Práctica 1: Aprendizaje Automático

1. Ejercicios sobre la búsqueda iterativa de óptimos

Eloy Bedia García & Miguel Moles Hurtado

10 de Marzo de 2018

APARTADO 1

EJERCICIO 1

1. Implementar el algoritmo de gradiente descendente.

$var \leftarrow Punto_{inicio}$

$f : \Re^n \rightarrow \Re$

$fderivadas \leftarrow (\frac{\partial f}{\partial x_0}, ..., \frac{\partial f}{\partial x_i}, ..., \frac{\partial f}{\partial x_n})$

$\eta \leftarrow Tasa_{aprendizaje}$

Sea $a \in \Re^n$ el anterior punto evaluado segun BGD, $\epsilon \in \Re \mid |f(x) - f(a)| \geq \epsilon$

$limit \leftarrow Max_{iteraciones}$

```
BatchGradientDescent = function(var, f, fderivadas, mu, epsilon = -Inf, limit = Inf) {
  #el vector 'y' sera utilizado para almacenar las nuevas componentes
  #Si las guardaramos directamente en 'var' alterariamos el punto a evaluar en ese momento.
  y <- var
  dim(y) <- dim(var)
  iteraciones <- 0

  eval_anterior <- Inf
  while(iteraciones < limit && abs(eval_anterior - eval(f)) >= epsilon)  {

    #Recorremos cada una de las componentes independientes de la funcion
    #Para cada componente de la funcion hay una derivada parcial
    for(i in 1:dim(var)) {
      #Evaluamos la derivada en el punto 'var' y seguimos el sentido negativo de la misma
      #Si eval(fderivada[i]) < 0 la variable 'var[i]' aumentara siguiendo el sentido negativo
      #Si eval(fderivada[i]) > 0 la variable 'var[i]' disminuira siguiendo el sentido negativo
      #Si eval(fderivada[i]) = 0 la variable 'var[i]' habra llegado a su valor optimo
      y[i] <- var[i] - mu * eval(fderivadas[i])
    }

    eval_anterior <- eval(f)
    #Actualizamos el punto que evaluaremos
    var <- y
    iteraciones <- iteraciones + 1

  }

  list(y,iteraciones)
}
```

EJERCICIO 2

2. Considerar la función $f(u, v) = (u^3e^{v-2} - 4v^3e^{-u})^2$. Usar gradiente descendente para encontrar un mínimo de esta función, comenzando desde el punto $(u, v) = (1, 1)$ y usando una tasa de aprendizaje $\eta = 0,05$.

a) Calcular analíticamente y mostrar la expresión del gradiente de la función $f(u, v)$

$\nabla f = (\frac{\partial(u^3e^{v-2}-4v^3e^{-u})^2}{\partial u}, \frac{\partial(u^3e^{v-2}-4v^3e^{-u})^2}{\partial v})$

$\nabla f = (2(u^3e^{v-2} - 4v^3e^{-u})(3u^2e^{v-2} + 4v^3e^{-u}), 2(u^3e^{v-2} - 4v^3e^{-u})(u^3e^{v-2} - 12v^2e^{-u}))$

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

```
## Para alcanzar el minimo de la función f, el algoritmo utilizado invierte  36  iteraciones
```

c) ¿En qué coordenadas (u, v) se alcanzó por primera vez un valor igual o menor a 10^{-14} en el apartado anterior.

