Objective

The analysis tries to use lasso regression to discover the most suitable explanatory attributes that influence the prices of housing. Of these, medv is the response variable while the other 13 variables are possible predictors. At the end of the analysis, we will arrange the top five attributes based on the strength of its influence on the response (medv).

Lasso Regression:

Using the lasso regression, we can obtain the important features of the dataset, that minimizes prediction error for a quantitative response variable. The lasso does this by imposing a constraint on the model parameters that cause regression coefficients for some variables to shrink toward zero.

Boston Dataset sklearn

The sklearn Boston dataset is used wisely in regression and is famous dataset from the 1970's. There are 506 instances and 14 attributes in the dataset.

Data Set Characteristics:

:Number of Instances: 506

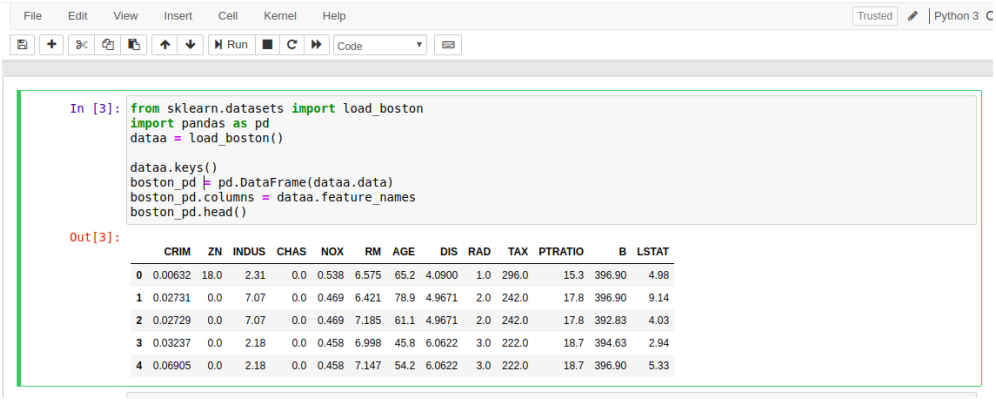
:Number of Attributes: 13 numeric/categorical predictive. Median Value (attribute 14) is usually the target.

:Attribute Information (in order):
- CRIM per capita crime rate by town
- ZN proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS proportion of non-retail business acres per town
- CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX nitric oxides concentration (parts per 10 million)
- RM average number of rooms per dwelling
- AGE proportion of owner-occupied units built prior to 1940
- DIS weighted distances to five Boston employment centres
- RAD index of accessibility to radial highways
- TAX full-value property-tax rate per $10,000
- PTRATIO pupil-teacher ratio by town
- B 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT % lower status of the population
- MEDV Median value of owner-occupied homes in $1000's

Code:

```
from sklearn.datasets import load_boston
import pandas as pd
bostona = load_boston()
bostona.keys()
boston_pd = pd.DataFrame(bostona.data)
boston_pd.columns = bostona.feature_names
boston_pd.head()
#the output/ MEDV
boston_y = pd.DataFrame(bostona.target, columns = ['MEDV'])
boston_y
import seaborn as sns
g = sns.pairplot(boston_pd)
from sklearn import linear_model
clf = linear_model.Lasso(alpha=0.1)
```

```
clf.fit(boston_pd,boston_y)
clf.coef_
import matplotlib.pyplot as plt
_=plt.plot(range(len(bostona.feature_names)),clf.coef_)
_=plt.xticks(range(len(bostona.feature_names)),bostona.feature_names,rotation=60)
_=plt.ylabel("Coefficients")
```

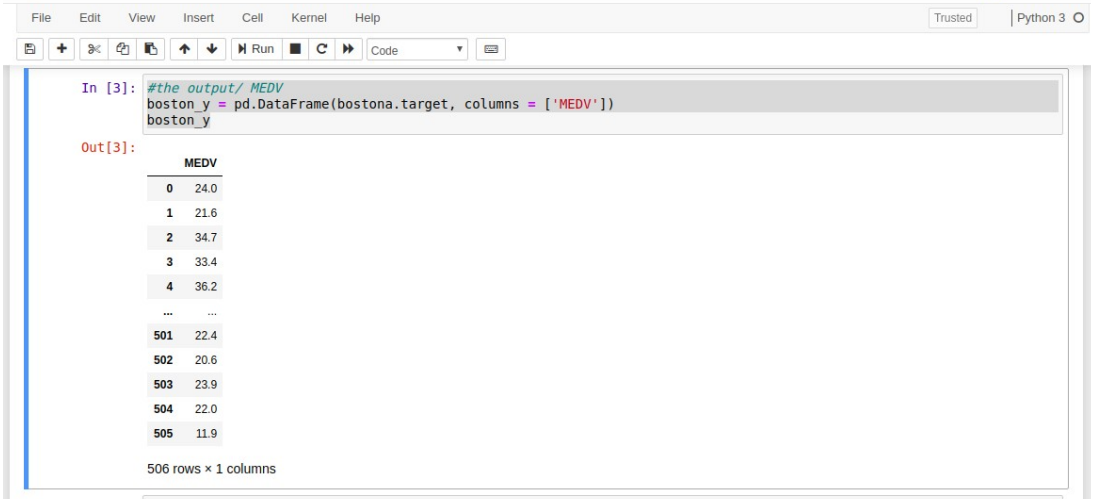


```
In [3]: from sklearn.datasets import load_boston
        import pandas as pd
        dataa = load_boston()

        dataa.keys()
        boston_pd = pd.DataFrame(dataa.data)
        boston_pd.columns = dataa.feature_names
        boston_pd.head()
```

