INSTITUTO DATA SCIENCE ARGENTINA

Elementos de probabilidad y estadística

Repaso de conceptos de Probabilidad y Estadística

Se escapa al alcance de este curso dar una introducción sistemática a los conceptos fundacionales de la probabilidad y la estadística. Por eso planteamos un repaso operativo de herramientas básicas.

Si algunos de los participantes no se sienten cómodos con sólo la parte operativa pueden buscar una justificación adecuada en un texto formal de matemáticas sobre la materia.

Si el alumno tiene entrenamiento formal en probabilidades y estadística este repaso el completamente superfluo.

El estudio de probabilidades y estadística se suele dividir en dos grandes partes:

- Estadística descriptiva
- Inferencia Estadística

Estadística Descriptiva

Probabilidad
Lo primero que tenemos que definir es lo que vamos a llamar una "Variable Aleatoria"

Vamos a pensar a la variable aleatoria como una máquina que puede producirnos valores de acuerdo a su funcionamiento interior (que nos es desconocido)

Un ejemplo natural es un dado. Un dado produce un valor cada vez que lo lanzamos.

Una secuencia de lanzamientos podría ser:

3, 2, 6, 1, 2, 5, 3, 4

Podemos preguntarnos: ¿Cual es la probabilidad de que la próxima tirada de, por ejemplo, 5?

Para responder debemos recordar que:

- La tirada va a producir uno de los siguientes resultados: 1, 2, 3, 4, 5, 6
- Todos los resultados son igualmente probables. (Esto es equivalente a decir que un dado está balanceado)

- La suma de las probabilidades de todos los resultados posibles tiene que ser 1

Puesto en símbolos:

- P(1) = P(2) = P(3) = P(4) = P(5) = P(6)
- P(1) + P(2) + P(3) + P(4) + P(5) + P(6) = 1

Consecuentemente si reemplazo en la segunda ecuación todas las probabilidades por P(5) me queda:

$$P(5) + P(5) + P(5) + P(5) + P(5) + P(5) = 1$$

Que es lo mismo que decir:

$$6 \times P(5) = 1$$

Lo que no puede sino llevarnos a que:

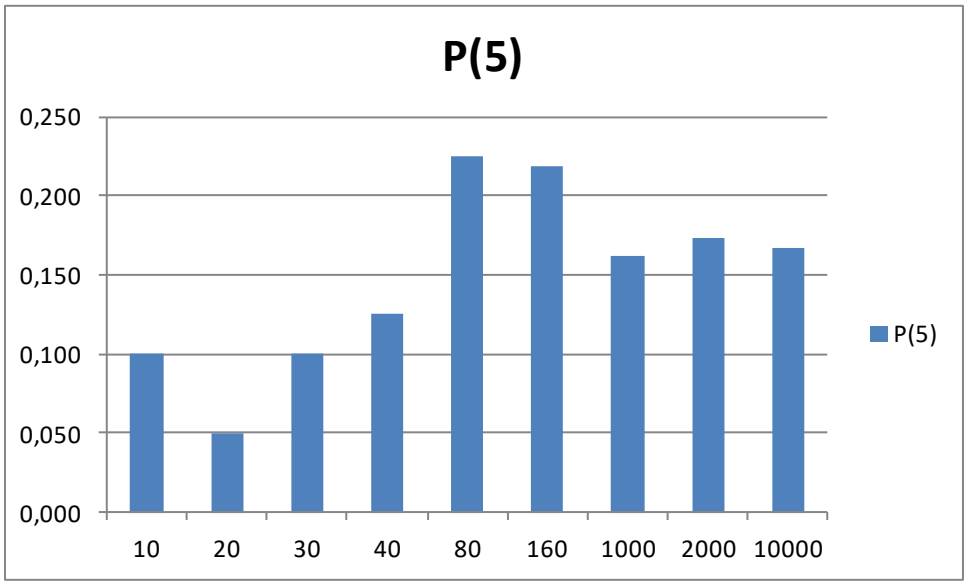
$$P(5) = 1/6$$

Esto nos lleva a una definición intuitiva del concepto de probabilidad: P(resultado X) es la cantidad de veces que aparece el resultado X al interrogar una variable aleatoria dividido por la cantidad total de veces que interrogamos la variable aleatoria en el límite en que hacemos esta operación muchas veces (en la jerga matemática, tendiendo a infinito)

Ejercicio 3.1
Conseguir un dado y lanzarlo 10 veces. Anotar los resultados obtenidos y calcular P(1) con 10 valores.

