Regresión Lineal Múltiple

Métodos Computacionales 2023 - Trabajo Práctico 1

Ignacio Pardo Luca Mazzarello

2023-04-13

Primera parte

El objetivo de esta sección es deducir una fórmula para la solución óptima β^* siguiendo los pasos a continuación:

a) Mostrar que el espacio columna de la matriz X es un subespacio vectorial de $\mathbb{R}^n$:

$$Col(X) = \{b \text{ en } \mathbb{R}^n \text{ tales que b} = \text{X}\beta \text{ con } \beta \text{ variando en } \mathbb{R}^p\}$$

Para mostrar que $Col(X)$ es un subespacio vectorial de $\mathbb{R}^n$, debemos demostrar que $Col(X)$ cumple con las siguientes propiedades:

1. El vector cero pertenece al espacio $Col(X)$.

$$Col(X) = Gen\{x_1, x_2, \ldots, x_p\}$$

El vector 0 está en $Col(X)$ porque $0 \times x_1 + 0 \times x_2$ es combinación lineal de los vectores de %Col(X)%.

2. La suma de dos vectores pertenecientes a $Col(X)$ también pertenece a $Col(X)$.

Sean u, v vectores en $Col(X)$, s_1, s_2 y t_1, t_2 escalares tales que:

$$u = s_1x_1 + s_2x_2 \iff u \in Col(X)$$

Por ser combinación lineal de los vectores de $Col(X)$.

$$v = t_1x_1 + t_2x_2 \iff v \in Col(X)$$

Por ser combinación lineal de los vectores de $Col(X)$.

Entonces:

$$u + v = s_1x_1 + s_2x_2 + t_1x_1 + t_2x_2$$
$$u + v = (s_1 + t_1)x_1 + (s_2 + t_2)x_2$$

Llamamos $c_1 = s_1 + t_1$ y $c_2 = s_2 + t_2$.

Entonces:

$$u + v = c_1x_1 + c_2x_2$$

$$c_1x_1 \in Col(X) \land c_2x_2 \in Col(X) \implies c_1x_1 + c_2x_2 \in Col(X) \implies u + v \in Col(X)$$

3. El producto de un escalar por un vector perteneciente a $Col(X)$ también pertenece a $Col(X)$.

Sea u un vector en $Col(X)$, s_1, s_2 escalares tales que:

$$u = s_1x_1 + s_2x_2 \iff u \in Col(X)$$

Llamemos c al escalar.

Entonces:

$$cu = c \times (s_1x_1 + s_2x_2)$$
$$cu = (c \times s_1)x_1 + (c \times s_2)x_2$$

Llamemos $c_1 = c \times s_1$ y $c_2 = c \times s_2$

Entonces:

$$cu = c_1x_1 + c_2x_2$$

$$c_1x_1 \in Col(X) \wedge c_2x_2 \in Col(X) \implies c_1x_1 + c_2x_2 \in Col(X) \implies cu \in Col(X)$$

Por lo tanto $Col(X)$ es un subespacio vectorial de $\mathbb{R}^n$.

b) Supongamos que cuando hablamos de vectores en $\mathbb{R}^n$ nos referimos a vectores columna de $\mathbb{R}^{n\times 1}$. Mostrar en ese caso que el producto escalar entre dos vectores u, v en $\mathbb{R}^n$ puede calcularse como:

$$u \cdot v = v^\intercal u$$

donde operación en el lado derecho de la igualdad es el producto de matrices usual.

Sean u, v vectores en $\mathbb{R}^n$.