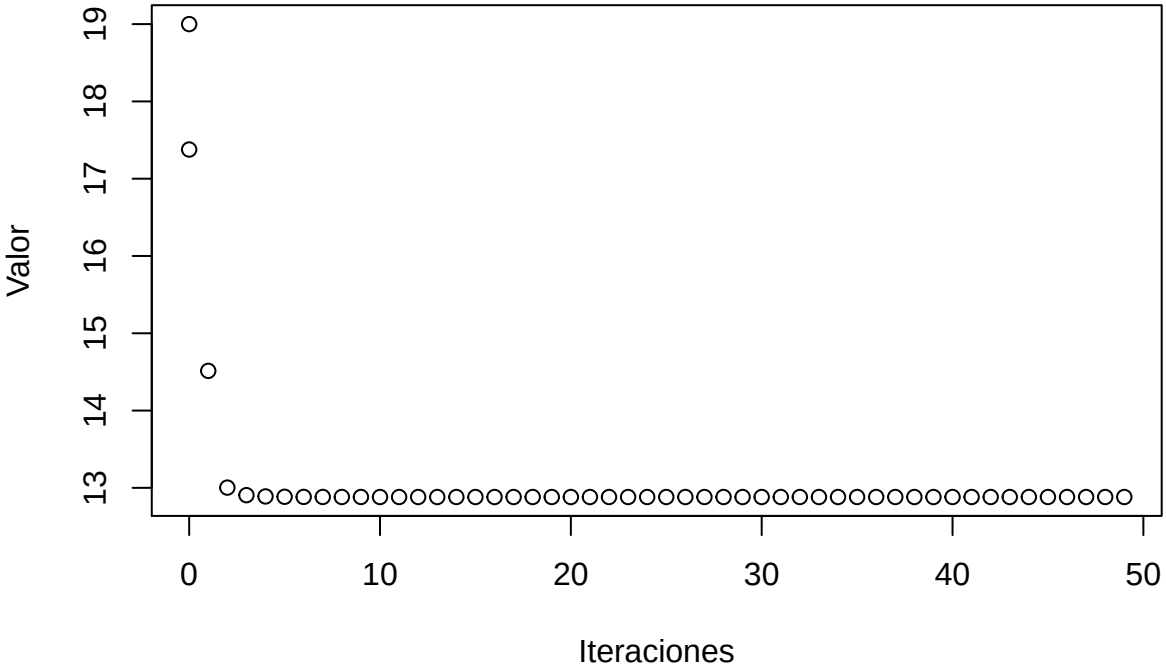
```
## En el punto ( 1.119544 , 0.653988 ). El valor f(u,v) en ese punto es  4.996011e-14
```

EJERCICIO 3

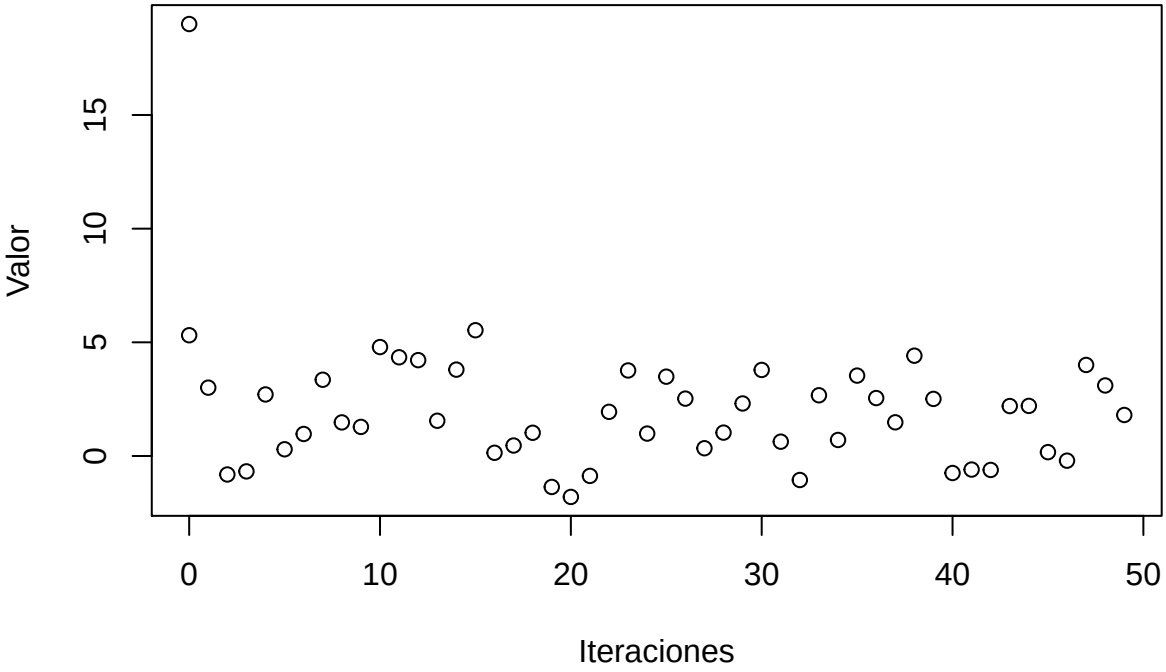
3. Considerar ahora la función $f(x,y) = (x-2)^2 + 2(y+2)^2 + 2sin(2\pi x)sin(2\pi y)$

