Práctica 1
Programación

Aprendizaje Automático - GII
Grupo 2
Curso 19-20

Universidad de Granada
Cumbreras Torrente, Paula
49087324-B

Índice de contenidos

Ejercicio sobre la búsqueda iterativa de óptimos

Apartado 1

El Descenso del Gradiente es un algoritmo de entrenamiento cuyo objetivo es la búsqueda de un óptimo local siguiendo la dirección del gradiente en cada iteración:

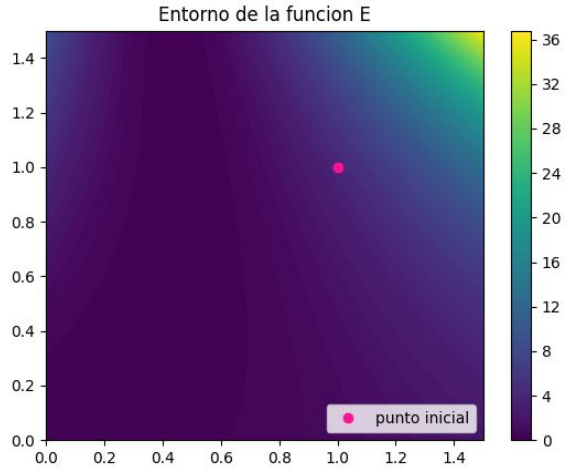
$$w_j = w_j - \eta \frac{\partial E_{in}}{\partial w_j}$$

Siendo w_j los pesos (o coordenadas del punto estudiado en la iteración j), η la tasa de aprendizaje y E_{in} la función que buscamos optimizar. El algoritmo desarrollado parte de un punto inicial e itera hasta que se repita un número determinado de veces o hasta que se llegue a un punto cuyo valor en la función sobrepase un valor buscado. Devuelve las coordenadas del punto alcanzado (w), el número de iteraciones realizadas y dos vectores donde se han ido almacenando, respectivamente, $E_{in}(w_j)$ y j.

Apartado 2

La función a estudiar es la siguiente:

$$E(u,v) = (ue^{v} - 2ve^{-u})^2$$

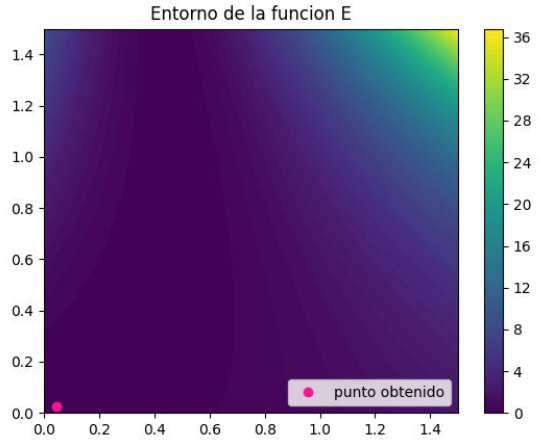


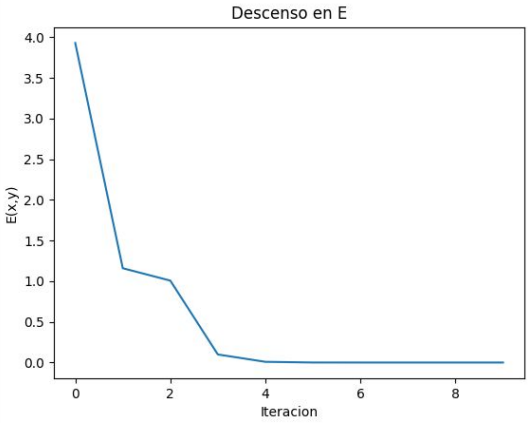
A la izquierda podemos observar el entorno de la función E. Vemos que las zonas más claras corresponden a valores más altos mientras que las zonas más oscuras tienden a 0. Se ha dibujado a su vez el punto inicial desde el que se ejecutará el gradiente descendente: puesto que buscamos hallar un óptimo mínimo, tras ejecutar ello deberíamos alcanzar un punto que se encuentre en una región más oscura en el entorno estudiado.

