In [1]:
```
import pandas as pd
```

In [2]:
```
df=pd.read_csv("f:/dataset/regression/california_house_price.csv")
df
```

Out[2]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price	Unnamed: 6
0	79545.45857	5.682861	7.009188	4.09	23086.80050	1.059034e+06	NaN
1	79248.64245	6.002900	6.730821	3.09	40173.07217	1.505891e+06	NaN
2	61287.06718	5.865890	8.512727	5.13	36882.15940	1.058988e+06	NaN
3	63345.24005	7.188236	5.586729	3.26	34310.24283	1.260617e+06	NaN
4	59982.19723	5.040555	7.839388	4.23	26354.10947	6.309435e+05	NaN
...	...	...	...	...	...	...	...
4995	60567.94414	7.830362	6.137356	3.46	22837.36103	1.060194e+06	NaN
4996	78491.27543	6.999135	6.576763	4.02	25616.11549	1.482618e+06	NaN
4997	63390.68689	7.250591	4.805081	2.13	33266.14549	1.030730e+06	NaN
4998	68001.33124	5.534388	7.130144	5.44	42625.62016	1.198657e+06	NaN
4999	65510.58180	5.992305	6.792336	4.07	46501.28380	1.298950e+06	NaN

5000 rows × 7 columns

In [3]:
```
df.drop('Unnamed: 6',axis=1,inplace=True)
```

In [4]:
```
df
```

Out[4]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
0	79545.45857	5.682861	7.009188	4.09	23086.80050	1.059034e+06
1	79248.64245	6.002900	6.730821	3.09	40173.07217	1.505891e+06
2	61287.06718	5.865890	8.512727	5.13	36882.15940	1.058988e+06
3	63345.24005	7.188236	5.586729	3.26	34310.24283	1.260617e+06
4	59982.19723	5.040555	7.839388	4.23	26354.10947	6.309435e+05
...	...	...	...	...	...	...
4995	60567.94414	7.830362	6.137356	3.46	22837.36103	1.060194e+06
4996	78491.27543	6.999135	6.576763	4.02	25616.11549	1.482618e+06
4997	63390.68689	7.250591	4.805081	2.13	33266.14549	1.030730e+06
4998	68001.33124	5.534388	7.130144	5.44	42625.62016	1.198657e+06
4999	65510.58180	5.992305	6.792336	4.07	46501.28380	1.298950e+06

5000 rows × 6 columns

In [5]:
```
from sklearn.model_selection import train_test_split
```

In [7]:
```
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=10)
```

In [13]:
```
from sklearn.preprocessing import StandardScaler,PolynomialFeatures
from sklearn.linear_model import LinearRegression,SGDRegressor
```

In [12]:
```
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)

model=LinearRegression()
model.fit(X_train_new,y_train)
print("Train:",model.score(X_train_new,y_train))
print("Test:",model.score(X_test_new,y_test))

model=SGDRegressor()
model.fit(X_train_new,y_train)
print("Train:",model.score(X_train_new,y_train))
print("Test:",model.score(X_test_new,y_test))
```

```
Train: 0.9180902378305174
Test: 0.9175905242475938
Train: 0.918065662263444
Test: 0.9176747199274476
```

In [25]:
```
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=10)

pf=PolynomialFeatures(degree=7)
X_train=pf.fit_transform(X_train)
X_test=pf.transform(X_test)

sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
print("----by Linear Regression----")
model=LinearRegression()
model.fit(X_train_new,y_train)
print("Train:",model.score(X_train_new,y_train))
print("Test:",model.score(X_test_new,y_test))
```

```
----by Linear Regression----
Train: 0.9346497407915338
Test: 0.6159467651008856
----by SGD----
Train: -6037893.479371959
Test: -8090783.876718583
```

