Exercício 6.
Nome: Lucas Kou Kinoshita
RM: 2019021557
data: 22/04/2025

Parte 1.

O primeiro conjunto de dados utilizado foi o *"Breast Cancer (diagnosis)"*, disponível na biblioteca *MLbench*. (As funções de treino e cálculo do resultado da ELM foram implementadas como sugerido nas notas de aula).

```r
ELM <- function(xin, yin, p, par) {
  n <- dim(xin)[2]

  if(par == 1){
    xin <- cbind(1, xin)
    Z <- matrix(runif((n+1)*p, -0.5, 0.5), nrow = (n+1), ncol = p)
  } else {
    Z <- matrix(runif(n*p, -0.5, 0.5), nrow = n, ncol = p)
  }

  H <- tanh(xin %*% Z)
  Haug <- cbind(1, H)

  w <- pseudoinverse(Haug) %*% yin

  return( list(w, H, Z))
}

## saída ELM para valores -1 e +1
YELM <- function(xin, Z, W, par){
  n <- dim(xin)[2]

  if (par == 1){
    xin <- cbind(1, xin)
  }

  H <- tanh(xin %*% Z)
  Haug <- cbind(1,H)
  Yhat <- sign(Haug %*% W)

  return(Yhat)
}
```

```r
# Load BreastCancer
# tratamento de dados
data("BreastCancer")
df <- BreastCancer
df <- df[, -1]  # drop 'Id'
df <- na.omit(df)

df[, 1:9] <- lapply(df[, 1:9], function(x) as.numeric(as.character(x)))
df$Class <- ifelse(df$Class == "malignant", 1, -1)

X <- as.matrix(df[, 1:9])
Y <- as.matrix(df$Class)
N <- nrow(X)

ntrain <- nrow(df)*0.7
reps <- 10
p <- 300
acc_treino <- numeric(reps)
acc_teste <- numeric(reps)
```

Um *loop* de dez repetições foi elaborado para que fosse possível obter a acurácia média (e seu desvio padrão) de treinamento e teste.

```r
for (r in 1:reps) {
  # Shuffle indices
  idx <- sample(N)

  # Train/Test split
  train_idx <- idx[1:ntrain]
  test_idx <- idx[(ntrain + 1):N]

  xin <- X[train_idx, ]
  yin <- Y[train_idx]
  xinteste <- X[test_idx, ]
  yteste <- Y[test_idx]

  # Train ELM
  retlist <- ELM(xin, yin, p, par = 1)
  w <- retlist[[1]]
  Z <- retlist[[3]]

  yhat_train <- YELM(xin, Z, w, 1)
  yhat_test <- YELM(xinteste, Z, w, 1)

  # Cálculo da acurácia
  acc_treino[r] <- mean(yhat_train == yin)
  acc_teste[r] <- mean(yhat_test == yteste)

}
```

Por fim, os resultados obtidos são expostos no final do script com quatro casas decimais de aproximação (listados na Tabela 1).

```r
# Resultados finais
cat("\nAcurácia média (treino):", round(mean(acc_treino), 4),
    "±", round(sd(acc_treino), 4), "\n")
cat("Acurácia média (teste):", round(mean(acc_teste), 4),
    "±", round(sd(acc_teste), 4), "\n")
```

Nº de neurônios	Acurácia e desvio padrão (treino)	Acurácia e desvio padrão (teste)
5	0.8925 ± 0.0189	0.8775 ± 0.0266
10	0.9167 ± 0.0141	0.9196 ± 0.0237
30	0.9573 ± 0.0076	0.9235 ± 0.0234
50	0.9707 ± 0.0042	0.9431 ± 0.0131
100	0.9839 ± 0.0034	0.9451 ± 0.012
300	1 ± 0	0.8245 ± 0.0474

Tabela 1: resultados obtidos pela ELM para o *dataset BreastCancer*

De forma semelhante, o *dataset Statlog (Heart)* também foi explorado, como este não está disponível imediatamente no *mlbench*, o processo de importação e tratamento foi levemente alterado, embora o restante dos procedimentos tenham permanecido idênticos.

```r
# Load statlog (Heart)
# tratamento
df <- read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/heart/hear
colnames(df) <- c(
  "age", "sex", "chest_pain", "rest_bp", "chol", "fbs", "rest_ecg",
  "max_hr", "ex_angina", "oldpeak", "slope", "ca", "thal", "target"
)
df <- na.omit(df)
df$target <- ifelse(df$target == 1, -1, 1)

df[, 1:9] <- lapply(df[, 1:14], function(x) as.numeric(as.character(x)))

X <- as.matrix(df[, 1:14])
Y <- as.matrix(df$target)
N <- nrow(X)
```

Nº de neurônios	Acurácia e desvio padrão (treino)	Acurácia e desvio padrão (teste)
5	0.6143 ± 0.0566	0.5593 ± 0.0632
10	0.6376 ± 0.0389	0.5877 ± 0.0463
30	0.6603 ± 0.0358	0.6099 ± 0.0369
50	0.6968 ± 0.0411	0.6469 ± 0.0484
100	0.745 ± 0.0309	0.658 ± 0.0511
300	0.8751 ± 0.0295	0.6444 ± 0.0637

Tabela 2: resultados obtidos pela ELM para o *dataset Statlog (Heart)*

Parte 2.

