Practica 1

Aprendizaje Automatico

José Manuel Pérez Lendínez

Ejercicio sobre la búsqueda iterativa de óptimos

1. Implementar el algoritmo de gradiente descendente.

Para esto he realizado una función a la que se le pasa como parámetros las derivadas parciales, la tasa de aprendizaje y una coordenada y devuelve el gradiente a ese punto

```
def gradiante(der_parcial_u,der_parcial_v, \
u, v, tasaAprendizaje, coordenada):
        valor_u = coordenada[0] - tasaAprendizaje \
        * np.float64(der_parcial_u.subs({u:coordenada[0],
         v: coordenada[1]}))
        valor_v = coordenada[1] - tasaAprendizaje \
        * np.float64(der_parcial_v.subs({u:coordenada[0],
        v: coordenada[1]}))
        return valor_u,valor_v
```

Esta función es la que llamaremos para calcular el gradiente iterativa mente dentro de un while en los siguientes ejercicios. Ejemplo del ejercicio 3

```
datos[0][0] = inicio[0]
datos[0][1] = inicio[1]
datos[0][2] = lam_formula(inicio[0],inicio[1])
i = 1;
valor_u,valor_v = gradiante(der_parcial_u, der_parcial_v, u, v,
    tasaAprendizaje,inicio)

while i < num_iteraciones:
        datos[i][0] = valor_u
        datos[i][1] = valor_v
        datos[i][2] = lam_formula(valor_u,valor_v)

        valor_u,valor_v = gradiante(der_parcial_u, der_parcial_v, u,
            v, tasaAprendizaje,[valor_u,valor_v])
        i = i + 1
```

La funciones para calcular el gradiente respecto a u y v del ejercicio 1 son:

$$u = u - \eta \frac{\partial f(u,v)}{\partial u} \tag{1}$$

$$v = v - \eta \frac{\partial f(u,v)}{\partial v} \tag{2}$$

2.Considerar la función:

$$E(u,v) = (u^2e^v - 2v^2e^{-u})^2 \tag{3}$$

Usar gradiente descendente para encontrar un mínimo de esta función, comenzando desde el punto (u,v)=(1,1) y usando una tasa de aprendizaje = 0,01

a.-Calcular analíticamente y mostrar la expresión del gradiente de la función E(u, v)

Sacamos primero las derivadas parciales de u y v.

$$\frac{\partial E(u,v)}{\partial u} = 2(u^2e^v - 2v^2e^{-u})(2e^vu + 2v^2e^{-u}) \tag{4}$$

$$\frac{\partial E(u,v)}{\partial v} = 2(u^2e^v - 2v^2e^{-u})(u^2e^v - 4e^{-u}v) \tag{5}$$

Lo siguiente sera poner el resultado que daría siendo (u,v)= (1,1) que daría como resultado (0.7552645787017029,0.950565232703529).

b.-¿Cuántas iteraciones tarda el algoritmo en obtener por primera vez un valor E(u,v) inferior a 10^{-14}?

