Universidade Federal de Minas Gerais

Exercício 4

Guilherme Vinícius Amorim

2017089081

Exercício 1:

Inicialmente, deve-se amostrar duas distribuições normais no espaço $\mathrm{R2}$, ou seja, duas distribuições com duas variáveis cada. As distribuição são caracterizadas como $\mathcal{N}(2, 2, \mathcal{O}2)$ e $\mathcal{N}(4, 4, \mathcal{O}2)$:

```
# Creating data for learning.
# Xc1: data type 'blue'
# Xc2: data type 'red'
# nc: Number of points of each class
# s1, s2: How these points are dispersed

s1<-0.4
s2<-0.4
nc<-200
Xc1<-matrix(s1*rnorm(nc*2)+2,ncol=2)
Xc2<-matrix(s2*rnorm(nc*2)+4,ncol=2)
```

Figura 1: Rotina em R que implementa as duas classes com distribuições normais.

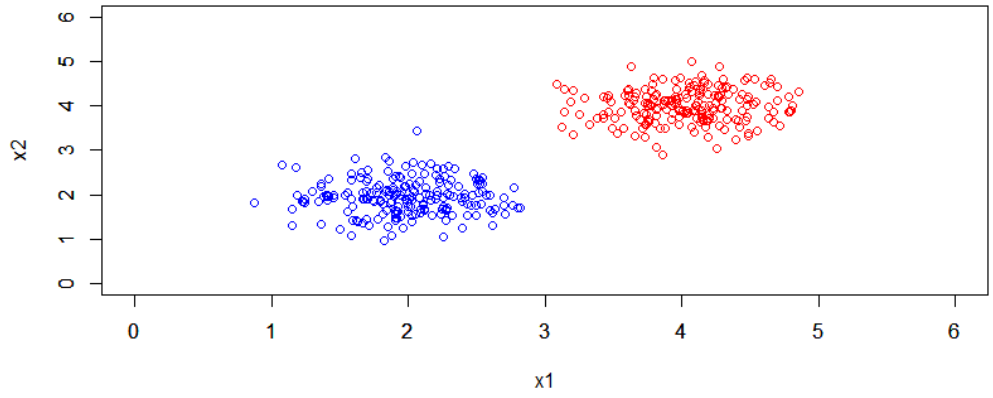

Figura 2: Classes um e dois dispersas no espaço R2.

Após amostrarmos as classes um e dois no espaço R2, a rotina do Perceptron foi utilizada para se encontrar o vetor **w** de parâmetros e, por conseguinte, a equação da reta que permite a classificação das classes. A rotina do Perceptron foi utilizada com os seguintes parâmetros:

$$\eta = 0.001$$
$$tolerância = 0.00000001$$
$$Número\ \max de\ épocas = 3000$$

Portanto, os parâmetros do vetor **w**, assim como os coeficientes da equação de reta que separa as duas classes são:

```
> W
                [,1]
[1,] -0.094711020
[2,]  0.028215177
[3,]  0.003155389
```

Figura 3: Vetor de parâmetros retornado pela rotina do Perceptron.

$$y = ax + b$$
$$\boldsymbol{w} = [-\theta, w1, w2]^{\boldsymbol{T}} = [-0.0947, 0.02822, 0.0032]^{\boldsymbol{T}}$$
$$a = -\frac{w1}{w2}$$
$$b = \theta / w2$$
$$y = -8.9419 * x + 30.0156$$

Segue abaixo o gráfico que mostra as classes um e dois no espaço R2, assim como a reta obtida pela rotina do Percepton:

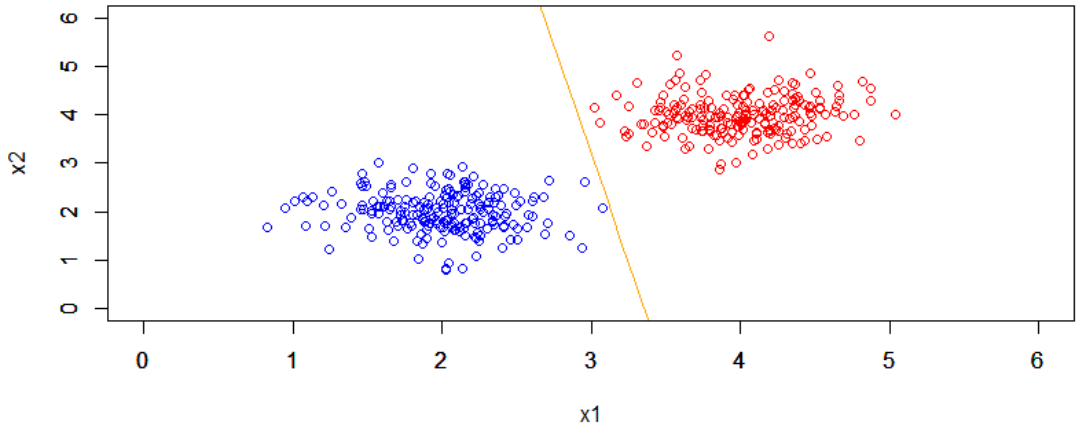

Figura 4: Classes um e dois e reta obtida pela rotina do Perceptron.

Além disso, para uma melhor ilustração de como a rotina do Perceptron consegue separar as classes um e dois, um *grid* foi criado de forma tal que, a partir de um gráfico

3D, separa-se possíveis novas entradas a partir da resposta linear obtida pelo Perceptron:

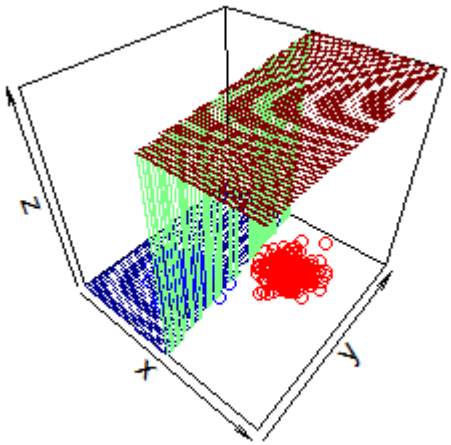

Figura 5: *Grid* com a separação entre classes um e dois.

Exercício 2:

Nesta segunda atividade o aluno deverá criar um conjunto de amostras de cada uma das duas distribuições do Exercício 1, ou seja, 200 amostras da classe 1 e 200 amostras da classe 2:

```
# Creating data for learning.
# Xc1: data type 'blue'
# Xc2: data type 'red'
# nc: Number of points of each class
# s1, s2: How these points are dispersed

s1<-0.4
s2<-0.4
nc<-200
Xc1<-matrix(s1*rnorm(nc*2)+2,ncol=2)
Xc2<-matrix(s2*rnorm(nc*2)+4,ncol=2)
```

Figura 6: Rotina em R que gera 200 amostras da classe 1 e 200 amostras da classe 2.

O aluno deverá utilizar essas amostras para criar dois conjuntos **balanceados**, um chamado de conjunto de treinamento que será usado para achar os pesos w e outro chamado de teste que servirá para avaliar a performance do seu separador dado pelos pesos encontrados no treinamento. O conjunto de treinamento irá conter 70% das amostras e o de teste 30%.

```
# Separating 70% of the data for training and 30% for test
index <- sample(1:200, size = 140, replace = FALSE)
X1Training <- as.matrix(Xc1[index,])
X1Test <- as.matrix(Xc1[-index,])

index <- sample(1:200, size = 140, replace = FALSE)
X2Training <- as.matrix(Xc2[index,])
X2Test <- as.matrix(Xc2[-index,])
```

Figura 7: Rotina em R que separa 70% dos dados para o treinamento e 30% dos dados para teste.

Após a separação dos dois conjuntos o aluno usará o conjunto de treinamento para encontrar os pesos do Perceptron e utilizará o conjunto de teste para avaliar a performance do Perceptron simples. Apresente a acurácia e a matriz de confusão.

O conjunto de dados de treinamento, que possui 140 dados da classe um e 140 dados da classe dois, foi utilizado para o treinamento do Perceptron. Os seguintes parâmetros do Perceptron foram utilizados:

$$\eta = 0.001$$

$$tolerância = 0.00000001$$

$$Número\max de\ épocas = 3000$$

Assim, o vetor **w** obtido foi:

```
> W
             [,1]
[1,] -0.18351005
[2,]  0.03193917
[3,]  0.03358862
```

Figura 8: Vetor de parâmetros **w** obtido pela rotina de treinamento.