Ao explorar os mesmos datasets, porém com a rotina desenvolvida para o treinamento a partir de um perceptron obtemos os seguintes resultados (com *loops* de dez repetições):

```
perceptron <- function(X, y, eta, tol, maxepocas, par) {
  N <- nrow(X)
  n <- ncol(X)
  error_curve <- numeric(maxepocas)

  if(par == 1){
    w <- as.matrix(runif(n+1) - 0.5)
    X <- cbind(1,X)
  } else {
    w <- as.matrix(runif(n) - 0.5)
  }

  nepocas <- 0
  erroepoca <- tol + 1

  while (erroepoca > tol && nepocas < maxepocas) {
    xseq <- sample(N)
    ei2 <- 0

    for (i in xseq) {
      yhat <- 1.0*((t(w) %*% X[i, ]) >= 0)
      erro <- y[i] - yhat
      dw <- eta * erro * X[i, ]
      w <- w + dw
      ei2 <- ei2 + erro^2
    }

    error_curve[nepocas] <- ei2 / N
    nepocas <- nepocas + 1
  }

  list(weights = w, error = error_curve[1:(nepocas - 1)])
}
```

```r
for (r in 1:reps) {
  # Shuffle indices
  idx <- sample(N)

  # Train/Test split
  train_idx <- idx[1:ntrain]
  test_idx <- idx[(ntrain + 1):N]

  xin <- X[train_idx, ]
  yd <- y[train_idx]
  xinteste <- X[test_idx, ]
  yteste <- y[test_idx]

  # Train perceptron
  retlist <- perceptron(xin, yd, eta = 0.1, tol = 0.01, maxepocas = 100, par = 1)
  wt <- retlist$weights

  # Train accuracy
  xitreino <- cbind(1, xin)
  ytreino_pred <- 1 * ((xitreino %*% wt) >= 0)
  acc_treino[r] <- 1 - mean((yd - ytreino_pred)^2)

  # Test accuracy
  xiteste <- cbind(1, xinteste)
  yteste_pred <- 1 * ((xiteste %*% wt) >= 0)
  acc_teste[r] <- 1 - mean((yteste - yteste_pred)^2)
  print(r)
}
```

Dataset	Acurácia e desvio padrão (treino)	Acurácia e desvio padrão (teste)
BreastCancer	0.9642 ± 0.0251	0.9569 ± 0.0323
Statlog (Heart)	0.6958 ± 0.1058	0.6531 ± 0.0758

Tabela 3: Resultado do treinamento com perceptron para os dois *datasets* em dez *loops*

Para valores de $\eta = 0.1$ e cem épocas, a acurácia para o *dataset Statlog (Heart)* pode ser considerada insuficiente, no entanto, ao realizar o treinamento com valores de $\eta = 0.01$ e quinhentas épocas (mantendo-se a quantidade de dez *loops*), obtemos:

Dataset	Acurácia e desvio padrão (treino)	Acurácia e desvio padrão (teste)
BreastCancer	0.9322 ± 0.0937	0.9358 ± 0.0816
Statlog (Heart)	0.7926 ± 0.1475	0.7741 ± 0.126

Tabela 4: Resultado após ajustes de parâmetros para o perceptron

Conclusão:.

Ao comparar os resultados do treinamento utilizando a ELM nos conjuntos de dados BreastCancer e Statlog (Heart), observamos que a ELM apresentou melhor desempenho com cerca de 100 neurônios ocultos, proporcionando um bom equilíbrio entre acurácia média nos dados de treino e teste. Para o *dataset BreastCancer* em específico, essa configuração resultou nas melhores acurácias com baixa variância (~1,2%), enquanto configurações com um número muito alto de neurônios (próximo de 300) demonstraram sinais de overfitting, com acurácia perfeita no treino e queda significativa no teste.

De forma semelhante, para o *dataset Statlog (Heart)*, a ELM também apresentou desempenho mais estável com aproximadamente 100 neurônios, embora as acurácias obtidas nesse conjunto tenham sido mais modestas, com um valor máximo de 65,8% nos dados de teste. Isso sugere uma maior complexidade ou menor separabilidade linear dos dados neste caso.

Por outro lado, no treinamento com o perceptron, observou-se que, ao variar o número de épocas, foram obtidos resultados com acurácia satisfatória. Sendo necessárias aproximadamente 100 épocas para o *dataset BreastCancer* e cerca de 500 épocas para o *Statlog (Heart)*, considerando uma taxa de aprendizado de 0.01 para ambos. Apesar de apresentar resultados satisfatórios em ambos os conjuntos, o perceptron exige maior tempo de execução em comparação à ELM, cujo treinamento se mostrou consideravelmente mais rápido e eficiente em termos computacionais.