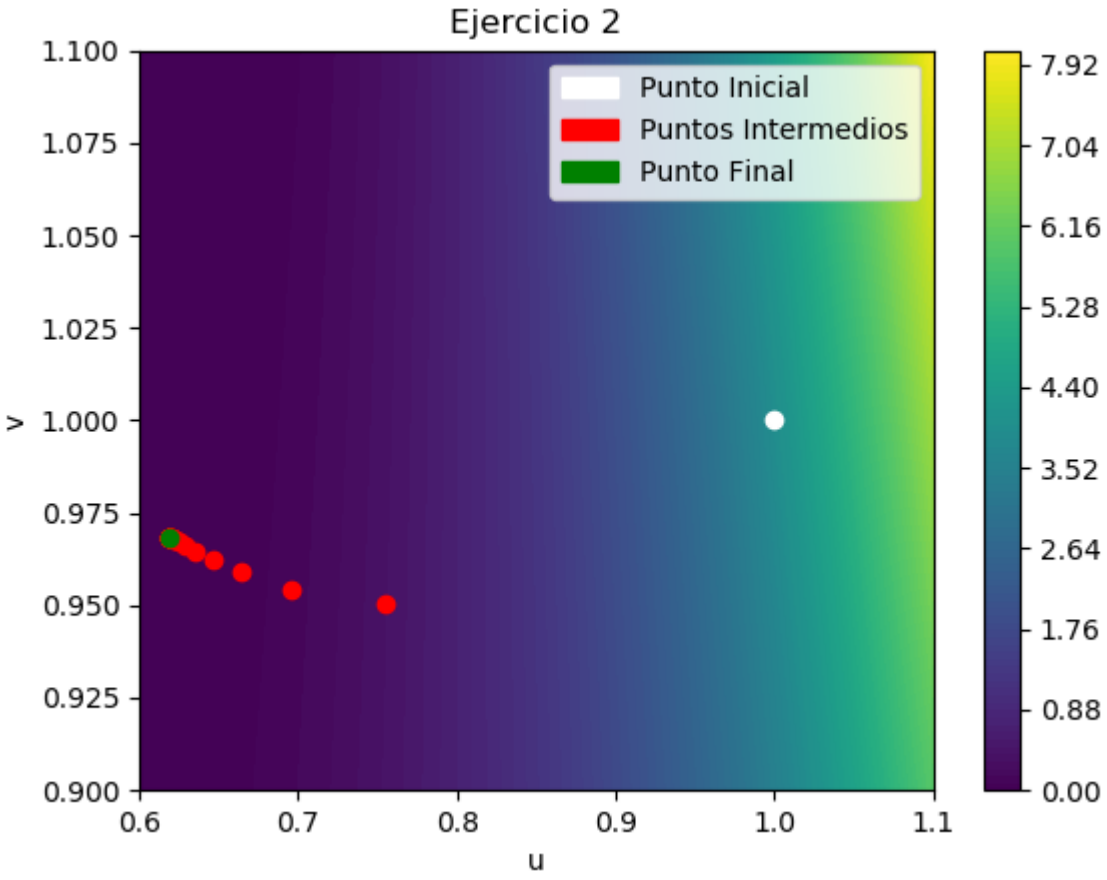
Se encuentra en 34 iteraciones contando la inicial de (1,1)

c-.¿En que coordenadas (u,v) se alcanzó por primera vez un valor E(u,v) inferior 10^{-14}?

Se encuentra en el punto (0.6192076784506378,0.9684482690100485)

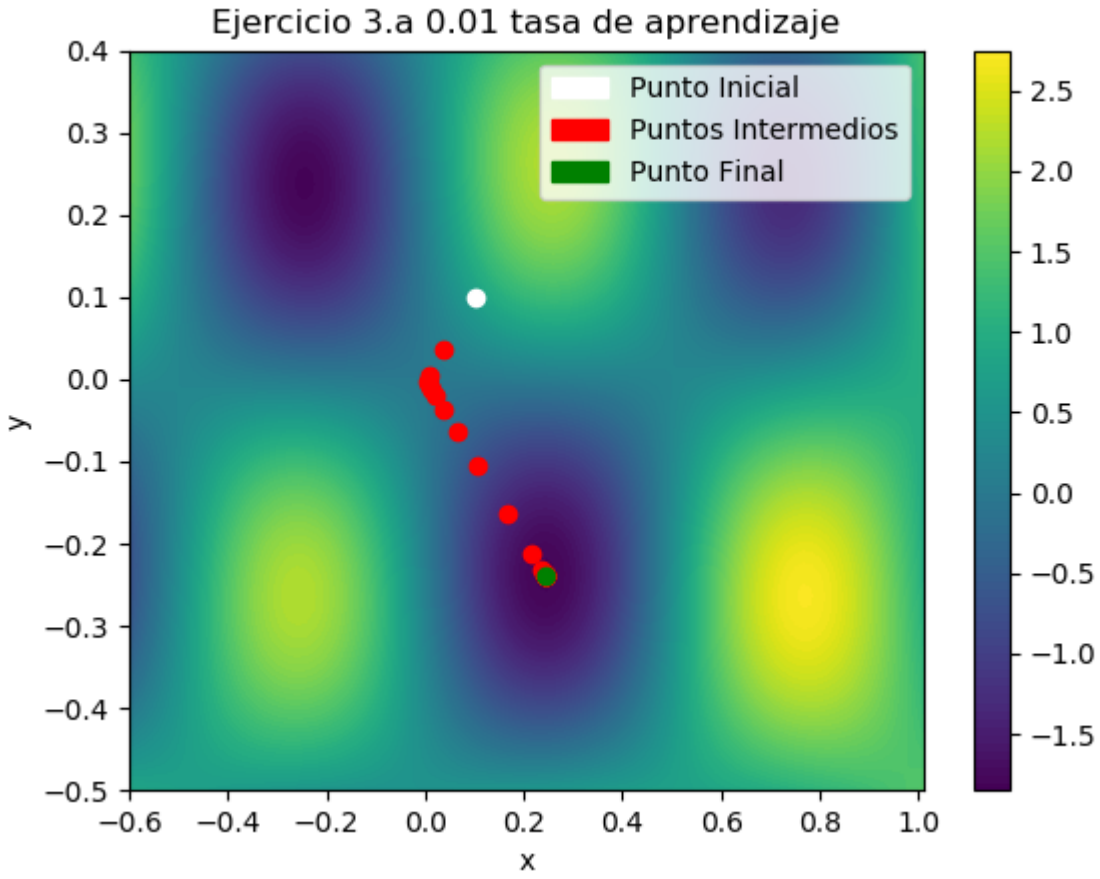
Opcional.

