Práctica 9 PLN

Pablo de Tarso Pedraz, Jorge López, Martín Hernández y Pablo Suárez

Parte 1: Creación del corpus

Se va a hacer scraping de la siguiente página: : https://www.conceptosjuridicos.com/codigo-penal/, la cual contiene artículos del código penal. Primero importamos las funciones creadas en los demás archivos

```
source("./data/generador.R")
source("./data/processor.R")
source("./install_requirements.R")

#writeAll() solo si necesita descargar de nuevo los archivos
```

Instalamos las dependecias

```
start_install()
library(quanteda)
```

```
## Package version: 4.1.0
## Unicode version: 14.0
## ICU version: 70.1

## Parallel computing: disabled

## See https://quanteda.io for tutorials and examples.
```

Con la siguiente función conseguimos el contenido de los artículos (Aun no tienen las cabeceras)

```
articulos = unlist(get_content())
substr(head(articulos),1,80)
```

```
## [1] "1. No será castigada ninguna acción ni omisión que no esté prevista como delito "
## [2] "1. No será castigado ningún delito con pena que no se halle prevista por ley ant"
## [3] "1. No podrá ejecutarse pena ni medida de seguridad sino en virtud de sentencia f"
## [4] "1. Las leyes penales no se aplicarán a casos distintos de los comprendidos expre"
## [5] "No hay pena sin dolo o imprudencia."
## [6] "1. Las medidas de seguridad se fundamentan en la peligrosidad criminal del sujet"
```

Despues usamos nuestro separador, que devuelve los nombres, docvars y cabeceras

```
utils = separator()
corpus_names = utils$names
corpus_starts = utils$starts
corpus_docvars = utils$docvars
head(corpus_names)
```

```
## [1] "TÍTULO PRELIMINAR.Artículo 1." "TÍTULO PRELIMINAR.Artículo 2."
## [3] "TÍTULO PRELIMINAR.Artículo 3." "TÍTULO PRELIMINAR.Artículo 4."
## [5] "TÍTULO PRELIMINAR.Artículo 5." "TÍTULO PRELIMINAR.Artículo 6."
```

```
head(corpus_starts)
```

```
## [1] "TÍTULO PRELIMINAR: De las garantías penales y de la aplicación de la Ley penal\nArtículo 1"
## [2] "TÍTULO PRELIMINAR: De las garantías penales y de la aplicación de la Ley penal\nArtículo 2"
## [3] "TÍTULO PRELIMINAR: De las garantías penales y de la aplicación de la Ley penal\nArtículo 3"
## [4] "TÍTULO PRELIMINAR: De las garantías penales y de la aplicación de la Ley penal\nArtículo 4"
## [5] "TÍTULO PRELIMINAR: De las garantías penales y de la aplicación de la Ley penal\nArtículo 5"
## [6] "TÍTULO PRELIMINAR: De las garantías penales y de la aplicación de la Ley penal\nArtículo 6"
```

```
head(corpus_docvars)
```

```
##            Libro                                                    Título
## Artículo 1  <NA> De las garantías penales y de la aplicación de la Ley penal
## Artículo 2  <NA> De las garantías penales y de la aplicación de la Ley penal
## Artículo 3  <NA> De las garantías penales y de la aplicación de la Ley penal
## Artículo 4  <NA> De las garantías penales y de la aplicación de la Ley penal
## Artículo 5  <NA> De las garantías penales y de la aplicación de la Ley penal
## Artículo 6  <NA> De las garantías penales y de la aplicación de la Ley penal
##            Capítulo Sección   Artículo
## Artículo 1     <NA>    <NA> Artículo 1
## Artículo 2     <NA>    <NA> Artículo 2
## Artículo 3     <NA>    <NA> Artículo 3
## Artículo 4     <NA>    <NA> Artículo 4
## Artículo 5     <NA>    <NA> Artículo 5
## Artículo 6     <NA>    <NA> Artículo 6
```

Nótese que en los docvars habrá NA's pues no todos los artículos tiene Libro/Título/Capítulo/Sección. Por ejemplo los primeros no están incluídos dentro de ningún libro. Después unimos las cabeceras con el resto del contenido de los artículos.

```
for ( i in 1:length(articulos)){
  articulos[[i]] = paste(corpus_starts[[i]],articulos[[i]],sep = "\n")
}
```

Y ya tenemos todo

```
corpus = corpus(articulos)
docvars(corpus) = corpus_docvars
names(corpus) = corpus_names
head(corpus)
```

```
## Corpus consisting of 6 documents and 5 docvars.
## TÍTULO PRELIMINAR.Artículo 1. :
## "TÍTULO PRELIMINAR: De las garantías penales y de la aplicaci..."
##
## TÍTULO PRELIMINAR.Artículo 2. :
## "TÍTULO PRELIMINAR: De las garantías penales y de la aplicaci..."
```

```
##
## TÍTULO PRELIMINAR.Artículo 3. :
## "TÍTULO PRELIMINAR: De las garantías penales y de la aplicaci..."
##
## TÍTULO PRELIMINAR.Artículo 4. :
## "TÍTULO PRELIMINAR: De las garantías penales y de la aplicaci..."
##
## TÍTULO PRELIMINAR.Artículo 5. :
## "TÍTULO PRELIMINAR: De las garantías penales y de la aplicaci..."
##
## TÍTULO PRELIMINAR.Artículo 6. :
## "TÍTULO PRELIMINAR: De las garantías penales y de la aplicaci..."
```

```
cat(corpus[[1]])
```

```
## TÍTULO PRELIMINAR: De las garantías penales y de la aplicación de la Ley penal
## Artículo 1
## 1. No será castigada ninguna acción ni omisión que no esté prevista como delito por ley anterior a su
## 2. Las medidas de seguridad sólo podrán aplicarse cuando concurran los presupuestos establecidos prev
```

Parte 2: Distancias

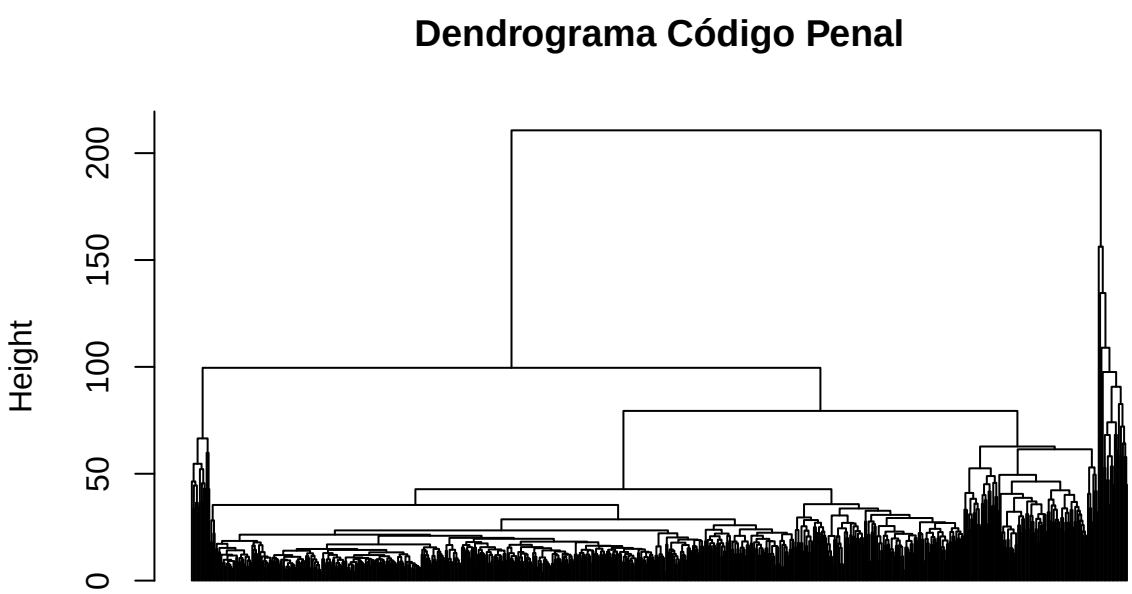
Primero creamos la matriz de distancias entre los textos.

```
dtm = dfm(tokens(corpus))
m <- as.matrix(dtm)
distMatrix <- dist(m, method = "euclidean")
```

Después agrupamos los textos usando las distancias para hacer el dendrograma

```
groups <- hclust(distMatrix)

plot(groups, cex = 0.9, hang = -1,
     labels=FALSE,
     main = "Dendrograma Código Penal",
     xlab = "Artículos",
) #labels = FALSE para que no aparezcan los nombres, sino no se aprecia nada
```



Por la cantidad de datos en el corpus es casi imposible apreciar nada, mucho menos sin tener los nombres. Sin embargo, si mostramos los nombres la pantalla se tapa con ellos al completo.

Para buscar los dos artículos más parecidos buscamos la menor distancia. Primero tenemos que transformar distMatrix en una matriz, pues de normal se devuelve en forma de tipo "dist", que contiene, entre otras cosas, el vector de distancias. Sin embargo, es más cómodo en forma de matriz. Despues cambiamos la diagonal de esta matriz a infinito, pues la diagonal representa la distancia de cada texto a sí mismo y este será 0 siempre.

```
distMatrix_full <- as.matrix(distMatrix)

diag(distMatrix_full) <- Inf

min_index <- which(distMatrix_full == min(distMatrix_full), arr.ind = TRUE)
i <- min_index[1, 1]
j <- min_index[1, 2]
cat(i,j)
```

```
## 609 608
```

Así, los dos textos más parecidos son:

```
text_1 <- strwrap(corpus[[i]], width = 80)
text_2 <- strwrap(corpus[[2]], width = 80)
cat(paste(text_1, collapse = "\n"))
```

```
## LIBRO II: Delitos y sus penas Título XXI: Delitos contra la Constitución
## Capítulo V: De los delitos cometidos por los funcionarios públicos contra las
## garantías constitucionales Sección I: De los delitos cometidos por los
## funcionarios públicos contra la libertad individual Artículo 531 La autoridad o
## funcionario público que, mediando causa por delito, decretare, practicare o
## prolongare la incomunicación de un detenido, preso o sentenciado, con violación
## de los plazos o demás garantías constitucionales o legales, será castigado con
## la pena de inhabilitación especial para empleo o cargo público por tiempo de
## dos a seis años.
```

```
cat(paste(text_2, collapse = "\n"))
```

```
## TÍTULO PRELIMINAR: De las garantías penales y de la aplicación de la Ley penal
## Artículo 2 1. No será castigado ningún delito con pena que no se halle prevista
## por ley anterior a su perpetración. Carecerán, igualmente, de efecto
## retroactivo las leyes que establezcan medidas de seguridad. 2. No obstante,
## tendrán efecto retroactivo aquellas leyes penales que favorezcan al reo, aunque
## al entrar en vigor hubiera recaído sentencia firme y el sujeto estuviese
## cumpliendo condena. En caso de duda sobre la determinación de la Ley más
## favorable, será oído el reo. Los hechos cometidos bajo la vigencia de una Ley
## temporal serán juzgados, sin embargo, conforme a ella, salvo que se disponga
## expresamente lo contrario.
```

Nótese que se ha hecho un wrap pues sino los textos se salen del pdf, por lo que no se estan consevando los saltos de línea.

Parte 3: Entidades nombradas

Importamos primero las librerias necesarias y cargamos los modelos

```
library(spacyr)
library(udpipe)

spacy_initialize(model = "es_core_news_sm")
```

```
## successfully initialized (spaCy Version: 3.7.6, language model: es_core_news_sm)
```

```
#ud_model <- udpipe_download_model(language = "spanish-ancora")
udmodel_es <- udpipe_load_model(file = 'spanish-ancora-ud-2.5-191206.udpipe')
```

Buscamos la entidades nombradas con scapyr:

```
ent = spacy_extract_entity(corpus)
head(ent)
```

```
##                          doc_id                              text ent_type
## 1 TÍTULO PRELIMINAR.Artículo 1.                TÍTULO PRELIMINAR      ORG
## 2 TÍTULO PRELIMINAR.Artículo 1.                           la Ley      LOC
## 3 TÍTULO PRELIMINAR.Artículo 1.                      Artículo 1\n1     MISC
## 4 TÍTULO PRELIMINAR.Artículo 1. No será castigada ninguna acción     MISC
```

```
## 5 TÍTULO PRELIMINAR.Artículo 1.         Las medidas de seguridad     MISC
## 6 TÍTULO PRELIMINAR.Artículo 1.                             la Ley      LOC
##   start_id length
## 1        1      2
## 2       13      2
## 3       17      4
## 4       22      5
## 5       45      4
## 6       59      2
```

Ahora veamos las más comunes

```
freqs = table(ent$text)
comunes = freqs[order(-freqs)][1:20]
comunes
```

```
##
##             Delitos              Código          Capítulo I        Capítulo III
##                 557                 106                  97                  85
##         Título XIII           Sección I          Título III      Administración
##                  85                  83                  79                  74
##         Capítulo II            Tribunal                Juez          Sección II
##                  73                  65                  63                  57
##         Título XVII                año                Será         Sección III
##                  55                  53                  53                  50
## Título XIX: Delitos              España              Estado            Capítulo
##                  48                  46                  41                  39
```

Podemos ver que las palabras que han salido casi todas pertenecen a las cabeceras de los artículos, como era de esperar. Además de otras palabras comunes en textos legales, como delito, código...

Ahora hagámoslo con Udpipe: Primero anotamos

```
annotations <- udpipe_annotate(udmodel_es, x = corpus)
annotations_df <- as.data.frame(annotations)
head(annotations_df[,5:10])
```

```
##   token_id      token      lemma  upos  xpos
## 1        1     TÍTULO     título  VERB  VERB
## 2        2 PRELIMINAR preliminar  VERB   AUX
## 3        3          :          : PUNCT PUNCT
## 4        4         De         de   ADP   ADP
## 5        5        las         el   DET   DET
## 6        6  garantías   garantía  NOUN  NOUN
##                                                 feats
## 1 Mood=Ind|Number=Sing|Person=1|Tense=Pres|VerbForm=Fin
## 2                                          VerbForm=Inf
## 3                                        PunctType=Colo
## 4                                          AdpType=Prep
## 5      Definite=Def|Gender=Fem|Number=Plur|PronType=Art
## 6                                Gender=Fem|Number=Plur
```

Y usamos keywords_rake

```r
keywords <- keywords_rake(
  annotations_df,
  term = "lemma",         #Usamos la columna "lemma" de la anotaciones
  group = "doc_id"        #Agrupaciones segun doc_id
)
head(keywords)
```

```
## [1] keyword ngram   freq    rake
## <0 rows> (or 0-length row.names)
```

Parece que no esta detectando nada. Esto puede ocurrir por frecuencia de elementos como puntuación

```r
keywords <- keywords_rake(
  annotations_df,
  term = "lemma",         #Usamos la columna "lemma" de la anotaciones
  group = "doc_id",        #Agrupaciones segun doc_id
  relevant = (annotations_df$xpos != "PUNCT")  #Nos permite seleccionar cuales filas son relevantes
  )

keywords = keywords[order(-keywords$freq),]   #Ordenamos nuestro dataframe en
#orden descendente según la frecuencia
head(keywords)
```

```
##      keyword ngram freq      rake
## 148 Libro II     2  547  6.486562
## 170        2     1  286  2.609610
## 171        3     1  156  2.545977
## 28   el pena     2  141 31.211216
## 67         a     1  113 17.720250
## 166        b     1  104  3.288462
```

Vemos que muchas letras sueltas y números que no nos interesan demasiado. Veamos como arreglar esto

```r
table(annotations_df$xpos)
```

```
##
##   ADJ   ADP   ADV   AUX CCONJ   DET  INTJ  NOUN   NUM  PART  PRON PROPN PUNCT
## 11232 24181  2024  2519  7502 18768     1 30916  5315    34  5307  8389 12778
## SCONJ   SYM  VERB
##  1857   164  7163
```

Algunos de estos no nos interesan: Por ejemplo:

```r
head(annotations_df[which(annotations_df$upos=="SYM"),6])
```

```
## [1] "b" "b" "b" "b" "c" "b"
```

```r
head(annotations_df[which(annotations_df$upos=="NUM"),6])
```

```
## [1] "1" "1" "2" "2" "1" "2"
```

Entonces los quitaremos con el parametro relevant de keywords_rake (permite decir que filas no son relevantes). También podemos modificar ngram_max, que cambia la máxima cantidad de palabras que puede tener en cuenta como una entidad (por defecto es solo 2)

```
keywords <- keywords_rake(
  annotations_df,
  term = "lemma",
  group = "doc_id",
  relevant = ( ! annotations_df$xpos  %in% c("PUNCT","SYM","NUM")),
  ngram_max = 5,
)

keywords = keywords[order(-keywords$freq),]  #Ordenamos for frecuencia

head(keywords,20)
```

```
##                               keyword ngram freq      rake
## 459                                 a     1  792 10.584209
## 495                          Libro II     2  547  6.290509
## 516                               año     1  530  3.569277
## 518                               mes     1  200  3.059308
## 423                            mes a      2  182 13.643517
## 270            el persona responsable     3  141 36.164505
## 362                          el pena      2  141 23.627220
## 358                      del artículo     2   93 24.273343
## 438                                 y     1   89 12.276257
## 322                            el que     2   75 29.347449
## 499                               bis     1   68  5.737805
## 268                    año y multa de     4   62 36.471832
## 385                         multa de     2   59 20.626297
## 292                      en todo caso     3   46 33.348942
## 346                      del apartado     2   46 25.865912
## 296                        en su caso     3   41 32.943241
## 522                              euro     1   30  2.652174
## 535                                 d     1   30  0.062500
## 29  de el pena capítulo I: de el pena     5   29 66.608278
## 451                              a g      2   29 11.393733
```

