Cadenas de Markov

Armando Ocampo

Librerías de trabajo

Antes de iniciar vamos a llamar a las siguiente librerías. Para su instalación necesitamos la función *install.packages()* y el nombre de la paquetería entre comillas. Por ejemplo, *install.packages('markovchain')*.

```
library(markovchain)
library(moments)
library(clipr)
library(readr)
```

Dudas de la clase previa

Copiando resultados desde R a Excel

A continuación se comparten algunos ejemplos de cómo se pueden copiar los datos de los objetos en R y pegar la información en un libro de excel. El primer ejemplo copia la información en el pisa papeles de la máquina (o clipboard) y permite su pegado posterior. Para ello se utiliza la función *write_clip()* de la paquetería *clipr*. En esta función solo se coloca el nombre del objeto que se desea copiar.

```
vacunas_df <- read_csv('../data/dataset_vacunas.csv')
# write_clip(vacunas_df)
```

Después de correr este comando abrimos un libro en excel y pegamos los datos.

Otra manera de realizar este proceso es mediante la función *write.table()* de la paquetería *utils*. Además del objeto, se agrega el destino de la información, en este caso de nuevo al *clipboard* del sistema. Se añade la separación por tabulador y se elimina el nombre de las filas para no redundar esta información.

```
# write.table(vacunas_df, "clipboard",
#             sep="\t", row.names=FALSE)
```

Al igual que en el código previo, abrimos una hoja de Excel y pegamos la información.

Transformación de los datos

Durante el proceso de minería de datos y desarrollo de modelos podemos encontrar información que no sigue una distribución normal o con valores extremos. En este caso se sugiere realizar una transformación de los datos. No obstante, queda la duda sobre el tipo de transformación a realizar. La función *skewness()* de la paquetería *moments* calcula el sesgo de la información y facilita este proceso. En ella se coloca el vector del cual se desea conocer el tipo de sesgo que presenta. De la misma manera, es válido realizar la transformación de la información al número de variables que sea necesario, previo al desarrollo de algoritmos o análisis de datos.

```
# Sesgo de PIB
skewness(vacunas_df$GDP)
```

```
## [1] 1.731123
```

```r
# Sesgo de esperanza de vida
skewness(vacunas_df$Life_expectancy)
```

```
## [1] -0.5082224
```

```r
# Sesgo de longitud de sépalo
skewness(iris$Sepal.Length)
```

```
## [1] 0.3117531
```

```r
# Sesgo de ancho de pétalo
skewness(iris$Petal.Width)
```

```
## [1] -0.1019342
```

Nota:si el sesgo es positivo se sugiere realizar una transformación cuadrática, logarítmica o inversa (1/x). Por el contrario, para sesgos negativos se recomiendan las siguientes transformaciones: sqrt(max(x+1) - x), log10(max(x+1) - x), 1/(max(x+1) - x)

Cadenas de Markov

Las cadenas de Markov permiten conocer la probabilidad de que suceda un evento dado la aparición de un evento previo. Para esto se necesitan las siguientes caracterísitcas:

1. Existe un número finito de estados.
2. La probabilidad de ocurrencia de un evento depende del suceso inmediatamente anterior.
3. Las probabilidades permanecen constantes en el tiempo.

Para ello, se define como un estado a los posibles eventos que puedan ocurrir en un experimento. Cabe destacar que se debe conocerce la probabilidad de ocurrencia de cada evento. A continuación se generará un ejemplo utilizando cadenas de Markov. En el cual se desarrolla una matriz de consumo de alimentos, tomando la relación de consumo de frutas, verduras y carne.

