```
In [1]: %matplotlib inline
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import scipy.stats as stats
        import sklearn
```

```
In [2]: from sklearn.datasets import load_boston
        boston=load_boston()
```

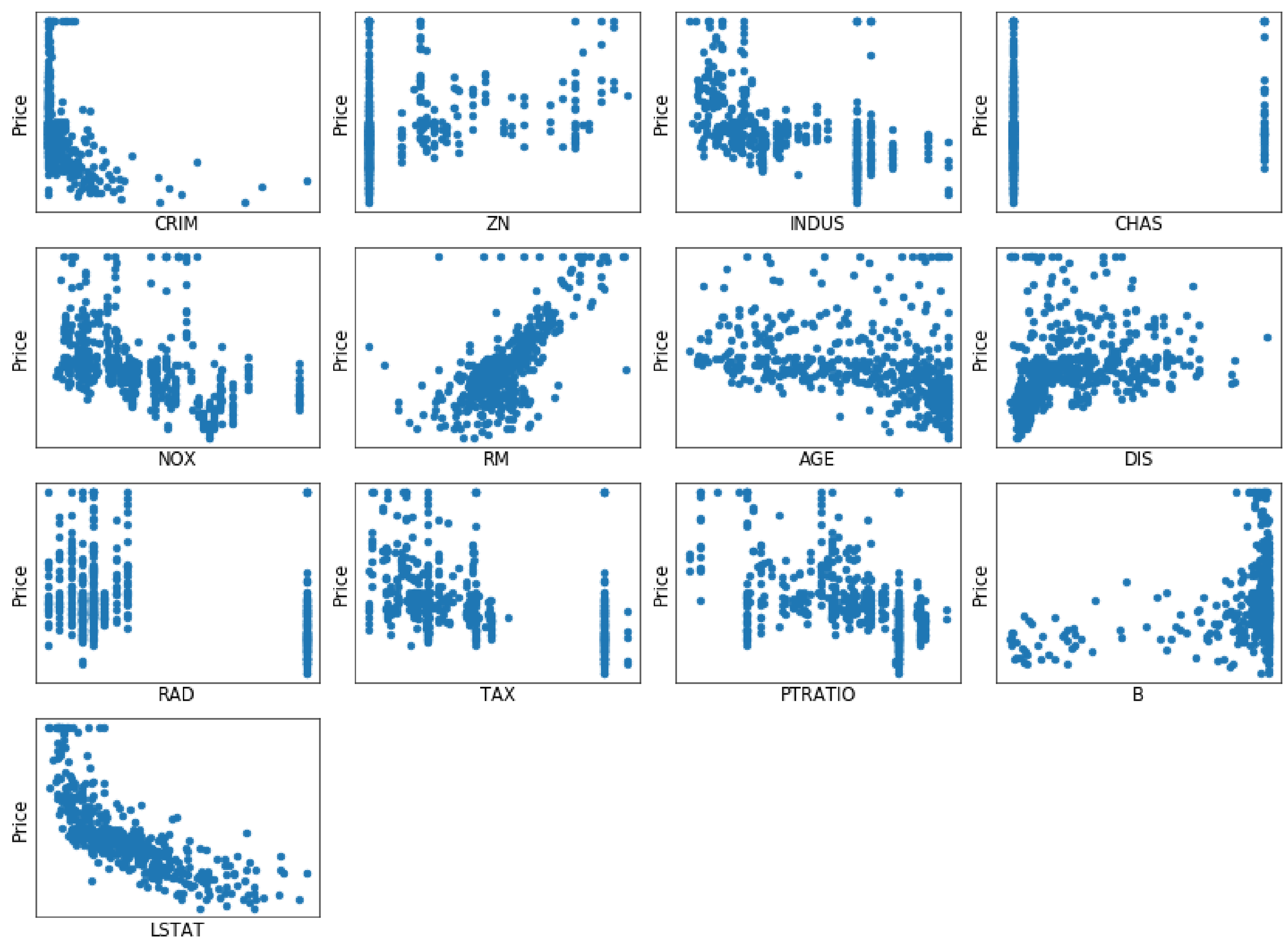
```
In [3]: boston.data.shape
Out[3]: (506, 13)
```

```
In [4]: boston.keys()
Out[4]: dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename'])
```

```
In [6]: print(boston.feature_names)

['CRIM' 'ZN' 'INDUS' 'CHAS' 'NOX' 'RM' 'AGE' 'DIS' 'RAD' 'TAX' 'PTRATIO'
 'B' 'LSTAT']
```

```
In [7]: print(boston.DESCR)

.. _boston_dataset:

Boston house prices dataset
---------------------------

**Data Set Characteristics:**

    :Number of Instances: 506

    :Number of Attributes: 13 numeric/categorical predictive. Median Value (attribute 14) is usu
ally 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

    :Missing Attribute Values: None

    :Creator: Harrison, D. and Rubinfeld, D.L.