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}, v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

Entonces:

$$v^\intercal = \begin{bmatrix} v_1 & v_2 & \dots & v_n \end{bmatrix}$$

$$u \cdot v = \begin{bmatrix} u_1 & u_2 & \dots & u_n \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \sum_{i=1}^{n} u_i v_i$$

c) Aplicando el teorema tomando como subespacio S el subespacio del ítem (a), el punto y de $\mathbb{R}^n$ como el vector de la variable dependiente, y el vector b como $b = X\beta^*$, convertir esta ecuación de optimalidad

$$||y - X\beta^*|| = \min_{\beta \text{ en } \mathbb{R}^p} ||y - X\beta||$$

en la condición de ortogonalidad que corresponde a la equivalencia 2 del teorema.

$$\forall \beta \in \mathbb{R}^{\mathrm{p}}, (\mathrm{y} - \mathrm{X}\beta^*) \cdot \mathrm{X}\beta = 0$$

Llamemos $t = X\beta^*$ y $s = X\beta$ donde s son todos los valores en el subespacio $S = X\beta$ con β variando en $\mathbb{R}^{\mathrm{p}}$, y t es el vector $X\beta^*$ que es el vector que minimiza la distancia entre y y $X\beta$. t es la *proyección ortogonal* de y sobre el subespacio S.

Entonces por el Teroema. Sea y un vector cualquiera de $\mathbb{R}^{\mathrm{n}}$ y S un subespacio de $\mathbb{R}^{\mathrm{n}}$. El vector de S que minimiza la distancia del subespacio S al vector y es aquel b de S tal que $y - b$ es ortogonal a todo vector s de S. Es decir, las siguientes dos condiciones son equivalentes:

$$\begin{aligned} & ||y - t|| = \min_{s \in S} ||y - s|| \\ \implies & (y - t) \cdot s = 0, \forall s \in S \\ \implies & (y - X\beta^*) \cdot X\beta = 0, \forall \beta \in \mathbb{R}^{\mathrm{p}} \end{aligned}$$

d) A la ecuación obtenida en el ítem (c), aplicarle la identidad del producto escalar vista en el item (b), para llegar a la ecuación:

$$X^\intercal (y - X\beta^*) \cdot \beta = 0$$

$$(y - X\beta^*) \cdot X\beta = 0$$

Llamemos $u = y - X\beta^*$ y $v = X\beta$ con $u \in \mathbb{R}^{\mathrm{n}}$ y $v \in \mathbb{R}^{\mathrm{n}}$.

$$\begin{aligned} u \cdot v = v^\intercal u & \iff (X\beta)^\intercal (y - X\beta^*) = 0 \\ & \iff \beta^\intercal X^\intercal (y - X\beta^*) = 0 \end{aligned}$$

Ahora llamamos $k = \beta^\intercal$ y $q = X^\intercal (y - X\beta^*)$.

Entonces, por propiedad de la transpuesta:

$$\begin{aligned} kq = q \cdot k^\intercal & \iff 0 = \beta^\intercal X^\intercal (y - X\beta^*) \\ & \qquad = X^\intercal (y - X\beta^*) \cdot \beta^{\intercal\intercal} \\ & \qquad = X^\intercal (y - X\beta^*) \cdot \beta \end{aligned}$$

$$\iff X^\intercal (y - X\beta^*) \cdot \beta = 0$$

e) Se sabe que el único vector que es ortogonal a todo vector v de $\mathbb{R}^{\mathrm{n}}$ es el vector nulo. Es decir, si u es un vector fijo tal que $u \cdot v = 0$ para todo v en $\mathbb{R}^{\mathrm{n}}$, entonces $u = 0$. Usando esto y la ecuación obtenida en el ítem (d), llegar a la fórmula:

$$X^\intercal X\beta^* = X^\intercal y$$

Partimos de la igualdad obtenida en el ítem (d):

$$\begin{aligned} 0 &= X^\intercal (y - X\beta^*) \cdot \beta \\ &= (X^\intercal y - X^\intercal X\beta^*) \cdot \beta \end{aligned}$$

Fijamos $(X^\intercal y - X^\intercal X\beta^*) = 0$ ya que es el único vector que es ortogonal a todo vector β de $\mathbb{R}^{\mathrm{p}}$.

