

POETA

2022-12-15

cargamos la librería que vamos a utilizar para dividir las palabras en sílabas

```
library(syllly)
```

```
## Warning: package 'syllly' was built under R version 4.2.2
```

```
#available.syllly.lang()  
#install.syllly.lang("es")  
library (syllly.es)
```

Creamos una función para pasarle los sonetos y que los divida en las estrofas correspondientes (estrofas formadas por 14 versos)

```
analizar_sonetos <- function(soneto){  
  con <- file(description = soneto, open = 'rt', encoding= 'UTF-8')  
  soneto <- readLines(con)  
  soneto <- paste (soneto, sep = "\n")  
  close(con)
```

```
#reemplazo los signos de puntuación por ""  
soneto <- gsub("[:punct:]", "", soneto)
```

```
#convierto a minúsculas el soneto  
soneto <- tolower(soneto)
```

```
#obtenemos las estrofas del soneto mandado  
cont = 1  
estrofas <- list()  
for (i in 1:(trunc(length(soneto)/14))){  
  estrofas[[i]] <- soneto[cont:(cont+13)]  
  cont = cont+14  
}  
estrofas  
}
```

```
estrofas <- analizar_sonetos('soneto_web.txt')
```

```
#Vemos la primera estrofa, por ejemplo  
estrofas[9]
```

```
## [[1]]  
## [1] "dilata tu nacer para la vida"
```

```
## [2] "que anticipas tu ser para tu muerte"
## [3] "ya besando unas manos cristalinas"
## [4] "ya anudándose a un blanco y liso cuello"
## [5] "ya esparciendo por el aquel cabello"
## [6] "que amor sacó entre el oro de sus minas"
## [7] "ya quebrando en aquellas perlas finas"
## [8] "palabras dulces mil sin merecello"
## [9] "ya cogiendo de cada labio bello"
## [10] "purpúreas rosas sin temor de espinas"
## [11] "estaba oh claro sol invidioso"
## [12] "cuando tu luz hiriendome los ojos"
## [13] "mató mi gloria y acabó mi suerte"
## [14] "si el cielo ya no es menos poderoso"
```

Creamos una función para contar las sílabas de cada verso, la cual devuelve una lista formada por objetos complejos (S4)

```
#Debemos pasarle una estrofa solamente(14 versos)
contarSilabas <- function(estrofa){
  palabras <- list()
  silabas <- list()
  for (linea in 1:14){
    palabras[[linea]] <- unlist(strsplit(estrofa[linea], " "))
    silabas[[linea]] <- hyphen(unlist(palabras[linea]), hyph.pattern="es", min.length = 1, quiet= TRUE)
  }
  silabas
}
```

```
#estrofas es una lista de 1 o varios elementos, donde cada uno corresponde a una estrofa
estrofas_silabas <- list()
for (i in 1:length(estrofas)){
  estrofas_silabas[[i]] <- contarSilabas(unlist(estrofas[i]))
}
#estrofas_silabas es una lista formada 14 listas
 #(ya que hay 14 versos), donde cada una corresponde a un elemento complejo (S4).
```

```
#Mostramos la separación por sílabas del primer verso de primera estrofa
estrofas_silabas[[1]][[1]]@hyphen[["word"]]
```

```
## [1] "oh"           "ex-cel-so"    "mu-ro"        "oh"           "to-rres"
## [6] "co-ro-na-das"
```

```
#También podemos obtener el número de sílabas por verso
estrofas_silabas[[1]][[1]]@desc[["num.syll"]]
```

```
## [1] 13
```

Obtenemos una lista con el número de sílabas de cada verso.

```
numero_silabas <- list()
for (i in 1:length(estrofas)){
```

```

silabas <- list()
for (j in 1:14){
  silabas[[j]] <- unlist(estrofas_silabas[[i]][[j]]@desc[["num.syll"]])
}
numero_silabas[[i]] <- unlist(silabas)
}

```

```

#mostramos el número de sílabas de cada verso en la primera estrofa,
#sin tener en cuenta aún las normas de la métrica (se aplicarán posteriormetne)
numero_silabas[[1]]

```

```
## [1] 13 11 10 11 11 13 9 11 11 12 10 10 10 14
```

Obtenemos una lista la cual contenga como elemento la última palabra de cada verso

```

#Creamos una funcion para ello, hay que pasarle la separación en sílabas de una única estrofa
ultima_palabra <- function(silabas){
  lista_ultimas <- list()
  for (i in 1:14){
    verso <- silabas[[i]]@hyphen[["word"]]
    lista_ultimas[[i]] <- verso[length(verso)]
  }
  lista_ultimas
}

```

```

#obtenemos una lista con las últimas palabras de cada verso en cada estrofa
ultimas_total <- list()
for (i in 1:length(estrofas)){
  ultimas_total[[i]] <- unlist(ultima_palabra(estrofas_silabas[[i]]))
}

```

```

#Mostramos la lista formada por las últimas palabras de la primera estrofa por ejemplo
ultimas_total[[1]]

```

```

## [1] "co-ro-na-das" "ga-llar-día" "an-da-lu-cía" "do-ra-das"
## [5] "le-van-ta-das" "día" "mía" "es-pa-das"
## [9] "des-po-jos" "ba-ña" "mío" "ojos"
## [13] "río" "es-pa-ña"

```

Creamos una función para detectar la existencia de sinalefas de los versos

```

library(stringr)
library(useful)

```

```
## Warning: package 'useful' was built under R version 4.2.2
```

```
## Loading required package: ggplot2
```

```

#le pasamos la lista de sílabas soneto a soneto (14 versos)
sinalefas <- function(silabas_soneto){
  #lista_sinalefas <- list()
  lista_inicio <- list()
  lista_fin <- list()
  for (i in 1:14){
    lista_inicio[[i]] <- str_detect(silabas_soneto[[i]]@hyphen[["word"]], pattern = "^[aeiouáéíóúh])|(
    lista_fin[[i]] <- str_detect(silabas_soneto[[i]]@hyphen[["word"]], pattern = "([aeiouáéíóú$)|(y$)".
  }
  lista_sinalefas <- list(lista_inicio, lista_fin)
}

```

```

sinalefas_cont <- function(lista_inicio, lista_fin){
  cont <- unlist(list((1:14)*0))
  for (i in 1:14){
    inicio <- lista_inicio[[i]][2:length(lista_inicio[[i]])]
    fin <- lista_fin[[i]][1:(length(lista_inicio[[i]])-1)]
    for (j in 1:length(inicio)){
      if (inicio[j] == TRUE & fin[[j]] == TRUE){
        cont[[i]] <- cont[[i]] + 1
      }
    }
  }
  cont
}

```

```

#Obtenemos una lista de listas, donde cada elemento representa a un soneto y dentro de cada elemento (l
contador_sinalefas <- list()
for (i in 1:length(estrofas)){
  posibles_sinalefas <- sinalefas(estrofas_silabas[[i]])
  contador_sinalefas[[i]] <- sinalefas_cont(posibles_sinalefas[[1]], posibles_sinalefas[[2]])
}
contador_sinalefas

```

```

## [[1]]
## [1] 1 1 1 1 1 3 0 0 2 1 1 0 0 4
##
## [[2]]
## [1] 1 1 1 1 1 2 0 1 3 2 0 2 2 2
##
## [[3]]
## [1] 1 1 1 1 1 0 0 0 3 0 1 1 0 2
##
## [[4]]
## [1] 1 1 0 2 2 1 0 1 0 0 1 2 0 2
##
## [[5]]
## [1] 2 4 1 0 1 0 0 0 0 1 1 0 2 2
##
## [[6]]
## [1] 1 3 0 1 1 1 1 1 0 2 1 0 0 2
##

```

```
## [[7]]
## [1] 1 3 0 0 2 0 0 0 2 0 1 1 0 2
##
## [[8]]
## [1] 0 1 1 0 1 1 2 0 4 1 0 1 2 2
##
## [[9]]
## [1] 0 1 1 4 1 3 1 0 0 1 1 0 2 3
```

#devuelve el número de sinalefas que hay en cada verso para cada soneto

Creamos una función para conocer el número de sílabas que hay en cada verso y qué tipo de palabra es la última de cada verso (llana, aguda o esdrújula), ya que si esta es esdrújula, se resta uno a la métrica, si es aguda se suma, y si es llana se mantiene igual.

```
#Debemos pasarle la última palabra de cada verso
tipo_palabra_recuento <- function(ultima, recuento){
  #buscamos si es esdrújula, llana o aguda
  ultima <- unlist(strsplit(ultima, "-"))

  #si lleva tilde
  tilde <- grep(pattern = "[áéíóú]", rev(ultima))

  aguda_notilde <- grep(pattern = "[^nsaeiou]$", rev(ultima[1]))

  tipo_palabra <- c("llana")
  if (length(tilde) != 0){
    if (tilde >= 3){
      recuento <- recuento -1 #palabra esdrújula
      tipo_palabra = "esdrújula"
    }
    if (tilde==1){
      recuento <- recuento + 1 #palabra aguda
      tipo_palabra <- c("aguda")
    }
  }else
  {
    if (length(aguda_notilde) != 0){
      recuento <- recuento +1 #palabra aguda que no termina en vocal, n o s
      tipo_palabra <- c("aguda")
    }
  }
  sol <- c(recuento, tipo_palabra)
}
```

#devuelve una cadena, donde el primer elemento es la métrica, y el segundo el tipo de palabra

```
#Aplicamos el recuento a la lista de sílabas de cada verso (se suma uno si es esdrújula, se resta uno s
num_silabas_final <- list()
for (i in 1:length(estrofas)){
  silabas_final <- list()
  for (j in 1:14){
    ultima <- ultimas_total[[i]][j]
```

```

    recuento <- numero_silabas[[i]][j]
    silabas_final[[j]] <- tipo_palabra_recuento(ultima, recuento)[1]
  }
  num_silabas_final[[i]] <- silabas_final
}

```

```

#accedemos a cada elemento de la lista que contiene el número de sílabas de cada verso para cada soneto
silabas_menos_sinalefas <- list()
for (i in 1:length(estrofas)){
  total_silabas <- list()
  for (j in 1:14){
    total_silabas[[j]] <- unlist(as.numeric(num_silabas_final[[i]][j])) - contador_sinalefas[[i]][j]
    #es necesario pasar a tipo numérico
  }
  silabas_menos_sinalefas[[i]] <- total_silabas
}

```

```

#Mostramos el recuento actual de sílabas de la segunda estrofa y comparamos con la lista anterior
unlist(silabas_menos_sinalefas[[2]])

```

```
## [1] 8 2 11 11 10 10 11 9 11 11 11 10 10 10
```

```
unlist(num_silabas_final[[2]])
```

```
## [1] "9" "3" "12" "12" "11" "12" "11" "10" "14" "13" "11" "12" "12" "12"
```

Creamos una función para quedarnos con el final de la palabra desde la vocal tónica (necesario para ver la métrica)

```
library(stringr)
```

```

#Debemos pasarle la lista de las últimas palabras de cada verso una a una(ultimas_total)
vocal_tonica <- function(ultima, recuento){
  tipo_palabra <- tipo_palabra_recuento(ultima, recuento)[2]
  if (tipo_palabra == "esdrújula"){
    tonica <- str_match(pattern = "[áéíóú].*", ultima)
    tonica <- gsub("-", "", tonica)
  }else if (tipo_palabra == "aguda"){
    ultima <- unlist(strsplit(ultima, "-"))
    tonica <- str_match(pattern = "[aeiouáéíóú].*", rev(ultima)[1])
    tonica <- gsub("-", "", tonica)
  }else{
    ultima <- gsub("-", "", ultima)
    tonica <- str_match(pattern = "[aeiou][^(aeiou)]{1,4}[aeiou][^aeiou]{0,2}$|[aeiou][aeiou][^aeiou]{0,2}$", ultima)
  }
  tonica
}

```

```

tonicas_total <- list()
for (i in 1:length(estrofas)){
  tonicas_estrofas <- list()
}

```

```
#Mostramos la lista formada por la terminación desde la vocal tónica de la última palabra de cada verso
#Esta lista corresponde a la primera estrofa.
tonicas_total[[1]]
```

