*Creado por:*

*Isabel Maniega*

# Random Forest Regression

In [1]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
# pip install scikit-learn
```

In [3]:
```
import numpy as np
from sklearn import datasets, linear_model
import matplotlib.pyplot as plt
import pandas as pd
```

In [4]:
```
boston = datasets.load_boston()
```

In [5]:
```
boston.data
```

Out[5]:
```
array([[6.3200e-03, 1.8000e+01, 2.3100e+00, ..., 1.5300e+01, 3.9690e+02,
        4.9800e+00],
       [2.7310e-02, 0.0000e+00, 7.0700e+00, ..., 1.7800e+01, 3.9690e+02,
        9.1400e+00],
       [2.7290e-02, 0.0000e+00, 7.0700e+00, ..., 1.7800e+01, 3.9283e+02,
        4.0300e+00],
       ...,
       [6.0760e-02, 0.0000e+00, 1.1930e+01, ..., 2.1000e+01, 3.9690e+02,
        5.6400e+00],
       [1.0959e-01, 0.0000e+00, 1.1930e+01, ..., 2.1000e+01, 3.9345e+02,
        6.4800e+00],
       [4.7410e-02, 0.0000e+00, 1.1930e+01, ..., 2.1000e+01, 3.9690e+02,
        7.8800e+00]])
```

In [6]:
```
print('Nombre de columnas:')
print(boston.feature_names)
```

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

In [7]:
```
df = pd.DataFrame(boston.data, columns=boston.feature_names)
df
```

Out[7]:

| | 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 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **501** | 0.06263 | 0.0 | 11.93 | 0.0 | 0.573 | 6.593 | 69.1 | 2.4786 | 1.0 | 273.0 | 21.0 | 391.99 | 9.67 |
| **502** | 0.04527 | 0.0 | 11.93 | 0.0 | 0.573 | 6.120 | 76.7 | 2.2875 | 1.0 | 273.0 | 21.0 | 396.90 | 9.08 |
| **503** | 0.06076 | 0.0 | 11.93 | 0.0 | 0.573 | 6.976 | 91.0 | 2.1675 | 1.0 | 273.0 | 21.0 | 396.90 | 5.64 |
| **504** | 0.10959 | 0.0 | 11.93 | 0.0 | 0.573 | 6.794 | 89.3 | 2.3889 | 1.0 | 273.0 | 21.0 | 393.45 | 6.48 |
| **505** | 0.04741 | 0.0 | 11.93 | 0.0 | 0.573 | 6.030 | 80.8 | 2.5050 | 1.0 | 273.0 | 21.0 | 396.90 | 7.88 |

506 rows × 13 columns

In [8]:
```
print("Informacion en el dataset:")
print(boston.keys())
```

```
Informacion en el dataset:
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename', 'data_module'])
```

In [9]:
```
print("Características del dataset:")
print(boston.DESCR)
```

```
Características del dataset:
.. _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 usually the targe
t.

    :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 black people 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 University.

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 regression
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 Internat
ional Conference of Machine Learning, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.
```

In [10]:
```
print("Cantidad de datos:")
print(boston.data.shape)
```

```
Cantidad de datos:
(506, 13)
```

In [11]:
```
# Seleccionamos como valor de la X la columna 6 (RM):
X = boston.data
```

In [12]:
```
y = boston.target
```

In [13]:
```
from sklearn.model_selection import train_test_split
# Separo los datos de "train" entrenamiento y "test" prueba para probar los algoritmos
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

In [14]:
```
from sklearn.ensemble import RandomForestRegressor

bar = RandomForestRegressor(n_estimators=300, max_depth=8)
```

In [15]:
```
bar.fit(X_train, y_train)
```

Out[15]:
```
▼                  RandomForestRegressor
RandomForestRegressor(max_depth=8, n_estimators=300)
```

In [16]:
```
y_pred = bar.predict(X_test)
y_pred
```

Out[16]:
```
array([44.3398787 , 27.28750354, 15.06300939, 32.08058357, 29.0641241 ,
       14.15286096, 18.02186891, 19.05327279, 21.67456347, 30.48173071,
       23.51312947, 14.01961423, 45.82066667, 21.68002651, 15.24203262,
       14.06162685, 33.20319627, 21.56945787, 24.26928624, 42.96421358,
       13.08476604, 15.40425136, 20.1098083 , 21.3362411 , 19.70256107,
       15.11549541, 19.57201829, 21.31930004, 17.80771642, 21.77590751,
       27.76195876, 22.53708805, 18.70305633, 20.6834134 , 15.63071435,
       30.74278793, 21.28108742, 20.72516693, 23.32236957, 21.69131747,
       16.20253013, 20.53926238, 44.1402732 , 16.44643871, 19.92143381,
       24.41265267, 26.24552881, 13.22686052, 21.56652891, 19.50516471,
       15.55558974, 18.60121777, 21.51928443, 21.70821962, 34.10102007,
       18.60758244, 32.87229454, 45.73485556, 14.52186326, 20.3935731 ,
       20.74888876, 30.95023569, 28.86502024, 34.95194282, 18.8671126 ,
       24.01963329, 20.34120416, 22.81679565, 20.65778088, 23.26726719,
       14.9540345 , 47.07589419, 17.10693726, 26.8978309 , 13.54340797,
       20.39892365, 19.25663575,  7.47320321, 22.30277043, 20.11328324,
       20.64790686, 23.86037784, 16.56051751, 33.9869438 , 23.32389331,
       44.10722251, 11.95344854, 26.37643247, 18.92783562, 26.04599677,
       29.71992971, 34.05021107, 27.19509647, 14.87268216, 30.74480788,
       23.03854503, 11.37367799, 26.26808939, 48.98806667, 21.97185577,
       24.39512112, 28.07295072])
```

In [17]:
```
y_test
```

Out[17]:
```
array([46.7, 24.5, 14.9, 30.1, 29.1, 17.2, 19.7, 17.1, 21.4, 24.8, 22.9,
       14.6, 50. , 23.1, 13.2, 19.1, 37.3, 20.6, 23.8, 50. ,  8.5, 13.1,
       21.8, 20.7, 17.1, 14.1, 17.5, 22. , 16.5, 23. , 23.3, 20.4, 16.6,
       18.2, 14.2, 32.5, 21.7, 19.6, 23.2, 24.4, 14.3, 14.5, 43.1, 13.1,
       21.4, 23.5, 29.4, 11.9, 19.6, 20.8, 23.2, 19.1, 20.9, 20.3, 32.9,
       20.6, 30.3, 41.7, 12.5, 18.5, 20.9, 32. , 25. , 50. , 18.6, 29.6,
       20.4, 22.4, 22.6, 22.9, 27.5, 43.5, 13.9, 25.1, 14.4, 19.3, 17.5,
        5. , 21. , 18.5, 21.7, 25. , 14.8, 33.2, 23.3, 21.9, 13.3, 24. ,
       16.8, 24.8, 32.4, 37.9, 24.3, 13.3, 33.1, 21.4, 13.9, 22. , 50. ,
       18.6, 28.7, 29.8])
```

In [18]:
```
print("Datos del modelo Bosques Aleatorios Regresión")
print()

print("precisión del modelo:")
print(bar.score(X_train, y_train))
```

```
Datos del modelo Bosques Aleatorios Regresión

precisión del modelo:
0.9746011155858634
```

*Creado por:*

*Isabel Maniega*