- Lanzarlo 10 veces más y calcular P(1) con 20 valores
- Lanzarlo 10 veces más y calcular P(1) con 30 valores
- Lanzarlo 10 veces más y calcular P(1) con 40 valores
- Graficar los cuatro valores de P(1) obtenidos
- Intercambiar los datos con un compañero y calcular P(1) con 80 valores
- Incorporar al gráfico el nuevo valor

Los resultados deberán parecerse a:



Ejercicio 3.2
Consiga una moneda y, lanzándola al aire, trate de reproducir el experimento anterior para P(cara)

- Lanzarlo 10 veces más y calcular P(cara) con 20 valores
- Lanzarlo 10 veces más y calcular P(cara) con 30 valores
- Lanzarlo 10 veces más y calcular P(cara) con 40 valores
- Graficar los cuatro valores de P(cara) obtenidos
- Intercambiar los datos con un compañero y calcular P(cara) con 80 valores
- Incorporar al gráfico el nuevo valor

Promedio:
Cualquier estudiante puede calcular promedios. Es necesario para su supervivencia como alumno el poder calcular su nota promedio.

Como todos recuerdan para calcular un promedio se suman todos los valores obtenidos y se divide por la cantidad de valores obtenidos:

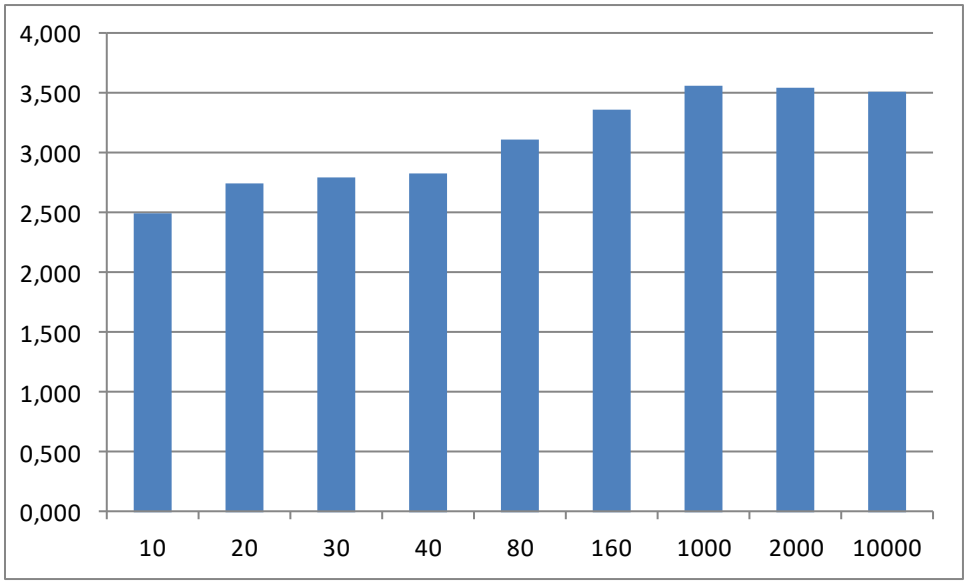
$$P = \frac{V_1 + V_2 + V_3 + V_4 + V_5 + V_6}{6}$$

El promedio suele tomarse como el valor más representativo de una población. Esto trae muchos inconvenientes, sobre todo para la gente sin entrenamiento en estadística ya que tiende a operar como si todos los casos fueran el promedio y la realidad puede ser muy distinta de eso.

Ejercicio 3.3

- Calcule el promedio de la tirada de los dados del ejercicio anterior para las primeras 10 tiradas
- Calcule el promedio de la tirada de los dados del ejercicio anterior para las primeras 20 tiradas
- Calcule el promedio de la tirada de los dados del ejercicio anterior para las primeras 30 tiradas
- Calcule el promedio de la tirada de los dados del ejercicio anterior para las primeras 40 tiradas
- Calcule el promedio de la tirada de los dados del ejercicio anterior para las primeras 80 tiradas (incorporando las de un compañero)
- Grafique los valores obtenidos

Deberá obtener un gráfico del tipo:



Ejercicio 3.4

- Calcule el talle de zapatos promedio en su familia (Incluya padres, hermanos, primos, amigos, etc hasta llegar a 10 personas)
- Consiga un zapato con el talle promedio
- Calce el zapato con el talle promedio a cada una de las 10 personas pidiéndole a cada una que indique el nivel de comodidad: 0 Imposible de usar a 10 muy cómodo.
- Calcule el nivel de comodidad promedio

El aprendizaje que nos deja el ejercicio es que las poblaciones NO SON EL PROMEDIO. Esto nos motiva para preguntarnos qué otras medidas pueden servirnos para describir con pocos números lo que le pasa a una población.

