UNAL: La Paz

INTRODUCCIÓN A R

Olga Maria Caballero Yance
Universidad Nacional Sede La Paz, Colombia

Yeiner Jose Duran Diaz
Universidad Nacional Sede La Paz, Colombia

Yefri Esteban Sanchez Almanza
Universidad Nacional Sede La Paz, Colombia

Yuleidys Mejia Gutierrez
Universidad Nacional Sede La Paz, Colombia

***Abstract*—R es un entorno de programación libre que se utiliza para el procesamiento y análisis estadístico de datos implementado en el lenguaje S de GNU, aunque para algoritmos computacionalmente exigentes se emplean lenguajes como C, C++ o Fortran. (Astudillo L, 2022). R es un software de programación utilizado en muchos ámbitos de investigación científica y matemáticas en cuál proporciona un sin fin de herramientas estadísticas y gráficas que permiten a sus usuarios generar o definir sus propias funciones. Esta herramienta también puede ser utilizada al momento de realizar cálculos matemáticos muy grandes. Una de las ventajas que ofrece esta aplicación es la fácil accesibilidad que se le tiene a esta; la cual funciona con paquetes fáciles y livianos al momento de descargar.**

Básicamente este es un conjunto de programas integrados el cual sirve para el manejo de datos, simulaciones, cálculos y realización de gráficos más que todo del índole matemático-estadístico.

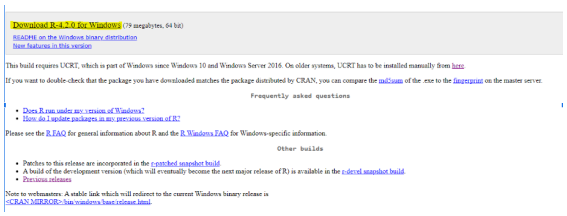
Abstract
R is a free programming environment used for statistical data processing and analysis implemented in the GNU S language, although for computationally demanding algorithms languages such as C, C++ or Fortran are used (Astudillo L, 2022).

R is a programming software used in many fields of scientific research and mathematics in which it provides an endless number of statistical and graphical tools that allow its users to generate or define their own functions. This tool can also be used when performing very large mathematical calculations. One of the advantages that this application offers is the easy accessibility that it has; which works with easy and light packages at the moment of downloading.

Basically this is a set of integrated programs which is used for data management, simulations, calculations and graphing mostly of a mathematical-statistical nature.

Pasos para la instalación local de R

Si queremos instalar R en nuestro computador solo basta con buscar en nuestro navegador “Descargar R”. La primera opción en el buscador es la última versión de R para Windows, Para esto tenemos que tener en cuenta que no todos los computadores aceptaran esta versión de windows puesto a que para esta versión se necesita desde Windows 10 y Windows Server 2016 y que cuenta con 79 megabytes, 64 bit. Si cuenta con las condiciones anteriores puede darle click en “Download R4.2.0 for Windows”.

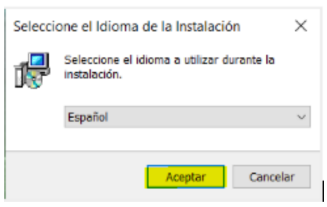


Si no tiene las características que se necesitan para esta versión puede buscar una version antigua de R que cumpla con nuestras caracteristicas en el siguiente link: https://cran.r-project.org/bin/windows/base/old/

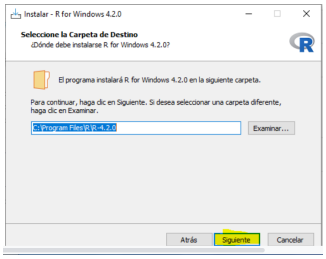
Tengamos en cuenta que en este caso estamos trabajando con windows, pero se puede instalar R en Linux, Mac o Windows. http://cran.rediris.es/

Cuando tengamos la versión adecuada para nuestro dispositivo se le da click a esta e inmediatamente se procede a descargar.

