Introducción a R Markdown

Lino AA Notarantonio

11/6/2020

Table of Contents

# Introducción

Queremos mostrar las capacidades de R Markdown en la creación de un documento que contenga de manera transparente datos, código y texto, con fórmulas. Como ejemplo, usaremos los mismos datos de la presentación para estimar un modelo de regresión lineal, con diagnóstico de la estimación.

Para simplificar el manejo de datos, conviene crear un *project* (directorio de trabajo) y poner los datos en este directorio. Antes de empezar con el manejo de datos, se dará una breve introducción a la redacción del documento usando el lenguaje R Markdown.

# Introducción a R Markdown

Un archivo R Markdown tiene la extensión **Rmd**. La gran diferencia en la redacción de un documento R Markdown se encuentra en el resultado final. En el encabezado de un archivo R Markdown se pueden pasar diferentes valores de salida:

* **output: word\_document**, para generar un documento Word;
* **output: powerpoint\_presentation**, para generar una presentación Powerpoint.

entre otros: véase, por ejemplo, [aquí](https://rmarkdown.rstudio.com/lesson-9.html).

Si la salida del programa R Markdown es un documento Word, podemos dividir el material usando secciones, subsecciones, subsubsecciones, etc., usando el tag **#**, **##**, **###**, respectivamente, hasta nivel 6.

Se pueden redactar fórmulas matemáticas, de manera semejante a (La)teX, usando

* **$ inline equation $**, para ecuaciones en el texto;
* **\[ displayed equation \]**, para ecuaciones centrada a parte.

Se pueden incluir en el documento de salida

* *bullet points*, listas ordenadas;
* tablas;
* nota a pie de página,

y más. Pueden bajar el Cheat Sheet [aquí](https://raw.githubusercontent.com/rstudio/cheatsheets/master/rmarkdown-2.0.pdf).

Para facilitar la comprensión del lenguaje R Markdown, ponemos a disposición de los participantes los archivos **Rmd** y los documentos (Word; Powerpoint) de salida.

# Análisis preliminar de los datos

El dataset se baja del sitio [Our World in Data](https://ourworldindata.org/grapher/learning-outcomes-vs-gdp-per-capita?time=latest) y agrupa observaciones de puntajes promedio de aprendizaje y de PIB por capita, entre otros.

## Carga de los datos

logdp <- read.csv("logdp.csv")  
class(logdp)

## [1] "data.frame"

La primera línea de código, a la derecha del símbolo de asignación, **<-**, carga los datos y asigna el resultado al objeto de R **logdp**, a la izquierda de **<-**.

La segunda línea de código muestra que el objeto es un **dataframe**.

Después,

dim(logdp)

## [1] 49988 6

muestra que el dataframe consiste de 49988 observaciones y 6 variables; el nombre de las variables se despliega con

names(logdp)

## [1] "Entity"   
## [2] "Code"   
## [3] "Year"   
## [4] "Average.learning.outcome.score...score.."  
## [5] "GDP.per.capita...."   
## [6] "Population"

También, se pueden desplegar las primeras

head(logdp)

## Entity Code Year Average.learning.outcome.score...score..  
## 1 Afghanistan AFG 1800 NA  
## 2 Afghanistan AFG 1801 NA  
## 3 Afghanistan AFG 1802 NA  
## 4 Afghanistan AFG 1803 NA  
## 5 Afghanistan AFG 1804 NA  
## 6 Afghanistan AFG 1805 NA  
## GDP.per.capita.... Population  
## 1 NA 3280000  
## 2 NA 3280000  
## 3 NA 3280000  
## 4 NA 3280000  
## 5 NA 3280000  
## 6 NA 3280000

y las últimas 6 observaciones

tail(logdp)

## Entity Code Year Average.learning.outcome.score...score..  
## 49983 Zimbabwe ZWE 2014 NA  
## 49984 Zimbabwe ZWE 2015 NA  
## 49985 Zimbabwe ZWE 2016 NA  
## 49986 Zimbabwe ZWE 2017 NA  
## 49987 Zimbabwe ZWE 2018 NA  
## 49988 Zimbabwe ZWE 2019 NA  
## GDP.per.capita.... Population  
## 49983 1797 13587000  
## 49984 1759 13815000  
## 49985 1729 14030000  
## 49986 NA 14237000  
## 49987 NA 14439000  
## 49988 NA 14645000

Detectamos la presencia de valores ausentes, **NA**, en el dataset.