Con una tasa de aprendizaje de 0.1 y buscando un valor de E inferior a 10^{-14}, se obtienen los siguientes resultados:

```
Número de iteraciones:  10
Coordenadas obtenidas: ( 0.044736290397782055 ,  0.023958714099141746 )
```

En las siguientes gráficas se puede observar como el punto obtenido se encuentra en la zona más oscura del entorno de la función y dado que solo se realizaron 10 ejecuciones, podemos afirmar el el valor que toma la función E en ese punto es inferior a 10^{-14} (gráfica de la izquierda). Además se añade el descenso que ha sufrido el valor de E en cada iteración realizada por el gradiente descendente (gráfica de la derecha)

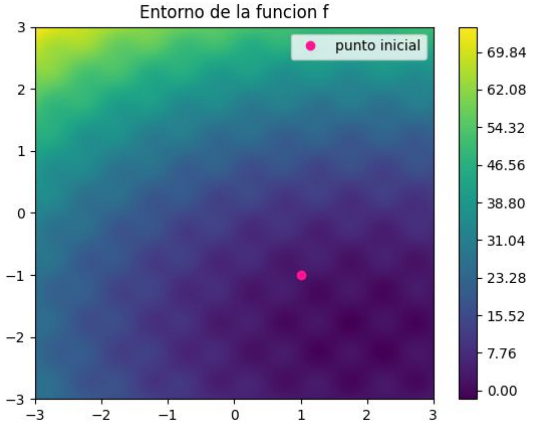




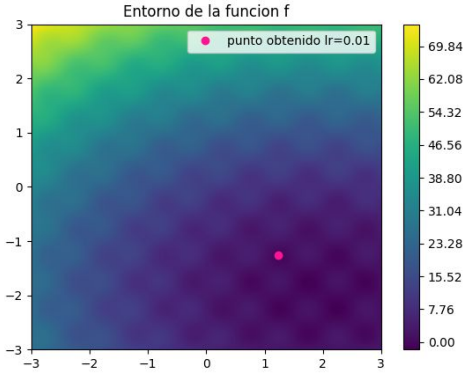
Apartado 3

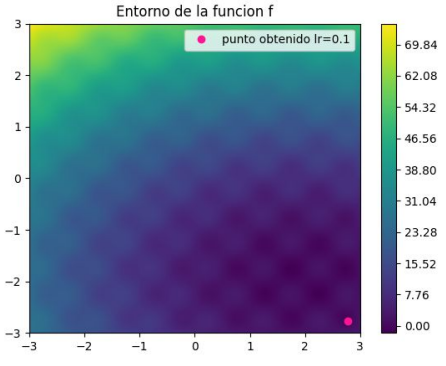
La siguiente función a estudiar es:

$$f(x,y)=(x-2)^2+2(y+2)^2+2\sin(2\pi x)\sin(2\pi y)$$



A la izquierda se muestra el entorno de la función f. En esta ocasión se toma como punto inicial el (1,-1) y se establece un máximo de iteraciones: 50. Estudiaremos el comportamiento del gradiente descendente en función de la tasa de aprendizaje elegida. Se muestran a continuación los puntos obtenidos con las tasas de aprendizaje 0.01 y 0.1, respectivamente

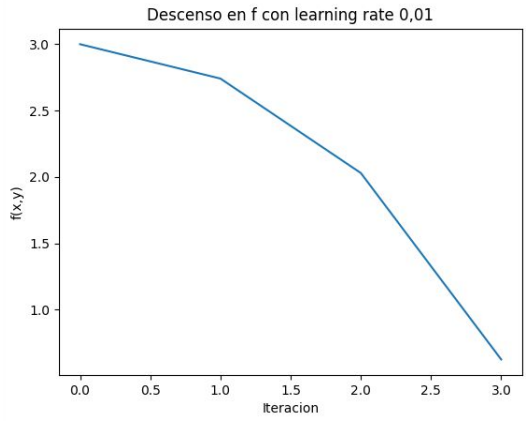


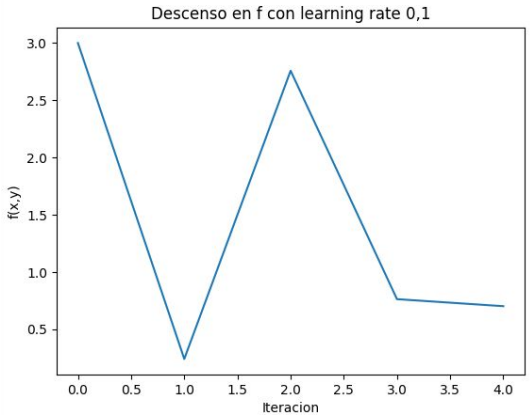


Learning rate igual a 0.01 Valor obtenido: 9.078452587988094	Learning rate igual a 0.1 Valor obtenido: 8.504780160190041

