Hands-on! Introducción al análisis de datos con R

Beatriz Fernández Blanco
Maria Guaita Céspedes

1. ¿Por qué R?

1.1 Un poco de historia...

Conocer cómo nació R es interesante para comprender sus características. R es un lenguaje de programación creado por Ross Ihaka y Robert Gentleman en los años 90, que eran estadísticos en la Universidad de Auckland (Nueva Zelanda) y querían crear un material mejor para dar el curso de introducción a la estadística a sus alumnos. Por eso crearon R, basado en el lenguaje S. R siempre ha sido un proyecto de uso libre desde junio de 1995. El hecho de que fuera creado por estadísticos y no por ingenieros para el desarrollo de software como pasa con otros lenguajes de programación como C y Java, explica que uo de sus puntos fuertes sea la **interactividad** y la **visualización** de los datos.

Figure 1: Logo de R y sus creadores

El mantenimiento y desarrollo de R es realizado por el **R Development Core Team**, un equipo de especialistas en ciencias computacionales y estadística provenientes de diferentes instituciones y lugares alrededor del mundo. Este equipo mantiene la versión ***base*** de R que,

como su nombre indica, es sobre la cual se crean otras implementaciones de R así como los paquetes que expanden su funcionalidad.

- https://bookdown.org/jboscomendoza/r-principiantes4/un-poco-de-historia.html
- http://rafalab.dfci.harvard.edu/dsbook/getting-started.html#fn1
- Ihaka, R., & Gentleman, R. (1996). R: a language for data analysis and graphics. Journal of computational and graphical statistics, 5(3), 299-314.

1.2 Ventajas de usar R

Vamos a hablar de R como herramienta de análisis de datos. Probablemente hayáis usado SPSS o Grahpad.

- Es de uso libre (forma parte del sistema GNU)
- Se puede usar en diferentes sistemas operativos: Windows, Linux, Mac
- Hay una gran comunidad de usuarios de R activa y en contínuo crecimiento y, por lo tanto, hay muchos recursos para aprender y hacer preguntas
- Ofrece muchas utilidades en el análisis de datos, en especial en cuanto a interactividad y visualización. En general, conocer un lenguaje de programación os facilitará el análisis de datos, sea cuál sea vuestro ámbito de especialización. Además, mejora otras habilidades transversales, como es el pensamiento lógico. Y... el perfil bioinformático/bioestadístico está muy valorado.

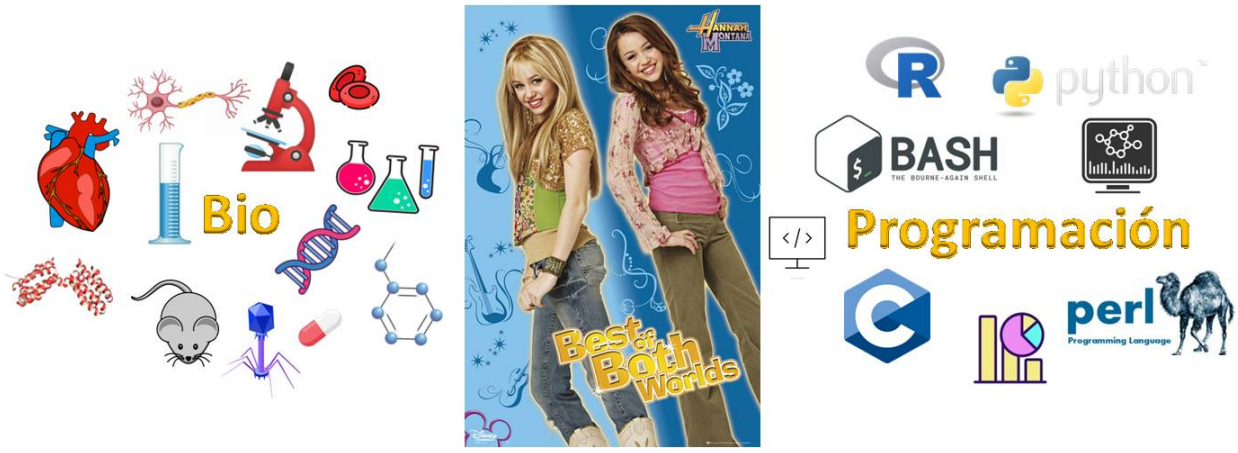


Figure 2: "Lo mejor de los dos mundos"

2. Instalación de R y RStudio

Aquí es necesario hablar de CRAN (Comprehensive R Archive Network), que es una red de servidores alrededor del mundo que almacena el código y documentación de R. Es decir,

CRAN es el sitio oficial a través del cual descargaremos R. En estos ordenadores ya tenemos instalado R, pero si queréis descargarlo en cas estos son los pasos que tenéis que seguir.

`cran.r-project.org`

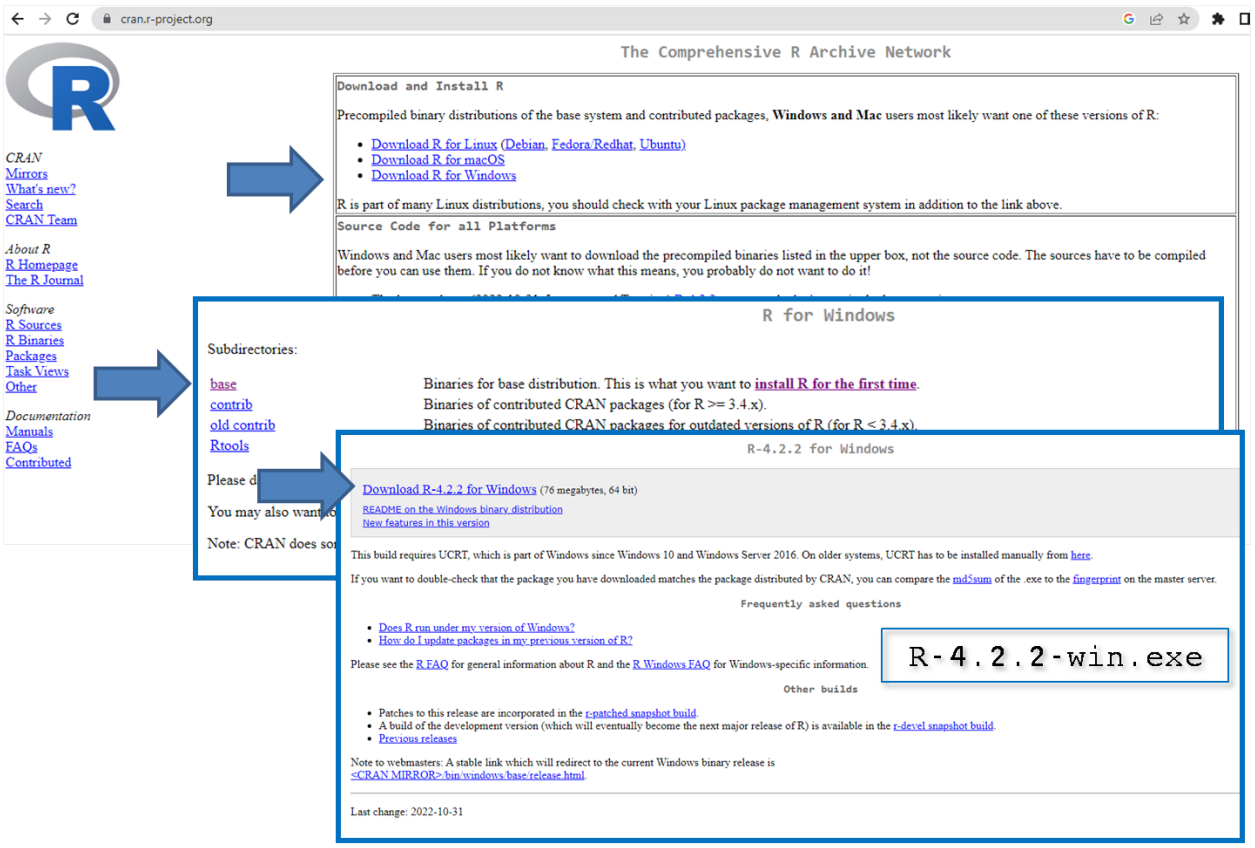


Figure 3: Instalación de R

Una vez instalado R podemos abrir la consola para ver qué apariencia tiene. Vemos además este mensaje de que R es un software libre. Aquí ya podríamos trabajar. Aprovechamos aquí para explicar algunos conceptos. La consola es el espacio donde R ejecuta las órdenes que le damos. Este símbolo de "mayor que" se llama prompt, y significa que R está listo para que le demos una orden.