Out[3]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33



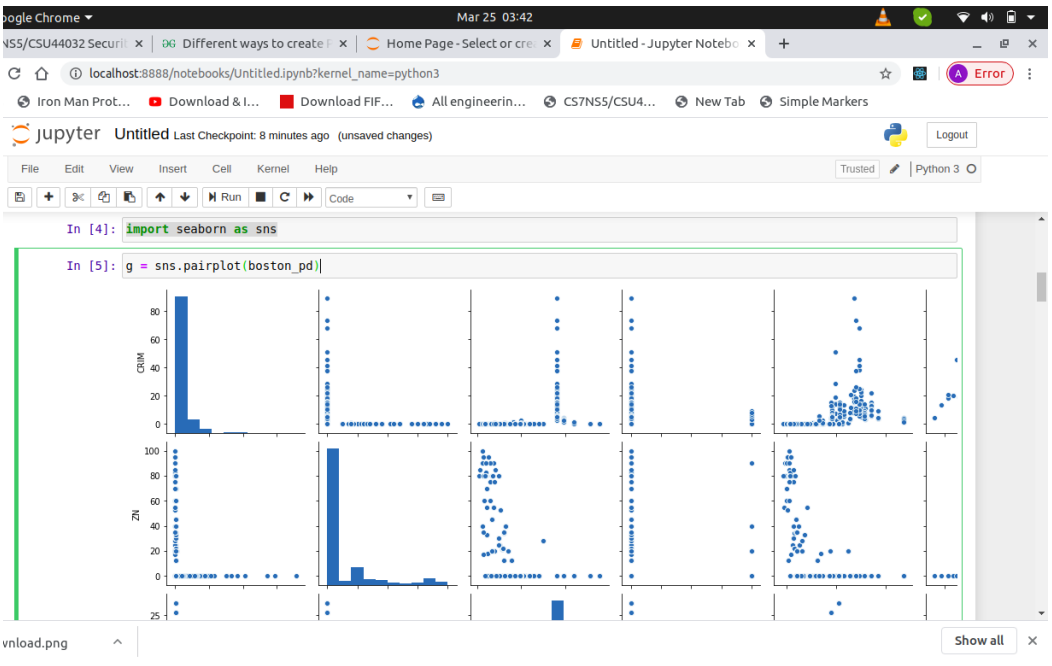
```
In [3]: #the output/ MEDV
        boston_y = pd.DataFrame(bostona.target, columns = ['MEDV'])
        boston_y
```

Out[3]:

	MEDV
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

506 rows × 1 columns

For better understanding, Here we will be using pairplot to visualize the relationship between two attributes, The pairplot function creates a grid of Axes such that each variable in data will by shared in the y-axis across a single row and in the x-axis across a single column.



File Edit View Insert Cell Kernel Help Trusted Python 3

Run Code

506 rows × 1 columns

In [4]:
```
from sklearn import linear_model
```

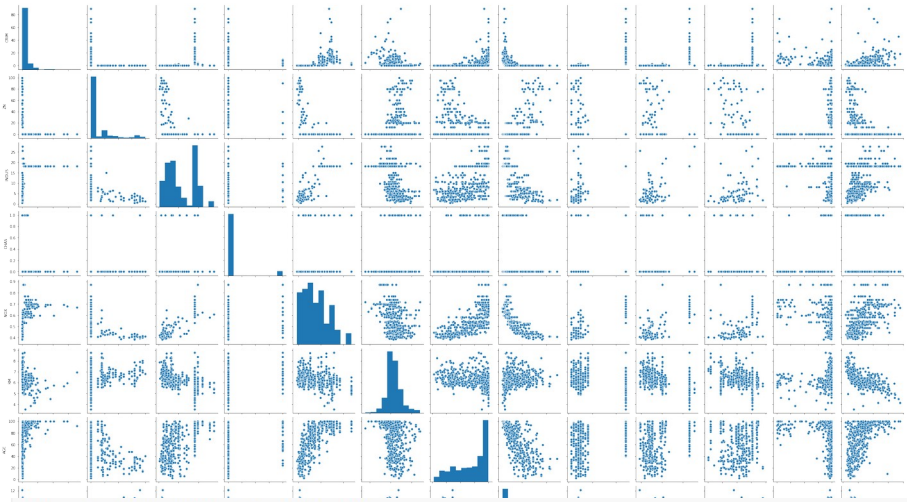
In [5]:
```
clf = linear_model.Lasso(alpha=0.1)
```

