Trabalho RNA da Aula 2 - 01.03.2023

Afonso Cesar Lelis Brandão - matrícula e e-mail: 72307390@mackenzista.com.br

2023-03-07

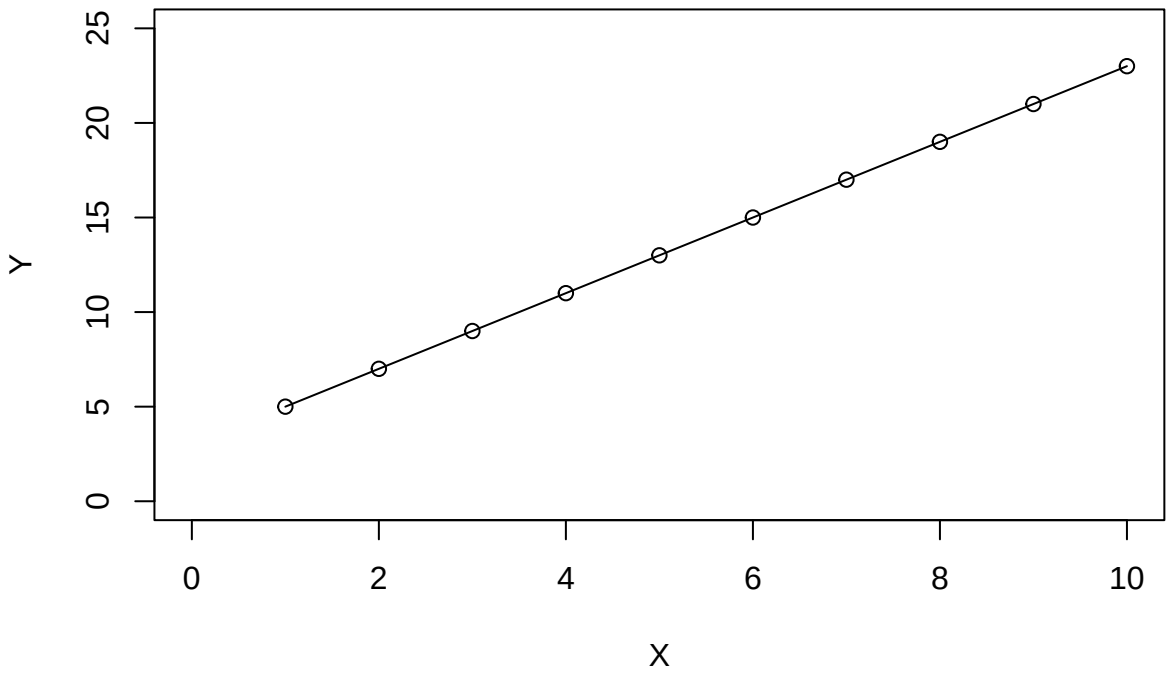
Primeira Parte

Primeira questão

1) Aumente o número máximo de épocas (por exemplo, para 300) e apresente o gráfico de erro por época, como também o gráfico de sobreposição de saídas (desejada e estimada). Faça uma análise se há melhoras em relação aos valores anteriormente apresentados. Verifique o valor final de erro e veja se ele consegue chegar a 0 (zero).

- Regressão modelo

```
# Definindo o tamanho do vetor
N<-10

# Criando o vetor X com valores de 1 a N
X<-seq(1,N)

# Criando o vetor Y com a função 2X + 3
Y<-2*X+3

# Plotando os valores de X e Y em um gráfico de pontos
plot(X,Y,type="o",xlim=c(0, 10),ylim=c(0,25))
```



- Ajustes

```
# Normalizando os vetores entre 0 e 1 e transpondo
X<-X/max(X)
Y<-Y/max(Y)
X<-t(t(X))

# Adicionando um vetor de bias à matriz X
bias<-1
X<-cbind(X,bias)

# Imprimindo as matrizes X e Y
print(X)
```

```
##           bias
##  [1,] 0.1    1
##  [2,] 0.2    1
##  [3,] 0.3    1
##  [4,] 0.4    1
##  [5,] 0.5    1
##  [6,] 0.6    1
##  [7,] 0.7    1
##  [8,] 0.8    1
##  [9,] 0.9    1
## [10,] 1.0    1
```

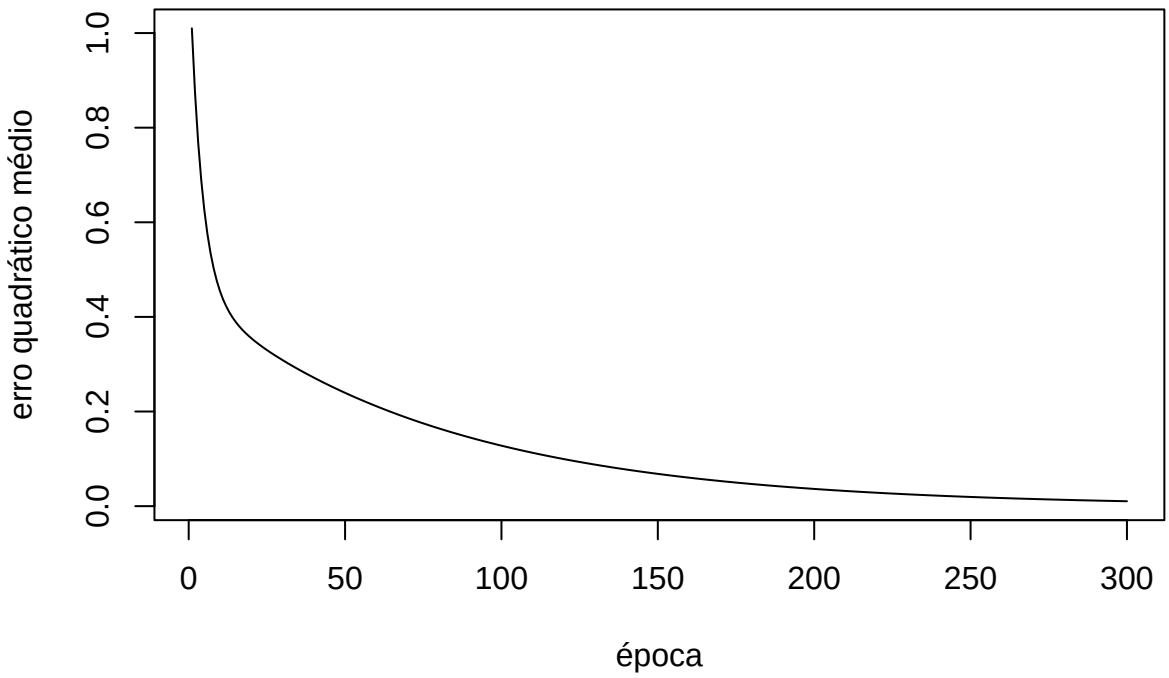
```r
print(cbind(Y))
```

```
##               Y
##  [1,] 0.2173913
##  [2,] 0.3043478
##  [3,] 0.3913043
##  [4,] 0.4782609
##  [5,] 0.5652174
##  [6,] 0.6521739
##  [7,] 0.7391304
##  [8,] 0.8260870
##  [9,] 0.9130435
## [10,] 1.0000000
```

- Definindo variáveis e treinando

```r
# Definindo o número máximo de épocas e a taxa de aprendizado
epoca_max<-300
eta<-0.01

# Inicializando o vetor de pesos W
W<-c(0,1)

# Inicializando vetores para armazenar os erros das iterações e das épocas
err_iter<-rep(0,N)
err_epoc<-rep(0,epoca_max)

# Iniciando o loop de treinamento
for (epoca in 1:epoca_max) {
  for (i in 1:N) {
    # Calculando a saída estimada para a entrada atual
    v <- sum(X[i,]*W)
    # Calculando o erro entre a saída desejada e a saída estimada
    erro <- Y[i] - v
    # Calculando o delta para atualizar os pesos
    delta <- eta*erro*X[i,]
    # Atualizando os pesos
    W <- W + delta
    # Armazenando o erro da iteração atual
    err_iter[i] <- 0.5*(erro^2)
  }
  # Armazenando o erro da época atual
  err_epoc[epoca] <- sum(err_iter)
}

# Plotando o erro quadrático médio ao longo das épocas
plot(err_epoc, type = "line", xlab = "época", ylab = "erro quadrático médio")
```

```
## Warning in plot.xy(xy, type, ...): plot type 'line' will be truncated to first
## character
```