Utilizando, agora, o conjunto de dados de teste, que possui 60 dados da classe um e 60 dados da classe 2, a acurácia encontrada, o erro médio quadrático final, assim como o erro percentual foram:

```
> accuracy
[1] 1
> error[length(error)]
[1] 0
> percentual_error
[1] 0
```

Figura 9: Acurácia, Erro quadrático médio e Erro Percentual obtidos na rotina de treinamento.

Segue abaixo um gráfico que acompanha a evolução do erro médio quadrático durante as épocas do treinamento:

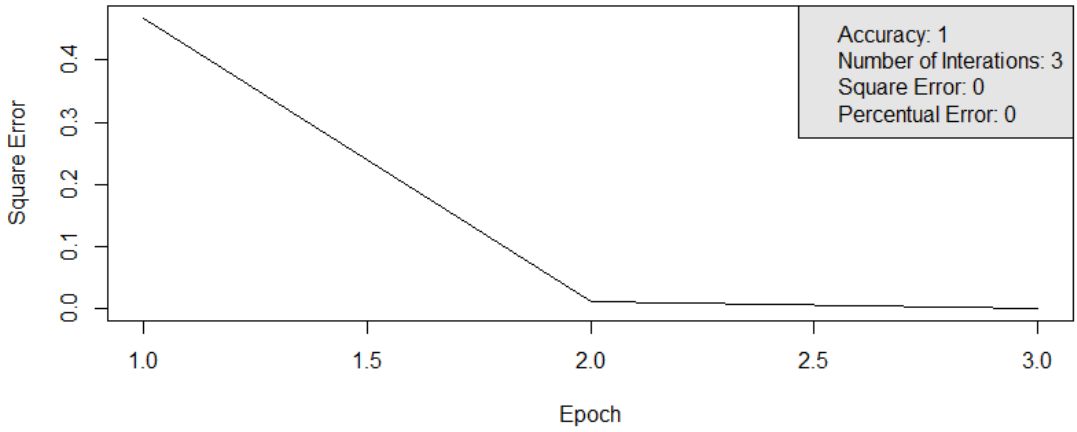

Figura 10: Gráfico que relaciona erro médio quadrático com a respectiva época durante o treinamento do Perceptron.

Exercício 3:

No Exercício 3 iremos trabalhar com uma base de dados conhecida como Iris (comando: *data*("*iris*")). Essa base de dados possui 150 amostras e 4 características, sendo 50 para cada uma das três espécies de plantas que constitui a base. Nesta atividade o aluno irá realizar o treinamento do *Perceptron* para separar a espécie 1 (50 primeiras amostras) das outras duas espécies e avaliar o desempenho do mesmo. Com isso a espécie 1 será a Classe 1 e o conjunto das espécies 2 e 3 será a Classe 2. O aluno deverá então:

1. Importar as funções *yperceptron* e *trainperceptron* desenvolvida por ele em sala de aula.
2. Carregar os dados da Iris e armazená-los, sendo que a Classe 1 será composta das 50 primeiras amostras e a Classe 2 das 100 amostras posteriores as 50 primeiras, como descrito na introdução do problema.

```
source("C:/Users/gvamorim/Google Drive/UFMG/Redes Neurais Artificiais/Algorithms/trainperceptron.R")
source("C:/Users/gvamorim/Google Drive/UFMG/Redes Neurais Artificiais/Algorithms/yperceptron.R")

data_iris<-iris

#---------------------------------------------------------------------

# Creating data for learning.
# X1: data type 0
# X2: data type 1

X1<-as.matrix(data_iris[1:50,1:4])
X2<-as.matrix(data_iris[51:150,1:4])
```

Figura 11: Importação das funções *yperceptron* e *trainperceptron* e separação entre as classes um e dois.

3. Rotular as amostras da Classe 1 com o valor de 0 e as amostras da Classe 2 com o valor 1.

```
# Creating yd:
yd<-c(rep(0,35),rep(1,70))
```

Figura 12: Rotulando a Classe um com o valor de 0 e a Classe dois com o valor de 1

4. Selecionar aleatoriamente 70% das amostras para o conjunto de treinamento e 30% para o conjunto de teste, para cada uma das duas classes.

```
# Separating 70% of the data for training and 30% for test
index <- sample(1:50, size = 35, replace = FALSE)
X1Training <- as.matrix(X1[index,])
X1Test <- as.matrix(X1[-index,])

index <- sample(1:100, size = 70, replace = FALSE)
X2Training <- as.matrix(X2[index,])
X2Test <- as.matrix(X2[-index,])
```

Figura 13: Separando 70% das amostras para treinamento e 30% das amostras para o conjunto de teste.

