Regresión Lineal y Mínimos Cuadrados

March 1, 2020

1 Regresión Lineal y Mínimos Cuadrados

Notebook siguiendo las indicaciones de Dot CSV, por Alejandro Valverde.

Bibliotecas neceasrias

[1]:
```
import numpy as np #Cálculo numérico y cálculos matemáticos avanzados

import matplotlib.pyplot as plt #Visualización gráfica
```

1.1 Dataset de precio de las casas en Boston

Se encuentra en la biblioteca sklearn, que es una biblioteca que incluye numerosos modelos y datasets destinados a la implementación y prueba de algoritmos de **machine learning**.

[2]:
```
from sklearn.datasets import load_boston #Modelos y datasets de machine learning
```

[3]:
```
#Cargar el dataset
boston = load_boston()
```

1.2 Mostrar la información del dataset

En este caso, se va a estudiar el comportamiento del atributo **MEDV**, que hace referencia al *valor medio de la vivienda*, usando el número medio de habitaciones.

[4]:
```
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 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

    :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 International Conference of Machine Learning, 236-243,
University of Massachusetts, Amherst. Morgan Kaufmann.
```

1.3 Objetivo:

Utilizando el número medio de habitaciones(**RM**), intentar predecir el valor medio de la vivienda (**MEDV**).

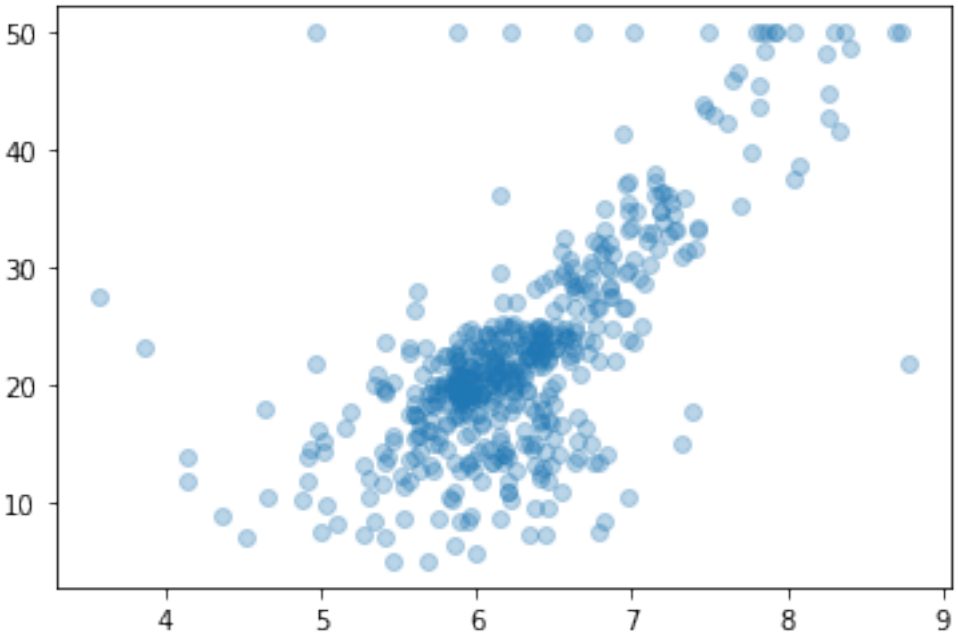
[5]:
```
#Se requieren que los datos esten en forma de array, por eso se usa np.array()

#Columna de valor medio de habitaciones
#[5] sería ala fila 5, pero como se quiere la columna hay que poner [:, 5]
rm = np.array(boston.data[:, 5])
#Columna de valor medio de la vivienda
medv = np.array(boston.target)
```

1.3.1 Gráfica de la nube de puntos

Donde la X va a ser **RM** y la Y va a ser **MEDV** usadno *matplot*.

[6]:
```
plt.scatter(rm, medv, alpha=0.3) #aplha es la transparencia de los puntos
plt.show()
```



1.3.2 Fórmula para minimizar el error cuadrático medio (MCO)

$\beta = (X^T X)^{-1} X^T Y$

[7]:
```
#Anadimos columna de 1s para tratar el término independiente
rm_mod = np.array([np.ones(506), rm]).T #Necesitamos la transpuesta
```

[8]:
```
#Escribir la formula (@ es multiplicacion matricial)
B = np.linalg.inv(rm_mod.T @ rm_mod) @ rm_mod.T @ medv
```

[9]:
```
B
```

[9]:
```
array([-34.67062078,   9.10210898])
```

1.3.3 Dibujo de la gráfica

[10]:
```
plt.plot([3.5, 9], [B[0]+B[1]*3.5, B[0]+B[1]*9], c="red")
plt.scatter(rm, medv, alpha=0.3) #aplha es la transparencia de los puntos
plt.show()
```