a) Usar gradiente descendente para minimizar esta función. Usar como punto inicial $(x_0 = 1, y_0 = 1)$, tasa de aprendizaje $\eta = 0.01$ y un máximo de 50 iteraciones. Generar un gráfico de cómo desciende el valor de la función con las iteraciones. Repetir el experimento pero usando $\eta = 0,1$, comentar las diferencias y su dependencia de η.

1º Experimento: Aplicamos el gradiente descendente con $\eta = 0.01$



2º Experimento: Aplicamos el gradiente descendente con $\eta = 0.1$



La única diferencia entre ambos experimentos ha sido el valor η, sin embargo, con esta alteración podemos observar que el primero a conseguido alcanzar el valor mínimo posible, por el contrario, el segundo ha estado lejos de conseguirlo.

El valor $f(u, v)$ en el segundo experimento podemos ver que ha estado aproximadamente en el intervalo $(-1, 7)$.

Recordemos que η es un factor multiplicativo que condiciona el incremento (o decremento) de la variable independiente.

El segundo experimento tenía una tasa de aprendizaje (η) con valor $0,1$. Con lo dicho anteriormente, podemos deducir que el valor $f(u, v)$ ha estado oscilando entorno al mínimo debido a su alta tasa de aprendizaje. Es decir, cuando estaba cerca del mínimo (tanto en sentido positivo como negativo) debido a su tasa de aprendizaje incrementabamos (o decrementabamos) demasiado el valor de la variable independiente, por lo que en vez de llegar al valor del mínimo, pasabamos al otro lado del mismo. Este proceso se ha repetido numerosas veces y por eso su grafica muestra tantos picos.

En el caso del 1^{er} experimento, la tasa de aprendizaje era adecuada y por ello en ningún momento hemos oscilado entorno al mínimo.

b) Obtener el valor mínimo y los valores de las variables (x, y) en donde se alcanzan cuando el punto de inicio se fija: $(2, 1, -2, 1)$, $(3, -3)$, $(1, 5, 1, 5)$, $(1, -1)$. Generar una tabla con los valores obtenidos

```
##                     X         Y          Z
## P(2.1,-2.1) 2.242323 -2.236772 -1.8199354
## P(3,-3)     2.732558 -2.714534 -0.3810873
## P(1.5,1.5)  1.753628  1.058348 18.0509626
## P(1,-1)     1.267442 -1.285466 -0.3810873
```