5. Utilizar as amostras de treinamento para fazer o treinamento do *Perceptron* utilizando a função *trainperceptron*.

   A rotina do Perceptron foi feita com os dados de treinamento e com os seguintes parâmetros:

$$\eta = 0.001$$
$$tolerância = 0.00000001$$
$$Número\ \max de\ épocas = 3000$$

6. Extrair o vetor de pesos da função *trainperceptron*.

```
> W
             [,1]
[1,]  0.10021170
[2,] -0.50749411
[3,] -0.05474149
[4,]  1.26434351
[5,] -0.06191491
```

Figura 14: Vetor de pesos encontrado pela rotina de treinamento.

7. Concatenar as amostras de teste e seus respectivos *y* e dar entrada na função *yperceptron* (a função *yperceptron* não recebe o y), utilizando o vetor de peso extraído.

```
# Xtest
numberOfSamples<-45
numberOfDimensions<-4
Xtest<-matrix(rbind(X1Test,X2Test), nrow = numberOfSamples, ncol = numberOfDimensions)

# Creating y:
y<-c(rep(0,15),rep(1,30))

# Calculating yhat
yhat<-as.double((cbind(1,Xtest) %*% W) >= 0)
```

Figura 15: Calculando o $\hat{y}$ dos dados separados para teste.

8. Calcular o erro percentual. (O erro é dado pelo número de amostras de teste classificadas de forma errada).
9. Imprimir a matriz de confusão.

```
# Calculating Percentual error
percentual_error <- 100*sum(abs(yhat-y))/45

# Calculating the confusion Matrix
Confusion<-table(y,yhat)
```

Figura 16: Calculando erro percentual e a matriz de Confusão.

```
> percentual_error
[1] 0
> Confusion
   yhat
y    0  1
  0 15  0
  1  0 30
```

Figura 17: Erro percentual e a matriz de Confusão encontrados.

10. Crie um loop para repetir 100 vezes os itens 4-8, armazenando o valor do erro percentual do item 8. Plote o erro percentual em função do número de iteração e imprima o valor da variância do erro.

A partir da repetição dos itens 4-8 100 vezes em um loop e armazenando o erro percentual em um vetor, segue abaixo o gráfico relacionando erro percentual e o número de iteração, assim como o valor da variância do erro foi obtido:

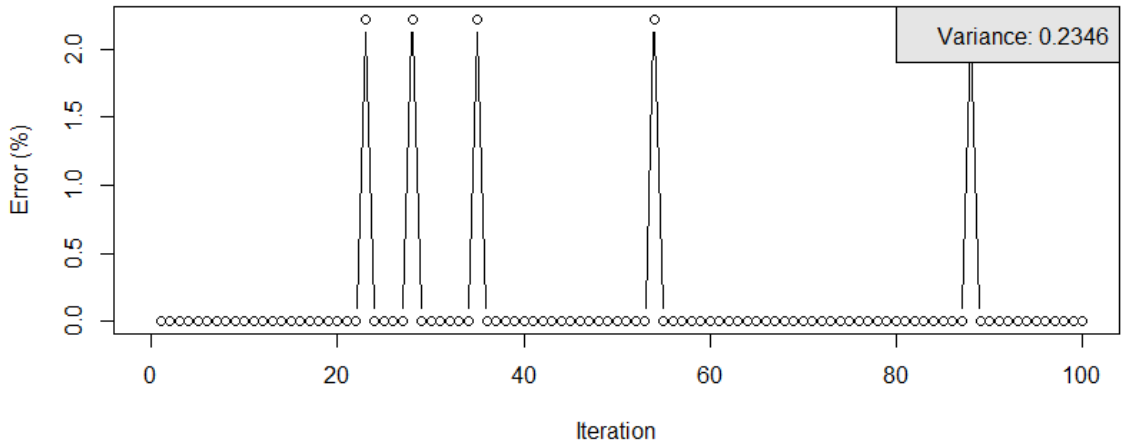

Figura 18: Gráfico relacionando o erro percentual por interação e o valor da variância desse erro.

.