## Manipulaciones del dataset

Vimos que el dataset contiene nombre de paises y las correspondientes siglas internacionales, por lo que consideramos que una de las variables es superflua; además, se puede acortar el nombre de las variables del puntaje promedio y del PIB por cápita.

**Comentario** Vale la pena subrayar que el dataset original no se modifica. Todas las manipulaciones quedarán en este documento solamente.

Empezamos a eliminar la primera variable (nombre completo del país):

logdp <- logdp[c(-1)]  
names(logdp)

## [1] "Code"   
## [2] "Year"   
## [3] "Average.learning.outcome.score...score.."  
## [4] "GDP.per.capita...."   
## [5] "Population"

El comando **logdp[c(-1)]** en el primera línea de código, a la derecha del operador de asignación **<-**, opera un *subsetting* del dataset **logdp**, es decir, la determinación de un subconjunto del dataset originario; el *subsetting* se determina por los corchetes, **logdp[ ]**.

El efecto del subsetting consiste en la eliminación de la primera variable, que se realiza usando el operador de *concatenación*, **c()**, aplicado al conjunto de las variables de interés, numeradas de la **1** hasta la **6**; en este caso, queremos eliminar la primera variable, eliminación que se puede hacer mediante el signo menos, **-**: **c(-1)**.

Después, para acortar el nombre de las variables del puntaje promedio y del PIB por cápita, ejecutamos

colnames(logdp)[3] <- "score"  
colnames(logdp)[4] <- "GDPcap"  
colnames(logdp)[5] <- "Pop"  
head(logdp)

## Code Year score GDPcap Pop  
## 1 AFG 1800 NA NA 3280000  
## 2 AFG 1801 NA NA 3280000  
## 3 AFG 1802 NA NA 3280000  
## 4 AFG 1803 NA NA 3280000  
## 5 AFG 1804 NA NA 3280000  
## 6 AFG 1805 NA NA 3280000

La función **colnames(logdp)** tiene como salida un vector de caracteres, que agrupa los nombres ordenados del dataframe.

El comando **colnames(logdp)[i]** indica el componente **i** de esta salida; las primeras dos líneas de código asignan el carácter **score** al tercer componente, **GDPcap** al cuarto y **Pop** al quinto, respectivamente.

## Manipulación de variables

La manipulación de una variable individual en un dataframe se realiza mediante la *extracción* de esta variable del dataframe. Por ejemplo, si queremos analizar la variable **Pop**, la extraemos del dataset mediante el operador **$**, usando la sintaxis **logdp$Pop**. Para desplegar las primeras seis observaciones de esta variable,

head(logdp$Pop)

## [1] 3280000 3280000 3280000 3280000 3280000 3280000

### Selección de un país

Para seleccionar un país específico, podemos pedir que la variable **Code** del dataset sea igual al valor del código deseado. Por ejemplo, si queremos usar los valores de Suecia (código **SWE** en el dataset):

swe <- logdp[logdp$Code == "SWE",]

Nótese como en el subsetting en la primera línea de código, se pone la coma, para indicar la selección de todas las observaciones (renglones) asociadas a Suecia (sintaxis semejante a la que se usa en Matlab).

**Comentario** El símbolo **=** es también un operador de asignación en R, con menor prioridad de **<-** (véase [aquí](https://stackoverflow.com/questions/1741820/what-are-the-differences-between-and-assignment-operators-in-r)).

El dataset de las solas observaciones de Suecia consiste de

dim(swe)

## [1] 720 5

720 observaciones y 5 variables.

## Estadísticos principales

La estimación de los estadísticos principales de un dataframe se realiza de manera sencilla usando la función **summary()**; por ejemplo, aplicada a la variable **Pop** :

summary(swe$Pop)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 2354189 3614120 5471638 5645125 7701750 10036000 500

o al dataframe completo

summary(swe)

## Code Year score GDPcap   
## Length:720 Length:720 Min. :411.6 Min. : 906   
## Class :character Class :character 1st Qu.:451.1 1st Qu.: 1231   
## Mode :character Mode :character Median :496.1 Median : 1343   
## Mean :479.4 Mean : 3793   
## 3rd Qu.:507.0 3rd Qu.: 1640   
## Max. :526.2 Max. :44371   
## NA's :710 NA's :3   
## Pop   
## Min. : 2354189   
## 1st Qu.: 3614120   
## Median : 5471638   
## Mean : 5645125   
## 3rd Qu.: 7701750   
## Max. :10036000   
## NA's :500

o solo a la parte numérica (nótese el subsetting que se realiza para obtener un subconjunto del dataframe originario):

summary(swe[c(-1, -2)])

## score GDPcap Pop   
## Min. :411.6 Min. : 906 Min. : 2354189   
## 1st Qu.:451.1 1st Qu.: 1231 1st Qu.: 3614120   
## Median :496.1 Median : 1343 Median : 5471638   
## Mean :479.4 Mean : 3793 Mean : 5645125   
## 3rd Qu.:507.0 3rd Qu.: 1640 3rd Qu.: 7701750   
## Max. :526.2 Max. :44371 Max. :10036000   
## NA's :710 NA's :3 NA's :500

La salida de la función **summary()** no contiene la desviación estándar.

Podemos aplicar la función **sd()** a cada variable, por ejemplo, **Pop**

sd(swe$Pop, na.rm = TRUE)

## [1] 2268649

o podemos usar la vectorización del lenguaje R y aplicar la función **sapply()** a la parte numèrica del dataframe, eliminando los valores ausentes

sapply(swe[c(-1,-2)], sd, na.rm = TRUE)

## score GDPcap Pop   
## 3.920617e+01 7.496684e+03 2.268649e+06

La función **sapply()** es parte de una familia de funciones que operan de manera semejante en diferentes contextos; todas tienen la misma sintaxis: **sapply(nombre\_dataset, función, ...)**, donde **...** indican parámetros adicionales.

# Regresión lineal

Como ejemplo de una estimación de una regresión lineal en R, queremos estimar el efecto de la educación (proxied por **score**) y el nivel de la población sobre el PIB por cápita:

**Comentario** Cuando se usan variables del mismo dataframe, conviene usar la función **attach()**

attach(swe)

Cuando se termina el uso del dataframe se usa **detach()** para terminar el uso del dataframe.

## Estimación del modelo de regresión

La estimación del modelo de regresión lineal se realiza mediante la función **lm()**

m <- lm(GDPcap ~ score + Pop)

Conviene asignar el resultado de la estimación a un objeto de R, en este caso, **m**.

El resumen de la estimación se obtiene usando la función **summary()**:

summary(m)

##   
## Call:  
## lm(formula = GDPcap ~ score + Pop)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3915 -1076 -347 1478 3988   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.180e+05 1.522e+04 -7.757 0.000111 \*\*\*  
## score 1.678e+01 3.464e+01 0.484 0.642948   
## Pop 1.597e-02 2.474e-03 6.453 0.000349 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2853 on 7 degrees of freedom  
## (710 observations deleted due to missingness)  
## Multiple R-squared: 0.931, Adjusted R-squared: 0.9113   
## F-statistic: 47.25 on 2 and 7 DF, p-value: 8.612e-05

Nótese que la estimación sólo usa observaciones válidas; en este caso, es el tamaño muestral, con

m$df.residual

## [1] 7

grados de libertad residuos.

## Diagnóstico

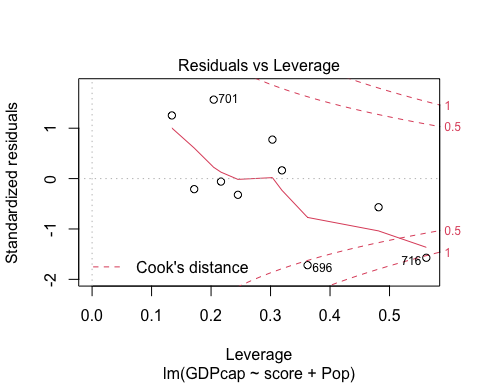
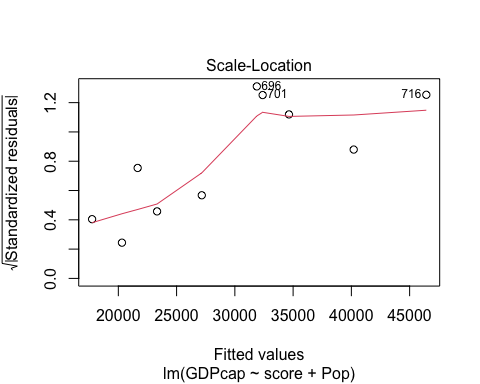
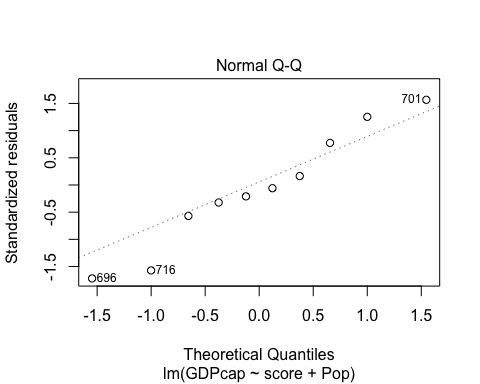
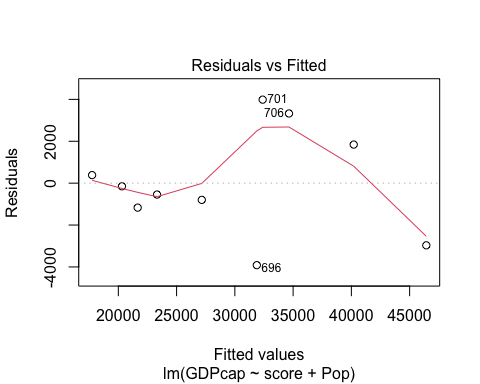
Un diagnóstico típico de una regresión lineal involucra una prueba de