Sin embargo, vamos a aprender a usar R dentro de RStudio, que es una interfaz que nos da muchísimas más funcionalidades que trabajar solo con la consola. La descarga de RStudio la haremos desde Posit.

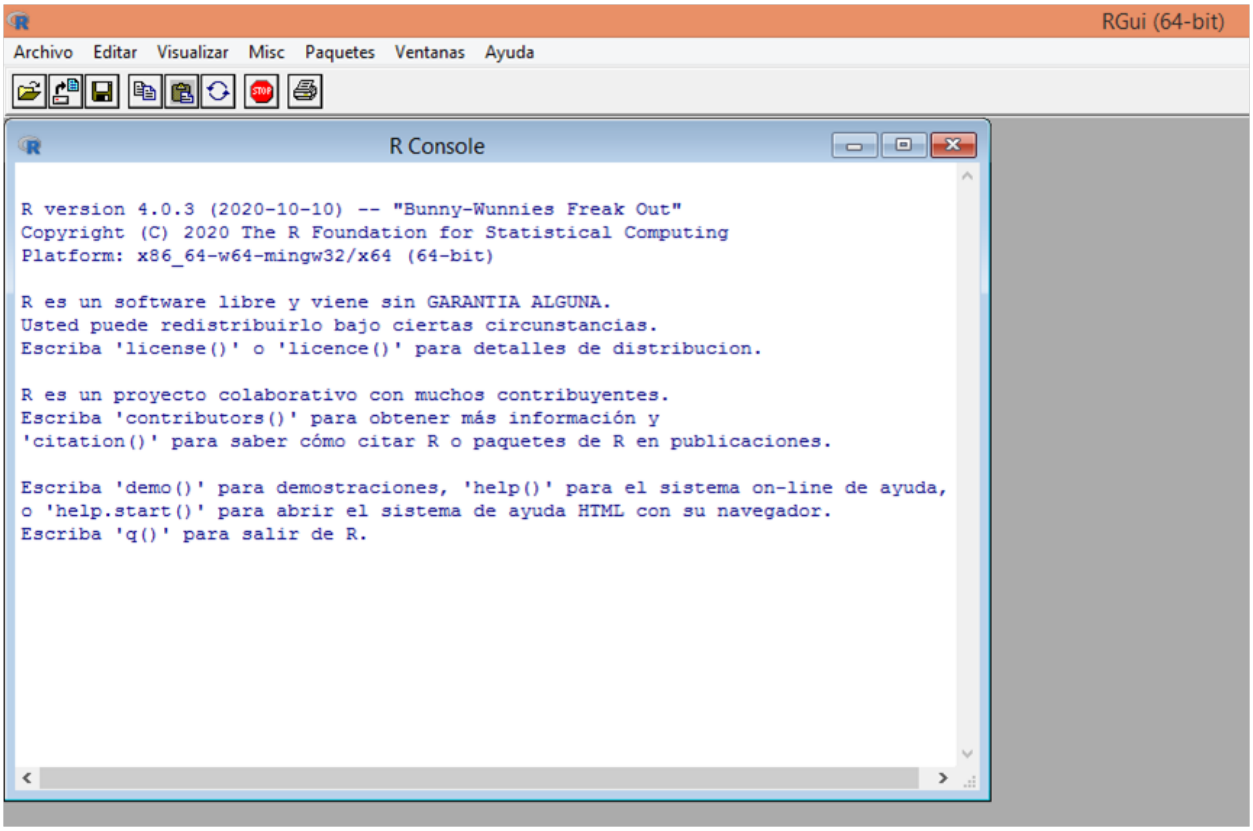


Figure 4: La consola de R

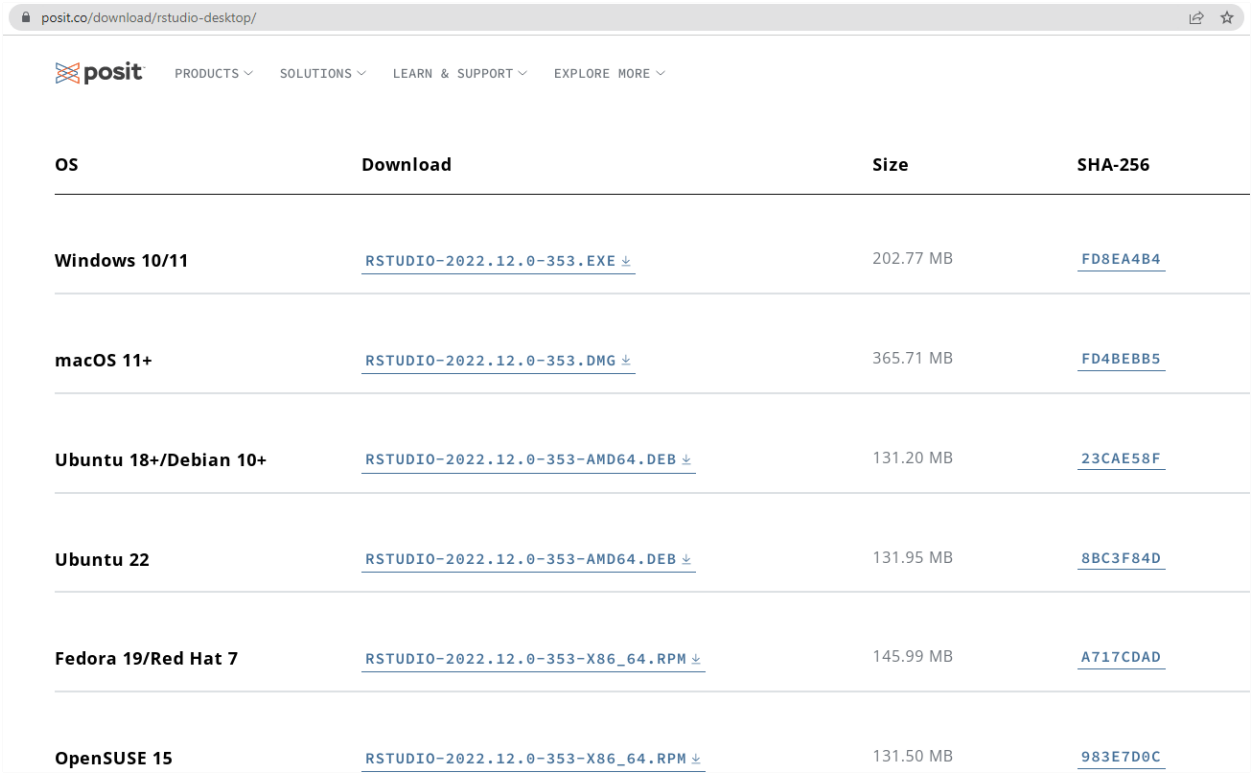


Figure 5: Instalación de RStudio

3. Exploramos RStudio ¡y escribimos las primeras líneas de código!

Cuando abrimos RStudio vemos varios paneles. A la izquierda tenemos la consola. Si creamos un documento nuevo de R vemos que se abre en el panel superior. A la derecha tenemos otros dos paneles con varias pestañas. Vamos a explorarlos un poco. Vamos a crear nuestro primer script. Un script es un bloque de código. Pinchamos en File > New File > R Script. Vemos que podemos crear otro tipo de documentos de R, pero no vamos a explicarlos.

En esta sesión vamos a trabajar con datos reales, pero antes necesitamos conocer cómo escribir en R. Vamos a ver algunas líneas de código sencillas para irnos familiarizando. En R podemos hacer operaciones matemáticas. Para ejecutar un comando pinchamos en "Run" o tecleamos CTRL+INTRO. En la consola es suficiente con INTRO.

Operadores matemáticos

```
3+2
```

```
## [1] 5
```

```
3-2
```

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

```
3*2
```

```
## [1] 6
```

```
3/2
```

```
## [1] 1.5
```

```
3**2
```

```
## [1] 9
```

Operadores lógicos

```r
1==2 # IMPORTANTE ESCRIBIR DOS VECES "="
```

```
## [1] FALSE
```

```r
1!=2
```

```
## [1] TRUE
```

```r
1<0
```

```
## [1] FALSE
```

```r
1>0
```

```
## [1] TRUE
```

Variables

Un concepto que tenemos que aprender es el de "variables". Las variables son sencillamente objetos en los que almacenamos información. La información que almacenamos puede ser de distinto tipo. Por ejemplo, podemos asignar el valor 2 a la variable a y el valor 3 a la variable b. Probad a ejecutar este código y veréis cómo quedan asignados los valores a las variables. Además fijáos en el panel superior derecho: las variables han pasado a formar parte del entorno, que es el conjunto de variables que hemos ido definiendo.

```r
a=2 # Los valores se pueden asignar con un =
b<-3 # O con <- (forma preferida por convención)
b < -3  # OJO! Los espacios cuentan
```

```
## [1] FALSE
```

```r
c <- "Hola"
d<-a+b
```

Además de las variables, véis que hay unas líneas que empiezan por #. Esto son comentarios y son líneas que R no ejecuta. Son muy útiles para irnos poniendo notas de lo que hace nuestro código. NOTA IMPORTANTE: podemos nombrar a las variables como queramos pero sabiendo que:

- **NUNCA** debemos nombrar una variable con un nombre de una variable que ya exista en R. Por ejemplo, estaría mal sum<-2+3, porque sum() ya existe en R y podemos crear conflictos. Ante la duda podemos ejecutar el comando help() para consultar si existe una variable, (help(sum)); si no nos devuelve nada entonces es que esa variable no está predefinida en R.
- Las variables **no contienen espacios**.
- Por convención se escriben en **minúscula** y donde querríamos poner un espacio pondremos un guión bajo. Estas convenciones pueden ser distintas en otros lenguajes de programación.
- Intentad que los nombres de vuestras variables sean **representativos** de su significado, tanto por facilitaros vuestro trabajo como por si lo lee otra persona. Imaginad que vuestro código tiene 200 líneas. Al final habréis creado muchas variables y necesitaréis saber qué es cada una de ellas.

```
# En el ejemplo anterior, imaginad que queremos calcular la media
# de las variables a y b
media_valores<-(a+b)/2
mean(a,b)
```

```
## [1] 2
```

Las variables que hemos definido hasta ahora solo contienen un valor, pero hay diferentes tipos de variables según el valor o valores que les asginemos. # Vectores

```
# Se definen con c() y separando los elementos con una coma
saludos<-c("Hola","Buenos días", "Buenas tardes")
length(saludos)
```

```
## [1] 3
```

```
# Acceder a los elementos de un vector
saludos[2]
```

```
## [1] "Buenos días"
```

```
# Reordenar un vector y sobreescribirlo
saludos<-saludos[c(3,1,2)]
```

Conjuntos de datos

```
meses_dias<-data.frame(meses=c("enero","febrero","marzo","abril","mayo","junio",
                               "julio","agosto","septiembre","octubre",
                               "noviembre","diciembre"),
                       dias=c(31,28,31,30,31,30,31,31,30,31,30,31))
```

Funciones

Algunas funciones ya están definidas. ¡¡La función help() la más útil de todas!!

```
log(10)
```

```
## [1] 2.302585
```

```
sum(a,b)
```

```
## [1] 5
```

```
help(sum)
```

```
## starting httpd help server ... done
```

Clases de objetos

Cualquier objeto que definamos pertenece a una clase. Podemos conocer la clase a la que pertenece un objeto con la función class():

```
c="4"
class(a)
```

```
## [1] "numeric"
```

```
class(b)
```

```
## [1] "numeric"
```

```r
class(c)
```

```
## [1] "character"
```

```r
class(sum)
```

```
## [1] "function"
```

Es importante saber con qué tipo de objetos estamos trabajando.

```r
c="4"
# a+c # Esto nos da error
```

En dos horas que dura este taller es imposible hacer un recopilatorio de todas las operaciones básicas en R, estos son solo algunos ejemplos. Tampoco hay nadie que se sepa todo lo que se puede hacer en R. Lo que sí nos parece importante es que conozcáis el gran libro de R, que no es otra cosa que Google. Lanzad una pregunta de algo que queréis hacer en R pero no sabéis cómo, por ejemplo yo quiero obtener los valores del 1 al 100 con un intervalo de 2 pero no sé cómo hacerlo. Vamos a buscar "how to create range of values with interval in r". Este es un ejemplo muy sencillo que casi cualquier libro introductorio de R va a cubrir. Pero hay ocasiones en que la cosa se complica más y veréis que las respuestas que da la gente son de lo más variadas. Normalmente no hay una única forma de hacer las cosas, por eso no hay código que esté bien o mal (siempre y cuando funcione).

4. Análisis de un conjunto de datos

Ahora que sabemos un poco de cómo funciona R vamos a seguir profundizando en los conjuntos de datos. Quizás este el tipo de objeto que más podéis usar en esta etapa. Para ello vamos a usar un conjunto de datos reales. Queremos que os imaginéis que es un estudio que estéis haciendo en vuestro TFG, TFM, etc. Los cursos de R suelen hacer estas explicaciones con conjuntos de datos que están en librerías de R que podéis instalar, pero preferimos daros algo que os podáis imaginar que es vuestro.

4.1 Descripción del conjunto de datos

Los datos están disponibles a través del portal **cBioPortal** como "TMB and immunotherapy (MSK,Nat Genet 2019)" (https://www.cbioportal.org/study/summary?id=tmb_mskcc_2018). En la pestaña "Clinical Data" seleccionamos todas las variables y descargamos los datos en formato ".tsv" (**Figura 1**).

Subimos los datos a Posit Cloud. Importamos el documento como "tmb".

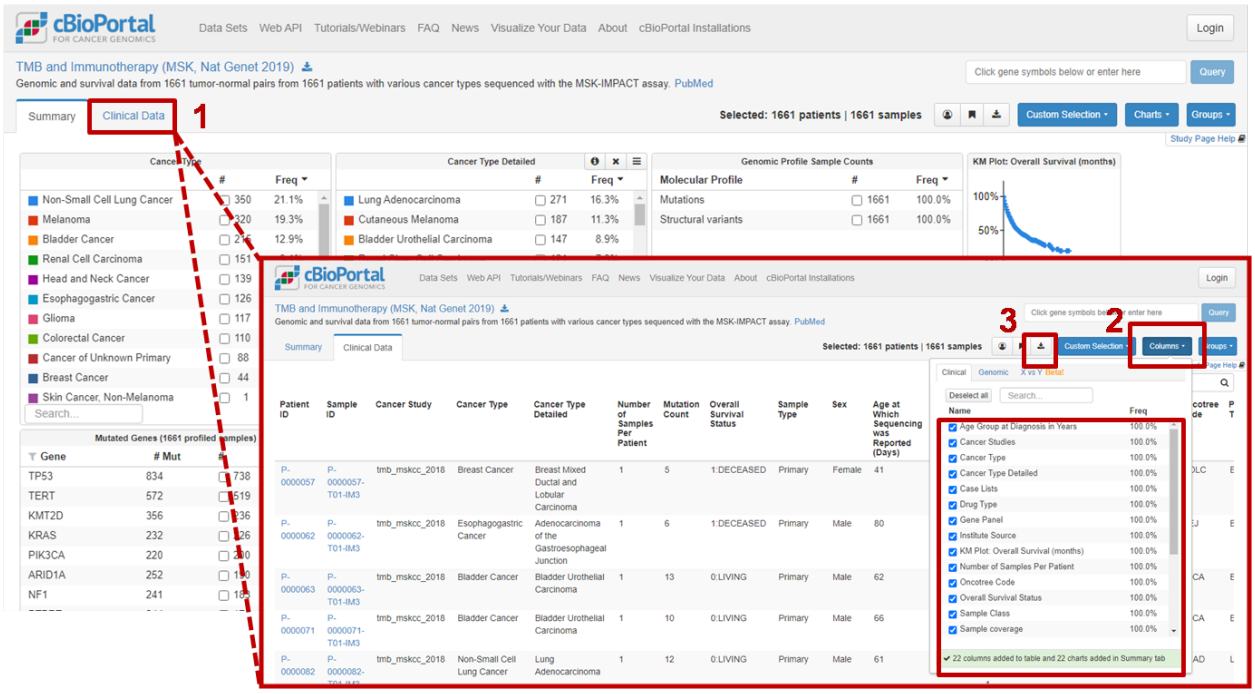


Figure 6: Descarga de los datos en cBioPortal

```
# Dos opciones igual de válidas:
tmb<-read.delim("tmb_mskcc_2018_clinical_data.tsv")
# Con read.table obligatorio poner el separador
tmb<-read.table("tmb_mskcc_2018_clinical_data.tsv",header=TRUE,sep="\t")
# ¿Y si fuera un excel? readxl: read_excel(),read_xls(),read_xlsx()
```

Este es un conjunto de datos genómicos y de supervivencia dentro del proyecto Mutation Profiling of Actionable Cancer Targets (MSK-IMPACT) del Memorial Sloan Kettering Cancer Center (MSKCC) (Samstein et al. 2019). En él se recogen datos pacientes de cáncer en estadios avanzados a los que se les ha realizado una secuenciación de exoma con el objetivo de relacionar la carga mutacional con la respuesta a inhibidores de puntos de control inmunitarios (ICI). Los ICI son anticuerpos contra determinadas dianas, como CTLA4 y PD-1/PDL.1 y supusieron una revolución en el tratamiento de pacientes en estadios avanzados. Sin embargo, la respuesta a largo plazo solo es efectiva en algunos pacientes, por lo que se necesitan biomarcadores para predecir qué pacientes se pueden beneficiar de ellos. En concreto, este estudio tenía por objetivo estudiar en una cohorte amplia de pacientes si la carga mutacional estaba relacionada con la respuesta a estos tratamientos, algo que se había observado en cohortes más pequeñas.

4.2. Análisis descriptivo

Vamos a comenzar explorando el conjunto de datos y también vamos a ir analizando algunas variables para ir incorporando nuevos conceptos.

```
# class(tmb) # vemos que R lo reconoce como data.frame
# str(tmb) # estructura del conjunto de datos
# tmb$Cancer.Type # acceder a las variables
# class(tmb$tmb$Cancer.Type)
```

PREGUNTA DIRIGIDA 1. Sabiendo cómo se accede a las variables y recordando que la media se puede calcular con la función mean(), ¿sabríais decirme cuál es la media de la variable Age.at.Which.Sequencing.was.Reported..Days.?

```
# Más detalles sobre el conjunto de datos
dim(tmb)
```

```
## [1] 1661   24
```

```
# colnames(tmb)
# rownames(tmb)
```

Filtrar una variable []

Supongamos que me interesa conocer cuál ha sido la media de supervivencia de los pacientes vivos y los exitus.

```
mean(tmb$Overall.Survival..Months.)
```

```
## [1] 14.07827
```

```
mean(tmb$Overall.Survival..Months.[tmb$Overall.Survival.Status=="1:DECEASED"])
```

```
## [1] 9.4375
```

```
mean(tmb$Overall.Survival..Months.[tmb$Overall.Survival.Status=="0:LIVING"])
```

```
## [1] 18.73583
```

PREGUNTA DIRIGIDA 2. Ahora que sabemos cómo filtrar una variable, ¿cuál es la media de la variable Overall.Survival..Months. en mujeres? # Función summary()

```r
summary(tmb$Overall.Survival..Months.)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    4.00   11.00   14.08   20.00   80.00
```

```r
summary(tmb$Overall.Survival..Months.[tmb$Overall.Survival.Status=="1:DECEASED"])
```

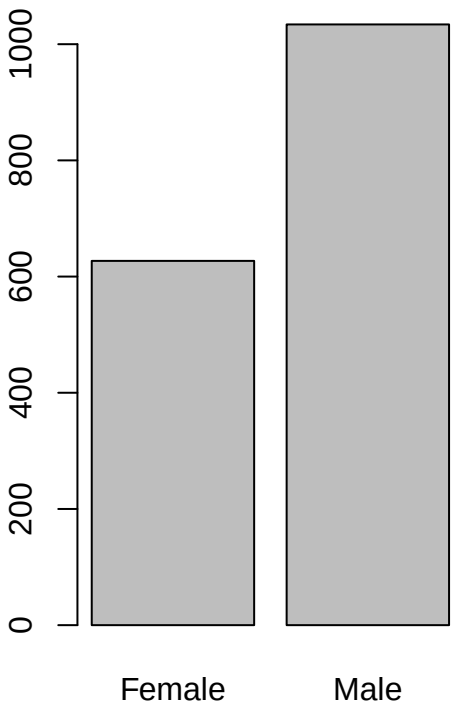
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   3.000   7.000   9.438  13.000  68.000
```

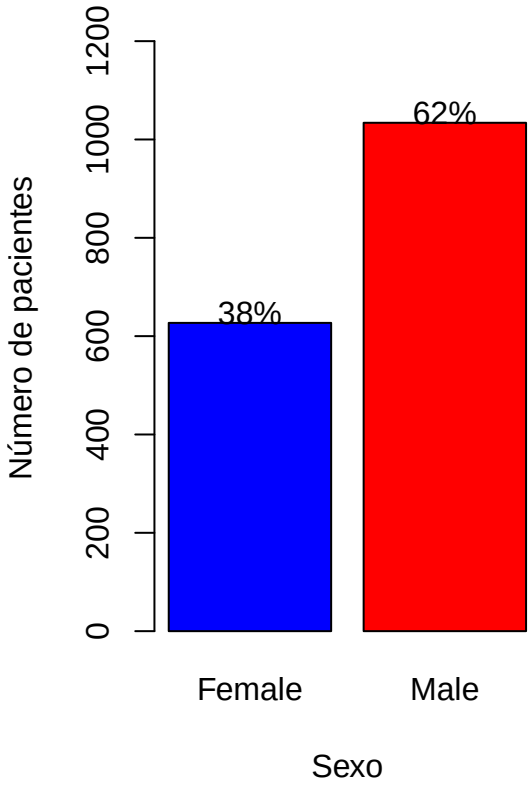
```r
summary(tmb$Overall.Survival..Months.[tmb$Overall.Survival.Status=="0:LIVING"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    8.00   16.00   18.74   25.00   80.00
```

Gráficos de frecuencia: funciones table() y barplot()

```r
par(mfrow=c(1,2))
# Sexo
sex.freq<-table(tmb$Sex)
barplot(sex.freq) # Lo más sencillo
sex.bar<-barplot(sex.freq,
        col=c("blue","red"),
        xlab="Sexo",
        ylab="Número de pacientes",
        ylim = c(0,max(sex.freq)+200),
        ) # Un poco más bonito
text(x=sex.bar, y=sex.freq+20,
     labels=paste0(round(sex.freq/sum(sex.freq)*100),"%") ,cex=1)
```






```r
# Edad
age.freq<-table(tmb$Age.Group.at.Diagnosis.in.Years)[c(1,3,4,5,2)]
age.bar<-barplot(age.freq,
        col=brewer.pal(5, "Set3"),
        xlab="Edad al diagnóstico (años)",
        ylab="Número de pacientes",
        ylim = c(0,max(age.freq)+30),
        cex.names=0.6
        )
text(age.bar, age.freq+20, paste0(round(age.freq/sum(age.freq)*100), "%") ,cex=1)
```

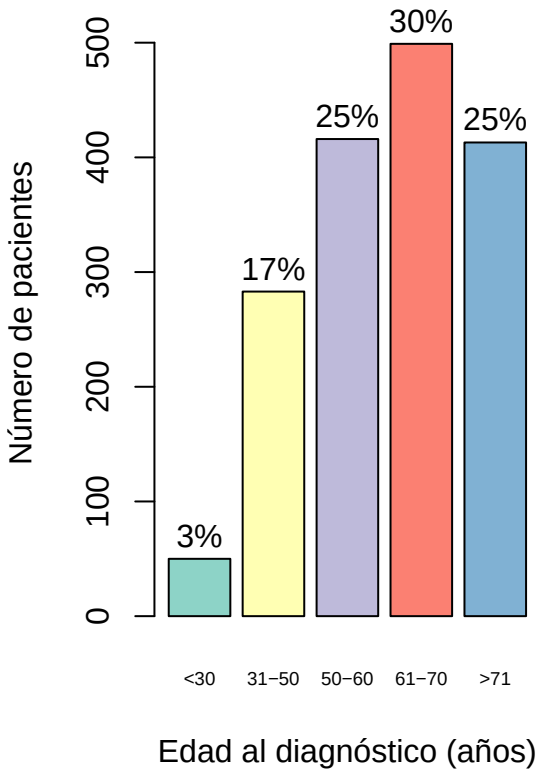


Gráfico de cajas y bigotes: función boxplot()

```
boxplot(tmb$TMB..nonsynonymous.~tmb$Overall.Survival.Status)
```

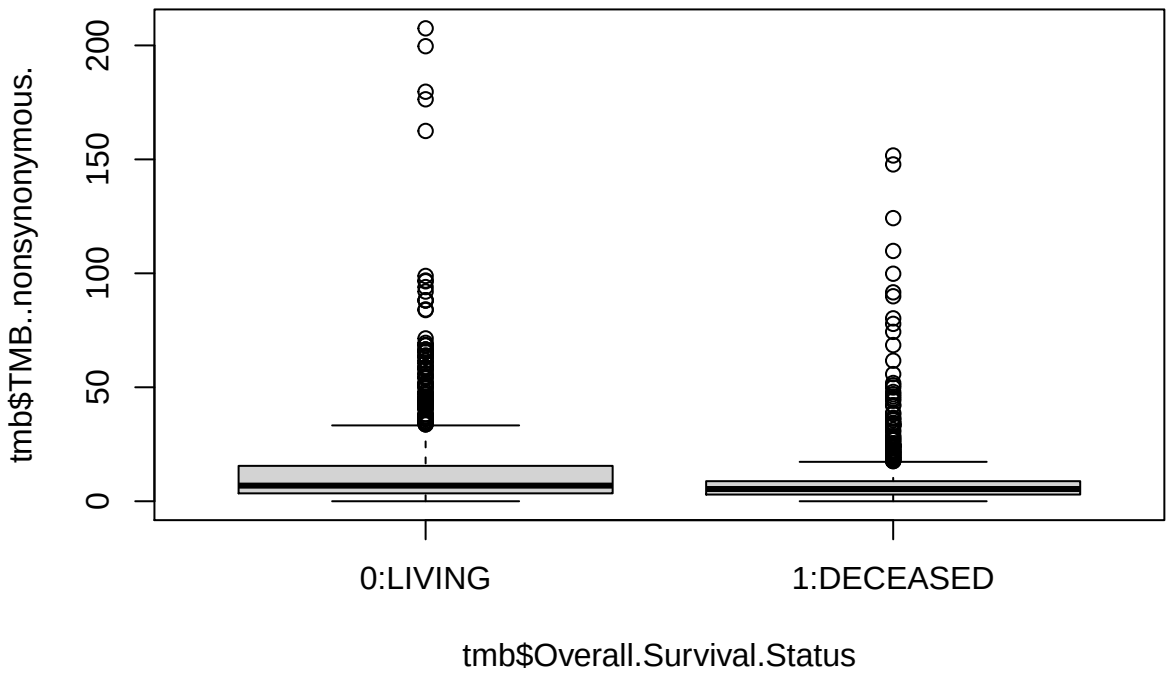
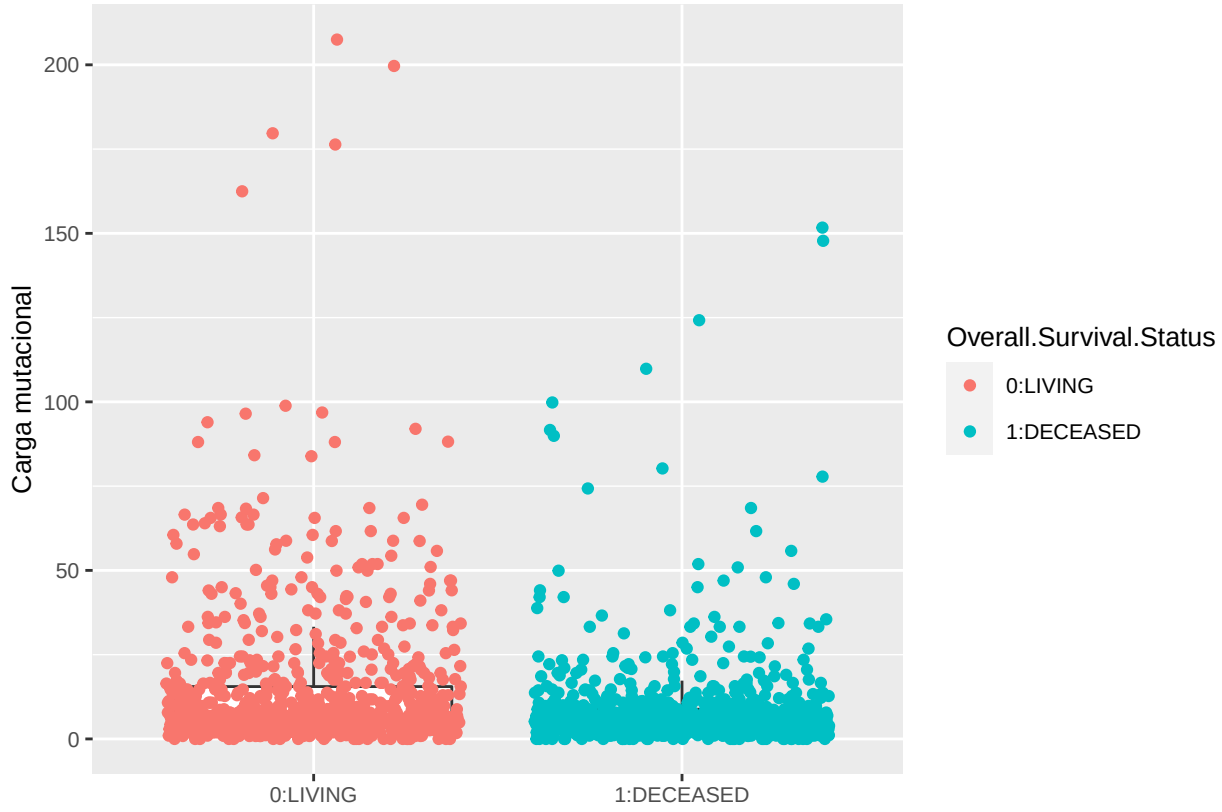


Gráfico de cajas y bigotes: solución con ggplot2

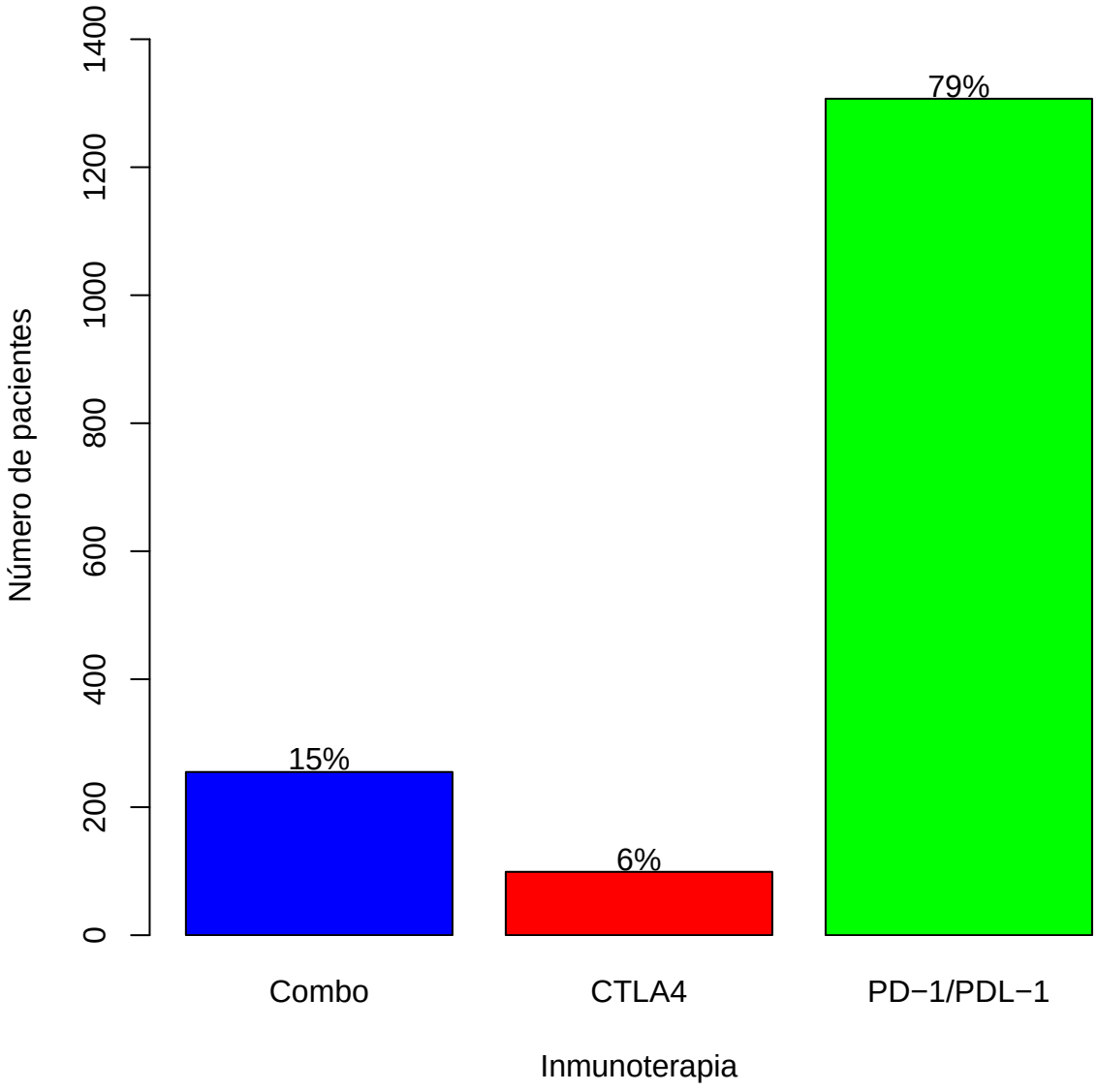
Este apartado nos sirve para introducir el paquete ggplot2, que nos ofrece muchas posibilidades para crear gráficos.

```
# Instalamos ggplot2
# install.packages("ggplot2")
ggplot(data=tmb,
       aes(x=Overall.Survival.Status,y=TMB..nonsynonymous.),
       fill=factor(Overall.Survival.Status))+
  geom_boxplot(outlier.shape = NA)+
  geom_jitter(aes(colour=Overall.Survival.Status))+
  theme(text=element_text(size=10))+
  labs(y="Carga mutacional",x=" ")
```



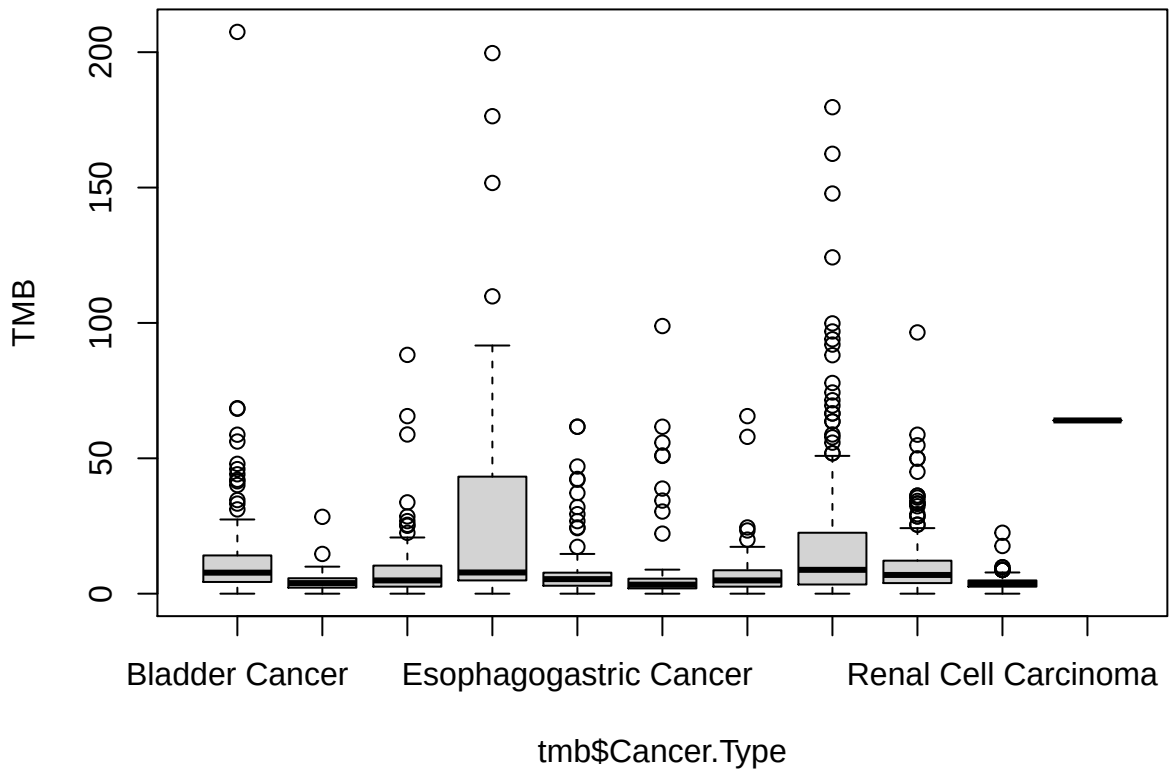
PREGUNTA DIRIGIDA 3. Os dejamos un rato para que hagáis un gráfico de barras para la variable Drug.Type. Queremos enseñar de forma gráfica cuál fue la inmunoterapia de elección más frecuente en esta cohorte.

```
# Inmunoterapia
drug.freq<-table(tmb$Drug.Type)
drug.bar<-barplot(drug.freq,
        col=c("blue","red","green"),
        ylim=c(0,1500),
        xlab="Inmunoterapia",
        ylab="Número de pacientes",
        xpd=FALSE
        )
text(drug.bar, drug.freq+20, paste0(round(drug.freq/sum(drug.freq)*100),"%") ,cex=1)
```



** PREGUNTA DIRIGIDA 4**. Intentad hacer un gráfico de cajas y bigotes para la variable TMB..nonsynonymous. según el tipo de tumor. Primero hacedlo sencillo ¿Veis alguna tendencia?

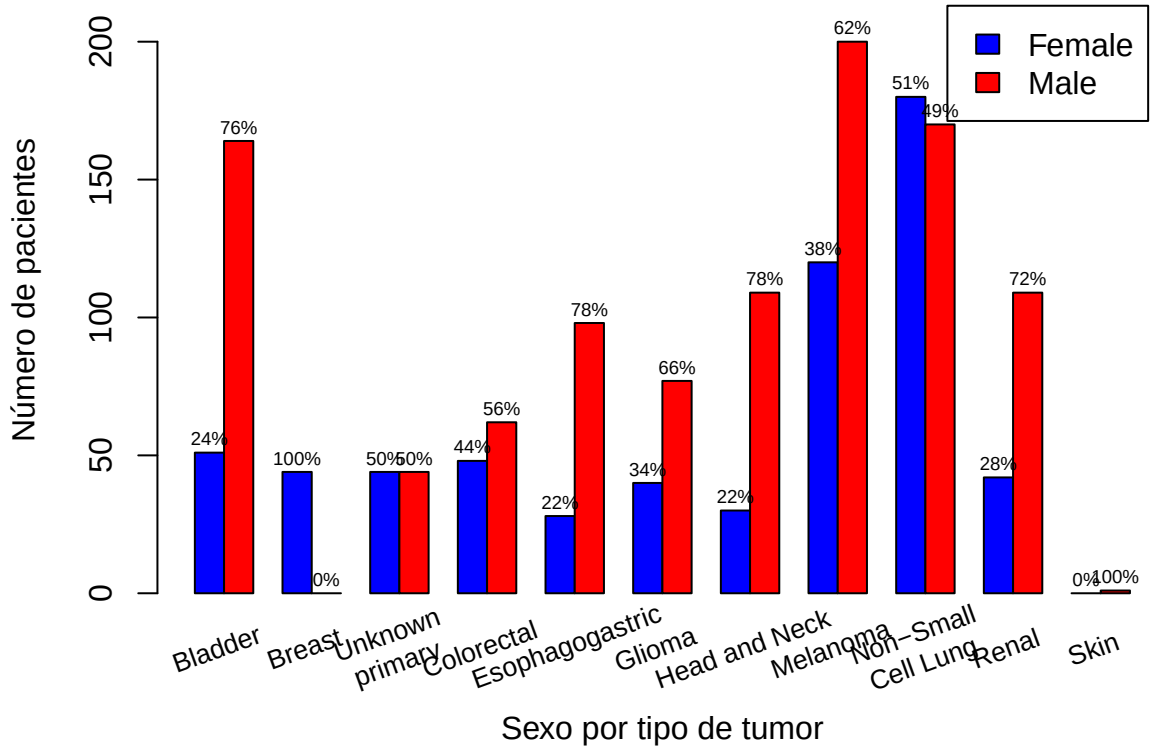
```
# Carga mutacional
boxplot(tmb$TMB..nonsynonymous.~tmb$Cancer.Type,ylab="TMB")
```



Estudiemos ahora algunas de las variables para cada tipo de tumor. Vemos que, excepto en el caso de cáncer de mama, que solo se da en mujeres, y en los tumores primarios de origen desconocido, cáncer colorectal y cáncer de pulmón de célula no pequeña, la mayoría de los casos se dan en hombres, lo que es especialmente notable en el caso del cáncer de vejiga, renal, esofagogástrico y cáncer de piel no melanoma.

```
sex.freq.tumor<-table(tmb$Sex,tmb$Cancer.Type)
sex.tumor.bar<-barplot(sex.freq.tumor,
        beside=TRUE,
        legend=rownames(sex.freq.tumor),
        col=c("blue","red"),
        xlab="Sexo por tipo de tumor",
        ylab="Número de pacientes",
        ylim = c(0,max(sex.freq.tumor)+20),
        xaxt="n",
        cex.names=0.5)
text(sex.tumor.bar, sex.freq.tumor+5 ,
     paste0(apply(sex.freq.tumor,2,function(x){round(x/sum(x)*100)}), "%") ,cex=0.6)
```

```
tumor.labels <- c("Bladder","Breast","Unknown\nprimary","Colorectal",
                  "Esophagogastric","Glioma","Head and Neck","Melanoma",
                  "Non-Small\nCell Lung","Renal","Skin")
text(cex=0.8, x=colMeans(sex.tumor.bar)-.25, y=-20,
     tumor.labels,xpd=TRUE, srt=20)
```

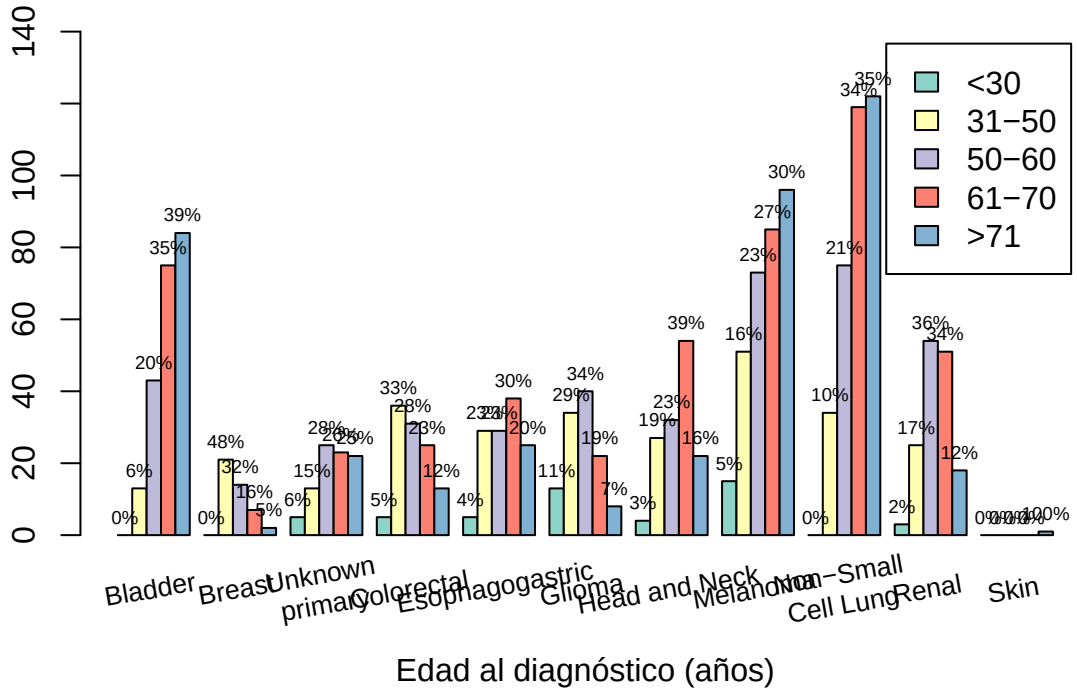


En cuanto a la edad, llama la atención que en el cáncer colorectal hay un porcentaje aparentemente mayor de pacientes de 31-50 años en comparación con los otros tipos.

```
age.freq.tumor<-table(tmb$Age.Group.at.Diagnosis.in.Years,
                      tmb$Cancer.Type)[c(1,3,4,5,2),]
age.tumor.bar<-barplot(age.freq.tumor,
        beside=TRUE,
        legend=rownames(age.freq.tumor),
        col=brewer.pal(5, "Set3"),
        xlab="Edad al diagnóstico (años)",
        ylim = c(0,max(age.freq.tumor)+20),
```

```
        xaxt="n",
        cex.names=0.5)
text(age.tumor.bar, age.freq.tumor+5,
     paste0(apply(age.freq.tumor,2,function(x){round(x/sum(x)*100)}), "%"),
     cex=0.6)

text(cex=0.8, x=colMeans(age.tumor.bar)-.25, y=-15,
     tumor.labels,
     xpd=TRUE, srt=10)
```

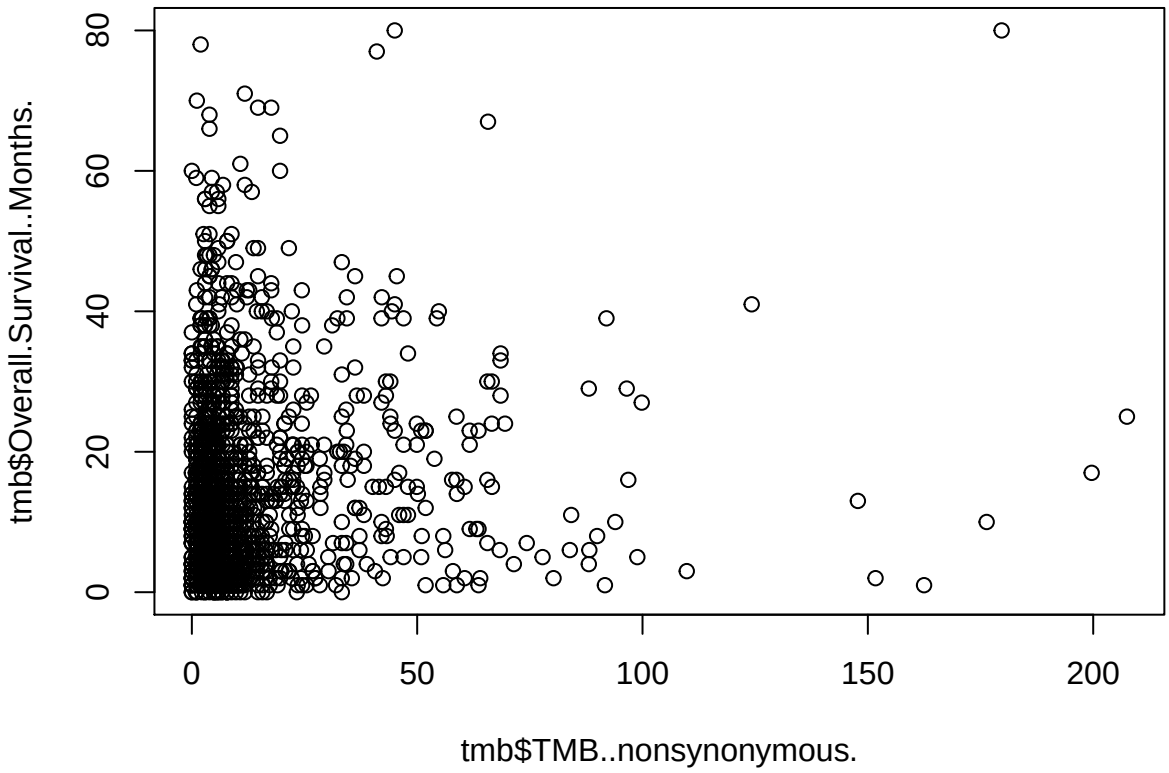


Nos hemos entretenido mirando cada una de las variables, pero lo importante de este artículo es ver si hay una relación entre carga mutacional y eficacia del tratamiento. La eficacia la podemos evaluar en términos de supervivencia global. Exploramos si la carga mutacional tiene relación con la supervivencia, ya que esto determinaría si los inhibidores de los puntos de control autoinmunes funcionan mejor en función de la carga mutacional. Sin embargo, vemos que no hay correlación entre estas dos variables, ni en la cohorte en general, ni en ningún tipo de cáncer en concreto, ya que los coeficientes de correlación son muy bajos.

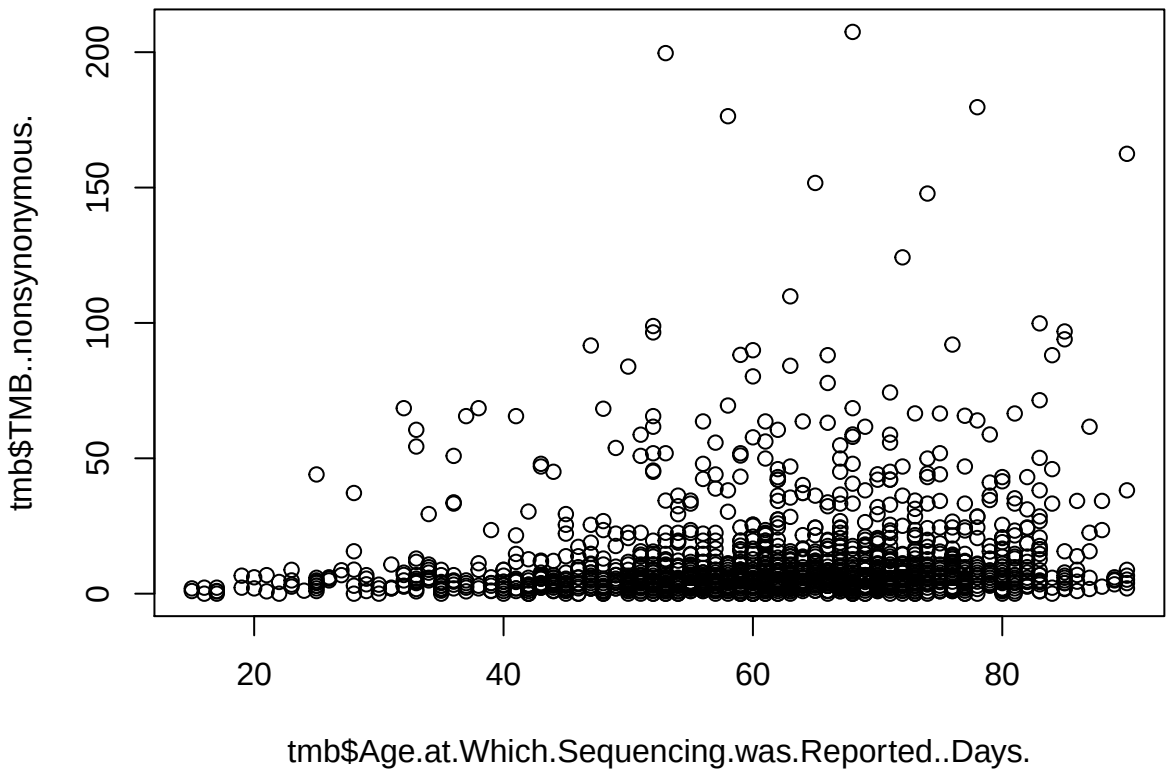
```r
cor(tmb$TMB,tmb$Overall.Survival..Months.)
```

```
## [1] 0.1066418
```

```r
cor.tmb.cancer<-sapply(unique(tmb$Cancer.Type),function(x){
        cor(tmb$TMB..nonsynonymous.[tmb$Cancer.Type==x],
            tmb$Overall.Survival..Months.[tmb$Cancer.Type==x])
        })
plot(tmb$TMB..nonsynonymous.,tmb$Overall.Survival..Months.)
```




```r
plot(tmb$Age.at.Which.Sequencing.was.Reported..Days.,tmb$TMB..nonsynonymous.)
```



Para acabar, os queremos mostrar algunas publicaciones que mencionan que sus figuras están hechas con ggplot2:

- https://mednexus.org/doi/full/10.1097/JBR.0000000000000110: creo que no tiene supplementary
- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7685753/
- https://www.sciencedirect.com/science/article/pii/S0092867420311417

5. Cómo seguir aprendiendo por tu cuenta

Un recurso muy sencillo, gratis y eficaz es swirl, que es un paquete para aprender R dentro de R. Vamos a instalarlo y ver cómo funciona. swirl: aprender R dentro de R

```
# install.packages("swirl")
```

También hay mucho material disponible en internet. Algunas recomendaciones:

- *An introduction to R" (31/10/2022). https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf
- Los libros de Rafael Irizarry, profesor de bioestadística en la Universidad de Harvard. Uno de sus libros es Introduction to Data Science, disponible gratuitamente en Leanpub (https://leanpub.com/datasciencebook). También hay la versión en español (http://rafalab.dfci.harvard.edu/dslibro/). Echad también un ojo a sus cursos en edX (https://www.edx.org/es/bio/rafael-irizarry). Aunque son de pago si queréis el certificado, el material está accesible gratuitamente durante bastante tiempo.
- Cursos gratuitos en Datacamp: https://www.datacamp.com/courses/free-introduction-to-r
- Esta página es muy resolutiva en temas de estadística: http://www.sthda.com/english/wiki/what-is-r-and-why-learning-r-programming Cualquier recurso que encontréis estará bien. La forma de aprender a usar R es usándolo. Ahora tenéis esta sesión en vuestro proyecto en Posit Cloud. Os animamos a que instaléis R y RStudio en casa, a que os entretengáis un rato a repasar lo que hemos visto, a que os salgan errores, a intentar solucionarlo,... y a que poco a poco os vayáis animando a utilizarlo para analizar vuestros propios datos. Veréis que cuando os deis cuenta de que R es como un folio en blanco en el que podéis pintar con absoluta libertad, ¡no querréis volver a usar otra cosa!

Samstein, Robert M., Chung-Han Lee, Alexander N. Shoushtari, Matthew D. Hellmann, Ronglai Shen, Yelena Y. Janjigian, David A. Barron, et al. 2019. "Tumor Mutational Load Predicts Survival After Immunotherapy Across Multiple Cancer Types." *Nature Genetics* 51 (2): 202–6. https://doi.org/10.1038/s41588-018-0312-8.