```
################################################################################
# LOAD THE DNA DATA AND CONCATENATE SEQUENCES
#
# function: correct.dna()
#
# parameter: cut.matrix (default=FALSE), a Boolean which indicates whether or
#   not to return a reduced quantity of DNA data
#
# return value: a list containing three elements
#    dna: a data frame containing the source data
#    sequence: a string containing the full concatenation of all 60-nucleotide
#      seqequences
#   nucleotids: a vector containing individual characters of all nucleotides
################################################################################
correct.dna <- function(cut.matrix = FALSE) {

  txt1 <- 'https://archive.ics.uci.edu/ml/machine-learning-databases/'
  txt2 <- 'molecular-biology/splice-junction-gene-sequences/splice.data'

  rfile <- paste(txt1, txt2, sep = '')
  dna <- read.table(rfile, sep = ',')

  if (cut.matrix == TRUE) {
    set.seed(666)
    ns <- which(dna[, 1] == 'N')
    dna <- dna[-sample(ns, length(ns) - 768), ]
  }

  for (col in 1:ncol(dna)) {
    if (is.factor(dna[, col])) {
      dna[, col] <- gdata::trim(as.character(dna[, col]))
    }
  }

  sequence <- paste(dna[, 'V3'], collapse = '')
  nucleotids <- unlist(strsplit(sequence, ''))

  # Which positions are each letter
  as <- gregexpr('A', sequence)[[1]] + 1
  ts <- gregexpr('T', sequence)[[1]] + 1
  cs <- gregexpr('C', sequence)[[1]] + 1
  gs <- gregexpr('G', sequence)[[1]] + 1

  tta <- table(nucleotids[as])
  ttt <- table(nucleotids[ts])
  ttc <- table(nucleotids[cs])
  ttg <- table(nucleotids[gs])

  tta <- tta[names(tta) %in% c('A', 'C','G', 'T')]
  ttt <- ttt[names(ttt) %in% c('A', 'C','G', 'T')]
  ttc <- ttc[names(ttc) %in% c('A', 'C','G', 'T')]
  ttg <- ttg[names(ttg) %in% c('A', 'C','G', 'T')]

  sa <- c(rep('A', tta[1]), rep('C', tta[2]),
          rep('G', tta[3]), rep('T', tta[4]))
  st <- c(rep('A', ttt[1]), rep('C', ttt[2]),
          rep('G', ttt[3]), rep('T', ttt[4]))
  sc <- c(rep('A', ttc[1]), rep('C', ttc[2]),
          rep('G', ttc[3]), rep('T', ttc[4]))
  sg <- c(rep('A', ttg[1]), rep('C', ttg[2]),
          rep('G', ttg[3]), rep('T', ttg[4]))

  set.seed(666)
  nucleotids[which(nucleotids == 'S')] <- sample(c('C', 'G'), 1)
  nucleotids[which(nucleotids == 'R')] <- sample(c('A', 'G'), 1)
  nucleotids[which(nucleotids == 'D')] <- sample(c('A', 'C', 'G'), 2)
  repeat {
    to.correct <- which(nucleotids == 'N') - 1

    for (i in to.correct) {
      base <- nucleotids[i]
      if (base == 'T') {
        nucleotids[i + 1] <- sample(st, 1)
      } else if (base == 'A') {
        nucleotids[i + 1] <- sample(sa, 1)
      } else if (base == 'C') {
        nucleotids[i + 1] <- sample(sc, 1)
      } else if (base == 'G') {
        nucleotids[i + 1] <- sample(sg, 1)
      }
    }

    if (sum(! nucleotids %in% c('A', 'C', 'T', 'G')) == 0) {
      break
    }
  }

  csequence <- paste(nucleotids, collapse = '')
  cdna <- dna
  for (r in 1:nrow(dna)) {
    i <- 60 * (r - 1) + 1
    cdna[r, 3] <- substr(csequence, i, i + 59)
  }

  return(invisible(list(dna = cdna,
                        sequence = csequence,
                        nucleotids = nucleotids)))
}
```