In [6]:
```
clf.fit(boston_pd,boston_y)
```

Out[6]:
```
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=1000,
      normalize=False, positive=False, precompute=False, random_state=None,
      selection='cyclic', tol=0.0001, warm_start=False)
```

In [7]:
```
clf.coef_
```

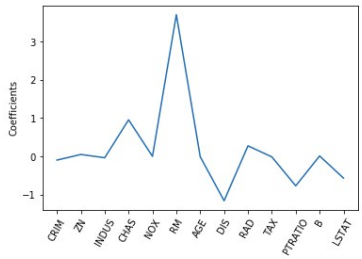
Out[7]:
```
array([-0.09789363,  0.04921111, -0.03661906,  0.95519003, -0.        ,
        3.70320175, -0.01003698, -1.16053834,  0.27470721, -0.01457017,
       -0.77065434,  0.01024917, -0.56876914])
```



File Edit View Insert Cell Kernel Help Trusted Python 3

Run Code

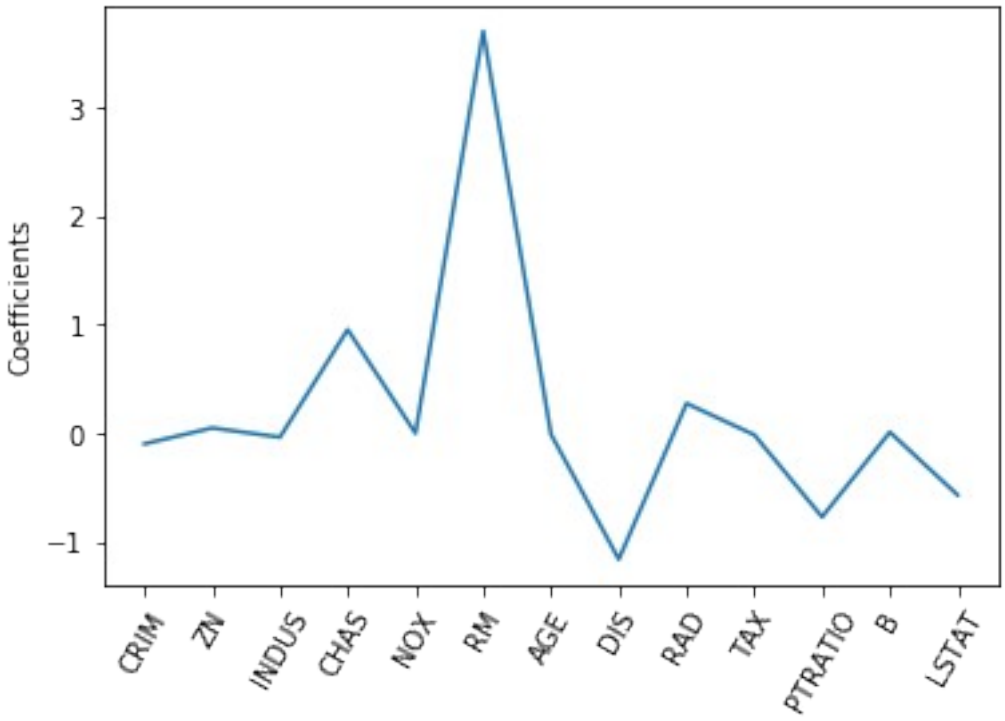
In [8]:
```
import matplotlib.pyplot as plt
_=plt.plot(range(len(bostona.feature_names)),clf.coef_)
_=plt.xticks(range(len(bostona.feature_names)),bostona.feature_names,rotation=60)
_=plt.ylabel("Coefficients")
```



In [ ]:

Conclusion

House prices also tend to be higher closer to the Charles River, and houses with more rooms are pricier.



Priority	Attributes
1	RM: average number of rooms per dwelling
2	CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
3	ZN: proportion of residential land zoned for lots over 25,000 sq.ft.
4	NOX: nitric oxides concentration (parts per 10 million)
5	CRIM per capita crime rate by town