¿Cuán apartados están los datos del promedio?
La medida que trata de responder a esa pregunta es el desvío estándar que se calcula como:

$$\sigma = \sqrt{\frac{(V_1 - P)^2 + (V_2 - P)^2 + (V_3 - P)^2 + (V_4 - P)^2 + (V_5 - P)^2 + (V_6 - P)^2}{6}}$$

Ejercicio 3.5
Tenemos para considerar tres alumnos cuyas notas van en la tabla adjunta:

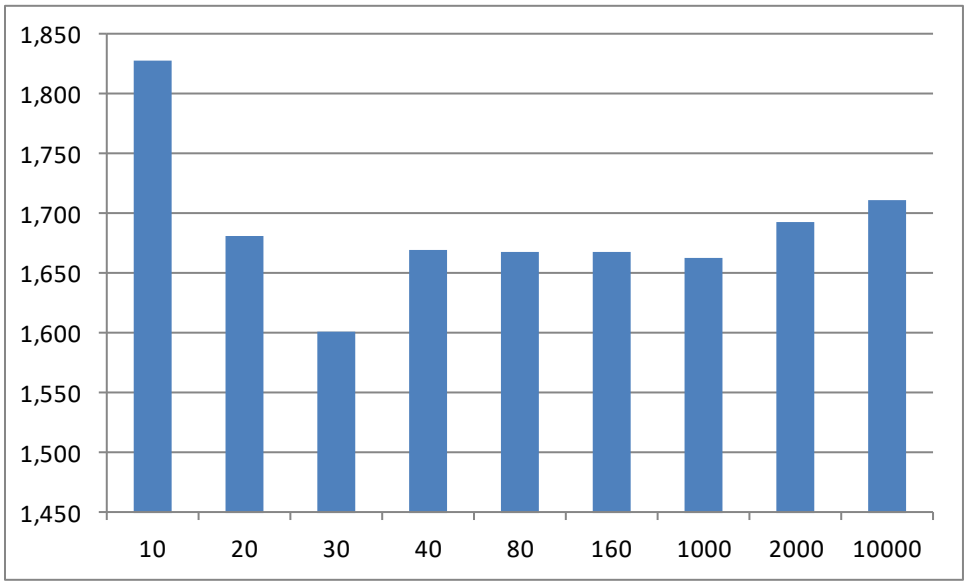
	Raul	Laura	Diego
Nota 1	**7**	**10**	**4**
Nota 2	**6**	**9**	**10**
Nota 3	**8**	**10**	**5**
Nota 4	**7**	**8**	**9**
Nota 5	**6**	**10**	**7**
Promedio	6,8	9,4	7
Desvío estándar	0,84	0,89	2,55

Laura es una alumna sobresaliente, que consistentemente saca muy buenas notas. Esto se refleja en su promedio que es alto y también en el pequeño desvío estándar.

Raúl es un alumno mediocre. Trata sólo de aprobar justo. Eso se refleja en su bajo promedio. Como sus notas también son siempre parecidas su desvío estándar es también pequeño.

Diego, en cambio, tiene un comportamiento muy variable. Esto no se refleja en su promedio que es similar al de Raúl pero sí en el desvío estándar que resulta mucho mayor.

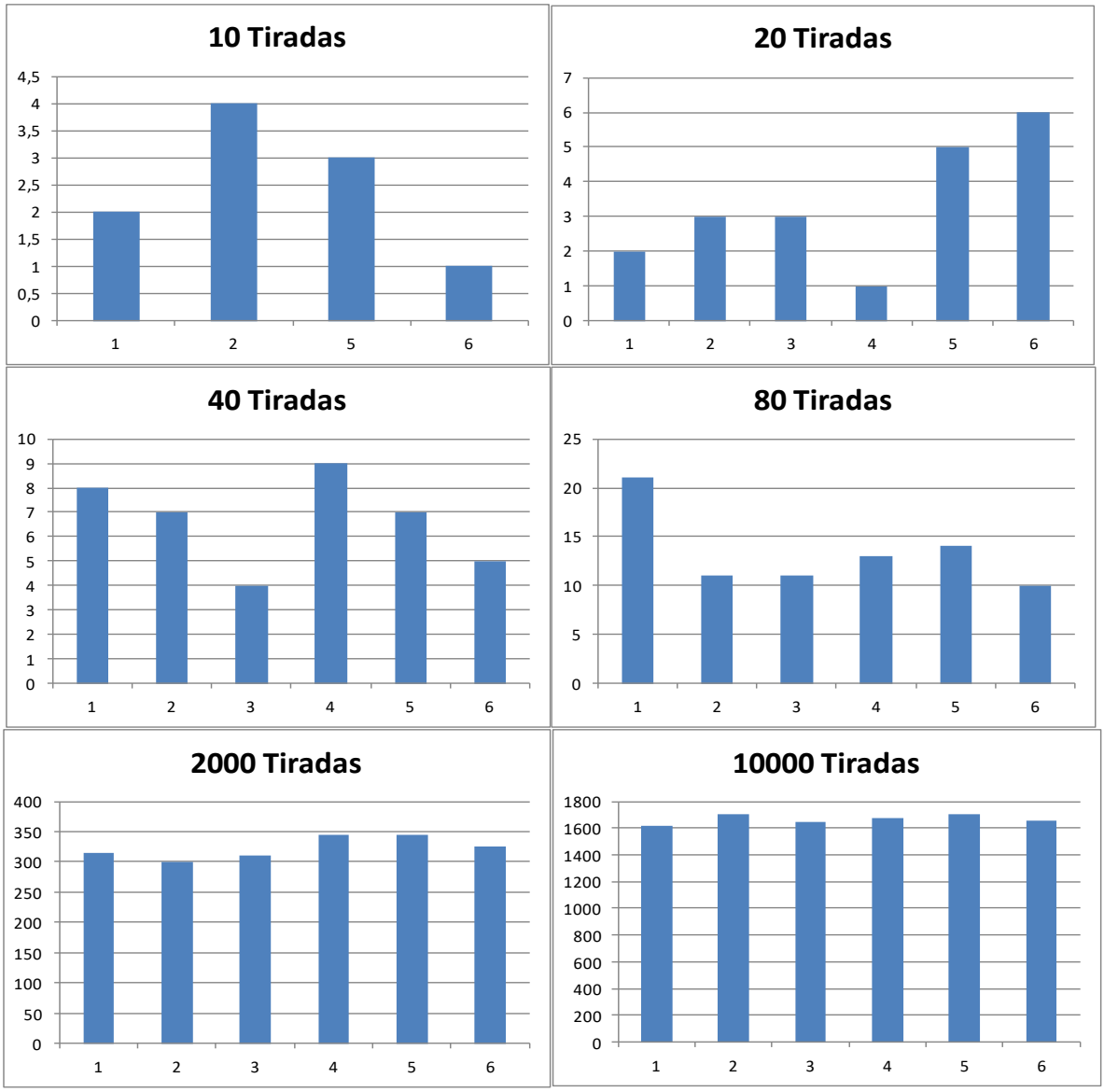
Ejercicio 3.6
Calcule los desvíos estándar para 10, 20, 30, 40 y 80 casos con los datos de las tiradas de dados y luego graficarlos.
Debería obtenerse algo similar a:



Histograma:
Una forma de darnos cuenta de donde caen los datos es realizar un recuento de frecuencias. Esto es, volviendo al caso de los dados, fijarse cuantas veces salió 1, cuantas salió 2, cuantas salió 3, cuantas salió 4, cuantas salió 5 y cuantas salió 6.

Luego se grafican esos 6 números.

Ejercicio 3.7
Realizar un recuento de frecuencias para las primeras 10, 20, 40 y 80 tiradas. Luego graficar los recuentos obtenidos.
Debería dar algo parecido a:



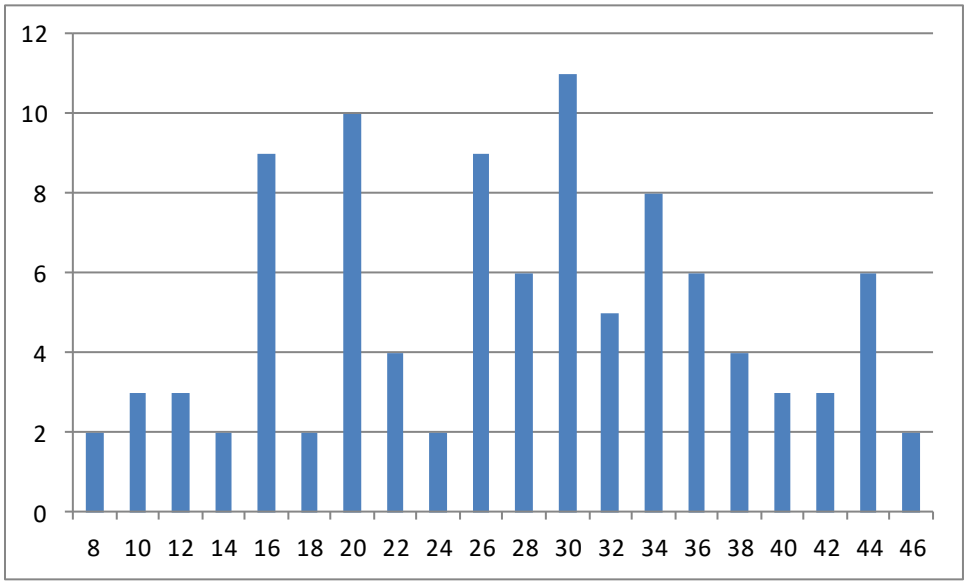
A medida que vamos considerando conjuntos de tiradas cada vez mayores la cantidad de veces que sale cada número es cada vez más pareja.

¿Qué ocurre si la variable que tratamos de medir no es discreta como el caso de los dados sino continua?

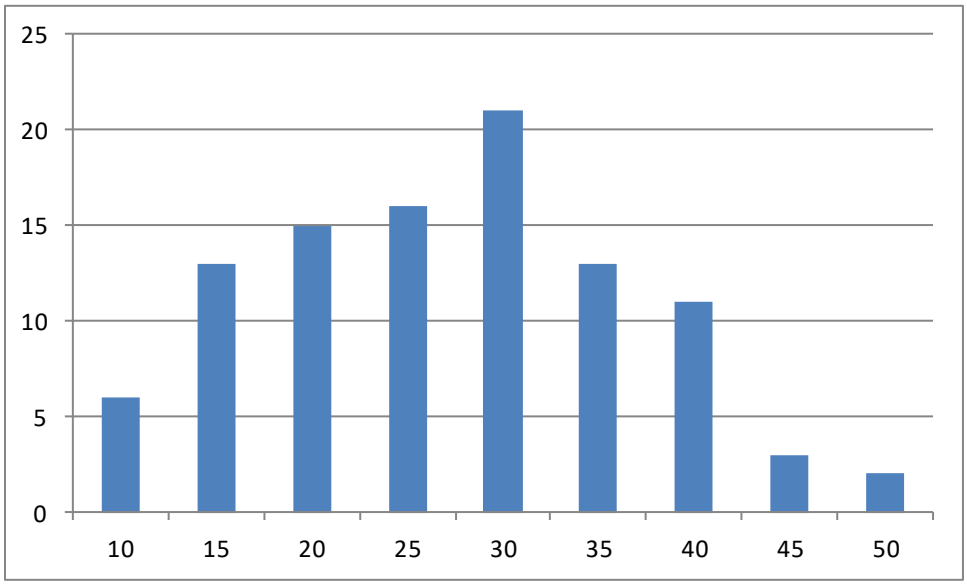
Recuerdo que, hace mucho tiempo, cuando el mundo era joven, hice mi primer trabajo práctico en la carrera de Física que consistía en medir las velocidades de los autos que pasaban frente a la Facultad.

Para eso habíamos medido con una cinta métrica un determinado recorrido en la avenida y luego, con cronómetros el tiempo que tardaban en pasar por puntos de control.
En principio, para nosotros, la velocidad que podíamos obtener podría darnos cualquier valor. Por ese motivo, para mostrar como un todo las velocidades que medíamos contamos cuantas iban entre 8 y 10 Km/h, cuantas entre 10 y 12 Km/h y así hasta cubrir el rango máximo.