EJERCICIO 4

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

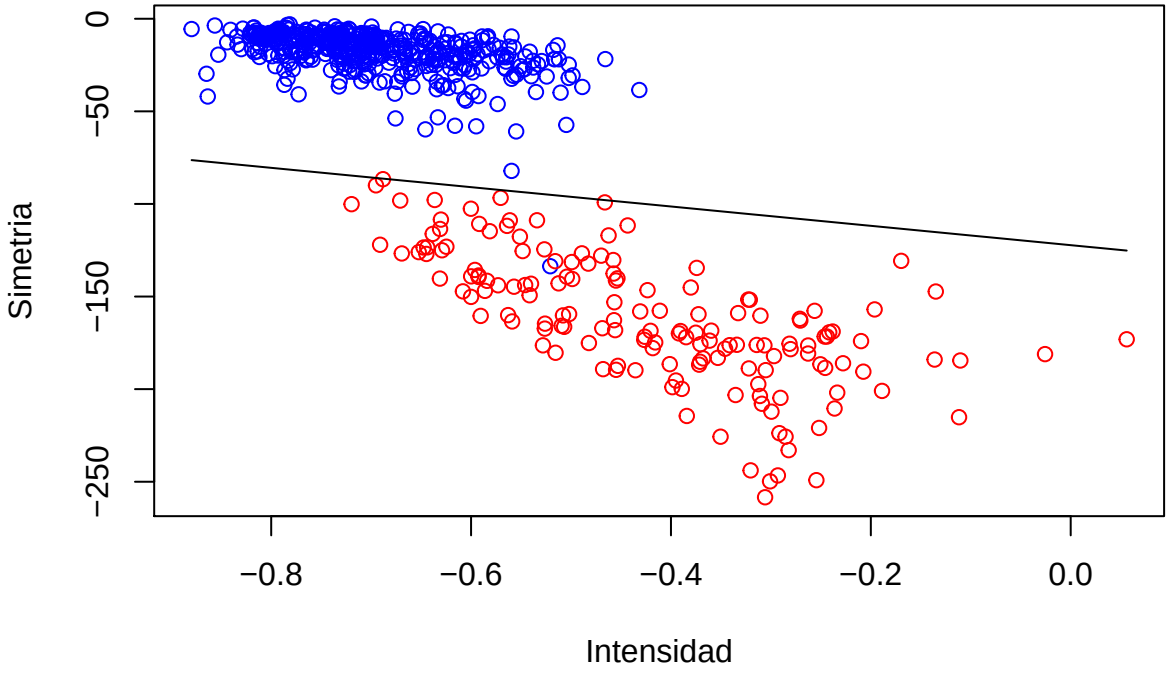
a) Encontrar un punto de inicio que te permita llegar al óptimo global sin quedar atascado en óptimos locales.
b) Encontrar una tasa de aprendizaje que sea lo suficientemente buena como para encontrar el óptimo rapidamente pero sin llegar a oscilar entorno a él.
c) Encontrar un criterio de parada fiable