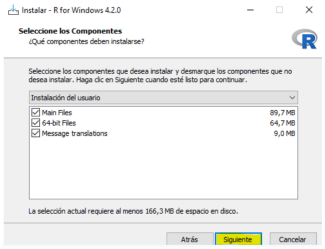
Cuando la descarga esté lista, nos dirigimos a nuestros archivos e iniciamos la instalación de este. Lo primero que nos pedirá es seleccionar el idioma:

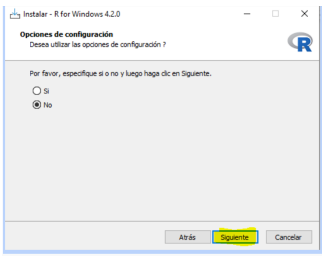


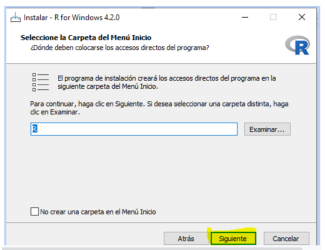
Se aceptan las condiciones de este y se selecciona donde se instalará R.

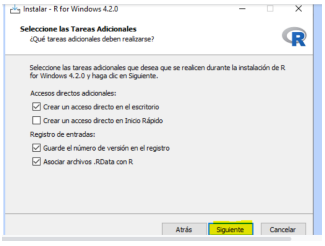


Se seleccionan los componentes que deben instalarse.

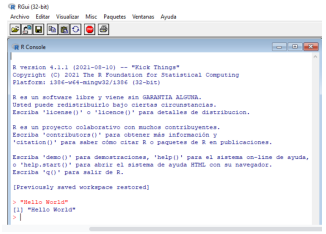








Luego de haber aceptado lo mencionado anteriormente este iniciara su instalación, esperamos un momento y listo, tendremos instalado R en nuestro equipo.



Si se desea, se puede instalar como paso siguiente Rstudio, RStudio es el entorno gráfico que nos permite utilizar de una manera más cómoda a R. Nos permite autocompletar el código, detectar errores en la sintaxis e incluye un sistema de menús de ayuda. Para instalarlo iremos a la web https: //www.rstudio.com/.

Alternativas de ejecución de R en la web

Al momento momento de ejecutar códigos en R la red cuenta con muchas alternativas que nos permiten llevar a cabo esto, estas son páginas webs en las que se puede acceder de forma gratuita y fácil en donde para poder no necesariamente se debe saber mucho de programación para poder manipularlas.

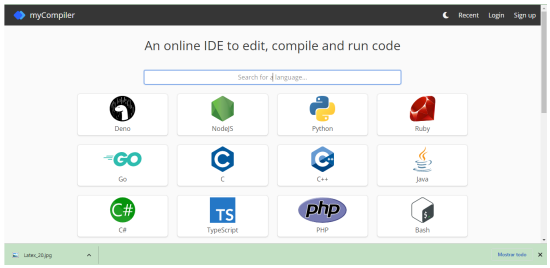
Algunos de estas páginas que cumplen las veces de compiladores de codigos para nuestra plataforma de r son los siguientes:

Onecompiler

A la hora de necesitar compilar algún código es uno de los compiladores en línea más completos debido a que cuenta con diversas funciones para R, sin embargo también permite ser usado para ejecutar códigos con muchos más lenguajes de programación. Ingresa a Onecompiler por medio del siguiente vinculo: https://onecompiler.com/

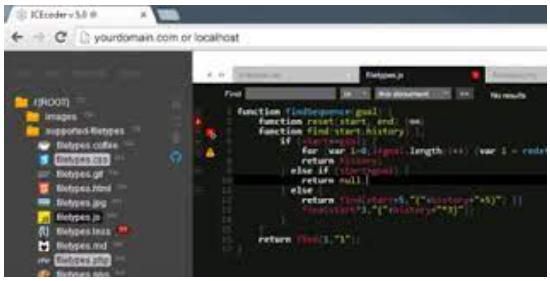
My compliler

Es otra página web que cumple esta función, este de entrada nos presenta diversos lenguajes de programación en donde uno de estos es r. Además de ser sencillo al momento de usarlo no es necesario iniciar sesión o realizar algún tipo de registro en este. Se ingresa haciendo click en el siguiente link: https://www.mycompiler.io/



Como otra opción presentamos es **Paiza**, esta es una página web la cual nos permite de forma gratuita compilar códigos en más de 20 lenguajes de programación en donde incluso se encuentra incluido R. Para su acceso se puede realizar ingresando por medio del siguiente link: https://paiza.io/es

Existen muchos compiladores fáciles de usar y de forma gratuita que permiten ejecutar códigos para el lenguaje de R, como uv.es, rdrr, entre otros. Tan solo debemos utilizar el que nos parezca más cómodo.