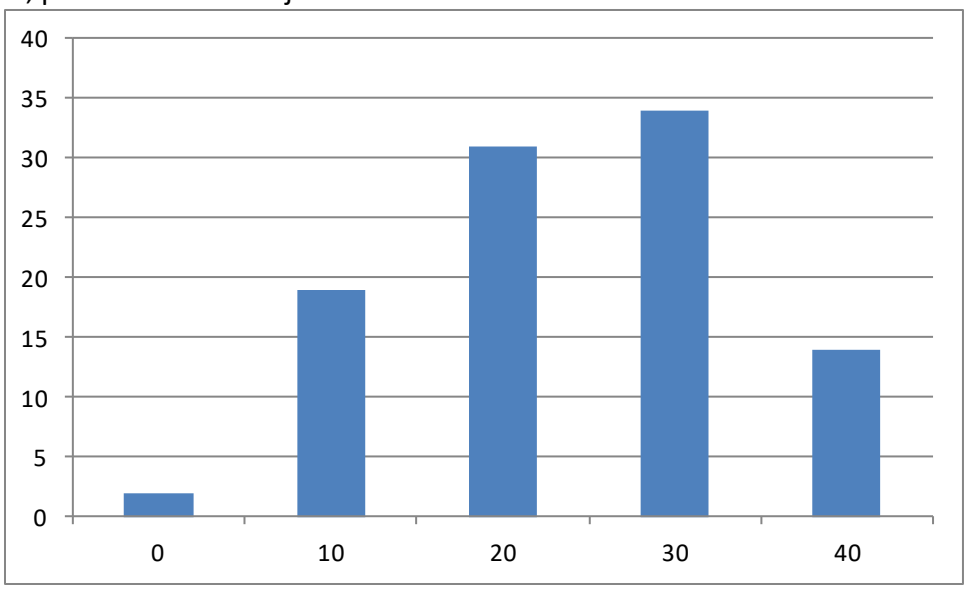
Al graficarlo obtuvimos algo como:



No nos resultaba muy ilustrativo de lo que estaba pasando por lo que se nos ocurrió cambiar los rangos a: 10 a 15 Km/h, 15 a 20 Km/h, etc. y obtuvimos:

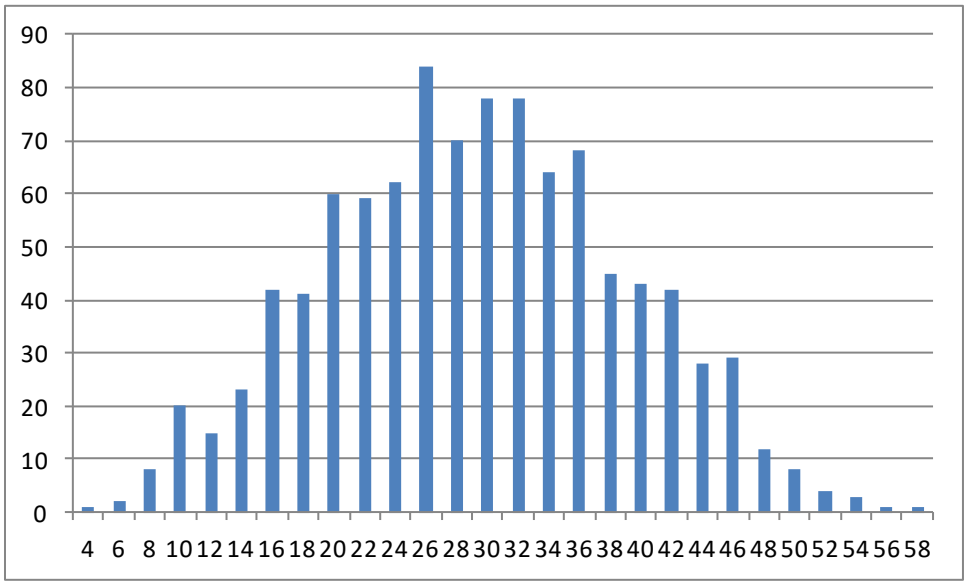


Ahora se empezaba a parecer un poco más a una curva suave. Pensamos si pasar de 2 a 5 fué bueno, pasar a 10 será mejor:



Con esto casi desaparecía toda la curva

Consternados fuimos a hablar con los docentes para entender que habíamos hecho mal. Nos consolaron diciendo que no había nada mal y nos sugirieron que compartiéramos los datos entre todos los grupos (y nos pasaron datos de grupos de años anteriores)