Siguen saliendo muchas palabras que no nos interesan “a”, “y”, “el que”. Para ello quitaremos otras categorías como las conjunciones y los determinantes. Así se concentrará más en los sustantivos y adjetivos

```
keywords <- keywords_rake(
  annotations_df,
  term = "lemma",
  group = "doc_id",
  relevant = ( ! annotations_df$xpos  %in% c("PUNCT","SYM","NUM","CCONJ","ADP")),
  ngram_max = 5,
)

keywords = keywords[order(-keywords$freq),]  #Ordenamos for frecuencia

head(keywords,20)
```

```
##                              keyword ngram freq       rake
## 3259                         delitos     1  924  0.1702786
## 3173                             año     1  801  0.4457831
## 1421                        el pena      2  752  4.9410469
## 3250                             mes     1  563  0.2075783
## 2343                        Libro II     2  559  2.9441630
## 3243                         prisión     1  527  0.2500000
## 1732                       el delito     2  478  4.1162668
## 3248                           multa     1  388  0.2211982
## 1366                    ser castigar     2  322  5.1726825
## 2616                             uno     1  295  2.1303561
## 1693                     el artículo     2  211  4.1910892
## 1791                         el juez     2  193  4.0083625
## 2479        inhabilitación especial     2  184  2.5564196
## 2940                       seguridad     1  171  1.2089041
## 2545                              el     1  158  2.3630922
## 2716                        artículo     1  155  1.8279970
## 3131                          tiempo     1  155  0.6386555
## 2858                          medida     1  151  1.4233129
## 844          el persona responsable     3  144  7.3047104
## 461  Libro i: disposiciones general     4  141 10.2483140
```

Ahora parece representar algo mejor los textos. Podemos ver que coincide en algunas con las calculadas con spacyr, "delitos", "año", "juez".

Otros:

Se han creado varias funciones auxiliares para dividir en bloques la tarea de hacer scraping y formatear los datos, de esta manera es más fácil de buscar errores y organizar el código. El resto de los archivos son:

- **Schema.txt**: contiene un esquema de las cabeceras de los artículos. Esto fue obtenido de antemano usando XPATH
- **generador.R**:
  - get_html_arts: devuelve los links de los artículos que nos interesan
  - writeAll: escribe en la carpeta artículos el contenido (en html) de los artículos
- **processor.R**:
  - get_articles: devuelve los nombres de los artículos que nos interesan en el formato deseado (Articulo 10)
  - get_content: recibe el contenido html de un artículo y devuelve el contenido real del artículo del código penal
  - get_titles: itera sobre schema.txt y crea un dataframe que incluye cada artículo con sus cabeceras (Libro, Título, Capítulo, Sección)
  - separator: utiliza el dataframe dado por get_titles y lo separa en 3 elementos: 1. un dataframe `<<docvars>>` que contiene el mismo dataframe anterior pero quitando el inicio de las cabeceras, es decir, pasa de `<<LIBRO I: de las disposiciones...>>` a `<<de las disposiciones...>>`, 2. un vector `<<names>>`, que contiene el nombre de cada artículo con el fomato `<<LIBRO I.Título II.Capítulo I.Artículo III.>>` (Por ejemplo) y 3. un vector `<<starts>>` que contiene los inicios de los artículos, que es solamente la concatenación, con `"\n"` como separador, de las cabeceras al completo de cada artículo

En un principio el código se ejecuta en local, pues processor.R lee de la carpeta artículos, las funciones del archivo generador no estan activas en el Rmd. Puede reescribir los artículos descomentando la primera función al principio #writeAll(). Lo mismo con el modelo de udpipe, de da ya descargado per se puede reinstalar descomentando la línea 134.