Creamos una función para comprobar la rima

7

```
}
}
```

```
#Comprobamos si cumple la rima
cumplir_rima <- list()
for (i in 1:length(estrofas)){
  cumplir_rima[[i]] <- unlist(cumpleRima(tonicas_total[[i]]))
}
```

```
#Es una lista de TRUE o FALSE, según el soneto cumpla la rima o no
cumplir_rima
```

```
## [[1]]
## [1] FALSE
##
## [[2]]
## [1] FALSE
##
## [[3]]
## [1] FALSE
##
## [[4]]
## [1] FALSE
##
## [[5]]
## [1] FALSE
##
## [[6]]
## [1] FALSE
##
## [[7]]
## [1] FALSE
##
## [[8]]
## [1] FALSE
##
## [[9]]
## [1] FALSE
```

```
#Comprobamos si cumple la métrica o no
cumplir_metrica <- list()
for (i in 1:length(estrofas)){
  resultado <- TRUE
  for (j in 1:14){
    if (silabas_menos_sinalefas[[i]][j] != 11){
      resultado <- FALSE
    }
  }
  cumplir_metrica[[i]] <- resultado
}
```



```
#Es una lista de TRUE o FALSE, según el soneto cumpla la métrica o no  
cumplir_metrica
```

```
## [[1]]  
## [1] FALSE  
##  
## [[2]]  
## [1] FALSE  
##  
## [[3]]  
## [1] FALSE  
##  
## [[4]]  
## [1] FALSE  
##  
## [[5]]  
## [1] FALSE  
##  
## [[6]]  
## [1] FALSE  
##  
## [[7]]  
## [1] FALSE  
##  
## [[8]]  
## [1] FALSE  
##  
## [[9]]  
## [1] FALSE
```

```
#Indica TRUE si es un soneto y False si no lo es (en el orden de las estrofas)  
final <- list()  
for (i in 1:length(cumplir_rima)){  
  if (cumplir_rima[i] == TRUE & cumplir_metrica[i]==TRUE){  
    final[[i]] <- TRUE  
  }else  
    final[[i]] <- FALSE  
}  
final
```

```
## [[1]]  
## [1] FALSE  
##  
## [[2]]  
## [1] FALSE  
##  
## [[3]]  
## [1] FALSE  
##  
## [[4]]  
## [1] FALSE  
##  
## [[5]]
```

```
## [1] FALSE
##
## [[6]]
## [1] FALSE
##
## [[7]]
## [1] FALSE
##
## [[8]]
## [1] FALSE
##
## [[9]]
## [1] FALSE
```

```
for (i in 1:length(final)){
  if (final[[i]] == TRUE){
    cat('El soneto', i, 'tiene la siguiente métrica = ', unlist(silabas_menos_sinalefas[[i]]), '-->',
        print("")
    cat('Tiene rima ', cumplir_rima[[i]], ',por lo que sí es un soneto. ')
    print("")
    print("")
  }else{
    cat('El soneto', i, 'tiene la siguiente métrica = ', unlist(silabas_menos_sinalefas[[i]]), '-->', c
        print("")
    cat('Tiene rima ', cumplir_rima[[i]], ',por lo que no es un soneto. ')
    print("")
    print("")
  }
}
```

```
## El soneto 1 tiene la siguiente métrica = 12 11 10 10 10 11 10 11 9 11 10 10 11 10 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
## El soneto 2 tiene la siguiente métrica = 8 2 11 11 10 10 11 9 11 11 11 10 10 10 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
## El soneto 3 tiene la siguiente métrica = 11 10 11 11 10 10 11 11 11 11 10 10 11 9 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
## El soneto 4 tiene la siguiente métrica = 10 11 11 9 10 11 11 10 10 11 11 10 10 10 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
## El soneto 5 tiene la siguiente métrica = 9 11 11 11 12 11 11 11 11 11 10 11 11 11 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
## El soneto 6 tiene la siguiente métrica = 11 10 11 9 10 11 11 10 12 11 11 11 10 10 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
## El soneto 7 tiene la siguiente métrica = 11 11 9 9 11 11 11 10 9 11 11 11 11 11 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
## El soneto 8 tiene la siguiente métrica = 11 9 10 11 11 11 10 11 10 11 10 11 10 9 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
```

```
## El soneto 9 tiene la siguiente métrica = 11 12 10 9 10 9 10 11 11 11 10 10 10 10 --> FALSE[1] ""
## Tiene rima FALSE ,por lo que no es un soneto. [1] ""
## [1] ""
```