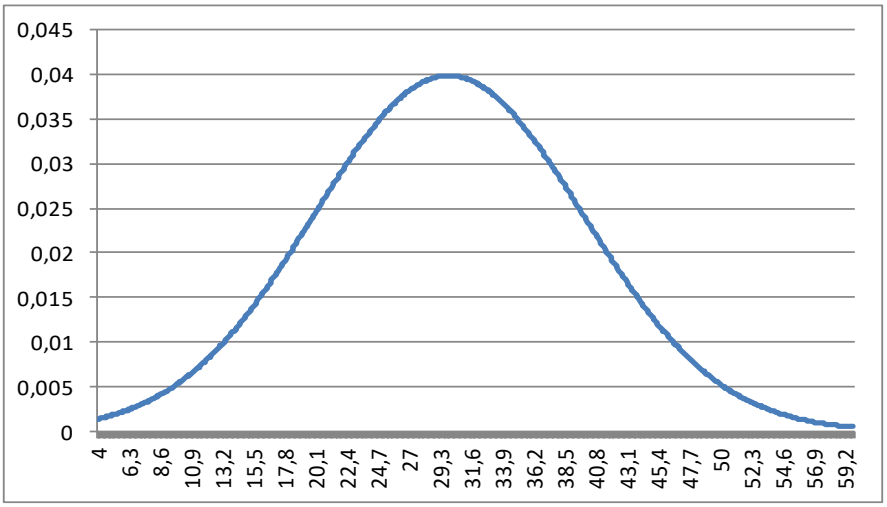
Ni que decir que obtener esta curva nos llenó de esperanza. Había una regularidad hasta en el tránsito!

Aprendimos dos lecciones:

- Cuando agrandamos el tamaño de la caja (de 2 a 5, por ejemplo) la curva tiende a definirse mejor, sin altibajos, pero, si exageramos, nos perdemos detalles (al pasar de 5 a 10, por ejemplo)
- Si agrandamos la cantidad de datos podemos ver más detalles con menos altibajos pero necesitamos trabajar más. (cuando compartimos los datos con otros grupos)

¿Qué pasaría si, por ejemplo, seguimos aumentando la cantidad de datos y, al mismo tiempo, achicando las cajas para saber lo más posible sobre la distribución de los datos?

La curva en cuestión iría pareciéndose a:



El nombre que se da a la curva a la cual tiende la distribución de los datos es "Función Distribución"

Hay diferentes funciones de distribución que se usan para modelar diferentes problemas:

- Binomial
- Poissón
- Normal
- LogNormal
- Beta
- Gamma
- Hipergeométrica
- T
- Chi-cuadrado

Existen muchas más y poco valor agregaría el agrandar la lista.

Ahora podemos pensar al promedio y al desvío estándar como formas de resumir la información contenida en toda la función distribución en sólo un par de números.
Existen otras medidas como:

- Mediana:
  Divide los datos dejando la misma cantidad de valores por debajo de su valor que por encima de su valor. Si la distribución es simétrica, como el caso de una distribución normal, la mediana coincide con el promedio.
- Cuartiles
  El primer cuartil deja el 25% de los datos por debajo del valor y el 75% por encima. El

segundo cuartil coincide con la mediana. El tercer cuartil deja el 75% de los valores por debajo y sólo el 25% por encima.

- Deciles
  Análogamente a los cuartiles, el primer decil deja el 10% de los datos por debajo de su valor y el 90% por encima. Así va avanzando hasta que el noveno decil deja el 90% de los datos por debajo y sólo el 10% por encima.
- Percentiles
  Son conceptualmente equivalentes a los deciles pero, por ejemplo, el primer percentil deja sólo el 1% de los datos por debajo y el 99% por encima.

Normalmente no sabemos cuál es la función distribución que van a seguir los resultados de una determinada situación. Afortunadamente existe un resultado que ayuda muchísimo: Si una distribución es simétrica entonces el promedio de 30 o más variables aleatorias de esta distribución tiene una distribución normal.

Este resultado es conocido como teorema del límite central y nos permite asumir que muchas variables aleatorias tienen distribuciones normales.

Inferencia Estadística

La estadística descriptiva trataba de resumir en unas pocas medidas la información del universo de valores posibles.

La inferencia estadística se propone algo mucho más arduo y desafiante: saber algo del universo a partir de una muestra de valores.

En el contexto de la inteligencia de negocios nos damos rápidamente cuenta que la inferencia estadística estará debajo de las herramientas que vayamos a usar por lo cual nos conviene irnos con máximo cuidado con este tema.

El problema de la moneda cargada
Tenemos que decidir si una moneda está cargada. Todo lo que podemos hacer es lanzarla, anotar los resultados y PENSAR!

La verdad es que sin haberla lanzado no podemos decir nada sobre ella. Entonces la lanzamos. Supongamos que sale cara. ¿Nos autoriza eso a pensar que está cargada?

Una moneda equilibrada (lo opuesto de cargada) tiene las mismas posibilidades de salir cara que seca. Que la primera vez haya salido cara no significa nada.