Aunque no se pide voy a mostrar una grafica de este ejercicio para ver como va evolucionando el gradiente descendente.

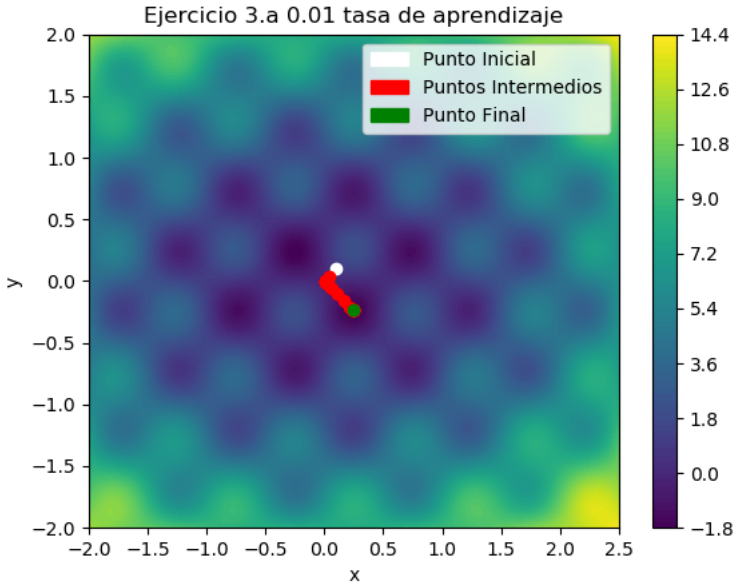


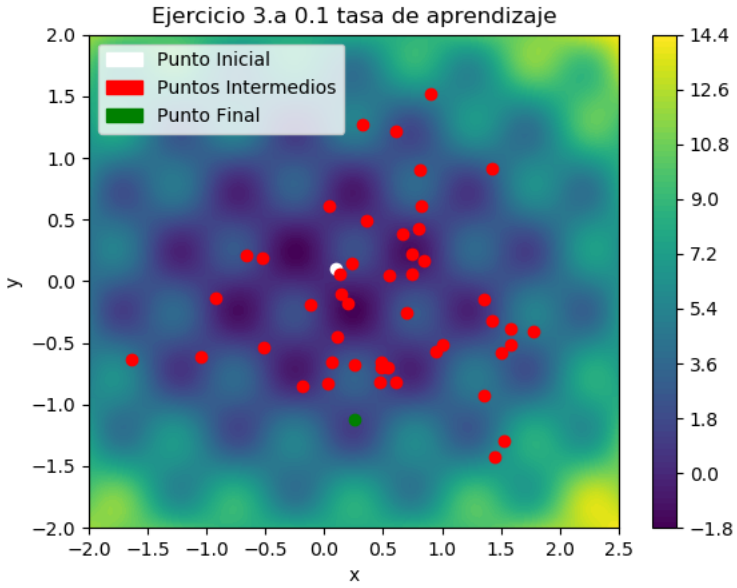
3. Considerar ahora la función f(x,y) = $x^2+2y^2+2sin(2\pi x)2sin(2\pi y)$
a.- Usar gradiente descendente para minimizar esta función. Usar como punto inicial (x = 0.1 y y = 0.1), tasa de aprendizaje de 0.01 y 50 iteraciones. Generar un grafico de como desciende le valor con las iteraciones. Repetir con una tasa de aprendizaje de 0.1

