

MAESTRÍA EN ESTADÍSTICA APLICADA/ MANEJO DE DATOS

Unidad 4: Selección, limpieza y transformación de datos

4.3 Limpieza de datos utilizando funciones de Rstudio

En un proyecto de Ciencias de Datos, es fundamental realizar un procesamiento previo de limpieza y transformaciones necesaria, de tal forma, la data quede lista para su posterior procesamiento análisis de los mismos, por medio de distintos estadísticos y gráficos que permitan describir las variables con las que se trabaja y determinar las posibles relaciones entre ellas. La calidad del conjunto de datos finalmente obtenido determinará la robustez y fiabilidad de los resultados del proyecto.

4.3.1 Manejo de String

Es común encontrarse problemas en campos tipo cadena de caracteres (string), tales como: espacios en blancos en los extremos, caracteres que no pertenecen a un patrón o formato, división o separación de contenido de un campo, estandarizar contenido, entre otros.

Para el procesamiento de cadenas de caracteres, se requiere el uso de la librería stringr.

```
library(stringr)
```

Para la aplicación de los siguientes funciones de limpieza de datos, supongamos que se dispone una tabla de datos de empleados:

Los datos se encuentran en el archivo Empleados.csv, por lo que se deben leer desde esa fuente.

```
df_emp<- read.csv("C:/Users/marcelochavez/Documents/MAESTRIA_ESTADISTICA_APLICADA_ESPOL/MODULO_2/TALLER,
str(df_emp)
```

```
## 'data.frame':    5 obs. of  7 variables:
## $ DNI           : int  123456789 234567891 345678912 456789123 567891234
## $ Nombres       : chr  "Carlos Daniel Macias Torres" "ADRIANA VALENTINA FLORES MORA" "LORENA Vanesa
## $ CiudadResidencia: chr  "Guayaquil" "Ambato" "Manta" "Quito" ...
## $ FechaNacimiento: chr  "10/5/1985" "4/6/1986" "18/12/1990" "10/10/1975" ...
## $ AniosLabor    : chr  "15 a\xflor" "10 a\xflor" "5 a\xflor" "20 a\xflor" ...
## $ Remuneracion   : int  2500 2000 1300 3400 2000
## $ Code          : chr  "111e" "e456" "893" "567" ...
```

Estado original de la data:

DNI	Nombres	CiudadResidencia	FechaNacimiento	AniosLabor	Remuneracion	Code
123456789	Carlos Daniel Macias Torres	Guayaquil	10/5/1985	15 años	2500	111e
234567891	ADRIANA VALENTINA FLORES MORA	Ambato	4/6/1986	10 años	2000	e456
345678912	LORENA Vanessa Peña Lino	Manta	18/12/1990	5 años	1300	893
456789123	GALO TONY DURANGO RIOS	Quito	10/10/1975	20 años	3400	567
567891234	Petra Josefina Suarez Vera	Guayaquil	22/1/1992	8 años	2000	865

Figure 1: Data sucia

Eliminación de espacios en blancos en los extremos

Se puede observar en la tabla de datos, que en el campo Ciudad de Residencia, existen espacios en blancos al inicio y final. Para eliminar los espacios en blanco se utiliza la función `str_trim(string=[cadena de texto], side=c("left","right","both"))`

```
df_emp$CiudadResidencia<-str_trim(string = df_emp$CiudadResidencia, side = "both")
View(df_emp)
```

Dividir el contenido de un campo

En el caso del campo fecha, fue cargado desde la fuente de origen como tipo string, pero es conveniente separar los elementos de la fecha en tres campos distintos. La función que nos permite realizar dicha acción es `str_split_fixed(string = [cadena de texto], pattern = "[char]", n=[cantidad de partes])`. Esta función permite crear una matriz con las partes obtenidas de la segmentación del contenido.

```
str_split_fixed(string = df_emp$FechaNacimiento, pattern = "/", n=3)
```

```
##      [,1] [,2] [,3]
## [1,] "10" "5"  "1985"
## [2,] "4"  "6"  "1986"
## [3,] "18" "12" "1990"
## [4,] "10" "10" "1975"
## [5,] "22" "1"  "1992"
```

Para asignar cada fila en el Dataframe, podemos utilizar la notación matricial `[,columna]`, de la siguiente forma:

```
df_emp$dia<-str_split_fixed(string = df_emp$FechaNacimiento, pattern = "/", n=3)[,1]
df_emp$mes<-str_split_fixed(string = df_emp$FechaNacimiento, pattern = "/", n=3)[,2]
df_emp$anio<-str_split_fixed(string = df_emp$FechaNacimiento, pattern = "/", n=3)[,3]
df_emp
```

```
##      DNI      Nombres CiudadResidencia FechaNacimiento
## 1 123456789 Carlos Daniel Macias Torres      Guayaquil      10/5/1985
## 2 234567891 ADRIANA VALENTINA FLORES MORA      Ambato        4/6/1986
## 3 345678912 LORENA Vanessa Pe\xfla Lino      Manta          18/12/1990
## 4 456789123      GALO TONY DURANGO RIOS      Quito            10/10/1975
## 5 567891234      Petra Josefina Suarez Vera      Guayaquil        22/1/1992
##  AniosLabor Remuneracion Code dia mes anio
## 1 15 a\xflor      2500 111e 10  5 1985
## 2 10 a\xflor      2000 e456  4  6 1986
## 3  5 a\xflor      1300  893 18 12 1990
## 4 20 a\xflor      3400  567 10 10 1975
## 5  8 a\xflor      2000  865 22  1 1992
```

Otra forma de separar contenido de un campo, es utilizar la función `separate()` de la library(tidyr), esto lo revisaremos en la siguiente sección.