Descripción de las estructuras de datos básicas de R

Datos básicos en R

R es un lenguaje de programación que permite operar con diferentes tipos de datos, los cuales se pueden apreciar a continuación:

Tipo de dato	Descripción	Definición
Numeric	Números decimales	`numero <- 1.0`
Integer	Números enteros	`int <- 1`
Character	Cadenas de texto	`str <- "un texto"`
Complex	Números complejos	`comp <- 3+2i`
Logical	Verdadero (TRUE) o falso (FALSE). Es a menudo el resultado de operaciones lógicas.	`a <- 1; b <- 2; a < b`
Factor	Este no es estrictamente un tipo de dato, pero vale la pena describirlo aquí. Una variable factor es una variable categórica. Los vectores de caracteres a menudo se almacenan como factores para explotar funciones para tratar datos categóricos. Por ejemplo, en análisis de regresión.	Aplique `as.factor()` a un vector de caracteres.

Tabla de los diferentes tipos de datos que se manejan en R.

De esta manera, se puede apreciar que R permite un manejo fácil en cualquier tipo de dato. Para este manejo R utiliza, a su vez, diferentes estructuras de datos.

Estructuras de datos

Vectores:

"La estructura más simple es el vector, que es una colección ordenada de números"(Introducción a R. 2000. pp 7).

A cada vector, como cualquier otra operación en R, se le puede asignar una variable de manera que se conserve allí toda la información correspondiente; respecto a los valores que dicho vector conserve, estos pueden ser N cantidad de datos y los tipos pueden ser cualquiera de los que sean necesarios. Por ejemplo:

Para crear un vector, que en este caso se considerará como x, y que tenga cinco números tales como: $10.4, 5.6, 3.1, 6.4 y 21.7$, se usa la orden o comando:

$> x < -c(10.4, 5.6, 3.1, 6.4, 21.7)$

Ahora, observe el siguiente ejemplo:

$> x < -c(2, 5.6, M, 6.4, O)$

Se puede observar que este vector puede conservar datos de tipo: int, float o str.

Algunas operaciones entre vectores

R en su lenguaje contiene innumerables funciones para realizar operaciones en cualquier cantidad de números. Dichas funciones también se pueden aplicar en los vectores. Estas funciones en su mayoría son: Operadores elementales como suma$(+)$, resta$(-)$, multiplicación $(*)$, división$()$; funciones como $log(logaritmo)$, exp$(exponencial)$, sin$(seno)$, cos$(coseno)$, tan$(tangente)$, sqrt$(raízcuadrada)$, $max(x)$(Genera el número máximo del vector), $min(x)$(Genera el número mínimo del vector); $sum(x)$(Suma de todos los datos del vector), $prod(x)$(Es el producto de todos los elementos), $length(x)$(Devuelve la longitud del vector, o la cantidad de elementos del vector).

Listas

En R las listas son componentes consistentes en una colección ordenada de objetos. La esencia de una matriz puede estar establecida por un vector numérico, un valor lógico, una matriz y una función. Para visualizar lo anteriormente dicho, tenemos la siguiente función:

$$> Lst < -list(nombre = "Pedro", esposa = "María", no_hijos = 3, edad_hijos = c(4, 7, 9))$$

Lst se considera una lista y para llamar algún componente dentro de ella se puede realizar de la siguiente manera:

$> Lst[[4]]$

Este comando llama al componente 4 en la lista; ahora bien, cómo es un componente vector, los elementos de dicho vector se pueden llamar de manera individual de la siguiente manera:

$> Lst[[4]][1]$

Las lista por su parte también se pueden concatenar haciendo uso de la función $c()$, como se muestra a continuación:

$> LisABC < -c(LisA, LisB, LisC)$

Matrices

Para construir matrices en R podemos utilizar las siguientes funciones:

matrix(vector,...): Construye una matriz con las entradas del vector. Tiene los siguientes parámetros básicos:

- **byrow:** Indica si se organizan por filas (TRUE), o por columnas (FALSE).
- **nrow o ncol:** Número de filas o de columnas.

```
x <- 1: 9
matrix(x, nrow = 3)
matrix(x, ncol = 4, byrow = TRUE)
```

```
> matrix(x, nrow = 3)
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> matrix(x, ncol = 4, byrow = TRUE)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9    1    2    3
```

También se pueden hacer matrices mediante concatenación de vectores:
-**cbind:** Concatenar matrices y vectores por columnas.
-**rbind:** Concatenar matrices y vectores por filas.

```
155  a <- c(1,3,6,9)
156  b <- c(5,6,9,7)
157  c <- rep(1,4)
158  rbind(a,b,c)
159  cbind(a,b,c)
160
```

```
> a <- c(1,3,6,9)
> b <- c(5,6,9,7)
> c <- rep(1,4)
> rbind(a,b,c)
  [,1] [,2] [,3] [,4]
a    1    3    6    9
b    5    6    9    7
c    1    1    1    1
> cbind(a,b,c)
     a b c
[1,] 1 5 1
[2,] 3 6 1
[3,] 6 9 1
[4,] 9 7 1
>
```

Entradas y trozos de matrices
En general las entradas de la matriz se indican como en los vectores pero con una función bidimensional que es $M[i, j]$, donde primero se presenta la fila y luego la columna.
$M[i,]$: Fila.
$M[, j]$: Columna.
Usando vectores de índices en vez de índices, se obtienen submatrices determinadas por las filas y columnas indicadas.

```
161  A <- matrix(1:20, nrow = 5)
162  A[3,4]
163  A[,4]
164  A[3,]
165  A[,1:2]
166
```

```
> A <- matrix(1:20, nrow = 5)
> A[3,4]
[1] 18
> A[,4]
[1] 16 17 18 19 20
> A[3,]
[1]  3  8 13 18
> A[,1:2]
     [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
[4,]    4    9
[5,]    5   10
>
```

- la función diag que nos da la matriz diagonal.

```
167  B <- matrix(1:16, nrow= 4)
168  print(B)
169  diag(B)
170
```

```
> B <- matrix(1:16, nrow= 4)
> print(B)
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
> diag(B)
[1]  1  6 11 16
```

Las funciones utilizadas para las dimensiones de las matrices son:
-**nrow:** Número de filas.
-**ncol:** Número de columnas.
-**dim:** Vector con ambas dimensiones.

```
171  B <- matrix(1:12, nrow= 4)
172  print(B)
173  nrow(B)
174  ncol(B)
175  dim(B)
```

```
> B <- matrix(1:12, nrow= 4)
> print(B)
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
> nrow(B)
[1] 4
> ncol(B)
[1] 3
> dim(B)
[1] 4 3
>
```

Algunas funciones que utilizan los vectores también se pueden aplicar en las matrices, así como a las filas y columnas de dicha matriz.

- **colSums:** Vector de sumas de columnas.
- **rowSums:** Vector de sumas de filas.
- **colMeans:** Vector de medias de columnas.
- **rowMeans:** Vector de medias de filas.

```
77  mean(B)
78  rowSums(B)
79  colSums(B)
80  colMeans(B)
```

```
> rowSums(B)
[1] 15 18 21 24
> colSums(B)
[1] 10 26 42
> colMeans(B)
[1]  2.5  6.5 10.5
>
```

la función **apply(matriz, FUN=función, MARGIN=...)** - **MARGIN=1: Fila a fila**. - **MARGIN=2: Columna a columna**. - **MARGIN=c(1,2): Entrada a entrada**.

```
apply(B, FUN=prod, MARGIN = 1)
```

```
> apply(B, FUN=prod, MARGIN = 1)
[1]  45 120 231 384
>
```