$$\begin{aligned} & (X^\intercal y - X^\intercal X\beta^*) = 0 \\ & \iff X^\intercal X\beta^* = X^\intercal y \end{aligned}$$

f) Finalmente, suponiendo que las columnas de X son linealmente independientes, se tiene que la matriz $X^\intercal X$ es invertible. Despejar β^* de la ecuación del ítem (e) para llegar a la fórmula de la solución óptima al problema de regresión.

$$\begin{aligned} X^\intercal X\beta^* &= X^\intercal \\ (X^\intercal X)^{-1}X^\intercal X\beta^* &= (X^\intercal X)^{-1}X^\intercal \\ \beta^* &= (X^\intercal X)^{-1}X^\intercal y \end{aligned}$$

Segunda parte.

En esta sección la idea es realizar regresión lineal en $\mathbb{R}^2$ y analizar como se comportan las soluciones obtenidas.

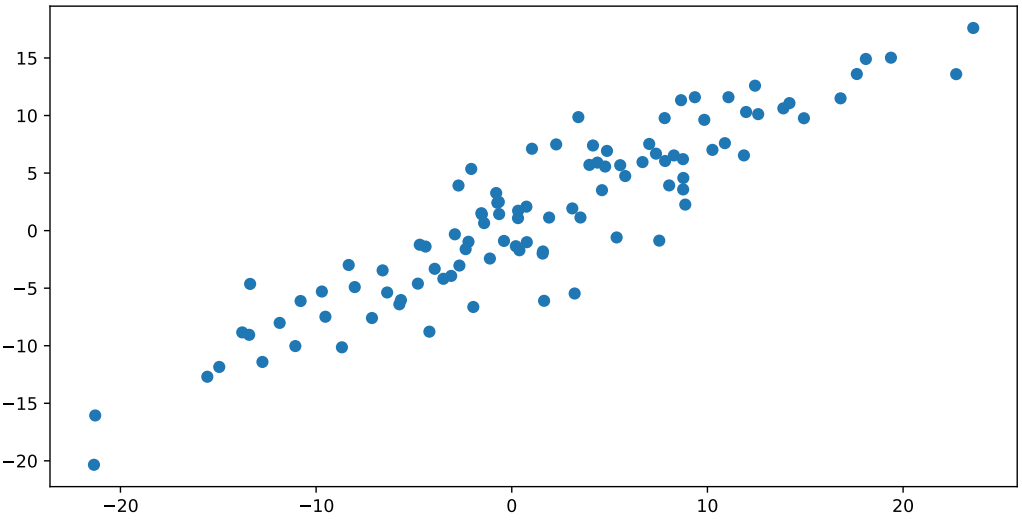
1. Usando los datos del archivo ejercicio_1.csv:

a) Graficar todos los puntos en el plano xy. Nota: La primer columna del archivo marca el valor de x y la segunda el valor de y de cada punto. Recomendamos usar la biblioteca pandas para leer los archivos con la función read_csv.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

```
data_ej1 = pd.read_csv('data/ejercicio_1.csv', sep=',')
x_key = data_ej1.keys()[0]
y_key = data_ej1.keys()[1]

x_values = data_ej1[x_key]
y_values = data_ej1[y_key]
```

```
plt.figure(figsize=(10, 5))
plt.plot(x_values, y_values, 'o')
```



b) Utilizando los conceptos teóricos desarrollados en la primera parte, hallar la recta que mejor aproxima a los datos.

$$||y - X\beta^*|| = \min_{\beta \text{ en } \mathbb{R}^p} ||y - X\beta|| \beta^* = (X^\intercal X)^{-1} X^\intercal y$$

```
X = np.array(x_values).reshape(-1, 1)
y = np.array(y_values).reshape(-1, 1)
X.shape, y.shape
```