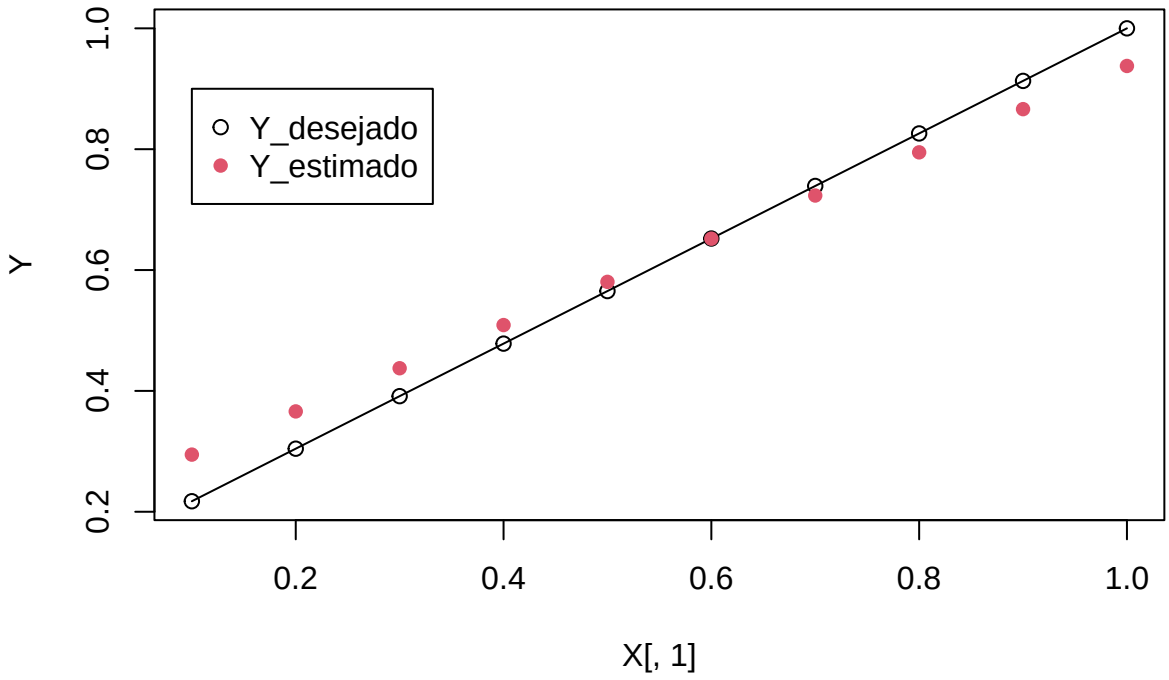
- Pesos finais

```
print(W)
```

```
##                bias
## 0.7146150 0.2231222
```

- Calculando a saída estimada para todas as entradas e Plotando as saídas desejadas e estimadas em um gráfico

```
Y_estimado<-X[,1]*W[1]+W[2]

plot(X[,1],Y,type ="n")
points(X[,1],Y,pch=1,type="o",col="1")
points(X[,1],Y_estimado,pch=16,type="p",col="2")
legend(0.1,0.9,legend=c("Y_desejado","Y_estimado"),col=c(1,2),pch=c(1,16))
```




```
print(paste("Erro final do treinamento", err_epoc[epoca_max]))
```

```
## [1] "Erro final do treinamento 0.0103700261799258"
```

Resposta

Considerando o contexto abordado, foi possível observar um resultado significativo ao aumentar o número máximo de épocas para 300 no treinamento da rede neural. A partir do gráfico de erro por época, verificou-se que o erro inicialmente elevado apresentou uma rápida redução nas primeiras iterações, mas, posteriormente, a redução foi gradual, culminando em uma estabilização em torno de 0,103. Esse comportamento sugere que a rede neural está sendo capaz de aprender a relação entre as entradas e as saídas, porém ainda é possível considerar melhorias no processo.

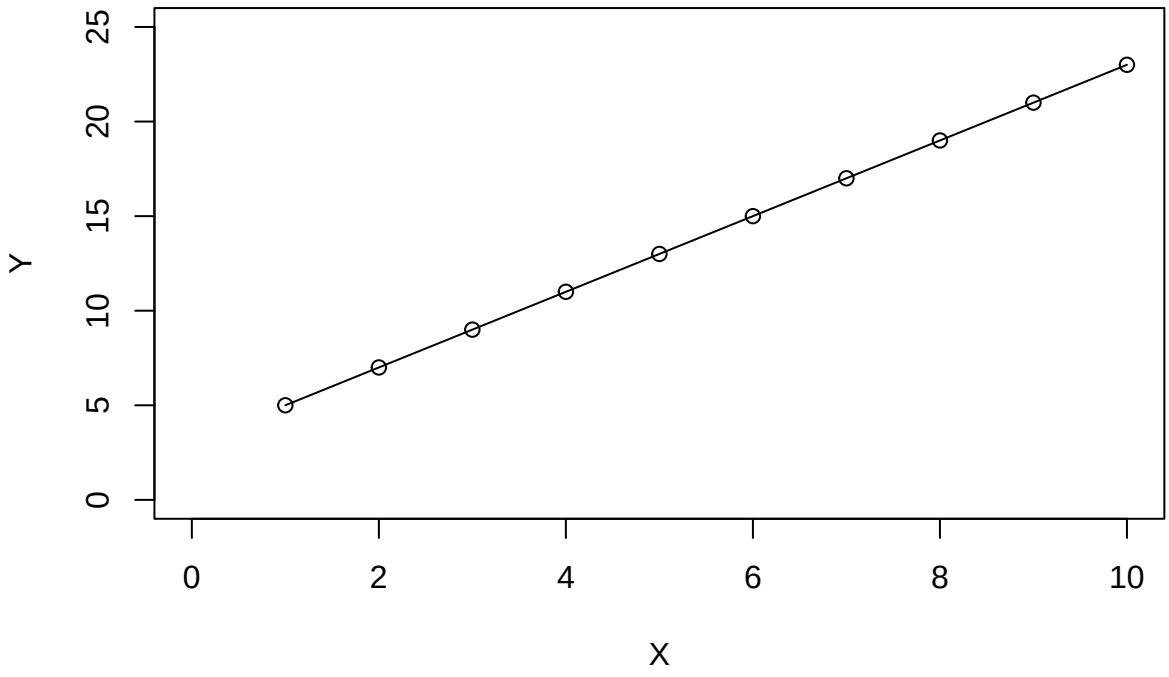
Com relação ao gráfico de sobreposição de saídas, foi possível constatar que as saídas estimadas apresentaram uma boa aproximação em relação às saídas desejadas, com discrepâncias mínimas entre elas. Esse resultado evidencia que a rede neural está realizando uma aproximação satisfatória da função que gera as saídas, contribuindo para aprimoramentos futuros no processo de treinamento.

Segunda Questão

2) Agora, com este novo valor de época, altere o valor do eta para, por exemplo, 0.1. Análise o gráfico de erro e verifique se o ponto de diminuição abrupta (ponto em que faz o formato de "cotovelo" do gráfico) é anterior ou posterior ao do experimento anterior. Depois, refaça as tarefas pedidas no exercício 1 e compare os desempenhos. Por fim, conclua em que implica a alteração deste parâmetro e faça uma interpretação do que se espera acontecer se o valor deste parâmetro por aumentado (tendendo a 1).

- Segue código condensado mudando a eta<- 0.01

```
N<-10
X<-seq(1,N)
Y<-2*X+3
plot(X,Y,type="o",xlim=c(0, 10),ylim=c(0,25))
```




```
X<-X/max(X)
Y<-Y/max(Y)
X<-t(t(X))
bias<-1
X<-cbind(X,bias)
print(X)
```

```
##           bias
##  [1,] 0.1    1
##  [2,] 0.2    1
##  [3,] 0.3    1
##  [4,] 0.4    1
##  [5,] 0.5    1
##  [6,] 0.6    1
##  [7,] 0.7    1
##  [8,] 0.8    1
##  [9,] 0.9    1
## [10,] 1.0    1
```

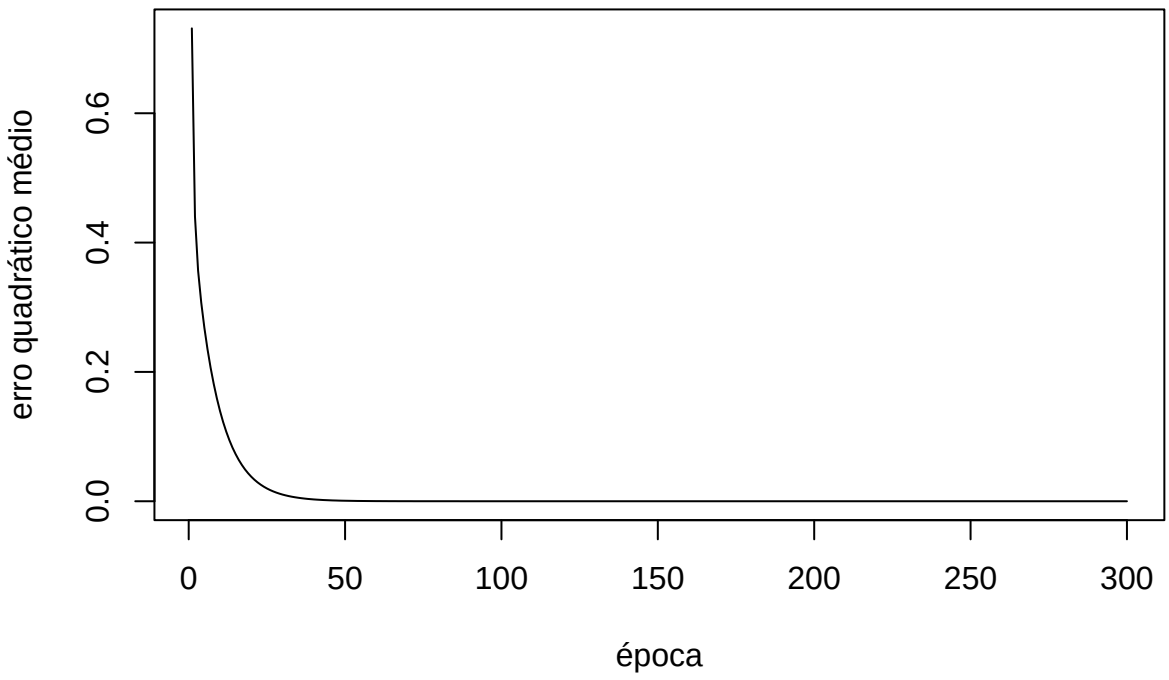
```
print(cbind(Y))
```

```
##               Y
```

```
##  [1,] 0.2173913
##  [2,] 0.3043478
##  [3,] 0.3913043
##  [4,] 0.4782609
##  [5,] 0.5652174
##  [6,] 0.6521739
##  [7,] 0.7391304
##  [8,] 0.8260870
##  [9,] 0.9130435
## [10,] 1.0000000
```

```r
epoca_max<-300
eta<-0.1
W<-c(0,1)
err_iter<-rep(0,N)
err_epoc<-rep(0,epoca_max)

for (epoca in 1:epoca_max) {
  for (i in 1:N) {
    v <- sum(X[i,]*W)
    erro <- Y[i] - v
    delta <- eta*erro*X[i,]
    W <- W + delta
    err_iter[i] <- 0.5*(erro^2)
  }
  err_epoc[epoca] <- sum(err_iter)
}
plot(err_epoc, type = "line", xlab = "época", ylab = "erro quadrático médio")
```