Primero se define la matriz con la tasa de consumo.

```r
consumo = matrix(c(0.2,0.3,0.5,0.4,0.2,0.4,0.1,0.6,0.3),
           nrow = 3,byrow = TRUE)
consumo
```

```
##      [,1] [,2] [,3]
## [1,]  0.2  0.3  0.5
## [2,]  0.4  0.2  0.4
## [3,]  0.1  0.6  0.3
```

Posteriormente se genera el objeto *markovchain*, a partir de la función *new()*, la cual forma parte de la paquetería *methods*. Con esta función se define el tipo de objeto, se genera la matriz de transición y los estados.

```r
mc <- new('markovchain', transitionMatrix = consumo,
          states = c('frutas', 'verduras', 'carne'),
          name = 'Cadena1')

str(mc) # lista de ragumentos
```

```
## Formal class 'markovchain' [package "markovchain"] with 4 slots
##   ..@ states          : chr [1:3] "frutas" "verduras" "carne"
##   ..@ byrow           : logi TRUE
##   ..@ transitionMatrix: num [1:3, 1:3] 0.2 0.4 0.1 0.3 0.2 0.6 0.5 0.4 0.3
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:3] "frutas" "verduras" "carne"
```

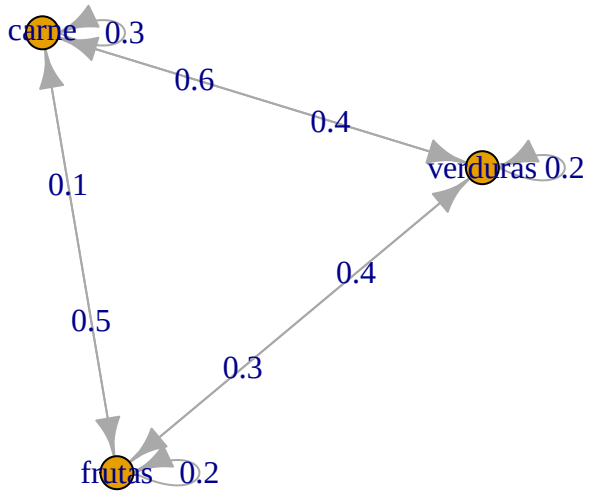
```
##   .. .. ..$ : chr [1:3] "frutas" "verduras" "carne"
##   ..@ name            : chr "Cadena1"
```

```
summary(mc)
```

```
## Cadena1  Markov chain that is composed by:
## Closed classes:
## frutas verduras carne
## Recurrent classes:
## {frutas,verduras,carne}
## Transient classes:
## NONE
## The Markov chain is irreducible
## The absorbing states are: NONE
```

Para una visualización de nuestro modelo utilizaremos la función *plot()* y el nombre de nuestro objeto tipo cadena. Este gráfico muestra la probabilidad de pasar de un estado a otro.

```
set.seed(123)
plot(mc)
```



La función *recurrentClasses()* nos permite identificar los eventos que conforman nuestra cadena. Esta información también puede ser visualizada mediante la función *summary()*.

```
recurrentClasses(mc)
```

```
## [[1]]
## [1] "frutas"   "verduras" "carne"
```

La función *transitionProbability()*, permite conocer la probabilidad de suceso entre un estado inicial y un estado

final. En el siguiente ejemplo se detalla la probabilidad de que un individuo consuma carne si previamente ha consumido frutas. Esto es igual a buscar mediante corchetes la posición en la cual se encuentra este valor en la matriz de transición, o colocar de manera directa los elemntos de estudio.

```
transitionProbability(object = mc, t0 = 'frutas',
                      t1 = 'carne')
```

```
## [1] 0.5
```

```
mc[1,3]
```

```
## [1] 0.5
```

```
mc['frutas', 'carne']
```

```
## [1] 0.5
```

No obstante, esta función no permite conocer cual sería la probabilidad de consumo de algún alimento en eventos posteriores. Para conocer la siguiente probabilidad se eleva a la n potencia la matriz de transición. Donde n es el estado de interés. En el siguiente ejemplo se identificará cual será la frecuencia de consumo de cada alimento en el segundo estado de transición.

```
n <- 2
mc ^ n
```

```
## Cadena1^2
##  A  3 - dimensional discrete Markov Chain defined by the following states:
##  frutas, verduras, carne
##  The transition matrix  (by rows)  is defined as follows:
##          frutas verduras carne
## frutas     0.21     0.42  0.37
## verduras   0.20     0.40  0.40
## carne      0.29     0.33  0.38
```