Viendo los valores obtenidos se puede afirmar que para esta función, partiendo del punto (1,-1) y con un máximo de 50 iteraciones la tasa 0.1, mayor que 0.01, obtiene un mejor resultado. Sin embargo, si observamos el descenso del valor de la función f en cada iteración del gradiente descendente (mostrado en las gráficas siguientes), observamos que una tasa tan

grande podría ciclar indefinidamente acercándose y alejándose del mínimo deseado. Sin embargo, una tasa de aprendizaje muy pequeña tendría un descenso muy lento y, si no se establece un máximo de iteraciones, haría que el algoritmo fuera ineficiente.





Se pide también ejecutar el gradiente descendente partiendo de una serie de puntos y recoger en una tabla las soluciones obtenidas:

Learning rate igual a 0.01

Punto inicial	Valor obtenido	Coord X	Coord Y
(2.1 , -2.1):	-1.8200785415471563	2.2438049693647883	-2.237925821486178
(3.0 , -3.0):	-0.38124949743809955	2.7309356482481055	-2.7132791261667037
(1.5 , 1.5):	18.042077497699836	1.777977600113794	1.0320075625733127
(1.0 , -1.0):	-0.3812494974381	1.269064351751895	-1.2867208738332965

Learning rate igual a 0.1

Punto inicial	Valor obtenido	Coord X	Coord Y
(2.1 , -2.1):	2.3457572351777434	3.1628233949667752	-1.5573141417090974
(3.0 , -3.0):	0.6724512149690361	2.23295375224537	-1.950108304646177
(1.5 , 1.5):	1.4843586838918261	1.0476671341326078	-1.2366418536799464
(1.0 , -1.0):	0.060882024317680195	0.8653565726444672	-1.6958777754413352

Apartado 4

La verdadera dificultad para hallar un mínimo global en una función arbitraria empleando este algoritmo es saber elegir el punto de partida. Como se puede observar en los datos recopilados en la tabla del apartado anterior, en el punto (1.5,1.5) con una tasa de aprendizaje, se obtiene un valor mayor al obtenido en el punto (3,-3) con la misma tasa, pero éste sigue sin ser un óptimo global. Como se ha comentado anteriormente, no solo se necesita determinar el punto inicial, sino que la correcta configuración de la tasa de aprendizaje también resulta fundamental para que el algoritmo funcione bien.

Ejercicio sobre Regresión Lineal

Apartado 1

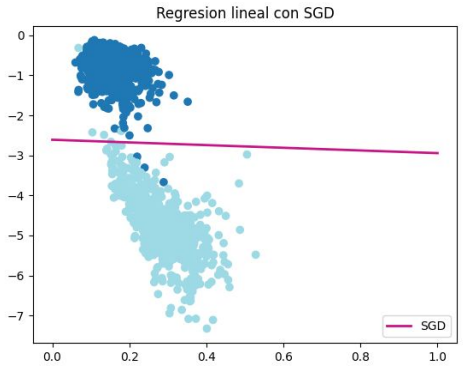
En este ejercicio estudiamos la regresión lineal sobre dos algoritmos: Pseudo-inversa y Gradiente Descendente Estocástico. La regresión lineal es un modelo matemático que trata de representar la tendencia de unas variables independientes o explicativas sobre variables dependientes o explicadas.

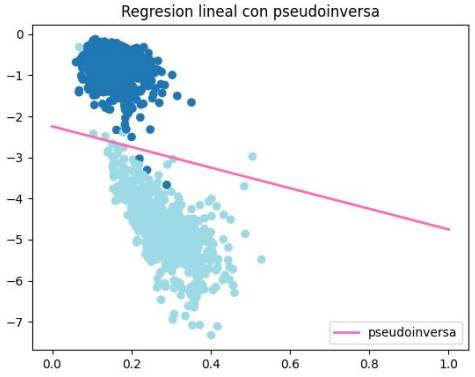
El algoritmo de la Pseudo-inversa permite resolver sistemas de mínimos cuadrados y viene dado por la siguiente fórmula:

$$R=A^t(AA^t)^{-1}$$

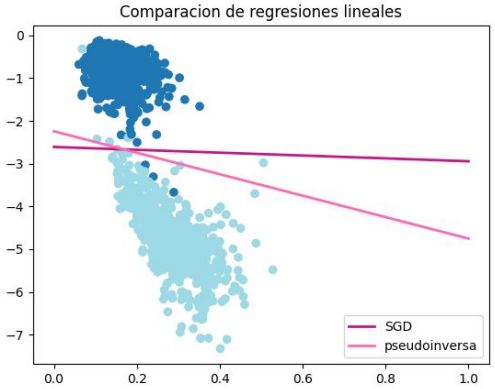
El Gradiente Descendente Estocástico (SGD), por otro lado, es una variante del algoritmo del gradiente descendente en la que se estudian pequeñas muestras del conjunto de datos para disminuir el coste computacional.

A partir del conjunto de datos proporcionados para la realización de la práctica, se obtienen las siguientes regresiones lineales para los algoritmos SGD y Pseudo-inversa



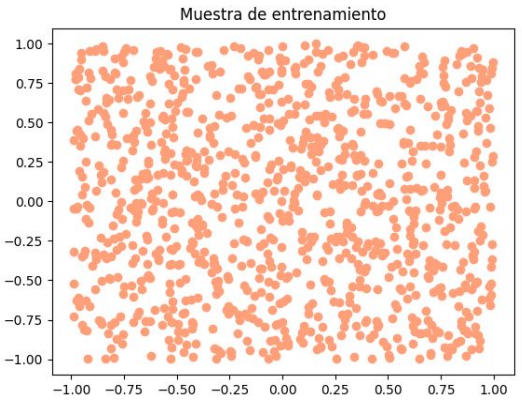


Bondad del resultado para grad. descendente estocástico:	Bondad del resultado para el algoritmo de la pseudoinversa:
Ein: 0.08447326519972755 Eout: 0.14321774186493383	Ein: 0.07918658628900388 Eout: 0.13095383720052575



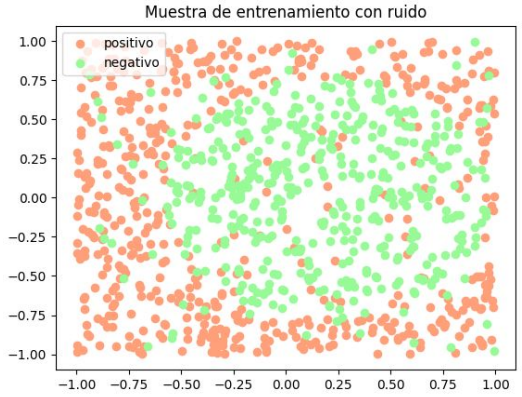
En la gráfica de la izquierda se muestra una comparación entre ambas regresiones. Se puede observar que tanto el error real (Ein) como el estimado (Eout) obtenidos en el resultado de la regresión lineal para el algoritmo de la pseudo-inversa son menores que para el SGD, por lo que su bondad es mayor.

Apartado 2



Para realizar el experimento debemos generar una muestra de entrenamiento sobre la que trabajaremos. En la gráfica de la derecha se muestran los 1000 puntos de la muestra generados en el cuadrado [-1,1]x[-1,1]. Obtenemos entonces las etiquetas aplicando sobre ellos la función dada por:

$$f(x,y) = \text{sign}((x-0{,}2)^2+y^2-0{,}6)$$



Tras esto aplicamos ruido, esto es, le cambiamos el signo a un 10% aleatorio de las etiquetas obtenidas. La muestra de entrenamiento obtenida tras esto es la mostrada en la gráfica, y sobre ella se ha ejecutado el SGD y se ha realizado un ajuste lineal. El error real obtenido ha sido

```
Ein:  1.0125017833744723
```

Tras ejecutar todo este proceso 1000 veces, se obtiene la siguiente bondad media de los resultados obtenidos

```
Errores Ein y Eout medios tras 1000reps del experimento:

Ein media:  1.0172888438227266
Eout media:  1.0292036645043086
```

Se puede observar que los valores medios obtenidos son considerablemente grandes en comparación con los obtenidos en el apartado anterior. Esto se debe, principalmente, a que el conjunto de datos del que partimos no sigue un comportamiento lineal y además ha sido aleatoriamente modificado.

Bibliografía

- https://docs.scipy.org
- http://www.sc.ehu.es
- https://arxiv.org
- https://es.wikipedia.org
- http://www.cs.us.es
- https://www.pyimagesearch.com