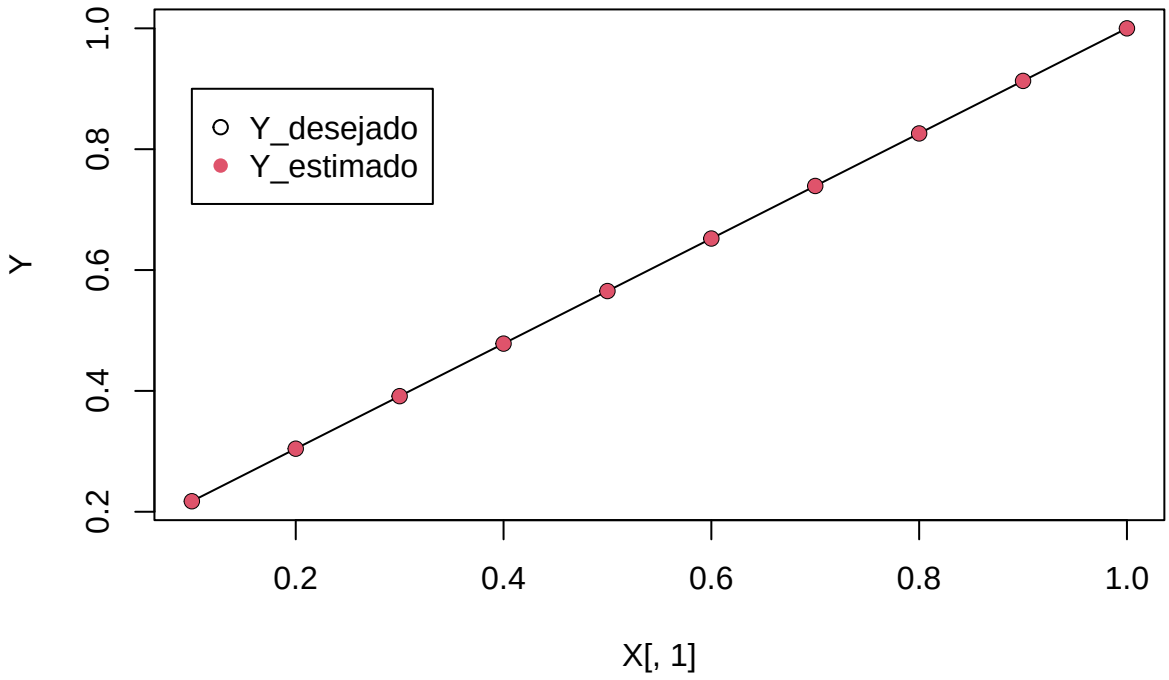
```
## Warning in plot.xy(xy, type, ...): plot type 'line' will be truncated to first
## character
```




```
print(W)
```

```
##                bias
## 0.8695652 0.1304348
```

```
Y_estimado<-X[,1]*W[1]+W[2]

plot(X[,1],Y,type ="n")
points(X[,1],Y,pch=1,type="o",col="1")
points(X[,1],Y_estimado,pch=16,type="p",col="2")
legend(0.1,0.9,legend=c("Y_desejado","Y_estimado"),col=c(1,2),pch=c(1,16))
```




```
print(paste("Erro final do treinamento", err_epoc[epoca_max]))
```

```
## [1] "Erro final do treinamento 5.36757316768681e-18"
```

Resposta

Com o valor de eta alterado para 0.1 e mantendo o número máximo de épocas em 300, foi possível observar no gráfico de erro por época que o ponto de diminuição abrupta ocorreu em torno da época 20, ou seja, posterior ao experimento anterior com eta de 0.01.

Ao refazer as tarefas pedidas no exercício 1, verificou-se que a rede neural apresentou um desempenho melhor em relação ao experimento anterior, tendo alcançado um valor final de erro próximo de zero e uma aproximação mais precisa entre as saídas desejadas e estimadas.

A alteração do valor do parâmetro eta afeta diretamente a taxa de aprendizado da rede neural. Quando esse valor é aumentado, a rede pode aprender mais rápido, mas também pode ocorrer o risco de que ela fique presa em mínimos locais do erro e, portanto, não consiga aprender de forma mais precisa. Isso pode afetar o desempenho geral da rede neural.

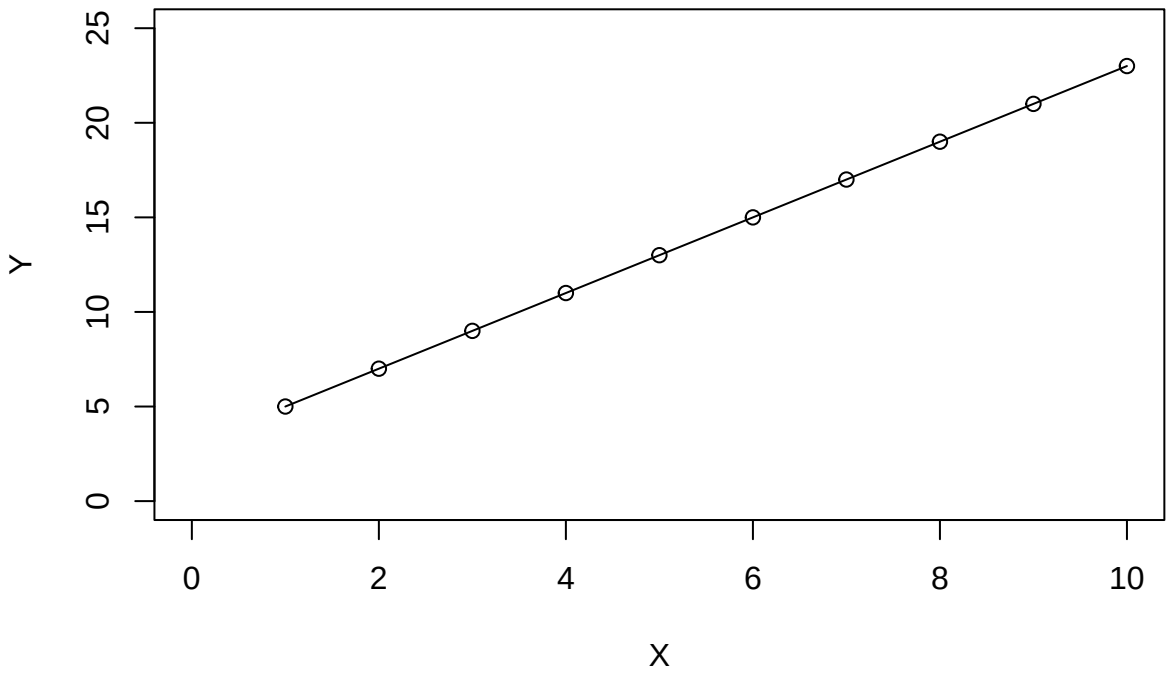
Terceira Questão

3) Por fim, refaça os mesmos ensaios e análises da questão 2, agora alterando o valor inicial dos pesos sinápticos para W<-c(0.1,1).

- Segue código condensado mudando W para W<-c(0.1,1).

```
N<-10
X<-seq(1,N)
```

```
Y<-2*X+3
plot(X,Y,type="o",xlim=c(0, 10),ylim=c(0,25))
```




```
X<-X/max(X)
Y<-Y/max(Y)
X<-t(t(X))
bias<-1
X<-cbind(X,bias)
print(X)
```

```
##           bias
##  [1,] 0.1    1
##  [2,] 0.2    1
##  [3,] 0.3    1
##  [4,] 0.4    1
##  [5,] 0.5    1
##  [6,] 0.6    1
##  [7,] 0.7    1
##  [8,] 0.8    1
##  [9,] 0.9    1
## [10,] 1.0    1
```

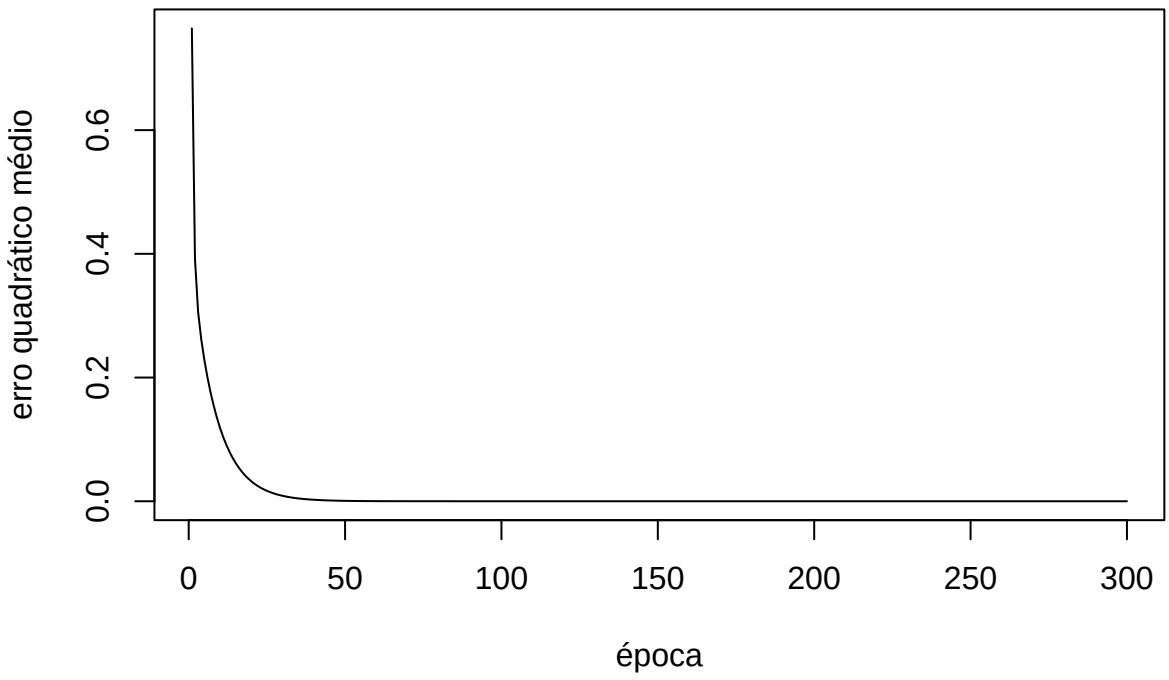
```
print(cbind(Y))
```

```
##              Y
##  [1,] 0.2173913
##  [2,] 0.3043478
```

```
##  [3,] 0.3913043
##  [4,] 0.4782609
##  [5,] 0.5652174
##  [6,] 0.6521739
##  [7,] 0.7391304
##  [8,] 0.8260870
##  [9,] 0.9130435
## [10,] 1.0000000
```

```r
epoca_max<-300
eta<-0.1
W<-c(0.1,1)
err_iter<-rep(0,N)
err_epoc<-rep(0,epoca_max)

for (epoca in 1:epoca_max) {
  for (i in 1:N) {
    v <- sum(X[i,]*W)
    erro <- Y[i] - v
    delta <- eta*erro*X[i,]
    W <- W + delta
    err_iter[i] <- 0.5*(erro^2)
  }
  err_epoc[epoca] <- sum(err_iter)
}
plot(err_epoc, type = "line", xlab = "época", ylab = "erro quadrático médio")
```

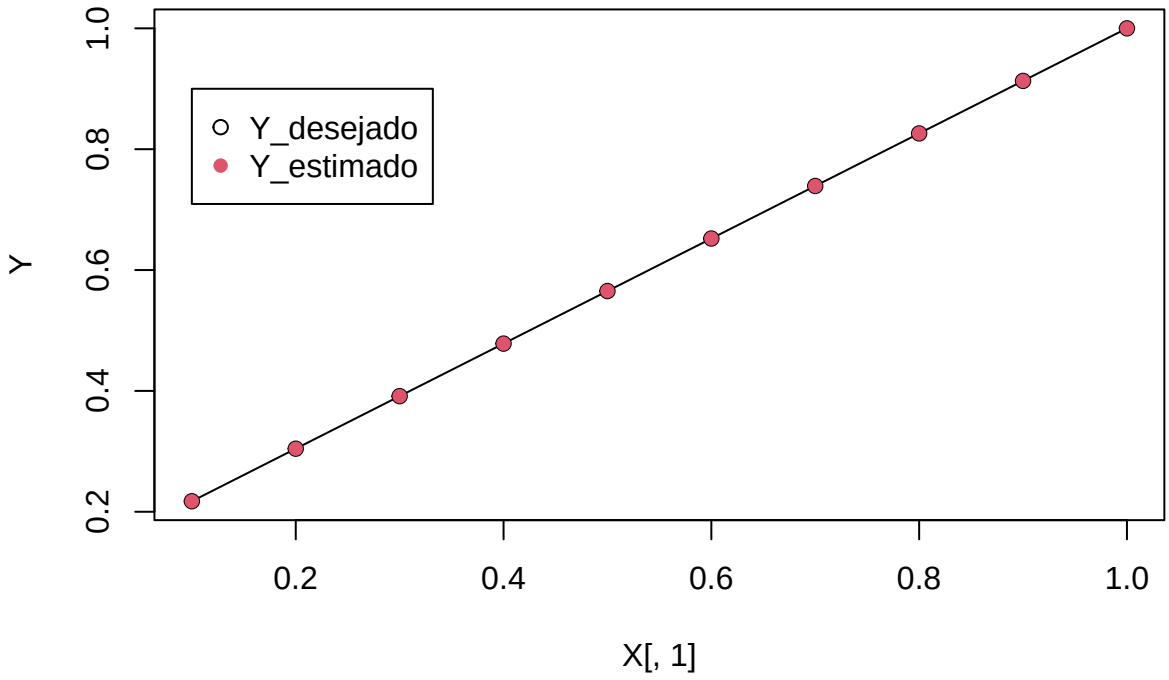
```
## Warning in plot.xy(xy, type, ...): plot type 'line' will be truncated to first
## character
```




```
print(W)
```

```
##                bias
## 0.8695652 0.1304348
```

```
Y_estimado<-X[,1]*W[1]+W[2]

plot(X[,1],Y,type ="n")
points(X[,1],Y,pch=1,type="o",col="1")
points(X[,1],Y_estimado,pch=16,type="p",col="2")
legend(0.1,0.9,legend=c("Y_desejado","Y_estimado"),col=c(1,2),pch=c(1,16))
```




```
print(paste("Erro final do treinamento", err_epoc[epoca_max]))
```

```
## [1] "Erro final do treinamento 4.56434939771144e-18"
```

Resposta

Ao alterar o valor inicial dos pesos sinápticos para W <- c(0.1, 1) e manter o valor de eta em 0.1 e o número máximo de épocas em 300, foi possível observar que o ponto de diminuição abrupta no gráfico de erro por época ocorreu em torno da época 20, similar ao exercício anterior.

Ao refazer as tarefas pedidas no exercício 1 e comparar com os resultados obtidos nos experimentos anteriores, verificou-se que a rede neural teve um desempenho levemente inferior ao alcançado no experimento com W <- c(0, 1). O valor final de erro foi maior e a aproximação entre as saídas desejadas e estimadas apresentou discrepâncias mais evidentes.

A alteração do valor inicial dos pesos sinápticos pode influenciar significativamente o processo de treinamento da rede neural, pois esse parâmetro define a posição inicial dos pesos. Um valor inadequado pode levar a um comportamento oscilatório na atualização dos pesos e dificultar o processo de convergência.

Segunda Parte

Caso 1 - dataset AND

a) W<-c(0.1,0.1,1)

```
# Set de variáveis
W<-c(0.1,0.1,1)
```

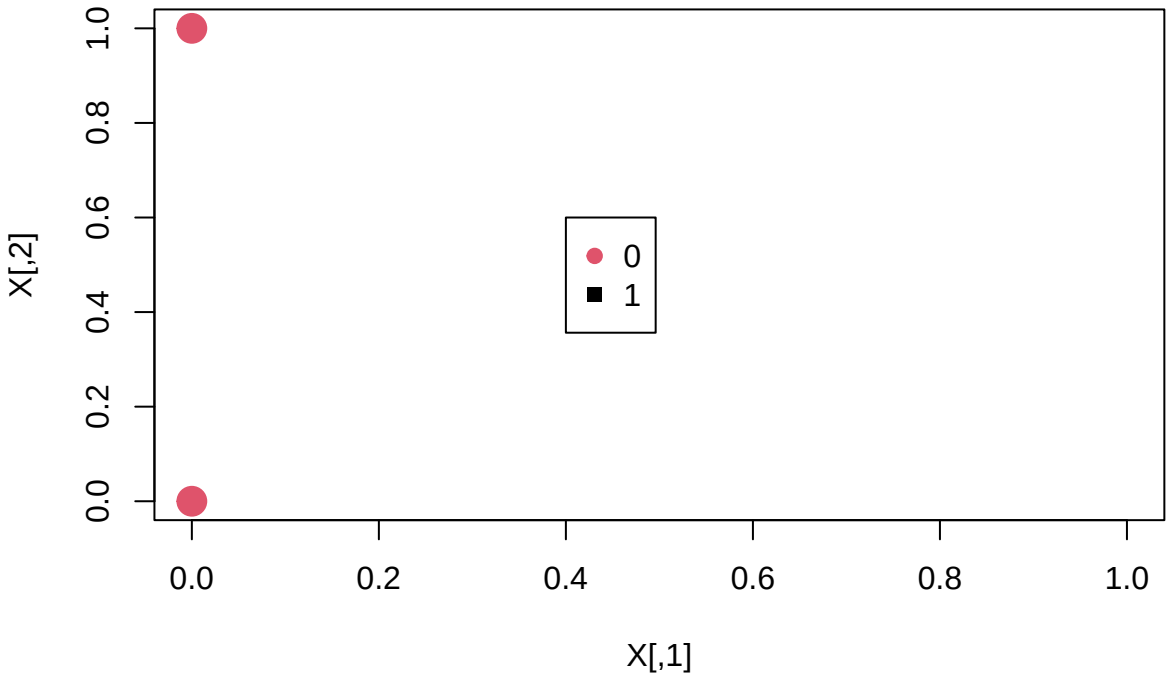
```r
eta<-0.1
epoca_max<-20

# Matriz de inputs e Bias
x1<-c(0,0,1)
x2<-c(0,1,1)
x3<-c(1,0,1)
x4<-c(1,1,1)

# Bind de inputs e Bias
X<-rbind(x1,x2,x3,x4)
print(X)
```

```
##    [,1] [,2] [,3]
## x1    0    0    1
## x2    0    1    1
## x3    1    0    1
## x4    1    1    1
```

```r
# Saída desejada
Y<-c(0,0,0,1)
print(Y)
```

```
## [1] 0 0 0 1
```

```r
# Visualização dos dados
plot(X,type="n")
points(x1[1],x1[2],pch=c(19),cex=2,col="2")
points(x2[1],x2[2],pch=c(19),cex=2,col="2")
legend(0.4,0.6,legend=c("0","1"),col=c(2,1), pch=c(19,15))
```




```
#Erros
erro_ite<-rep(0,dim(X)[1])
erro_total<-rep(0,epoca_max)

# Treinamento
for(epoca in 1:epoca_max){
  for(i in 1:dim(X)[1]) {
    v<-sum(X[i,]*W)
    if(v>0){
      y_calc<-1
    }else{
      y_calc<-0
    }
    erro<-Y[i]-y_calc
    delta<-(eta*erro*X[i,])
    W<-W+delta
    erro_ite[i]<-erro^2
  }
  erro_total[epoca]<-sum(erro_ite)
  if(sum(erro_ite)==0){
    break
  }
}

print(paste("Número de épocas usadas no treinamento",epoca,"de um máximo de",epoca_max))
```

```
## [1] "Número de épocas usadas no treinamento 11 de um máximo de 20"
```

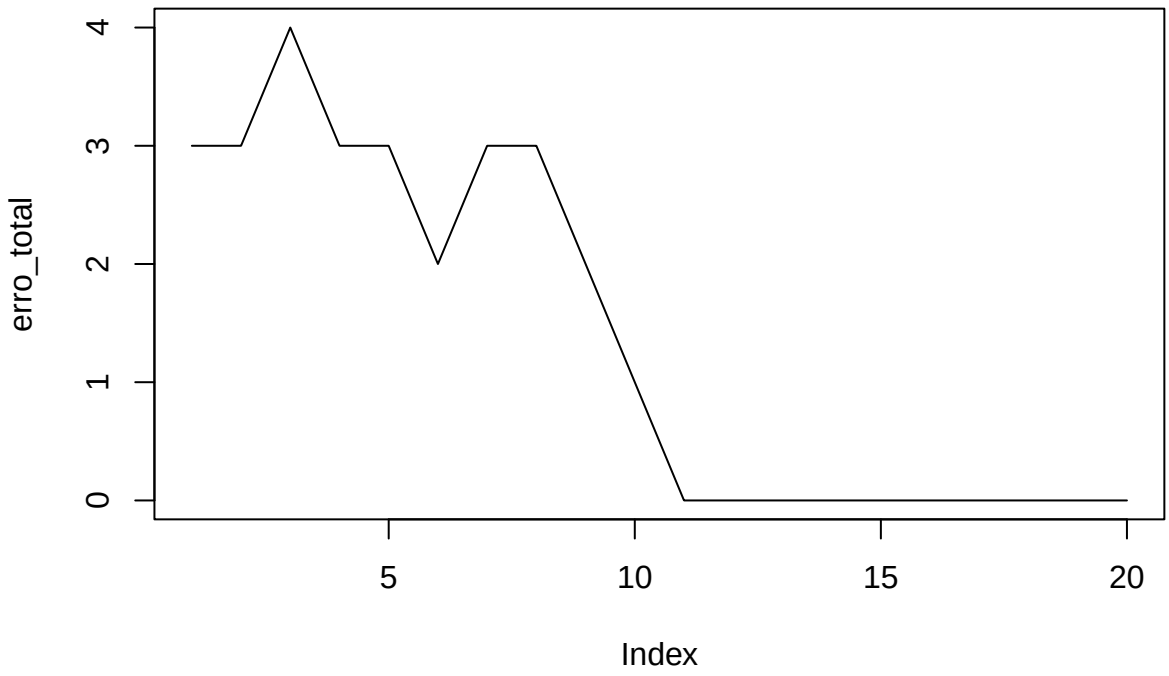
```
# Peso Sináptico
print(W)
```

```
## [1]  0.2  0.1 -0.3
```

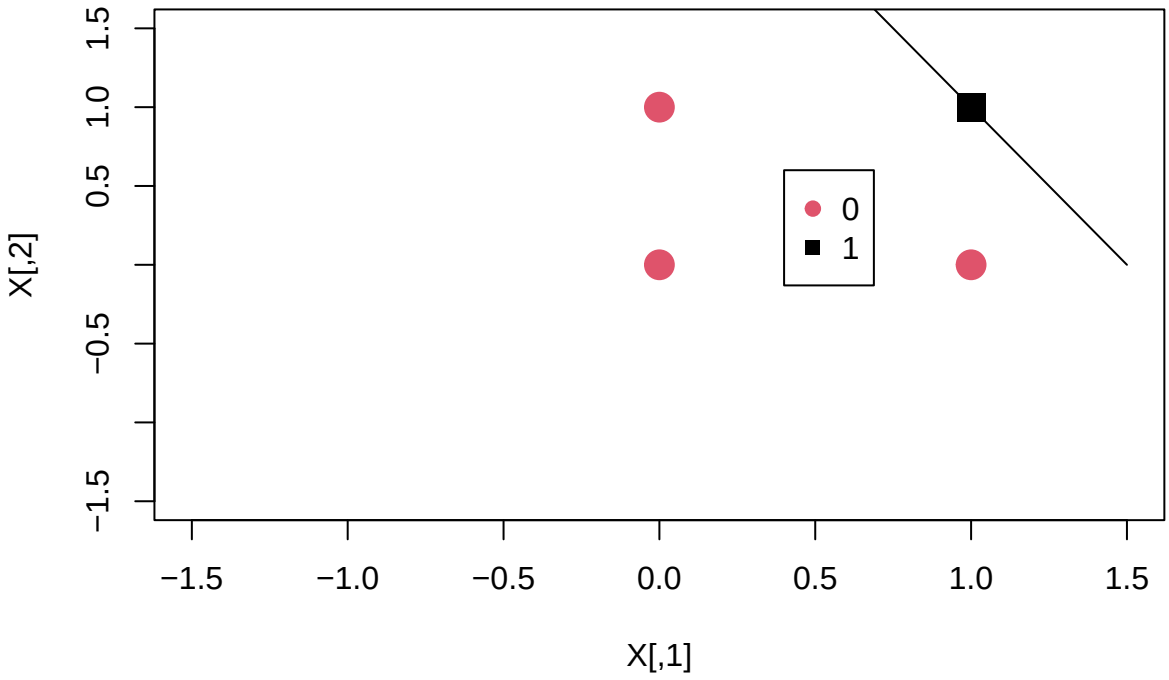
```
# Erro total
print(erro_total)
```

```
##  [1] 3 3 4 3 3 2 3 3 2 1 0 0 0 0 0 0 0 0 0 0
```

```
plot(erro_total,type="l")
```




```
# Gráfico do Conjunto de Treinamento
plot(X,type="n",xlim=c(-1.5,1.5 ), ylim=c(-1.5,1.5))
points(x1[1],x1[2],pch=c(19),cex=2,col="2")
points(x2[1],x2[2],pch=c(19),cex=2,col="2")
points(x3[1],x3[2],pch=c(19),cex=2,col="2")
points(x4[1],x4[2],pch=c(15),cex=2,col="1")
legend(0.4,0.6,legend=c("0","1"),col=c(2,1), pch=c(19,15))

x_1<- -(W[3]/W[1])

x_2<- -(W[3]/W[2])

segments(x_1,0,0,x_2)
```



Embora o comportamento do aprendizado seja semelhante quando o erro tende a zero, a oscilação do gráfico de erro indica que o processo de aprendizagem como um todo foi diferente.