Extraer fragmentos de una cadena que corresponden a un determinado patrón

Como se puede observar en la tabla de datos de Empleados, existe el campo años de labor como tipo texto, nos interesaría tomar solo el componente numérico de dicho contenido.

```
formato<-"[0-9]+" #formato de uno o más números desde el inicio de la cadena
df_emp$tiempo<-str_extract(string = df_emp$AniosLabor, pattern = formato)
df_emp
```

```
##      DNI      Nombres CiudadResidencia FechaNacimiento
## 1 123456789 Carlos Daniel Macias Torres      Guayaquil      10/5/1985
```

```
## 2 234567891 ADRIANA VALENTINA FLORES MORA      Ambato      4/6/1986
## 3 345678912 LORENA Vanessa Pe\xf1a Lino      Manta      18/12/1990
## 4 456789123 GALO TONY DURANGO RIOS      Quito      10/10/1975
## 5 567891234 Petra Josefina Suarez Vera      Guayaquil    22/1/1992
##   AniosLabor Remuneracion Code dia mes anio tiempo
## 1 15 a\xflor      2500 111e 10 5 1985 15
## 2 10 a\xflor      2000 e456 4 6 1986 10
## 3 5 a\xflor      1300 893 18 12 1990 5
## 4 20 a\xflor      3400 567 10 10 1975 20
## 5 8 a\xflor      2000 865 22 1 1992 8
```

Reemplazar/suprimir contenido parcial

En el caso que se requiera reemplazar o suprimir contenido parcial en una cadena de texto, se puede utilizar la función `gsub()`. Como se puede observar en la tabla de Empleados, el campo `Code` tiene un caracter incorrecto “e”, se desea suprimir dicho contenido, para lo cual, se utiliza la función `gsub(“formato”, “reemplazo”, datos)`.

```
gsub("e", "", df_emp$Code)
```

```
## [1] "111" "456" "893" "567" "865"
```

Ahora se debe actualizar el contenido del campo:

```
df_emp$Code<-gsub("e", "", df_emp$Code)
```

Homogenizar mayúsculas/ minúsculas

Si observamos el campo de nombre en la tabla Empleados, nos podemos percatar que algunos se muestran en mayúsculas y otros no, por tal razón, si por procesamiento es necesario homogenizar dicho contenido, se puede utilizar las funciones `str_to_upper()` o `str_to_lower()`.

```
df_emp$Nombres<-str_to_upper(string = df_emp$Nombres)
```

4.3.2 Manejo de Fechas

El procesamiento de campos tipo fecha también es importante darle un orden correspondiente. En la tabla Empleados existe el campo “Fecha de Nacimiento”, el tipo de dato origen de este campo en texto, sin embargo se lo requiere convertir en tipo `date` para utilizar las funciones de procesamiento de fechas. Para realizar la conversión se utiliza el comando `as.Date()`.

Los formatos disponibles en R para el manejo de fechas son:

El formato que se utilizará para el campo fecha es `%d/%m/%Y`.

```
df_emp$FechaNac2<-as.Date(df_emp$FechaNacimiento, format="%d/%m/%Y")
```

En caso de requerir que el nombre del mes se incorpore como una columna adicional, se puede utilizar la función `format(campo, formato)`:

```
df_emp$nmnes<-format(df_emp$FechaNac2, "%B")
```

Si se requiere realizar una operación de diferencias entre fechas, se utiliza el comando `difftime(fecha final, fecha inicio, units=“”).` Por ejemplo, se requiere incorporar una nueva columna a la tabla de datos con la cantidad de días de edad del empleado.

```
fecha_actual=Sys.Date()
difftime(fecha_actual, df_emp$FechaNacimiento, units = "days" )
```

```
## Time differences in days
## [1] 735275.8 737435.8 732139.8 735122.8 731012.8
```

Para el procesamiento de datos de fecha también se puede aplicar las funciones del paquete `lubridate`.

lubridate

Con esta librería resulta muy sencillo trabajar con fechas y horas. Nos permite convertir cadenas de caracteres al formato de fecha POSIXct, reconociendo distintos formatos. Algunos ejemplos:

– Año, mes y día separados por guiones o barras

```
#install.packages("lubridate")
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
## The following objects are masked from 'package:base':
##
##      date, intersect, setdiff, union
fechas <- c("2017-05-03", "2017/05/03")
ymd(fechas)
```

```
## [1] "2017-05-03" "2017-05-03"
```

– Día, mes y año sin separación o incluso escritos (en inglés)

```
fechas <- c("03052017", "3 May 2017")
dmy(fechas)
```

```
## [1] "2017-05-03" "2017-05-03"
```

– Fechas y horas, pudiendo indicar la zona horaria

```
ymd_hms("2017-05-03 09:00:00", tz = "America/Bogota")
```

```
## [1] "2017-05-03 09:00:00 -05"
```