Solución ejercicios - Sesión 3

Capacitación en R y herramientas de productividad - nivel intermedio

PE Ciencia de Datos para la Producción Estadística

Solución ejercicios - Sesión 3

```
library(feather)
library(dplyr)
library(stringr)
library(tm)
```

A partir de la base de datos de la encuesta Casen 2020, disponible en la carpeta de la clase de hoy `casen_2020_edit.feather`, realiza las siguientes tareas.

Ejercicio 1

1- Ayudándote de expresiones regulares, y de la manera más suscinta que puedas, selecciona todas las variables de cuestionario de:

- el módulo de Ocupación
- el módulo de vivienda

1.1 Módulo de Ocupación

Con la ayuda del helper "matches" implementado dentro de select es posible entregarle a la función una expresión regular que ofrezca el patrón requerido.

En el caso de ocupación se puede utilizar la siguiente expresión, que rastrea a todas las variables que comienzan con "o" y que pueden continuar con 1 o 2 dígitos, con "ficio" o con "cup", además de otras variables que comiencen con "rama" o con "activ".

```
casen_o <- casen %>%
  select(matches("^o((\\d{1,2})|ficio|cup)|^rama|^activ"))
```

Son 36 variables las que componen el módulo de ocupación, entre variables de cuestionario y variables construidas en procesamiento.

```
length(names(casen_o))
```

```
## [1] 36
```

```
names(casen_o)
```

```
##  [1] "o1"         "o2"         "o3"         "o3b"        "o4"
##  [6] "o6"         "o7"         "o7_esp"     "o9a"        "o9b"
## [11] "oficio4_08" "oficio1_08" "oficio4_88" "oficio1_88" "o15"
## [16] "o16"        "o17"        "o24"        "rama4"      "rama1"
## [21] "rama4_rev3" "rama1_rev3" "o29"        "o30"        "o31"
## [26] "o32"        "o32_esp"    "o32b"       "o33a"       "o33b"
## [31] "o34"        "o35"        "o36"        "activ"      "activ2"
## [36] "ocup_inf"
```

1.1 Módulo de Vivienda

En el caso del módulo de vivienda era bastante sencillo. Si bien se puede capturar de forma más estricta el patrón de nombres, para aislar las variables de vivienda en casen solo bastaba con disponer un patrón que detectara nombres que partieran con "v" y contuvieran uno o más números.

Son 19 variables.

```
casen_v <- casen %>%
  select(matches("^v\\d+"))
```

Son 19 las variables que componen el módulo de vivienda.

```
length(names(casen_v))
```

```
## [1] 19
```

```
names(casen_v)
```

```
##  [1] "v1"           "v1_casa"      "v1_depto"     "v13"
##  [5] "v13_propia"   "v13_arrendada" "v13_cedida"   "v19_preg"
##  [9] "v19"          "v20"          "v20_esp"      "v20_red"
## [13] "v22"          "v23"          "v23_sistema"  "v23_cajon"
## [17] "v27"          "v28"          "v29"
```

Ejercicio 2

2- La tabla `casen_2020_edit.feather` contiene las variables o9a, o9b y o24, que representan, respectivamente, "ocupación", "tareas en la ocupación" y "rama de actividad económica" de las personas que declaran estar ocupadas en el periodo de referencia.

Estas variables son utilizadas como insumo para la codificación automática de los clasificadores de ocupación y rama de actividad económica.

Crea una función que reciba como argumento una variable character y la procese de la siguiente manera:

- Pase todos los caracteres de una glosa a minúscula
- Remueva todos los signos de puntuación y caracteres especiales
- Remueva todos los números
- Extraiga espacios adicionales entre palabras
- Remueva *stopwords* (para esto pueden usar las librerías `tm`, `quanteda`, entre otras) La función debe retornar una variable con glosas procesadas.

Existen variadas estrategias para lograr este ejercicio. En este caso se implementó lo siguiente:.

Se tomó el vector de stopwords de la librería `tm`y se ordenaron en un patrón de búsqueda rodeando cada palabra con la expresión `\\b`, que delimita una palabra para que la búsqueda sea estricta.

Luego se creo una función `clean_strings()` que implementa los siguientes pasos solicitados. No es trivial la utilización de `str_replace_all()` en lugar de `str_extract_all()`. Cuando la segunda función extrae los patrones solicitados no deja espacios, pudiendo juntar palabras, lo que crea expresiones que no estaban ahí originalmente. Dado eso, la recomendación es reemplazar los patrones de búsqueda por espacios vacios (" "), y luego con las funciones `str_trim()` y `str_squish()` es posible remover los espacios a principio y al final de la cadena, y dentro de la cadena, respectivamente.

```
sw_regex = paste(stopwords('es'), collapse = '\\b|\\b')
sw_regex = paste0('\\b', sw_regex, '\\b')

clean_strings = function(data, svar) {
  data %>%
    mutate('{{svar}}_clean' := str_to_lower({{svar}}) %>%
             str_replace_all("[[:punct:]]", ' ') %>%
             str_replace_all("\\d", ' ') %>%
             str_replace_all(sw_regex, ' ') %>%
             str_trim() %>%
             str_squish())

}
```

No era requerido en la tarea, pero podemos evaluar la función en "o9a" para observar su resultado.

```r
casen = casen %>%
  clean_strings(o9a)
casen %>%
  filter(o9a!="") %>%
  select(starts_with("o9a")) %>%
  head()
```

```
## # A tibble: 6 x 2
##   o9a                                                o9a_clean
##   <chr>                                              <chr>
## 1 VENDEDOR(A) INDEPENDIENTE                          vendedor independiente
## 2 GASTRONOMÍA                                        gastronomía
## 3 ARTESANA EN PASTELERÍA                             artesana pastelería
## 4 DGAC                                               dgac
## 5 MAESTRA DE COCINA                                  maestra cocina
## 6 OPERADOR EN PRODUCCIÓN PLATA DE HARINA DE PESCADO operador producción plata h~
```