Aprendizaje automático

Práctica 1. Regresión

Universidad de Zaragoza, curso 2022/2023

Juan Eizaguerri Serrano 816079

Elección de las métricas de error

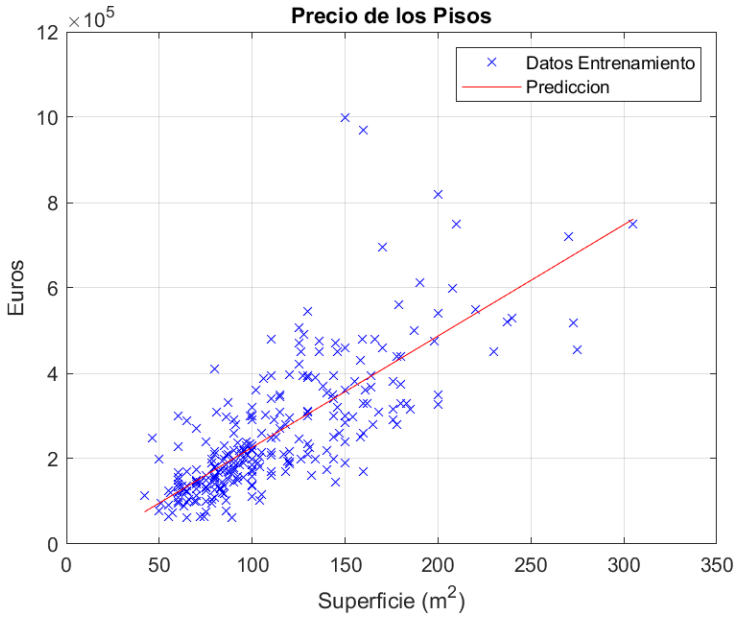
A la hora de entrenar, tiene sentido utilizar una función de coste cuadrático por su facilidad para ser derivada, incluso sabiendo que los errores grandes van a afectar más de lo que deberían a la solución.

Para comprobar el modelo se pueden utilizar funciones mejores. Entre los datos hay una cantidad considerable de espurios por lo que coste cuadrático y RMSE quedan descartadas.

Para que los espurios afecten menos al resultado se puede utilizar **MAE**, o **MRE** si se quiere en términos de porcentaje.

Regresión monovariable con ecuación normal

Utilizando la ecuación normal se entrena el modelo para las entradas de entrenamiento.



Se busca una recta que predice un salida para cada valor de x (superficie), que responde a la siguiente ecuación:

$y = X\theta = \theta_1 X_1 + \theta_2 x_2$ *(Dado* $x_1 = 1$*)*

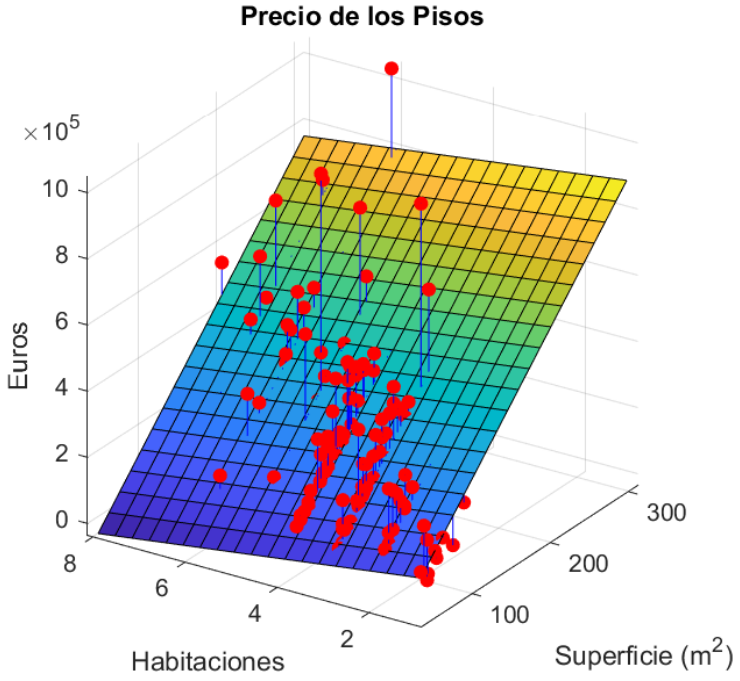
Donde x_2 es la superficie de la vivienda.

La resolución de la ecuación normal tiene como objetivo calcular la θ que minimiza el coste.

Una vez entrenado, se puede evaluar el modelo con la métrica indicada en el apartado anterior y los datos del set de test. Utilizando MRE se obtiene un error de 0.2744.

