Creado por:

Isabel Maniega

Regresión Lineal Simple

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

https://scikit-learn.org/stable/install.html

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

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

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

In [5]:
```
print("Informacion en el dataset:")
print(boston.keys())
```

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

In [6]:
```
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 [7]:
```
print("Cantidad de datos:")
print(boston.data.shape)
```

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

In [8]:
```
print('Nombre de columnas:')
print(boston.feature_names)
```

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

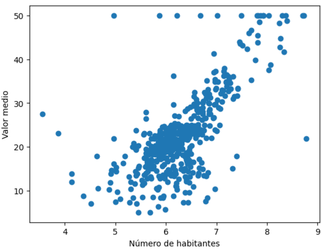
In [10]:
```
# Seleccionamos como valor de la X la 5ª columna del dataset

X = boston.data[:, np.newaxis, 5]
```

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

In [13]:
```
# Gráficamos los datos correspondientes

plt.scatter(X, y)
plt.xlabel("Número de habitantes")
plt.ylabel("Valor medio")
plt.show()
```



In [18]:
```
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 [19]:
```
# creamos el regresor
lr = linear_model.LinearRegression()
```

In [20]:
```
# entrenamos el modelo
lr.fit(X_train, y_train)
```

Out[20]:
```
LinearRegression()
```

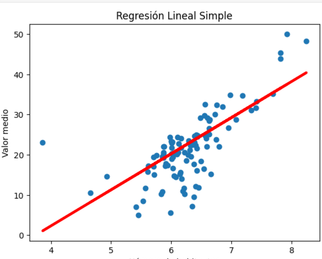
In [21]:
```
y_pred = lr.predict(X_test)
y_pred
```

Out[21]:
```
array([21.96735625, 25.22413749, 40.3955344 , 32.93992776, 19.05150528,
       23.31313362, 28.98334228, 25.43049002, 20.41522635, 21.26755202,
       23.06192185, 24.16545929, 25.52918036, 30.8674306 , 21.70717263,
       18.77337795, 27.35943759, 24.12060005,  1.06294774, 35.40718627,
       22.10193399, 18.07357372, 21.68922893, 25.28694043, 32.14143318,
       26.96467623, 24.67685469, 23.33107732, 19.46421034, 24.01293786,
       27.03645102, 23.67200759, 25.30488413, 23.84247272, 23.60920464,
       16.82799212, 22.51463905, 20.8817625 , 37.48865527, 21.79689112,
       20.27167676, 19.32066075, 18.92589939, 23.16061219, 16.79957114,
       19.71542212, 25.89702618, 20.60363518, 22.83762562, 19.14122377,
       25.84319508, 26.11235056, 20.50494484, 23.0350063 , 20.65746628,
       32.90404036, 25.65478625, 24.56022065, 23.66303574, 20.31653601,
       21.08811503, 27.89774854, 29.91641459, 20.21784567, 17.62498126,
       24.04882525, 19.06944898, 23.86041642, 25.99571652,  8.14173672,
       17.66086866, 20.4421419 , 23.64509204, 21.59951044, 28.7680179 ,
       16.76368375, 25.0805879 , 20.31653601, 22.26342727, 19.07842083,
       24.35386812, 15.32818788, 10.60002338, 20.34345156, 21.20474907,
       36.56455481, 18.89898384, 16.33303499, 21.37521421, 17.59806572,
       21.50079194, 14.97828577, 23.5822891 , 26.76729554, 36.56455481,
       25.88805433, 19.10533637, 22.23651173, 20.11915533, 22.73893528,
       23.95910676, 20.06532423])
```

In [22]:
```
y_test
```

Out[22]:
```
array([11. , 29.8, 48.3, 33.2, 19.3,  7.2, 34.9, 24.1, 23.2, 20.5, 22.3,
       24.8, 22. , 34.7, 21.7, 10.2, 22. , 24.5, 23.1, 35.2, 11.7, 19.9,
       24.1, 32.5, 31. , 23.7, 18.4, 22.5, 17.8, 21.6, 32.4, 24.7, 24.8,
       22.3, 17.7,  8.5, 18.5, 14.5, 50. , 14. , 20.3, 17.2, 10.8, 19.6,
       17.2, 17.4, 25.1, 20.4, 23.5, 22. , 28.4, 15.2, 16.6, 21.2, 14.8,
       31.7, 29.1, 29.1, 23.1, 18.9, 20.1, 32. , 28.7, 23.4, 19.4, 16.1,
       22. , 12.1, 28.7, 10.5, 15.1, 19.4,  9.5, 15.6, 26.7, 15.7, 16.5,
       21.7, 20.5, 20.6, 11.8,  5. , 14.6, 22.9, 24.3, 45.4, 19.5, 11.7,
       22.7, 17.1, 15.2,  7. , 23. , 30.1, 43.8, 26.6, 20.3, 10.2,  5.6,
       20. , 25. , 24.3])
```

In [23]:
```
plt.scatter(X_test, y_test)
plt.plot(X_test, y_pred, color='red', linewidth=3)
plt.title("Regresión Lineal Simple")
plt.xlabel("Número de habitantes")
plt.ylabel("Valor medio")
plt.show()
```



In [24]:
```
print('DATOS DEL MODELO REGRESIÓN LINEAL SIMPLE')
print()
print('Valor de la pendiente o coeficiente "a":')
print(lr.coef_)
print('Valor de la intersección o coeficiente "b":')
print(lr.intercept_)
print()
print('La ecuación del modelo es igual a:')
print('y = ', lr.coef_, 'x ', lr.intercept_)
```

```
DATOS DEL MODELO REGRESIÓN LINEAL SIMPLE

Valor de la pendiente o coeficiente "a":
[8.97184915]
Valor de la intersección o coeficiente "b":
-33.59530551205124

La ecuación del modelo es igual a:
y =  [8.97184915] x  -33.59530551205124
```

In [25]:
```
print("Precisión del modelo:")
print(lr.score(X_train, y_train))
```

```
Precisión del modelo:
0.4766422843384107
```

Creado por:

Isabel Maniega