OPERACIONES ALGEBRAICAS CON MATRICES

Suma: +

Producto por escalar: *

Producto: *

Transpuesta: t

Determinante: det

Inversa: solve

Rango: qr(...)$rank

```
#OPERACIONES CON MATRICES
A <- matrix(1:9, nrow = 3)
B <- matrix(2, nrow = 3, ncol = 3)
C <- matrix(1:12, nrow = 3)
D <- matrix(1:4, nrow = 2)
#Suma
A+B
#Resta
A-B
# Escalar por matriz
2*A
#
A*B
#Multiplicacion
A%*%B
#Transpuesta
t(C)
#Determinante
det(B)
#Inversa
solve(D)
#Rango
qr(B)$rank
```

```
> A <- matrix(1:9, nrow = 3)
> B <- matrix(2, nrow = 3, ncol = 3)
> C <- matrix(1:12, nrow = 3)
> D <- matrix(1:4, nrow = 2)
> #Suma
> A+B
     [,1] [,2] [,3]
[1,]    3    6    9
[2,]    4    7   10
[3,]    5    8   11
> #Resta
> A-B
     [,1] [,2] [,3]
[1,]   -1    2    5
[2,]    0    3    6
[3,]    1    4    7
> # Escalar por matriz
> 2*A
     [,1] [,2] [,3]
[1,]    2    8   14
[2,]    4   10   16
[3,]    6   12   18
> #
> A*B
     [,1] [,2] [,3]
[1,]    2    8   14
[2,]    4   10   16
[3,]    6   12   18
> #Multiplicacion
> A%*%B
     [,1] [,2] [,3]
[1,]   24   24   24
[2,]   30   30   30
[3,]   36   36   36
```

```
> #Transpuesta
> t(C)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
> #Determinante
> det(B)
[1] 0
> #Inversa
> solve(D)
     [,1] [,2]
[1,]   -2  1.5
[2,]    1 -0.5
> #Rango
> qr(B)$rank
[1] 1
>
```

Data Frames

Los Data Frames o conocidos en español como Hojas de Datos se pueden considerar como matrices en las que las columnas contienen elementos de diferentes atributos. Al momento de imprimirse se puede visualizar como una matriz, diferenciándose esta de las matrices en la practicidad de la visualización y organización de los datos con los cuales se esté trabajando.

La función para crear un Data Frame es:

$> Datos < -data.frame($
$Nombre = Alumnos,$
$Edad = Edades$
$)$

Donde Alumnos puede ser la variable de un vector cuyos elementos sean de tipo str, y Edades un vector donde sus elementos sean de tipo int o float.

Ahora para llamar o indexar en un Data Frames se puede realizar con el signo $ o de la manera en que se hace llama alguna columna o elemento en una matriz.

$> Datos[, 1]$

$> Datos\$Nombre$

Attach y detach

Estas funciones se pueden ejecutar para acceder de forma directa a cada una de las columnas del Data Frames, con attach() se inicia el proceso y se finaliza con detach().

$> attach(Datos)$

$> Nombre$

$> detach(Datos)$ #Se finaliza todo el acceso directo de manera que si se llama la variable Nombre no aparecerá nada.

Agregar y eliminar columnas y filas en un Data Frames

Una de las funciones más populares para agregar columnas y filas en un data frame es haciendo uso de las siguientes funciones:

$> cbind()$ # Esta agrega columnas

$> rbind()$ # Es la que agrega filas

Para eliminar columnas en un data frame se puede hacer uso del siguiente comando:

Primero se crea otra variable que sea un data frame con las columnas que se desean conservar, se puede observar de la siguiente manera:

$> Datos1 < -Datos[, -c(1)]$

O se puede hacer de la siguiente manera:

$> Datos1 < -Datos[, c(Edad)]$

Con cualquiera de las dos se estaría eliminando la columna “Nombre”.

Ordenar y Filtrar

Para ordenar los datos en un data frame se puede hacer uso de la función order ( ), de la siguiente manera:

$> x < -order(Datos\$Edad)$#Para ordenar los datos de menor a mayor

$> x < -order(-Datos\$Edad)$ #Para ordenar los datos de mayor a menor

Para realizar el filtrado, cabe resaltar que esta es una acción de sustraer una parte del data frame en caso de que se cumpla un condicional necesario.

Esta acción se puede realizar por medio del siguiente código: $> subset(Datos, Edad == 8)$

$> subset(Datos, Edad < 8)$

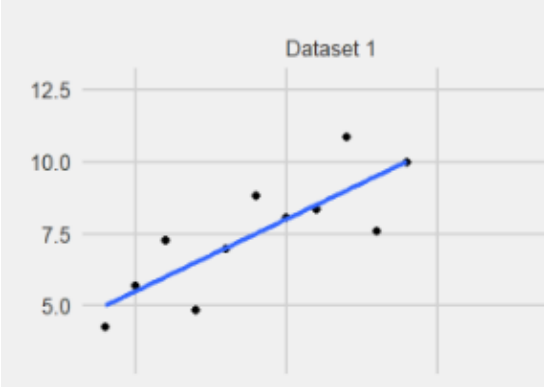
$> subset(Datos, Edad > 8)$

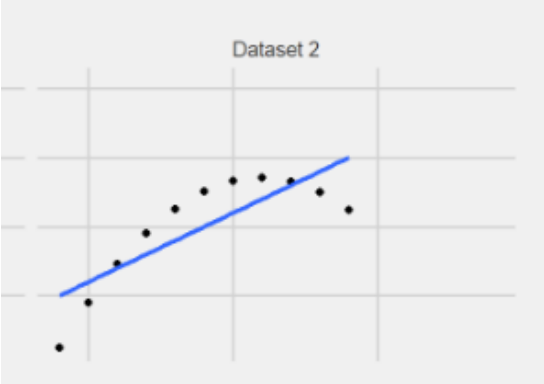
De esta manera se pueden obtener los datos que el programador o usuario desee.

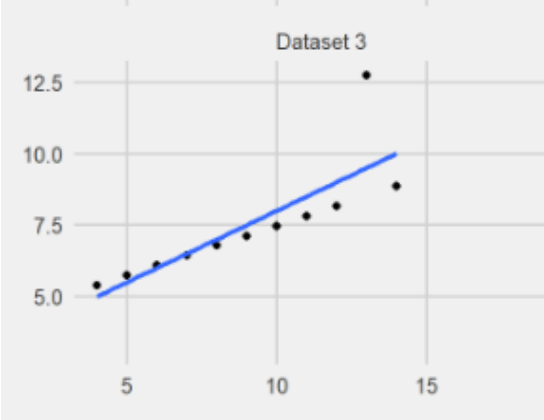
Visualización de los datos

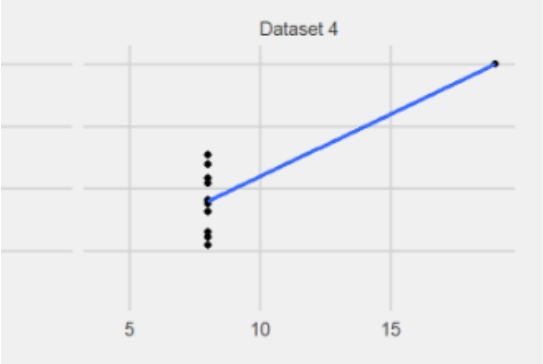
La visualización de datos ha ganado espacio en diversas publicaciones y en las últimas décadas ya constituye una disciplina en sí misma. Su relevancia podría justificarse de muchas maneras, pero hay un punto que es especialmente relevante para el análisis de datos: los parámetros y coeficientes con los que solemos trabajar no siempre son tan simples de interpretar como pensamos. Por ejemplo, algunos autores recomiendan fuertemente graficar las predicciones del modelo ante distintos valores, antes que los coeficientes del modelo (McElreath, 2016).

Este punto dista de ser uno moderno. En 1973 el estadístico Francis Anscombe aportó la siguiente evidencia sobre cómo muchas de las variables que solemos tomar como de resumen, o incluso de relación entre ellas, puede engañarnos si no visualizamos correctamente los datos:









Si prestan atención, van a poder ver que la nube de puntos de cada uno de los datasets es bien distinta. El primero de arriba a la izquierda parece un scatter plot alrededor de una media, que podría modelarse linealmente. El segundo, el de arriba a la derecha, parecería tener una relación más cuadrática. El tercero, el de abajo a la izquierda, muestra una relación lineal aparentemente con baja volatilidad, pero un punto bien alejado de esa recta, mientras que el último muestra casi ninguna relación entre x e y, con la excepción de un punto muy alejado del resto.

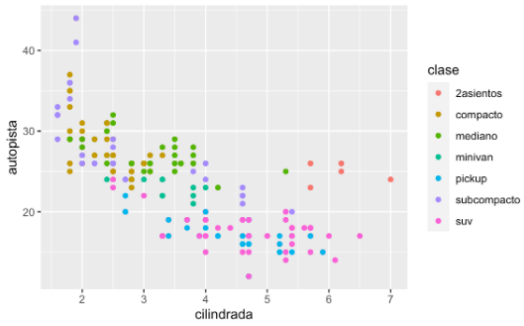
¿Pero que sucede cuando trazamos una línea que minimiza la distancia cuadrática entre los puntos? La recta (azul) es ¡la misma! ¿Y qué pasa con los promedios de las variables y la correlación entre ellas? También son iguales:

```
## # A tibble: 4 x 4
##   dataset   promedioX promedioY corXY
##   <chr>         <dbl>     <dbl> <dbl>
## 1 Dataset 1         9      7.50 0.816
## 2 Dataset 2         9      7.50 0.816
## 3 Dataset 3         9      7.5  0.816
## 4 Dataset 4         9      7.50 0.817
```

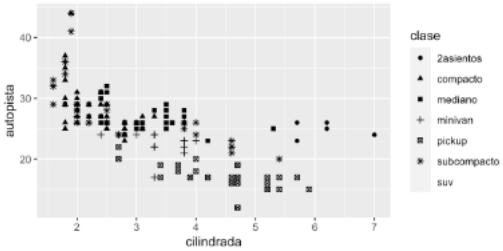
De esta manera queda en claro que, en algunas ocasiones, trabajar con resúmenes de nuestros datos

puede llevarnos a extraer conclusiones erróneas sobre cómo deberíamos analizarlos. Durante los años se fue mejorando este punto que hizo Anscombe y... ahora sabemos que una nube de puntos que se parece a un dinosaurio: si nos guiamos solo por las clásicas medidas de resumen, son indistinguibles.

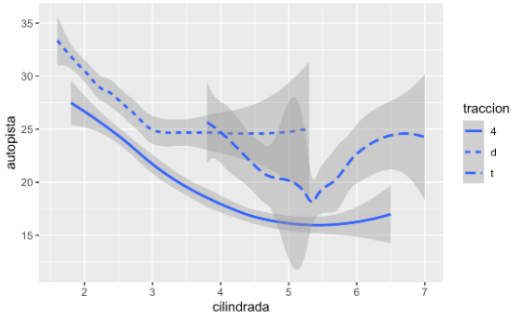
Creando un gráfico con ggplot Una de las funciones mas practicas en R es la función $ggplot()$, esta generalmente facilita la visualización de los datos conservados en un data frame y los muestra en pantalla por medio de un grafico de distribución, que conserva los puntos y su relación. La función se puede visualizar como anteriormente se mostro:
$> ggplot(data = millas) + geom\_point(mapping = aes(x = cilindrada, y = autopista, color = clase))$
Si se observa con detalle la función se puede deducir que los datos pueden alterarse en su visualización para que se puedan analizar de manera mas facil. La imagen de esta función queda de la siguiente forma:



Ahora bien, si se desean expresar dichos puntos de otra manera, basta con hacer uso de la subfunción $shappe$ en ggplot.
$> ggplot(data = millas) + geom\_point(mapping = aes(x = cilindrada, y = autopista, shape = clase))$



Por otro lado, $ggplot()$ se puede ayudar de la función $geom\_smooth()$, esta función ayudará a dibujar una linea de tendencia en los datos. $> ggplot(data = millas) + geom\_smooth(mapping = aes(x = cilindrada, y = autopista, linetype = traccion))$



REFERENCES

1. Montane, M. (2020, 11 mayo). 3 Visualizaciones de datos en R — Ciencia de datos para curiosos. Bookdown. https://bookdown.org/martinmontaneb/CienciaDeDatosParaCuriosos/visualizaciones-de-datos-en-r.html
2. P. Bravo, & F. Salgado. (2019). Introducción a R y SIG. ”Instituto de Estudios de Régimen Seccional del Ecuador”. Tomado de: https://bookdown.org/chescosalgado/intro_r/tipos-y-estructuras-de-datos-en-r.html
3. Notas sobre R: Un entorno de programación para Análisis de Datos y Gráficos.(2000-05-16.) R Development Core Team.
4. D. Pacios.(2018-2019). Curso de R. Tomado de: https://www.ucm.es/data/cont/docs/1346-2019-05-04-Curso%20de%20R.pdf
5. Estructura de datos en R. Análisis de datos usando el paquete estadístico R. Tomado de http://halweb.uc3m.es/esp/Personal/personas/jmmarin/esp/ManualR/PRINCIPAL.htm