Regresión multivariable con ecuación normal

Similar al apartado anterior, se entrena el modelo con los datos de entrenamiento para obtener el siguiente resultado.



Al haber 2 parámetros de entrada, el resultado es un plano que para cada par de valores (superficie, habitaciones) predice una predicción del precio con la siguiente ecuación.

$$y = X\theta = \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \quad (Dado\ x_1 = 1)$$

Donde x_2 es la superficie y x_3 las habitaciones.

Al evaluar el modelo con los datos de test y la métrica MRE se obtiene el resultado 0.2296.

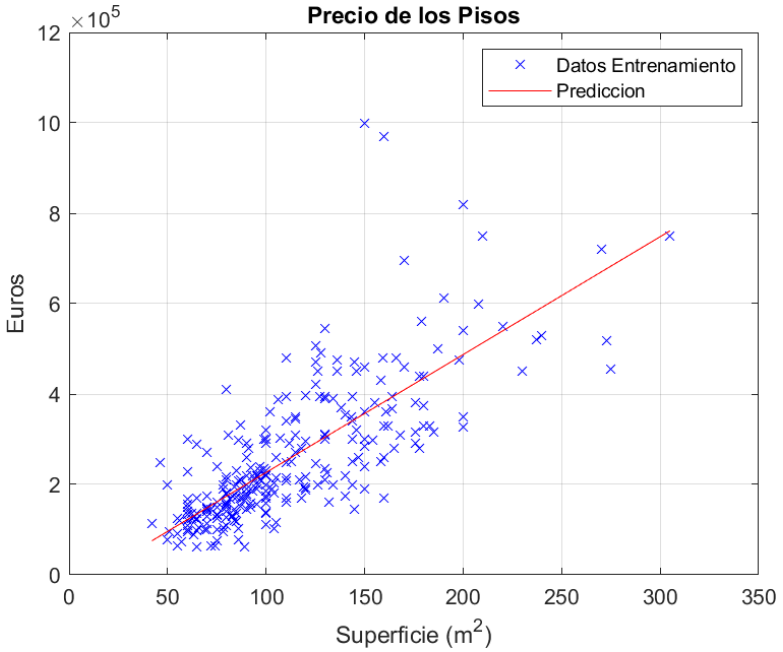
Regresión monovariable con descenso de gradiente

En este apartado se busca la θ óptima mediante un método iterativo de descenso de gradiente.

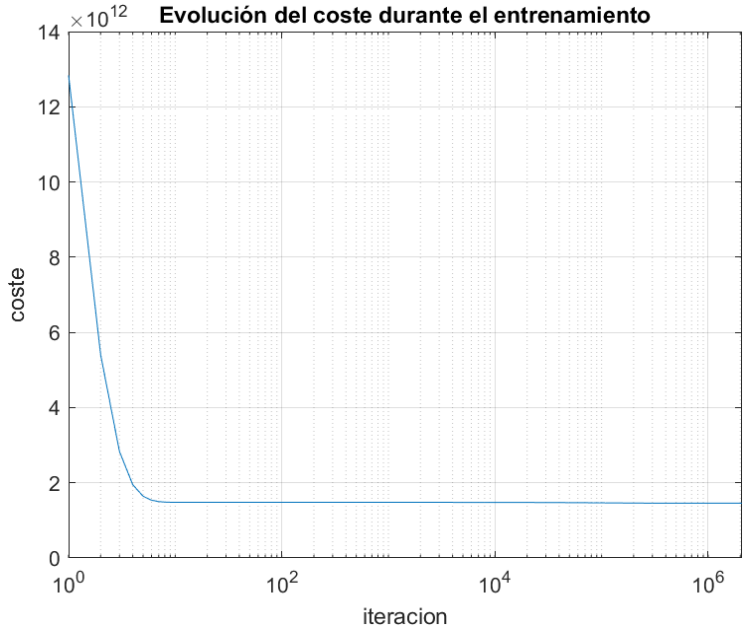
Para cada iteración se calcula el gradiente y se utiliza, multiplicado por un factor de aprendizaje α, para modificar el peso. Se utiliza como condición de parada la diferencia entre iteraciones del coste, si este valor es menor que un parámetro ε se deja de iterar. En la implementación también se ha añadido un parámetro *max_iters* que indica el número máximo de iteraciones que se realizarán para asegurar la terminación del programa.

Se utilizan números pseudoaleatorios como valores iniciales de los elementos de θ.