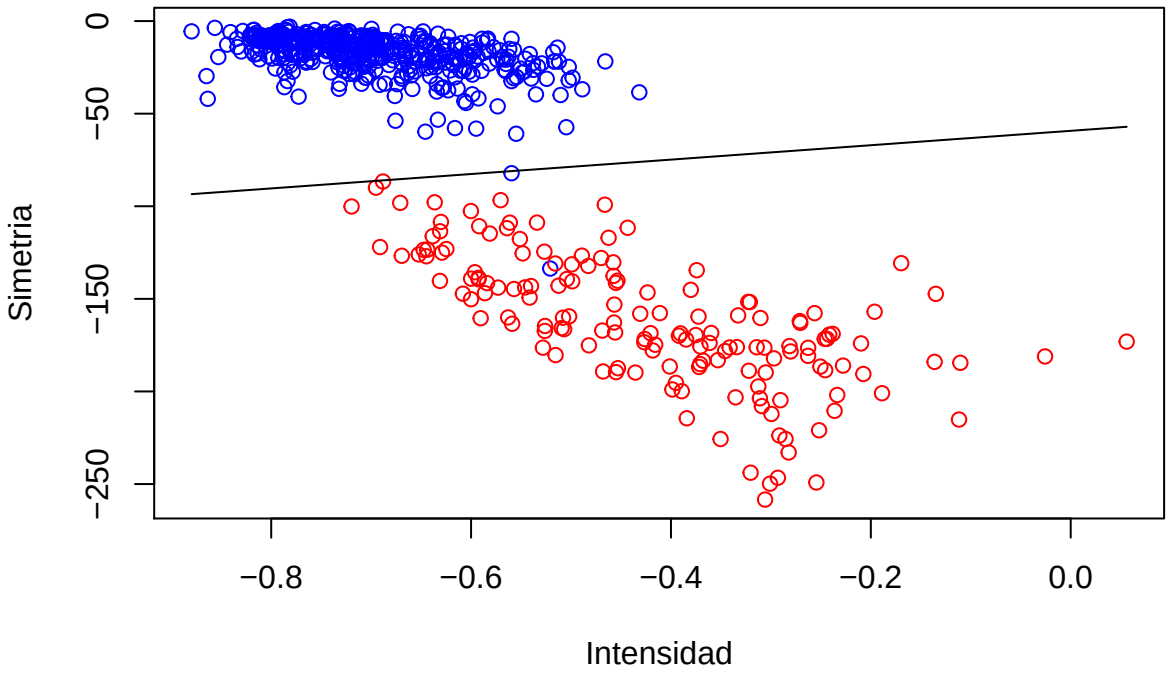
APARTADO 2

Este ejercicio ajusta modelos de regresión a vectores de características extraidos de imágenes de digitos manuscritos. En particular se extraen dos característcas concretas: el valor medio del nivel de gris y simetría del número respecto de su eje vertical. Solo se seleccionarán para este ejercicio las imágenes de los números 1 y 5.

EJERCICIO 1

1. Estimar un modelo de regresión lineal a partir de los datos proporcionados de dichos números ($Intensidad\_promedio, Simetria$) usando tanto el algoritmo de la pseudo-inversa como Gradiente descendente estocástico (SGD). 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 E_{in} y E_{out} (para E_{out} calcular las predicciones usando los datos del fichero de test). (usar $Regress\_Lin(datos, label)$ como llamada para la función (opcional)).





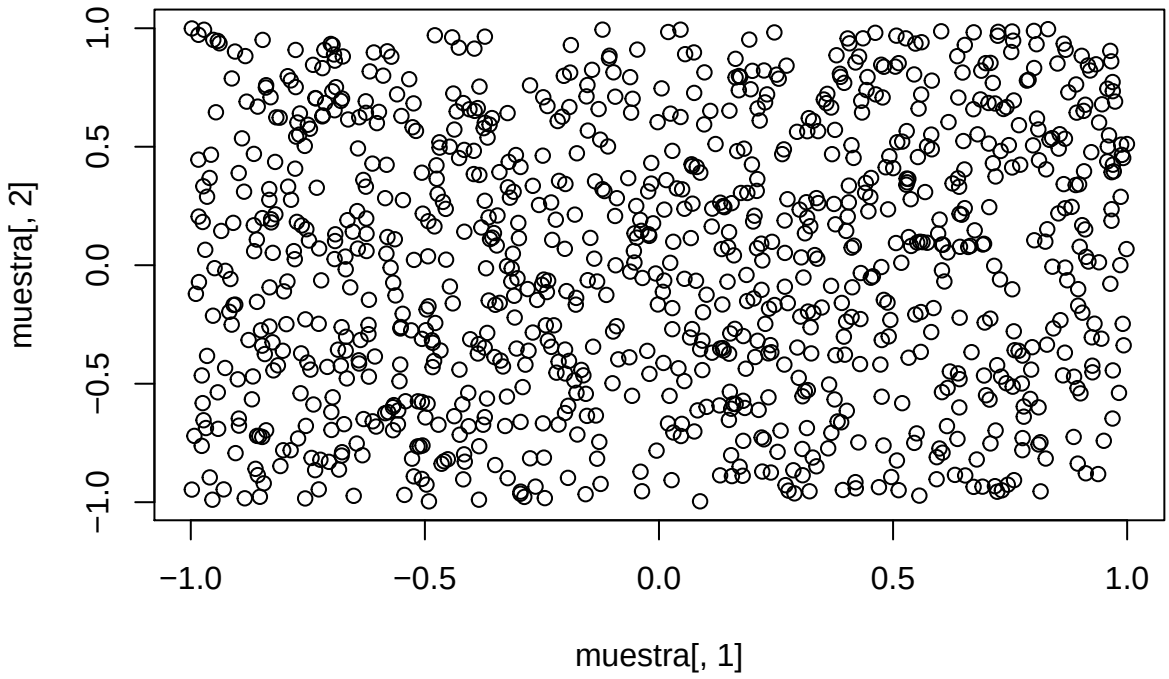

```
## Ein vale  15.44291
## Eout vale  3.180284
```