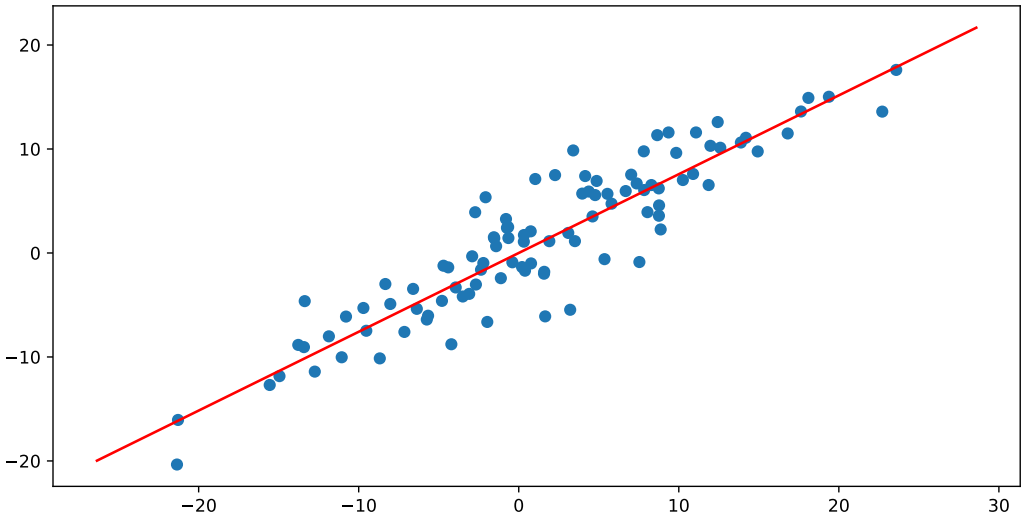
```
## ((100, 1), (100, 1))
```

```
def linear_regression(X, y):
    X_t = X.T
    X_t_X = np.dot(X_t, X)
    X_t_X_inv = np.linalg.inv(X_t_X)
    X_t_X_inv_X_t = np.dot(X_t_X_inv, X_t)
    B = np.dot(X_t_X_inv_X_t, y)
    return B
```

```
b = linear_regression(X, y)
b
```

```
## array([[0.75785414]])
```

```
def plot_regression_line(X, y, reg_func=linear_regression):

    b = reg_func(X, y).flatten()

    plt.figure(figsize=(10, 5))

    plt.plot(X, y, 'o')

    min_x = np.min(X) - 5
    max_x = np.max(X) + 5

    line_x = np.linspace(min_x, max_x, 1000)
    line_y = b[0] * line_x

    if len(b) > 1:
        line_y += b[1]

    plt.plot(line_x, line_y, 'r')

    plt.show()

plot_regression_line(X, y)
```

