In [5]:
```python
#PRACTICAL NO.8
#Aim: Hyperparameter tuning for lasso regression can be done in python without
import pandas as pd
df=pd.read_csv("BostonHousing.csv")
df
```

Out[5]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	med
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67	22.
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	20.
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	23.
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.48	22.
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396.90	7.88	11.

506 rows × 14 columns

In [6]:
```python
df.head(3)
```

Out[6]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	medv
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7

In [7]:
```
x=df.iloc[:,:-1]
x
```

Out[7]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33
...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.48
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396.90	7.88

506 rows × 13 columns

In [8]:
```
y=df.iloc[:,-1]
y
```

Out[8]:
```
0      24.0
1      21.6
2      34.7
3      33.4
4      36.2
       ...
501    22.4
502    20.6
503    23.9
504    22.0
505    11.9
Name: medv, Length: 506, dtype: float64
```

In [9]:
```
from sklearn.linear_model import Lasso
model=Lasso()
```

In [10]:
```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,random_state=1,test_size=0.25)
```

In [11]:
```
model.fit(xtrain,ytrain)
```

Out[11]:
```
Lasso()
```

In [12]:
```
from sklearn.model_selection import RepeatedKFold
cv=RepeatedKFold(n_splits=10,n_repeats=3,random_state=1)
```

In [13]:
```
from sklearn.metrics import r2_score
ypred=model.predict(xtest)
r2_score(ytest,ypred)
```

Out[13]:
```
0.6621980770523261
```

In [14]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x_sc=sc.fit_transform(x)
xtrain,xtest,ytrain,ytest=train_test_split(x_sc,y,random_state=1,test_size=0.2
model1=Lasso()
parms={'alpha':[0.00001,0.0001,0.001,0.01]}
from sklearn.model_selection import GridSearchCV
search=GridSearchCV(model1,parms,cv=cv)
result=search.fit(x_sc,y)
result.best_params_
```

Out[14]:
```
{'alpha': 0.01}
```

In [15]:
```
model2=Lasso(alpha=0.01)
model2.fit(xtrain,ytrain)
```

Out[15]:
```
Lasso(alpha=0.01)
```

In [16]:
```
ypred=model2.predict(xtest)
r2_score(ytest,ypred)
```

Out[16]:
```
0.7787372388293925
```

In [17]:
```
from sklearn.linear_model import Ridge
model2=Ridge()
```

In [18]:
```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,random_state=1,test_size=0.25)
```

In [19]:
```
model2.fit(xtrain,ytrain)
```

Out[19]:
```
Ridge()
```

In [20]:
```
from sklearn.model_selection import RepeatedKFold
cv=RepeatedKFold(n_splits=10,n_repeats=3,random_state=1)
```

In [21]:
```
from sklearn.metrics import r2_score
ypred=model.predict(xtest)
r2_score(ytest,ypred)
```

Out[21]:
```
0.6621980770523261
```

In [22]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x_sc=sc.fit_transform(x)
xtrain,xtest,ytrain,ytest=train_test_split(x_sc,y,random_state=1,test_size=0.2
model2=Ridge()
parms={'alpha':[0.00001,0.0001,0.001,0.01]}
from sklearn.model_selection import GridSearchCV
search=GridSearchCV(model1,parms,cv=cv)
result=search.fit(x_sc,y)
result.best_params_
```

Out[22]:
```
{'alpha': 0.01}
```

In [23]:
```
model2=Lasso(alpha=0.01)
model2.fit(xtrain,ytrain)
```

Out[23]:
```
Lasso(alpha=0.01)
```

In [24]:
```
ypred=model2.predict(xtest)
r2_score(ytest,ypred) #RIDGE REGRESSION
```

Out[24]:
```
0.7787372388293925
```

In [ ]: