Machine Learning on the Boston House Prices dataset (regression model)

Linear Regression

Dataset: http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_boston.html (http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_boston.html)

In [1]:

```
import numpy as np
import pandas as pd
import scipy.stats as stats
import matplotlib.pyplot as plt
import sklearn
%matplotlib inline
```

In [2]:

```
# Load Boston Housing dataset (already included at sklearn)
from sklearn.datasets import load_boston
boston = load_boston()
```

In [3]:

```
# Description
print(boston.DESCR)
```

```
Boston House Prices dataset
===========================

Notes
------
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 2
5,000 sq.ft.
        - INDUS    proportion of non-retail business acres per town
        - CHAS     Charles River dummy variable (= 1 if tract bounds riv
er; 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 194
0
        - 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 black
s 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.
http://archive.ics.uci.edu/ml/datasets/Housing


This dataset was taken from the StatLib library which is maintained at C
arnegie Mellon University.

The Boston house-price data of Harrison, D. and Rubinfeld, D.L. 'Hedonic
prices and the demand for clean air', J. Environ. Economics & Managemen
t,
vol.5, 81-102, 1978.   Used in Belsley, Kuh & Welsch, 'Regression diagno
stics
...', 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 paper
s that address regression
problems.

**References**

   - Belsley, Kuh & Welsch, 'Regression diagnostics: Identifying Influen
tial Data and Sources of Collinearity', Wiley, 1980. 244-261.
```

```
   - Quinlan,R. (1993). Combining Instance-Based and Model-Based Learnin
g. In Proceedings on the Tenth International Conference of Machine Learn
ing, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.
   - many more! (see http://archive.ics.uci.edu/ml/datasets/Housing)
```

In [4]:

```
print(boston.feature_names)
```

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

In [5]:

```
# Load the dataset in Pandas
df = pd.DataFrame(boston.data)
```

In [6]:

```
# Load the columns names (features attribute)
df.columns = boston.feature_names
```

In [7]:

```
df.head()
```

Out[7]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LS
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
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
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
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
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

In [8]:

```
# Importando o módulo de regressão linear
from sklearn.linear_model import LinearRegression
```

In [9]:

```
X = df
```

In [10]:

```
target = pd.DataFrame(boston.target)
target.columns = ['PRICE']
```

In [11]:

```
Y = target
```

In [12]:

```
from sklearn.model_selection import train_test_split
```

In [13]:

```
# Split dataset train / test
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.33, random_s
tate = 5)
```

In [14]:

```
# Make an instance of Linear Regression
regr = LinearRegression()
```

In [15]:

```
# Treinando o modelo
regr.fit(X_train, Y_train)
```

Out[15]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=Fa
lse)
```

In [16]:

```
# Coeficientes
print("Intercept: ", regr.intercept_)
print("Number of coefficients: ", len(regr.coef_[0]))
print("Coefficients: ", regr.coef_)
```

```
Intercept:  [32.85893263]
Number of coefficients:  13
Coefficients:  [[-1.56381297e-01  3.85490972e-02 -2.50629921e-02  7.8643
9684e-01
  -1.29469121e+01  4.00268857e+00 -1.16023395e-02 -1.36828811e+00
   3.41756915e-01 -1.35148823e-02 -9.88866034e-01  1.20588215e-02
  -4.72644280e-01]]
```

In [17]:

```
print('Coefficient of determination (R2): %.4f' % regr.score(X, Y))
```

```
Coefficient of determination (R2): 0.7333
```

In [ ]:

```

```