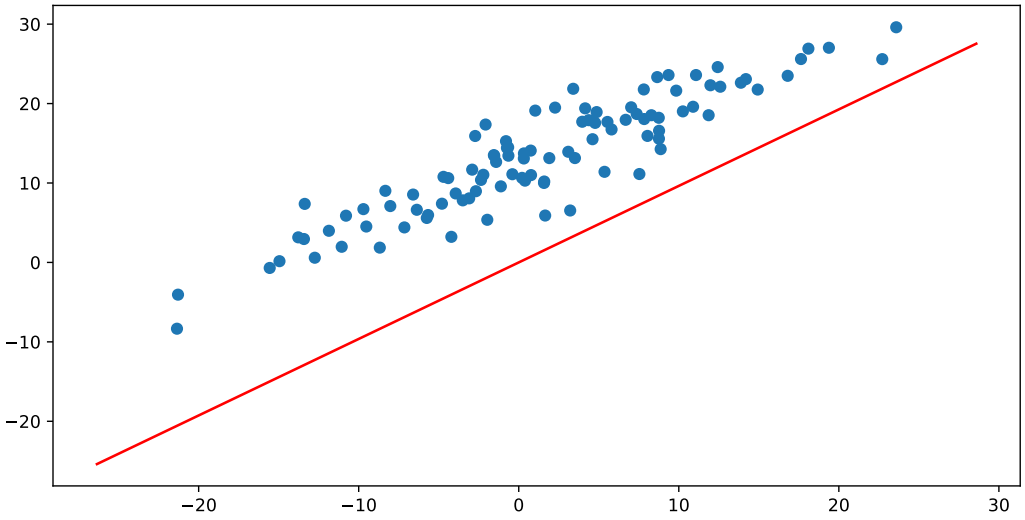


c) Realizar nuevamente los incisos (a) y (b) pero considerando los puntos

$$(x_i, y_i + 12) \text{ con } i = 1 \dots n$$

donde (x_i, y_i) eran los puntos originales. ¿Es buena la aproximación realizada?, ¿cuál es el problema?

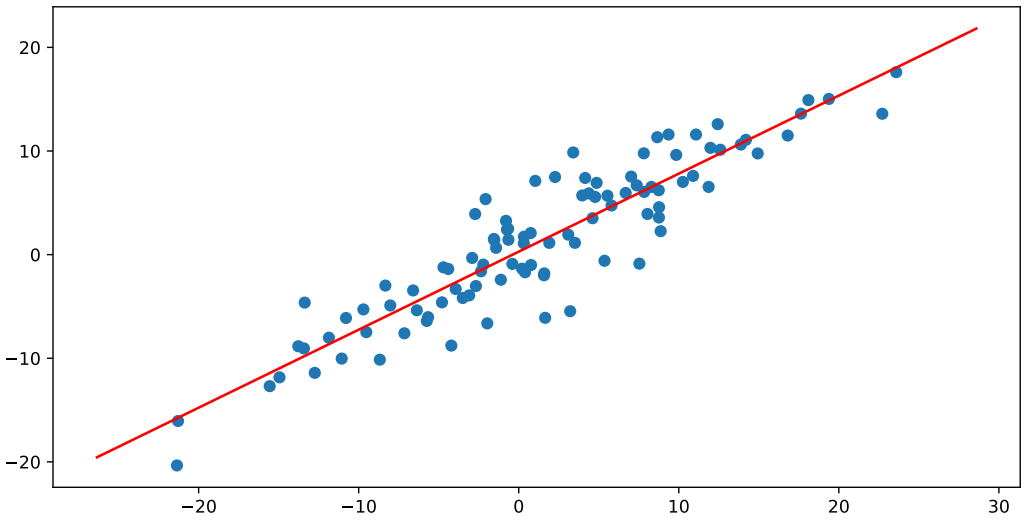
```
plot_regression_line(X, y + 12)
```



d) ¿Cómo se podría extender el modelo para poder aproximar cualquier recta en el plano?

```
def linear_regression_(X, y):
    ones = np.ones((X.shape[0], 1))
    X = np.concatenate((X, ones), axis=1)
    X_t = X.T
    X_t_X = np.dot(X_t, X)
```

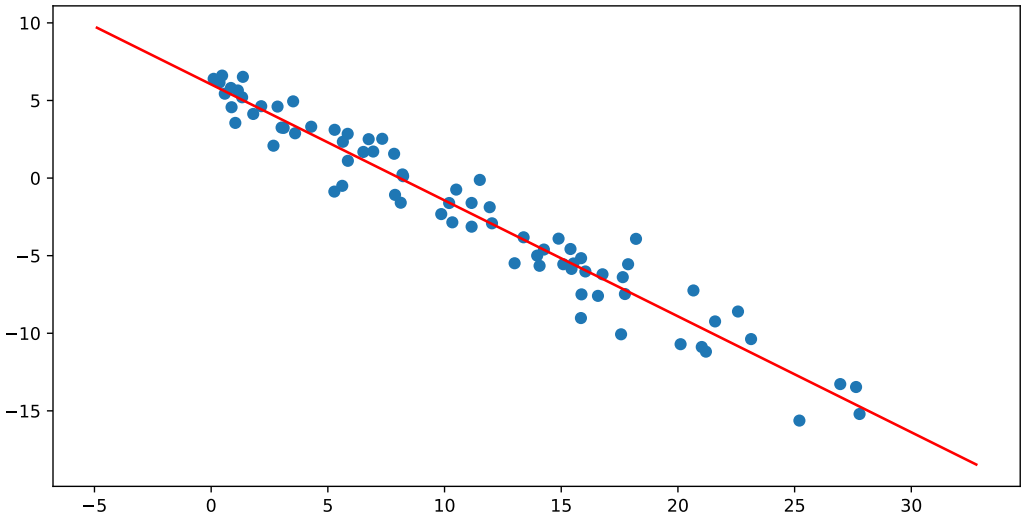
```
    X_t_X_inv = np.linalg.inv(X_t_X)
    X_t_X_inv_X_t = np.dot(X_t_X_inv, X_t)
    B = np.dot(X_t_X_inv_X_t, y)
    return B

plot_regression_line(X, y, linear_regression_)
```