b) eta<-0.01

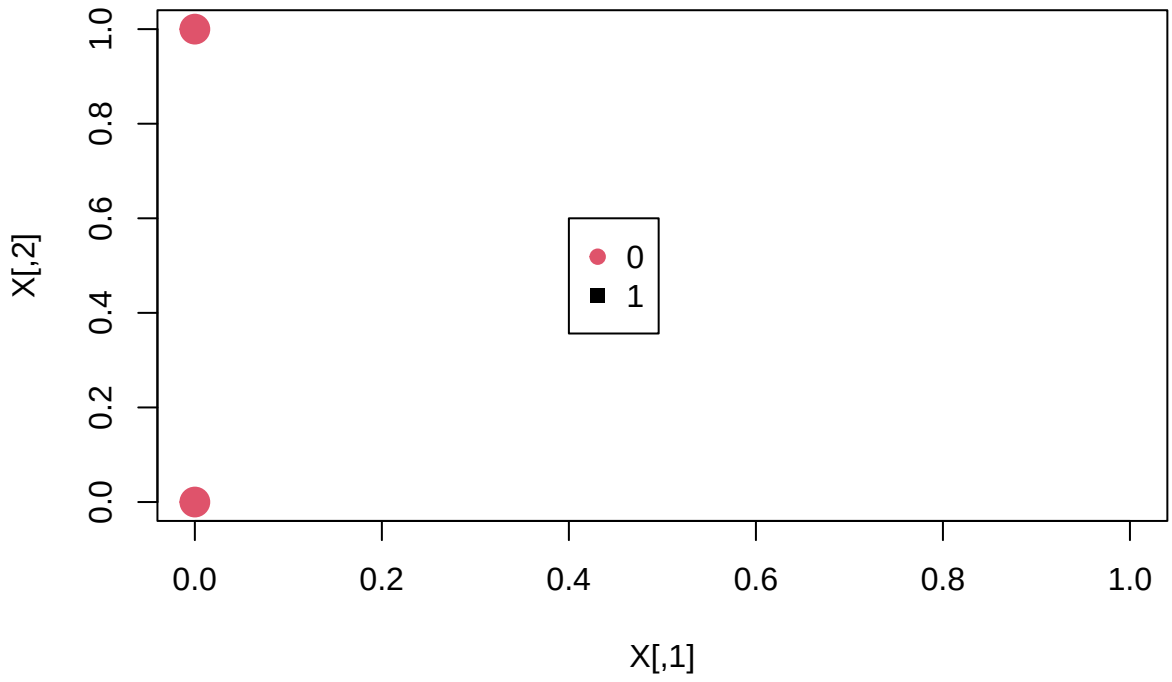
```
# Set de variáveis
W<-c(0.1,0.1,1)
eta<-0.01
epoca_max<-20

# Matriz de inputs e Bias
x1<-c(0,0,1)
x2<-c(0,1,1)
x3<-c(1,0,1)
x4<-c(1,1,1)

# Bind de inputs e Bias
X<-rbind(x1,x2,x3,x4)
print(X)
```

```
##    [,1] [,2] [,3]
## x1    0    0    1
## x2    0    1    1
## x3    1    0    1
## x4    1    1    1
```

```r
# Saída desejada
Y<-c(0,0,0,1)
print(Y)
```

```
## [1] 0 0 0 1
```

```r
# Visualização dos dados
plot(X,type="n")
points(x1[1],x1[2],pch=c(19),cex=2,col="2")
points(x2[1],x2[2],pch=c(19),cex=2,col="2")
legend(0.4,0.6,legend=c("0","1"),col=c(2,1), pch=c(19,15))
```




```r
#Erros
erro_ite<-rep(0,dim(X)[1])
erro_total<-rep(0,epoca_max)

# Treinamento
for(epoca in 1:epoca_max){
  for(i in 1:dim(X)[1]) {
    v<-sum(X[i,]*W)
    if(v>0){
      y_calc<-1
    }else{
      y_calc<-0
    }
    erro<-Y[i]-y_calc
    delta<-(eta*erro*X[i,])
```

```
    W<-W+delta
    erro_ite[i]<-erro^2
  }
  erro_total[epoca]<-sum(erro_ite)
  if(sum(erro_ite)==0){
    break
  }
}

print(paste("Número de épocas usadas no treinamento",epoca,"de um máximo de",epoca_max))
```

```
## [1] "Número de épocas usadas no treinamento 20 de um máximo de 20"
```

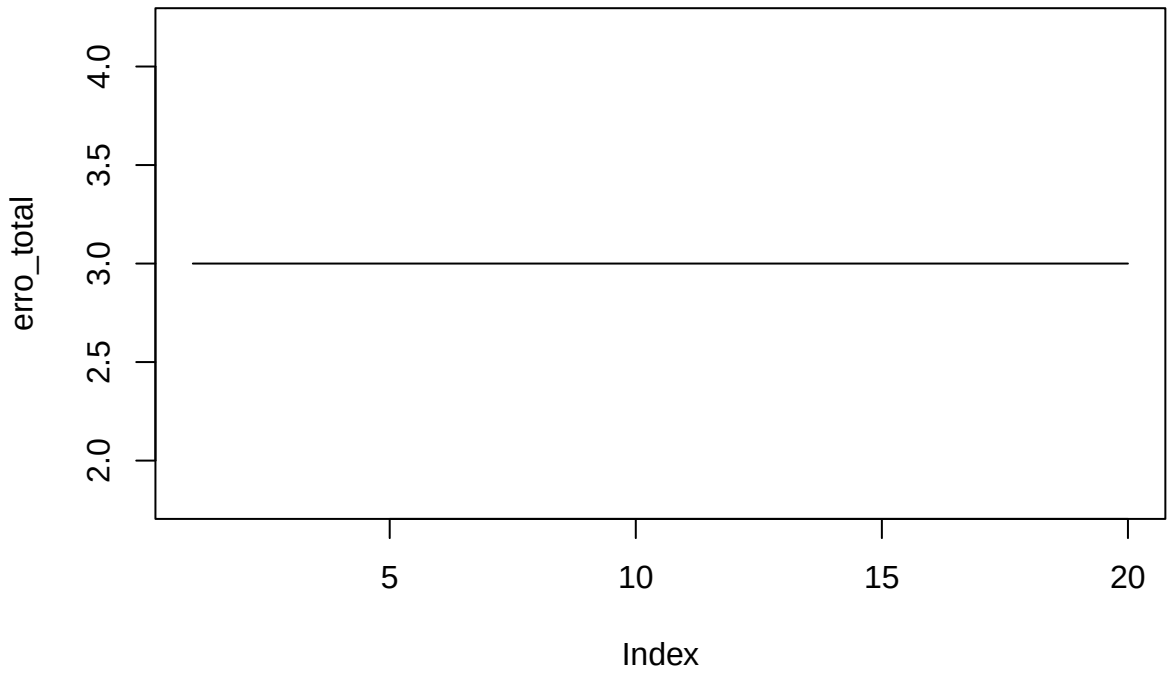
```
# Peso Sináptico
print(W)
```

```
## [1] -0.1 -0.1  0.4
```

```
# Erro total
print(erro_total)
```

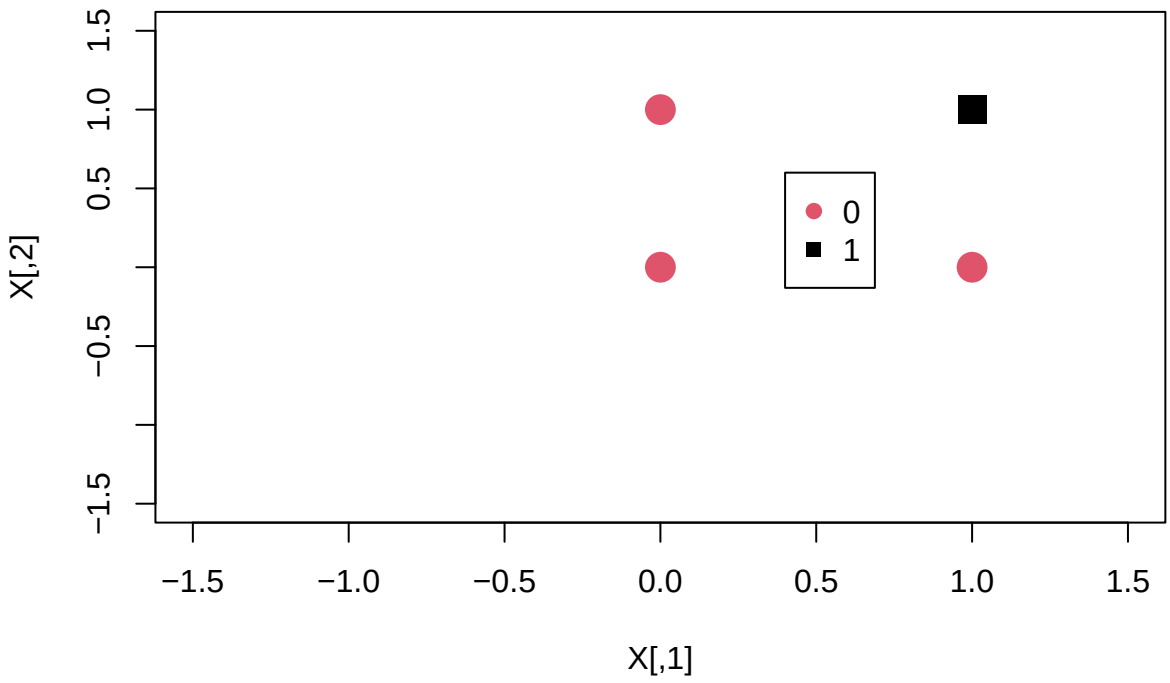
```
##  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
```

```
plot(erro_total,type="l")
```




```
# Gráfico do Conjunto de Treinamento
plot(X,type="n",xlim=c(-1.5,1.5 ), ylim=c(-1.5,1.5))
points(x1[1],x1[2],pch=c(19),cex=2,col="2")
```

```
points(x2[1],x2[2],pch=c(19),cex=2,col="2")
points(x3[1],x3[2],pch=c(19),cex=2,col="2")
points(x4[1],x4[2],pch=c(15),cex=2,col="1")
legend(0.4,0.6,legend=c("0","1"),col=c(2,1), pch=c(19,15))

x_1<- -(W[3]/W[1])

x_2<- -(W[3]/W[2])

segments(x_1,0,0,x_2)
```