Es posible omitir el valor de n, y solo elevar a la potencia adecuada la matriz de transición. A continuación se muestra la probabilidad de consumo de cada alimento en el tercer evento.

```
mc ^ 3
```

```
## Cadena1^3
##  A  3 - dimensional discrete Markov Chain defined by the following states:
##  frutas, verduras, carne
##  The transition matrix  (by rows)  is defined as follows:
##          frutas verduras carne
## frutas    0.247    0.369 0.384
## verduras  0.240    0.380 0.380
## carne     0.228    0.381 0.391
```

Podemos continuar con el estado 4...

```
mc ^ 4
```

```
## Cadena1^4
##  A  3 - dimensional discrete Markov Chain defined by the following states:
##  frutas, verduras, carne
##  The transition matrix  (by rows)  is defined as follows:
##          frutas verduras  carne
## frutas   0.2354   0.3783 0.3863
## verduras 0.2380   0.3760 0.3860
## carne    0.2371   0.3792 0.3837
```

Y solo modificamos el valor de n para el estado 5

```
mc ^ 5
```

```
## Cadena1^5
##  A  3 - dimensional discrete Markov Chain defined by the following states:
##  frutas, verduras, carne
##  The transition matrix  (by rows)  is defined as follows:
##           frutas verduras   carne
## frutas   0.23703  0.37806 0.38491
## verduras 0.23660  0.37820 0.38520
## carne    0.23747  0.37719 0.38534
```

Cabe destacar que conforme vamos avanzando en los estados, la matriz presenta cambios sutiles, hasta el punto de permanecer constante. A esto se le conoce como estado de equilibrio. Chequemos el estado 6.

```
mc ^ 6
```

```
## Cadena1^6
##  A  3 - dimensional discrete Markov Chain defined by the following states:
##  frutas, verduras, carne
##  The transition matrix  (by rows)  is defined as follows:
##            frutas verduras    carne
## frutas   0.237121 0.377667 0.385212
## verduras 0.237120 0.377740 0.385140
## carne    0.236904 0.377883 0.385213
```

Y el estado 7.

```
mc ^ 7
```

```
## Cadena1^7
##  A  3 - dimensional discrete Markov Chain defined by the following states:
##  frutas, verduras, carne
##  The transition matrix  (by rows)  is defined as follows:
##             frutas  verduras     carne
## frutas   0.2370122 0.3777969 0.3851909
## verduras 0.2370340 0.3777680 0.3851980
## carne    0.2370553 0.3777756 0.3851691
```

Para evitar continuar elevando a la n potencia la matriz de transición, la función *steadyStates()* arroja los valores de distribución estacionaria para cada evento.

```
estables <- steadyStates(mc)
estables
```

```
##        frutas  verduras     carne
## [1,] 0.237037 0.3777778 0.3851852
```

Además de conocer la probabilidad de consumo o de suceso de un evento es posible determinar el total de usuarios que realizarian un evento dado un número determinado de pasos. En el siguiente ejemplo se parte de el consumo de aliementos de 1000 perosnas. 200 consumieron inicialmente frutas, 300 verduras y 500 carne. A partir de la información reportada en la cadena d eMarkov, ¿Cuál será la relación de consumo de alimentos en el tercer paso?

Para esto se genera un vector de consumo inicial. Posteriormente se múltiplica la matriz de transición por el n de pasos necesarios. A continuación se multiplica esta matriz por el vector de consumo. Derivado de este modelo concluimos que el consumo de frutas se traduce a aproximadamente 235 personas para frutas, 378 para verduras y 386 individuos para carne.

```r
set.seed(123)
X0 = c(200,300,500)
n = 3
Xn = X0*(mc^n)
Xn
```

```
##      frutas verduras carne
## [1,]  235.4    378.3 386.3
```

```r
sum(Xn)
```

```
## [1] 1000
```