2. Usando los datos del archivo ejercicio_2.csv:

a) Graficar y aproximar los puntos con una recta.

```
data_ej2 = pd.read_csv('data/ejercicio_2.csv', sep=',')

x_2_key = data_ej2.keys()[0]
y_2_key = data_ej2.keys()[1]

x_2_values = data_ej2[x_2_key]
y_2_values = data_ej2[y_2_key]

X2 = np.array(x_2_values).reshape(-1, 1)
y2 = np.array(y_2_values).reshape(-1, 1)

plot_regression_line(X2, y2, linear_regression_)
```



b) Imaginemos que los datos forman parte de mediciones de algún tipo, como por ejemplo la temperatura de un procesador a lo largo del tiempo), y queremos predecir cuál va a ser la temperatura en el futuro. ¿Es buena la aproximación que realizamos?, ¿cuál fue el problema en este caso?

Tercera parte. Regresión lineal en datos reales.

En esta sección utilizaremos el conjunto de datos provisto en Machine Learning Repository. Este consiste en datos de ventas de 414 casas en Taiwan. La información provista por casa es (en orden):

i) La fecha en que se realizó la transacción. Expresada en formato

$$\text{año} + \frac{\text{numero\_mes}}{12}$$

ii) La edad de la casa en años.

iii) La distancia a la estación de tren o subte más cercana en metros.

iv) La cantidad de almacenes alcanzables a pie.

v) La latitud en grados.

vi) La longitud en grados.

vii) El precio por Ping. La cual es una unidad utilizada en Taiwan que representa 3,3 metros cuadrados.

Vamos a dividir este conjunto de datos en dos:

i) Datos de entrenamiento: usamos los datos desde la observación 1 a la 315 inclusive.

ii) Datos de test: usamos los datos desde la observación 316 a la 414 inclusive.

1. Teniendo en cuenta la teoría desarrollada en la primer parte del trabajo práctico y usando los datos de entrenamiento:

```
real_estate = pd.read_csv('data/real_estate.csv', sep=';')
real_estate.drop(['No'], axis=1, inplace=True)

X_matrix = np.array(real_estate.drop(['Y house price of unit area'], axis=1)).reshape(-1, 6)
y_matrix = np.array(real_estate['Y house price of unit area']).reshape(-1, 1)
```

```
X_matrix.shape, y_matrix.shape
```

```
## ((414, 6), (414, 1))
```

```
X_train = X_matrix[0:315]
y_train = y_matrix[0:315]

X_test = X_matrix[315:]
y_test = y_matrix[315:]

X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

```
## ((315, 6), (315, 1), (99, 6), (99, 1))
```

a) Estimar los parámetros $\hat{\beta}$ que minimizan el error cuadrático medio para este problema.

```
b_top = linear_regression(X_train, y_train)
b_top
```

```
## array([[ 3.46731867e+00],
##        [-2.82548977e-01],
##        [-5.33948393e-03],
##        [ 1.09746715e+00],
##        [ 2.43898979e+02],
##        [-1.07177833e+02]])
```

b) Encontrar $\hat{y}$ la estimación de la variable de respuesta.

