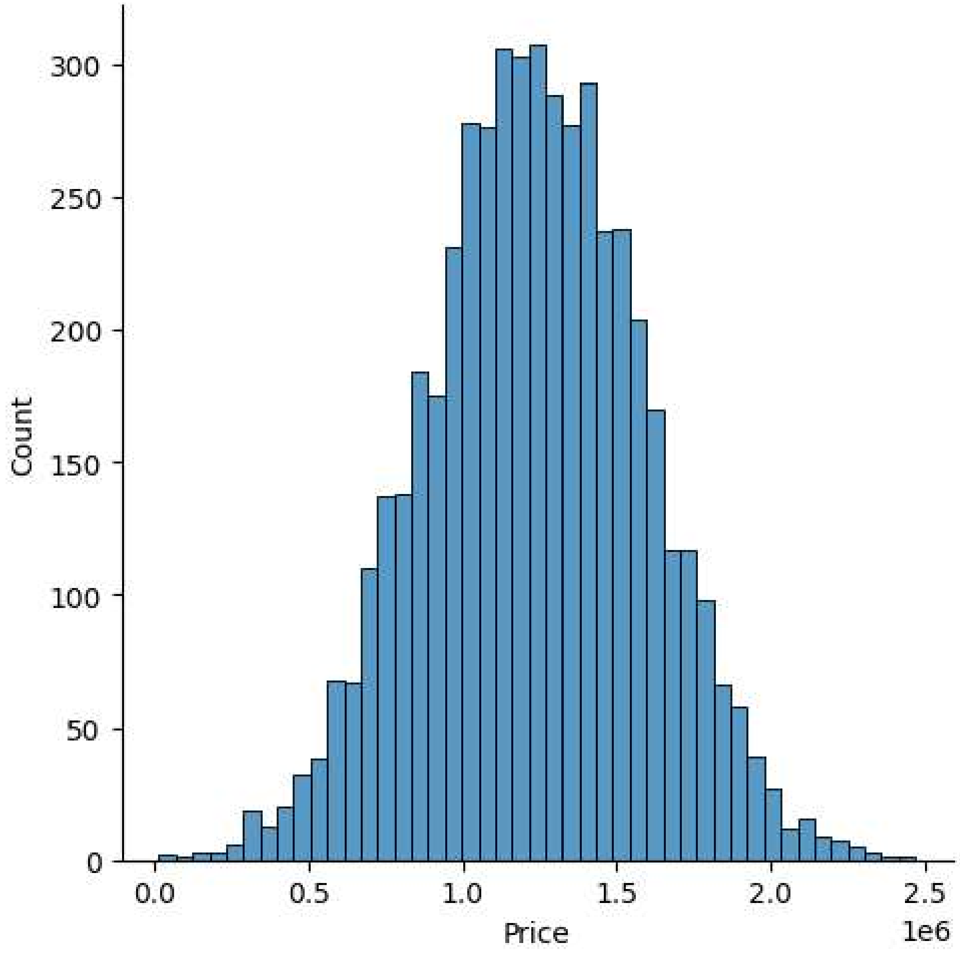


In [8]:

```
sns.displot(df['Price'])
```

Out[8]:

<seaborn.axisgrid.FacetGrid at 0x27626aea1d0>

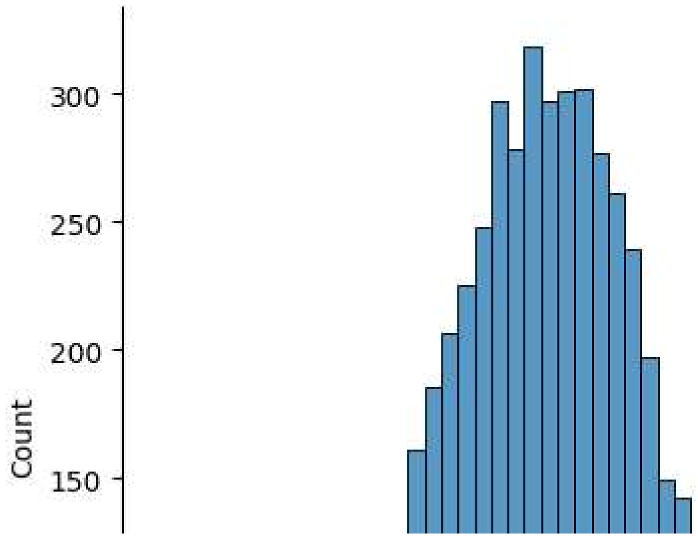



In [9]:

```
sns.displot(df['Area Population'])
```

Out[9]:

<seaborn.axisgrid.FacetGrid at 0x27627f2d8d0>



In [10]:

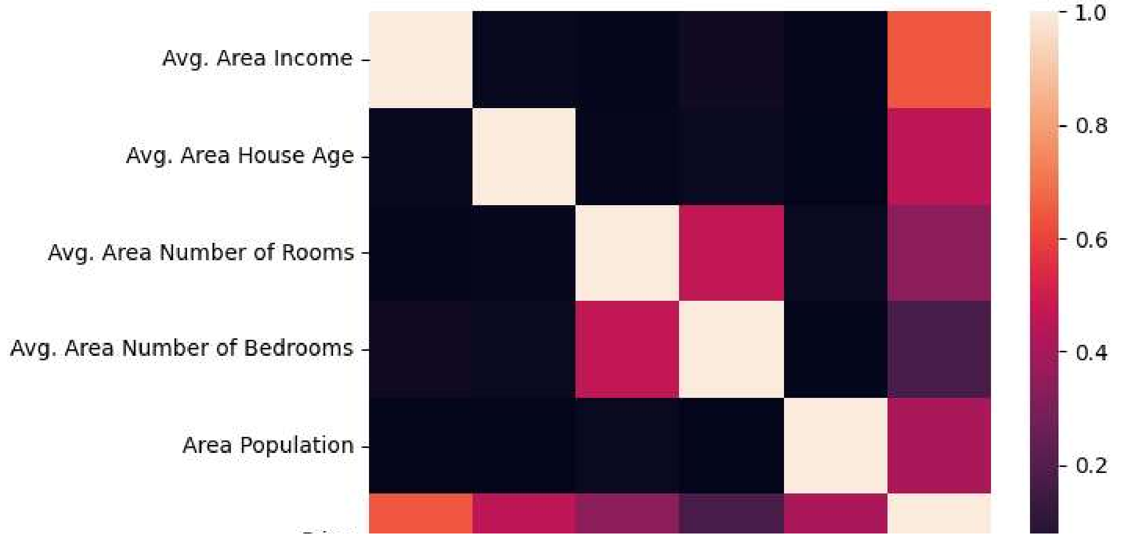
```python
Housedf=df[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
        'Avg. Area Number of Bedrooms', 'Area Population', 'Price']]
```

In [11]:

```python
sns.heatmap(Housedf.corr())
```

Out[11]:

<Axes: >




In [12]:

```python
X=Housedf[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
        'Avg. Area Number of Bedrooms', 'Area Population']]
y=df['Price']
```

In [13]:

```python
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=101)
```

In [14]:

```python
from sklearn.linear_model import LinearRegression
lm=LinearRegression()
lm.fit(X_train,y_train)
```

Out[14]:

```
▾ LinearRegression
LinearRegression()
```

In [15]:

```python
print(lm.intercept_)
```

```
-2641372.6673014304
```

In [16]:

```
coeff_df=pd.DataFrame(lm.coef_,X.columns,columns=['coefficient'])
coeff_df
```

Out[16]:

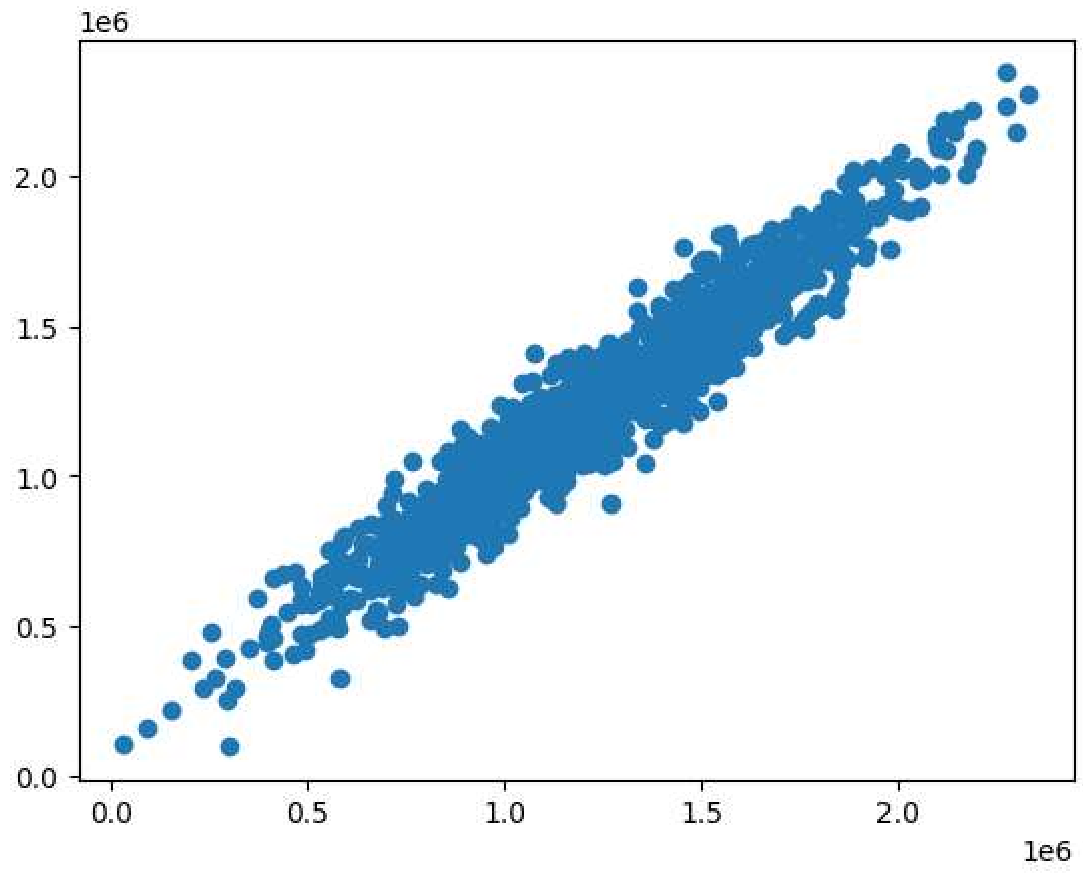
| | coefficient |
|---|---|
| **Avg. Area Income** | 21.617635 |
| **Avg. Area House Age** | 165221.119872 |
| **Avg. Area Number of Rooms** | 121405.376596 |
| **Avg. Area Number of Bedrooms** | 1318.718783 |
| **Area Population** | 15.225196 |