Vamos a mostrar como se llega con 0.01 de tasa de aprendizaje en la siguiente grafica.



Ahora vamos a comparar la anterior con la grafica con tasa de 0.1 ampliando la anterior para verlo mas claro.





Como se ve en el caso de la tasa de 0.1 es demasiado grande y nunca consigue converger sino que va dando saltos de un lado a otro.

b.- Vamos a generar una tabla con los valores que se obtienen con distintos inicios para el algoritmos. Los puntos de inicio serán desde (0.1,0.1), (1,1), (-0.5,-0.5) y (-1,-1)

Valores Inicio	X	Y	resultado
(0.1,0.1)	0.24380496936476242	-0.2379258214861717	-1.8200785415471565
(1,1)	1.2180702996828958	0.7128119496190772	0.5932693743258357
(-0.5,-0.5)	-0.7313774603925802	-0.2378553629018492	-1.332481062330978
(-1,1)	-1.2180702996828958	-0.7128119496190772	0.5932693743258357

4. ¿Cual seria su conclusión sobre la verdadera dificultad de encontrar el mínimo global de una función global de una función arbitraria?

El primer problema es que si la función tienes muchos mínimos locales con gradiente descendiente seria difícil alcanzar el mínimo global. Puesto que esto dependería de la posición inicial que se le diera al algoritmo. Con distintas posiciones se pueden dar distintos resultados.
Otra dificulta es decidir una buena tasa de aprendizaje ya que en algunos problemas podría ser mejores unas que otras.

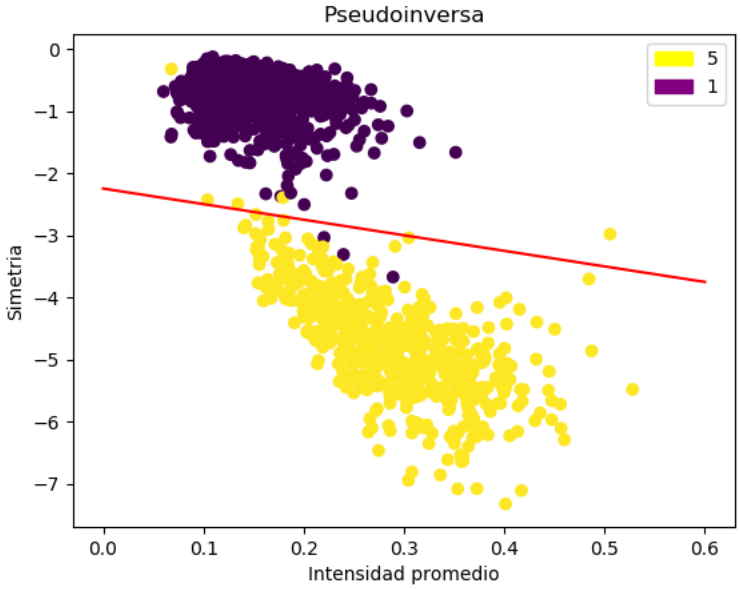
Ejercicio sobre regresión lineal

1. Estimar un modelo de regresión lineal a partir de los datos proporcionados de dichos numero(Intensidad promedio, Simetría) usando tanto el algoritmo de la pseudoinversa como gradiente descendente estocástico. Las Etiquetas serán -1,1, una para cada vector de cada uno de los números. Pintar las soluciones obtenidas junto con los datos usados en el ajuste. Valorar la bondad del resultado usando Ein y Eout (para Eout calcular las predicciones usando los datos del fichero test).