Hay que encontrar manualmente un valor de α que funcione para el problema.
Una ejecución con $\alpha=1*10^{-7}$ y $\varepsilon=0.01$ termina tras poco más de 2 millones de iteraciones y obtiene los valores de θ para la siguiente ecuación de la recta, casi idénticos a los encontrados mediante la ecuación normal con los que se obtiene el mismo coste MRE de 0.2744.



Se puede comprobar la convergencia dibujando la gráfica de evolución del coste. Para visualizarlo de forma más cómoda, se puede utilizar escala logarítmica en el eje x.

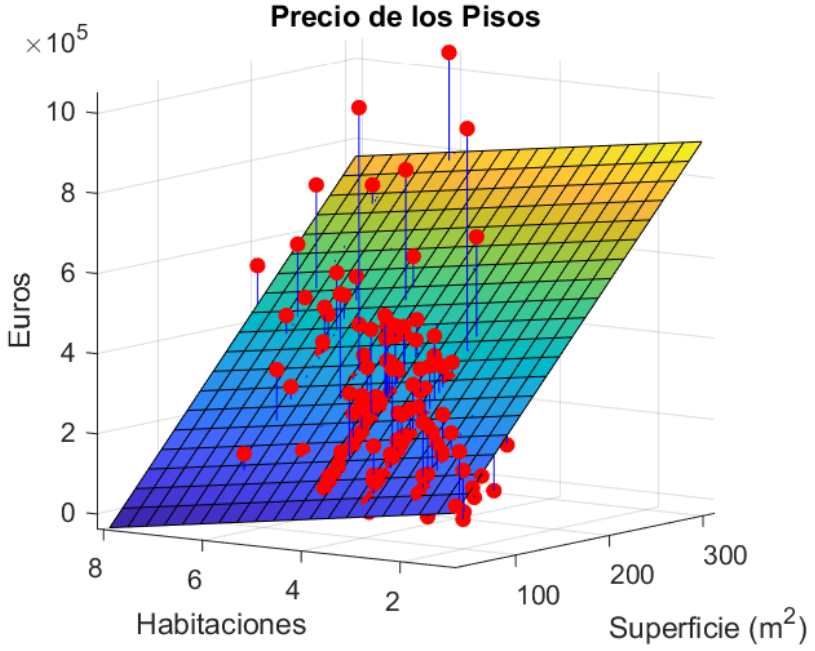


Se observa que se alcanza un resultado muy bueno después de muy pocas iteraciones. Unas 100 iteraciones serían suficientes para obtener una recta relativamente buena en la mayoría de los casos. Sin embargo, para llegar al punto en el que el coste se modifique en 0.01 entre iteraciones se aumenta el número de iteraciones en varios órdenes de magnitud.

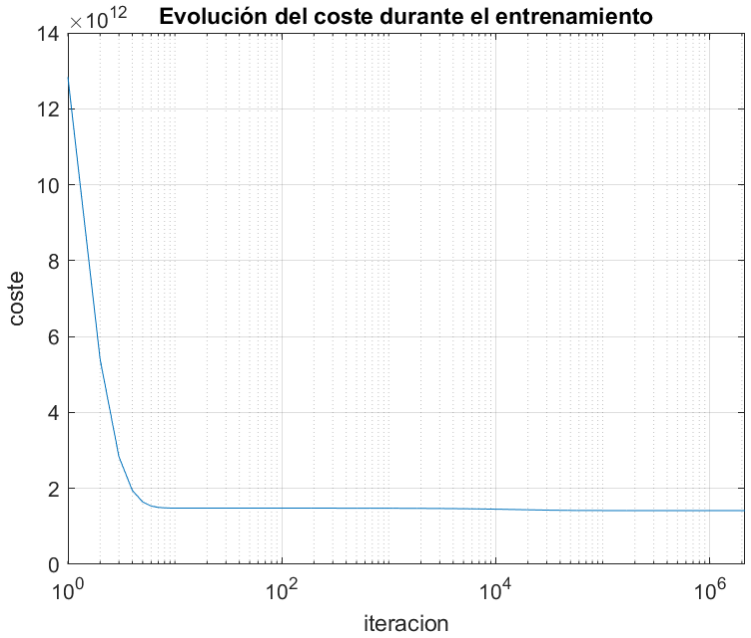
Para este problema con una o dos variables esto no supone un problema ya que termina igualmente en escasos segundos.

Regresión multivariable con descenso de gradiente

La escala de los distintos atributos no es para nada la misma, lo que significa que unos atributos tienen más impacto que otros en la función de coste, y por tanto, a la hora de entrenar el modelo. Escalar los atributos puede ayudar a facilitar la convergencia del descenso de gradiente. Utilizando los mismos parámetros de entrenamiento que en el apartado anterior se obtienen los siguientes resultados.