*** EJERCICIO 2***

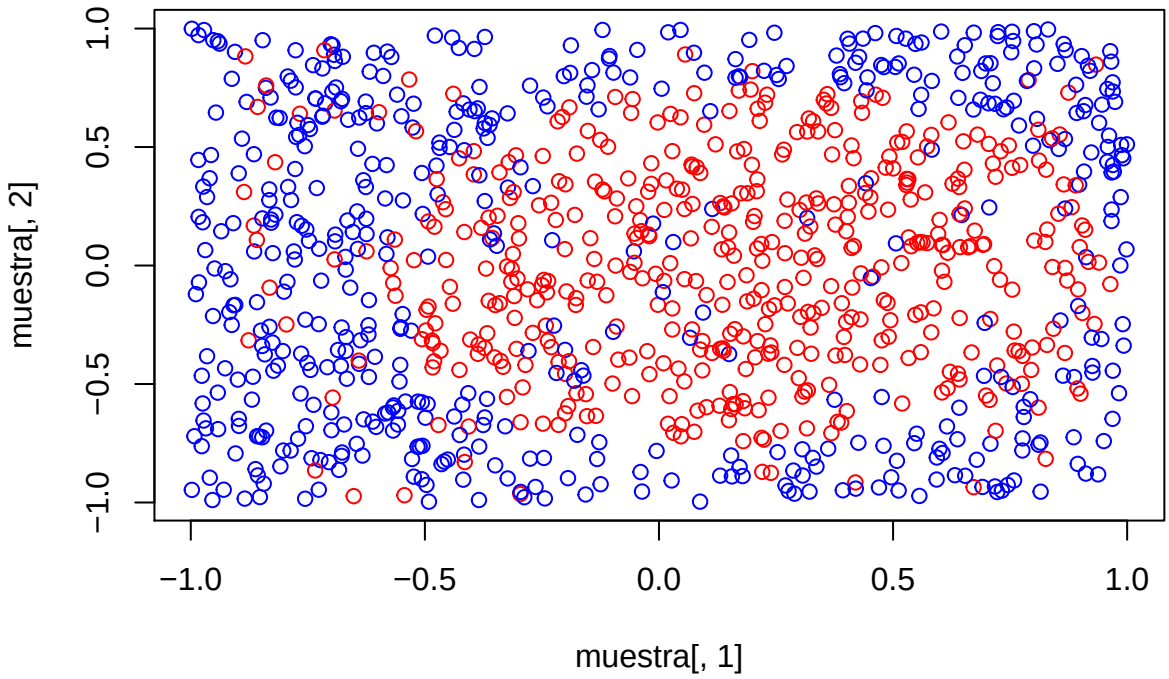
2. En este apartado exploramos como se transforman los errores E_{in} y E_{out} cuando aumentamos la complejidad del modelo lineal usado. Ahora hacemos uso de la función $simula\_unif(N, 2, size)$ que nos devuelve N coordenadas $2D$ de puntos uniformemente muestreados dentro del cuadrado definido por $[-size, size]$x$ [-size, size]$

