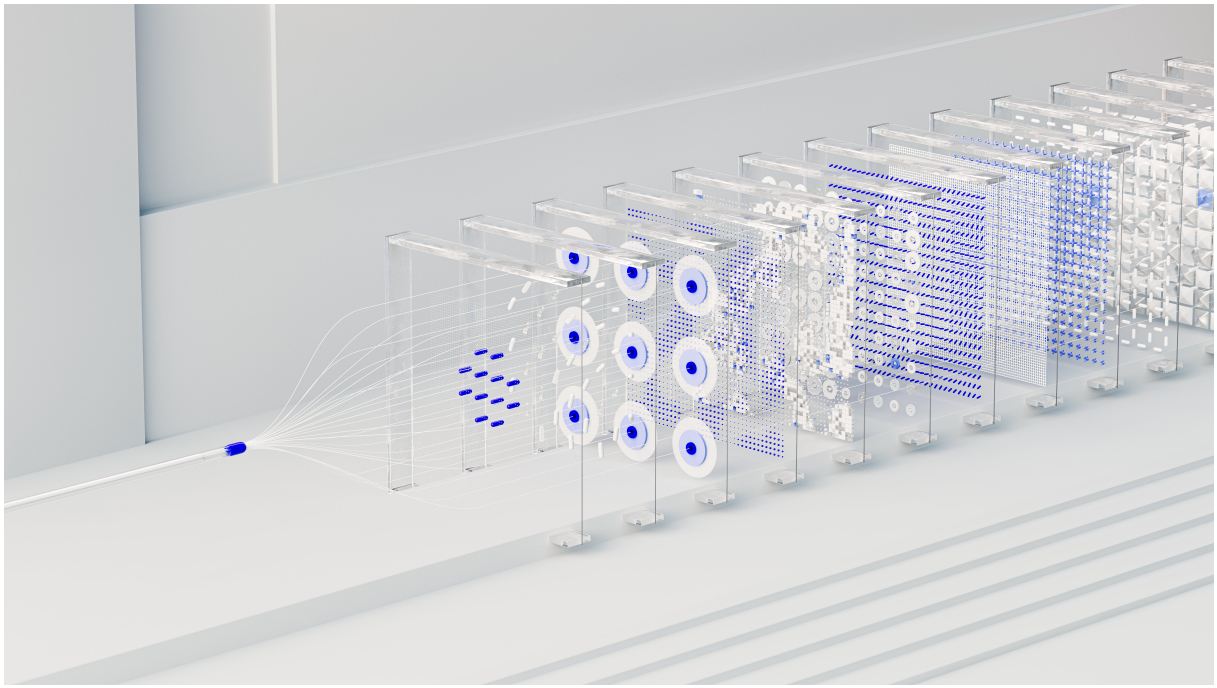
CASO PRÁCTICO 101 HERRAMIENTAS PARA EL CIENTÍFICO DE DATOS



R COMO HERRAMIENTA DEL CIENTÍFICO DE DATOS
CIENCIA DE DATOS PARA NEGOCIOS

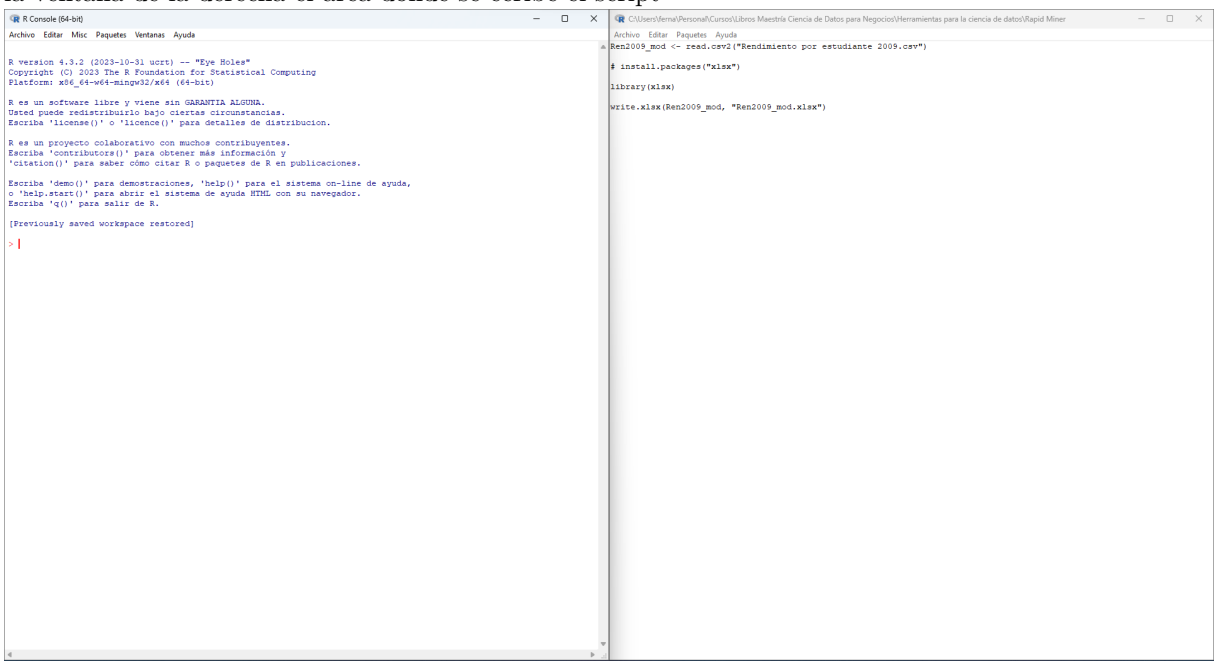
Fernando Aleisy González
22 de enero de 2024

Índice

Descripción de la Herramienta

R es un lenguaje que fue desarrollado por estadísticos como un entorno para el análisis de datos, lo que implica que no es un lenguaje para el desarrollo de software. La interactividad que nos ofrece R es fundamental para el éxito en el campo del análisis de los datos ya que esta permite una rápida exploración de los datos (Irizarry, 2019).

Figura 1: Entorno de R con ventanas separadas. En la ventana de la izquierda se muestra la consola y en la ventana de la derecha el area donde se ecribe el script



Por qué usar R

R es un entorno de software libre para computación estadística y gráficos. Compila y se ejecuta en una amplia variedad de plataformas UNIX, Windows y MacOS lo que permite que el desarrollo se de más rápido que en los softwares de uso comercial debido a que los usuarios hacen desarrollos, los documentan y los suben al CRAN de R de manera cotidiana (Galán et al., 2016). Estos aportes de la comunidad y de los mismo desarrolladores se les conoce como paquetes y amplían las capacidades base de R.

Los paquetes más importantes para el tratamiento de datos se han recopilado en lo que se conoce el tidyverse. Los paquetes tidyverse comparten la misma filosofía en el formato de datos y programación, y están diseñados para trabajar de forma conjunta cubriendo todas las tareas en el análisis de un proyecto típico en ciencia de datos (Sánchez, sf).

Por los anterior, existen algunas razones de peso para decidirse por el uso de R en el análisis de datos:

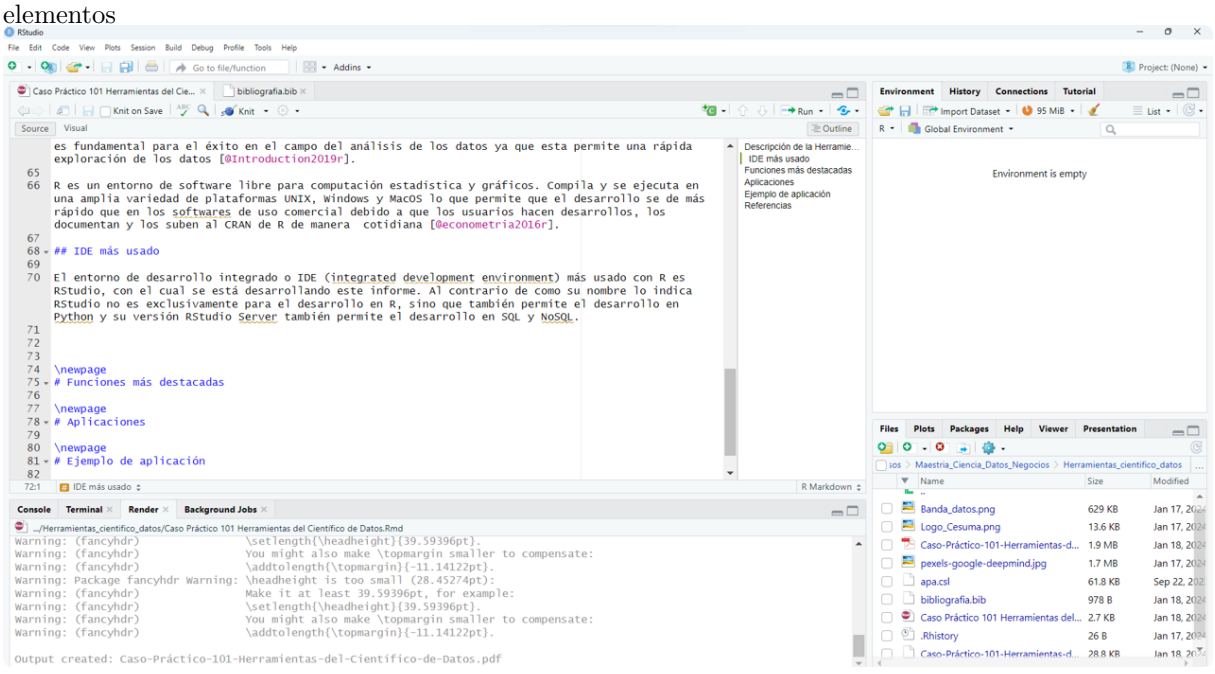
- R es ampliamente utilizado en la comunidad estadística y científica para realizar análisis estadísticos complejos, por lo que en Internet se puede encontrar respuestas a casi todos los inconvenientes que se presenten a la hora de realizar un análisis de datos.
- R es ampliamente utilizado en entornos académicos y de investigación, lo que hace que sea más fácil colaborar y compartir código y resultados con colegas.
- R tiene una amplia gama de funciones y paquetes específicos para estadísticas descriptivas, inferencia estadística, regresión, análisis multivariado y más.
- R proporciona herramientas poderosas para crear visualizaciones de datos de alta calidad. La librería `ggplot2` es especialmente popular por su flexibilidad y capacidad para producir gráficos estéticos y personalizables.

- R está diseñado para trabajar con conjuntos de datos de manera eficiente. Su sintaxis es especialmente adecuada para manipular y transformar datos, lo que facilita la limpieza y preparación de datos para análisis.
- R cuenta con una comunidad activa de usuarios y desarrolladores. Existen numerosos paquetes que amplían las funcionalidades de R para abordar diversas necesidades analíticas. Esta riqueza de paquetes permite a los usuarios acceder a una amplia gama de herramientas especializadas.
- R facilita la reproducibilidad en la investigación y análisis de datos. Los scripts y notebooks de R pueden documentar de manera efectiva los pasos realizados, lo que facilita compartir y reproducir análisis.
- R puede integrarse con otros lenguajes de programación, como C, C++, Java y Python, en especial cuando se utiliza el IDE RStudio. Esto permite aprovechar bibliotecas específicas de otros lenguajes cuando sea necesario.
- R es un software de código abierto, lo que significa que es gratuito y está disponible para su modificación y distribución. Esto ha contribuido a su popularidad y a la creación de una comunidad activa de usuarios y desarrolladores.

RStudio, el IDE más usado

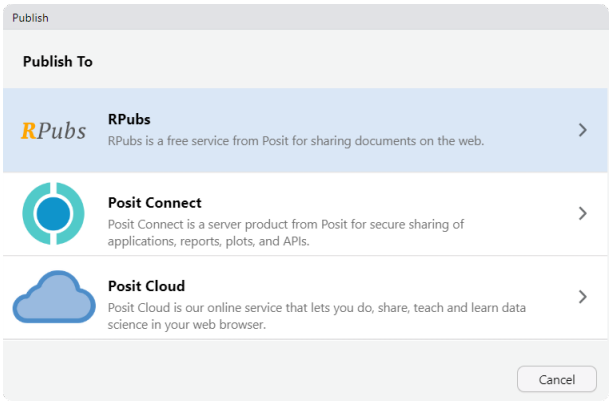
El entorno de desarrollo integrado o IDE (integrated development environment) más usado con R es RStudio, con el cual se está desarrollando este informe. Al contrario de como su nombre lo indica RStudio no es exclusivamente para el desarrollo en R, sino que también permite el desarrollo en Python y su versión RStudio Server también permite el desarrollo en SQL y NoSQL.

Figura 2: Vista general del IDE de RStudio. RStudio se divide en la zona de los scrits, el terminal y un panel lateral donde se muestra las variables creadas, el directorio de trabajo, las gráficas, entre otros elementos



Utilizando el procesador de texto RMarkdown que nos ofrece RStudio se abre la posibilidad de crear documentos en el lenguaje Markdown, el cual es un lenguaje de marcado sencillo que sirve para agregar formato, vínculos e imágenes con facilidad al texto simple, lo que permite crear paginas web de manera sencilla que luego se pueden alojar en RPubs, Posit Connect o Posit Cloud, siendo la primera la más popular.

Figura 3: plataforma de publicación abierta para documentos HTML producidos con RMarkdown desde RStudio



RMarkdown también nos ofrece la posibilidad de exportar nuestros informes a un documento de Word y si instalamos una versión ligera de LaTex, también podremos exportar los informes en pdf permitiendo utilizar el lenguaje de marcado de LaTex en RStudio como complemento al lenguaje de marcado Markdown (Güngör, 2019). Cabe recalcar que no se debe mezclar LaTex con Markdown, por lo que se debe avitar ciertas formalidades si se desea aprovechar el etiquetado de RMardown y mostrar tablas (data.frames) y gráficas hechas con código R.

En términos sencillos RMarkdown es un procesador de texto que ofrece además la posibilidad de incluir trozos de código desde R u otros formatos como Python o SQL (Boccardo & Ruiz, 2019).

Funciones más destacadas

R es un lenguaje de programación y entorno de software ampliamente utilizado en estadísticas y análisis de datos. Algunas de las funciones más destacadas de R incluyen:

Programación Funcional:

R es un lenguaje de programación funcional, lo que significa que admite programación funcional y permite el uso de funciones como objetos de primera clase. Esto implica que los usuarios de R pueden crear funciones que permiten reciclar código recurrente.

Por ejemplo, se puede calcular el área y el perímetro de un rectángulo multiplicando la base por la altura y multiplicando la suma de la base con la altura por 2.

```
b <- 5; h <- 6
cat("El área del rectángulo es", b*h, " y su perímetro es", (b+h)*2)
```

```
## El área del rectángulo es 30  y su perímetro es 22
```

Podemos crear una función que nos dé este resultado para cualquier rectángulo.

```
aypRectangulo <- function(b, h) cat(
  "El área del rectángulo es", b*h, " y su perímetro es", (b+h)*2, "\n")
```

Con esta función, con una sola linea se puede obtener el mismo resultado que el inicial.

```
aypRectangulo(5,6)
```

```
## El área del rectángulo es 30  y su perímetro es 22
```

Ahora probemos esta función con tres rectángulos de bases 6, 8 y 2 y alturas 3, 4, 2 respectivamente

```
aypRectangulo(6,3)
```

```
## El área del rectángulo es 18  y su perímetro es 18
```

```
aypRectangulo(8,4)
```

```
## El área del rectángulo es 32  y su perímetro es 24
```

```
aypRectangulo(2,2)
```

```
## El área del rectángulo es 4  y su perímetro es 8
```

Si se quiere todo en una sola linea, se puede usar la función `mapply` ya que se va a tomar, no una lista `sapply` sino una matriz (dos listas o vectores)

```
mapply(aypRectangulo, c(6, 8, 2), c(3, 4, 2))[0]
```

```
## El área del rectángulo es 18  y su perímetro es 18
## El área del rectángulo es 32  y su perímetro es 24
## El área del rectángulo es 4  y su perímetro es 8

## list()
```

Al final nos indica que el resultado de la función `mapply` es una lista (`list()`).

Programación Orientada a Datos:

R se diseñó pensando en la manipulación y análisis de datos. Su sintaxis es especialmente eficiente para trabajar con marcos de datos y realizar operaciones en columnas y filas. Esta característica se puede aprovechar para reciclar un conjunto de funciones que siempre se van a aplicar a los objetos que pertenecen a la misma clase. Un ejemplo de esto es la clase `data.frame`.

- SIMULANDO LA FUNCIÓN `_init_`:

Se define el método iniciador de la clase "`rectangulo`"

```
rectangulo <- function(altura = 0, base = 0){
    obj <- list(
    altura = altura,
    base     = base
    )
    class(obj) <- "rectangulo"
    return(obj)
}
```

- METODOS

Se utiliza una función que pedirá usar el método. Se suele utilizar el mismo nombre para la función que llama al método y el método.

```
Nombre <- function(...) UseMethod("Nombre")

Nombre.NombreClase <- function(...){
    ...
}
```

Por ejemplo, creamos el método que nos permita imprimir en forma de tabla los datos de los objetos de nuestra clase.

```
printt <- function(obj) UseMethod("printt")

printt.rectangulo <- function(df){
    n <- paste(names(df))
    m = ""
    for(i in 1:length(df)){
        texto   <- rep(" ", nchar(names(df)[i])-nchar(df[[i]])-1)
        texto   <- paste(texto ,collapse = "")
        m   <- paste(m, texto, df[[i]], "  ", sep = "")
    }
    cat(n, "\n", m, "\n")
}
```

A continuación, creamos el método que permite calcular el área del rectángulo y guardarlo en la entrada `Area` del objeto.

```
area <- function(obj) UseMethod("area")

area.rectangulo <- function(obj){
    obj$area = obj$altura*obj$base
    cat("El área del rectángulo es", obj$area, "\n")
    return(obj)
}
```

- Instanciar un objeto en la clase

Ahora miremos un ejemplo. Se define el método iniciador de la clase "`rectangulo`".

```
rect1 <- rectangulo(5,6)
printt(rect1)
```

```
## altura base
##      5    6
```

Se utiliza el método `area` que nos devuelve el área del rectángulo.

```
rect1 <- area(rect1)
```

```
## El área del rectángulo es 30
```

Ahora el objeto tiene la propiedad `area` a la cual se puede acceder por medio de `obj$area`. En este caso, el objeto es `rect1` (`rect1$area`).

```
rect1$area
```

```
## [1] 30
```

Veamos que elementos o propiedades tiene nuestro objeto:

```
printt(rect1)
```

```
## altura base area
##      5    6   30
```

Se deja el ejercicio de crear el método `perimetro` (perímetro).

Comunidad Activa y Paquetes:

La comunidad de usuarios de R es activa y contribuye con numerosos paquetes que amplían las funcionalidades del lenguaje. Existen paquetes para una amplia variedad de tareas, desde análisis de imágenes hasta procesamiento de texto.

Desarrollo de Paquetes y Extensiones:

R facilita el desarrollo de paquetes, lo que ha llevado a una rica colección de extensiones que abordan diversas necesidades analíticas y científicas.

Reproducibilidad y Documentación:

La capacidad de generar informes reproducibles es una característica importante de R. `R Markdown` permite integrar código, resultados y narrativa en un solo documento, facilitando la generación de informes y documentos interactivos brindando la posibilidad de exportar los informes a formato `Html` y publicarlos en un repositorio, `Word` o `pdf` con una versión ligera de `LaTex`.

Análisis Estadístico:*

R es especialmente fuerte en análisis estadísticos. Ofrece una amplia gama de funciones y paquetes para estadísticas descriptivas, inferencia estadística, pruebas de hipótesis, análisis de regresión y más.

Manipulación de Datos:

R facilita la manipulación de datos con funciones como `subset()`, `merge()`, `aggregate()`, y otras. Además, la manipulación de datos se simplifica mediante paquetes como `dplyr` y `tidyr`.

Modelado Predictivo y Machine Learning:

R cuenta con una amplia variedad de paquetes para modelado predictivo y machine learning, como `caret`, `randomForest`, `glmnet`, `xgboost`, entre otros.

Visualización de Datos:

Si bien, R base tiene la función `plot()` que acompañada con algunas funciones de bajo nivel, el usuario puede crear múltiples estilos de gráficas, estos estilos se encuentran guardados en la librería `ggplot2`, lo que La implica que su uso puede simplificar el desarrollo de visualizaciones ahorrando tiempo. La librería `ggplot2` en R es muy popular para la creación de gráficos y visualización de datos. Proporciona una sintaxis clara y flexible para producir gráficos de alta calidad.

Interfaz con Bases de Datos y Otros Lenguajes:

R puede conectarse a bases de datos y trabajar con datos almacenados en diferentes formatos. También se puede integrar con otros lenguajes de programación como `C`, `C++`, `Java` y `Python`.

Estas son solo algunas de las funciones destacadas de R. Su versatilidad y la comunidad activa de usuarios y desarrolladores continúan impulsando su popularidad en el ámbito del análisis de datos y la investigación estadística.

Aplicaciones

R es ampliamente utilizado en diversas aplicaciones, especialmente en áreas relacionadas con estadísticas, análisis de datos, y ciencia de datos. Algunas de las principales aplicaciones de R incluyen:

- **Análisis Estadístico:** R es extremadamente poderoso para realizar análisis estadísticos complejos. Se utiliza para llevar a cabo pruebas de hipótesis, análisis de varianza, regresiones y otros métodos estadísticos.
- **Ciencia de Datos:** R es una herramienta fundamental en la ciencia de datos. Se utiliza para limpiar, explorar y analizar conjuntos de datos, así como para construir modelos predictivos y realizar minería de datos.
- **Visualización de Datos:** La librería `ggplot2` de R es conocida por su capacidad para crear visualizaciones de datos de alta calidad. R se utiliza para generar gráficos, diagramas y tablas que ayudan a visualizar patrones y tendencias en los datos.
- **Bio informática y Genómica:** En la investigación en biología y genómica, R es utilizado para analizar datos de secuenciación genética, realizar estudios de expresión génica, y visualizar resultados en forma de gráficos y diagramas.
- **Economía y Finanzas:** R es utilizado en el análisis financiero y económico para modelar series temporales, calcular métricas financieras, y realizar pronósticos económicos.
- **Investigación Académica:** R es una herramienta comúnmente utilizada en la investigación académica en una variedad de disciplinas, desde las ciencias sociales hasta la biología y la física. Facilita el análisis y la visualización de datos experimentales.
- **Educación en Estadísticas:** R es utilizado en entornos académicos para enseñar estadísticas y análisis de datos. Su accesibilidad y naturaleza de código abierto hacen que sea una opción popular en cursos de estadística.
- **Desarrollo de Modelos Predictivos:** R es ampliamente utilizado en la construcción de modelos predictivos en áreas como la predicción de ventas, la evaluación del riesgo crediticio y la predicción de enfermedades.
- **Epidemiología:** R se utiliza en `epidemiología` para analizar datos relacionados con la propagación de enfermedades, modelar tasas de incidencia y realizar análisis de riesgos.
- **Automatización de Informes:** La integración de R con R `Markdown` permite la creación de informes reproducibles que combinan código, análisis y narrativa en un solo documento, facilitando la comunicación de resultados.
- **Análisis de Redes Sociales:** En sociología y otras disciplinas, R se utiliza para el análisis de redes sociales, permitiendo visualizar y analizar patrones de interacción en redes complejas.

Estas son solo algunas de las muchas aplicaciones de R. Su flexibilidad y la amplia variedad de paquetes disponibles hacen que sea una herramienta valiosa en diferentes campos de la investigación y la toma de decisiones basada en datos.

Ejemplo de aplicación

Como ejemplo de aplicación se mostrara el proceso de minería de datos a una serie temporal. El primer paso es ingresar los datos del archivo `Rmissing.csv`.

Importar datos

Para ingresas o leer los datos de un archivo `csv` se debe tener en cuenta si los valores están separados por `,` o por `;`. En el primer caso se utiliza la función `read.csv()` y en el segundo `read.csv2()`

```
datos_t <- read.csv("Rmissing.csv")
```

Tablas

Para la visualización de las tablas, al exportar al pdf, se utiliza la librería `kableExtra`.

```
library(kableExtra)
```

En la tabla 1 se muestra los 10 primeros datos. Con esta visualización se puede dar idea de la estructura de los datos y planificar la limpieza de los mismo.

Nota: Se muestra el código para fines didácticos, aunque este se puede obtener en el repositorio: https://github.com/FernandoAleisy/Maestria_Ciencia_Datos_Negocios.git

```
kbl(head(datos_t, 10),
    caption = "Los 10 primeros registros de la serie de tiempo (columna mydata).
    El registro 3 es un dato faltante.",
    label = 'datos',
    booktabs = T,
    longtable = T)
```

Tabla 1: Los 10 primeros registros de la serie de tiempo (columna mydata). El registro 3 es un dato faltante.

X	mydata
1	32.80146
2	42.46548
3	NA
4	32.20406
5	55.55765
6	33.05086
7	43.40162
8	37.76832
9	22.84418
10	36.42888

La función `summary` nos indica la cantidad de datos faltantes de cada variable de un `data.frame`. A continuación, se muestra la tabla 2 con los resultados de la función con el código `kbl(summary(dato_t)...)`

```
kbl(summary(datos_t),
    caption = "Resumen estadístico de las varibales del data.frame datos\\_t en
    el que se incluye la información de los datos faltantes",
    label = 'summary',
    booktabs = T,
    longtable = T)
```

Tabla 2: Resumen estadístico de las varibales del data.frame datos_t en el que se incluye la información de los datos faltantes

X	mydata
Min. : 1.00	Min. : 2.683
1st Qu.: 63.25	1st Qu.: 28.078
Median :125.50	Median : 34.573
Mean :125.50	Mean : 50.710
3rd Qu.:187.75	3rd Qu.: 42.465
Max. :250.00	Max. :999.000
NA	NA's :5

Gráficas

Lo que se observa de este resumen estadístico es que la variable `mydata` tiene 5 datos faltantes y al menos un dato atípico o `outliers`. La mejor forma de visualizar los `outliers` es por medio de una gráfica.

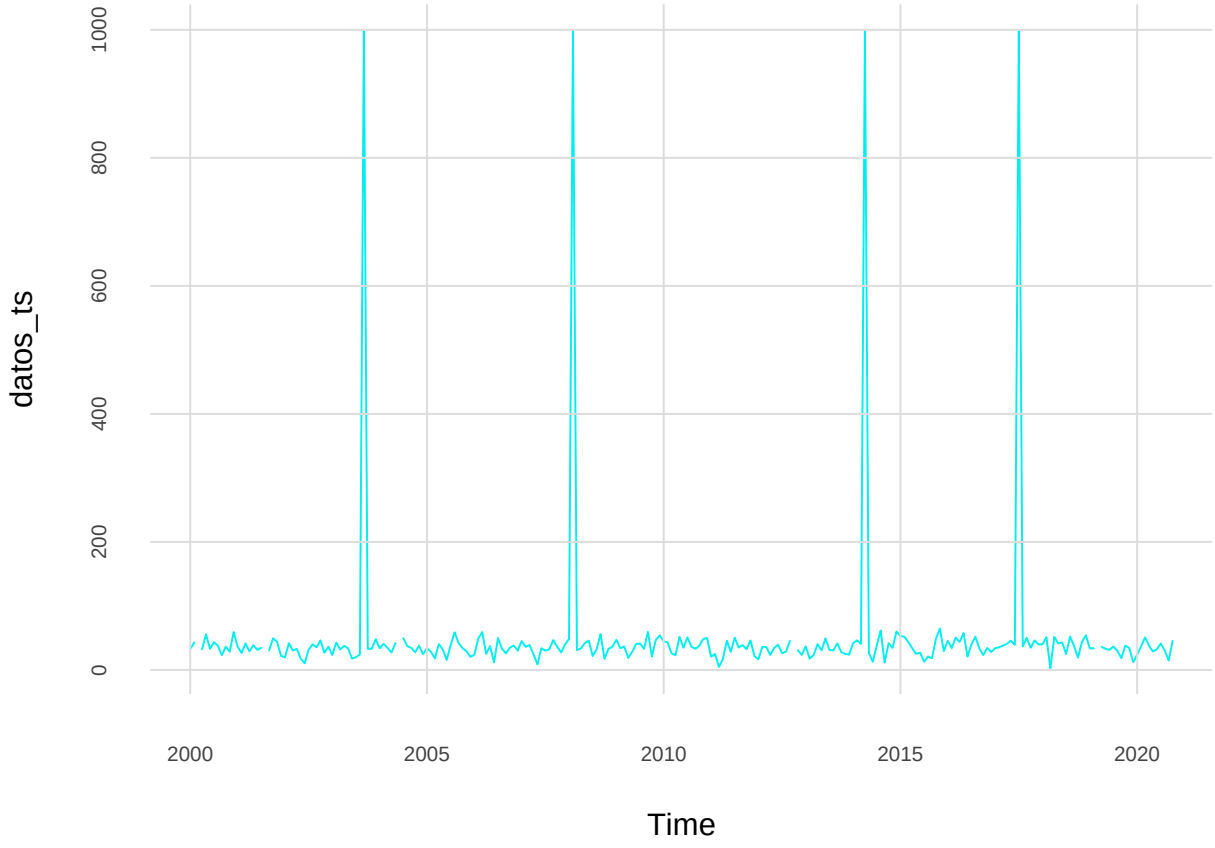
R base

Para gragicar una serie de tiempo se debe convertir a la variable principal en una variable del tipo temporal. Suponiendo que los datos son mensuales y el primer registro corresponde a enero del año 2000, su crea la variable `datos_ts` como una del tipo `ts`.

```
datos_ts <- ts(datos_t$mydata, start = c(2000, 1), frequency = 12)
```

Con R base y un poco de trabajo se puede lograr una buena gráfica, como en el ejemplo de la gráfica 1.

```
par(mar = c(4, 4, 0, 0))
plot(datos_ts, col = "cyan2", axes = FALSE)
for (i in c(1,2)) axis(i, col = "white", cex.axis = 0.7, col.axis = "gray26")
grid(col = "gainsboro", lwd = 1, lty = 1)
```



Grafica 1: Datos temporales con valores faltantes y datos atípicos. Se ha creado un tema minimalista con fuunciones base de R

En la gráfica 1 se evidencia 4 `outliers` (4 picos) y también se alcanza a ver los datos faltante en los lugares donde se evidencia los saltos de la gráfica.

La librería ggplot2

La librería ggplot2 es hecha por la comunidad y está basada en la función de R base `plot()` y las funciones de bajo nivel que permite agregar y/o modificar características de las gráficas. Para poder usar la serie temporal con la librería `ggplot2`, más precisamente con la función `ggplot()`, se debe utilizar un pequeño truco, el cual consiste en crear un `data.frame` como se muestra a continuación.

```
datos_t <- data.frame(fecha = time(datos_ts), valor = as.numeric(datos_ts))
```

Con la función `class()` combinada con la función `sapply()` se puede ver el tipo de variables que posee el `data.frame`. Observamos que la variable que se está analizando es del tipo `numeric` el cual se requiere para poder realizar la gráfica con la función `ggplot()`.

```
sapply(datos_t, class)
```

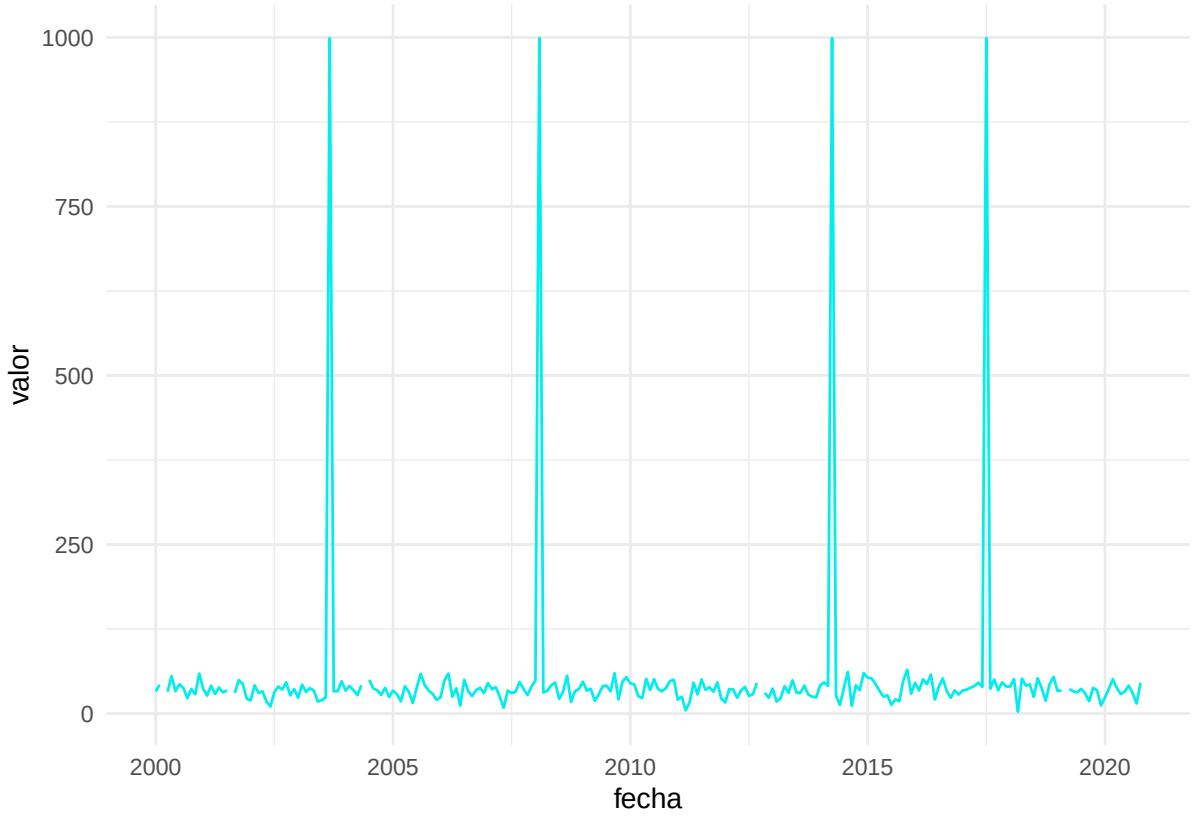
```
##     fecha     valor
##      "ts" "numeric"
```

La ventaja de la librería o paquete `ggplot2` es que tiene temas que permite crear gráficas más estilizadas con solo hacer el llamado de la función del tema deseado. A continuación, se carga la librería.

```
library(ggplot2)
```

Con la librería `ggplot2` se muestra la gráfica 2

```
ggplot(datos_t, aes(x = fecha, y = valor))+
  geom_line(colour = "cyan2", lwd = 0.5) +
  theme_minimal()
```



Grafica 2: Datos temporales con valores faltantes y datos atípicos. Se ha aplicado el tema minimalista de la librería **ggplot2**

Con la gráfica 2 se obtiene un tema parecido al creado con R base. Uno puede crear sus propios temas y crear una función que pueda ser utilizada cada vez que sea necesario.

Nuevamente, con esta gráfica, se observan 4 `outliers` y identifica que la variable tiene datos faltantes.

Limpiando la variable

Para cambiar los `outleirs` e imputar los valores faltante, en una serie temporal, se suele utilizar la interpolación. El paquete `forscat` permite realizar estas dos actividades por separados o a la vez con la función `tsclean()`.

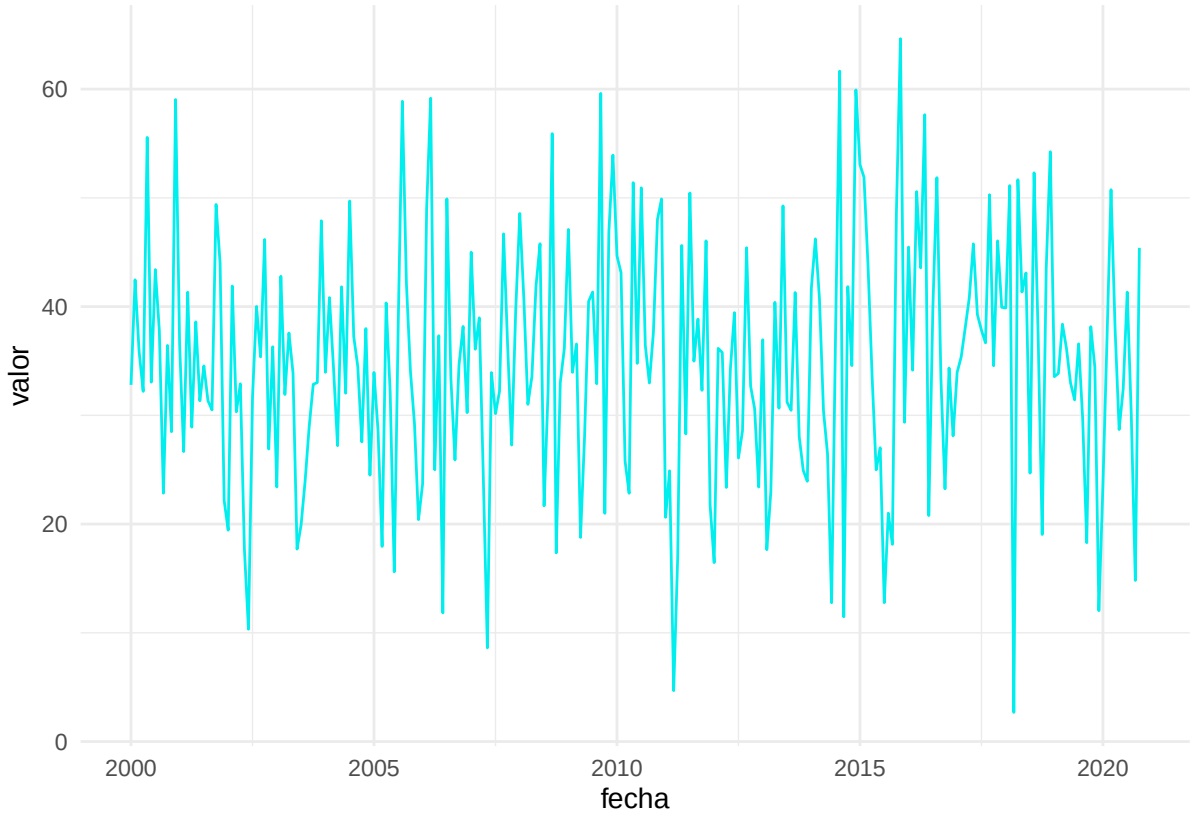
```
library(forecast)
```

Ahora ya podemos limpiar los datos:

```
datos_t$valor <- as.numeric(tsclean(datos_ts))
```

Al realizar la gráfica se observa que han desperecido los `outliers` y los datos faltantes, gráfica 3.

```
ggplot(datos_t, aes(x = fecha, y = valor))+
  geom_line(colour = "cyan2", lwd = 0.5) +
  theme_minimal()
```



Grafica 3: Datos temporales luego de realizar la mineria de datos o la limpieza de datos

O directamente con la función `summary()` se obtiene el resumen estadístico en el que ya no aparecen los datos faltantes y el valor máximo de la variable ya es razonable. Algo extra, a la hora de hacer un informe es que el código no aparezca, por lo que se puede usar el parámetro `echo=FALSE` en el `Chunk` (“‘{r}‘) quedando “‘{$recho = FALSE$}‘. Como se puede ver, el código de la tabla 3 ya no se ve cuando se usa ese parámetro en `Chunk`.

Tabla 3: Resumen estadístico de las varibales del data.frame datos_t después de la limpiza de los datos

fecha	valor
Min. :2000	Min. : 2.683
1st Qu.:2005	1st Qu.:28.157
Median :2010	Median :34.510
Mean :2010	Mean :34.945
3rd Qu.:2016	3rd Qu.:41.784
Max. :2021	Max. :64.633

Referencias

Boccardo, G., & Ruiz, F. (2019). *RStudio para estadística descriptiva en ciencias sociales*. 23. https://bookdown.org/gboccardo/manual-ED-UCH/

Correa, J., & González, N. (2002). *Gráficos estadísticos con r*.

Galán, J., Feregrino, J., Ruíz, L., Quintana, L., Mendoza, M., & Rosales, R. (2016). *Econometría aplicada usando r*. 23.

Güngör, M. (2019). *Rmarkdown for beginners - part 2: Using the papaja template*. https://osf.io/w39vn/

Irizarry, R. A. (2019). *Introduction to data science, data analysis and prediction algorithms with r*. https://rafalab.dfci.harvard.edu/dsbook/

López, J. (2006). *Guía casi completa de BIBTEX*. https://ctan.math.illinois.edu/info/spanish/guia-bibtex/guia-bibtex.pdf

Sánchez, R. (sf). *Ciencias de datos con r*. 10.

Santana, A., & Hernández, C. (2016). *Gráficos en r: La función plot*. https://estadistica-dma.ulpgc.es/cursoR4ULPGC/9d-grafPlot.html

The R Foundation. (2002). *Statutes of "the r foundation for statistical computing"*. https://www.r-project.org/foundation/

Zhu, H. (2024). *Create awesome LaTeX table with knitr::kable and kableExtr*. https://haozhu233.github.io/kableExtra/awesome_table_in_pdf.pdf