A continuación se muestra la evolución del coste. Mantiene una forma similar a la del apartado anterior y se pueden extraer las mismas conclusiones.

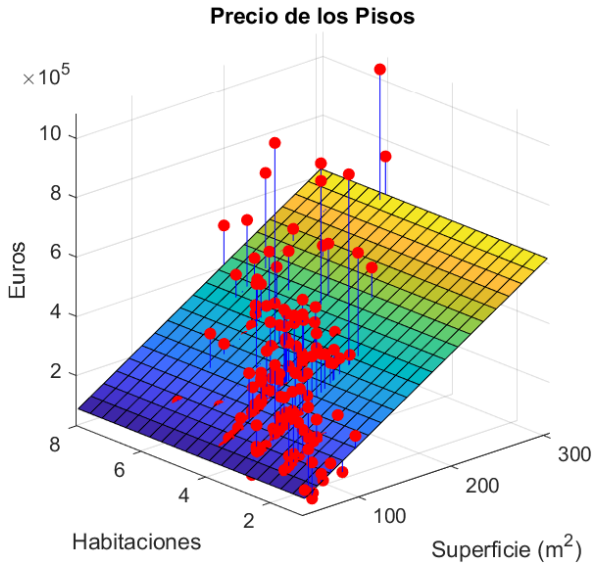


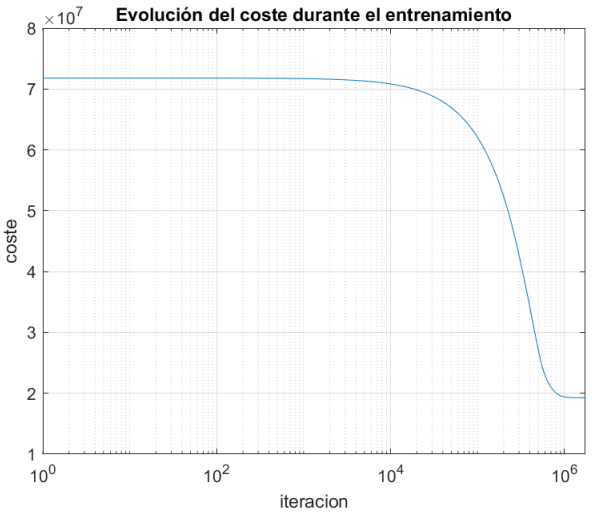
El coste MRE calculado con los datos de test es de 0.2296, una vez más idéntico al resultado obtenido con la ecuación normal.

Regresión multivariable con descenso de gradiente y regresión robusta

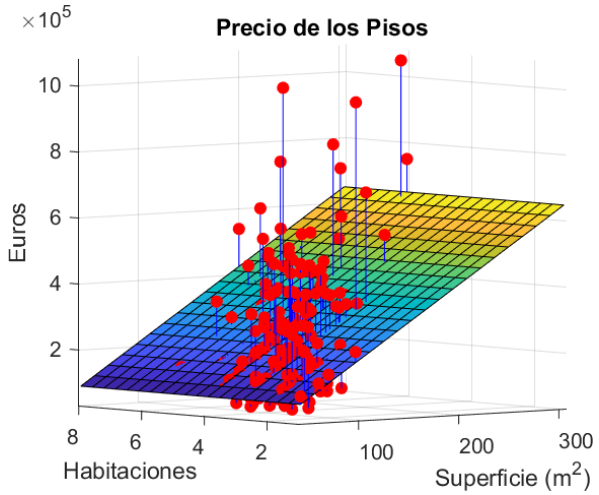
En lugar de utilizar un coste cuadrático se va a usar el coste de Huber, que es igualmente derivable y no da más influencia a los errores grandes de los espurios. Hay un nuevo parámetro δ que ajustar, que determina cuánto influyen los datos con mayor residuo. Un δ cercano a 1 produce costes más altos que resulta en un entrenamiento más rápido, pero más afectado por espurios (y por tanto con un coste final sobre los datos de test más alto). A medida que se disminuye δ, los datos con mayor residuo tienen menos efecto sobre el coste, por lo que el número de iteraciones para obtener un resultado aumenta, pero la función del plano obtenida se ajusta mejor a los datos de entrenamiento y produce mejores predicciones sobre los datos de test.