This is a copy of UCI ML housing dataset.
https://archive.ics.uci.edu/ml/machine-learning-databases/housing/


This dataset was taken from the StatLib library which is maintained at Carnegie Mellon Universit
y.

The Boston house-price data of Harrison, D. and Rubinfeld, D.L. 'Hedonic
prices and the demand for clean air', J. Environ. Economics & Management,
vol.5, 81-102, 1978.   Used in Belsley, Kuh & Welsch, 'Regression diagnostics
...', Wiley, 1980.   N.B. Various transformations are used in the table on
pages 244-261 of the latter.

The Boston house-price data has been used in many machine learning papers that address regressio
n
problems.

.. topic:: References

   - Belsley, Kuh & Welsch, 'Regression diagnostics: Identifying Influential Data and Sources of
Collinearity', Wiley, 1980. 244-261.
   - Quinlan,R. (1993). Combining Instance-Based and Model-Based Learning. In Proceedings on the
Tenth International Conference of Machine Learning, 236-243, University of Massachusetts, Amhers
t. Morgan Kaufmann.
```

```
In [11]: fig = plt.figure(figsize=(15.20, 11.20), dpi=55)
         for index, feature_name in enumerate(boston.feature_names):
             #plt.figure(figsize=(4, 3))
             ax = plt.subplot(4,4, index+1)
             plt.setp(ax, xticks=(), yticks=())
             plt.scatter(boston.data[:, index], boston.target)
             plt.ylabel('Price', size=15)
             plt.xlabel(feature_name, size=15)
             plt.xticks(())
             plt.yticks(())
             plt.tight_layout()
         plt.show()
```




```
In [12]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(boston.data, boston.target)
```

```
In [14]: from sklearn.linear_model import LinearRegression
```

```
In [16]: reg=LinearRegression()
         reg.fit(X_train,y_train)
Out[16]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
In [17]: print ('Coefficients: ', reg.coef_)
         print ('Intercept: ',reg.intercept_)

Coefficients:  [-1.11995878e-01  3.53809376e-02 -4.98098042e-02  3.78940892e+00
 -1.87539467e+01  3.75745293e+00 -2.62685706e-03 -1.51422155e+00
  2.37332493e-01 -9.08970264e-03 -9.50031849e-01  6.58527002e-03
 -4.85430576e-01]
Intercept:  38.18817864533051
```

```
In [19]: yhat=reg.predict(X_test)
         print(yhat)

[23.9563393  27.51376046 30.34619072 25.21882714 40.99932787 24.6306371
 27.121765   25.95049304 22.64044692 25.01665508 22.60808632 19.20255309
 23.87623355 23.83967101 23.97890429 20.8235471  18.48327695 14.50354832
 22.49556167 15.85144582 29.1284556  21.19812507 30.16225312 23.4690187
 30.16800214 19.30787511 32.91136892 37.10715804 18.54678584 27.4293857
 16.71081648 23.03658687 22.96912881 13.79132777 27.97001553 26.74250669
 25.3099915   2.71455367 34.52360982 20.09376826 18.23100354 12.88457468
 21.03442935 11.91127407 17.59528035 29.10994335 30.81557519 25.13912671
 27.48411007 15.81891536 35.47456971 28.86411296 18.47921574 19.8968678
 23.50874188 16.63981257 16.41680743 14.43984765 26.64226548 25.09256559
 30.16325972 42.08550996 36.71944733 20.90654566 31.90592819 40.53977321
 23.9779029  17.45063858 20.68451963  5.7594844   8.46158281 16.19460835
 12.56384831 22.50195322 30.25262854 15.5856531  27.97265199 23.05710396
 20.86229052 16.69819147 32.03398697 17.08411028 14.11666651 24.08531414
 32.44242314 25.28770184 18.19580611 28.35264132 15.63388791 20.31361406
 25.93444895 20.45663051 16.97312117 22.34635212 24.56359016 18.33926891
  6.26386178 13.53873011 19.19295704 35.80351361 17.04974193 23.2639437
 25.14017487 20.89749447 22.88702201 29.48581079 36.54192299 24.72418249
 20.66849931 15.6996415  22.1114804  16.74785044 13.85833872 19.47672498
 27.70707126 18.90953359 19.47806553 13.480187   23.57684958 31.35267954
 31.0405453  29.23323563 16.1516374  32.8006281  36.15306456 15.09545044
 13.90317491]
```

Model evaluation

```
In [20]: from sklearn.metrics import r2_score
```

```
In [21]: print("Mean absolute error: %.2f" % np.mean(np.absolute(yhat - y_test)))
         print("Residual sum of squares (MSE): %.2f" % np.mean((yhat - y_test) ** 2))
         print("R2-score: %.2f" % r2_score(yhat , y_test) )

Mean absolute error: 3.65
Residual sum of squares (MSE): 30.30
R2-score: 0.44
```

```
In [ ]:
```