Facultad de Ciencias Sociales y Jurídicas

Grado en Estadística y Empresa

UNIVERSIDAD DE JAÉN
Facultad de Ciencias Sociales y Jurídicas

Trabajo Fin de Grado

POSIBILIDADES INTERACTIVAS DE R COMO ENTORNO DE TRABAJO PARA UN ANÁLISIS DINÁMICO DE DATOS

Alumna: Lourdes Ballesteros Bosques

Junio, 2017

RESUMEN. En este trabajo se describen las posibilidades interactivas del software estadístico R como entorno de trabajo para un análisis dinámico de datos. Se incluye un ejemplo de una función creada para un análisis a medida y se utiliza RStudio como entorno de desarrollo integrado (IDE) para R que facilita la creación de documentos o presentaciones que integran texto y código.

Palabras clave: Estadística; R; Software estadístico; RStudio; Script; R Markdown; R Notebook; R Presentation; Shiny; Aplicaciones web

ABSTRACT. In this work, we describe interactive possibilities of R Statistical Software as an environment for the dynamic analysis of data. An example with an R function created to obtain an *ad hoc* analysis is included, and RStudio is employed as an integrated development environment (IDE) for R which facilitates the creation of documents or presentations that integrate text and code.

Keywords: Statistics; R; Statistical software; RStudio; Script; R Markdown; R Notebook; R Presentation; Shiny; Web apps

Contenido

1 INTRODUCCIÓN

El objetivo del Grado en Estadística y Empresa es formar a los estudiantes para que adquieran la capacidad de aplicar métodos y modelos de estadística e investigación operativa en el ámbito empresarial, así como realizar un análisis de datos y tomar decisiones en un ámbito de incertidumbre.

Algunas de las competencias generales que pretendemos adquirir a través de la realización de este Trabajo de Fin de Grado son:

- Habilidad en herramientas informáticas, en este caso R y RStudio.
- Habilidad de gestión de la información, es decir buscar y analizar información de distintas fuentes.
- Capacidad de resolución de problemas, toma de decisiones, razonamiento crítico y autocritico, capacidad de aprendizaje y trabajo autónomo.
- Aptitud de preocupación por la calidad.
- Capacidad de aplicar los conocimientos aprendidos a lo largo del Grado y de ponerlos en práctica.

Y en cuanto a las competencias específicas:

- Aplicar los conceptos básicos de Descripción y Exploración Estadística de Datos.
- Utilizar sistemas operativos mediante interfaz gráfica.
- Utilizar herramientas informáticas en empresa.
- Conocer y utilizar entornos de análisis y programación estadística (R).

Para conseguir dichas competencias vamos a utilizar el programa estadístico R.

1.1. Características fundamentales de R

El software R es muy significativo en el mundo de la estadística, una herramienta muy potente en el mercado. Son muchos los usuarios de R, tanto particulares como empresas, que lo utilizan para el análisis de datos, pero ¿qué es realmente R?, ¿cómo se usa y porque deberíamos utilizarlo?

R se puede definir desde dos puntos de vista. Por un lado, es un lenguaje de programación y, por otro, es un entorno de trabajo, ambos orientados al ámbito estadístico y a la elaboración de gráficos.

Las principales características de R son:

- Cuenta con numerosos paquetes y librerías para definir nuestras propias funciones.

- Cuenta con un código abierto, es decir, los usuarios son los que desarrollan funciones y paquetes que pueden ser compartidos con el público.
- Posibilidad de representar gráficos en numerosos formatos y de alta calidad.
- Permite leer lenguajes de otros softwares.
- Y, sobre todo, es un software gratuito.

En cualquier caso, existen otros numerosos lenguajes de programación para trabajar, entonces, ¿por qué R?

Nos encontramos en el mercado con 3 lenguajes ampliamente utilizados, que son SPSS, SAS y R.

SPSS fue lanzado originalmente para las ciencias sociales, con un uso sencillo a través de los botones de la interfaz gráfica.

SAS es más utilizado en la comunidad estadística, sobre todo en el ámbito empresarial que requiere el conocimiento de sintaxis ya que ya que se necesita el uso de comandos para la mayoría de sus opciones.

R se trata de un programa estadístico y un lenguaje de programación de uso libre, utiliza comandos para acceder a las distintas opciones mediante la sintaxis computacional, es de distribución gratuita y de código abierto.

Con motivo de aclarar las diferencias entre los distintos programas estadísticos, en la Figura 1 se muestran algunas de las diferencias en aspectos generales de dichos programas estadísticos.

Figura 1. Comparación entre los programas estadísticos SPSS, SAS y R.

	Programa estadístico		
Aspecto	SPSS	SAS	R
Amigabilidad con el usuario	Excelente	Baja-Regular	Baja-Regular
Manipulación de datos	Baja	Buena	Buena
Calidad de gráficos	Regular	Buena-Excelente	Excelente
Control de procesos	Baja	Excelente	Excelente
Costo	U$S 1500	U$S 7200	Gratis
Código fuente disponible	No	No	Sí
Variedad análisis estadísticos	Buena	Buena-Excelente	Excelente
Documentación	Excelente	Buena	Buena-Excelente
Soporte técnico	Bueno	Bueno	Bajo
Sistema operativo	Windows®	Windows® Macintosh® Linux	Windows® Macintosh® Linux

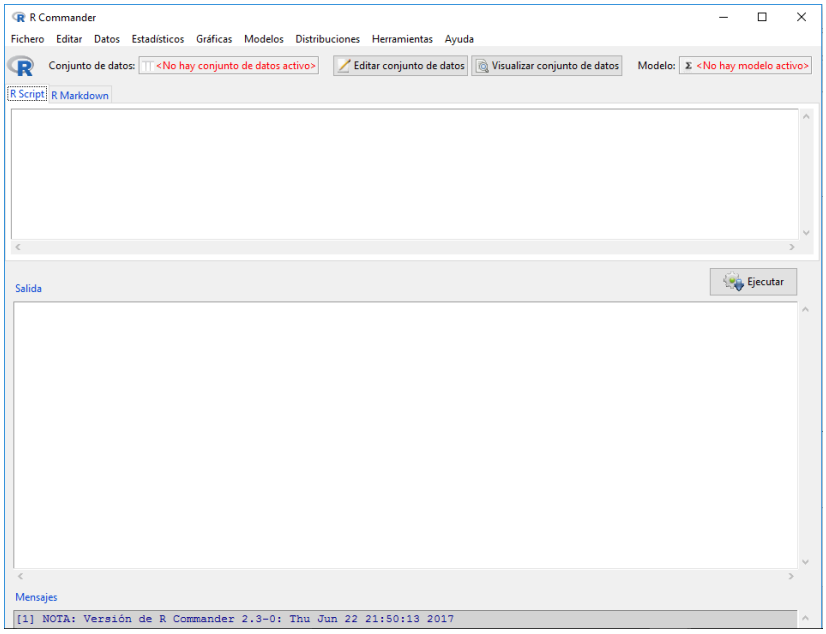
Fuente:http://www.scielo.org.ar/scielo.php?script=sci_arttext&pid=S1667782X2008000200007

Centrándonos en R, sus puntos débiles, según la tabla, son el soporte técnico y la amigabilidad con el usuario.

Aunque en estos aspectos se encuentre en desventaja con SPSS y SAS, en nuestra opinión esto no es realmente así.

Por un lado, sí es cierto que R utiliza sintaxis, por lo tanto, puede ser poco atractivo para los usuarios poco familiarizados con la programación o con poco interés por aprender gran cantidad de instrucciones. Pero para R, existe "R Commander", por ejemplo, que es una Interfaz Gráfica de Usuario (GUI) que cuenta con un menú de funciones como vemos en la Figura 2.

Figura 2. R Commander



Fuente: Elaboración propia.

Esto acerca más a los usuarios a R ya que permite la elaboración de muchas capacidades estadísticas del entorno R, pero sin que el usuario tenga que conocer los comandos.

La otra desventaja era el soporte técnico, pero, ¿es cierto que esto es realmente así?

Es cierto que R no cuenta con apoyo oficial de los propietarios, pero sí que cuenta con una gran comunidad de usuarios dispuestos a resolver cualquier problema. Dado que existe una alta colaboración entre los usuarios de R, los problemas en paquetes son mejorados y además existen numerosos foros en Internet para solucionar problemas y esto puede ser una buena herramienta de ayuda, y de forma gratuita. Además, la comunidad de R ha facilitado diferentes manuales o documentos de forma gratuita

En cuanto a los puntos fuertes de R nos encontramos la calidad de los gráficos, el control de procesos, la variedad de análisis estadísticos y la documentación.

En lo que a gráficos se refiere, R destaca frente a los otros debido a que permite su diseño

personalizado y, además, ofrece la posibilidad de exportar los gráficos en diferentes formatos sin necesidad de más sintaxis. En cuanto, a la elaboración de tablas, en nuestro Trabajo Fin de Grado mostraremos la función *ktable* del paquete knitr, puesto que realiza unas tablas atractivas y de forma sencilla. Pero existen otras opciones como pueden ser el comando *pandoc.table* del paquete *pander*, siendo el aspecto de dicha tabla de la forma en que vemos en la Figura 3.

Figura 3. Tabla con la función pandoc.table

MEDIA	DT	VARIANZA	P25	MEDIANA	P75	ASIMETRIA	CURTOSIS
0.25	1.17	1.37	-0.58	0.25	1.06	0.24	-0.29
0.06	1.03	1.06	-0.83	0.12	0.89	-0.04	-0.97

Fuente: Elaboración propia.

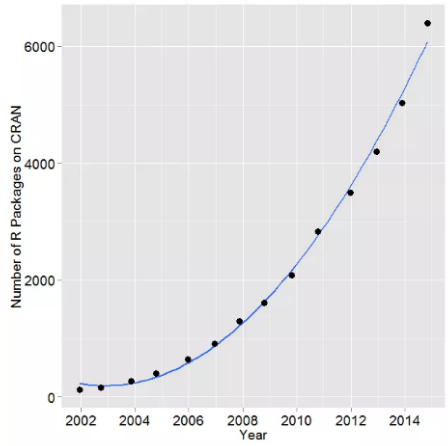
Otra opción es a través de la función *xtable* del paquete *xtable*.

A través de este Trabajo también veremos que R puede realizar tablas de forma similar a las de SPSS.

Por control de procesos nos referimos a que utiliza diferentes algoritmos con diferentes variantes y, como R requiere sintaxis, facilita el control de los procedimientos estadísticos. Eso es debido a que R es más flexible por ser un código abierto.

La variedad de estadísticos se debe a que R implementa algoritmos modernos y robustos y, además, cuenta con numerosos paquetes que están continuamente mejorándolos y a disposición de los usuarios. Para hacernos una idea del gran crecimiento de los paquetes de R tenemos la Figura 4 en la que podemos ver como el crecimiento de R es asombroso.

Figura 4. Numero de paquetes de R disponibles



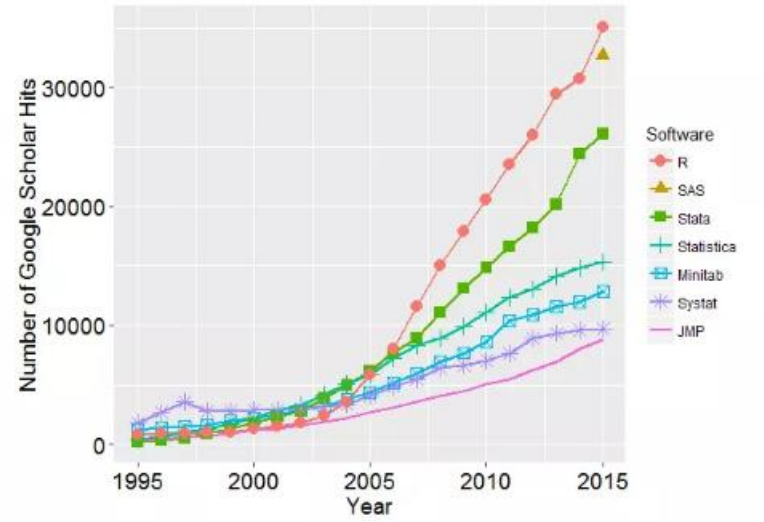
Fuente: http://r4stats.com/articles/popularity/

Y, por último, destaca por su documentación, es decir, como ya dijimos anteriormente, los usuarios de R han dejado numerosos manuales e informes a disposición de los demás de forma gratuita.

A todo esto, hay que añadir que se trata de un software gratuito y está disponible en internet. R consiste en una serie de paquetes que se van actualizando y están siempre disponibles para los usuarios, y esto es un aspecto claramente ventajoso. Además de su disponibilidad en los distintos sistemas operativos, R funciona de manera íntegra y estable en los 3 sistemas operativos de mayor uso, que son Windows, Macintosh y Linux. Y, además destaca por otra gran ventaja, permite exportar documentos a diferentes formatos, como Microsoft Word o PDF.

Hasta hace pocos años R era desconocido en el ámbito académico, pero como hemos visto anteriormente debido a sus numerosas fortalezas como ser un lenguaje libre, abierto y gratis hace que se esté extendiendo su uso como vemos en la Figura 5.

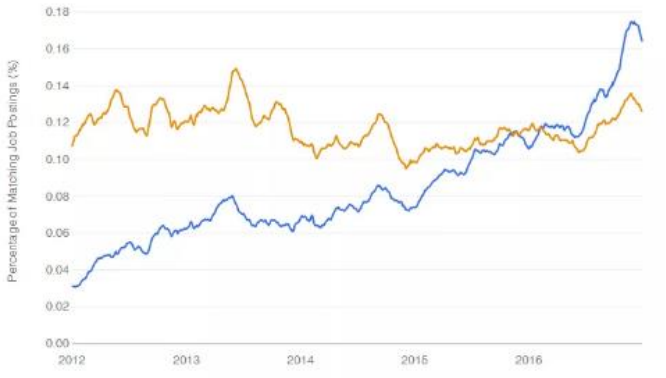
Figura 5. Número de artículos académicos



Fuente: http://r4stats.com/articles/popularity/

Pero su crecimiento no ha sido solo en el ámbito académico, también se puede extrapolar al ámbito empresarial como vemos en la Figura 6 donde vemos que R ha crecido de forma sostenida hasta que finalmente el porcentaje de puestos de trabajo ofertados en lo que se requiere R es mayor que para SAS.

Figura 6. Tendencias de empleo para R (azul) y SAS (naranja)



Fuente: http://r4stats.com/articles/popularity/

1.2. Interacción con R a través de RStudio

Entre las interfaces para la interacción con R, nos encontramos con un entorno de desarrollo integrado (IDE), llamado RStudio.

RStudio es uno de los entornos más populares de R, ya que facilita el uso interactivo de R y con él podemos llegar a exprimir al máximo el potencial del lenguaje de R.

Dentro de RStudio nos encontramos con otras herramientas como R Markdown o Shiny.

R Markdown convierte un análisis de datos en informes, presentaciones o documentos, mientras que Shiny se utiliza para crear aplicaciones webs interactivas (apps) para que los usuarios puedan interactuar con sus datos sin tener que cambiar el código usado.

En primer lugar, para descargar R, podemos realizar la descarga desde el siguiente enlace: https://cran.r-project.org/bin/windows/base/

Después procedemos a descargarnos R Studio; lo podemos realizar desde el siguiente enlace: https://www.rstudio.com/products/rstudio/download3/

La descarga de R Studio está disponible tanto para Windows, Linux y para MAC. En la Figura 7 aparece destacada la opción de instalación en el sistema operativo Windows.

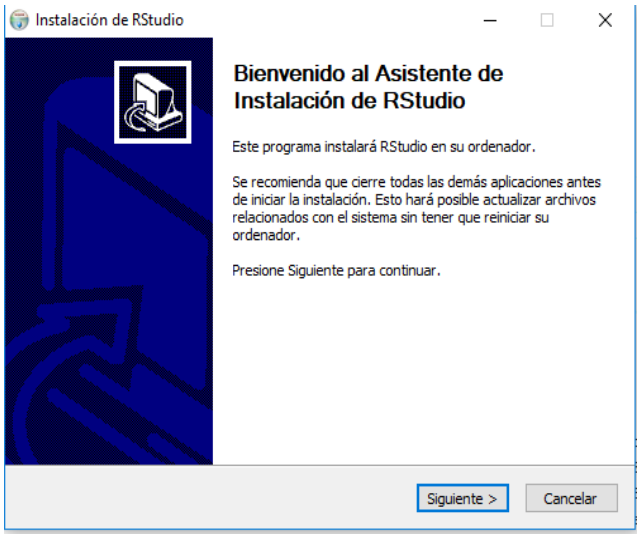
Figura 7. Descargas RStudio

Installers	Size	Date	MD5
RStudio 1.0.143 - Windows Vista/7/8/10	81.9 MB	2017-04-19	76bb84296b9202759b3eb1de555a2231
RStudio 1.0.143 - Mac OS X 10.6+ (64-bit)	71.2 MB	2017-04-19	c7f1ed865428b225b202fd1b431954b4
RStudio 1.0.143 - Ubuntu 12.04+/Debian 8+ (32-bit)	85.5 MB	2017-04-19	21ca14bffcdc1a2361ead2d763d0313d
RStudio 1.0.143 - Ubuntu 12.04+/Debian 8+ (64-bit)	92.1 MB	2017-04-19	75761eae209158d8415d562b3771fbec
RStudio 1.0.143 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (32-bit)	84.7 MB	2017-04-19	2c356d4ee50667ad4042ee196afb3c53
RStudio 1.0.143 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (64-bit)	85.7 MB	2017-04-19	7ab5fc240351debe491c6c5a7acb6068

Fuente: Elaboración propia.

Una vez nos hayamos descargado nuestro archivo correspondiente, su instalación es bastante sencilla, en la Figura 8 vamos a ver cómo se realizaría para Windows ya que es el sistema operativo utilizado en este caso.

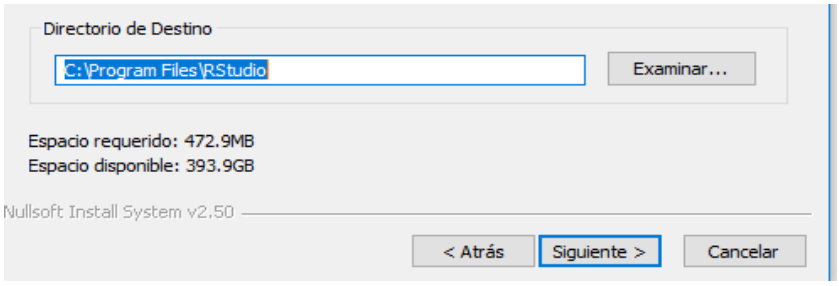
Figura 8. Instalación RStudio



Fuente: Elaboración propia.

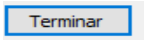
Esta sería la primera pantalla que nos aparecería en el proceso de instalación, pulsamos siguiente y ahora nos dice cual queremos que sea nuestro directorio de trabajo como podemos ver en la Figura 9, seguimos pulsando siguiente a las sucesivas ventanas, hasta que se llegue a la de instalación.

Figura 9. Directorio de destino



Fuente: Elaboración propia.

Cuando lleguemos a la ventana donde nos aparezca 

pinchamos y se procede a la instalación, una vez se haya completado el proceso nos aparece 

y el proceso habrá finalizado, por lo que ya tendremos RStudio en nuestro escritorio, tal y como podemos observar en la Figura 10.

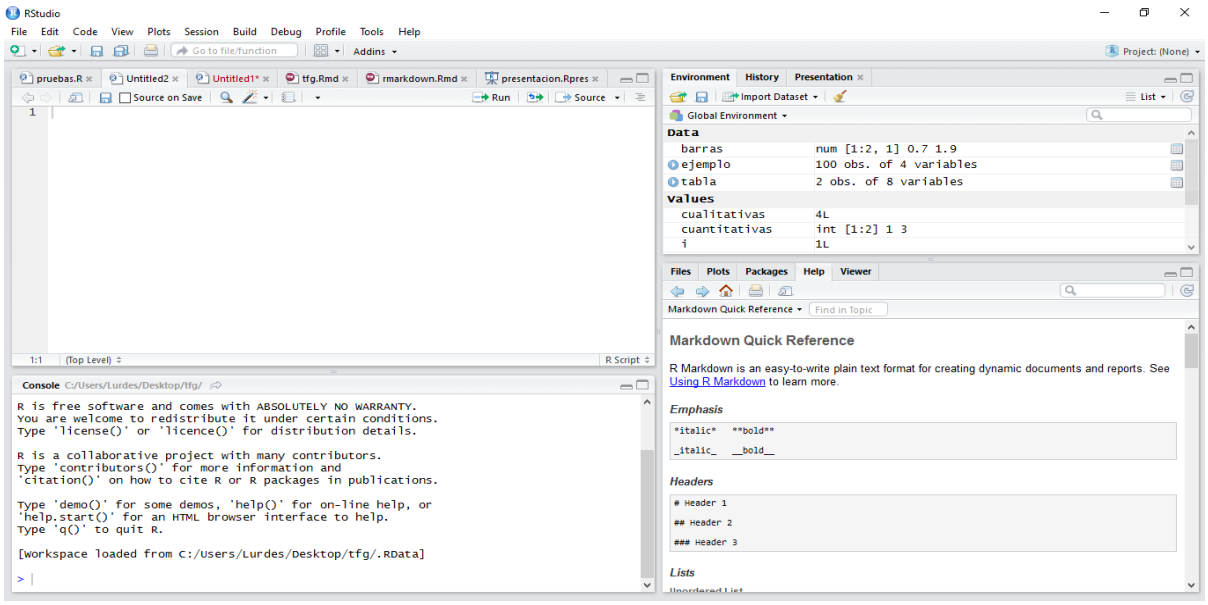
Figura 10. Icono RStudio

Fuente: Elaboración propia.

Una vez tengamos RStudio, su utilización es bastante trivial.

En la Figura 11 nos encontramos con la pantalla principal de R Studio.

Figura 11. Pantalla principal RStudio



Fuente: Elaboración propia.

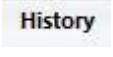
Esta sería la pantalla principal. Lo primero que nos encontraríamos al abrir RStudio, como podemos observar cuenta con 4 pantallas diferenciadas, la primera sería el editor de RStudio, debajo de esta nos encontramos con la consola, que es el espacio de trabajo.

En la parte derecha en primer lugar tenemos el historial de objetos que se encuentran en la memoria.

Desde ella tenemos la opción de:

- Limpiar el historial 

- Importar datos 

- Muestra los comandos de los informes con los que se han trabajado 

En la parte inferior se encuentra el directorio de trabajo: en él se pueden visualizar los gráficos, los paquetes, una ayuda y el visor HTML.

Para comenzar a trabajar simplemente tenemos que pulsar  donde veremos las distintas alternativas de R que posteriormente analizaremos.

2 ANÁLISIS A MEDIDA. FUNCIONES. SCRIPTS

En la realización de este Trabajo Fin de Grado hemos partido de un supuesto en el que una empresa nos contrata para que le proporcionemos una herramienta de análisis de datos a la medida. Concretamente, vamos a suponer que ellos necesitan una herramienta capaz de obtener un análisis descriptivo básico que luego comentaremos, de tal manera que ellos faciliten un fichero de datos de entrada y la herramienta proporcione un análisis diferenciado de las variables del fichero en función de su carácter: la herramienta debe ser capaz de identificar las variables cualitativas y las cuantitativas y elaborar un análisis descriptivo adecuado para cada una de ellas.

El contexto en el que este objetivo se puede abordar mediante R es el de la creación de un *script* o guión de código que incluya alguna función capaz de realizar la tarea requerida. Un *script* es un conjunto de órdenes para realizar una acción que se puede ejecutar por lotes o línea a línea.

A continuación, vamos a describir cómo hemos implementado la función en el script que facilitaríamos a la empresa.

En primer lugar, vemos la forma en que se define la función, se cargan las librerías necesarias y se genera un bucle para identificar las variables cualitativas y cuantitativas:

```
funcion <- function(datos){
  library(xtable)
  library(e1071)
  cualitativas <- c()
  cuantitativas <- c()
  for(i in (1 : ncol(datos))){
    if(is.numeric(datos[, i])){
      cuantitativas <- c(cuantitativas, i)
    }
    if(is.factor(datos[, i])){
      cualitativas <- c(cualitativas, i)
    }
  }
```

En el script podemos observar como el primer paso es definir la función, después se procede a cargar los paquetes necesarios para nuestro estudio, en este caso **xtable**, utilizada para realizar tablas en R Studio a través de la función **xtable** de este paquete y **e1071** donde encontramos las

funciones para realizar la asimetría y la curtosis. Después se procede a la identificación de si las variables de estudio son variables cualitativas o cuantitativas.

En el siguiente fragmento de código generamos una hoja de datos que contiene los descriptivos para las variables cuantitativas:

```
  tabla <- data.frame(MEDIA = numeric(length(cuantitativas)), DT =
numeric(length(cuantitativas)), VARIANZA =
numeric(length(cuantitativas)), P25 =
numeric(length(cuantitativas)), MEDIANA =
numeric(length(cuantitativas)), P75 =
numeric(length(cuantitativas)), ASIMETRIA =
numeric(length(cuantitativas)), CURTOSIS =
numeric(length(cuantitativas)))
  for(i in (1 : length(cuantitativas))){
    tabla[i,] <- c(mean(datos[, cuantitativas[i]]), sd(datos[,
cuantitativas[i]]), var(datos[, cuantitativas[i]]), quantile(datos[,
cuantitativas[i]], 0.25), median(datos[, cuantitativas[i]]),
quantile(datos[, cuantitativas[i]], 0.75), skewness(datos[,
cuantitativas[i]]), kurtosis(datos[, cuantitativas[i]]))
```

Como podemos ver, se construye una tabla donde se muestran la media, la desviación típica, varianza, percentil 25 y 75, asimetría y curtosis de cada variable cuantitativa.

A continuación, y dentro del mismo bucle que se aplica al conjunto de las variables cuantitativas, obtenemos un histograma y un diagrama de caja para cada variable que se guardará en formato JPG con un nombre que se basa en el de la variable que describe para permitir su identificación posterior:

```
jpeg(paste("hist_", names(datos)[cuantitativas[i]],".jpg",sep = ""))
hist(datos[,    cuantitativas[i]],    main    =    "",    xlab    =
names(datos)[cuantitativas[i]], ylab = "frecuencias")
    dev.off() jpeg(paste("boxplot_", names(datos)[cuantitativas[i]],
".jpg", sep = ""))
    boxplot(datos[, cuantitativas[i]])
    dev.off()
  }
```

En el caso de las variables cualitativas se genera un gráfico de sectores y un gráfico de barras.

```
  for(i in ( 1 : length(cualitativas))){
    tablai <- prop.table(table(datos[,cualitativas[i]]))
    tablai <- round(100*tablai,2)
jpeg(paste("sectores", names(datos)[cualitativas[i]], ".jpg", sep = ""))
    sectores<-pie(prop.table(table(datos[, cualitativas[i]])),labels = paste(names(tablai), tablai, "%"), main = paste("diagrama de sectores de", names(datos)[cualitativas[i]]))
    dev.off()
    tablai <- table(datos[, cualitativas[i]])
jpeg(paste("barras", names(datos)[cualitativas[i]], ".jpg", sep = ""))
    barras<-barplot(tablai, col = rainbow(length(names(tablai))), xlab = "x", ylab = "Frecuencias absolutas")
    text(barras, tablai + 1, labels = tablai, xpd = TRUE)
    title(main = paste("Distribución de frecuencias de ", names(datos)[cualitativas[i]]), font.main = 4)
    dev.off()
  }
```

Finalmente, la función termina generando la tabla de estadísticos descriptivos en un fichero HTML que puede ser manejado con comodidad desde otros entornos.

```
write(print(xtable(tabla, digit = 2), type = "html"), file = "tabla.htm")
}
```

En un supuesto de que la empresa quiera ampliar el script, es decir, si quisieran hacer un estudio más elaborado con más funciones o gráficos, sería posible realizarlo, modificando las líneas de código correspondiente.

3 GENERACIÓN DINÁMICA DE INFORMES MEDIANTE R MARKDOWN

Como ya hemos comentado en la introducción, R es un lenguaje de programación orientado al análisis estadístico que, por tanto, permite realizar tareas a la medida de las necesidades del usuario. En el apartado anterior hemos realizado un ejemplo al respecto.

El principal inconveniente que se suele considerar en el uso de R es acerca de las salidas, ya que normalmente, en otros programas estadísticos basados en entornos tipo ventana y manejados mediante menús en vez de código, las tablas y las figuras resultantes de los análisis son fácilmente exportables a los editores de texto más comunes.

Sin embargo, R y más concretamente R manejado desde la interfaz RStudio posee unas enormes posibilidades, a día de hoy quizá poco conocidas, a la hora de incorporar tanto las tablas como las figuras resultantes a un informe, que pueda materializarse en un documento escrito o bien en una presentación. Además, el entorno en el que se puede redactar dicho informe incluye tanto el código de R necesario para llevar a cabo las técnicas estadísticas oportunas como el propio texto. Finalmente, una enorme ventaja en la redacción de estos informes, como veremos a continuación, es que son *dinámicos*, es decir, los resultados varían en el informe cuando varían los propios datos, sin tener que modificar el informe de origen.

Siguiendo con el supuesto de que hemos sido contratados por una empresa para obtener un programa que proporcione un análisis a medida, como el descrito en el apartado anterior, ahora vamos a suponer que la misma empresa nos solicita una manera de proporcionar informes relativos a el uso de ese programa con distintas hojas de datos, de manera que ellos obtengan un documento o una presentación vinculados a cada hoja de datos sin tener que modificar el informe *padre* inicial. El entorno adecuado donde puede darse respuesta a este problema es R Markdown.

3.1 Introducción a R Markdown

R Markdown es un formato para la creación de documentos, presentaciones y apps de forma dinámica que contiene salidas y *widgets*[1]. Mediante Markdown, el usuario escribe su informe en la pantalla y luego lo lanza, en algún formato a elegir, generando el resultado final.

En un documento R Markdown podemos encontrar tanto lenguaje del código Markdown como de R.

[1] Son una pequeña aplicación o programa cuyo objetivo es dar fácil acceso a las funciones y obtener información visual. Los widgets son utilizados en Shiny.

Para trabajar con R Markdown podemos seleccionarlo desde el menú de RStudio o bien con la siguiente función para instalar el paquete:

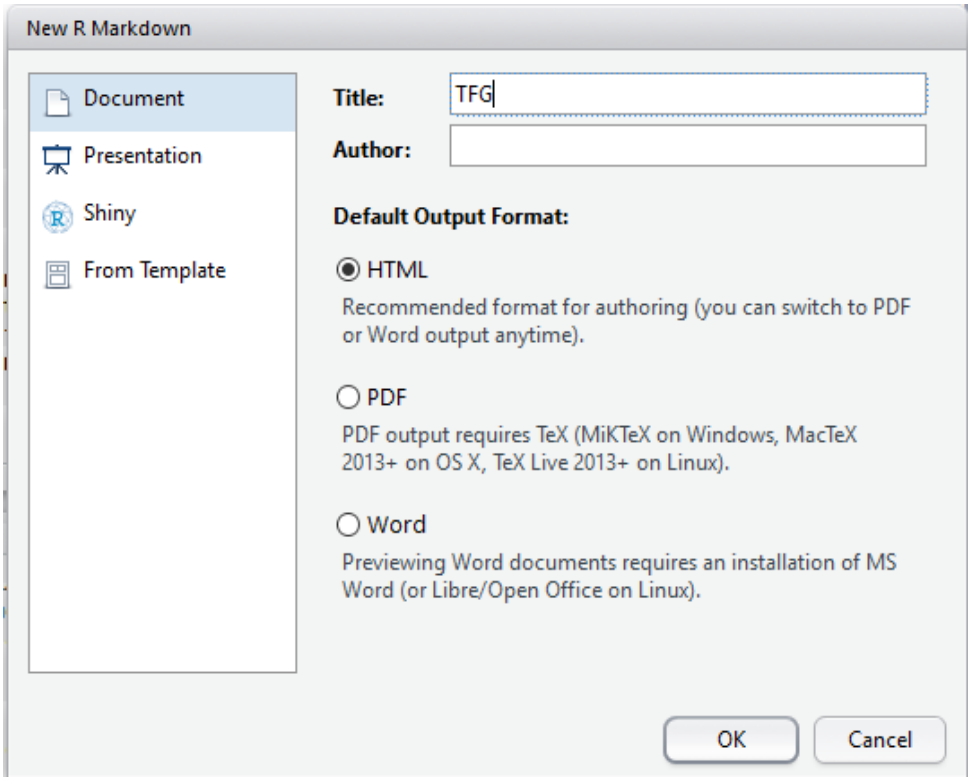
```
install.packages("rmarkdown")
```

R Markdown utilizado desde R Studio permite generar tres tipos de formatos de salida:

1. Documentos, con formatos HTML, PDF[2] y Microsoft Word.
2. Presentaciones en formatos HTML y PDF.
3. Aplicaciones mediante Shiny (documento y presentaciones)

En la Figura 12 podemos observar las diferentes opciones de R Markdown como la elaboración de documentos, presentaciones o Shiny.

Figura 12. R Markdown

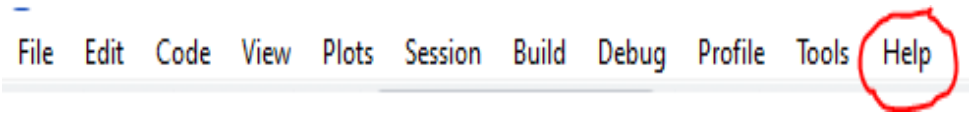


Fuente: Elaboración propia.

R Studio cuenta con numerosas ayudas para R Markdown. Acceder a ellas es relativamente sencillo, lo único que tenemos que hacer es irnos a la ayuda de RStudio, como podemos ver en la Figura 13.

[2] Para obtener documentos en formato PDF es necesario tener instalada alguna distribución de LaTeX.

Figura 13. Menú R Studio



Fuente: Elaboración propia.

Una vez en la ayuda seleccionamos *cheatsheets*, que son como "chuletas" de R Markdown que muestran algunas de las opciones posibles. R Markdown cuenta con dos ayudas de este tipo:

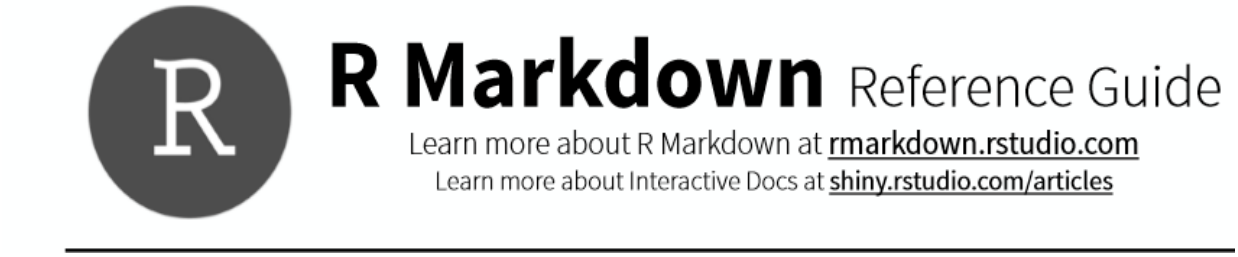
1. R Markdown Cheat Sheet. (Figura 14)

Figura 14. R Markdown Cheat Sheet

Fuente: Elaboración propia.

2. R Markdown Reference Guide. (Figura 15)

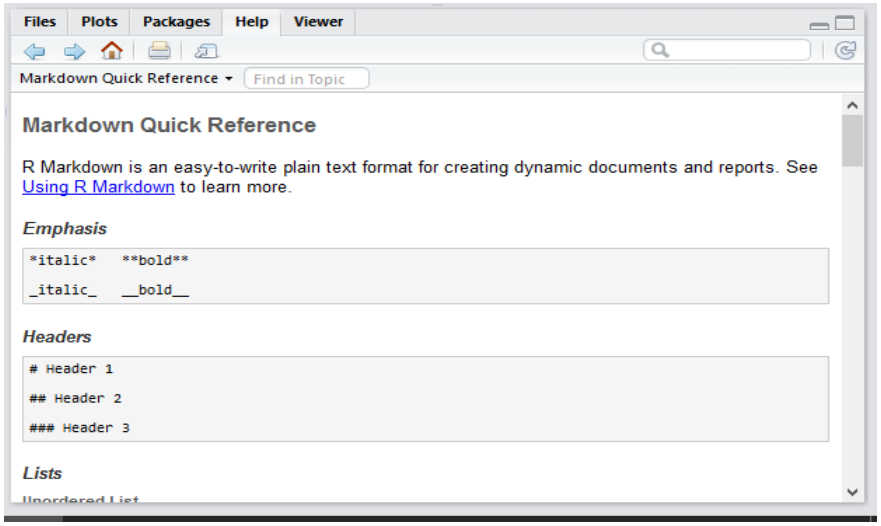
Figura 15. R Markdown Reference Guide



Fuente: Elaboración propia.

También cuenta con otra ayuda, la cual está en la ayuda del menú y se titula Markdown Quick Reference: esta ayuda se nos abre directamente en RStudio, no como las anteriores que son desde internet, como podemos ver en la Figura 16.

Figura 16. Markdown Quick Reference



Fuente: Elaboración propia.

3.2 Estructura y edición de formato en un documento R Markdown

Como vimos en la Figura 12, R Markdown cuenta con tres opciones para la creación de archivos: Documentos, Presentaciones y Shiny. En cualquiera de los tres, lo primero que nos encontramos en un encabezado YAML[3], como podemos ver en la Figura 17, en el cual podemos explicar el tipo de documento que queremos crear.

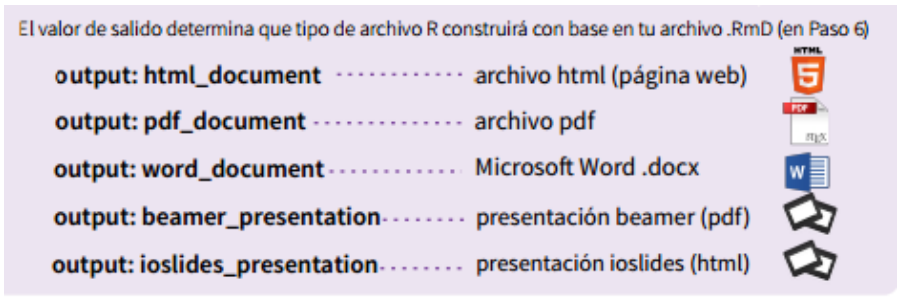
Figura 17. Encabezado

```
1 ---
2 title: "Untitled"
3 output: html_document
4 ---
5
```

Fuente: Elaboración propia.

Las posibles opciones de salida que se especifican en este encabezado son las que aparecen en la Figura 18.

Figura 18. Tipos de archivos de R

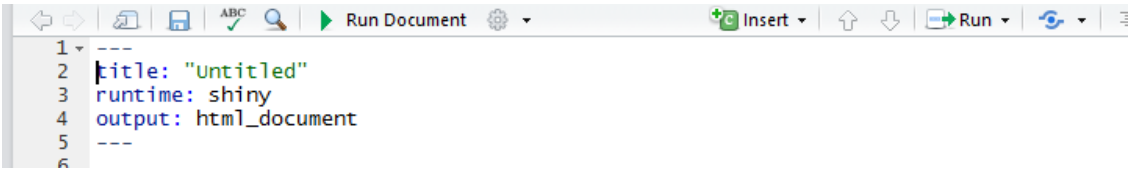


Fuente: https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-spanish.pdf

[3] Se trata de un formato de codificación de un objeto con motivo de transmitirlo en un formato más legible.

Para Shiny el encabezado seria de la forma en la que aparece en la Figura 19.

Figura 19. Encabezado Shiny



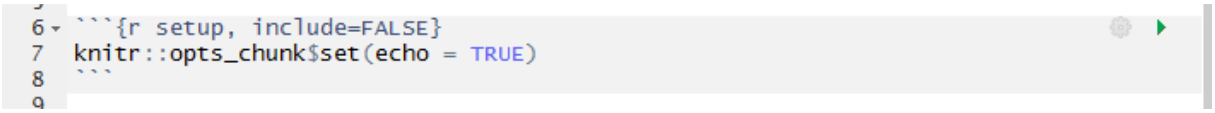
Fuente: Elaboración propia.

Las posibilidades de edición tanto de los documentos como de las presentaciones son similares. Algunas de ellas son:

- Negrita y cursiva.
- Listas.
- Encabezados.
- Hipervínculos.

Como R Markdown permite combinar lenguaje Markdown y R**,** para incrustar un trozo del código de R debemos rodearlo con ``` e incluir {r} al comienzo para avisar de que se trata de código R y no texto, como puede verse en la Figura 20. Estos trozos de código de R se conocen como *chunks*.

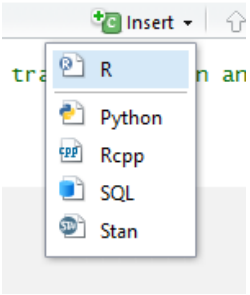
Figura 20. Introducir código de R



Fuente: Elaboración propia.

Para añadir un *chunk* solo debemos pulsar en "Insert" como vemos en la Figura 21.

Figura 21. Añadir chunk



Fuente: Elaboración propia.

Hay que mencionar que R Markdown permite incorporar código de otros lenguajes de programación, como Python o SQL, entre otros (Figura 22)

Figura 22. Lenguajes de programación insertables en R Markdown

Fuente: Elaboración propia.

Existen diferentes modificaciones de los chunk.

Si utilizamos eval= FALSE se omitirán los resultados en el informe final, por lo tanto, se crea una copia del código.

Si utilizamos echo= FALSE se omitirá el código del informe final y se realizará una copia de los resultados del informe final.

Existen otros argumentos para personalizar el código, los cuales podemos ver en la Figura 23.

Figura 23. Opciones de los chunk

opción	defecto	efecto
eval	TRUE	Indica si se va a evaluar el código e incluir los resultados
echo	TRUE	Indica si se muestra el código a la par de los resultados
warning	TRUE	Indica si se muestran advertencias
error	FALSE	Indica si se muestran errores
message	TRUE	Indica si se muestran mensajes
tidy	FALSE	Indica si se muestra código de forma organizada
results	"markup"	Opciones: "markup", "asis", "hold", o "hide"
cache	FALSE	Indica si se guardan resultados en *cache*
comment	"##"	Carácter de comentario para anteponer a resultados
fig.width	7	Ancho en pulgadas para figuras generadas en el trozo
fig.height	7	Alto en pulgadas para figuras generadas en el trozo

Fuente: https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-spanish.pdf

Para concluir, R Markdown proporciona informes dinámicos y reproducibles de R de forma rápida, donde se pueden incluir fragmentos del código R con el paquete **knitr**. Permite actualizar el documento cambiando los trozos de código.

3.3 Ejemplo

Como vimos en el apartado anterior: ANÁLISIS A MEDIDA. FUNCIONES. SCRIPTS, ya le proporcionamos a la empresa a través del programa un análisis a medida, pero el objetivo de este apartado es la de proporcionar un documento que permita plasmar resultados de esa función (o de otros análisis) relativos a distintas hojas de datos sin tener que modificar el informe inicial:

dicho de otra forma, queremos un documento dinámico que adapte las salidas a las distintas hojas de datos que suponen las entradas. Para dicho objetivo el entorno adecuado para realizarlo es R Markdown.

Al igual que creamos un script en R Studio, a través de R Markdown también podemos realizarlo: para elaborar dicho informe utilizaremos algunas de las herramientas explicadas anteriormente, las cuales están detalladas paso a paso en el ejemplo presentado a continuación.

```
---
title: "ANÁLISIS DE DATOS A MEDIDA"
output:
  word_document: default
---
```

Aquí tenemos nuestro encabezado, en el que observamos que le hemos pedido que nos devuelva nuestro documento en formato Word y cuyo título es "ANALISIS DE DATOS A MEDIDA"

```
##INFORME
Lo que nuestra empresa nos solicito fue un script aplicable a sus ficheros de datos como vimos en el capítulo 2, a través de este informe se realiza dicho script pero en forma de documento.
A partir de este informe lo que queremos conseguir es:
**En primer lugar** queremos que se nos muestre una tabla para las variables cuantitativas, donde nos encontramos diferentes estadísticos, como, la media, desviación típica, varianza, percentil 25, mediana, percentil 75, asimetría y curtosis.
**En segundo lugar** realizaremos las representaciones gráficas tanto para las variables cuantitativas como cualitativas.
###1. TABLA
En la tabla podremos observar diferentes estadísticos como:
* Media(m~x~).
* Desviación típica(s~x~).
* Varianza(s^2^~x~).
* Percentil 25(P~25~).
* Mediana(P~50~).
* Percentil 75(P~75~).
* Asimetría.
* Curtosis.
```

Hemos utilizado * para elaborar una lista, que posteriormente veremos su resultado. Así como, hemos utilizado subíndices y superíndices mediante ~ y ^.

```
A continuación, en la Tabla 1 se nos muestran nuestros estadísticos en forma de tabla.
*Tabla 1. Estadísticos*
```{r, echo=FALSE}
library(knitr)
setwd("C:/Users/pc/Desktop/TFG")
source("FUNCION.R")
ejemplo <- data.frame(x=rnorm(100), y=factor(rbinom(100,1,0.5)), x=rnorm(100), y=factor(rbinom(100,1,0.5)))
```
```

```
tabla <- round(funcion(ejemplo), 2)
kable(tabla)
```
Fuente: Elaboración propia

Este es nuestro primer *chunk* sabemos que lo es, ya que como dijimos anteriormente estos se escriben entre ``` e incluyen {r}, en este caso lo hemos utilizado para escribir en primer lugar nuestro directorio de trabajo, posteriormente hemos cargado nuestra función: FUNCION.R. a través del comandado s*ource* que sirve para ejecutar el código de un fichero, después hemos simulado los datos con los que vamos a trabajar y, por último, hemos elaborado la tabla correspondiente a nuestros datos.

```
###2. REPRESENTACIONES GRAFICAS
Para las variables cuantitativas realizaremos el histograma y el boxplot.
*Histograma* es la representación gráfica de la distribución de frecuencias en forma de barras.
En la Ilustración 1 podemos ver el histograma correspondiente a la variable x.
*Ilustración 1. Histograma de x*
![histograma](hist_x.jpg)
 *Fuente: Elaboración propia*
```

Aquí podemos ver como insertamos en R Notebook nuestros gráficos para ello ponemos en primer lugar el símbolo de exclamación (!), seguido del nombre entre [] y por último el nombre de nuestro gráfico, ya que, en este caso, estamos situado en nuestro directorio de trabajo, en caso de no ser así, se tiene que poner el camino a seguir hasta *hist_x.jpg*, en este caso.

Con el resto de gráficos seguiremos el mismo proceso.

```
En la Ilustración 2 podemos ver el histograma correspondiente a x.1.
*Ilustración 2. Histograma de x.1*
![histograma x.1](hist_x.1.jpg)
*Fuente: Elaboración propia*


*Boxplot*, o diagrama de cajas y bigotes.
En la Ilustración 3 podemos ver el boxplot correspondiente a x.
*Ilustración 3. Boxplot x*
![boxplot x](boxplot_x.jpg)
*Fuente: Elaboración propia*


En la Ilustracion 4 podemos ver el boxplot correspondiente a x.1.
*Ilustración 4. Boxplot x.1*
![boxplot x.1](boxplot_x.1.jpg)
*Fuente: Elaboración propia*


Y, por último, para las variables cualitativas realizamos los gráficos de sectores y de barras.
```

```
**Gráficos de barras**
En la Ilustración 5 podemos ver el gráfico de barras correspondiente
a y.
*Ilustración 5. Gráfico de barras y*
![barras de y](barrasy.jpg)
*Fuente: Elaboración propia*


En la ilustración 6 podemos ver el grafico de barras correspondiente
a y.1.
*Ilustración 6. Gráfico de barras y.1*
![barras de y.1](barrasy.1.jpg)
*Fuente: Elaboración propia*


**Diagramas de sectores**
En la Ilustración 7 podemos ver el diagrama de sectores correspondiente
a y.
*Ilustración 7. Diagrama de sectores y*
![sectores y](sectoresy.jpg)
*Fuente: Elaboración propia*


Y, por último, en la Ilustración 8, podemos ver el diagrama de sectores
correspondiente a y.1.
*Ilustración 8. Diagrama de barras y.1*
![sectores y.1](sectoresy.1.jpg)
*Fuente: Elaboración propia*
```

A lo largo del informe hemos podido observar que se han ido añadiendo algunas opciones de escritura cómo ##, * o **, las cuales cuando ejecutemos el archivo se verán en formato subtítulos, cursivas y negritas.

"FUNCION.R" es el script que contiene la función que describimos en el Capítulo 2. ANÁLISIS A MEDIDA. FUNCIONES. SCRIPTS. Y, por último, tenemos nuestro script en este caso realizado mediante un *chunk*. Los resultados lo veremos en el siguiente apartado en el de R NOTEBOOK.

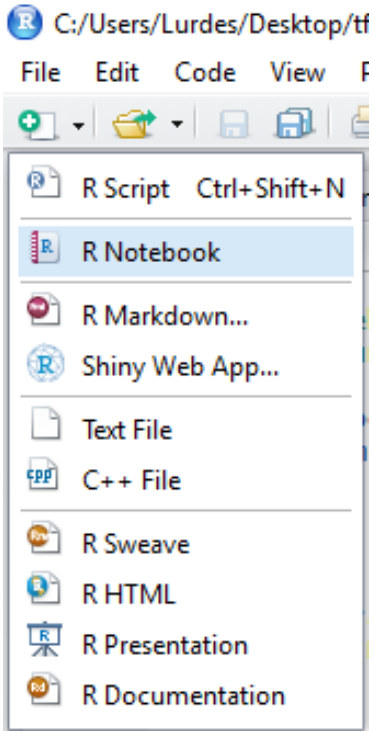
4 R NOTEBOOK

R Notebook es la forma en que R Markdown es capaz de realizar documentos, tanto en Microsoft Word, PDF o HTML, por lo tanto, es un documento R Markdown con trozos que se pueden ejecutar de forma independiente e interactiva.

R Notebooks son una nueva característica de RStudio y solo están disponibles en la versión 1.0 o superior de RStudio.

Vamos a comentar un breve ejemplo de un documento R Markdown, creado a través del menú de R Notebook. Para empezar a trabajar con un fichero R Notebook debemos ejecutarlo tal y como vemos en la Figura 24.

Figura 24. Abrir R Notebook



Fuente: Elaboración propia.

O también se puede a través de R Markdown como vimos en la Figura 12. Algunas de las diferencias que nos podemos encontrar a la hora de la creación de documentos son las de la Figura 25.

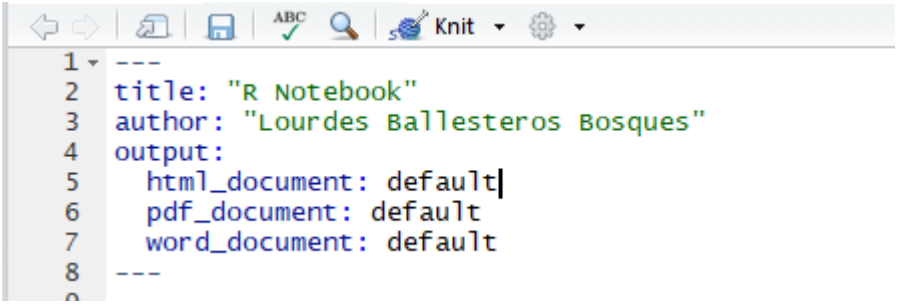
Figura 25. Diferencias R Notebook

	Cuadernos R Markdown	Portátiles Tradicionales
Representación en texto plano	✓	
El mismo editor / herramientas utilizadas para scripts R	✓	
Funciona bien con el control de versiones	✓	
Enfoque en la producción	✓	
Salida en línea con código	✓	✓
Salida en caché en todas las sesiones	✓	✓
Compartir código y salida en un solo archivo	✓	✓
Modelo de ejecución enfatizado	Interactivo y por lotes	Interactivo

Fuente: *https://blog.rstudio.org/2016/10/05/r-notebooks/*

En la Figura 26 se muestra lo primero que nos encontramos cuando abrimos un R Notebook, su correspondiente encabezado.

Figura 26. Encabezado de R Notebook



Fuente: Elaboración propia.

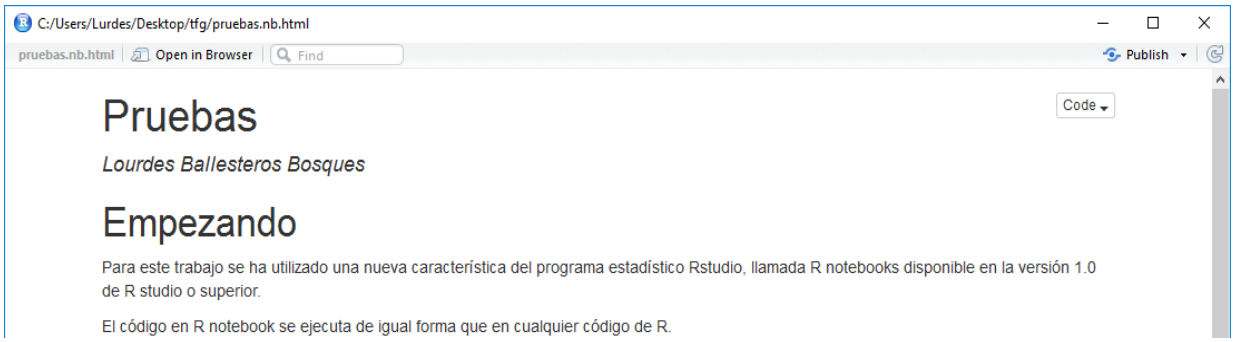
En la línea 2 podemos observar que es donde ponemos el nombre que le queremos poner a nuestro documento, es decir, el título de nuestro fichero.

En la línea 3 nos encontramos con el autor.

Y, por último, en la línea 4 nos encontramos con los tipos de salidas. En la imagen podemos ver cómo nos encontramos con 3 opciones, PDF, Word y HTML, pero la que se ejecutaría es la que se encuentra en primera posición, en este caso, HTML.

Cuando generamos la función nos queda como en la Figura 27.

Figura 27. Vista R Notebook en HTML

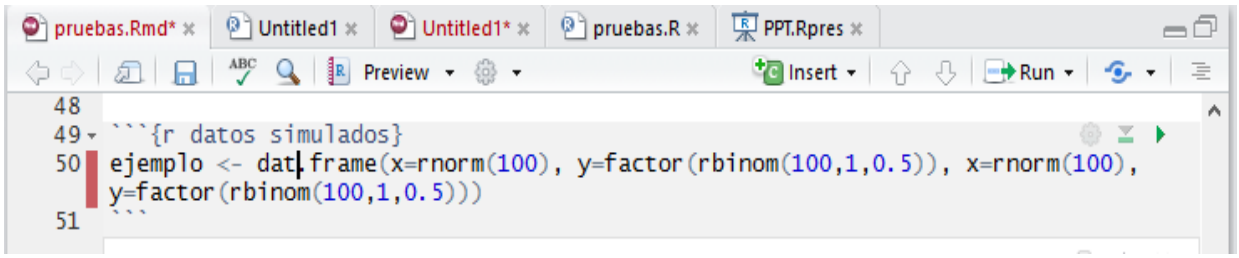


Fuente: Elaboración propia.

Una vez realizada una breve introducción a cómo funciona R Notebook, vamos a proceder a hacer un análisis del mismo.

El código en R Notebook mezcla texto con código de R. El código se envía por *chunk* y la ejecución de detiene en el punto donde se encuentra el error en el caso de que existan errores. Esto se puede apreciar visualmente con una línea verde que aparece a la izquierda del código, que va avanzando conforme se va ejecutando y permite ir viendo el progreso, pero se detiene en el primer error que encuentra, convirtiéndose la línea de color verde en color rojo, como vemos en la Figura 28.

Figura 28. Error en ejecución del chunk

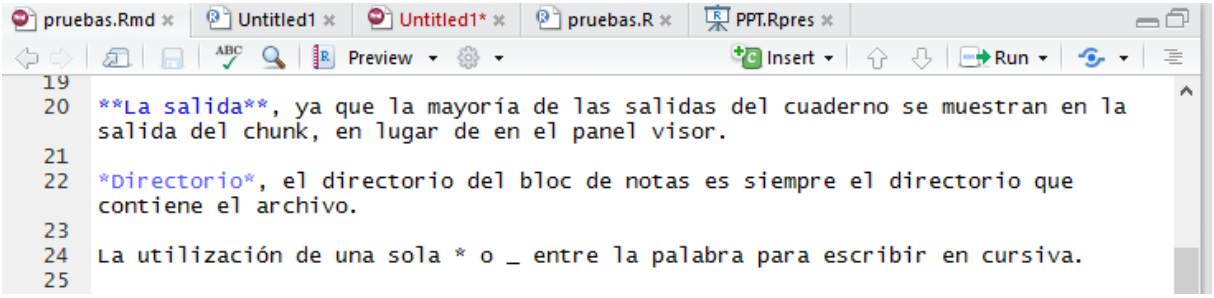


Fuente: Elaboración propia.

Algunas de las características que podemos encontrar en R Notebook es la forma de edición, R Notebook cuenta con algunas opciones de escritura que ya hemos mencionado como, por ejemplo, la utilización de ** entre la palabra para ponerlo en negrita o la utilización de una sola * o _ entre la palabra para escribir en cursiva.

En la Figura 29 podemos ver cómo se realizaría.

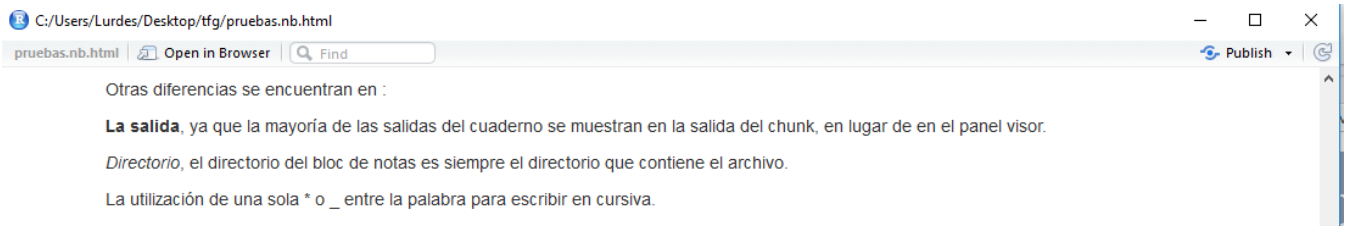
Figura 29. Realización de negritas y cursivas



Fuente: Elaboración propia.

Siendo el resultado en nuestro HTML como en la Figura 30.

Figura 30. Vista de negritas y cursivas en HTML



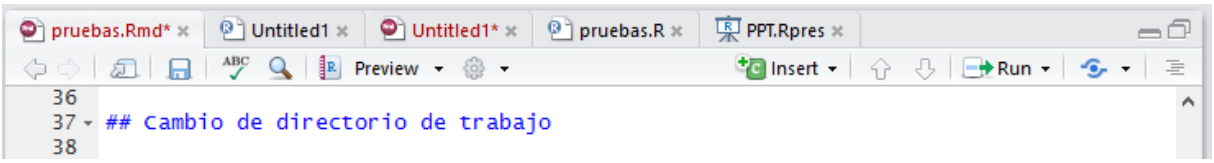
Fuente: Elaboración propia.

Como podemos observar la palabra "La salida" nos aparece en negrita, mientras que "Directorio" nos aparece en cursiva.

También permite la utilización de encabezados. Para ello debemos de usar # para el título principal, ## para el subíndice y así sucesivamente.

En la Figura 31 se ve el entorno de trabajo.

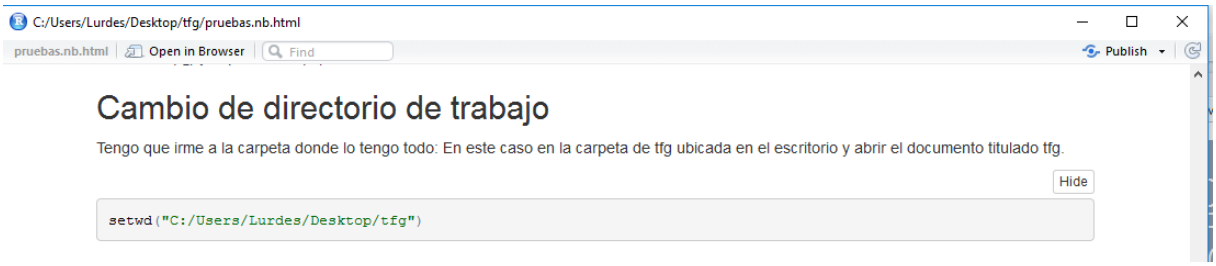
Figura 31. Realización encabezados



Fuente: Elaboración propia.

Y la Figura 32 sería nuestro resultado, de igual forma lo realizamos para el título.

Figura 32. Vista encabezados en HTML



Fuente: Elaboración propia.

Otras opciones de R son la elaboración de una lista tanto de forma ordenada como desordenada, saltos de línea manuales mediante la utilización de dos o más espacios, utilización de links, posibilidad de añadir imágenes ubicadas tanto en internet como en el equipo, también permite citar, permite la elaboración de ecuaciones en LaTeX, saltos de página, elaboración de tablas, el tachado en la escritura(~~hola~~), subíndices(x_2) y superíndices(x^2), como aparece en la Figura 33.

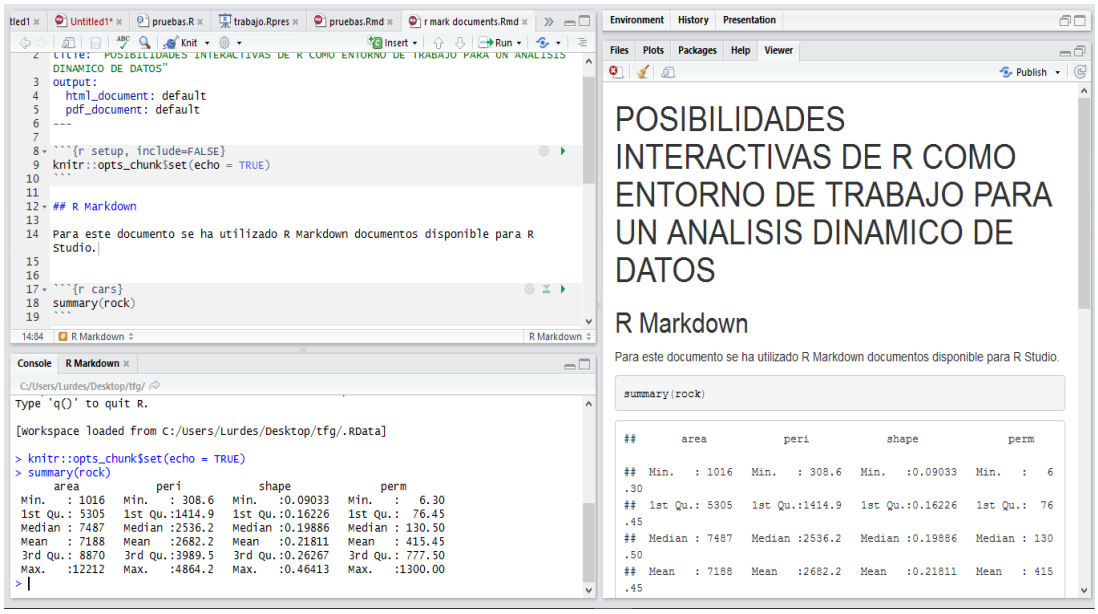
Figura 33. Opciones escritura

```
34  escritura(~~hola~~), subíndices(x~2~) y superíndices(x^2^).
35
36
```

Fuente: Elaboración propia.

En la Figura 34 podemos ver como se visualizaría nuestro R Notebook en línea desde R Studio.

Figura 34. Vista previa de R Notebook desde R Studio



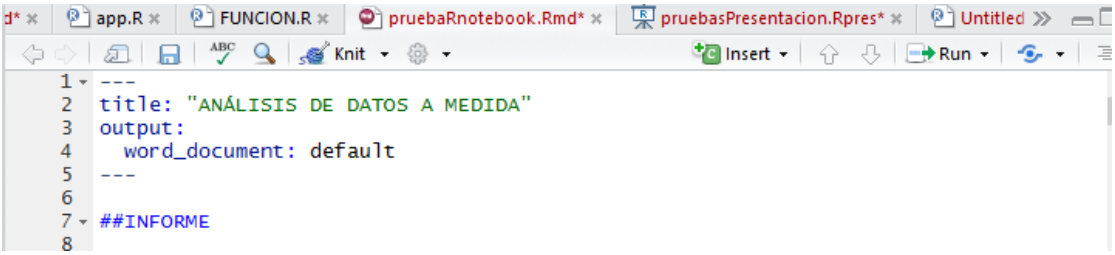
Fuente: Elaboración propia.

4.1 Ejemplo

En el ejemplo anterior, del apartado de R Markdown vimos cómo se escribe el código en los documentos R Markdown. En este ejemplo se nos muestra cómo se genera el documento R Markdown a través de R Notebook en formato Microsoft Word.

Como vimos anteriormente nuestro trabajo aparecería como en la Figura 35.

Figura 35. R Notebook



Fuente: Elaboración propia.

El cual su visualización en Word seria de la siguiente forma.

ANÁLISIS DE DATOS A MEDIDA

INFORME

Lo que nuestra empresa nos solicito fue un script aplicable a sus ficheros de datos como vimos en el capítulo 2, a través de este informe se realiza dicho script pero en forma de documento.

A partir de este informe lo que queremos conseguir es:

En primer lugar, queremos que se nos muestre una tabla para las variables cuantitativas, donde nos encontramos diferentes estadísticos, como, la media, desviación típica, varianza, percentil 25, mediana, percentil 75, asimetría y curtosis.

En segundo lugar, realizaremos las representaciones gráficas tanto para las variables cuantitativas como cualitativas.

1. TABLA

En la tabla podremos observar diferentes estadísticos como:

- Media(m_x).
- Desviación típica(s_x).
- Varianza(s^2_x).
- Percentil 25(P_{25}).
- Mediana(P_{50}).
- Percentil 75(P_{75}).
- Asimetría.
- Curtosis.

A continuación, en la Tabla 1 se nos muestran nuestros estadísticos en forma de tabla.

Tabla 1. Estadísticos

MEDIA	DT	VARIANZA	P25	MEDIANA	P75	ASIMETRIA	CURTOSIS
0.09	0.95	0.89	-0.44	0.01	0.57	0.19	0.04
0.05	0.88	0.77	-0.56	0.00	0.74	0.55	0.39

Fuente: Elaboración propia

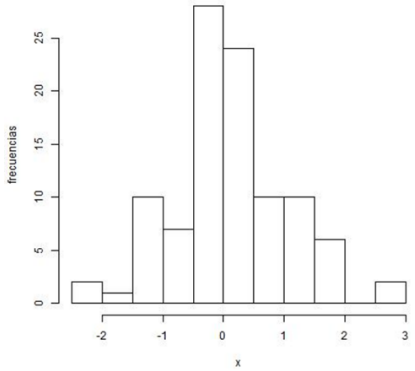
2. REPRESENTACIONES GRAFICAS

Para las variables cuantitativas realizaremos el histograma y el boxplot.

Histograma es la representación gráfica de la distribución de frecuencias en forma de barras.

En la Ilustración 1 podemos ver el histograma correspondiente a la variable x.

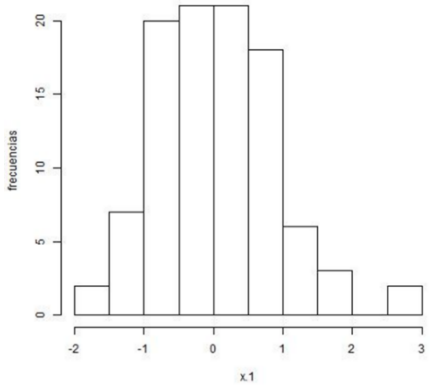
Ilustración 1. Histograma de x



Fuente: Elaboración propia

En la Ilustración 2 podemos ver el histograma correspondiente a x.1.

Ilustración 2. Histograma de x.1

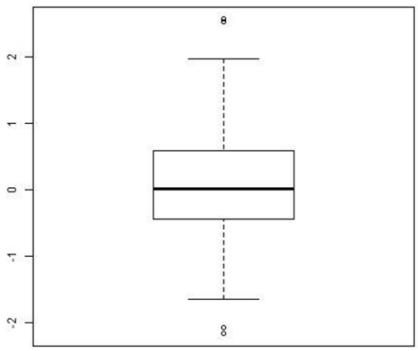


Fuente: Elaboración propia

Boxplot, o diagrama de cajas y bigotes.

En la Ilustración 3 podemos ver el boxplot correspondiente a x.

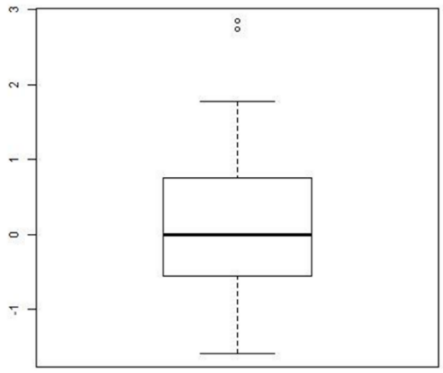
Ilustración 3. Boxplot x



Fuente: Elaboración propia

En la Ilustración 4 podemos ver el boxplot correspondiente a x.1.

Ilustración 4. Boxplot x.1



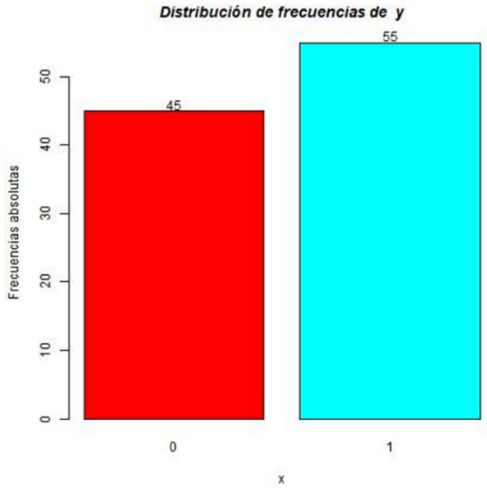
Fuente: Elaboración propia

Y, por último, para las variables cualitativas realizamos los gráficos de sectores y de barras.

Gráficos de barras

En la Ilustración 5 podemos ver el gráfico de barras correspondiente a y.

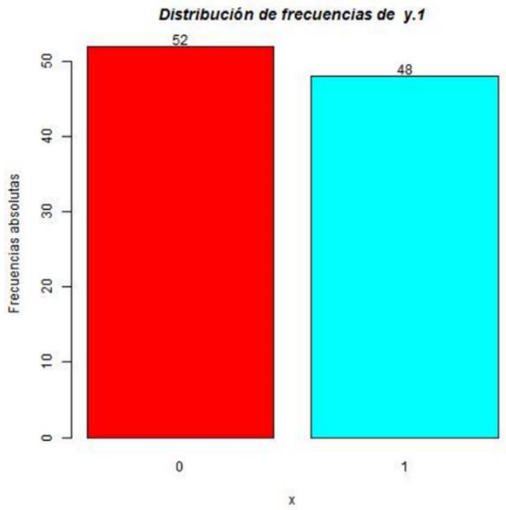
Ilustración 5. Gráfico de barras y



Fuente: Elaboración propia

En la Ilustración 6 podemos ver el grafico de barras correspondiente a y.1.

Ilustración 6. Gráfico de barras y.1

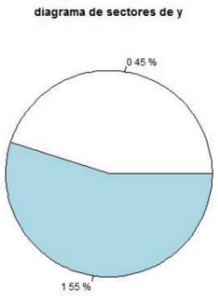


Fuente: Elaboración propia

Diagramas de sectores

En la Ilustración 7 podemos ver el diagrama de sectores correspondiente a y.

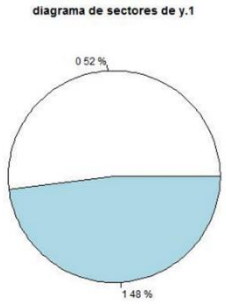
Ilustración 7. Diagrama de sectores y



Fuente: Elaboración propia

Y, por último, en la Ilustración 8, podemos ver el diagrama de sectores correspondiente a y.1.

Ilustración 8. Diagrama de barras y.1



Fuente: Elaboración propia

Como podemos observar R Notebook es una forma dinámica y relativamente sencilla de crear documentos, facilitando el trabajo en la creación de documentos ya que se puede hacer directamente desde el mismo programa.

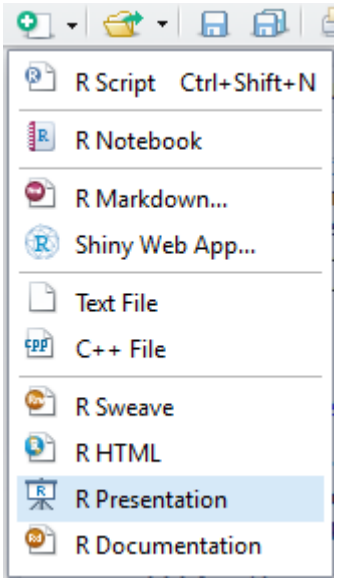
5. R PRESENTATION

Siguiendo con nuestro supuesto inicial, con motivo de poder simplificarle trabajo a la empresa la cual nos ha contratado, además de facilitarle la elaboración de documentos también es posible la elaboración presentaciones a través de R Markdown: para ello se utiliza R Presentation.

Las presentaciones son una característica de RStudio que permite la creación de presentaciones HTML de una forma sencilla, usando una combinación de R y R Markdown.

Para trabajar con las presentaciones debemos seguir los mismos pasos que para trabajar con R Notebook, como en la Figura 36.

Figura 36. R Presentation



Fuente: Elaboración propia.

Las presentaciones realizadas con R Studio cuentan con las siguientes características:

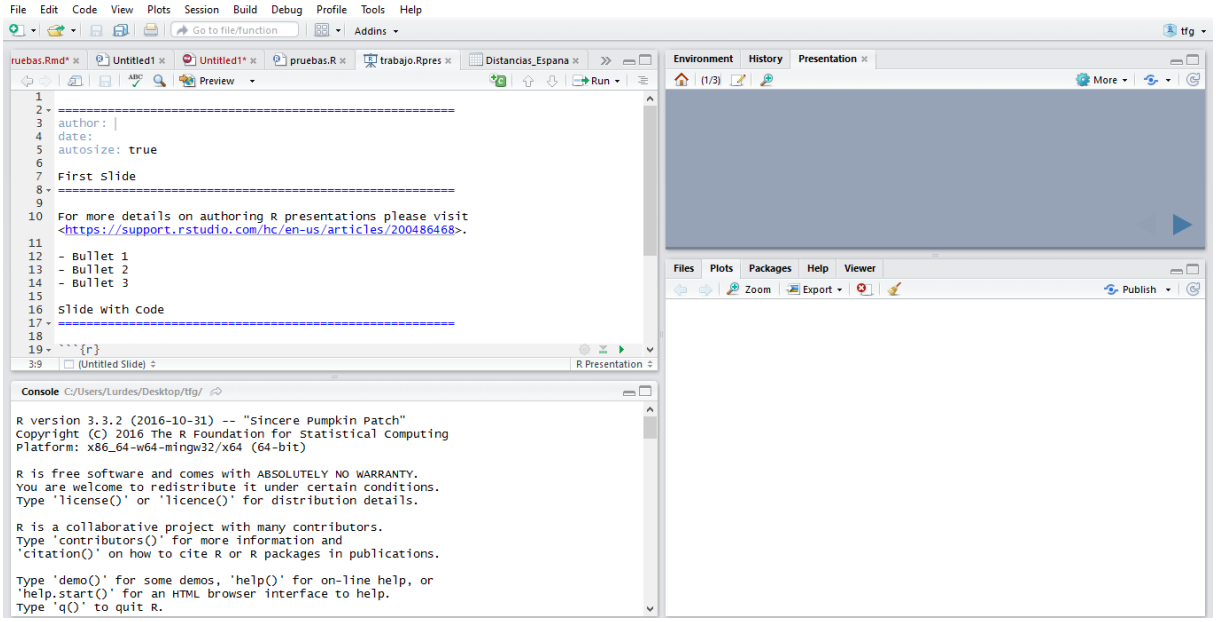
- Sintaxis muy sencilla.
- Fácil incorporación del código R.
- Soporte para las ecuaciones LaTeX usando MathJax.
- Diseño flexible.
- Numerosas opciones para las transiciones de las diapositivas.
- Posibilidad de personalizar la apariencia de las transparencias.
- Posibilidad de compatibilizar la creación y la visualización de las presentaciones.
- Se puede reproducir desde RStudio o como presentación HTML en el navegador.
- Se puede publicar fácilmente como un archivo HTML o para Rpubs[4].

[4] Es un portal de libre acceso, donde podemos encontrar trabajos realizados por otros usuarios o publicar nuestro trabajo.

El objetivo de R Presentation es crear diapositivas usando el código de R y las ecuaciones LaTeX de forma sencilla. R Presentation es especialmente útil, ya que facilita la presentación de resultados en un entorno profesional, que se puede presentar desde el mismo programa que estamos usando para trabajar y ver el código.

Una vez seleccionado R Presentation lo primero que nos encontraremos será la pantalla principal, como en la Figura 37.

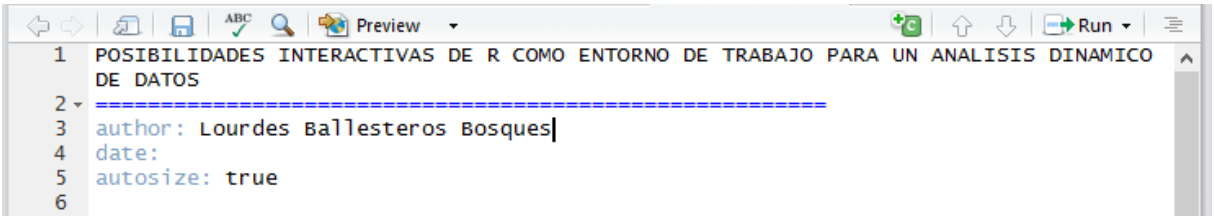
Figura 37. Pantalla principal de R Presentation



Fuente: Elaboración propia.

Para comenzar a trabajar, observamos que al igual que pasaba en R Notebook, lo primero que nos encontramos es el encabezado como en la Figura 38.

Figura 38. Encabezado R Presentation



Fuente: Elaboración propia.

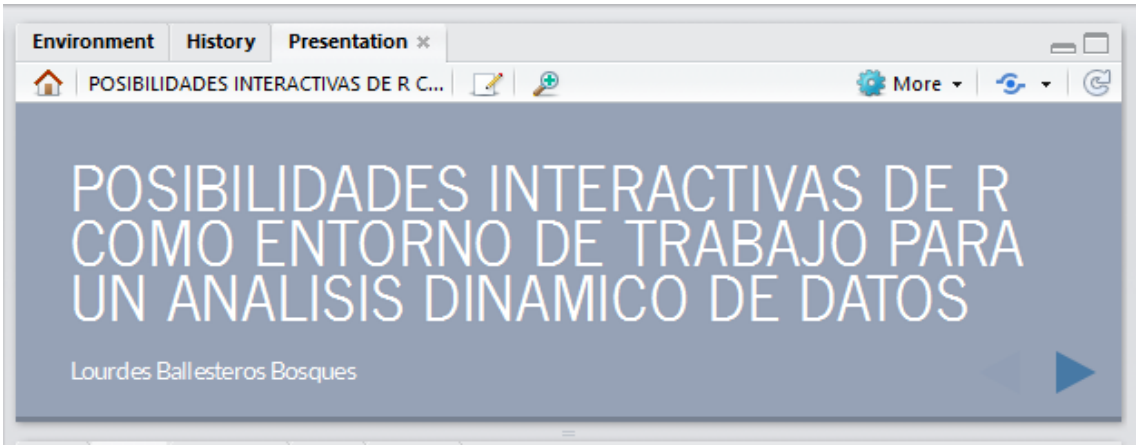
Como podemos observar en la fila 1 es donde se pone el título de nuestro trabajo, y posteriormente el autor, la fecha y por último si queremos tamaño automático.

Para la creación de presentaciones tenemos 2 opciones.

- HTML.
- PDF.

En la Figura 39 podemos ver cómo quedaría nuestra portada.

Figura 39. Visualización R Presentation



Fuente: Elaboración propia.

Como podemos observar nuestra presentación se muestra en la ventana de R donde se encuentra el historial, como se dijo anteriormente.

Si quisiéramos ampliar la visualización de la presentación, solo tenemos que pinchar en la lupa  situada en el menú de la presentación.

Por defecto R Presentation nos pone en la segunda diapositiva una página para obtener más detalles sobre la creación de presentaciones en R.

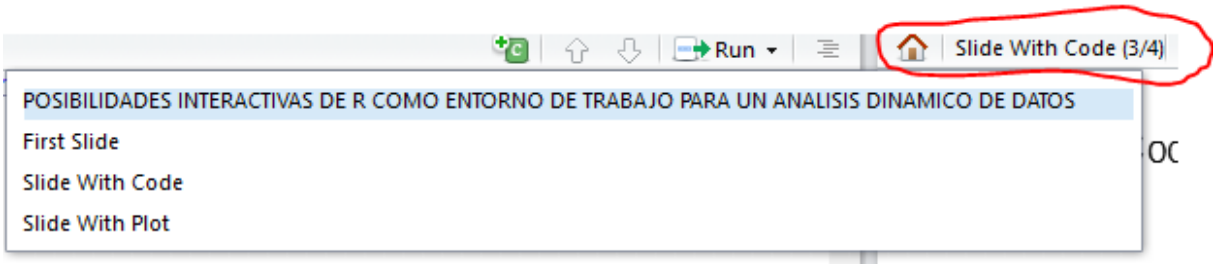
La página es la siguiente: <https://support.rstudio.com/hc/en-us/articles/200486468>

5.1 Características para R Presentation

R Studio cuenta con algunos aspectos para hacer más productivas las presentaciones, como, por ejemplo:

- Cada vez que se guarda la presentación, la vista previa se actualiza y se navega a la diapositiva que estamos trabajando y editando.
- En la parte inferior del editor de código se puede cambiar rápidamente de diapositiva.
- Si se está mirando una diapositiva en la vista previa, si se presiona el botón **editar**  situado en la barra de herramientas, se va inmediatamente a su ubicación en el archivo de origen.
- Cuenta con un menú de navegación sobre las diapositivas (Figura 40).

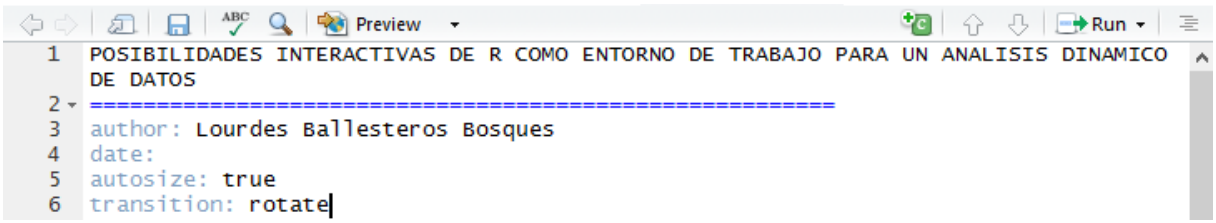
Figura 40. Navegación diapositivas



Fuente: Elaboración propia.

R Presentation cuenta con numerosas transiciones para las presentaciones. Aunque por defecto utiliza la lineal, es posible cambiar el modo de transición. Para determinar en el modo de transición, se debe escribir en el encabezado de R Presentation, de la forma en que aparece en la Figura 41.

Figura 41. Transiciones



Fuente: Elaboración propia.

Como podemos observar, ahora se ha añadido una nueva línea, titulada *transition*. En ella podemos indicar el modo de transición que queremos para nuestra presentación.

Las distintas alternativas son:

- Ninguna.
- Lineal.
- Girar.
- Descolocarse.
- Enfocar.
- Cóncavo.

Esto se aplicará a toda la presentación, pero se puede cambiar la transición de una diapositiva en concreto.

Otra opción a tener en cuenta es la velocidad, para ello utilizamos `transition-speed:` cuyas opciones son lento y rápido.

También cuenta con numerosos tipos de diapositivas y secciones. Para especificar el tipo de diapositiva utilizamos `type:` y existen 4 modelos:

- Sección (“section”).
- Subsección (“sub-section”).
- Rápido/sugerencia (“prompt”).
- Alerta (“alert”).

Debemos colocarlo de la forma en que aparece en la Figura 42.

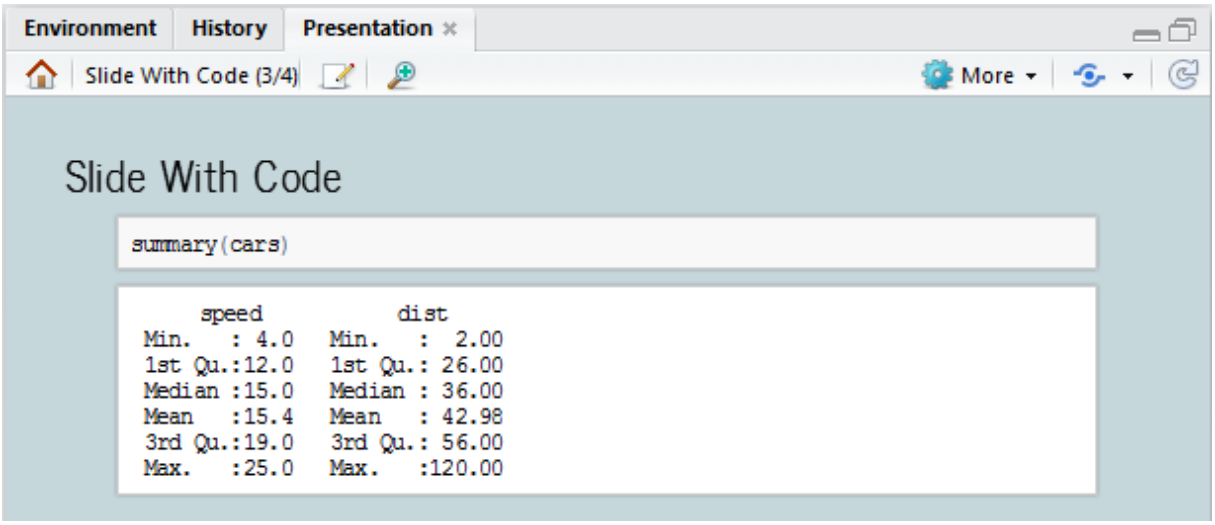
Figura 42. Tipo diapositiva

```
18  Slide with Code
19  ==========================================================
20  type: prompt
21
22  ```{r}
```

Fuente: Elaboración propia.

El resultado sería el de la Figura 43.

Figura 43. Visualización diapositiva



Fuente: Elaboración propia.

Como podemos observar, ahora la diapositiva es de azul más clarito. Cada tipo utilizará un color distinto y además en el caso de utilizar distintos tipos de diapositivas el texto será de diferente tamaño y color.

Cuenta con la opción de mostrar el texto de forma incremental. Para hacerlo se debe poner como en la Figura 44.

Figura 44. Forma de mostrar el texto

```
19  Slide with Code
20
21  ==========================================================
22  incremental: true
```

Fuente: Elaboración propia.

Para este campo solo se cuenta con dos opciones, *true* para que aparezca de forma incrementar, y *false*, que aparecerá como aparece por defecto.

Para añadir un enlace a una página de internet solo debemos escribir [linked phrase] seguido de la página web a la que queremos vincularlo como aparece en la Figura 45.

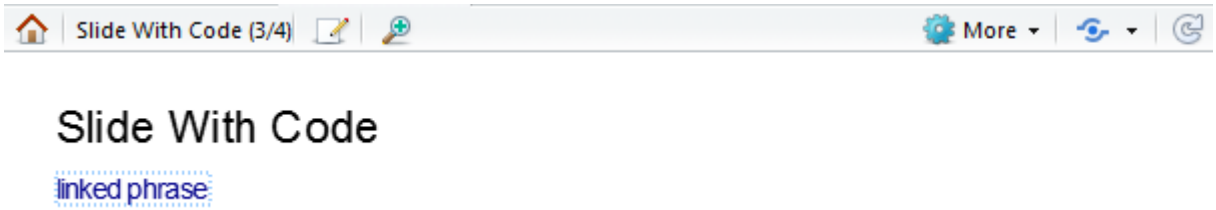
Figura 45. Poner enlaces

```
19  Slide With Code
20  ========================================================
21  [linked phrase](https://www.rstudio.com/)
22
```

Fuente: Elaboración propia.

Visualmente se verá como en la Figura 46.

Figura 46. Visualización enlaces en diapositivas



Fuente: Elaboración propia.

Y, obviamente pinchando, nos conducirá a la página que le hayamos indicado, en este caso a la de RStudio.

También cuenta con la opción de movernos a otra diapositiva, como vemos en la Figura 47.

Figura 47. Ir a otra diapositiva

```
19  Slide With Code
20  ========================================================
21  [Go to slide 1](#/First Slide)
```

Fuente: Elaboración propia.

Cuenta con la posibilidad de vincularlo a uno de nuestros archivos como en la Figura 48.

Figura 48. Ir a otro archivo

```
19  Slide With Code
20  ========================================================
21  source: pruebas.R
22
```

Fuente: Elaboración propia.

Esto nos conduciría a uno de nuestros archivos que ya tengamos creados de R.

Además, nos permite especificar que resalte una línea en concreto (Figura 49).

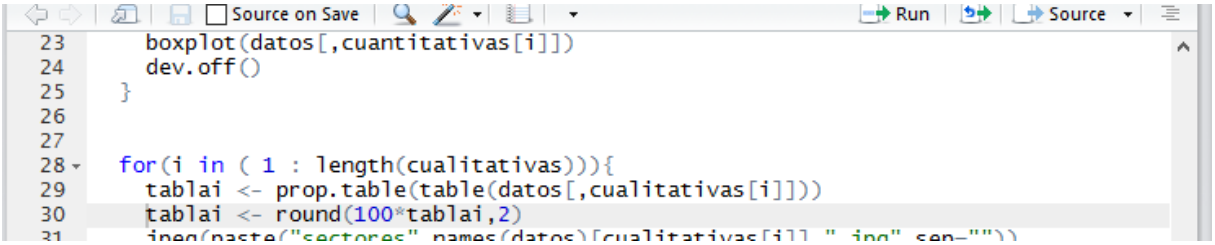
Figura 49. Resaltar una linea

```
19  Slide With Code
20  ========================================================
21  source: pruebas.R 30
```

Fuente: Elaboración propia.

Esto nos lleva a la línea 30 de nuestro archivo titulado FUNCION y nos la resalta (Figura 50).

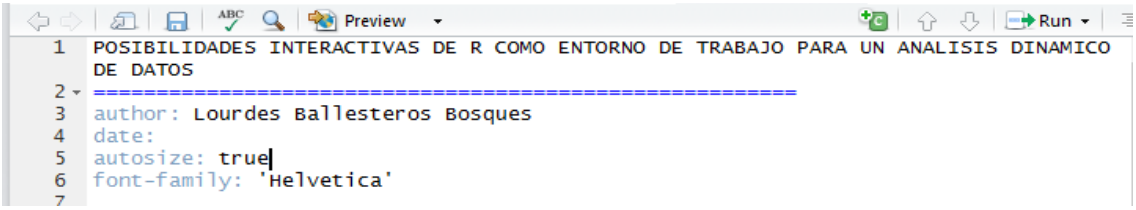
Figura 50. Línea 30 resaltada



Fuente: Elaboración propia.

Es posible cambiar la fuente: por defecto utiliza "Lato". Para cambiarla solo debemos indicarlo en el encabezado, como en la Figura 51.

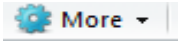
Figura 51. Fuentes

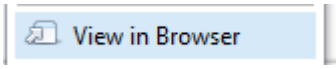


Fuente: Elaboración propia.

Por último, la visualización y distribución de las presentaciones puede ser de dos formas:

- En línea dentro de RStudio.
- En un navegador web externo.

Para verlo desde un navegador web externo solo debemos pinchar 

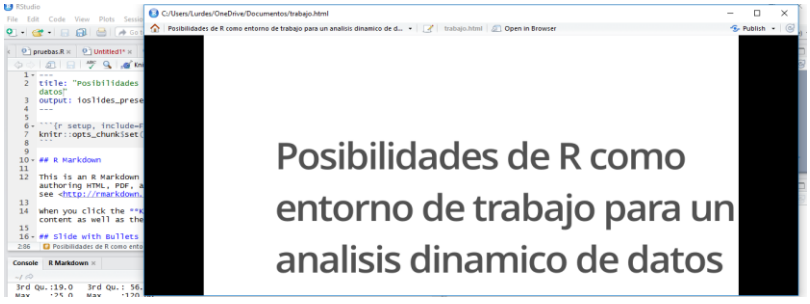
y 

ubicadas en las herramientas de presentación.

La visualización externa (Figura 52) tiene varias ventajas:

- Las transacciones de las diapositivas están animadas.
- Puede pasar al modo de pantalla completa pulsando la tecla F.
- Puede ver una vista general navegable de sus diapositivas pulsando la tecla O.

Figura 52. Visualización externa



Fuente: Elaboración propia.

5.2 Ejemplo

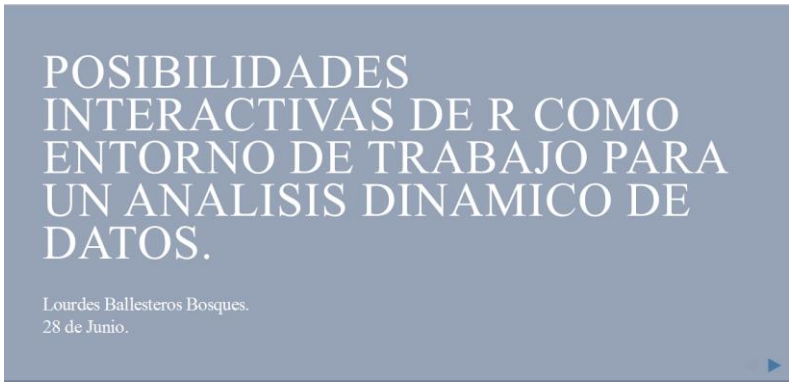
Al igual que creamos un documento a través de R Markdown, ahora vamos a realizar un ejemplo en forma de presentación a través de R Presentation.

Lo primero que vamos a ver es la forma en la que se tiene que trabajar en R Studio para la elaboración de presentaciones, al igual que para los documentos, se realiza a través de *chunk.*

```
POSIBILIDADES INTERACTIVAS DE R COMO ENTORNO DE TRABAJO PARA UN ANALISIS DINAMICO DE DATOS.
========================================================
author: Lourdes Ballesteros Bosques.
date: 28 de Junio.
autosize: true
font-family: Times New Roman
```

Hasta aquí sería la portada de nuestro trabajo, donde se podrá ver el título con el que lo hemos llamado, el autor y la fecha. También dejamos indicado la fuente de escritura que queremos, en este caso, Times New Roman (si no le indicamos la fuente que queremos utilizará por defecto "Lato", como vimos en la Figura 39) y el ajuste de tamaño automático, quedando como resultado el que aparece en la Figura 53.

Figura 53. Primera diapositiva



Fuente: Elaboración propia

Para diferenciar las diapositivas debemos colocar ====, siendo el título de la diapositiva lo que quede por encima de ella.

Destacar que, para las palabras con tildes existe un problema de codificación de fuentes por lo que debemos escribirlas de la siguiente forma:

á = á

é = é

í = í

ú= í

ó= í

Dicha escritura la utilizaremos en numerosos fragmentos de código de este ejemplo.

```
DATOS
============================================================
type: section

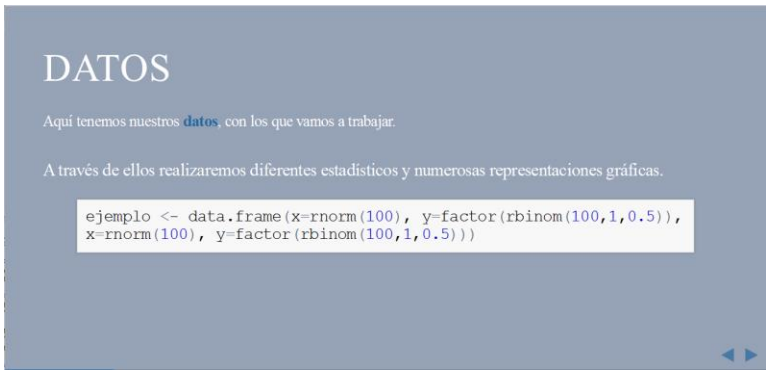

<small> Aqu&iacute; tenemos nuestro **datos**, con los que vamos a
trabajar. </small>
A trav&eacute;s de ellos realizaremos diferentes estad&iacute;sticos
y numerosas representaciones gr&aacute;ficas.
```{r}
ejemplo <- data.frame(x=rnorm(100), y=factor(rbinom(100,1,0.5)),
x=rnorm(100), y=factor(rbinom(100,1,0.5)))
```
```

Aquí podemos observar cómo indicamos que se trata de una sección a través de `type`.

En esta diapositiva también nos permite observar que la frase colocada entre <small> y </small> aparecerá más pequeña que el resto del texto de nuestra diapositiva. Y, además, la palabra datos al estar entre ** , aparecerá en negrita.

En la Figura 54 podemos ver la diapositiva de nuestros datos.

Figura 54. Segunda diapositiva



Fuente: Elaboración propia

La siguiente diapositiva será la que contenga nuestra función, en este caso el tipo de diapositiva que utilizaremos será la de subsección para diferenciarlas de las anteriores.

```
FUNCION
============================================================
type: sub-section

Como podemos ver no se muestra el c&oacute;digo ya que es el mismo
script que utilizamos en el Cap&iacute;tulo 2 de este Trabajo Fin de
Grado.
Para cargar dicho script utilizamos el comando *source* que sirve para
ejecutar el c&oacute;digo de un fichero, en este caso, **FUNCION.R**
```

```
```{r}
setwd("C:/Users/pc/Desktop/TFG")
source("FUNCION.R")
```
```

En dicha diapositiva utilizamos negritas (**) y cursivas (*).

En la Figura 55 tenemos la diapositiva que contiene nuestra función, como podemos observar esta diapositiva es azul más clarito, esto se debe a que como dijimos antes, se trata de una subsección.

Figura 55. Tercera diapositiva

Fuente: Elaboración propia

La siguiente diapositiva nos mostrara los resultados de la tabla, para dicha diapositiva utilizaremos otro tipo de diapositiva: “prompt”.

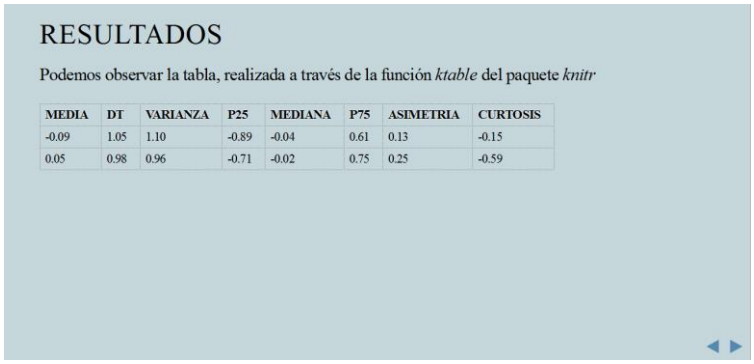
```
RESULTADOS
============================================================
type:prompt

Podemos observar la tabla, realizada a trav&eacute;s de la funci&oacute;n *ktable* del paquete *knitr*
```{r, echo=FALSE}
tabla <- round(funcion(ejemplo),2)
kable(tabla)
```
```

Como podemos ver al inicio del *chunk* dejamos indicado que echo=FALSE esto quiere decir que en la diapositiva no se mostrara el código como en las anteriores, sino que solo se mostraran la tabla correspondiente a nuestros datos.

En la Figura 56, podemos ver nuestra tabla. También podemos observar como el color de la diapositiva es diferentes a las de las demás, y, además, según el tipo de diapositiva también cambia el color de la fuente de escritura.

Figura 56. Cuarta diapositiva



Fuente: Elaboración propia

En las siguientes diapositivas realizaremos los gráficos correspondientes a nuestra función, para estas diapositivas utilizaremos otro tipo de diapositiva denominado "alert".

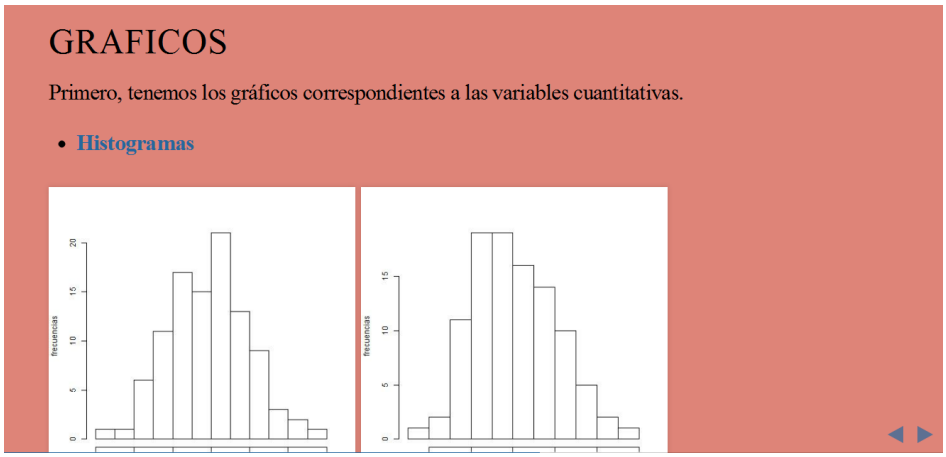
```
GRAFICOS
============================================================
type: alert

Primero, tenemos los gr&aacute;ficos correspondientes a las variables
cuantitativas.
+ **Histogramas**
![histograma](hist_x.jpg)
![histograma x.1](hist_x.1.jpg)
```

Aquí podemos observar cómo se insertan imágenes, se realiza de igual forma que en R Notebook, en primer lugar, se pone un símbolo de exclamación (!), seguido del nombre, y por último el nombre de nuestra imagen, ya que estamos situados en nuestro directorio de no ser así habría que indicar la ruta.

En la Figura 57 podemos ver como ahora la diapositiva es de color rojo. En esta diapositiva se muestra la palabra **histogramas** en negrita y en forma de lista mediante la utilización del símbolo de sumar (+).

Figura 57. Quinta diapositiva



Fuente: Elaboración propia

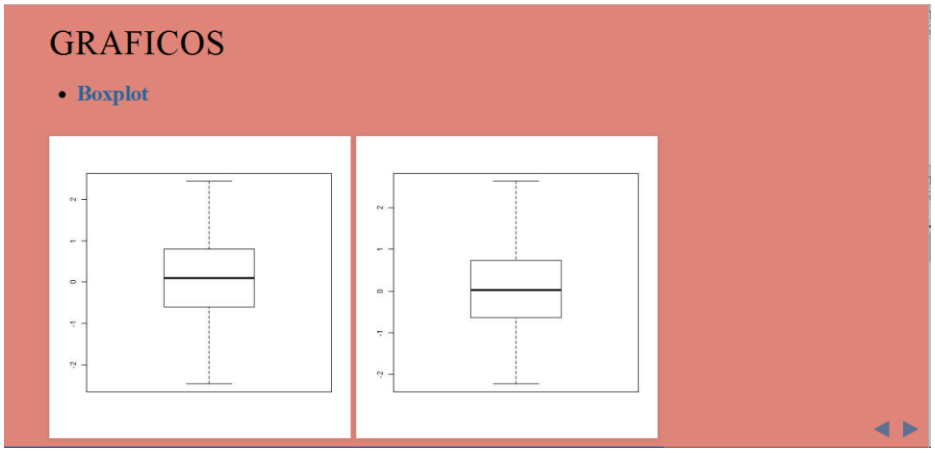
La siguiente diapositiva está realizada igual que la anterior, en ella tenemos los gráficos de cajas y bigotes.

```
GRAFICOS
============================================================
type: alert

* **Boxplot**
![boxplot x](boxplot_x.jpg)
![boxplot x.1](boxplot_x.1.jpg)
```

En la Figura 58 se muestra los gráficos de cajas y bigotes donde la palabra boxplot está editada de igual forma que histogramas en la diapositiva anterior, es decir, en negrita.

Figura 58.Sexta diapositiva



Fuente: Elaboración propia

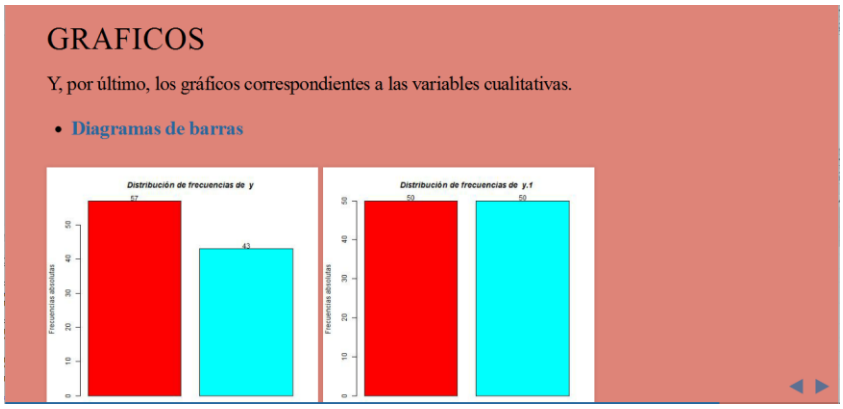
En las siguientes diapositivas se nos mostraran los gráficos correspondientes a las variables cualitativas.

```
GRAFICOS
============================================================
type: alert

Y, por &uacute;ltimo, los gr&aacute;ficos correspondientes a las
variables cualitativas.
* **Diagramas de barras**
![barras de y](barrasy.jpg)
![barras de y.1](barrasy.1.jpg)
```

En la Figura 59 podemos observar los diagramas de barras correspondientes a nuestros datos.

Figura 59. Séptima diapositiva



Fuente: Elaboración propia

Y, por último, elaboramos los diagramas de sectores para las variables cualitativas.

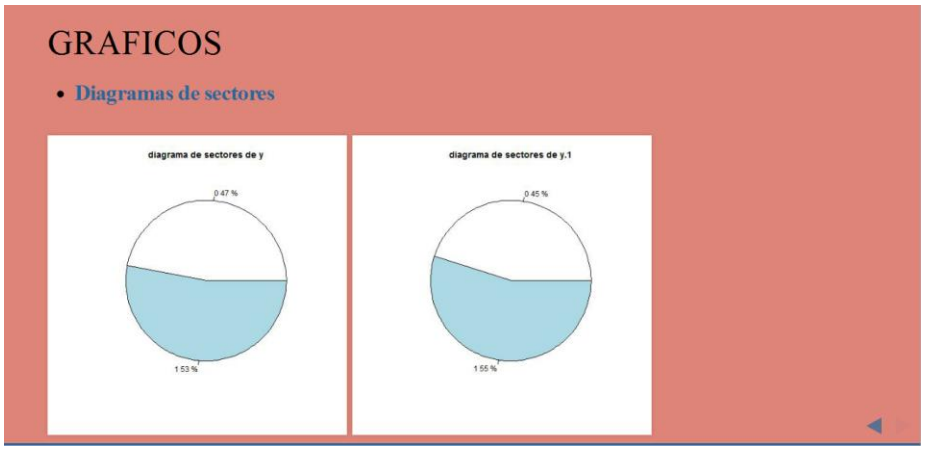
```
GRAFICOS
============================================================
type: alert

* **Diagramas de sectores**

![sectores y](sectoresy.jpg)
![sectores y.1](sectoresy.1.jpg)
```

En la Figura 60 podemos observar nuestra última diapositiva que se corresponde con los diagramas de sectores.

Figura 60. Ultima diapositiva



Fuente: Elaboración propia

6 SHINY APPS

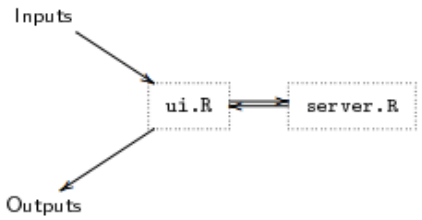
Shiny es un entorno de trabajo de R Studio para realizar aplicaciones webs en R. Lo que hace es convertir los scripts de R en aplicaciones web interactivas que cualquiera puede usar.

Las apps de Shiny están formados por dos archivos:

- Un script para la interfaz del usuario que recibe los inputs y muestra los outputs (ui. R)
- Un script donde se realizan los cálculos (server. R)

En la Figura 61 podemos ver un esquema de los outputs e inputs de Shiny.

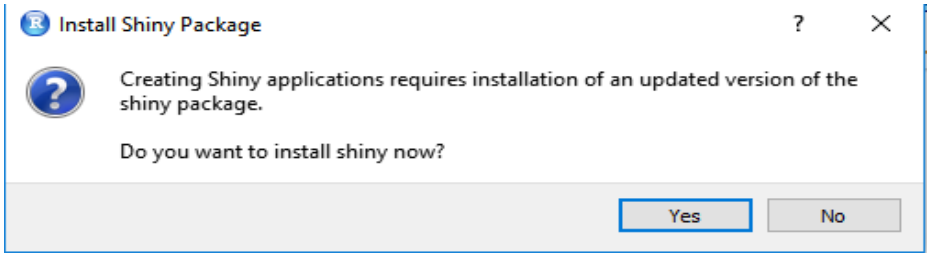
Figura 61. Esquema Shiny



Fuente: https://rua.ua.es/dspace/bitstream/10045/54325/1/shiny.pdf

Al igual que con los anteriores para trabajar con Shiny debemos pulsar Shiny Web App... , al pulsarlo lo primero que nos aparece es la ventana de la Figura 62.

Figura 62. Instalacion Shiny

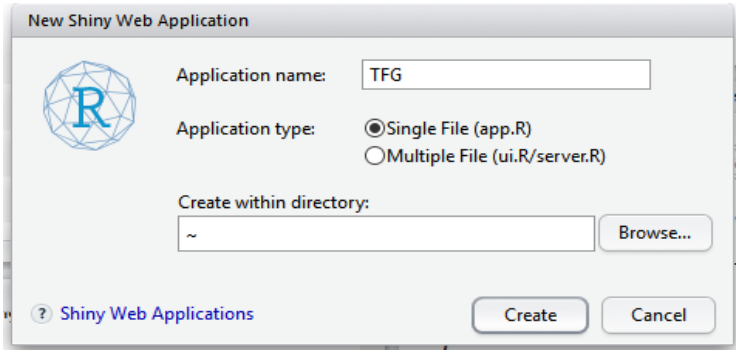


Fuente: Elaboración propia

Quiere decir que debemos instalar un paquete para poder trabajar con Shiny.

Una vez instalado el paquete nos aparece la siguiente ventana de la Figura 63.

Figura 63. Nombre aplicación

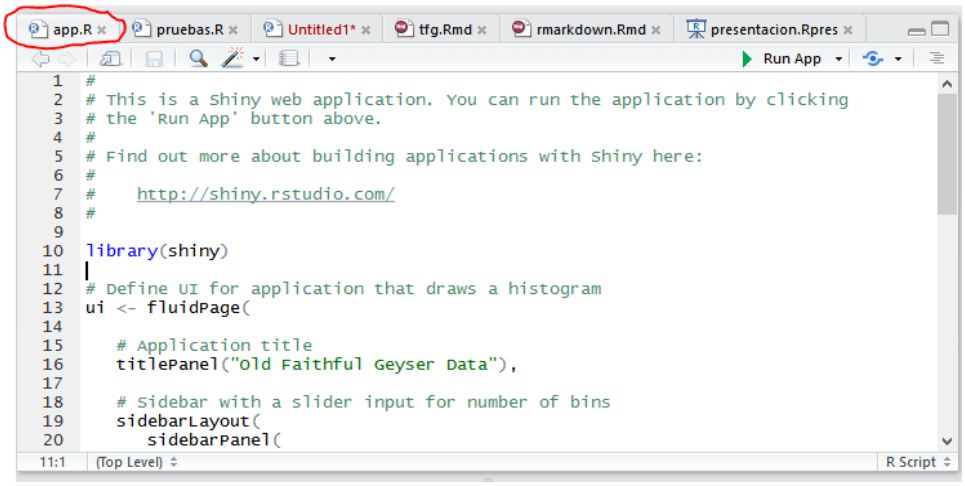


Fuente: Elaboración propia

Aquí indicaremos el nombre de nuestra aplicación, el tipo y el directorio.

Y a continuación se nos abre la siguiente pestaña de la Figura 64.

Figura 64. Pantalla principal Shiny



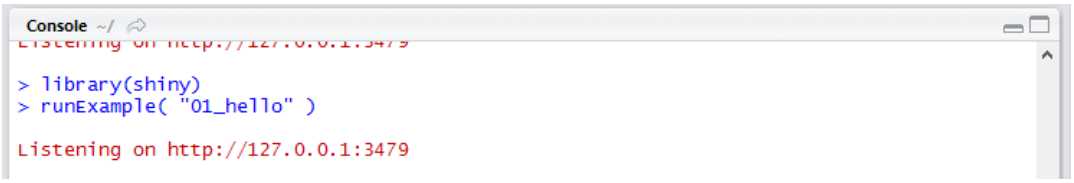
Fuente: Elaboración propia

A partir de aquí ya podemos, empezar a trabajar.

Como vemos, RStudio nos muestra una página donde encontrar información acerca de Shiny. Cuenta con algunos ejemplos, para poder ver cuáles son sus características, los cuales ahora vamos a ver.

Empezaremos con el ejemplo de Hello Shiny para ello debemos escribirlo en la consola, de la forma en que aparece en la Figura 65.

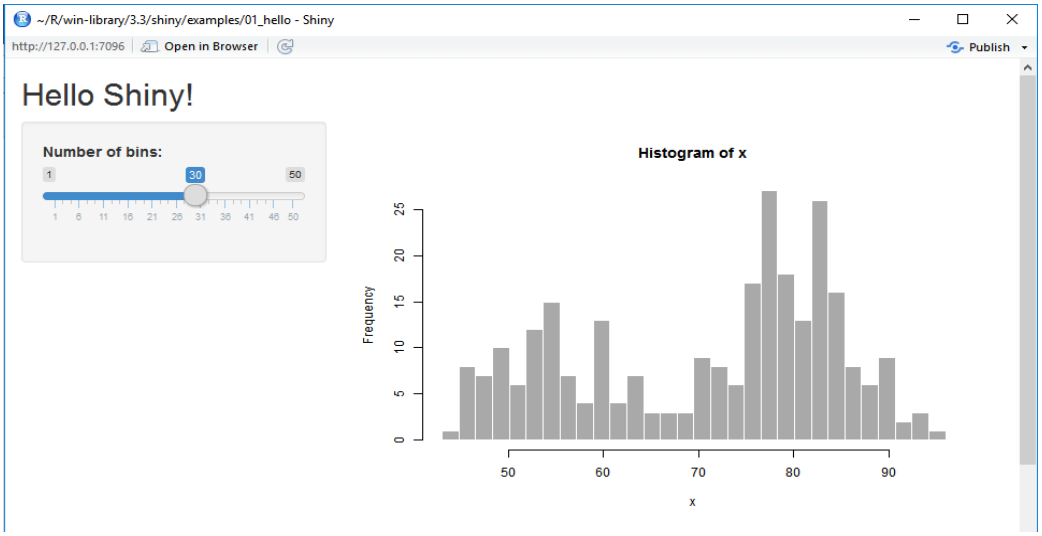
Figura 65. Ejemplo Hello Shiny



Fuente: Elaboración propia.

y nos aparece el ejemplo de la Figura 66.

Figura 66. Resultado ejemplo Hello Shiny



Fuente: Elaboración propia

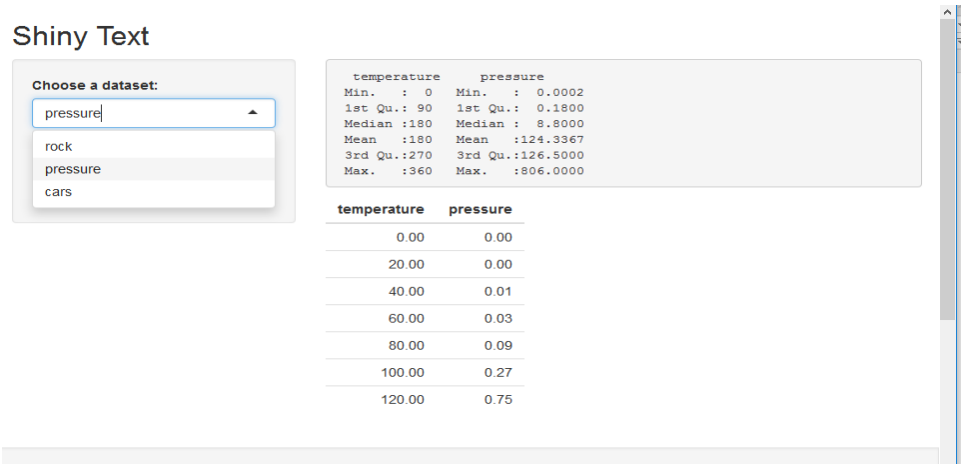
En este ejemplo podemos ver un histograma y nos permite seleccionar el número de barras que queremos que nos muestre.

Otro ejemplo es el siguiente:

```
> runExample ( "02_text" )
```

En la Figura 67 podemos ver el ejemplo de Shiny text.

Figura 67. Ejemplo Shiny text



Fuente: Elaboración propia

En el podemos ver como Shiny permite la visualización de tablas en HTML. Además, cuenta con las posibilidades de seleccionar los datos que queremos y elegir el número de observaciones que queremos que nos muestre.

En el podemos ver la secuencia de comando (ui. R) que controla el aspecto y diseño de la aplicación. (Figura 68).

Figura 68. Interfaz Ui.R

server.R ui.R ↴ show below

```
library(shiny)

# Define UI for dataset viewer application
fluidPage(

  # Application title
  titlePanel("Shiny Text"),

  # Sidebar with controls to select a dataset and specify the
  # number of observations to view
  sidebarLayout(
    sidebarPanel(
      selectInput("dataset", "Choose a dataset:",
                  choices = c("rock", "pressure", "cars")),

      numericInput("obs", "Number of observations to view:", 10)
    ),

    # Show a summary of the dataset and an HTML table with the
    # requested number of observations
    mainPanel(
      verbatimTextOutput("summary"),

      tableOutput("view")
    )
  )
)
```

Fuente: Elaboración propia

Y también observamos el script que contiene las instrucciones para realizar los cálculos (server.R) en la Figura 69.

Figura 69. El script (server.R)

server.R ui.R ↴ show below

```
library(shiny)
library(datasets)

# Define server logic required to summarize and view the selected
# dataset
function(input, output) {

  # Return the requested dataset
  datasetInput <- reactive({
    switch(input$dataset,
           "rock" = rock,
           "pressure" = pressure,
           "cars" = cars)
  })

  # Generate a summary of the dataset
  output$summary <- renderPrint({
    dataset <- datasetInput()
    summary(dataset)
  })

  # Show the first "n" observations
  output$view <- renderTable({
    head(datasetInput(), n = input$obs)
  })
}
```

Fuente: Elaboración propia

6.1 Herramientas de Shiny

Shiny cuenta con 11 buenos ejemplos que son los siguientes:

```
runExample("01_hello") # Un histograma

runExample("02_text") # Tablas y estructura de datos

runExample("03_reactivity") # Reactividad

runExample("04_mpg") # Variables globales

runExample("05_sliders") # Barras deslizantes

runExample("06_tabsets") # Paneles con pestañas

runExample("07_widgets") # Más widgets

runExample("08_html") # Shiny app construida a partir de HTML

runExample("09_upload") # Carga de archivos

runExample("10_download") # Descarga de archivos

runExample("11_timer") # Temporizador automático
```

A partir de dichos ejemplos nos encontramos con una serie de herramientas. como, por ejemplo:

- Diferentes deslizadores. Permiten:
  - Introducir valores y rangos.
  - Formatos personalizados de visualización.
- Tablas interactivas. Aquí podemos observar varias cosas en primer lugar podemos seleccionar el número de entradas, permite ordenar los valores y cuenta con la opción de búsqueda.
- Widgets.
- Subir archivos.
- Descargar archivos.
- HTML.

Shiny, al igual que R Markdown, también cuenta con una ayuda que podemos encontrar en el menú de RStudio, llamada Interactive Web Apps, como podemos observar en la Figura 70.

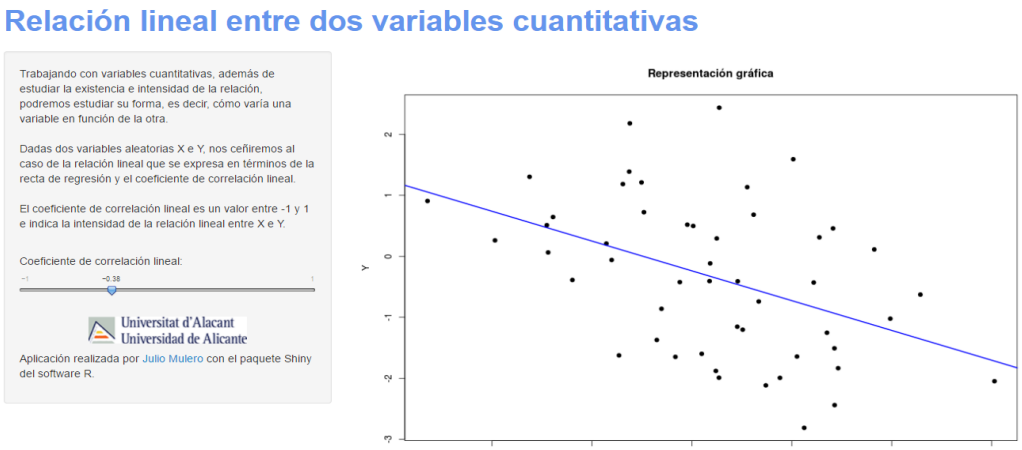
Figura 70. Ayuda Shiny

Fuente: Elaboración propia

La importancia de Shiny radica en la interactividad con la que cuentan sus aplicaciones, ya que permite manipular los datos sin tener que manipular el código.

Las aplicaciones creadas con Shiny pueden estar enfocadas a numerosos ámbitos, pero es especialmente útil en la docencia, ya que facilita el aprendizaje, por ejemplo, de la forma en la que aparece en la Figura 71.

Figura 71. Ejemplo Shiny



Fuente:https://juliomulero.shinyapps.io/correlacion/

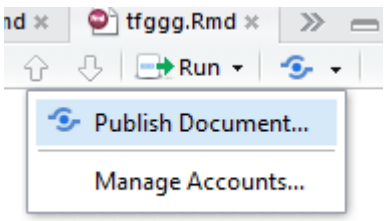
En este ejemplo se puede observar la relación lineal entre dos variables cuantitativas. Gracias a Shiny se puede observar cómo va cambiando la representación gráfica según el coeficiente de correlación lineal, el cual indica la intensidad de la relación lineal entre X e Y.

Por último, una vez creado el documento interactivo tenemos la opción de publicarlos, con ellos contamos con dos opciones

1. ShinyApps.io. Para documentos interactivos en el servidor de RStudio.
2. R Pubs.com. Que se utiliza para compartir documentos no interactivos en el sitio de publicación de Markdown de RStudio, sin costo.

Para publicar en R Pubs es muy sencillo, solo tenemos que pulsar en Publish Document como en la Figura 72.

Figura 72. Publicar R Pubs



Fuente: Elaboración propia

7 CONCLUSIONES

En los apartados anteriores hemos visto algunas de las posibilidades con las que cuenta RStudio, posiblemente desconocidas para muchos de sus usuarios. Con este trabajo lo que se ha pretendido es conocer más este software estadístico utilizado por muchos, pero del que pocos explotan sus numerosas posibilidades.

Más concretamente, en este trabajo hemos tratado de demostrar la facilidad con que se pueden crear documentos, presentaciones y apps a través de RStudio. Estas posibilidades facilitan el trabajo tanto en el ámbito empresarial como en la docencia. En este sentido, algunas de las dificultades que nos encontramos a lo largo del Grado es la elaboración de informes, en los cuales se nos pedía mostrar tanto el código como los resultados: esto dificultaba nuestro trabajo. Pero gracias a R Markdown este problema se puede solventar, ya que el mismo programa estadístico facilitará el documento de Word o de otro formato donde se aparecerán ambas cosas perfectamente diferenciadas. Esta utilidad se puede extrapolar al ámbito empresarial o profesional, ya que el objetivo principal de este trabajo fue la elaboración de un análisis estadístico a través de un script para una empresa y, una vez realizado el script, se procedió a la realización del informe y la presentación, todo ello se realizó de forma sencilla y dinámica a través de R Markdown.

8 BIBLIOGRAFÍA

- Braun, W.J. and Murdoch, D.J. (2007), *A first course in statistical programming with R*, New York: Cambridge, pp. 47-80

- Beeley, C. (2016), *Web Application Development with R Using Shiny - Second Edition*, Reino Unido, Packt Publishing.

- Contreras García, J.M., Molina Portillo, E. y Arteaga Cezón, P. (2010), *Introducción a la programación estadística con R para profesores*, Granada, Grupo de Educación Estadística, Universidad de Granada, pp. 1-92

- Gorgas García, J., Cardiel López, N. y Zamorano Calvo, J. (2011), *Estadística básica para estudiantes de ciencias*, España, Facultad de Ciencias Físicas, Universidad Complutense de Madrid

- Kabacoff, R. (2015), *R in Action, Second Edition: Data analysis and graphics with R*, Estados Unidos, Manning Publications, pp. 1-30

- Raja, K. and Sharan, R. (2016), *R Data Science Essentials*, Reino Unido, Packt Publishing.

- Muenchen, R. (2015), "The Popularity of Data Science Software", *r4stats*. Disponible online: http://r4stats.com/articles/popularity/

- Zuur, A.F., Leno, E.N. and Meesters, E.H.W.G (2009), *A Beginners´s Guide to R*, New York: Springer, pp. 127-167

ANEXO 1

En este anexo podemos ver la función en el script que facilitaríamos a la empresa, que ya describimos en el apartado ANÁLISIS A MEDIDA. FUNCIONES. SCRIPTS

```
funcion <- function(datos){
  library(xtable)
  library(e1071)
  cualitativas <- c()
  cuantitativas <- c()
  for(i in (1 : ncol(datos))){
    if(is.numeric(datos[, i])){
      cuantitativas <- c(cuantitativas, i)
    }
    if(is.factor(datos[, i])){
      cualitativas <- c(cualitativas, i)
    }
  }
  tabla <- data.frame(MEDIA = numeric(length(cuantitativas)), DT =
numeric(length(cuantitativas)), VARIANZA =
numeric(length(cuantitativas)), P25 =
numeric(length(cuantitativas)), MEDIANA =
numeric(length(cuantitativas)), P75 =
numeric(length(cuantitativas)), ASIMETRIA =
numeric(length(cuantitativas)), CURTOSIS =
numeric(length(cuantitativas)))
  for(i in (1 : length(cuantitativas))){
    tabla[i,] <- c(mean(datos[, cuantitativas[i]]), sd(datos[,
cuantitativas[i]]), var(datos[, cuantitativas[i]]), quantile(datos[,
cuantitativas[i]], 0.25), median(datos[, cuantitativas[i]]),
quantile(datos[, cuantitativas[i]], 0.75), skewness(datos[,
cuantitativas[i]]), kurtosis(datos[, cuantitativas[i]]))
jpeg(paste("hist_", names(datos)[cuantitativas[i]],".jpg",sep = ""))
hist(datos[,    cuantitativas[i]],    main    =    "",    xlab    =
```

```
names(datos)[cuantitativas[i]], ylab = "frecuencias")
    dev.off() jpeg(paste("boxplot_", names(datos)[cuantitativas[i]],
".jpg", sep = ""))
    boxplot(datos[, cuantitativas[i]])
    dev.off()
  }
  for(i in ( 1 : length(cualitativas))){
    tablai <- prop.table(table(datos[,cualitativas[i]]))
    tablai <- round(100*tablai,2)
    jpeg(paste("sectores", names(datos)[cualitativas[i]], ".jpg", sep
= ""))
    sectores<-pie(prop.table(table(datos[, cualitativas[i]])),labels
= paste(names(tablai), tablai, "%"), main = paste("diagrama de
sectores de", names(datos)[cualitativas[i]]))
    dev.off()
    tablai <- table(datos[, cualitativas[i]])
    jpeg(paste("barras", names(datos)[cualitativas[i]], ".jpg", sep =
""))
    barras<-barplot(tablai, col = rainbow(length(names(tablai))),
xlab = "x", ylab = "Frecuencias absolutas")
    text(barras, tablai + 1, labels = tablai, xpd = TRUE)
    title(main = paste("Distribución de frecuencias de ",
names(datos)[cualitativas[i]]), font.main = 4)
    dev.off()
  }
write(print(xtable(tabla, digit = 2), type = "html"), file =
"tabla.htm")
}
```

ANEXO 2

Informe realizado a través de R Markdown, visto en el capítulo: GENERACIÓN DINÁMICA DE INFORMES MEDIANTE R MARKDOWN

```
---
title: "ANÁLISIS DE DATOS A MEDIDA"
output:
  word_document: default
---
##INFORME
Lo que nuestra empresa nos solicito fue un script aplicable a sus ficheros de datos como vimos en el capítulo 2, a través de este informe se realiza dicho script pero en forma de documento.
A partir de este informe lo que queremos conseguir es:
**En primer lugar**, queremos que se nos muestre una tabla para las variables cuantitativas, donde nos encontramos diferentes estadísticos, como, la media, desviación típica, varianza, percentil 25, mediana, percentil 75, asimetría y curtosis.
**En segundo lugar**, realizaremos las representaciones gráficas tanto para las variables cuantitativas como cualitativas.
###1. TABLA
En la tabla podremos observar diferentes estadísticos como:
* Media(m~x~).
* Desviación típica(s~x~).
* Varianza(s^2^~x~).
* Percentil 25(P~25~).
* Mediana(P~50~).
* Percentil 75(P~75~).
* Asimetría.
* Curtosis.
A continuación, en la Tabla 1 se nos muestran nuestros estadísticos en forma de tabla.
*Tabla 1. Estadísticos*
```{r, echo=FALSE}
library(knitr)
setwd("C:/Users/pc/Desktop/TFG")
source("FUNCION.R")
ejemplo  <-  data.frame(x=rnorm(100),  y=factor(rbinom(100,1,0.5)), x=rnorm(100), y=factor(rbinom(100,1,0.5)))
tabla <- round(funcion(ejemplo), 2)
kable(tabla)
```
Fuente: Elaboración propia

###2. REPRESENTACIONES GRAFICAS
Para las variables cuantitativas realizaremos el histograma y el boxplot.
Histograma es la representación gráfica de la distribución de
```

frecuencias en forma de barras.
En la Ilustración 1 podemos ver el histograma correspondiente a la variable x.
Ilustración 1. Histograma de x
![histograma](hist_x.jpg)
 Fuente: Elaboración propia

En la Ilustración 2 podemos ver el histograma correspondiente a x.1.
Ilustración 2. Histograma de x.1
![histograma x.1](hist_x.1.jpg)
Fuente: Elaboración propia

Boxplot, o diagrama de cajas y bigotes.
En la Ilustración 3 podemos ver el boxplot correspondiente a x.
Ilustración 3. Boxplot x
![boxplot x](boxplot_x.jpg)
Fuente: Elaboración propia

En la Ilustracion 4 podemos ver el boxplot correspondiente a x.1.
Ilustración 4. Boxplot x.1
![boxplot x.1](boxplot_x.1.jpg)
Fuente: Elaboración propia

Y, por último, para las variables cualitativas realizamos los gráficos de sectores y de barras.
Gráficos de barras
En la Ilustración 5 podemos ver el gráfico de barras correspondiente a y.
Ilustración 5. Gráfico de barras y
![barras de y](barrasy.jpg)
Fuente: Elaboración propia

En la Ilustración 6 podemos ver el grafico de barras correspondiente a y.1.
Ilustración 6. Gráfico de barras y.1
![barras de y.1](barrasy.1.jpg)
Fuente: Elaboración propia

Diagramas de sectores
En la Ilustración 7 podemos ver el diagrama de sectores correspondiente a y.
Ilustración 7. Diagrama de sectores y
![sectores y](sectoresy.jpg)
Fuente: Elaboración propia

Y, por último, en la Ilustración 8, podemos ver el diagrama de sectores correspondiente a y.1.
Ilustración 8. Diagrama de barras y.1
![sectores y.1](sectoresy.1.jpg)

ANEXO 3

Aquí podemos ver la presentación realizada a través de R Presentation, como vimos en el capítulo: R PRESENTATION.

```
POSIBILIDADES INTERACTIVAS DE R COMO ENTORNO DE TRABAJO PARA UN ANALISIS DINAMICO DE DATOS.
========================================================
author: Lourdes Ballesteros Bosques.
date: 28 de Junio.
autosize: true
font-family: Times New Roman


DATOS
========================================================
type: section


<small> Aqu&iacute; tenemos nuestros **datos**, con los que vamos a trabajar. </small>

A trav&eacute;s de ellos realizaremos diferentes estad&iacute;sticos y numerosas representaciones gr&aacute;ficas.

```{r}
ejemplo  <-  data.frame(x=rnorm(100),  y=factor(rbinom(100,1,0.5)), x=rnorm(100), y=factor(rbinom(100,1,0.5)))
```


FUNCION
==
type: sub-section

Como podemos ver no se muestra el código ya que es el mismo script que utilizamos en el Capítulo 2 de este Trabajo Fin de Grado.
Para cargar dicho script utilizamos el comando *source* que sirve para ejecutar el código de un fichero, en este caso, **FUNCION.R**

```{r}
setwd("C:/Users/pc/Desktop/TFG")
source("FUNCION.R")
```


RESULTADOS
==
type:prompt
```

```
Podemos observar la tabla, realizada a trav&eacute;s de la funci&oacute;n *ktable* del paquete *knitr*

```{r, echo=FALSE}
tabla <- round(funcion(ejemplo),2)
kable(tabla)
```

GRAFICOS
===
type: alert

Primero, tenemos los gráficos correspondientes a las variables cuantitativas.

+ **Histogramas**

![histograma](hist_x.jpg)
![histograma x.1](hist_x.1.jpg)

GRAFICOS
===
type: alert

* **Boxplot**

![boxplot x](boxplot_x.jpg)
![boxplot x.1](boxplot_x.1.jpg)

GRAFICOS
===
type: alert

Y, por último, los gráficos correspondientes a las variables cualitativas.

* **Diagramas de barras**

![barras de y](barrasy.jpg)
![barras de y.1](barrasy.1.jpg)

GRAFICOS
===
type: alert

* **Diagramas de sectores**
![sectores y](sectoresy.jpg)
![sectores y.1](sectoresy.1.jpg)
```