In [17]:

```
predictions=lm.predict(X_test)
plt.scatter(y_test,predictions)
```

Out[17]:

```
<matplotlib.collections.PathCollection at 0x27628a15310>
```

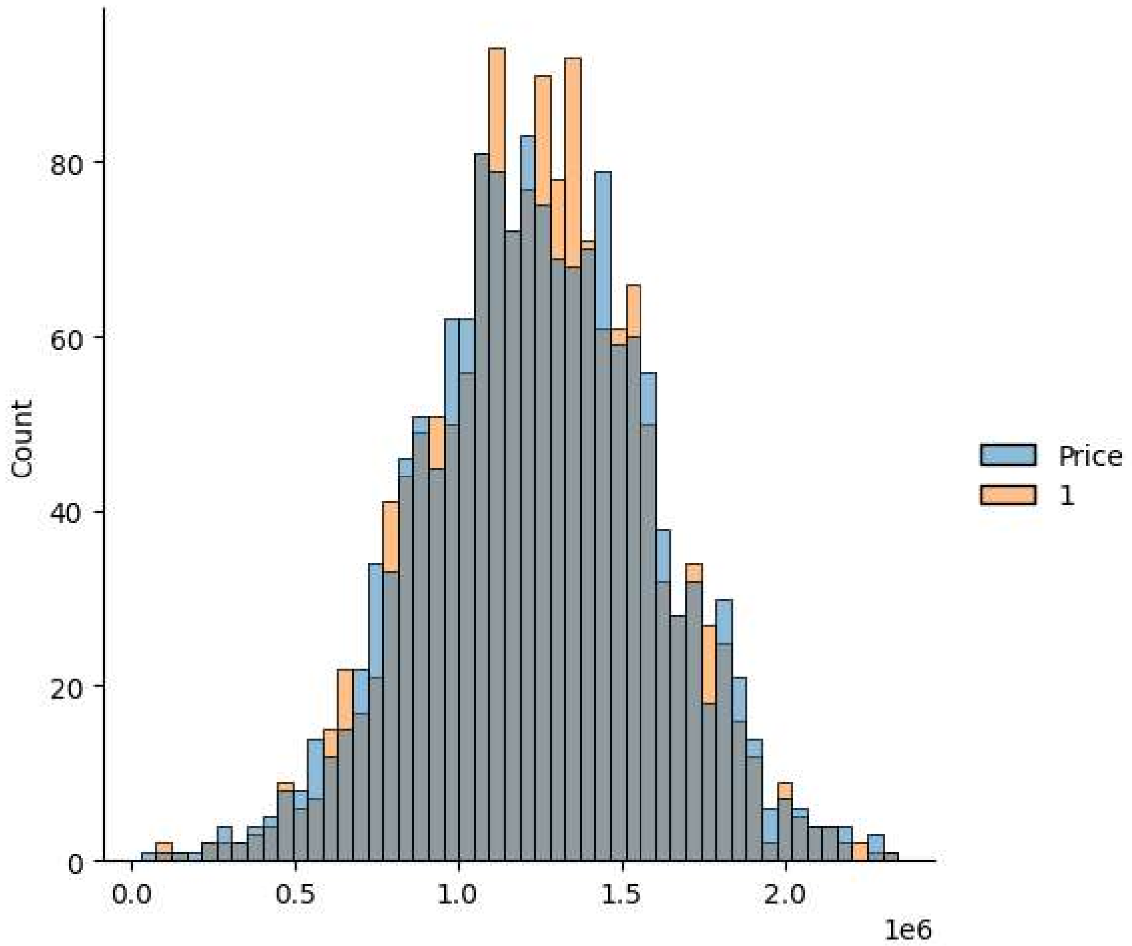


In [18]:

```
sns.displot((y_test,predictions),bins=50)
```

Out[18]:

```
<seaborn.axisgrid.FacetGrid at 0x27628a2f490>
```




In [19]:

```
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,predictions))
print('MSE:',metrics.mean_squared_error(y_test,predictions))
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_test,predictions)))
```

```
MAE: 81257.55795855941
MSE: 10169125565.897606
RMSE: 100842.08231635048
```

In [20]:

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

In [21]:

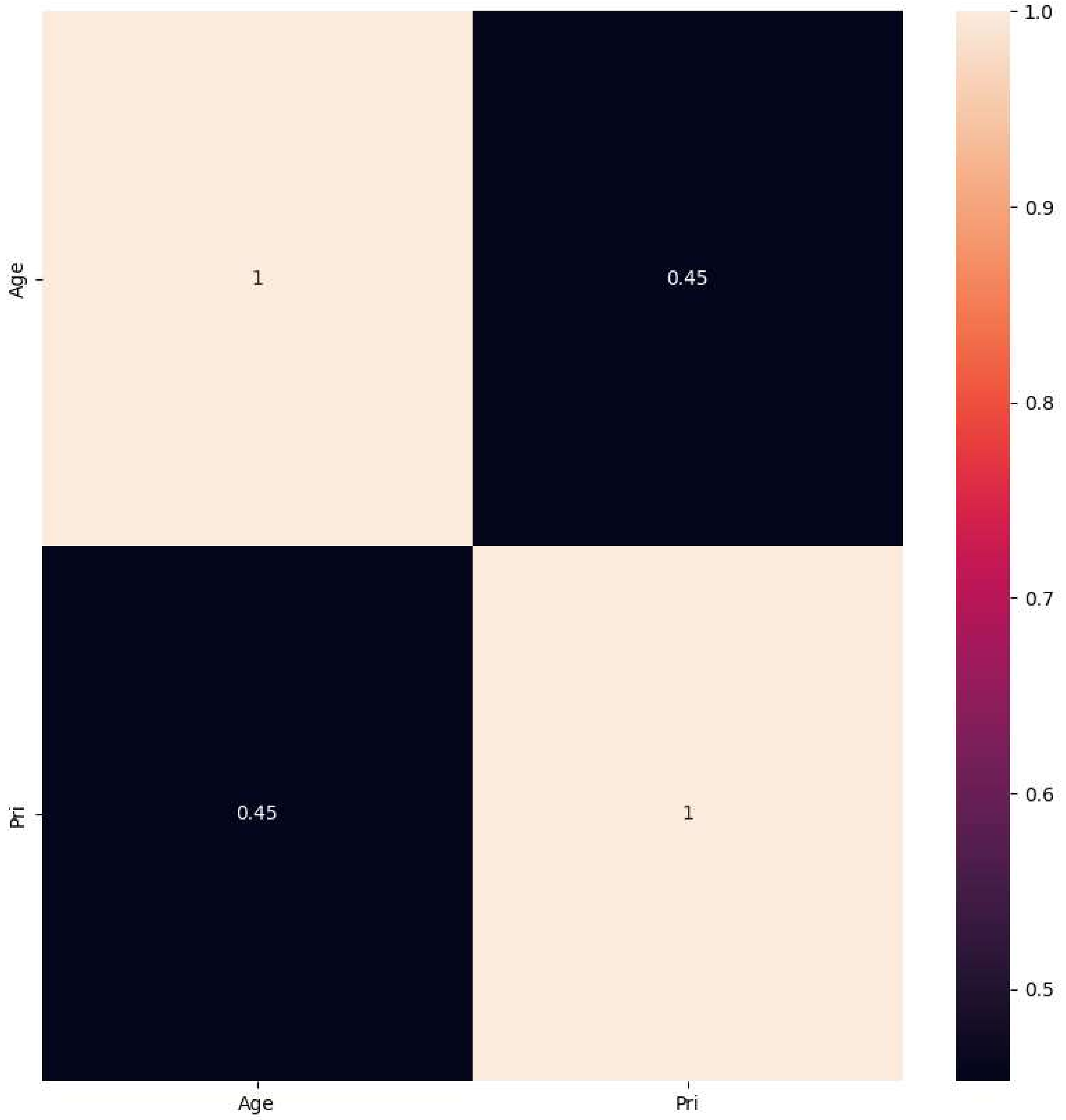
```
df=df[['Avg. Area House Age','Price']]
df.columns=['Age','Pri']
```

In [22]:

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

```

Out[22]:

<Axes: >



In [23]:

```
features = df.columns[0:2]
target = df.columns[-1]
#X and y values
X = df[features].values
y = df[target].values
#splot
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state
print("The dimension of X_train is {}".format(X_train.shape))
print("The dimension of X_test is {}".format(X_test.shape))
#Scale features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

```

```
The dimension of X_train is (3500, 2)
The dimension of X_test is (1500, 2)
```

In [24]:

```
#Model
lr = LinearRegression()
#Fit model
lr.fit(X_train, y_train)
#predict
#prediction = lr.predict(X_test)
#actual
actual = y_test
train_score_lr = lr.score(X_train, y_train)
test_score_lr = lr.score(X_test, y_test)
print("\nLinear Regression 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))
```

```

Linear Regression Model:

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

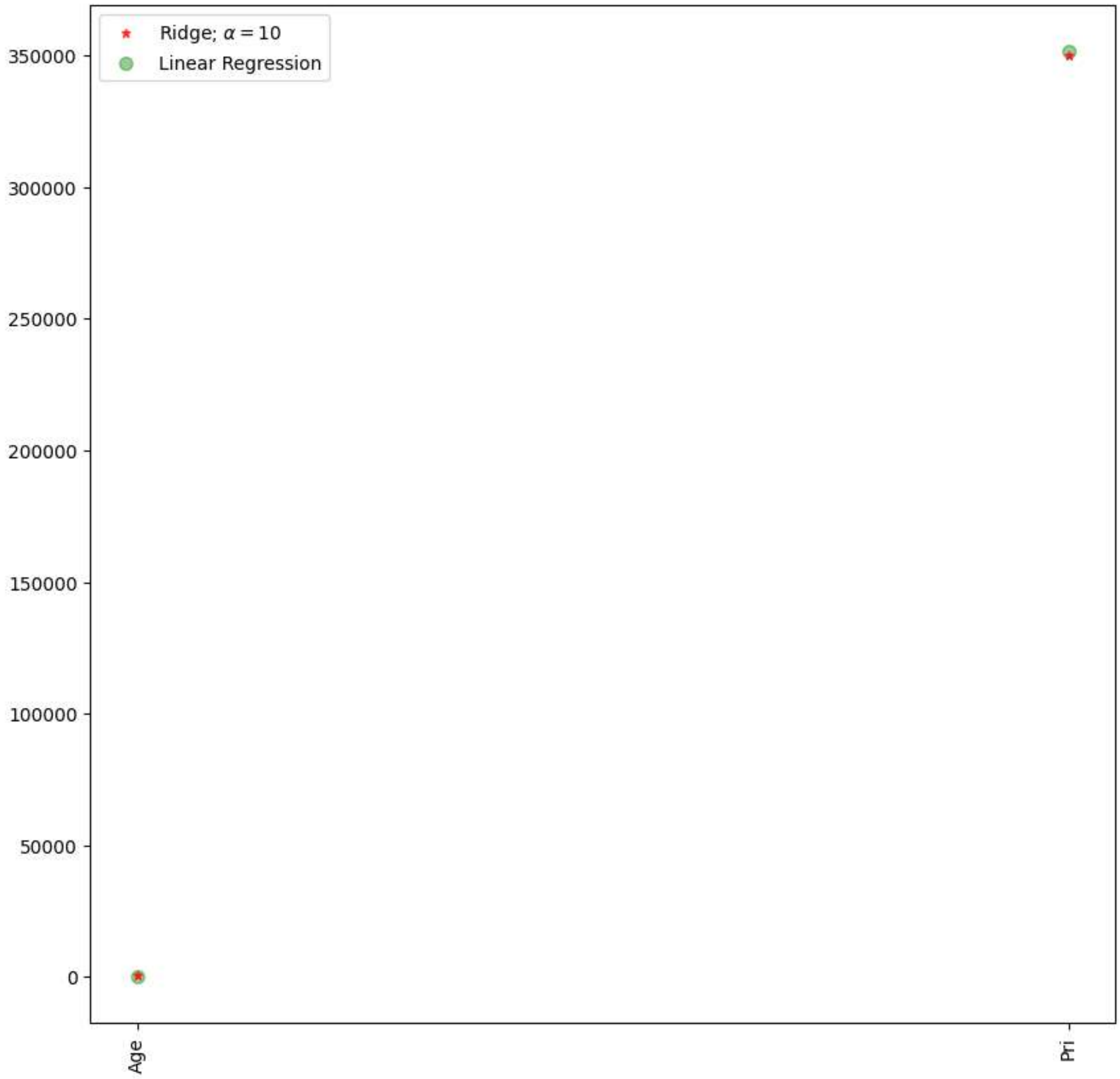
In [25]:

```
#Ridge Regression Model
ridgeReg = Ridge(alpha=10)
ridgeReg.fit(X_train,y_train)
#train and test scorefor ridge regression
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 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 0.9999898738321127
The test score for ridge model is 0.9999900217008085
```

In [26]:

```
plt.figure(figsize = (10, 10))
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,c
#plt.plot(rr100.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='
plt.xticks(rotation = 90)
plt.legend()
plt.show()
```



In [27]:

```
#Lasso regression model
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 ls model is {}".format(train_score_ls))
print("The test score for ls model is {}".format(test_score_ls))