EXPERIMENTO

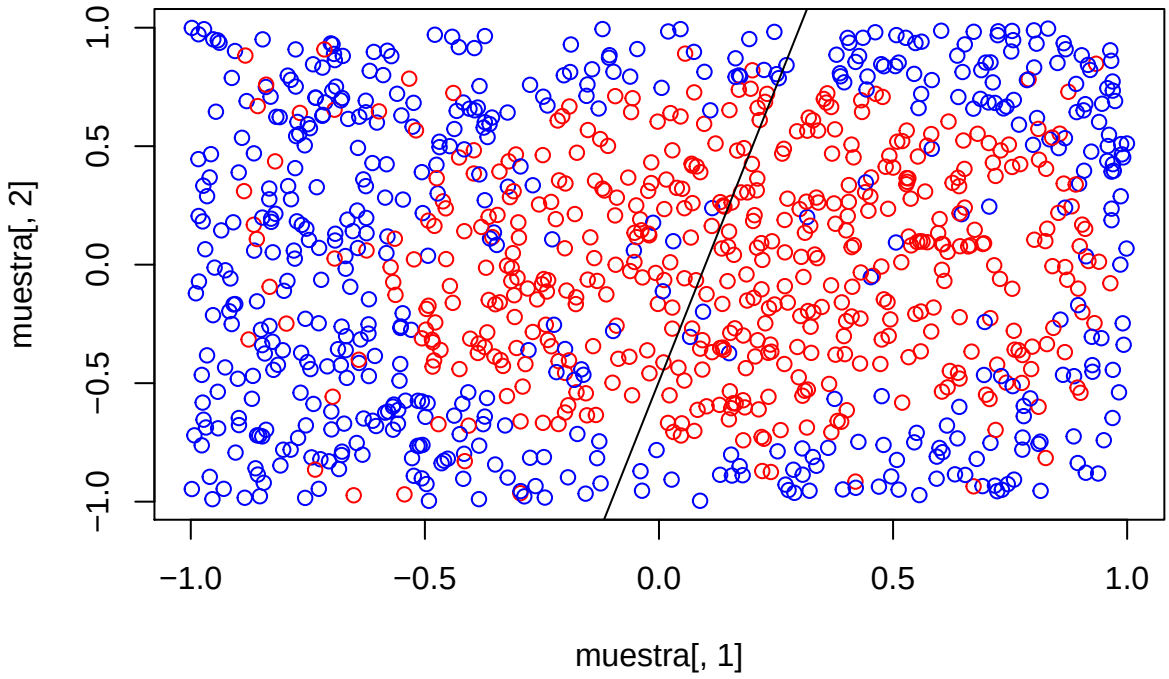
a) Generar una muestra de entrenamiento de $N = 1000$ puntos en el cuadrado $\chi = [-1, 1]$x$[-1, 1]$. Pintar el mapa de puntos $2D$. (ver función de ayuda)



b) Consideremos la función $f(x_1, x_2) = sign((x_1 - 0,2)^2 + x_2^2 - 0,6)$ que usaremos para asignar una etiqueta a cada punto de la muestra anterior. Introducimos ruido sobre las etiquetas cambiando aleatoriamente el signo de un 10% de las mismas. Pintar el mapa de etiquetas obtenido.



c) Usando como vector de características $(1, x_1, x_2)$ ajustar un modelo de regresion lineal al conjunto de datos generado y estimar los pesos ω. Estimar el error de ajuste E_{in} usando Gradiente Descendente Estocástico (SGD).




```
## Ein vale  59.84376
```

d) Ejecutar todo el experimento definido por (a)-(c) 1000 veces (generamos 1000 muestras diferentes) y

1) Calcular el valor medio de los errores E_{in} de las 1000 muestras.

2) Generar 1000 puntos nuevos por cada iteración y calcular con ellos el valor de E_{out} en dicha iteración. Calcular el valor medio de E_{out} en todas las iteraciones.

```
## Ein medio vale  841.2201
```

```
## Eout medio vale  999.3557
```

e) Valore que tan bueno considera que es el ajuste con este modelo lineal a la vista de los valores medios obtenidos de E_{in} y E_{out}

El error mostrado es grande, ya que partimos de que la hipótesis de que los datos son separables, es falsa. Para que el ajuste sea óptimo, y por tanto los errores se minimicen, es necesario realizar transformaciones no lineales sobre la función hipótesis, para lograr que la hipótesis, antes falsa, ahora pueda cumplirse, para que la división pueda realizarse con más precisión. Al incluir el ruido del 10% de los datos, hacemos que incluso teniendo un conjunto separable de datos, la separación de ambos no sea perfecta. Tras calcular el error, podemos afirmar que el porcentaje de aciertos es escaso. No se ha podido aprender nada del conjunto de datos.