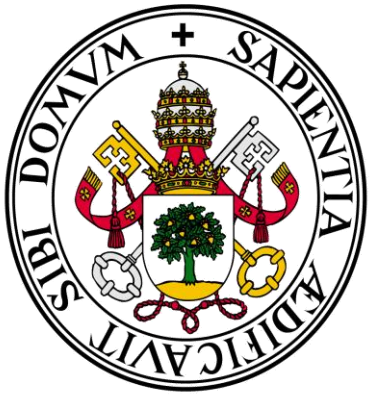


Universidad de Valladolid

Facultad de Ciencias Económicas y Empresariales

Trabajo de Fin de Grado

Grado en Economía

Análisis de regresión con R

Presentado por:

Alberto Abril Arroyo

Tutelado por:

María Mercedes Prieto Alaiz

Valladolid, 20 de Julio de 2018

Resumen

El presente trabajo, basado en el lenguaje de programación de R, pretende demostrar las posibilidades de dicho programa en el marco de un análisis estadístico simple. Sus herramientas, su gran funcionalidad, así como su acceso gratuito son algunas de las numerosas ventajas de este software, las cuales trataremos de analizar a través de un profundo estudio de su funcionamiento. Para ello, y tras una primera parte en la que presentaremos las capacidades básicas de R, los tipos de estructuras de datos y algunas funciones generales, realizaremos un análisis de regresión, estudiando los principales aspectos econométricos acerca de la validación del modelo. Con todo ello pretendemos aprender el funcionamiento del programa e ilustrar con un ejemplo práctico la capacidad de este, para poder realizar estudios estadísticos econométricos y económicos.

Por último, como conclusión general, podemos extraer que el análisis econométrico con R conlleva dificultades relacionadas con la falta de una interfaz gráfica amigable, pero que, a pesar de ello, es una herramienta de gran potencial a la que todo el mundo puede acceder gracias a su licencia de código abierto y beneficiarse de la posibilidad de compartir, con un código común entre usuarios, las funcionalidades generadas.

Abstract

The present assignment, based on the R programming language, aims to demonstrate the possibilities of such program within the framework of a simple statistical analysis. Its tools, its great functionality, as well as its free access are some of the numerous advantages of this software, which we will try to analyze through an in-depth study of its operation. For this, and after a first part in which we will present the basic capacities of R, the types of data structures and the general functions, we will perform a regression analysis, studying the main economic aspects in the validation of the model. With all this, we intend to know the operation of the program and illustrate with a practical example the capacity of it, to be able to carry out statistical, econometric and economic studies.

Finally, as a general conclusion, it can be inferred that the econometric analysis with R involves difficulties related to the lack of a friendly graphical interface; despite this, it is a tool of great potential that everyone can access thanks to its open-source license and benefit from the possibility of sharing, with a common code among users, the functionalities generated.

Palabras clave: lenguaje de programación, licencia de código abierto, análisis de regresión, interfaz gráfica.

Clasificación JEL: C12, C13, C88.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES:

1.INTRODUCCIÓN

R es un lenguaje y entorno de programación que sirve para hacer análisis estadísticos y generar gráficos.

Fue desarrollado inicialmente por Robert Gentleman y Ross Ihaka del Departamento de Estadística de la Universidad de Auckland en 1993, Nueva Zelanda.

Su desarrollo actual es responsabilidad del R Development Core Team, quienes consiguen financiación a través de su propia fundación, pero principalmente de la Free Software Fundation, ya que está integrado en uno de sus proyectos llamado GNU (que a su vez trata de crear un sistema operativo libre).

Al estar integrado en la Free Software Fundation, es un programa que se distribuye con una licencia de código abierto (General Public License), recibiendo por tanto la denominación de software libre.

R es un lenguaje de programación orientado a objetos, que junto con S-Plus son los dos más usados de este tipo. Una consecuencia de la programación orientada a objetos es que nos permite usar herramientas creadas por otros usuarios (que una vez publicadas se agrupan por temas), así como crear nuestras propias herramientas.

Entre las principales ventajas del programa destaca su gran funcionalidad, su fácil instalación y sus gráficos de alta calidad. Por el contrario, su principal desventaja es que al no disponer de ventanas o menús puede resultar menos intuitivo que otros programas de pago con Eviews o Matlab.

Para lo cual han surgido GUIs (Graphipal Users Interfaces) como RComander y RStudio cuya función es la de crear un entorno gráfico, aunque mantiene la esencia de la ventana de comandos.

A lo largo de este trabajo vamos a estudiar las estructuras de datos, las funciones que se pueden elaborar con el programa y por último realizaremos un análisis de regresión básico con R. En el segundo apartado, instalaremos el programa R, la GUIs RStudio y veremos cómo descargar y leer paquetes de

datos con el programa. En el tercer apartado, veremos las principales estructuras de datos en R (vectores, data frames, factores y matrices). En el cuarto apartado, analizaremos algunas funciones que tiene implementado R y crearemos algún ejemplo de las nuestras propias. El apartado cinco trata sobre el análisis descriptivo de datos (tanto gráfico como estadístico). Y por último, el apartado seis está dedicado a un análisis de regresión básico como medio para hacer un estudio de las herramientas que tiene el programa para realizar estudios econométricos

De entre todas las fuentes destaca por su importancia Marqués, F. (2017): *R en profundidad* cuya lectura es esencial para la elaboración de los primeros cinco apartados y Heiss, F. (2016): *Using R for Introductory Econometrics* para la elaboración de la parte de regresión.

2. INSTALACIÓN Y FORMAS DE TRABAJAR EN R

A lo largo de este trabajo, vamos a utilizar RStudio, un entorno de desarrollo integrado (IDE) para R que funciona con la versión estándar de R disponible en CRAN. Al igual que R, RStudio es software libre y lo que pretende es crear un entorno que sea tan sencillo e intuitivo como sea posible. Para instalar este software, es necesario descargar primero el software R:

- Para lo cual accedemos a la página web de R: http://cran.r-project.org y clicamos en la opción Download R, seleccionando el sistema operativo que cada usuario tenga en su ordenador.
- Seleccionamos el subdirectorio base, lugar en el que están los principales ficheros de R. Elegimos Download R 3.5.0 for Windows que nos permite guardar el fichero ejecutable o directamente ejecutarle.
- Una vez ejecutado el programa en nuestro equipo, seleccionamos el idioma de instalación y seguimos las indicaciones del asistente de instalación de R, aceptando los términos de la licencia de R.

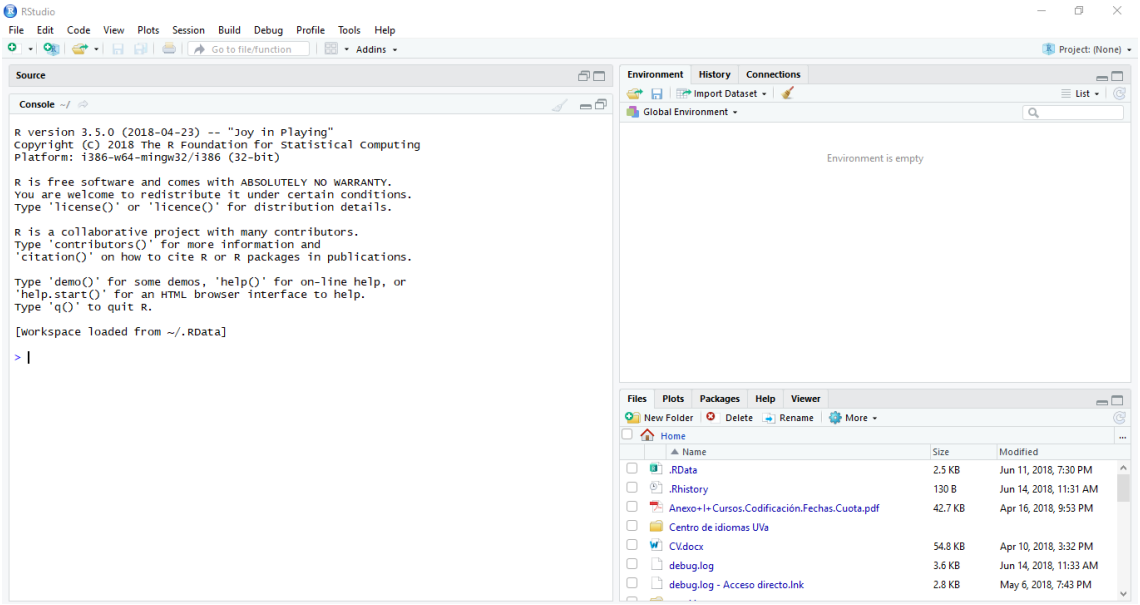
Una vez instalado el programa principal podemos descargar RStudio:

- Para lo cual accedemos a la página web de RStudio: https://www.rstudio.com y clicamos en la opción Download RStudio.

- Seleccionamos la versión de licencia de código abierto, eligiendo el sistema operativo que cada usuario tenga en su ordenador.
- Elegimos el subdirectorio de nuestro ordenador donde guardamos el archivo ejecutable. Y ejecutamos el programa.
- Una vez ejecutado, seguimos las indicaciones del asistente de instalación de RStudio hasta finalizar la instalación.

Como se puede ver en la ilustración 1, al abrir el programa nos aparece una pantalla dividida en 3 ventanas:

Ilustración 1: Ventanas de RStudio



- La ventana izquierda nos muestra la consola (o ventana de comandos), donde ejecutaremos las sentencias y el programa nos devolverá los resultados del análisis.
- La ventana superior derecha nos muestra por un lado el historial de comandos que hemos ejecutado. Y por otro lado los objetos que tenemos definidos, en la subventana “Environment”.
- Y por último, la parte inferior derecha nos muestra diferentes subventanas: “Files” sirve para ver el directorio de trabajo; “Plots” muestra los gráficos que realizaremos con R; “Packages” muestra las librerías que tenemos instaladas en nuestro ordenador y que, para

usarlas, debemos cargar en la memoria, y por último también contamos con una sección de ayuda donde podremos consultar las dudas que vayan surgiendo sobre el uso de R (subventana “Help”).

Hay dos formas de trabajar con R: la primera consiste escribir directamente las sentencias en la consola, de esta forma sólo se puede ejecutar una orden. La segunda forma de trabajar es editando un fichero de texto (script file) con las sentencias que queremos ejecutar a la vez. Para trabajar de esta segunda forma, abrimos el script, seleccionamos las sentencias que queremos ejecutar, las copiamos en nuestra consola y pulsando la tecla *Enter* las ejecutamos todas a la vez.

Para finalizar este apartado vamos a ver cómo se puede descargar y cargar un nuevo paquete para añadir más funcionalidades de las que vienen por defecto en R. Primeramente vamos a cargar un paquete ya instalado.

- Accedemos a la solapa Packages de la ventana inferior derecha del programa.
- Veremos una lista de librerías ya instaladas en nuestro ordenador.
- Marcamos la(s) librería(s) o paquete(s) que queremos cargar e inmediatamente veremos cómo se ejecutan en la consola.

Si el paquete no está en esa lista deberemos instalarlo mediante el siguiente procedimiento:

- Accedemos a la solapa Packages de la ventana inferior derecha del programa.
- Seleccionamos la solapa “Install”
- En el cuadro de diálogo que nos aparece introducimos el nombre del paquete que queremos descargar y clicamos en instalar.
- Por último, cargamos el paquete que acabamos de instalar.

Por ejemplo, si queremos instalar y cargar el paquete *foreign* podemos hacerlo mediante las siguientes sentencias:

```
install.packages("foreign")

library("foreign", lib.loc="~/R/win-library/3.5")
```

3. ESTRUCTURAS DE DATOS EN R

3.1. Vectores

Para crear un vector utilizamos la siguiente sentencia rentapc=c(12000, 24000, 36000, …). Donde "rentapc" es el nombre que queremos dar al vector creado y 12000, 24000 y 36000 son escalares, pero pueden ser otros vectores ya creados, caracteres, en definitiva cualquier objeto.

Podemos estar interesados en designar una columna (variable) /fila (observación) como vector independiente en un data frame. Esto sirve por ejemplo para cuando queremos calcular el producto de dos variables y se realiza mediante la sentencia: v=dataframe$v1. Donde v representa el nombre que le queremos dar a esa nueva variable y v1 el nombre que tenía la fila/columna

Existen diferentes tipos de vectores:

- Vectores numéricos: el definido en el caso anterior.
- Vectores lógicos: expresan si las observaciones cumplen o no una condición por ejemplo ser mayor que 3. Si la cumple el vector lo refleja mediante el término TRUE y si no la cumple lo refleja mediante el término FALSE:

  ricos<-rentapc>24000

  Cuando introdujésemos la variable ricos y nos devolvería un vector de TRUE y FALSE en vez del valor de la renta per cápita:

  summary(ricos)

  Mode FALSE TRUE

  logical 145 44

- Vectores de caracteres: los elementos del vector son letras o palabras que se escriben entre comillas; por ejemplo si una observación muestral está casada o soltera: estado<-c("casada","soltera","soltera")

Otro tipo de vectores son las secuencias regulares: vectores numéricos cuyos elementos están ordenados de manera determinada. Ejemplo: secuencia de números naturales del uno al diez:

s1=seq(1,10) donde s1 es el nombre que se le daría al objeto generado.

Si quisiésemos generar la misma secuencia pero cuyos elementos estuvieran diferenciados en 2 unidades en vez de una:

s2=seq(1,10,by=2) ó simplemente s2=seq(1,10,2). Como resultado obtendríamos:

s2

[1] 1 3 5 7 9

Ahora bien, podemos seleccionar unas determinadas observaciones que nos interesa evaluar dentro de un marco de datos concreto, para lo cual añadimos al final una sentencia: el número de la primera, dos puntos y el de la última observación que nos interesan y todo ello entre corchetes. Ejemplo:

estado [2:3]

En este caso, podríamos visualizar si de la observación 2 a la 3 están casadas o solteras.

También nos puede interesar seleccionar un solo elemento de un vector:

estado[2]

En este caso podríamos ver el resultado del segundo elemento del vector estado (pudiéndolo definir como un nuevo vector si nos interesa):
estado1<-estado[2]

Por último, si no interesa operar con vectores: podemos usar los operadores suma, resta, producto, cociente y potencia de los vectores A, B, de la siguiente manera: v1=A+B, v2=A-B, v3=A*B, v4=A/B, v5=A^B.

3.2 Matrices

Las matrices son objetos que están formados por vectores de la misma longitud y del mismo tipo.

Para la creación de la matriz M que está formada por los elementos v1, v2, v3, v4… y tiene una dimensión de r filas y c columnas:

M=matrix(c(v1,v2,v3,v4,…),nrow=r,ncol=c)

Es importante saber que los elementos v1, v2, etc se van ordenando por columnas no por filas, de manera que el tercer elemento del vector será el primer elemento de la tercera fila de la matriz. Si queremos cambiar esto, y que se vayan ordenando por filas de manera que el tercer elemento del vector fuera el tercer elemento de la primera fila, añadiríamos *byrow=TRUE* al final de la sentencia:

M=matrix(c(v1,v2,v3,v4,…),nrow=r,ncol=c,byrow=TRUE)

También, es frecuente que necesitemos seleccionar algún objeto en particular de una matriz:

- Podemos estar interesados en seleccionar un elemento de la matriz, por ejemplo de la matriz A el elemento que se sitúa en la segunda fila, tercera columna: A[2,3]
- Podemos estar interesados en seleccionar una fila o columna entera. Por ejemplo seleccionar la fila dos de la matriz A: A[2,]. O bien seleccionar la columna tres de la matriz A: A[,3]
- Por último podemos estar interesados en seleccionar una submatriz dentro de la matriz A para poder trabajar con ella a parte: submatrizA<-A[2:4,1:3]. En este caso hemos seleccionado de la fila dos a la cuatro y de la columna uno a la tres por lo que nos quedaría una submatriz cuadrada de tres filas y tres columnas.

Por último, los operadores básicos de suma y resta funcionan de manera idéntica que con los vectores u otros objetos, mientras que el operador producto de las matrices A, B, sería: A%*%B. Por otra parte, la operación cociente de matrices no se encuentra definida por lo que es interesante

conocer el operador matriz inversa: *solve*. Por ejemplo, para hallar la inversa de la matriz A: inversaA<-solve(A).

Para hallar la transpuesta de A la sentencia sería: transpuesta<-t(A)

3.3 Data.frames

Un data.frame es una colección de vectores que pueden ser de diferente tipo pero que necesariamente deben de tener la misma longitud. Es frecuente que todos sus vectores sean vectores columna numéricos de igual longitud (lo que en otros programas podríamos considerar como variable, cada componente/columna representa una variable, y todas de igual longitud, mismo número de filas, donde cada fila representaría una observación). Sirve para trabajar con todos los datos a la vez.

Para la creación de un nuevo marco de datos podemos usar directamente la siguiente sentencia:

datos=data.frame(estado, rentapc)

En este caso hemos usado las variables “rentapc” y “estado” ya definidas.

El caso de importar datos ya existentes es mucho más frecuente que el anterior y consiste en introducir los datos indirectamente desde otro programa como puede ser EXCEL, SPSS, etc. Para abrir los datos nos vamos a la pestaña de “Environment” en la sección superior derecha del programa, y clicamos en la opción de “Import Dataset”.

Una vez abiertos los datos los podremos visualizar en la sección superior izquierda de la pantalla, pero para poder empezar a trabajar con ellos deberemos crear un objeto que los identifique como un data.frame a través de la siguiente sentencia:

datos=data.frame(“nombre del archivo q acabamos de abrir”)

R permite guardar la información de la ventana de comandos (consola) en un archivo de formato RData; para lo cual debemos de pulsar el icono de guardar

en el panel superior izquierda de la pantalla y los exportará al directorio que le indiquemos.

3.4 Listas

Objeto formado a su vez por un conjunto de objetos ordenados de una forma determinada. A cada objeto que forma una lista se le denomina componente. Dichas componentes pueden ser caracteres, vectores, matrices, etc; en definitiva no es necesario que las componentes sean objetos del mismo tipo. Sentencias:

- nombrelista=list(componente1,componente2,componente3,…)
- nombrelista[[2]] sirve para que R te seleccione la segunda componente de la lista.
- nombrelista[[2]][4] en el caso de que la componentes dos fuera un vector, R nos devolvería el cuarto elemento del vector.

3.5. Factores:

Es un vector de caracteres. Estos caracteres son valores que puede tomar una variable categórica como por ejemplo las “escalas likert”, que sirven para evaluar la calidad de un restaurante, el grado de acuerdo o desacuerdo en una encuesta, etc. Cada factor tiene tantos niveles como valores distintos pueda tomar la variable categórica a la que representa. Por ejemplo, las escalas likert suelen tener cinco niveles, donde las puntuaciones cercanas a uno representan una nota negativa y cercanas a 5 una nota positiva.

Creación de un factor: es importante tener en cuenta que un factor se crea a partir de un vector ya definido por ejemplo los vectores v1 y v2:

```
v1<-c(1,1,3,1,2,5,5,4)

Fv1<-factor(v1)
```

Si nos interesa centrarnos en las observaciones que han dado cierta puntuación (por ejemplo la puntuación 1) excluyendo al resto podemos usar la sentencia:

factor(1:5,1) ó bien factor(1:5,exclude=2:5)

4. FUNCIONES

Debemos distinguir dos tipos de funciones: las que incorpora R por defecto (funciones de variable entera, variable real, funciones vectoriales, funciones matriciales, funciones de información, de conversión, funciones de cadena y funciones de selección y manipulación) y funciones que podemos definir nosotros mediante siguiente procedimiento:

Abrir el editor de R que se encuentra en el menú file/new file/R Script. Una vez en el editor escribimos la siguiente sintaxis para definir nuestra función:

"Nombre de la funcion"=function(argumento1,argumento2,argumento3,…)

{Sentencias+return(object)}

Para finalizar pulsamos el botón Run para ejecutar la sintaxis y así poder utilizar la función creada cuando se necesite.

Por ejemplo vamos a crear la función varianza de la variable A (que debe estar definida previamente como objeto):

varianzaA=function(A){n=length(A)+v=sum((A-(sum(A)/n))^2/n+return(v)}

Otro ejemplo podría ser el de la media de los valores de la variable A:

mediaA=function(A){n=length(A)+u=sum(A)/n+return(u)}

Por último un ejemplo un poco más complejo podría ser la función que resuelve cualquier ecuación de segundo grado donde los parámetros que multiplican a la variable son a, b, c:

Soluciónecuación=function(a,b,c)

```
{v=(-b+sqrt(b^2-4*a*c))/2*a+u=(-b-sqrt(b^2-4*a*c))/2*a+return(c(u,v))}
```

5. ANÁLISIS DESCRIPTIVO BÁSICO

Este apartado está dedicado a mostrar algunas de las rutinas de R para resumir, desde el punto de vista estadístico, la información de las variables. Para ello vamos a utilizar información de 189 países sobre la esperanza de vida, duración esperada de la escolarización, duración media de la escolarización, desarrollo y renta per cápita.

Lo primero que debemos hacer es introducir los datos en R, para lo cual accedemos a la solapa “Environment” en la ventana superior derecha y clicamos en “Import Dataset”. Seleccionamos el directorio de nuestro ordenador donde se encuentran los datos y seleccionamos la opción importar.

Una vez que los datos están introducidos, para poder trabajar con ellos, vamos a identificarlos en conjunto como un data frame y cada columna como un vector:

```
datos<-data.frame(pr10)

attach(datos)
```

La función *summary* proporciona el valor mínimo, máximo, el valor medio, la mediana y el primer y tercer cuartil de cada variable. Por ejemplo, si quisiéramos obtener dichos valores de la variable riqueza:

```
summary(riqueza)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.5807  3.5601 10.6049 16.9349 22.7621 123.1244 
```

Además de estos resúmenes, hay otros estadísticos de gran importancia que se pueden obtener por medio de las siguientes funciones que exponemos a continuación:

- Varianza: es una medida de la dispersión de los datos, siempre es positiva y cuanto mayor sea, más dispersos están los datos:

```
var(riqueza)

[1] 352.5296
```

- Desviación típica: es la raíz cuadrada de la varianza y está expresada en las mismas unidades de medida de la variable.

```
sd(riqueza)

[1] 18.77577
```

- Error estándar: sirve para medir la desviación típica de la distribución de la media muestral. La siguiente función que hemos construido nos proporciona el error standard:

```
SE=function(A){n=length(A)+v=sd(A)/n+return(v)}
```

- Una vez tenemos calculada la función SE podemos calcular el error estándar de la variable riqueza:

```
SE(riqueza)

[1] 0.09934271
```

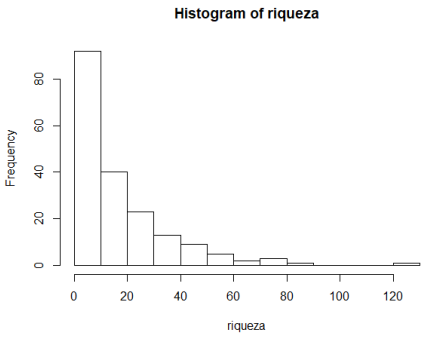
- Otra medida de dispersión que se usa para comparar el grado de dispersión relativo es el coeficiente de variación. La siguiente función, que también hemos construido, nos devuelve el coeficiente de variación:

```
coeficiente_var=function(A){cv=sd(A)/mean(A)+return(cv)}
```

Por último, vamos a ver algunos métodos gráficos para resumir la información de nuestras variables:

- Histograma de frecuencias: nos permite hacernos una idea visual de la forma de su distribución y analizar, por ejemplo, si la distribución se parece al aspecto de una distribución normal, su simetría, etc.. Su sentencia es *hist* y el resultado se muestra en la Ilustración 2:

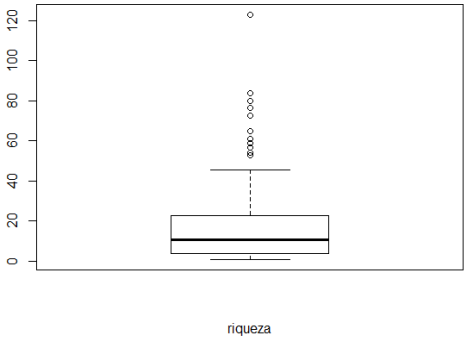
Ilustración 2: Histograma variable riqueza



- Gráfico de caja y bigotes: permite ver si hay valores atípicos y ver la simetría de la distribución. Su sentencia es *boxplot* y el resultado se muestra en la ilustración 3:

  boxplot(riqueza,xlab="riqueza")

  Ilustración 3: Diagrama de caja de la variable riqueza

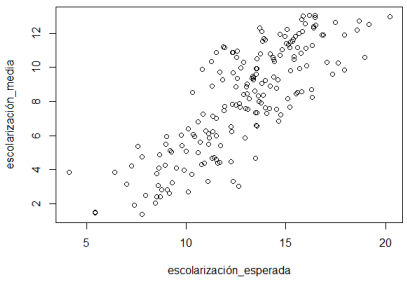


- Para realizar un gráfico de dispersión entre las variables “escolarización esperada” y “escolarización media” escribimos la siguiente sentencia:

  plot(escolarización_esperada,escolarización_media)

  El resultado se puede ver a continuación en la ilustración 4:

Ilustración 4: Diagrama de dispersión



Se puede observar en el gráfico que la relación entre las dos variables es lineal positiva: a mayor escolarización esperada, mayor escolarización media.

6. REGRESIÓN

En este apartado vamos a tratar cómo estimar un modelo de regresión por mínimos cuadrados ordinarios, así como la creación de variables ficticias, cómo realizar contrastes de hipótesis y diferentes formas de predicción usando el modelo. Además, veremos algunos procedimientos para detectar si se cumplen o no algunas hipótesis clásicas (casos de normalidad, homocedasticidad, incorrelación entre perturbaciones y regresores exógenos).

6.1. Estimación:

Vamos a trabajar con los mismos datos del apartado anterior, estimando un modelo que trate de explicar la esperanza de vida en función del desarrollo, la escolarización esperada, la escolarización media y la riqueza per cápita.

El comando para realizar la regresión es el siguiente:

lm(variable.dependiente ~variable.independiente1+variable.independiente2…)

En nuestro caso:

lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza+desarrollo)

El resultado incluiría sólo los coeficientes de regresión estimados, por lo que si queremos obtener más información podemos usar la función *summary*:

```
summary(lm(esperanza_vida~escolarización_esperada+escolarización_media+
riqueza+desarrollo))
```

El resultado aparece en la Tabla 1:

Tabla 1: Resultado de la regresión

```
Call:

lm(formula = esperanza_vida ~ escolarización_esperada + escolarización_media +
    riqueza + desarrollo)

Residuals:
Min                      1Q          Median      3Q          Max
-165.264                 -24.871     0.4412      30.589      96.709

Coefficients:
                         Estimate    Std. Error  t value     Pr(>|t|)
(Intercept)              5.127.620   206.839     24.790      < 2e-16     ***
escolarización_esperada  0.99979     0.22869     4.372       2.06e-05    ***
escolarización_media     0.35809     0.20020     1.789       0.07533     .
riqueza                  0.06681     0.02355     2.837       0.00506     **
desarrollo               531.093     116.537     4.557       9.45e-06    ***
---
Signif. codes:
 0 '***' 0.001 '**' 0.01 '*' 0.05  '.' 0.1 ' ' 1

Residual standard error: 4.612 on 183 degrees of freedom
Multiple R-squared:  0.7038,    Adjusted R-squared:  0.6973
F-statistic: 108.7 on 4 and 183 DF,  p-value: < 2.2e-16
```

La salida muestra información de los residuos de la regresión (su valor máximo y mínimo, su media y su primer y tercer cuartil); el valor de los coeficientes de la regresión, su error estándar, el valor del estadístico t (que es el valor muestral del estadístico del contraste de la significación individual de cada una de las variables en la regresión) y su p-valor asociado; por último muestra el error estándar de los residuos, la bondad del ajuste medido por el r-cuadrado y el r-cuadrado ajustado y el valor del estadístico F para contrastar la significación conjunta de los regresores, junto con su p-valor y diferentes características de la distribución dicho estadístico de contraste.

6.2. Creación de dummies

En este apartado crearemos nuestra propia variable dummy para incluirla en la regresión. En el apartado anterior ya realizamos el análisis incluyendo la variable desarrollo, pero ahora vamos a definir una nueva variable (desarrollo1). Esta variable va a estar definida en función de si cada país tiene más de 10.60489 miles de dólares de renta per cápita (valor de la mediana de la variable riqueza):

```
desarrollo1<-as.numeric(riqueza>10.60489)

table(desarrollo1)
```

El resultado se muestra a continuación en la tabla 2:

Tabla 2: Dummy variable "desarrollo1"

desarrollo1	
0	1
94	95

Como se puede apreciar en la salida, 94 países no cumplen con la condición de tener más de 10.60489 miles de dólares de renta per cápita y 95 sí que la cumplen. Para estos últimos la variable desarrollo1 tomará un valor de 1 y para los 94 restantes tomará un valor de 0.

A continuación, realizamos una regresión incluyendo la variable desarrollo1:

```
lm(esperanza_vida~desarrollo1+escolarización_esperada+escolarización_media+riqueza)
```

El resultado se muestra a continuación en la tabla 3:

Tabla 3: Coeficientes de la regresión

Coefficients:				
(Intercept)	desarrollo1	escolarización_esperada	escolarización_media	riqueza
47.97378	1.32213	1.26372	0.59838	0.08007

También se hubiera podido estimar de forma idéntica considerando desarrollo1 como una variable lógica en vez de numérica:

```
desarrollo1<-as.logical(riqueza>10.60489)
```

Ahora bien, si la variable desarrollo1 hubiese tenido más de dos categorías estaríamos hablando de una variable categórica. La mejor forma de trabajar con ellas es como si fueran factores.

Vamos a generar una variable categórica (desarrollo11) a partir de la variable riqueza, agrupando las observaciones de esta última variable en cuatro grupos[1]:

```
desarrollo11=car::recode(riqueza,"lo:3.6906=1;3.6906:10.6671=2;10.6671:22.8006=3;22.8006:hi=4")
```

A continuación, identificamos esta nueva variable como un factor:

```
desarrollo11<-as.factor(desarrollo11)
```

Por último, vamos a realizar una regresión que trate de explicar la esperanza de vida en función de esta nueva variable:

```
lm(esperanza_vida~desarrollo11)
```

El resultado se muestra a continuación en la tabla 4:

[1] Los puntos de corte elegidos son los valores del primer cuartil, la mediana y el tercer cuartil de la variable riqueza.

Tabla 4: Regresión con variable categórica

```
Call:
lm(formula = esperanza_vida ~ desarrollo11)

Coefficients:
  (Intercept)  desarrollo112  desarrollo113  desarrollo114
       61.623          8.214         11.593         18.119
```

6.3. Contrastes de hipótesis

En este apartado vamos a tratar los principales contrastes sobre los parámetros de una regresión.

Un tipo de hipótesis que frecuentemente interesa contrastar es que algún parámetro es igual a un escalar (este escalar frecuentemente es cero). La veracidad de esta hipótesis se puede testar mediante el estadístico t, que bajo la hipótesis nula se distribuye como una t de Student de (n-k-1) grados de libertad, siendo n el número de observaciones y k el número de regresores de la regresión:

$$t = \frac{\widehat{\beta}_j - 0}{SE(\widehat{\beta}_j)} \xrightarrow{H_o: \beta_j = 0} t_{(n-k-1)}$$

La hipótesis alternativa puede ser que el valor del parámetro es mayor, menor o distinto del de la hipótesis nula. Dependiendo de cuál de estas opciones se elija, el test será de una o de dos colas. Nosotros supondremos que la hipótesis alternativa es $H_1: \beta_j \neq 0$, es decir el test será de dos colas.

En R para realizar este contraste se necesita instalar y cargar el paquete *lmtest* y posteriormente usar el comando *coeftest:*

```
coeftest(lm(esperanza_vida~riqueza+escolarización_esperada+escolarización_media+desarrollo))
```

El resultado se muestra a continuación en la tabla 5:

Tabla 5: T test

```
t test of coefficients:

                          Estimate Std. Error t value  Pr(>|t|)
(Intercept)             51.276199   2.068387 24.7904 < 2.2e-16 ***
riqueza                  0.066812   0.023549  2.8372  0.005065 **
escolarización_esperada  0.999792   0.228686  4.3719 2.063e-05 ***
escolarización_media     0.358093   0.200205  1.7886  0.075328 .
desarrollo               5.310926   1.165373  4.5573 9.451e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

No obstante, hay contrastes cuya hipótesis nula involucra a más de un parámetro de la regresión, por ejemplo: (H_0: $\beta_1 = \beta_2 = 0$) ó (H_0: $\beta_j = 0$).

El estadístico que permite contrastar estas restricciones de la regresión es el estadístico F, que bajo la hipótesis nula se distribuye como una F de Snedecor con q (número de parámetros iguales a cero) grados de libertad en el numerador y (n-k-1) grados de libertad en el denominador:

$$F = \frac{SCRr - SCRsr}{SCRsr} * \frac{n-k-1}{q} \xrightarrow{Ho} F^{q}_{(n-k-1)}$$

Siendo SCR_r la suma de cuadrados residual del modelo restringido (considerando cierta la hipótesis nula) y SCR_{sr} la suma de cuadrados residual del modelo sin restringir.

En R para realizar este tipo de contrastes, se necesita instalar y cargar el paquete *car* y posteriormente usar el comando *linear.hypothesis.* En nuestro caso vamos a testar la hipótesis de que en los coeficientes de las variables escolarización media y esperada sean cero:

```
>myHO<-c("escolarización_esperada","escolarización_media")
>linearHypothesis(lm(esperanza_vida~riqueza+escolarización_esperada+escolarización_media+desarrollo),myHO)
```

El resultado se muestra a continuación en la tabla 6:

Tabla 6: F test

```
Linear hypothesis test

Hypothesis:
escolarización_esperada = 0
escolarización_media = 0

Model 1: restricted model
Model 2: esperanza_vida ~ riqueza + escolarización_esperada + escolarización_media +
    desarrollo

  Res.Df    RSS   Df   Sum of Sq       F          Pr(>F)
1    185   4792.3
2    183   3892.4  2    899.92        21.155      5.433e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

6.4. Predicción

La predicción es uno de los objetivos fundamentales del análisis de regresión y en este apartado, vamos a pronosticar el valor de la variable dependiente de nuestro modelo, dados ciertos valores de los regresores.

Para ello usaremos las relaciones entre las variables especificadas en el apartado de la estimación:

```
regre1=lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza+desarrollo)
```

Ahora introducimos los valores de los regresores de la nueva observación en el programa, identificándolos como un dataframe:

```
pred_dataframe=data.frame(desarrollo=0,escolarización_esperada=5.4,escolarización_media=1.5,riqueza=1.908)
```

A continuación, utilizamos el comando *predict* para sustituir los valores de las variables en la recta de regresión, y así obtener el valor predicho de la variable endógena:

```
predict(regre1,pred_dataframe)
```

```
57.33969
```

Si nos interesa obtener un intervalo de confianza (al 95%) para este último valor añadiríamos *interval="confidence"* a la sentencia. Y si también queremos determinar el error estándar de la predicción añadiríamos *se.fit=TRUE*.

```
predict(regre1,pred_dataframe,interval="confidence",se.fit=TRUE)
```

El resultado se muestra a continuación en la tabla 7:

Tabla 7: Intervalo de confianza y error estándar de la predicción

fit	lwr	upr	SE	df	residual.scale
57.33969	55.1626	59.51678	1.103435	183	4.611946

6.5. Normalidad

En este apartado vamos a ver la validación de la hipótesis clásica de normalidad. Dicha hipótesis consiste en que las perturbaciones de la regresión deben distribuirse como una normal. Para testar la normalidad se pueden hacer pruebas gráficas o contrastes de hipótesis basadas en los residuos, dado que las perturbaciones no son observables.

En los contrastes de hipótesis que vamos a ver en este apartado, la hipótesis nula consiste en que las perturbaciones se distribuyen como una normal frente a la hipótesis alternativa de que las perturbaciones no se distribuyen como una normal.

Las pruebas estadísticas que vamos a utilizar para detectar la falta de normalidad son: Shapiro-Wilks y Kolmogorov-Smirnov. Las sentencias para realizar dichos test son *shapiro.test* y *ks.test* respectivamente:

```
shapiro.test(resid(regre1))
```

```
ks.test(resid(regre1),"pnorm")
```

El resultado se muestra a continuación en la tabla 8:

Tabla 8: Contrastes de hipótesis normalidad

	Estadístico	P-valor
Shapiro-Wilks	W = 0.97303	0.001071
Kolmogorov-Smirnov	D = 0.33427	< 2.2e-16

Como podemos ver en la tabla anterior, con ambos contrastes, se rechaza la hipótesis de normalidad por lo que, la distribución de las perturbaciones del modelo no es una normal y, por tanto, la de los estimadores por mínimos cuadrados ordinarios (MCO) tampoco. No obstante, hay que señalar que la distribución asintótica de los estimadores por MCO es una distribución normal, lo que implica que las distribuciones habituales de los estadísticos de contraste sólo tienen validez asintótica.

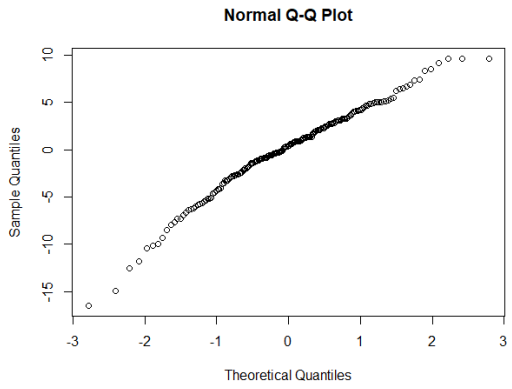
Por otro lado, también se puede detectar la falta de normalidad mediante pruebas gráficas. La prueba que vamos a utilizar es el gráfico cuantil (q-q plot), aunque también hay otras complementarias, como el histograma y el diagrama de caja de los residuos.

Para el q-q plot utilizaremos la siguiente sentencia:

```
qqnorm(resid(regre1))
```

El resultado se muestra a continuación en la ilustración 5:

Ilustración 5: Gráfico cuantil



Como se puede apreciar en el gráfico, si trazásemos una línea de cuarenta y cinco grados en el primer cuadrante, las observaciones no se ajustarían a la línea (quedaría por encima).

6.6. Heterocedasticidad

En este apartado vamos a estudiar cómo validar la hipótesis clásica de igualdad de varianzas de las perturbaciones. Para ello nos vamos a servir del test de White que es una prueba estadística que trata de buscar alguna relación entre las varianzas de las perturbaciones y los regresores. Si se da esta relación, deduciríamos que la varianza de las perturbaciones no sería constante.

La hipótesis nula en el test de White es la de homocedasticidad (igualdad de varianza entre las perturbaciones) y la alternativa es heterocedasticidad (varianzas distintas).

El procedimiento según Cavero, J. (2018), es el siguiente:

- Se estima por MCO los residuos del modelo de regresión original.
- Se realiza una regresión auxiliar donde se estiman estos residuos al cuadrado frente a las variables explicativas del modelo original, las variables explicativas del modelo original al cuadrado y los productos cruzados dos a dos de los regresores de la regresión original.
- Se obtiene el r-cuadrado de esta regresión auxiliar y se multiplica su valor por el número de observaciones de la muestra. Este producto sigue una distribución asintótica chi-cuadrado, de grados de libertad igual al número de regresores de la regresión auxiliar.
- Por último, vemos si el valor del estadístico cae en la zona de aceptación o en la de rechazo para el nivel de confianza establecido.

Para llevar a cabo este procedimiento en R, es necesario instalar y cargar los paquetes *het.test* y *vars*.

Una vez cargados estos paquetes identificamos las variables de nuestra regresión original como un data frame[2]:

```
dataset=data.frame(esperanza_vida,riqueza,escolarización_esperada,escolarización_media)
```

A continuación, indicamos que la regresión es un modelo de vectores autoregresivos (modelo de ecuaciones simultaneas), esto quiere decir que se producen interacciones simultaneas entre distintas variables (en una ecuación los residuos están como variable explicativa y en otra como variable endógena):

```
regre11<-VAR(dataset)
```

Para finalizar, obtenemos el estadístico usando el comando *whites.htest*:

```
whites.htest(regre11)
```

Los resultados se muestran en la tabla 9, para el contraste que no contiene términos cruzados:

Tabla 9: Resultado test de White

```
White's Test for Heteroskedasticity:
====================================
No Cross Terms
H0: Homoskedasticity
H1: Heteroskedasticity
Test Statistic:
188.5585

Degrees of Freedom:
80

P-value:
0.0000
```

Podemos comprobar que el modelo es heterocedástico, por lo que los estimadores de los parámetros por MCO no son óptimos, la matriz de varianzas

[2] Nótese que no incluimos la variable dicotómica (desarrollo) en la regresión puesto que no tiene sentido analizar la dispersión de sus datos.

y covarianzas de los estimadores ya no coincide con la de MCO y las distribuciones para hacer los contrastes no son válidas.

No obstante, si consideramos la utilización de una estimación de la matriz de varianzas y covarianzas robusta (y por lo tanto errores estándar robustos) los estadísticos se pueden ajustar asintóticamente, permitiendo usar las pruebas t, F para muestras grandes (Rosales, 2010).

Para poder aplicar este procedimiento en R es necesario instalar y cargar el paquete *car.* Una vez cargado el paquete, para calcular la estimación robusta de la matriz de varianzas y covarianzas utilizamos el comando *hccm* (heteroscedasticity-corrected covariance matrices):

```
hccm(lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza))
```

El resultado se muestra a continuación en la tabla 10:

Tabla 10: Matriz de varianzas y covarianzas robusta

	(Intercept)	e.esperada	e.media	riqueza
(Intercept)	4.45238170	-0.398021903	0.0860156019	0.0188458291
e.esperada	-0.39802190	0.047977297	-0.0250172821	-0.0018101280
e.media	0.08601560	-0.025017282	0.0298234121	-0.0005624932
riqueza	0.01884583	-0.001810128	-0.0005624932	0.0005042912

Y por último, para ver los resultados de los contrastes de significación correctos y los errores estándar robustos usamos el comando *coeftest,* pero indicando al final de la sentencia que la matriz de varianzas y covarianzas es la robusta:

```
coeftest(lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza),vcov=hccm)
```

El resultado se muestra a continuación en la tabla 11:

Tabla 11:Contrastes de significación individual válidos

```
t test of coefficients:

                          Estimate Std. Error t value  Pr(>|t|)
(Intercept)              46.670461   2.110067 22.1180 < 2.2e-16 ***
escolarización_esperada   1.376540   0.219037  6.2845 2.321e-09 ***
escolarización_media      0.636211   0.172695  3.6840 0.0003018 ***
riqueza                   0.090761   0.022456  4.0417 7.793e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

6.7. Autocorrelación

En este apartado vamos a tratar de detectar si se cumple o no la hipótesis clásica de incorrelación entre las perturbaciones. Bajo esta hipótesis la matriz de varianzas y covarianzas de las perturbaciones debería tener todos los elementos fuera de la diagonal principal iguales a cero. Para ello vamos a utilizar información de una determinada región comunitaria sobre el consumo de gasolina per cápita, el PIB per cápita y el precio de la gasolina. Los datos son trimestrales y van desde el año 1988 hasta el año 2012.

Existen diferentes contrastes para testar si se cumple o no esta hipótesis. El más usado es el contraste de Durbin-Watson, cuya hipótesis nula plantea la ausencia de autocorrelación frente a la alternativa de existencia de autocorrelación según un modelo autorregresivo de orden 1:

AR(1): $\varepsilon_t = \rho\varepsilon_{t-1} + u_t$

H_0: $\rho = 0 \rightarrow$incorrelación

H_1: $\rho \neq 0 \rightarrow$AR(1)

El estadístico de Durbin Watson es $d \approx 2(1-\hat{p})$ y aceptamos la hipótesis nula cuando $\hat{p}$ sea próximo a cero y d próximo a dos.

Para aplicar este contraste en R lo primero es introducir los datos, para lo cual accedemos a la solapa “Environment” en la ventana superior derecha y clicamos en “Import Dataset”. Seleccionamos el directorio de nuestro ordenador donde se encuentran los datos y seleccionamos la opción importar.

Una vez que los datos están introducidos, para poder trabajar con ellos, vamos a identificarlos en conjunto como un data frame y cada columna como un vector:

```
datos<-data.frame(pr9)

attach(datos)
```

A continuación, especificamos un modelo que trate de explicar el consumo de gasolina en función del PIB per cápita y del precio de la gasolina:

```
reg<-lm(CONSUMO~PIBPC+PRECIO)
```

Por último, para conocer el valor del estadístico de Durbin, es necesario cargar el paquete *car*, una vez cargado usamos el comando *durbinWatsonTest:*

```
durbinWatsonTest(reg)
```

El resultado se muestra a continuación en la tabla 12:

Tabla 12: Durbin Watson test

lag	Autocorrelation	D-W Statistic	p-value
1	0.4677095	0.6863876	0
Alternative hypothesis: rho != 0			

El problema principal de este contraste es que la hipótesis alternativa sólo plantea un caso de autocorrelación según un esquema autorregresivo de orden uno, por lo que, si sus perturbaciones siguen un modelo de medias móviles o si el orden del retardo es mayor que uno, no lo detectaría.

Otro contraste que sí que tiene esto en cuenta es el contraste de Breusch y Godfrey, cuya hipótesis nula plantea la ausencia de autocorrelación frente a la alternativa que plantea un modelo AR(m) o MA(m).

Procedimiento:

- Estimar el modelo por MCO y obtener los residuos.
- Estimar estos residuos frente a m retardos suyos y todas las variables explicativas del modelo. Obtener el r-cuadrado de esta regresión.

- Calcular el estadístico del contraste: número de observaciones multiplicado por el r-cuadrado de la anterior regresión.
- Este estadístico se distribuye asintóticamente como una chi-cuadrado con m grados de libertad.

El problema reside en fijar m (número de retardos del modelo AR o MA). En nuestro caso vamos a suponer que m=1.

Para realizar el contraste con RStudio se necesita cargar el paquete *lmtest* y utilizar el comando *bgtest:*

```
bgtest(reg)
```

El resultado se muestra a continuación en la tabla 13:

Tabla 13: Resultado LM test

Breusch-Godfrey test for serial correlation of order up to 1 data: reg LM test = 34.948, df = 1, p-value = 3.386e-09

Como se puede ver en la tabla se rechaza la hipótesis nula por lo que se daría el caso de autocorrelación entre las perturbaciones. Los estimadores de los parámetros por MCO no son óptimos, la matriz de varianzas y covarianzas de los estimadores ya no coincide con la de MCO y las distribuciones para hacer los contrastes no son válidas (ni las finitas ni las asintóticas).

Por último, vamos a tratar un método gráfico para analizar la autocorrelación entre las perturbaciones, el correlograma. Se trata de un gráfico mediante el cual se pueden identificar los diferentes esquemas autorregresivos que siguen las perturbaciones y se calcula a partir de los coeficientes de correlación entre las perturbaciones de diferentes periodos. Para obtener este gráfico, primero hace falta calcular los residuos de la regresión:

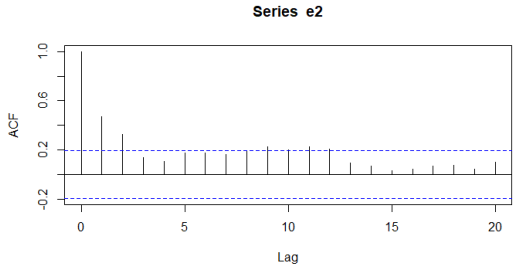
```
e2<-resid(reg)
```

Y posteriormente, usar el comando *acf* (aucotocorrelation function) para obtener el gráfico:

```
acf(e2)
```

El resultado se muestra a continuación en la ilustración 6:

Ilustración 6: Correlograma



Se puede apreciar que la correlación decrece rápidamente hacia cero, por lo que podría encajar con un modelo autoregresivo.

6.8. Hausman

En este apartado vamos a tratar el tema de los regresores estocásticos. Una de las hipótesis clásicas es la de que la matriz de regresores es no aleatoria, es decir que si repitiésemos el experimento los valores obtenidos de los regresores serían los mismos.

La validez de esta hipótesis se puede contrastar mediante el test de exogeneidad de Hausman. La hipótesis nula considera los regresores como variables exógenas al modelo (es decir no aleatorias), mientras que la hipótesis alternativa los considera como variables endógenas en el modelo (es decir aleatorias). Si se rechaza la hipótesis nula y se estima por MCO los estimadores no cumplirían ninguna propiedad, las varianzas estarían mal calculadas y las distribuciones de los contrastes no serían válidas.

La lógica del contraste consiste en estimar los parámetros del modelo por MCO y por Variables Instrumentales; si los regresores son exógenos ambas estimaciones son consistentes, por lo que su diferencia tiende a ser pequeña, próxima a cero si la muestra es grande. Por el contrario, si su diferencia es muy significativa, se rechazaría la hipótesis nula (Gujarati, 2010).

Para realizar el contraste en RStudio seguiremos los siguientes pasos:

- Realizaremos una regresión auxiliar de las variables que posiblemente sean endógenas frente a los instrumentos de estas variables y las variables exógenas del modelo y guardamos los valores estimados de los residuos de este modelo.
- Introducimos en la regresión original los valores estimados de los residuos y realizamos la estimación de los coeficientes. Si el coeficiente de los residuos es significativo rechazamos la hipótesis nula de exogeneidad.

Inicialmente, sólo vamos a incluir en el modelo las variables explicativas: riqueza, desarrollo y escolarización media:

```
regre2<-lm(esperanza_vida~riqueza+desarrollo+escolarización_media)
```

Vamos a contrastar si la variable “escolarización media” es aleatoria o no. Para lo cual utilizaremos como instrumento de esa variable la “escolarización esperada” (suponemos que está correlacionada con la “escolarización media” y que está incorrelacionada con las perturbaciones).

Hallamos los residuos de la regresión auxiliar:

```
stage1=lm(escolarización_media~escolarización_esperada+riqueza+desarrollo)

e1<-resid(stage1)
```

Introducimos los residuos como regresor adicional en la regresión original:

```
stage2<-lm(esperanza_vida~riqueza+desarrollo+escolarización_media+e1)
```

Vemos la significación individual de la variable “e1”:

```
coeftest(stage2)
```

El resultado se muestra a continuación en la tabla 14:

Tabla 14: Resultado test de Hausman

```
t test of coefficients:

                      Estimate Std. Error t value  Pr(>|t|)
(Intercept)          52.569050   1.820085 28.8827 < 2.2e-16 ***
riqueza               0.051342   0.024495  2.0960   0.03746 *
desarrollo            2.364338   1.542796  1.5325   0.12713
escolarización_media  2.018744   0.322803  6.2538 2.755e-09 ***
e1                   -1.660652   0.379847 -4.3719 2.063e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Rechazamos la hipótesis nula de exogeneidad y las consecuencias de estimar por MCO son las previamente descritas.

7. CONCLUSIONES

En este trabajo se han enseñado las herramientas para programar en R, así como su funcionamiento básico para luego poder aplicarlo en la elaboración de estudios estadísticos simples.

Después de realizar el análisis de regresión sobre el programa RStudio, haber visto sus utilidades y haber trabajado con él, se ha comprobado que presenta gran cantidad de funcionalidades para elaborar análisis econométricos y gráficos. No obstante, conviene añadir que presenta algunas limitaciones con respecto a otros programas donde no es necesario el lenguaje de comandos ya que este viene implementado. En cambio, R presenta una gran ventaja frente a otros programas: la creación de nuevas librerías está a plena disposición del usuario, pudiéndolas compartir con el resto de los usuarios (conviene tener en cuenta que al distribuirse bajo licencia de código abierto muchos usuarios se decantarán por este software).

Gracias al ejemplo que hemos desarrollado, se ha podido realizar un análisis de regresión con R, haciendo uso de las funciones necesarias del programa e instalando las librerías disponibles. Esto ha servido para comprobar que es un software que se puede adaptar a multitud de estudios con aplicaciones tanto en el campo de la estadística como para muchas otras áreas debido a la enorme

cantidad de funcionalidades que tiene. Sin embargo, el análisis puede ser más complicado de realizar que en otros programas como EViews o Matlab.

Por todo ello, podemos concluir que, aunque la utilización de R conlleva dificultades, este software tiene aplicaciones prácticas enormes para la economía.

8. BIBLIOGRAFÍA

Marqués, F. (2017): *R en profundidad. Programación, gráficos y estadística.* Editorial RC Libros, Madrid.

Heiss, F. (2016): *Using R for Introductory Econometrics.* Editorial Createspace, Düsseldorf.

Gujarati, D. N. (2010): *Econometría.* Editorial Mcgraw-Hill, Madrid.

Pérez, G. (2015): "Introducción al análisis econométrico con R". Disponible en https://uvadoc.uva.es/bitstream/10324/15515/1/TFG-E-96.pdf [consulta 8/07/2018]

Cavero, J. (2018): "Material Docente de Econometría II". Disponible en https://es.scribd.com/document/377835688/Copia-de-Material-Teoria-EconometrII-2015-2016 [consulta 5/07/2018].

Rosales, R. (2010): "Errores estándar robustos en heterocedasticidad", Universidad de los Andes, Bogotá. Disponible en https://economia.uniandes.edu.co/files/profesores/ramon_rosales_alvarez/docs/econometria1/salidas%20de%20clase/EJC_25-_Heteroscedasticidad_EE_Robustos.pdf [consulta 5/07/2018].

9. ANEXO

```
install.packages("foreign")

library("foreign", lib.loc="~/R/win-library/3.5")

rentapc=c(12000, 24000, 36000)

ricos<-rentapc>24000

summary(ricos)

estado<-c(“casada”,”soltera”,”soltera”)

s1=seq(1,10)

s2=seq(1,10,by=2)

estado [2:3]

estado1<-estado[2]

datos=data.frame(estado, rentapc)

v1<-c(1,1,3,1,2,5,5,4)

Fv1<-factor(v1)

varianzaA=function(A){n=length(A)+v=sum((A-(sum(A)/n))^2/n+return(v)}

mediaA=function(A){n=length(A)+u=sum(A)/n+return(u)}

Soluciónecuación=function(a,b,c){v=(-b+sqrt(b^2-4*a*c))/2*a+u=(-b-sqrt(b^2-
4*a*c))/2*a+return(c(u,v))}

datos<-data.frame(pr10)

attach(datos)

summary(riqueza)

var(riqueza)

sd(riqueza)

SE=function(A){n=length(A)+v=sd(A)/n+return(v)}
```

```
SE(riqueza)

coeficiente_var=function(A){cv=sd(A)/mean(A)+return(cv)}

boxplot(riqueza,xlab="riqueza")

plot(escolarización_esperada,escolarización_media)

lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza+desarrollo)

summary(lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza+desarrollo))

desarrollo1<-as.numeric(riqueza>10.60489)

table(desarrollo1)

lm(esperanza_vida~desarrollo1+escolarización_esperada+escolarización_media+riqueza)

desarrollo1<-as.logical(riqueza>10.60489)

desarrollo11=car::recode(riqueza,"lo:3.6906=1;3.6906:10.6671=2;10.6671:22.8006=3;22.8006:hi=4")

desarrollo11<-as.factor(desarrollo11)

lm(esperanza_vida~desarrollo11)

coeftest(lm(esperanza_vida~riqueza+escolarización_esperada+escolarización_media+desarrollo))

myHO<-c("escolarización_esperada","escolarización_media")

linearHypothesis(lm(esperanza_vida~riqueza+escolarización_esperada+escolarización_media+desarrollo),myHO)

regre1=lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza+desarrollo)

pred_dataframe=data.frame(desarrollo=0,escolarización_esperada=5.4,escolarización_media=1.5,riqueza=1.908)
```

```
predict(regre1,pred_dataframe)

predict(regre1,pred_dataframe,interval="confidence",se.fit=TRUE)

shapiro.test(resid(regre1))

ks.test(resid(regre1),"pnorm")

qqnorm(resid(regre1))

dataset=data.frame(esperanza_vida,riqueza,escolarización_esperada,escolarización_media)

regre11<-VAR(dataset)

whites.htest(regre11)

hccm(lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza))

coeftest(lm(esperanza_vida~escolarización_esperada+escolarización_media+riqueza),vcov=hccm)

datos<-data.frame(pr9)

attach(datos)

reg<-lm(CONSUMO~PIBPC+PRECIO)

durbinWatsonTest(reg)

bgtest(reg)

e2<-resid(reg)

acf(e2)

regre2<-lm(esperanza_vida~riqueza+desarrollo+escolarización_media)

stage1=lm(escolarización_media~escolarización_esperada+riqueza+desarrollo)

e1<-resid(stage1)

stage2<-lm(esperanza_vida~riqueza+desarrollo+escolarización_media+e1)

coeftest(stage2)
```