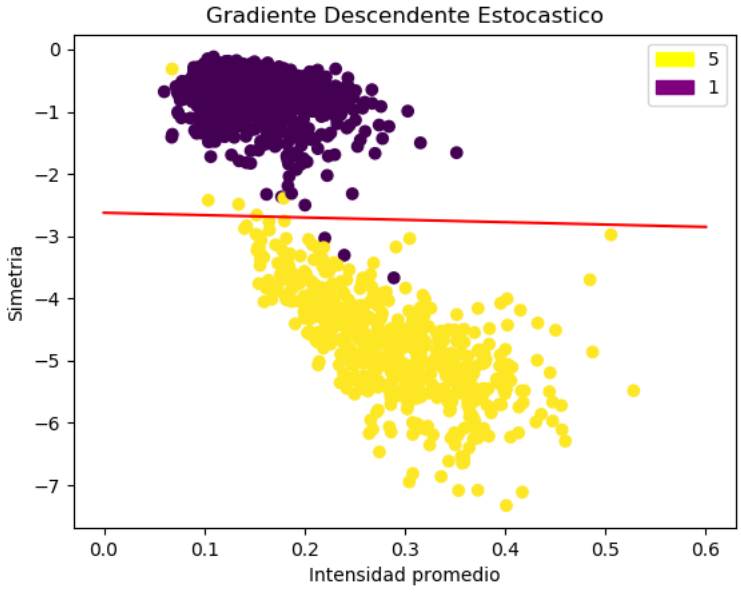
Para la pseudoinversa solamente tendremos una función a la que se le pasara como parámetro las datos de entrenamientos y las clases de los datos. A continuación aprovechamos las operaciones con matrices para realizar la formula de la pseudoinversa.Utilizaremos x para los datos e y para las clase.

$$(x^t x)^{-1} x^t y \tag{6}$$

Esta formula nos devuelve los pesos validos para nuestro problema. Los pesos son (-1.11588016 -1.24859546 -0.49753165) y la grafica de regresion es la siguiente:



Para el método del gradiente descendiente estocástico se consigue por medio de mini-bacth y de utilizar el algoritmo del gradiente descendente. Se vasa en separar los datos en pequeñas secuencias de desordenados. Para esto se desordenan los datos de entrenamiento y cogemos muestras de un tamaño especificado como parámetro. Estos datos se procesan de forma parecida a el gradiente descendiente obteniendo. También es necesario pesarle como parámetro el valor de la tasa de aprendizaje y un numero de iteraciones. Cuando se trabajan con todos los datos que teníamos desordenados se vuelven a desordenar los datos para seguir cogiendo mini-batch. El algoritmo para después de un numero de iteraciones. El problema de este algoritmo es que no siempre da los mismos resultados y en unas iteraciones me ha dado valores mas próximos a los de la pseudoinversa y en otros unos bastantes mas alejados. Para intentar evitar este problema he ido almacenando el conjunto de pesos que mejor error me ha dado con los datos de prueba. En este caso en el ejemplo nos ha dado como pesos unos mas próximos a la pseudoinversa. Los pesos son ()-1.24197953 -0.179284 -0.47334248)



Los errores medidos son los siguientes. He cogido dos muestras del gradiente descendente estocástico, una con buenos resultados y otra con peores resultados

Algoritmo	Ein	Eout
Pseudoinversa	0.07918658628900431	0.13095383720052586
GDE (Buenos Resultados)	0.08362802132720665	0.14205101406628193
GDE (Malos Resultados)	0.12754745375905444	0.20844389603082997

Con esto llegamos a la conclusión de que siempre que sea posible utilizar la pseudoInversa y se cumplan en los datos las restricciones para utilizar esta sera la que necesitaremos elegir. Aunque comprobar si se cumplen las restricciones de este problema puede ser muy costoso. En cambio el gradiente descendente estocástico nos asegura que puede ser utilizado en cualquier conjunto de datos aunque no de resultados tan buenos como la pseudoinversa.