```
y_pred = np.dot(X_train, b_top)
y_pred.shape
```

```
## (315, 1)
```

c) ¿Cuánto vale el error cuadrático medio?

Definimos error cuadrático medio como

$$\text{ECM} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

donde y_i son observaciones de una variable y $\hat{y}_i$ estimaciones de las mismas.

```
def ECM(y, y_pred):
    return np.sum(np.power(y - y_pred, 2)) / len(y)

ECM(y_train, y_pred)
```

```
## 84.01910082051779
```

2. Utilizando los datos de test, analizar cuál es el error cuadrático medio al utilizar los parámetros $\hat{\beta}$ estimados en el punto anterior.

```
y_eval = np.dot(X_test, b_top)
ECM(y_test, y_eval)
```

```
## 59.706941844978374
```

a) ¿Es la estimación mejor que sobre los datos originales?, ¿a qué se debe la discrepancia?

La estimación es mejor que sobre los datos originales ya que el error cuadrático medio es menor. Esto se debe a que los datos de entrenamiento son más cercanos a los datos de test que a los datos "originales".

b) ¿Qué sucede con el ECM del segundo conjunto de casas si se realiza la regresión sobre todos los datos al mismo tiempo (es decir, las 414 casas)?

```
b_ = linear_regression(X_matrix, y_matrix)
y_pred_ = np.dot(X_matrix, b_)
ECM(y_matrix, y_pred_)
```

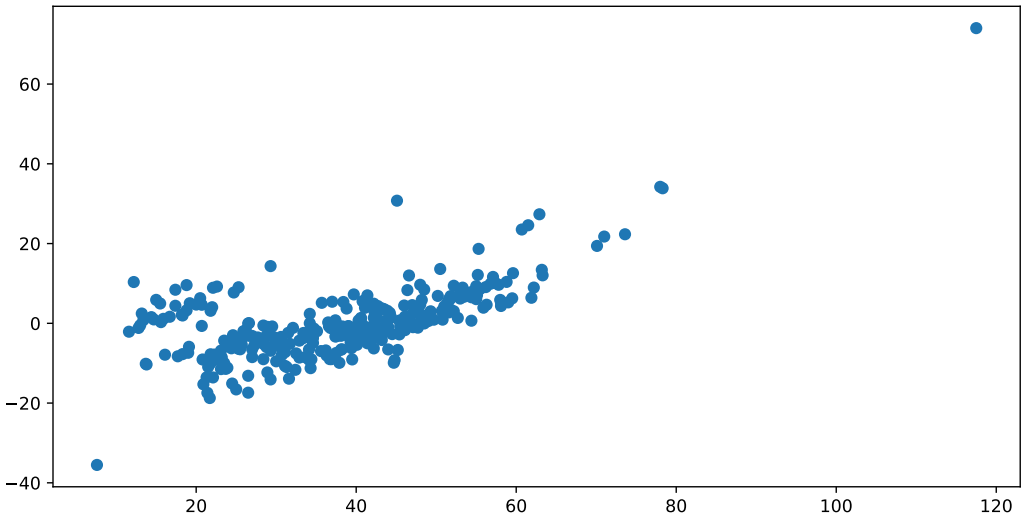
```
## 77.99210449987768
```

3. Graficar el error cometido por cada casa. Es decir el valor absoluto de la diferencia entre el precio por Ping real y el estimado.

```
y_pred.shape
```

```
## (315, 1)
```

```
y_matrix_plt = y_matrix[:y_pred.shape[0]].T[0]
y_pred_plt = y_pred.T[0]

plt.figure(figsize=(10, 5))
plt.plot(y_matrix_plt, y_matrix_plt - y_pred_plt, 'o')
plt.show()
```



4. Imaginemos que se agrega una nueva columna a los datos que informa el año en que la misma fue construida. ¿Disminuiría esto el ECM?