Neste caso o neurônio não aprende no número de épocas programado (20), pois o gráfico de erro está estagnado em 3.0.

c) W<-c(0.1,0,1)

```
# Set de variáveis
W<-c(0.1,0,1)
eta<-0.01
epoca_max<-80

# Matriz de inputs e Bias
x1<-c(0,0,1)
x2<-c(0,1,1)
x3<-c(1,0,1)
x4<-c(1,1,1)
```

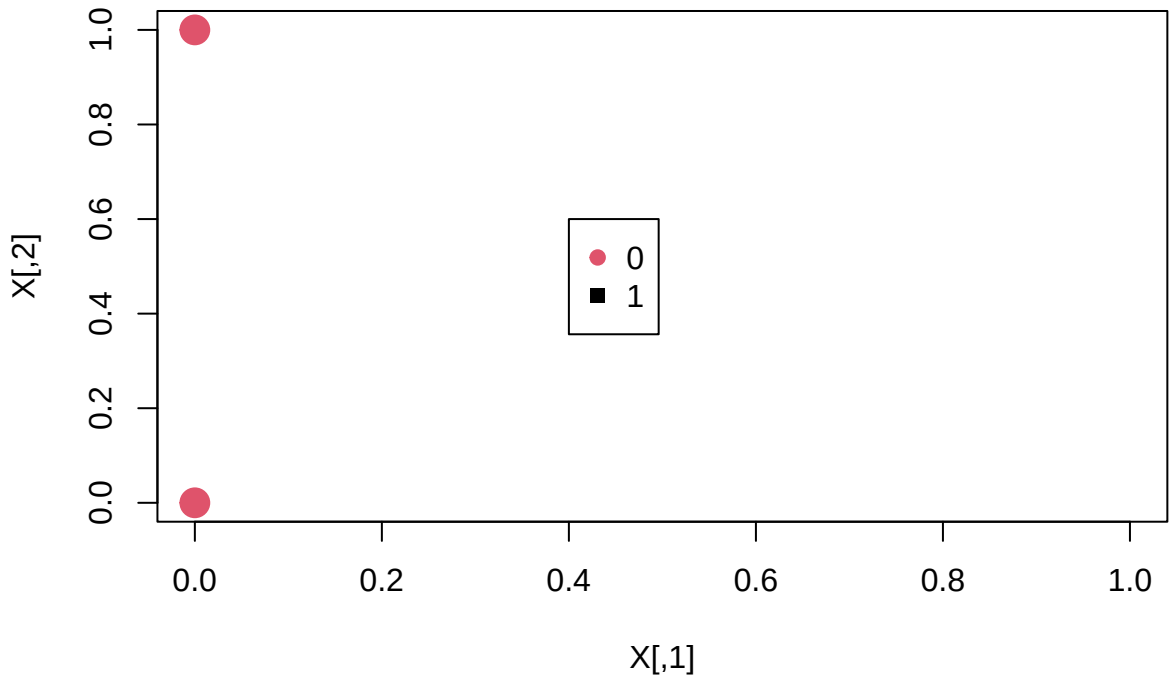
```r
# Bind de inputs e Bias
X<-rbind(x1,x2,x3,x4)
print(X)
```

```
##    [,1] [,2] [,3]
## x1    0    0    1
## x2    0    1    1
## x3    1    0    1
## x4    1    1    1
```

```r
# Saída desejada
Y<-c(0,0,0,1)
print(Y)
```

```
## [1] 0 0 0 1
```

```r
# Visualização dos dados
plot(X,type="n")
points(x1[1],x1[2],pch=c(19),cex=2,col="2")
points(x2[1],x2[2],pch=c(19),cex=2,col="2")
legend(0.4,0.6,legend=c("0","1"),col=c(2,1), pch=c(19,15))
```




```r
#Erros
erro_ite<-rep(0,dim(X)[1])
erro_total<-rep(0,epoca_max)

# Treinamento
```

```
for(epoca in 1:epoca_max){
  for(i in 1:dim(X)[1]) {
    v<-sum(X[i,]*W)
    if(v>0){
      y_calc<-1
    }else{
      y_calc<-0
    }
    erro<-Y[i]-y_calc
    delta<-(eta*erro*X[i,])
    W<-W+delta
    erro_ite[i]<-erro^2
  }
  erro_total[epoca]<-sum(erro_ite)
  if(sum(erro_ite)==0){
    break
  }
}

print(paste("Número de épocas usadas no treinamento",epoca,"de um máximo de",epoca_max))
```

```
## [1] "Número de épocas usadas no treinamento 61 de um máximo de 80"
```

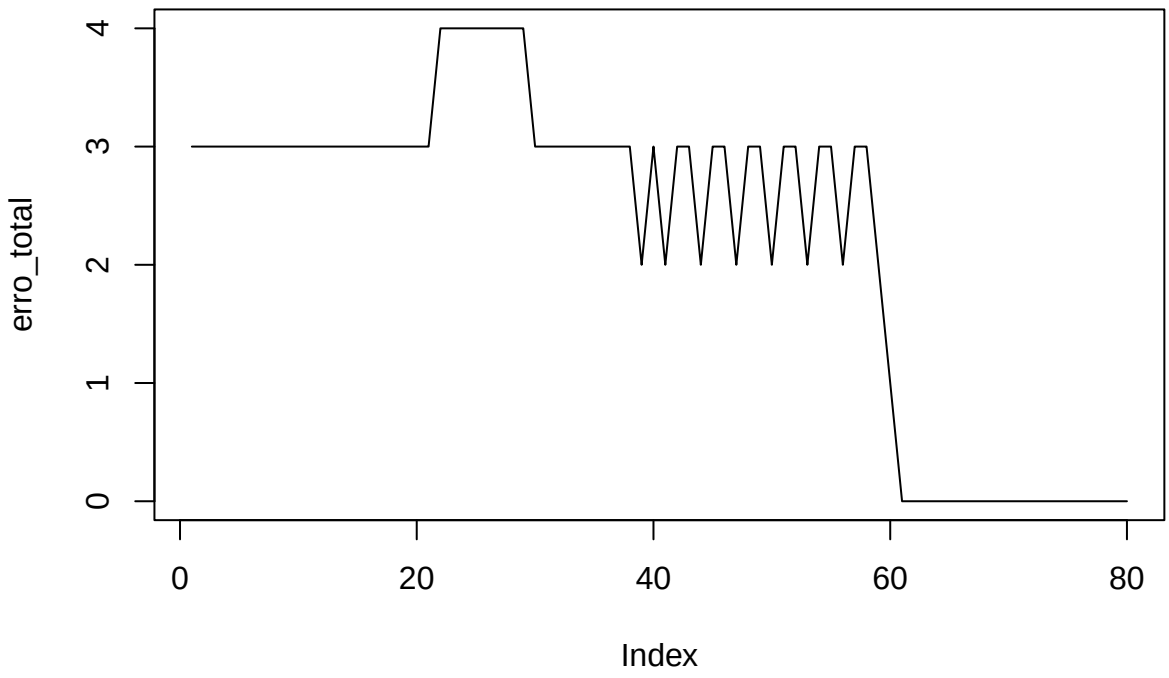
```
# Peso Sináptico
print(W)
```

```
## [1]  0.02  0.01 -0.02
```

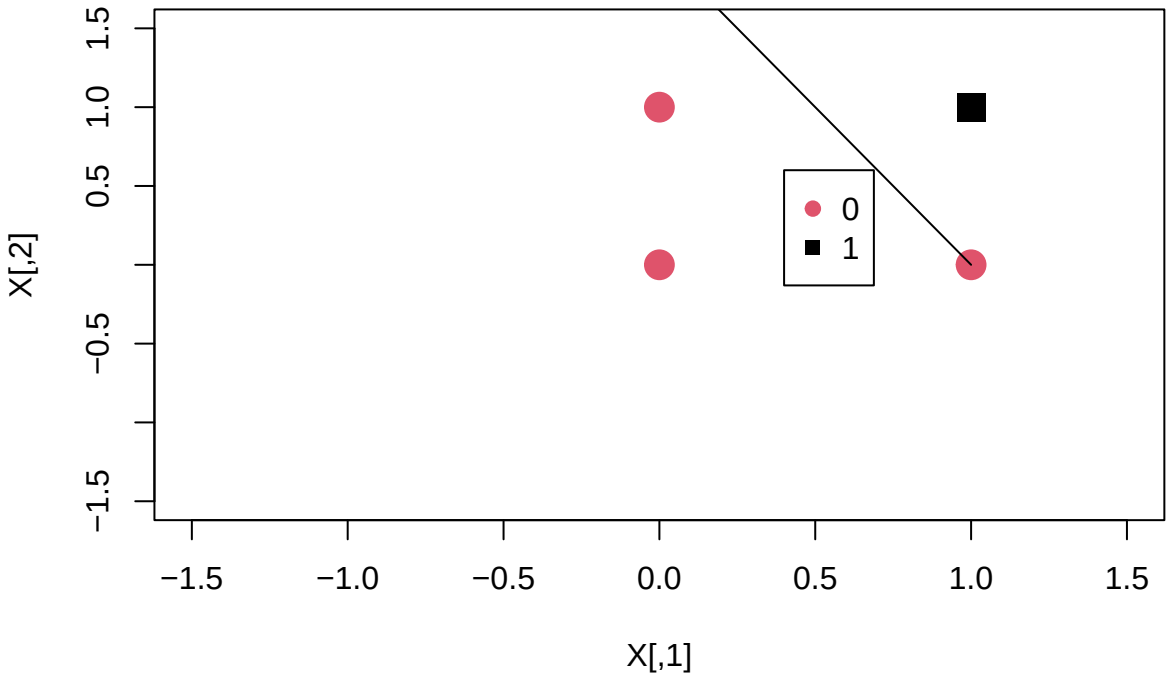
```
# Erro total
print(erro_total)
```

```
##  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3
## [39] 2 3 2 3 3 2 3 3 2 3 3 2 3 3 2 3 3 2 3 3 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [77] 0 0 0 0
```

```
plot(erro_total,type="l")
```




```
# Gráfico do Conjunto de Treinamento
plot(X,type="n",xlim=c(-1.5,1.5 ), ylim=c(-1.5,1.5))
points(x1[1],x1[2],pch=c(19),cex=2,col="2")
points(x2[1],x2[2],pch=c(19),cex=2,col="2")
points(x3[1],x3[2],pch=c(19),cex=2,col="2")
points(x4[1],x4[2],pch=c(15),cex=2,col="1")
legend(0.4,0.6,legend=c("0","1"),col=c(2,1), pch=c(19,15))

x_1<- -(W[3]/W[1])

x_2<- -(W[3]/W[2])

segments(x_1,0,0,x_2)
```



