Prueba Final - R aplicado al análisis cualitativo

Victoria Nuñez

30/9/2021

Introducción

El presente **documento** es realizado en el marco el curso de *R aplicado al análisis cualitativo* de la *Facultad de Ciencias Sociales*. Contiene los resultados del procesamiento de las últimas dos leyes de presupuesto nacional, tanto de la **Ley 19924** del **Presupuesto Nacional para el quinquenio 2020-2024** ubicada en la web https://www.impo.com.uy/bases/leyes/19924-2020, como la anterior **Ley 19355** del **Presupuesto Nacional para el quinquenio 2015-2019** ubicada en https://www.impo.com.uy/bases/leyes/19355-2015 y un análisis comparativo entre ambas.

En primer lugar se realiza un **Web scraping** de las páginas mencionadas, para acceder al conteido de ambas leyes.

Luego del proceso de extracción y de reconstrucción de la base de datos, se crea un **corpus** de datos, con su correspondiente limpieza.

Una vez obtenido el *corpus*, se realizan técnicas de **minería de texto**, como ser *frecuencia de ocurrencia*, *asociación de palabras*, *agrupación*, *diccionarios* y *análisis de sentimiento*.

Por útlimo se *ilustran los datos* y se presentan algunas reflexiones.

Web scraping

Cargo las librerías necesarias y las páginas web de referencia:

```
library(rvest)
library(dplyr)

ley_ppto15 = read_html("https://www.impo.com.uy/bases/leyes/19355-2015")
ley_ppto20 = read_html("https://www.impo.com.uy/bases/leyes/19924-2020")
```

Se utiliza el paquete *rvest* para realizar el scraping y análisis web, desarrollado por Hadley Wickham (*Documentación rvest:* https://cran.r-project.org/web/packages/rvest/rvest.pdf).

Luego, con la extensión *SelectorGadget de Chrome* se busca el nombre del nodo en la pagina web correspondiente para descargar el contenido de la **Ley de presupuesto** y convertirla en un *dataframe*.

```
## 2015
contenido_ley15 = ley_ppto15 %>%
  html_nodes("pre") %>%        # SelectorGadget
  html_text()
```

```
contenido_ley15_df<-as.data.frame(contenido_ley15)

## 2020
contenido_ley20 = ley_ppto20 %>%
  html_nodes("pre") %>%        # SelectorGadget
  html_text()

contenido_ley20_df<-as.data.frame(contenido_ley20)
```

Análisis de texto

Cargo las librerías necesarias para este paso:

```
library(quanteda)
library(readtext)
library(stringr)
library(dplyr)
library(ggplot2)
library(quanteda.textstats)
```

Se utiliza el paquete *quanteda* para análisis de texto, que administra y analiza datos textuales, desarrollado por Kenneth Benoit y otros colaboradores (*Documentación quanteda:* https://cran.r-project.org/web/packages/quanteda/quanteda.pdf).

Creo un corpus con el paquete quanteda para cada año:

```
## 2015
myCorpus15 <- corpus(contenido_ley15_df$contenido_ley15)

## 2020
myCorpus20 <- corpus(contenido_ley20_df$contenido_ley20)
```

A continuación se aplican distintas técnias de limpieza de texto quitando datos no relevantes. Para esto, se carga el archivo de *stopwords* propios y modismos vistos en el curso, y luego con un *Document feature matrix (DFM)*, se aplican algunos argumentos que permiten limpiar las palabras que no interesan a efectos del análisis, como son palabras de menos de 4 letras y algunas otras palabras especificas del formato de leyes y decretos que no aportan al análisis, así como limpieza de numeros y puntuaciones o palabras que se repiten en ambas.

```
stop = read.csv("stopes.csv", sep = ";")
vector = as.character(stop$X0)
vector_ley = c("Notas","artículo","redacción","decreto","ley","inciso",
               "norma","vigencia","ejecutora","unidad","ministerio",
               "present","nacion","nacional","presente","uruguayos","pesos",
               "dirección","millones","mil","programa","artículos", "cargo",
               "ciento", "diciembre")
```

```
## 2015
mydfm15 <- dfm(tokens(contenido_ley15_df$contenido_ley15,
                  remove_punct = TRUE,
                  remove_numbers = TRUE),
```

```
                  tolower=TRUE,
                  verbose = TRUE) %>%
quanteda::dfm_remove(pattern = c(quanteda::stopwords("spanish"),
                                 vector,vector_ley),min_nchar = 4)

mydfm15_2<-as.data.frame(mydfm15)

## 2020
mydfm20 <- dfm(tokens(contenido_ley20_df$contenido_ley20,
                  remove_punct = TRUE,
                  remove_numbers = TRUE),
                  tolower=TRUE,
                  verbose = TRUE) %>%
quanteda::dfm_remove(pattern = c(quanteda::stopwords("spanish"),
                                 vector,vector_ley),min_nchar = 4)

mydfm20_2<-as.data.frame(mydfm20)
```

Nube de Palabras

Para ilustrar estos resultados se realiza una *nube de palabras*, con la función *textplot_wordcloud* del paquete quanteda.

```
library(quanteda.textplots)

## 2015
textplot_wordcloud(mydfm15, min.count = 3,max_words = 150,random.order = TRUE,
                   rot.per = .50, colors = RColorBrewer::brewer.pal(8,"Dark2"))
```


```
## 2020
textplot_wordcloud(mydfm20, min.count = 3,max_words = 150,random.order = TRUE,
                   rot.per = .50, colors = RColorBrewer::brewer.pal(8,"Dark2"))
```

Se observan en ambas nubes, las palabras que más se mencionan en los documentos. En la ley de 2015 podemos observar palabras como: créditos, gasto, salud, financiación. Mientras en la ley de 2020 palabras como: técnico, social, servicios, funcionarios.

Palabras más frecuentes

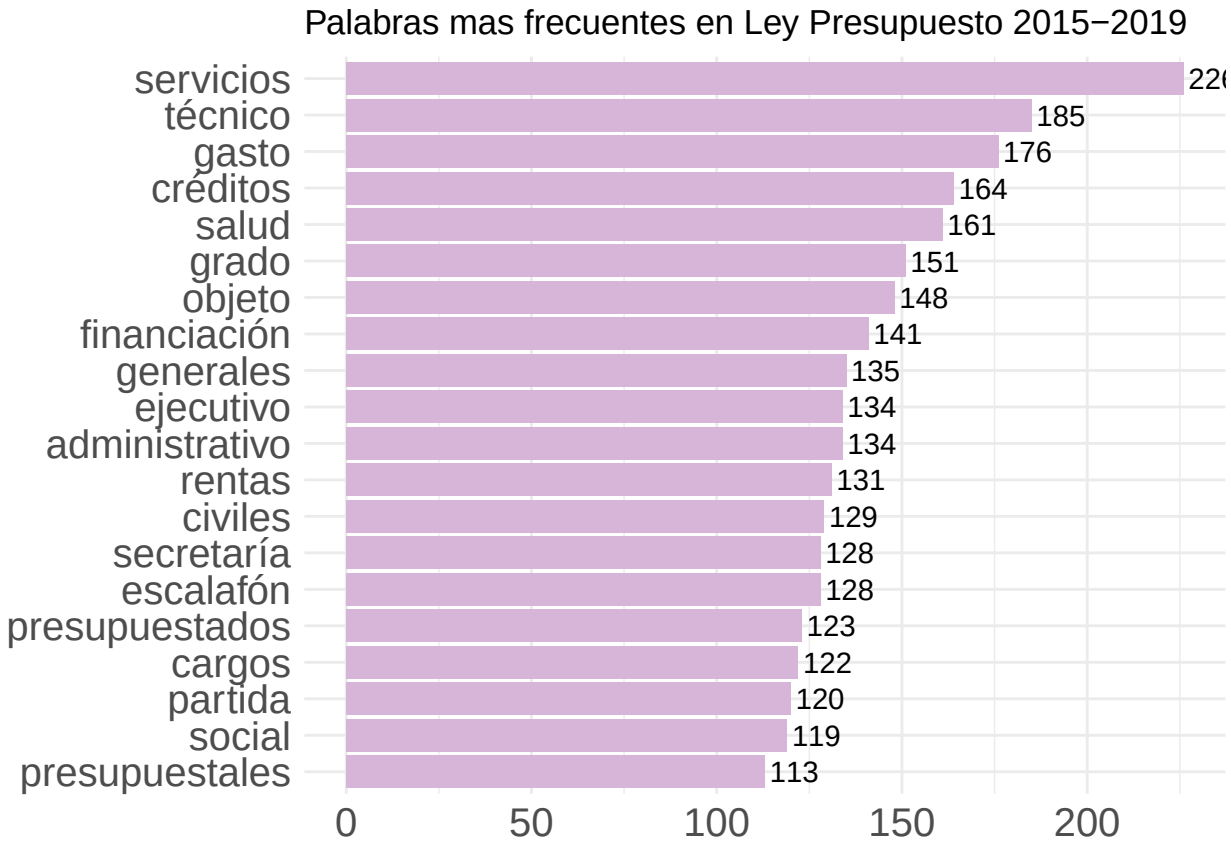
Podemos observar cuáles son las *palabras más frecuentes* con la función *topfeatures*.

```
color = c(rep("#D7B5D8",20))

## 2015
top_ley15 = data.frame(topfeatures(mydfm15,20))
top_ley15$palabra = rownames(top_ley15)

## 2020
top_ley20 = data.frame(topfeatures(mydfm20,20))
top_ley20$palabra = rownames(top_ley20)
```

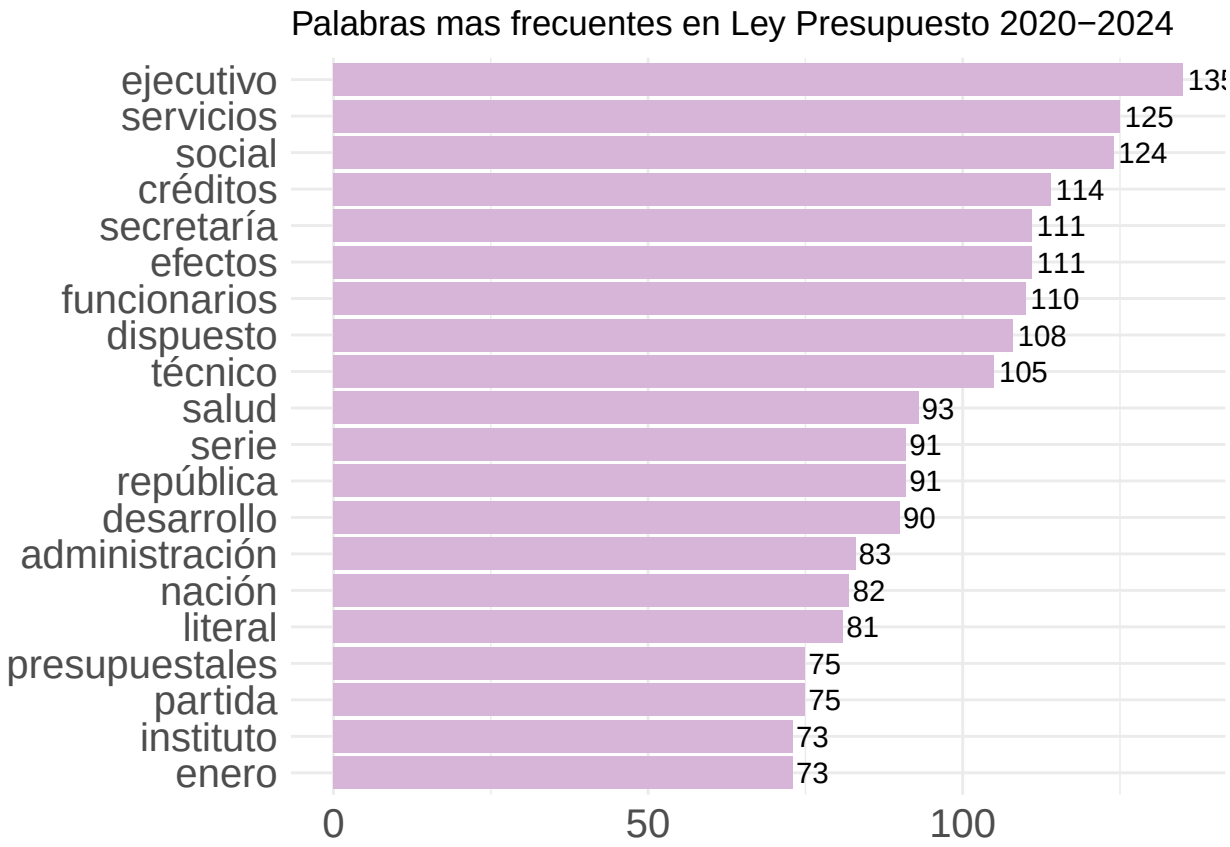
Y luego, graficamos los resultados para cada año de la siguiente forma:

```
## 2015
ley_pplot15 = top_ley15[1:20, ] %>%
  ggplot(aes(x = reorder(palabra, topfeatures.mydfm15..20.),
             y = topfeatures.mydfm15..20., fill = palabra)) +
  geom_col(show.legend = FALSE) +
```

```
  coord_flip() +
  geom_text(aes(hjust = -0.1, label = topfeatures.mydfm15..20.)) +
  theme_minimal() +
  theme(axis.title.y = element_blank(), axis.title.x = element_blank(), axis.text = element_text(size =
  ggtitle("Palabras mas frecuentes en Ley Presupuesto 2015-2019") +
  scale_fill_manual(values = color)

ley_pplot15
```




```
## 2020
ley_pplot20 = top_ley20[1:20, ] %>%
  ggplot(aes(x = reorder(palabra, topfeatures.mydfm20..20.),
             y = topfeatures.mydfm20..20., fill = palabra)) +
  geom_col(show.legend = FALSE) +
  coord_flip() +
  geom_text(aes(hjust = -0.1, label = topfeatures.mydfm20..20.)) +
  theme_minimal() +
  theme(axis.title.y = element_blank(), axis.title.x = element_blank(), axis.text = element_text(size =
  ggtitle("Palabras mas frecuentes en Ley Presupuesto 2020-2024") +
  scale_fill_manual(values = color)

ley_pplot20
```



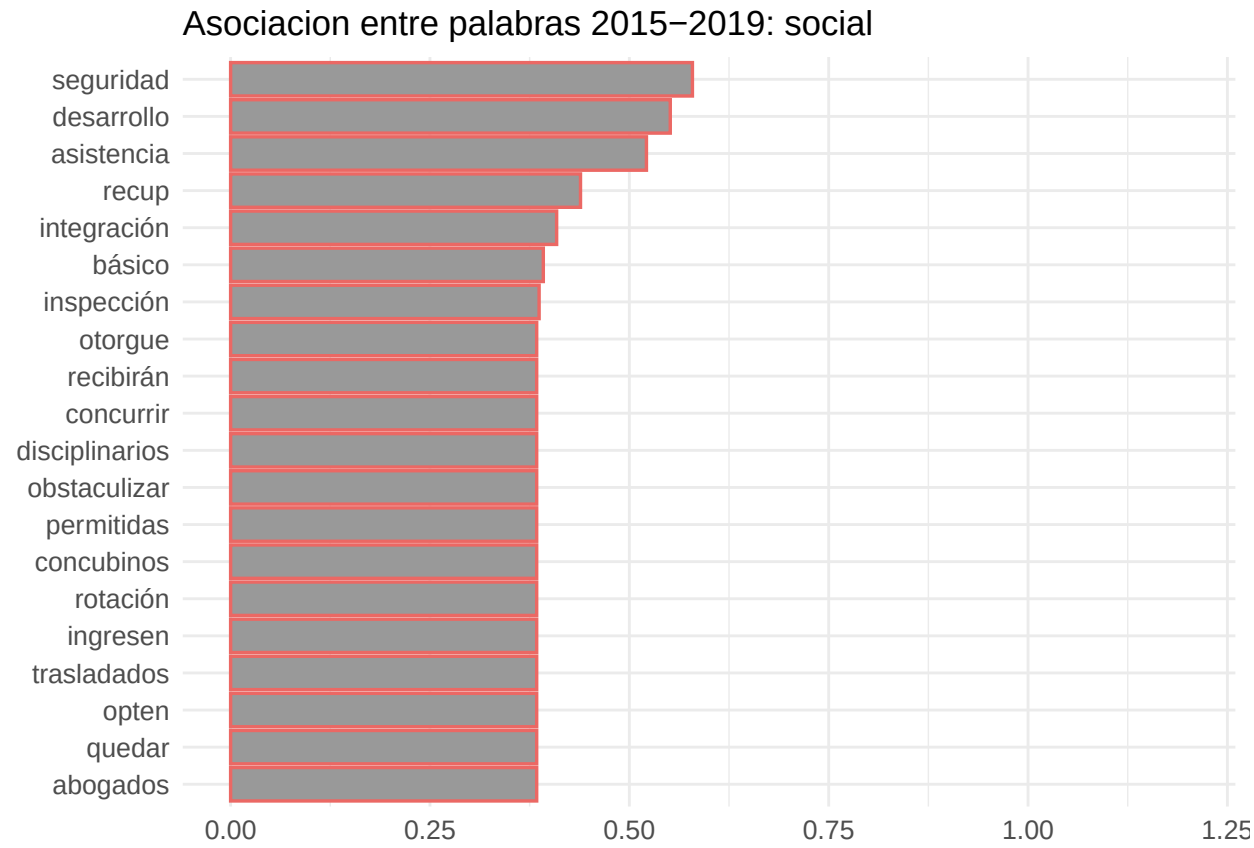
Nuevamente vemos las palabras que más se repiten en ambas leyes ordenadas decrecientemente desde las más pronunciadas.

Asociación de palabras

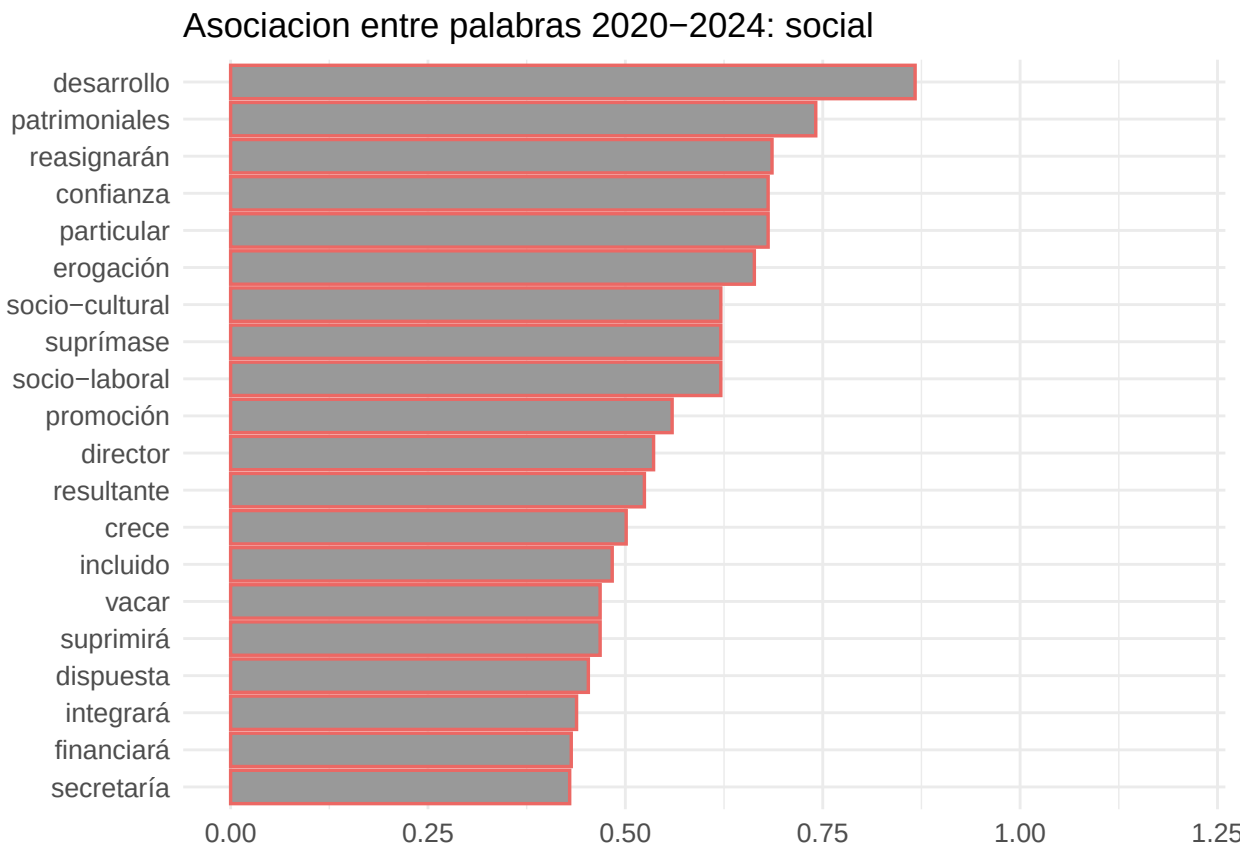
También podemos realizar una *Asociación de palabras*, que nos permite observar para las palabras más relevantes, con cuales otras palabras se asocian en el texto frecuentemente. Lo hago en primer lugar con la palabra **"social"**:

```
## 2015
asociacion15 = mydfm15  %>%
  quanteda.textstats::textstat_simil(selection = "social",
                                     margin = "features", method = "correlation")%>%
  as.data.frame()%>%
  arrange(-correlation)%>%
  top_n(20)%>%
  rename(valor = correlation,palabra=feature1) %>%
  select(palabra,valor)%>%
  ggplot2::ggplot(aes(x = reorder(palabra,valor),
                      y = valor)) +
  geom_col(show.legend = FALSE,colour='#eb6864',size=0.6,fill="#999999") +
  coord_flip() +
  theme_minimal() +
  scale_y_continuous(limits = c(0, 1.2))+
  theme(axis.title.y = element_blank(), axis.title.x = element_blank(), axis.text = element_text(size =
  ggtitle(paste("Asociacion entre palabras 2015-2019:","social"))
```

```
asociacion15
```



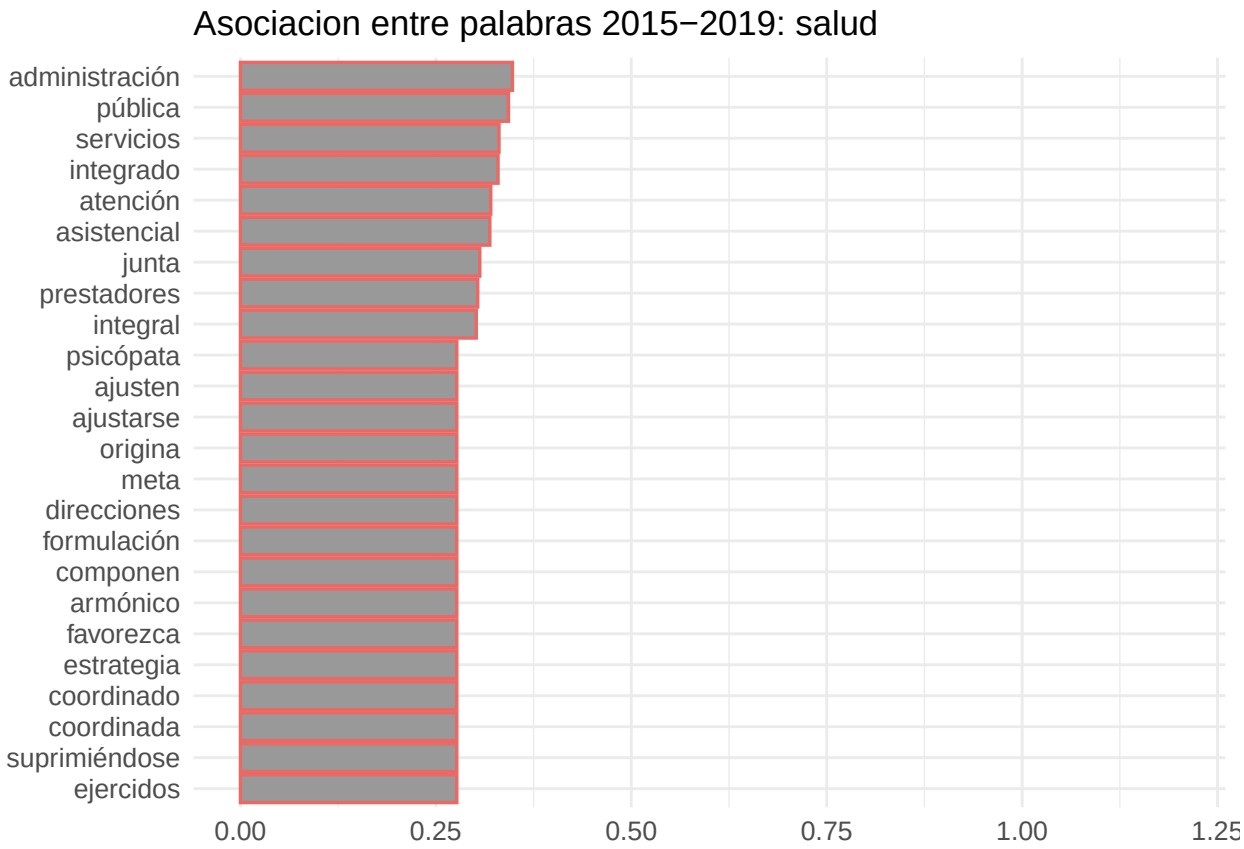

```
## 2020
asociacion20 = mydfm20  %>%
  quanteda.textstats::textstat_simil(selection = "social",
                                     margin = "features", method = "correlation")%>%
  as.data.frame()%>%
  arrange(-correlation)%>%
  top_n(20)%>%
  rename(valor = correlation,palabra=feature1) %>%
  select(palabra,valor)%>%
  ggplot2::ggplot(aes(x = reorder(palabra,valor),
                      y = valor)) +
  geom_col(show.legend = FALSE,colour='#eb6864',size=0.6,fill="#999999") +
  coord_flip() +
  theme_minimal() +
  scale_y_continuous(limits = c(0, 1.2))+
  theme(axis.title.y = element_blank(), axis.title.x = element_blank(), axis.text = element_text(size =
  ggtitle(paste("Asociacion entre palabras 2020-2024:","social"))

asociacion20
```



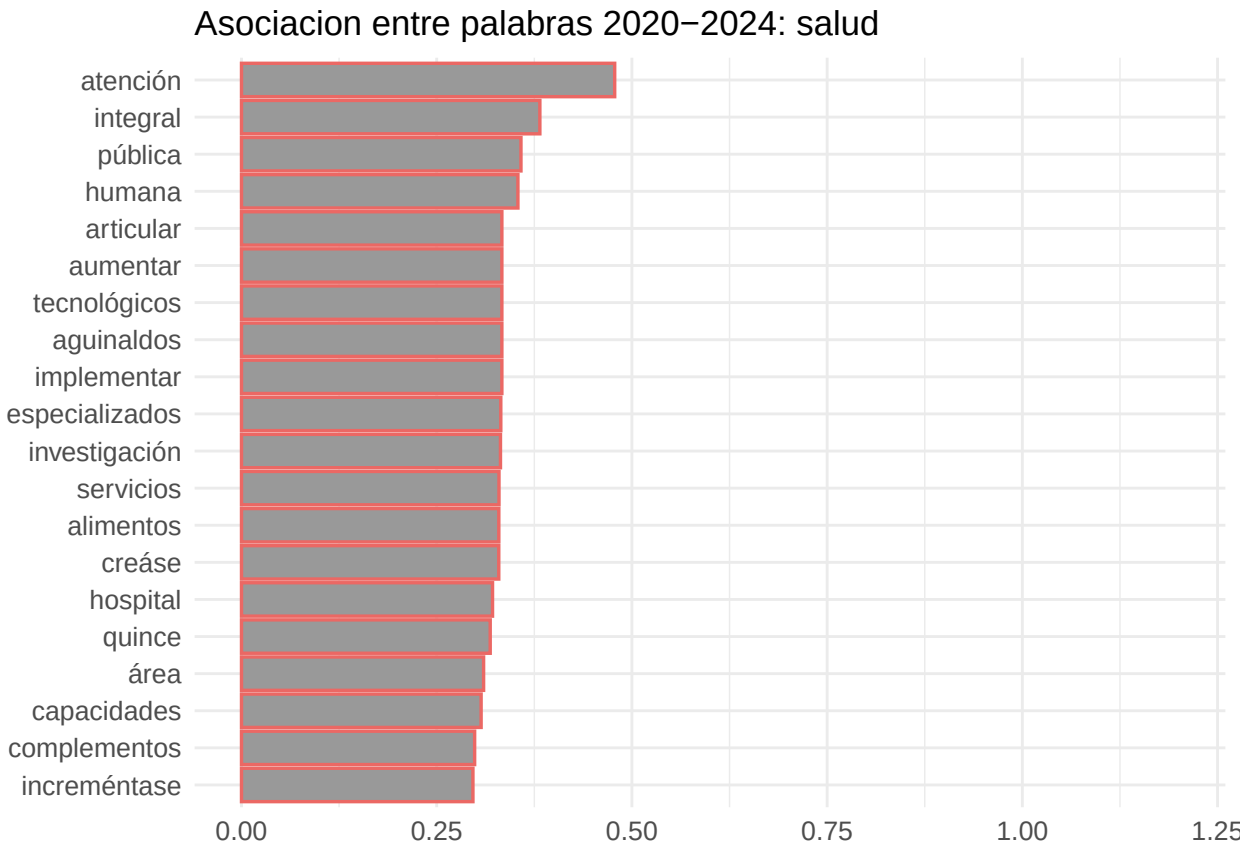
Observamos que en la ley del 2015, con la palabra social se asocian palabras como: seguridad, asistencia, recuperación. Mientras que en la ley del 2020, palabras como: reasignarán, confianza, erogación.

Y luego hacemos la asociación con la palabra **"salud"**:

```
## 2015
asociacion15_2 = mydfm15  %>%
  quanteda.textstats::textstat_simil(selection = "salud",
                                     margin = "features", method = "correlation")%>%
  as.data.frame()%>%
  arrange(-correlation)%>%
  top_n(20)%>%
  rename(valor = correlation,palabra=feature1) %>%
  select(palabra,valor)%>%
  ggplot2::ggplot(aes(x = reorder(palabra,valor),
                      y = valor)) +
  geom_col(show.legend = FALSE,colour='#eb6864',size=0.6,fill="#999999") +
  coord_flip() +
  theme_minimal() +
  scale_y_continuous(limits = c(0, 1.2))+
  theme(axis.title.y = element_blank(), axis.title.x = element_blank(), axis.text = element_text(size =
  ggtitle(paste("Asociacion entre palabras 2015-2019:","salud"))

asociacion15_2
```




```
## 2020
asociacion20_2 = mydfm20  %>%
  quanteda.textstats::textstat_simil(selection = "salud",
                                     margin = "features", method = "correlation")%>%
  as.data.frame()%>%
  arrange(-correlation)%>%
  top_n(20)%>%
  rename(valor = correlation,palabra=feature1) %>%
  select(palabra,valor)%>%
  ggplot2::ggplot(aes(x = reorder(palabra,valor),
                      y = valor)) +
  geom_col(show.legend = FALSE,colour='#eb6864',size=0.6,fill="#999999") +
  coord_flip() +
  theme_minimal() +
  scale_y_continuous(limits = c(0, 1.2))+
  theme(axis.title.y = element_blank(), axis.title.x = element_blank(), axis.text = element_text(size =
  ggtitle(paste("Asociacion entre palabras 2020-2024:","salud"))

asociacion20_2
```



Se observan asociaciones similares a la palabra salud en ambas leyes como: pública, integrado, ateción.

Diccionarios

Utilizamos la función *dictionary* de quanteda donde defino mi *diccionario* con las categorías de interés. Las mismas son: *"social, economía y seguridad"* y dentro de estas grandes categorías incluimos palabras que pertenezcan a las mismas para poder agruparlas.

```
midic <- dictionary(list(social = c("social","politica social",
                                    "politicas sociales", "plan social",
                                    "planes sociales", "sociedad","salud",
                                    "educaci?n","educacion","desarrollo",
                                    "educa*"),
                         economia = c("econom?a","empleo", "desempleo",
                                      "crisis", "economia","fiscal","dolar*",
                                      "ajuste","finan*","credito","crédito",
                                      "créditos","reasigna*","partida",
                                      "econom*","pesos"),
                         seguridad = c("seguridad","robo","reforma",
                                       "delincuente","polic?a","policia",
                                       "delinq*")))
```

Luego se aplica este diccionario a los dataframe que estamos analizando y se obtiene las proporciones de estos grupos de palabras empleadas en el documento.

```
## 2015
midic_result15<-dfm_lookup(mydfm15,dictionary=midic,nomatch="no_aparece")
midic_result15=convert(midic_result15, to = "data.frame")

tabla15 = midic_result15  %>%
  summarise(social = sum(social), social_prop =
              (sum(social)/sum(social,economia,seguridad,no_aparece))*100,
            economia = sum(economia), economia_prop =
              (sum(economia)/sum(social,economia,seguridad,no_aparece))*100,
            seguridad = sum(seguridad), seguridad_prop =
              (sum(seguridad)/sum(social,economia,seguridad,no_aparece))*100)

knitr::kable(tabla15,caption = "Proporción de grupos de palabras 2015-2019")
```

Table 1: Proporción de grupos de palabras 2015-2019

social	social_prop	economia	economia_prop	seguridad	seguridad_prop
602	2.210878	811	2.978442	62	0.2276984

```
## 2020
midic_result20<-dfm_lookup(mydfm20,dictionary=midic,nomatch="no_aparece")
midic_result20=convert(midic_result20, to = "data.frame")

tabla20 = midic_result20  %>%
  summarise(social = sum(social), social_prop =
              (sum(social)/sum(social,economia,seguridad,no_aparece))*100,
            economia = sum(economia), economia_prop =
              (sum(economia)/sum(social,economia,seguridad,no_aparece))*100,
            seguridad = sum(seguridad), seguridad_prop =
              (sum(seguridad)/sum(social,economia,seguridad,no_aparece))*100)

knitr::kable(tabla20,caption = "Proporción de grupos de palabras 2020-2024")
```

Table 2: Proporción de grupos de palabras 2020-2024

social	social_prop	economia	economia_prop	seguridad	seguridad_prop
428	2.065637	586	2.828185	40	0.1930502

Análisis de sentimiento

Estos análisis son métodos de lingüística computacional que sirven para identificar y extraer información subjetiva de un documento. Observamos dos tipos.

1. **Diccionario LWIC-Spanish**: usa la función *dfm_lookup* de quanteda para identificar las emociones presentes en el diccionario y establecer puntajes para cada uno, a partir de la estandarización de los mismos.

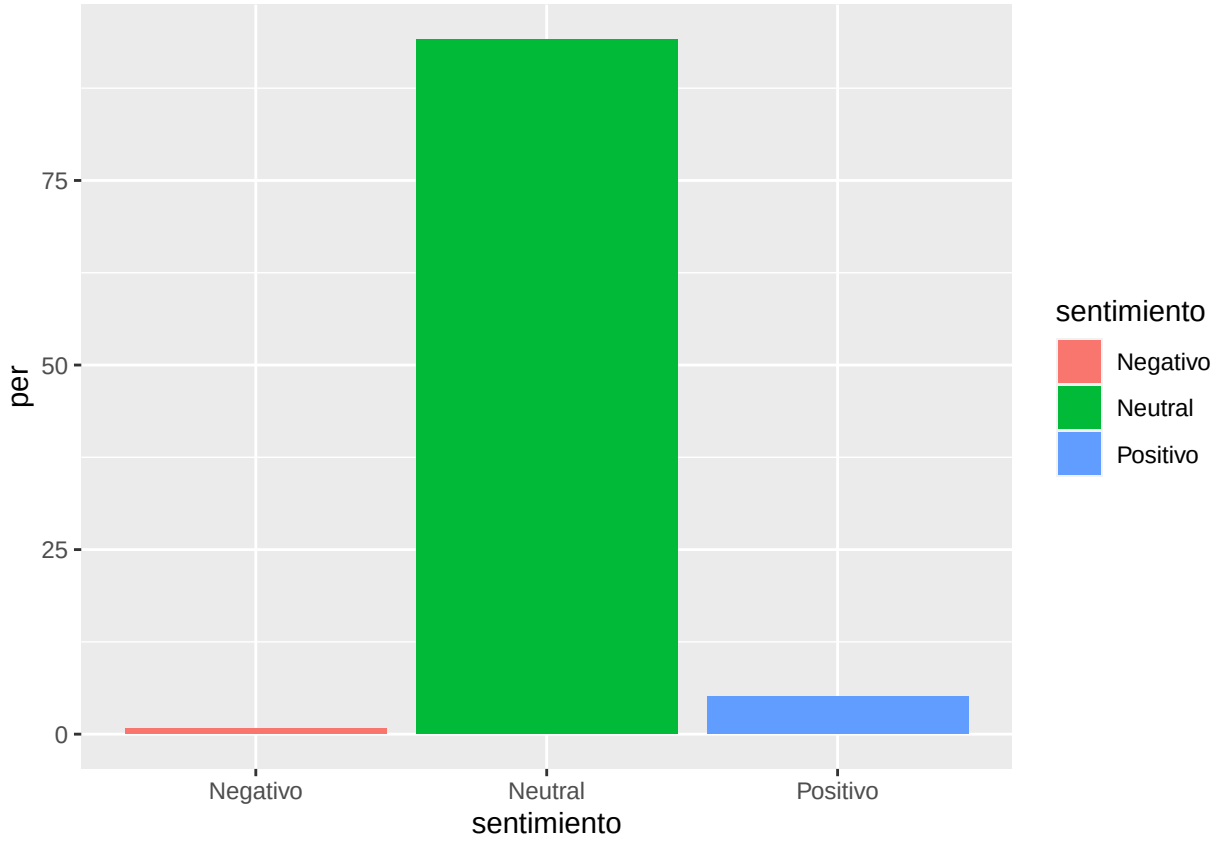
```
lwic <- readRDS("EmoPosNeg_SPA.rds")

## 2015
sent_dfm15<- dfm_lookup(mydfm15, dictionary = lwic)
sent15=convert(sent_dfm15, to = "data.frame")

## 2020
sent_dfm20<- dfm_lookup(mydfm20, dictionary = lwic)
sent20=convert(sent_dfm20, to = "data.frame")
```

Creamos un score con la difrencia entre terminos positivos y negativos, y los vinculamos con las variables de agregación de los documentos.

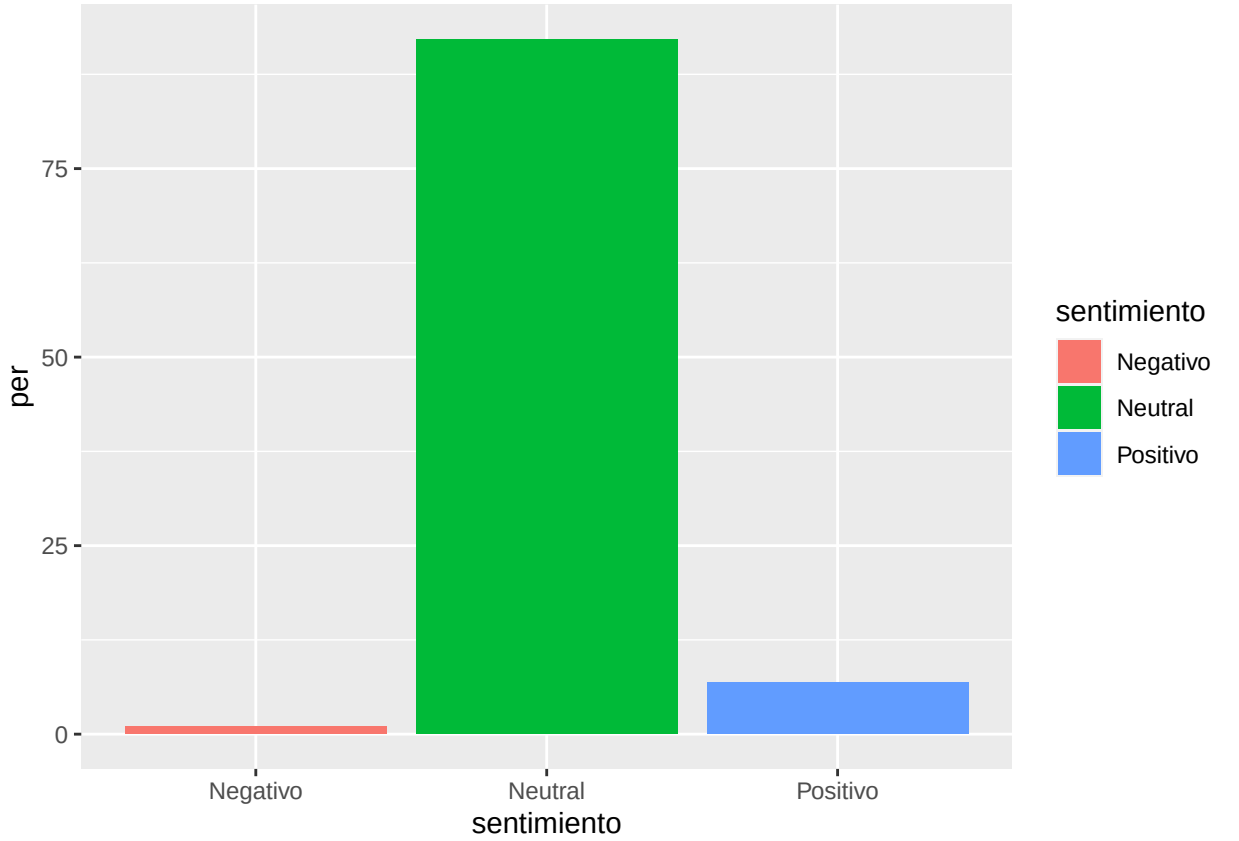
```
## 2015
sent15$puntaje <- sent15$EmoPos-sent15$EmoNeg
sent15$sentimiento=ifelse(sent15$puntaje<0,"Negativo","Positivo")
sent15$sentimiento=ifelse(sent15$puntaje==0,"Neutral",sent15$sentimiento)

## 2020
sent20$puntaje <- sent20$EmoPos-sent20$EmoNeg
sent20$sentimiento=ifelse(sent20$puntaje<0,"Negativo","Positivo")
sent20$sentimiento=ifelse(sent20$puntaje==0,"Neutral",sent20$sentimiento)
```

Y luego graficamos:

```
## 2015
sentimiento15 <- sent15 %>% group_by(sentimiento) %>% summarise(count=n()) %>%
  mutate(per = prop.table(count)*100)

ggplot(sentimiento15, aes(x=sentimiento,y=per, fill=sentimiento))+
  geom_bar(position="dodge", stat="identity")
```




```
## 2020
sentimiento20 <- sent20 %>% group_by(sentimiento) %>% summarise(count=n()) %>%
  mutate(per = prop.table(count)*100)

ggplot(sentimiento20, aes(x=sentimiento,y=per, fill=sentimiento))+
  geom_bar(position="dodge", stat="identity")
```



Tenemos menos de un 5% aproximadamente de palabras asociadas tanto a positivo como a negativo en ambos casos, la gran mayoría se concentran en *Neutral*.

2. **Metodo Syuzhet**: utiliza la función *get_sentiment* del paquete *syuzhet* para asignar puntajes según el método y lenguaje indicado. Es un diccionario de sentimientos desarrollado en el Laboratorio Literario de Nebraska.

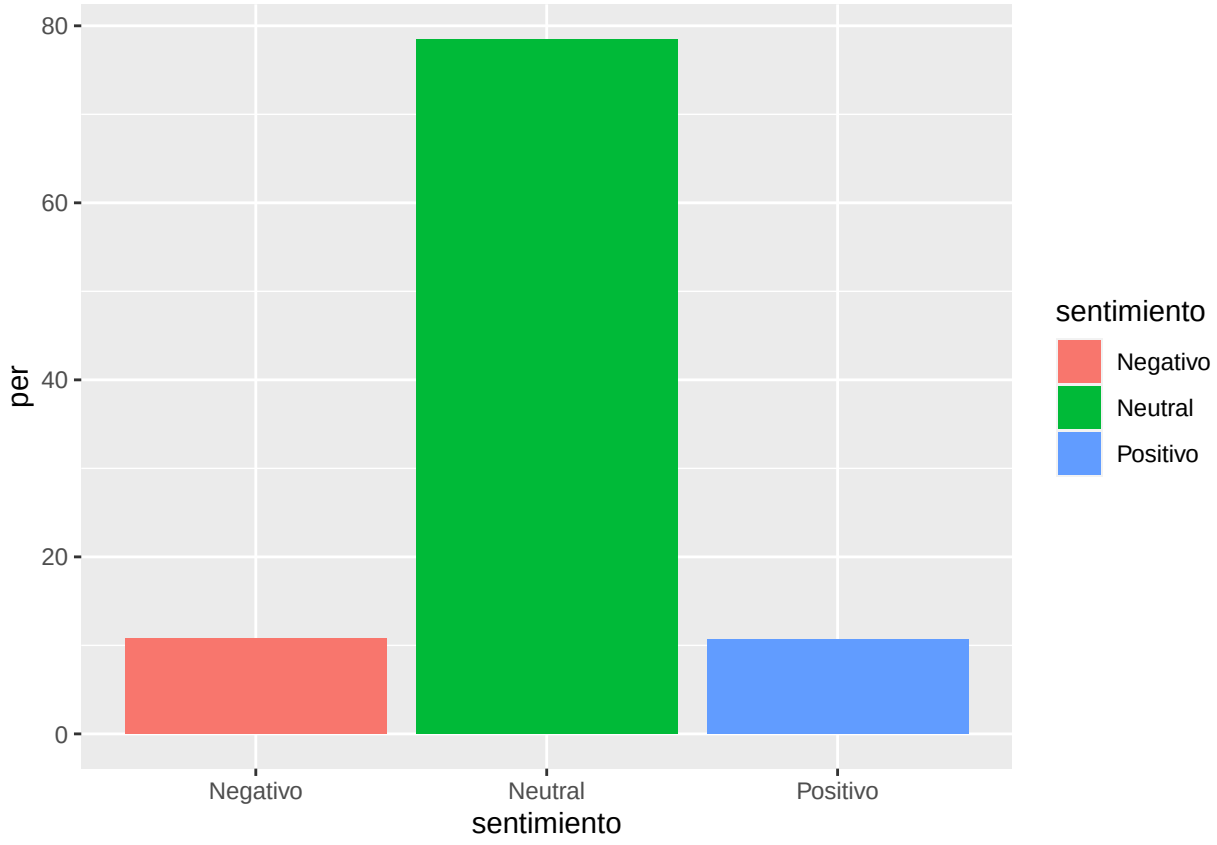
En idioma español se utiliza el diccionario de sentimientos *nrc*, que identifica la presencia en el texto de ocho emociones diferentes con valores asociados y dos sentimientos. Las emociones son ira, miedo, anticipación, confianza, sorpresa, tristeza, alegría. Los sentimientos: positivo y negativo.

```
library(syuzhet)

## 2015
Sentiment15 <- get_nrc_sentiment(contenido_ley15_df$contenido_ley15, language = "spanish")
art_df_senti15 <- cbind(contenido_ley15_df, Sentiment15)

art_df_senti15$puntaje<-art_df_senti15$positive-art_df_senti15$negative
art_df_senti15$sentimiento=ifelse(art_df_senti15$puntaje<0,"Negativo","Positivo")
art_df_senti15$sentimiento=ifelse(art_df_senti15$puntaje==0,"Neutral",art_df_senti15$sentimiento)

art_sent15 <- art_df_senti15 %>% group_by(sentimiento) %>% summarise(count=n()) %>%
  mutate(per = round(prop.table(count)*100,1))

## 2020
Sentiment20 <- get_nrc_sentiment(contenido_ley20_df$contenido_ley20, language = "spanish")
```

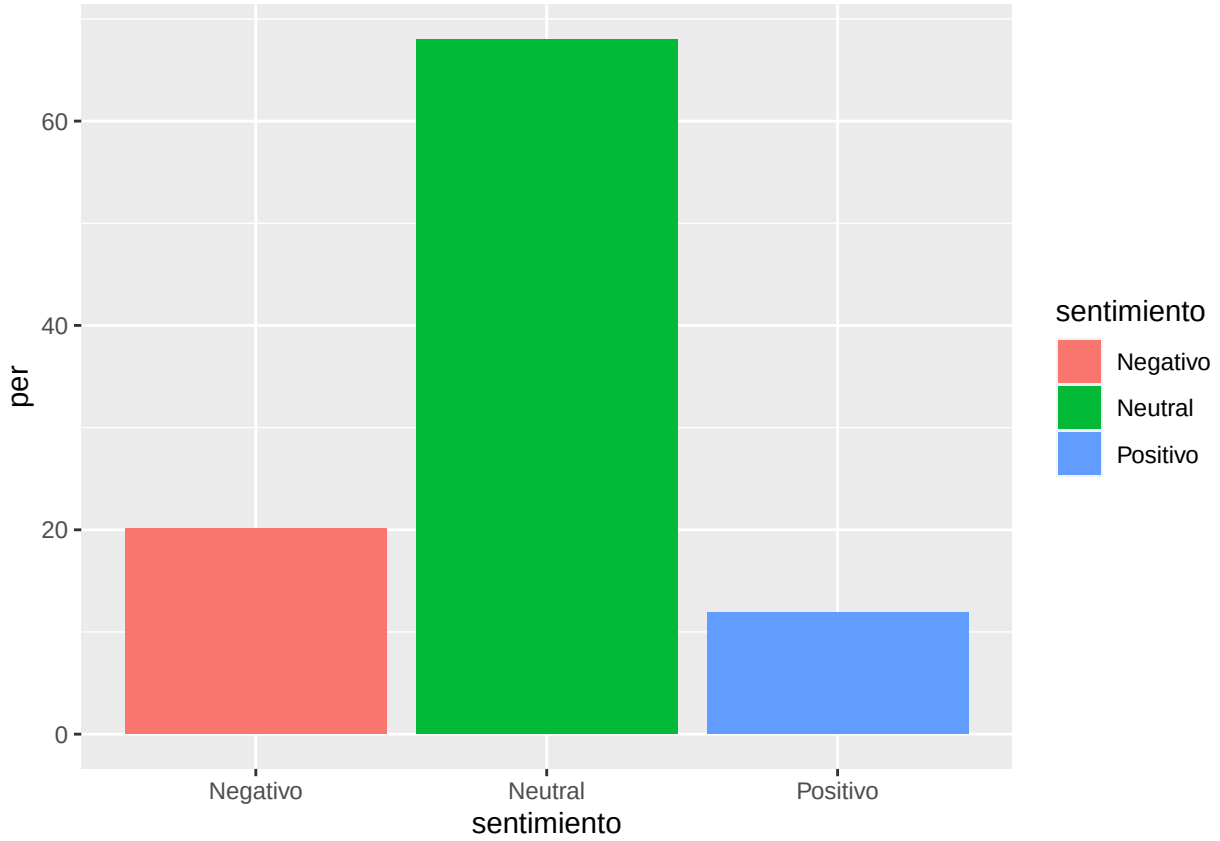
```
art_df_senti20 <- cbind(contenido_ley20_df, Sentiment20)

art_df_senti20$puntaje<-art_df_senti20$positive-art_df_senti20$negative
art_df_senti20$sentimiento=ifelse(art_df_senti20$puntaje<0,"Negativo","Positivo")
art_df_senti20$sentimiento=ifelse(art_df_senti20$puntaje==0,"Neutral",art_df_senti20$sentimiento)

art_sent20 <- art_df_senti20 %>% group_by(sentimiento) %>% summarise(count=n()) %>%
  mutate(per = round(prop.table(count)*100,1))
```

Y graficamente:

```
## 2015
ggplot(art_sent15, aes(x=sentimiento, y=per, fill=sentimiento))+
  geom_bar(position="dodge", stat="identity")
```




```
## 2020
ggplot(art_sent20, aes(x=sentimiento, y=per, fill=sentimiento))+
  geom_bar(position="dodge", stat="identity")
```



En este método se clasifican más palabras asociadas a sentimientos que en el anterior. Aproximadamente un 10% para el 2015 se asocian a positivo y otro 10% a negativo. Para el 2020 aproximadamente un 20% se asocia a negativo y un poco más del 10% a positivo.

Reflexiones finales sobre el trabajo

En este trabajo se aplicaron varias técnicas de minería y análisis de texto para obtener un pantallazo general de las leyes de presupuesto más recientes de nuestro país.

Se observaron las palabras más utilizadas en ambos documentos, así como las relacionadas a determinadas categorías arbitrarias, y/o relacionadas a ciertos sentimientos.

De esta manera, identificando patrones o correlaciones entre palabras logramos encontrar cierta información y presentarla de manera amigable.

Si bien las fuentes de información elegidas no permitieron aplicar ciertas técnicas por ser un texto plano, si fue posible sintetizar su contenido y realizar un análisis comparativo aplicando varias técnicas vistas en el curso para procesar, transformar, extraer características y analizar descriptivamente ambas leyes, principalmente utilizando herramientas de análisis visual.