In [26]:
```
df.corr()
```

Out[26]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
Avg. Area Income	1.000000	-0.002007	-0.011032	0.019788	-0.016234	0.639734
Avg. Area House Age	-0.002007	1.000000	-0.009428	0.006149	-0.018743	0.452543
Avg. Area Number of Rooms	-0.011032	-0.009428	1.000000	0.462695	0.002040	0.335664
Avg. Area Number of Bedrooms	0.019788	0.006149	0.462695	1.000000	-0.022168	0.171071

Area Population	-0.016234	-0.018743	0.002040	-0.022168	1.000000	0.408556
Price	0.639734	0.452543	0.335664	0.171071	0.408556	1.000000

In [27]:
```
df.drop('Avg. Area Number of Bedrooms',axis=1,inplace=True)
```

In [44]:
```
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=10)

pf=PolynomialFeatures(degree=7)
X_train=pf.fit_transform(X_train)
X_test=pf.transform(X_test)

print(X_train.shape)
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
print("----by Linear Regression----")
model=LinearRegression()
model.fit(X_train_new,y_train)
print("Train:",model.score(X_train_new,y_train))
print("Test:",model.score(X_test_new,y_test))
```

```
(3750, 330)
----by Linear Regression----
Train: 0.9245637625649561
Test: 0.7940360329971728
```

In [38]:
```
from sklearn.feature_selection import SelectKBest,f_regression
```

In [77]:
```
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=10)


pf=PolynomialFeatures(degree=5)
X_train=pf.fit_transform(X_train)
X_test=pf.transform(X_test)
print(X_train.shape)
skb=SelectKBest(score_func=f_regression,k=int(X_train.shape[1]*.97))
X_train=skb.fit_transform(X_train,y_train)
X_test=skb.transform(X_test)
print(X_train.shape)
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
print("----by Linear Regression----")
model=LinearRegression()
model.fit(X_train_new,y_train)
print("Train:",model.score(X_train_new,y_train))
print("Test:",model.score(X_test_new,y_test))
```

```
(3750, 126)
(3750, 122)
----by Linear Regression----
Train: 0.9200484939109539
Test: 0.9131833985635731
```

In [78]:
```
df=pd.read_csv("f:/dataset/regression/california_house_price.csv")
df.drop('Unnamed: 6',axis=1,inplace=True)
```

In [79]:
```
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=10)
```

In [93]:
```
from sklearn.linear_model import Ridge,LinearRegression,Lasso
```

In [116…
```
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)

model=LinearRegression()
model.fit(X_train_new,y_train)
print(model.coef_)

model=Ridge(alpha=0.01)
model.fit(X_train_new,y_train)
print(model.coef_)

model=Lasso(alpha=900)
model.fit(X_train_new,y_train)
print(model.coef_)
```

```
[230555.55061463 163578.16953912 121128.37064076    587.56856667
 149679.72352755]
[230554.92103    163577.73405129 121127.94724861    587.75315896
 149679.31617384]
[229630.576087   162681.86218585 120465.89865509      0.
 148751.25022972]
```

In [84]:

Out[84]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
0	79545.45857	5.682861	7.009188	4.09	23086.80050	1.059034e+06
1	79248.64245	6.002900	6.730821	3.09	40173.07217	1.505891e+06
2	61287.06718	5.865890	8.512727	5.13	36882.15940	1.058988e+06
3	63345.24005	7.188236	5.586729	3.26	34310.24283	1.260617e+06
4	59982.19723	5.040555	7.839388	4.23	26354.10947	6.309435e+05
...	...	...	...	...	...	...
4995	60567.94414	7.830362	6.137356	3.46	22837.36103	1.060194e+06
4996	78491.27543	6.999135	6.576763	4.02	25616.11549	1.482618e+06
4997	63390.68689	7.250591	4.805081	2.13	33266.14549	1.030730e+06
4998	68001.33124	5.534388	7.130144	5.44	42625.62016	1.198657e+06
4999	65510.58180	5.992305	6.792336	4.07	46501.28380	1.298950e+06

5000 rows × 6 columns

In [ ]: