**R/R STUDIO**

**Introducción**

Dos comandos que utilizaras muy seguidos son:

* ***(Ctrl + L)*:** Se encarga de borrar la consola.
* ***(Ctrl + Enter)*:** Realiza la operación que selecciones.

Asignar un valor a una variable dentro de R se hace mediante el par de signos **<-** o directamente **=**

Para elegir una variable de un dataset usaremos ***$.*** Ejemplo: (nombre\_df$nombre\_variable) mtcars$hp

**Tipo de dato**

Dentro de la consola de R Studio, la función ***install.packages*** nos va a ayudar a instalar paquetes, como su nombre lo indica.

La función **str** nos va a mostrar la estructura que tiene el dataset que le pasemos.  
Dentro de la consola podemos obtener más información sobre nuestro dataset anteponiendo el signo ***?***. Por ejemplo: ?mtcars

Existen los datos tipo **int** y **num**, la diferencia es que num son números con decimal mientras que int son enteros.

**dbl** son float también

Para convertir un tipo de dato a otro utilizaremos la función **as.logical**

Ejemplos:

mtcars$vs = **as.logical**(mtcars**$**vs(variable))

**str**(mtcars)

**summary**(orange)

**Estructura del dataset**

**Summary:** función quenos devuelve un resumen estadístico del dataset.

**Transform**: función para modificar los valores de un dataset.

Ejemplos:

#Tranformamos la variable 'wt'(peso) de libras a kilos

wt = (mtcars**$**wt\*1000)/2

#Pasamos este cambio al dataset

mtcars.new = **transform**(mtcars,wt=wt\*1000/2)

**Vectores**

Un **vector** es un ente matemático que se usa para guardar información de un mismo tipo, dentro de R se crean los vectores con la función **c**. Esta **c** es una referencia a la palabra concatenar. Eso indica que todo lo que está en el paréntesis quedara concatenado en un vector

**sum** es una función que como su nombre lo indica, retorna la suma de los valores que le indiquemos.

Ejemplo:

tiempo\_platzi = **c**(25,5,10,15,10)

tiempo\_lectura = **c**(30,10,5,10,15)

tiempo\_aprendizaje = tiempo\_platzi + tiempo\_lectura

**Matrices**

Una matriz debe tener mismo tipo de datos, por otro lado, un dataframe puede tener diferentes.  
Para crear una matriz en R utilizaremos la función matrix cuyos argumentos son:

1. la información de los elementos.
2. **nrow**: número de filas.
3. **ncol**: número de columnas.
4. **byrow**: booleano para indicar si llenar la matriz por filas.

**colSums** es una función que por argumento recibe una matriz y te retorna la suma de los valores de sus columnas.

Para seleccionar rápidamente un elemento de una matriz solamente debemos indicar entre corchetes el número de la fila y de la columna.

Ejemplo:

tiempo\_platzi = **c**(25,5,10,15,10)

tiempo\_lectura = **c**(30,10,5,10,15)

tiempo\_matrix = **matrix**(**c**(tiempo\_platzi, tiempo\_lectura,

**nrow** = 2, **byrow** = TRUE))

dias = **c**("Lunes", "Martes", "Miercoles", "Jueves", "Viernes")

Tiempo = **c**("tiempo\_platzi"," tiempo\_lectura")

colnames(tiempo\_matrix) = días #Nombre de la columna

rownames(tiempo\_matrix) =Tiempo #Nombre de las filas

colSums(tiempo\_matrix)

final\_matrix = **rbind**(tiempo\_matrix, **c**(10,15,30,5,0)) # función para añadir una fila.

colSums(final\_matrix)

final\_matrix[1,5] #fila 1, columna 5

**Operadores para comparar y ubicar datos**

En R cuentas con los operadores de comparación comunes como **==** o **|,** pero además cuentas con el operador:

* **%in%** Que sirve para verificar si un elemento se encuentra en el dataset

Para hacer una selección de elementos de un vector, matriz o df podemos usar la función **subset**.

Podemos renombrar una variable para ello debemos tener instalado el paquete ***plyr***. En caso de no tener el paquete instalado solamente corremos en la consola el código **install.packages(“plyr”).**

Ejemplos:

mtcars[mtcars$cyl **<** 6,] #Simpre con coma al final

#Busca en el dataset que coches tienen menos de 6 cilindros

orange[orange$GDP.PC **>=** 15000,] #Que países tienen un Per Cápita mayor a 15000

neworange = **subset**(orange,Internet.penetration...population **>** 80

**&** Education.invest...GDP **>=** 4.5)

#Mas del 80% de la población recibe internet y tienen más de 4.5 de su PIB en educación

**rename**(orange,**c**("Creat.Ind...GDP" = "Aporte\_ec\_nja")) #Antiguo nombre = nuevo\_nombre

**Factores y listas**

**head**: es una función que nos retorna los primeros elementos de un dataset, por defecto nos retorna los primeros 6.

**tail**: función similar a head solamente que esta función nos retorna los últimos elementos.

Además de poder visualizar un dataset con **str** podemos instalar el paquete ***dplyr***: **install.packages(“dplyr”).** Una vez instalado usamos la función **glimpse**.

Una lista es un vector genérico que puede contener objetos de todo tipo, en R para crear una lista solamente debes llamar a la función **list** y pasarle como argumentos los elementos.

R no trabaja con índices en **0**, si no, que directamente comienza en **1.**

Ejemplo:

**head**(orangeec, n = 3) #n= y se especifica el número de filas a mostrar

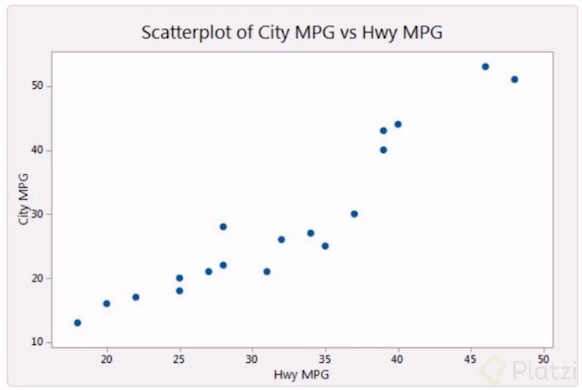
my\_vector = 1:8 #Del 1 al 8

my\_matrix = **matrix**(1:9, ncol = 3)

my\_df = mtcars[1:4,] #De la fila 1 a la 4 y todas las columnas

my\_list = **list**(my\_vector,my\_matrix,my\_df)

**Gráficas de dispersion**

Los ejes solamente pueden ser valores numéricos y los puntos no se pueden unir.

Ejemplo:

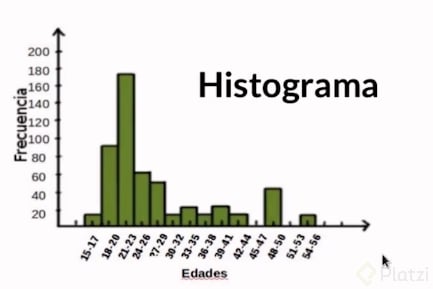
**plot**(mtcars$mpg ~ mtcars$cyl,

xlab = "cilindros", ylab = "millas por galón",

main = "Relación cilindros y millas por galón")

**Histogramas**

Sirve para ver la distribución de las frecuencias de una variable, es diferente a la gráfica de barras.

Se puede usar la librería **ggplot2**

Ejemplo:

**hist**(mtcars$hp,

xlab='Caballos',

main = 'Coches según caballos')

**ggplot**()+

**geom\_histogram**(**data**=mtcars,

**aes**(x=hp), **fill** = 'blue', #Color del relleno de las barras

**color** = 'red', #Color de las líneas exteriores de las barras

binwidth = 20) + #Ancho de barras

**labs**(x = 'Caballos', y = 'Cantidad de coches',

**title** = 'Caballos en los coches')+

**xlim**(c(80,280))+

**theme**(legend.position = 'none')+

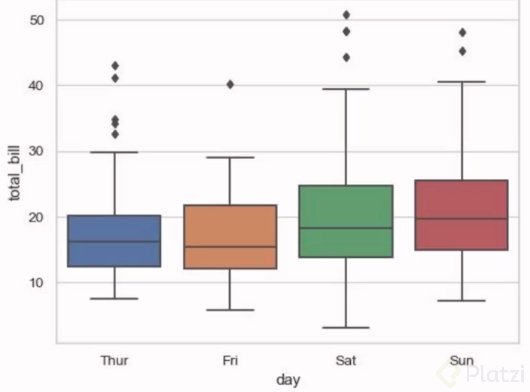
**theme**(panel.background = element\_blank(),

panel.grid.major = element\_blank(),

panel.grid.minor = element\_blank())#theme es para quitar el fondo de cuadrícula y oscuro que sale por defecto.

**Boxplot**

Los 5 puntos clave en estadística descriptiva se pueden visualizar en el box plot:

* **Primer cuartil**: es el piso de la caja o línea inferior.
* **Tercer cuartil**: es el techo de la caja o línea superior.
* **Mediana**: es la línea que se encuentra dentro de la caja.
* **Mínimo**: la extensión inferior de la caja.
* **Máximo**: la extensión superior de la caja.

Ejemplo:

#Libreria ggplot

**ggplot**(mtcars, aes(x=**as.factor**(cyl), y=hp, **fill**=cyl))+ #’as.factor’ es para que la variable se convierta en ‘factor’

**geom\_boxplot**(**alpha**=0.6)+ #alpha para cambiar el color de las cajas

**labs**(x="cilindros", y="caballos",

**title**= "Caballos en los coches")+

**theme**(legend.position = "none")+

**theme**(panel.background = element\_blank(),

panel.grid.major = element\_blank(),

panel.grid.minor = element\_blank())

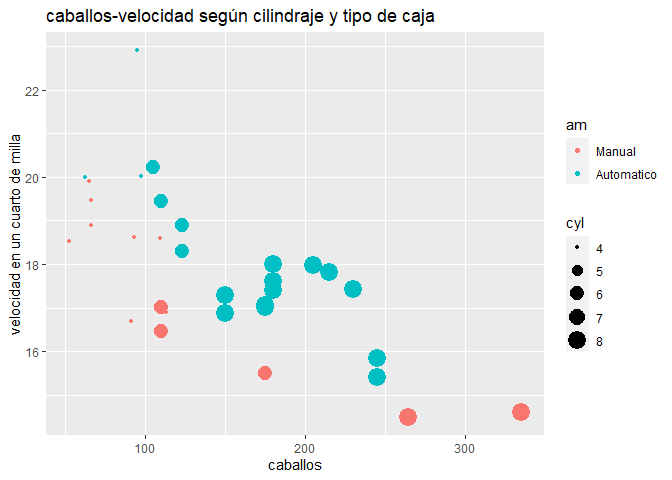
**Gráficas con más de dos variables**

**ggplot**(mtcars, **aes**(hp,qsec))+

**geom\_point**(**aes**(**color**=am, **size**=cyl))+

**labs**(x='caballos', y='velocidad en un cuarto de milla',

title='caballos-velocidad según cilindraje y tipo de caja')



**Correlación (gráficas)**

La función **pairs** nos permite cruzar todas las variables del dataset a modo de tabla donde el eje x de una gráfica corresponde a la columna donde se encuentra y el eje y a la fila.

**select**: función para seleccionar variables o columnas.

**filter**: función para filtrar datos de un dataset, retorna las filas que pasen el filtro.

Cuando en las gráficas los puntos están de forma **descendentes** es que hay una **correlación negativa** es decir a más de X, menos de Y.

Cuando los puntos van de forma **ascendente**, es al revés, **correlación positiva**. Cuanto más de X, más de Y.

Ejemplos:

newdata = **subset**(mtcars, select=c(2,7:8,11,12)) #Columna 2, columna de la 7 a la 8 y luego la 11 y 12

**pairs**(newdata)

**pairs**(mtcars[,-c(1,3,4,5,6,9,10)])

coches\_eficientes = **filter**(mtcars,mpg >= 30) #Filtramos que la variable 'mpg' sea mayor a 30

#Libreria stringr

merc = mtcars %>%

**filter**(**str\_detect**(model, "Merc")) #Busca un string en la variable 'model' que tenga la palabra 'Merc'

**Correlación (estadística)**

La función **cor** nos retorna la correlación entre los datos.

El valor de una correlación va de **-1** (correlación **negativa**(indirecta)) a **1**(correlación **positiva**(directa)), si se acerca a **0** **no hay** correlación.

Ejemplos:

**cor**(mtcars[,2:6])

**cor**(orange[,2:6], **use**=**'complete.obs'**) #Que 'use' solo las obs(observaciones) completas,

#es decir, que obvie los NA

**Desviación estándar**

La fórmula del coeficiente de variación nos es útil al momento de evaluar estos casos:  
**(desviación estándar) /(media) \* 100 = coeficiente\_var**

Si el coeficiente **es mayor al 25%** entonces los datos **no son homogéneos**, **varían mucho** respecto a la media

Dentro de R podemos sacar la desviación estándar con la función **sd** y la media con **mean**.

Conceptos:

**- Desviación estándar:** en un conjunto de datos es una medida de dispersión, que nos indica cuánto pueden alejarse los valores respecto al promedio (media). Es útil para determinar entre qué rango puede moverse una determinada variable. (Por lo tanto, es útil para buscar probabilidades de que un evento ocurra)

**- Coeficiente de variación:** este expresa la desviación estándar como porcentaje de la media, mostrando una mejor interpretación porcentual del grado de variabilidad que la desviación estándar.

Ejemplos:

**summary**(mtcars) #resumen estadístico

desviacion = **sd**(mtcars$mpg) # desviacion = 6,02

media = **mean**(mtcars$mpg) # media = 20,09

coef\_variacion = **(desviacion/media) \*100** # = 29,99% se pasa del 25% por lo que hay

# desviación de los datos respecto

#a la media. Por lo que nos apoyaremos en la mediana.

#Quitamos valores nulos

media = **mean**(orange$Creat.Ind...GDP, na.**rm** = TRUE) # rm = remove. Elimina los NA

**Ajustar datos (crear columnas, filas..)**

Usamos **mutate** para crear columna nueva

Ejemplos:

mtcars = mtcars **%>%**

**mutate**(velocidad\_cuarto\_milla = **ifelse**(qsec > 16,

"Menos de 16 seg",

"Más de 16 seg"))

orange **%>%**

**arrange** (**desc** (Creat.Ind…GDP))

top\_econ\_naranaja = orange **%>%**

**filter**(Country **%in%** **c**("Mexico", "Panama", "Argentina",

"Colombia", "Brasil"))

**R-Markdown**

Es momento de generar nuestro documento con todas las gráficas y observaciones que hemos realizado a nuestro dataset, para ello necesitamos instalar el paquete **rmarkdown**: **install.packages(“rmarkdown”).**

R Markdown nos permite generar archivos en formato **HTML, PDF y Word**. La **mejor** opción es trabajar en un formato **HTML** para compartirlo por internet y posteriormente convertirlo ya sea a PDF o Word.

Dentro de nuestro archivo de R Markdown iremos escribiendo con sintaxis de markdown el archivo y cuando escribamos código por sí solo se va a ejecutar y añadir las gráficas o cálculos a nuestro archivo.