* normalidad;
* heterocedasticidad;
* autocorrelación serial, y
* adecuación funcional

### Diagnóstico visual

Un diagnóstico visual se hace usando la función **plot()** aplicada al objeto de la estimación:

plot(m)



### Diagnóstico formal

En un diagnóstico formal, se prueban hipótesis estadísticas acerca de los errores de la regresión lineal.

La hipótesis nula, en cada caso, es el cumplimiento de los correspondientes supuestos del modelo de regresión lineal clásico (Teorema de Gauss-Markov).

En las pruebas de hipótesis usaremos como nivel de significancia.

#### Prueba de normalidad

La prueba de normalidad se puede realizar usando la prueba de Shapiro-Wilk, implementada con la función **shapiro.test()** en la librería **stats**, cargada por defecto cuando R arranca.

shapiro.test(m$residuals)

##   
## Shapiro-Wilk normality test  
##   
## data: m$residuals  
## W = 0.96408, p-value = 0.8312

Concluimos que los errores son normales (más precisamente, no existe evidencia de falta de normalidad en los residuos).

Las demás pruebas de diagnóstico usan funciones de la librería **lmtest** (que se debe bajar y cargar en la sesión). Si se quiere usar la matriz de varianza-covarianza robusta a heterocedasticidad, entonces conviene también (bajar y) cargar la librería **sandwich**.

Esta librería no se encuentra en la distribución estándar de R, por lo que se debe bajar localmente (en la computadora) y, después, cargar en la sesión de trabajo.

install.packages("lmtest")

Una vez bajada localmente, sólo se deberá cargar la librería en la sesión de trabajo cuando sea necesario.

library(lmtest)

### Prueba de heterocedasticidad

La prueba de Breusch-Pagan se implementa con la función **bptest()**:

bptest(m)

##   
## studentized Breusch-Pagan test  
##   
## data: m  
## BP = 5.3138, df = 2, p-value = 0.07017

No existe evidencia de heterocedasticidad en el modelo.

### Prueba de autocorrelación serial

#### Prueba de Durbin-Watson

dwtest(m)

##   
## Durbin-Watson test  
##   
## data: m  
## DW = 1.7503, p-value = 0.1076  
## alternative hypothesis: true autocorrelation is greater than 0

#### Prueba de Breusch-Godfrey

bgtest(m, order = 6)

##   
## Breusch-Godfrey test for serial correlation of order up to 6  
##   
## data: m  
## LM test = 9.9503, df = 6, p-value = 0.1268

No se detecta evidencia de autocorrelación serial (hasta 6 rezagos) en el modelo.

### Prueba de adecuación funcional

Por simplicidad, se incluye sólamente la prueba RESET de Ramsey.

#### Prueba RESET

En esta prueba se quiere determinar si el modelo necesita términos cuadráticos o cúbicos. La hipótesis nula es que no son necesarios.

La prueba en R se realiza usando ya sea los valores ajustados o las potencias de los regresores.

Prueba con los valores ajustados:

resettest(m, power = 2:3, type = "fitted")

##   
## RESET test  
##   
## data: m  
## RESET = 1.9783, df1 = 2, df2 = 5, p-value = 0.2328

Prueba con los regresores:

resettest(m, power = 2:3, type = "regressor")

##   
## RESET test  
##   
## data: m  
## RESET = 1.8203, df1 = 4, df2 = 3, p-value = 0.3251

No se detecta la necesidad de términos cuadráticos o cúbicos.

detach(swe)