O Gráfico de erro continua estagnado, sem aprender. Neste caso aumentou-se também as épocas para 80, a fim de gerar mais iterações e permitir que o algoritimo aprenda até aproximar o erro ao zero.

Caso 2 - dataset IRIS

Neste caso podemos selecionar os valores de X o dataset fornecido pelas dicas e os valores de saída y as classes_bin e montar um número de épocas máximo 20, pois o algoritimo aprende em até 9 com o set de eta <- 0.01 e W <- c(0.1, 0, 1)

```
library("plyr")
data("iris")
dataset<-iris[which(iris$Species!="versicolor"),][1:2-3]
classes<-as.numeric(dataset$Species)
classes_bin<-mapvalues(classes,from=c(1,3),to=c(0,1))
print((classes_bin))
```

```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [38] 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
x1<-dataset[,1]/max(dataset[,1])
x2<-dataset[,2]/max(dataset[,2])

X <- cbind(x1, x2, rep(1, length(x1)))
Y <- classes_bin

W <- c(0.1, 0, 1)
```

```r
eta <- 0.01
epoca_max <- 20

erro_ite <- rep(0, nrow(X))
erro_total <- rep(0, epoca_max)

for (epoca in 1:epoca_max) {
  for (i in 1:nrow(X)) {
    # Calcular a saída da rede neural
    v <- sum(X[i, ] * W)
    if (v > 0) {
      y_calc <- 1
    } else {
      y_calc <- 0
    }
    # Calcular o erro
    erro <- Y[i] - y_calc
    # Atualizar os pesos da rede neural
    delta <- eta * erro * X[i, ]
    W <- W + delta
    # Armazenar o erro para a iteração atual
    erro_ite[i] <- erro ^ 2
  }
  erro_total[epoca] <- sum(erro_ite)
  if (sum(erro_ite) == 0) {
    break
  }
}

print(paste("Número de épocas usadas no treinamento", epoca, "de um máximo de", epoca_max))
```

```
## [1] "Número de épocas usadas no treinamento 9 de um máximo de 20"
```

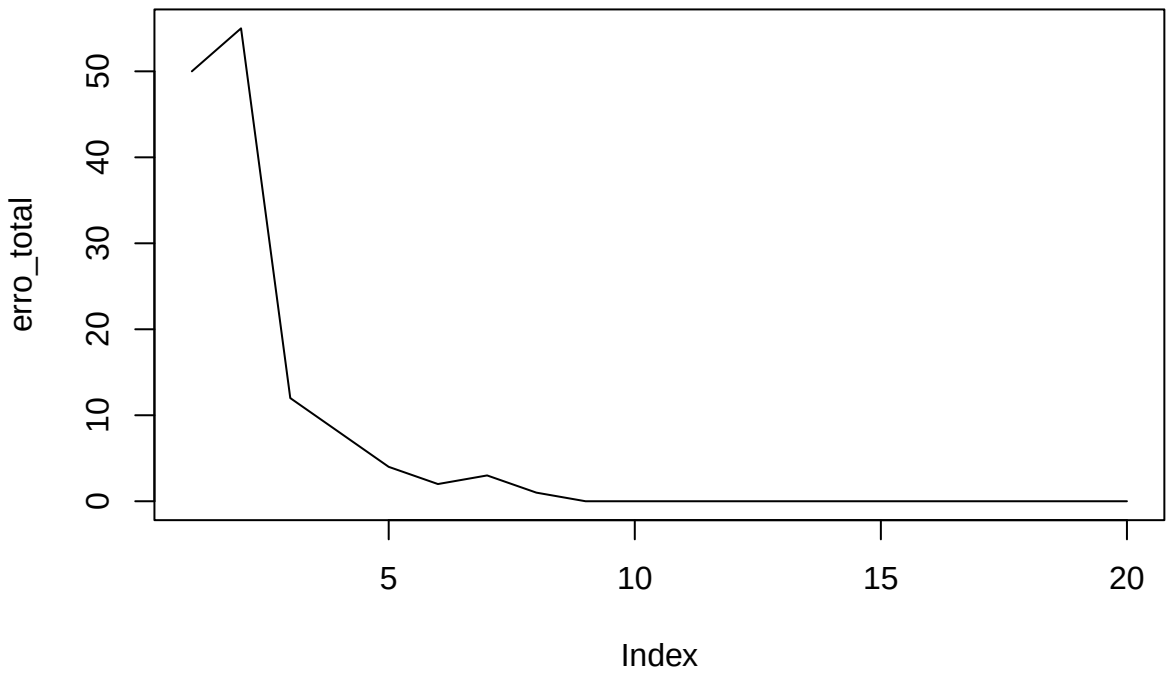
```r
print(W)
```

```
##           x1           x2
## -0.007101449  0.029600000 -0.010000000
```

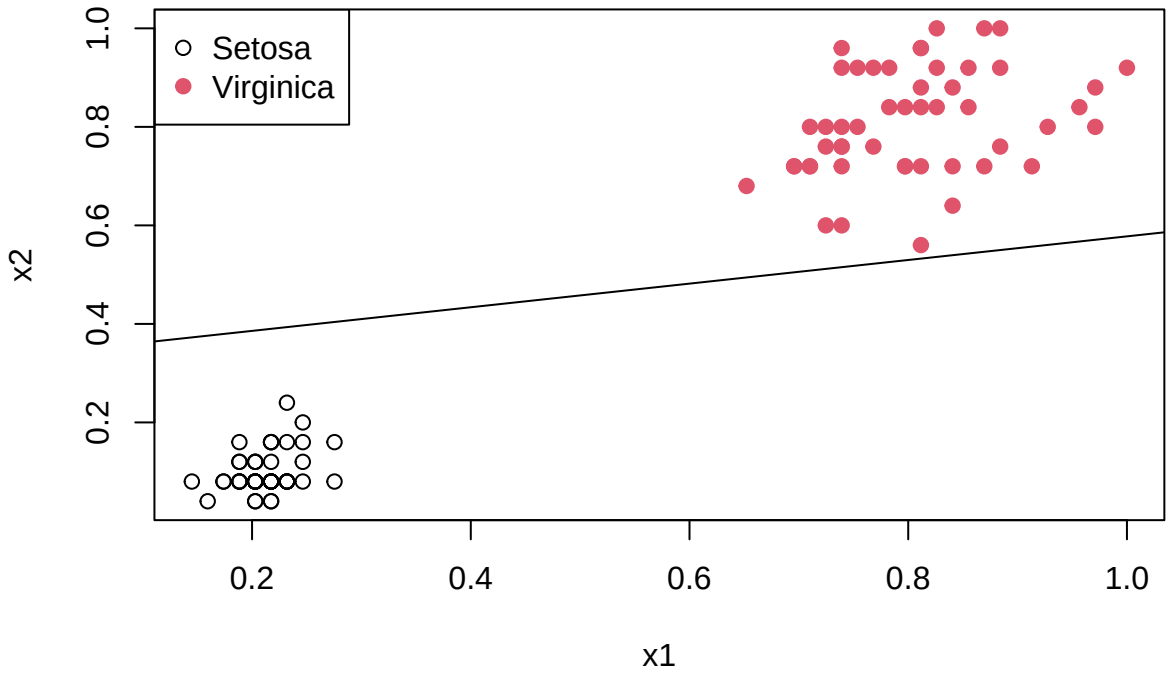
```r
print(erro_total)
```

```
##  [1] 50 55 12  8  4  2  3  1  0  0  0  0  0  0  0  0  0  0  0  0
```

```r
plot(erro_total, type = "l")
```




```
plot(x1, x2, pch = ifelse(classes_bin == 0, 1, 19), col = classes_bin + 1)
legend("topleft", legend = c("Setosa", "Virginica"), col = c(1, 2), pch = c(1, 19))
x_1 <- -(W[3] / W[1])
x_2 <- -(W[3] / W[2])
abline(a = x_2, b = -W[1] / W[2])
```



No gráfico podemos ver que o algoritimo aprende e coloca a linha de separação entre os sets de setosa e virginica, com o erro tendo a curva de aprendizado exponencialmente acentuada à zero antes das cinco iterações. Após as cinco primeiras iterações vemos que ele sobe o erro para baixar tendendo a zero pós as 9 iteraçoes.