```

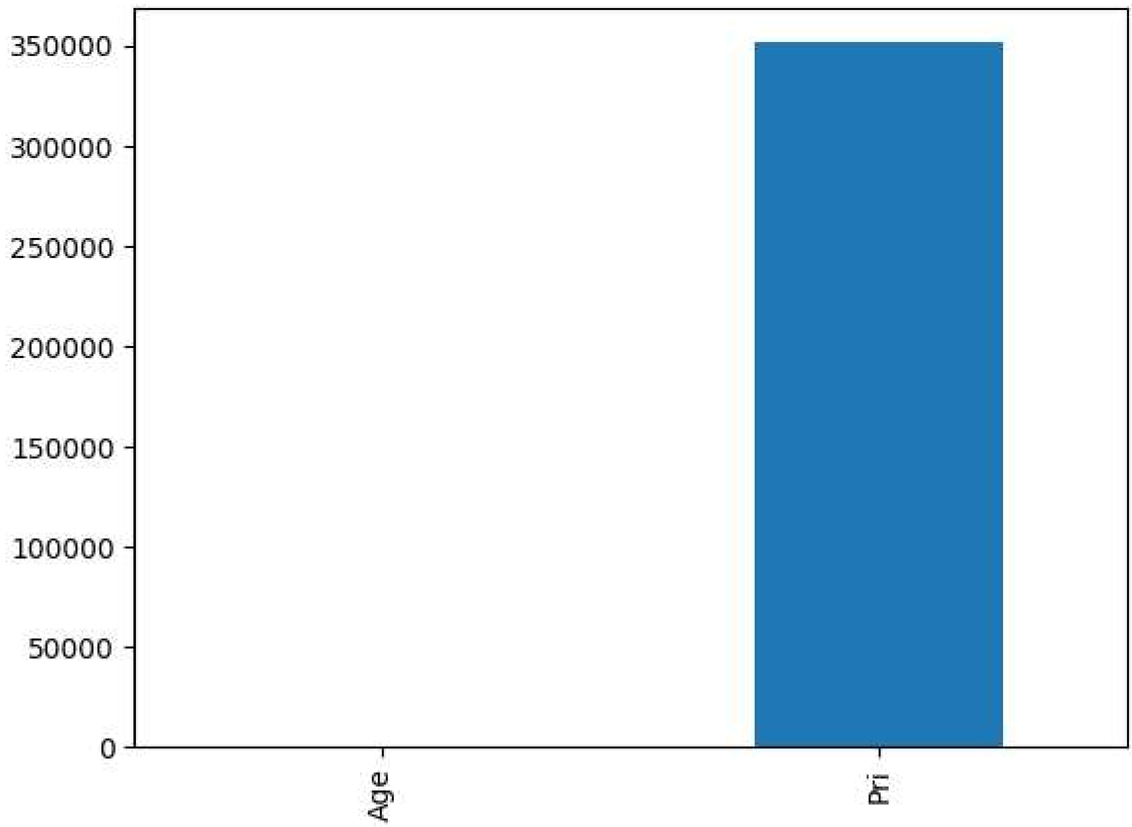
```
Lasso Model: 

The train score for ls model is 0.9999999991901091
The test score for ls model is 0.9999999991882496
```

In [28]:

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

Out[28]:

```
<Axes: >
```



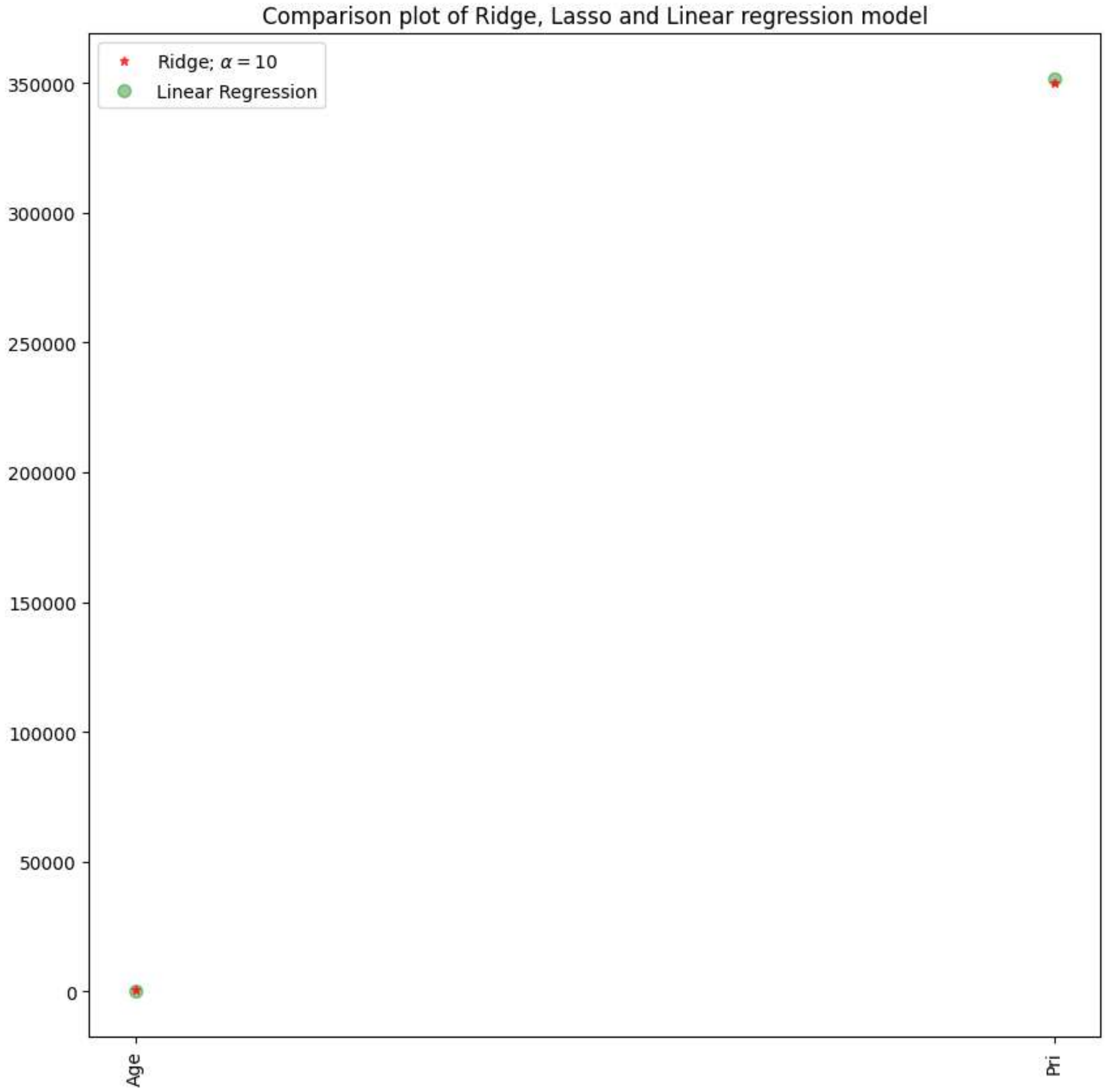
In [29]:

```
#Using the linear CV model
from sklearn.linear_model import LassoCV
#Lasso Cross validation
lasso_cv = LassoCV(alphas = [0.0001, 0.001,0.01, 0.1, 1, 10], random_state=0).fit(X_t
#score
print(lasso_cv.score(X_train, y_train))
print(lasso_cv.score(X_test, y_test))

```

```
0.9999999999723694
0.999999999973398
```

In [30]:

```
#plot size
plt.figure(figsize = (10, 10))
#add plot for ridge regression
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,c
#add plot for lasso regression
#plt.plot(lasso_cv.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,color='bl
#add plot for linear model
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,color='
#rotate axis
plt.xticks(rotation = 90)
plt.legend()
plt.title("Comparison plot of Ridge, Lasso and Linear regression model")
plt.show()

```



In [31]:

```
#Using the linear CV model
from sklearn.linear_model import RidgeCV
#Ridge Cross validation
ridge_cv = RidgeCV(alphas = [0.0001, 0.001,0.01, 0.1, 1, 10]).fit(X_train, y_train)
#score
print("The train score for ridge model is {}".format(ridge_cv.score(X_train, y_train)
print("The train score for ridge model is {}".format(ridge_cv.score(X_test, y_test))
```

```
The train score for ridge model is 0.9999999999999982
The train score for ridge model is 0.9999999999999983
```

In [32]:

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

```
[0. 1.]
9.88272950053215e-06
```

In [33]:

```
y_pred_elastic=regr.predict(X_train)

```

In [34]:

```
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 1628866435688.2292
```

In [ ]:

```

```