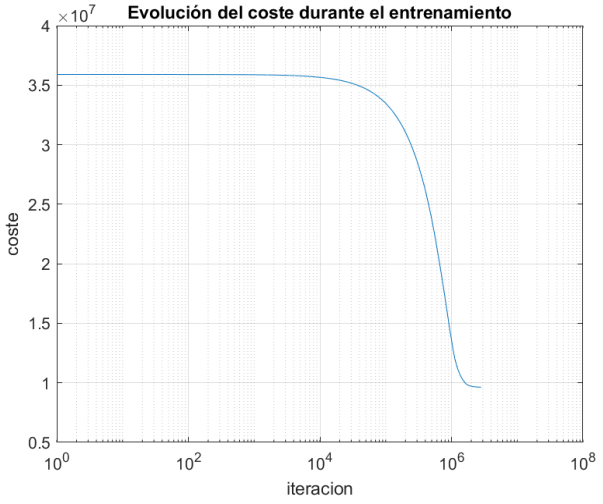
δ=1: Iteraciones: 1743739 Coste MRE: 0.2453



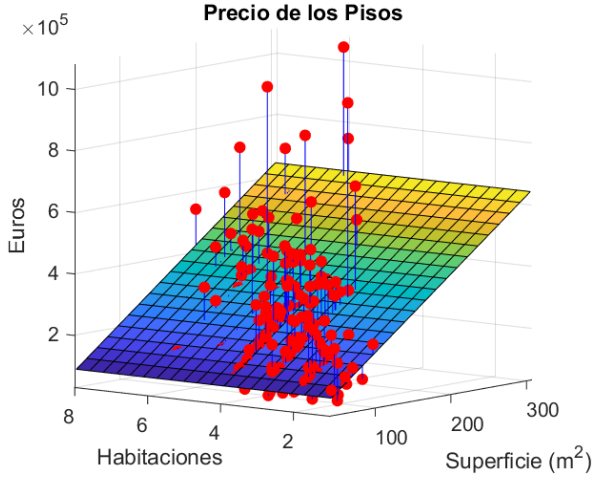


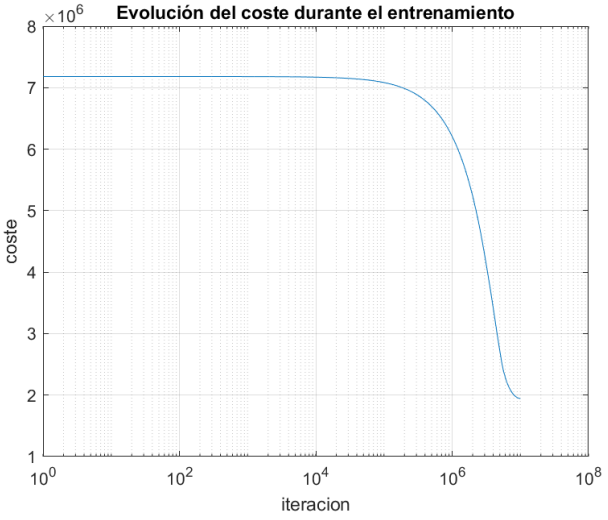
δ=0.5: Iteraciones: 2829475 Coste MRE: 0.2409





δ=0.1: Iteraciones: 10264346 Coste MRE: 0.2258





Conclusiones finales

Aunque no tiene sentido comparar modelos con distintos números de parámetros entre sí, las medidas de coste tomadas sí que nos permiten comparar las distintas implementaciones para un mismo problema. A lo largo de la práctica se ha demostrado que se pueden alcanzar resultados casi idénticos con el algoritmo de descenso de gradiente y con la ecuación normal. En estos ejercicios se ha utilizado un número de parámetros muy bajo, por lo que la ecuación normal obtiene resultados igual de buenos y en menos tiempo que el algoritmo de descenso de gradiente, sin embargo, la ecuación normal no es buena para un número muy alto de parámetros porque incluye calcular la inversa de una matriz, que es muy costosa y necesita muchas operaciones que al ser de coma flotante se va perdiendo precisión, por lo que sería difícil utilizarla en problemas más grandes.

De forma similar, en esta práctica no se ha normalizado los datos de entrada porque no era necesario para obtener buenos resultados en un tiempo razonable, aunque hubiese sido beneficioso por la diferencia entre las escalas de los distintos parámetros, pero sería indispensable en otro tipo de problemas.

Por otro lado, se ha demostrado que utilizar regresión robusta puede dar lugar a un modelo más preciso, especialmente en casos con muchos espurios entre los datos de entrada. Esto es a cambio de ralentizar el proceso de entrenamiento del modelo. A medida que se le da menos influencia a las entradas, más cuesta que se adapte θ.