Entonces, para ganar conocimiento, la lanzamos una segunda vez. Supongamos que, de nuevo, sale cara. ¿Podemos asumir que está cargada?

La respuesta es que difícilmente podamos sacar esa conclusión. Una moneda equilibrada tiene un 25% de probabilidades de caer cara dos veces seguidas.

¿Ya se imaginan como sigue esto? Por supuesto, la arrojamos por tercera vez. Supongamos que vuelve a salir cara. ¿Asumimos ahora que está cargada? Esto dependerá de los límites que asignemos a nuestra credibilidad. Si no estamos dispuestos a creer que se trata de una casualidad ningún evento menos probable, por ejemplo, que el 10% entonces todavía estaremos aceptando que puede estar equilibrada.

Obviamente, si la seguimos lanzando, en algún momento cruzaremos la barrera del 10% y nos veremos obligados a rechazar creer en que la moneda esté equilibrada.
Este es el proceso que se conoce como "Test de Hipótesis"

Repasemos sus elementos:

- Hipótesis nula
  Es lo que nos inclinaríamos a creer ante la falta de información.
  En el caso que nos ocupa es que la moneda está equilibrada.
- Hipótesis alternativa
  Es lo que necesita demostración.
  Para la moneda es que esté cargada.
- Nivel de confianza
  Es el límite de credibilidad que nos trazamos. Es 1 - probabilidad de aceptar la hipótesis nula aunque debiera rechazarla.

Con esto podemos cometer dos tipos de errores:

- No rechazar la hipótesis nula y que sea falsa
  En el caso de la moneda sería creerse que está equilibrada y que esté cargada.
- Rechazar la hipótesis nula y que sea verdadera
  Sería afirmar que está cargada y que, en realidad, esté equilibrada.

Dependiendo del tipo de problema que nos toque ambos errores pueden tener consecuencias muy distintas. Pensando en las consecuencias de cada tipo de error y en los costos de seguir experimentando es que se establecen los niveles de confianza.

Vamos a dar algunos ejemplos muy simples para que podamos practicar el uso que se da de estos conceptos:

Vamos a asumir que:

- El 15% de los seres humanos tienen sobrepeso.
- Tomamos una muestra aleatoria de 5 habitantes de la ciudad de Buenos Aires y encontramos que 2 de ellos tienen sobrepeso.

¿Podremos asumir que los habitantes de la ciudad de Buenos Aires tienen la misma tasa de sobrepeso que el resto de la humanidad?

Empezamos por establecer los elementos del test:

- Hipótesis nula: Si, tienen la misma tasa de sobrepeso.
- Hipótesis alternativa: No, no tienen la misma tasa de sobrepeso.
- Nivel de confianza: 90%

Lo primero que necesitamos calcular es la probabilidad de que ninguno de los 5 tenga sobrepeso.

Para ello decimos que al elegir el primero tendría un 85% de probabilidades de no tener sobrepeso.

Si se nos diera este caso, que tampoco el segundo tenga sobrepeso sería el 85% del 85% lo que podemos aproximar por un 72.25%

¿Y que el tercero tampoco tuviera sobrepeso? Pues el 85% del 85% del 85%, o sea, el 61.42%

Siguiendo el mismo razonamiento tendríamos:
4 sin sobrepeso 52.20%
5 sin sobrepeso 44.38%

Ahora queremos ver que probabilidades de que aparezca uno con sobrepeso entre cuatro sin. Con lo que venimos haciendo sería:

P(1ro con sobrepeso, 4 sin sobrepeso) = 15% x 85% x 85% x 85% x 85% = 7.83%

Ahora bien, para lo que buscamos que sea el primero el que tiene sobrepeso, el segundo, el tercero, el cuarto o el quinto no nos preocupa. Son cinco formas de alcanzar 4 sin sobrepeso y 1 con sobrepeso. Además, la probabilidad de cualquiera de esos casos es la misma por lo que:

P(1 con sobrepeso, 4 sin sobrepeso) = 5 x P(1ro con sobrepeso, otros 4 sin sobrepeso) = 5 x 7.83% = 39.15%

Siguiendo por este camino podemos tratar de calcular la probabilidad de que dos tengan sobrepeso y sólo tres no lo tengan. Para evitar que las cuentas nos abrumen vamos a recurrir reconocer que estamos trabajando con una variable aleatoria que sólo tiene dos salidas posibles:

- Con sobrepeso
- Sin sobrepeso

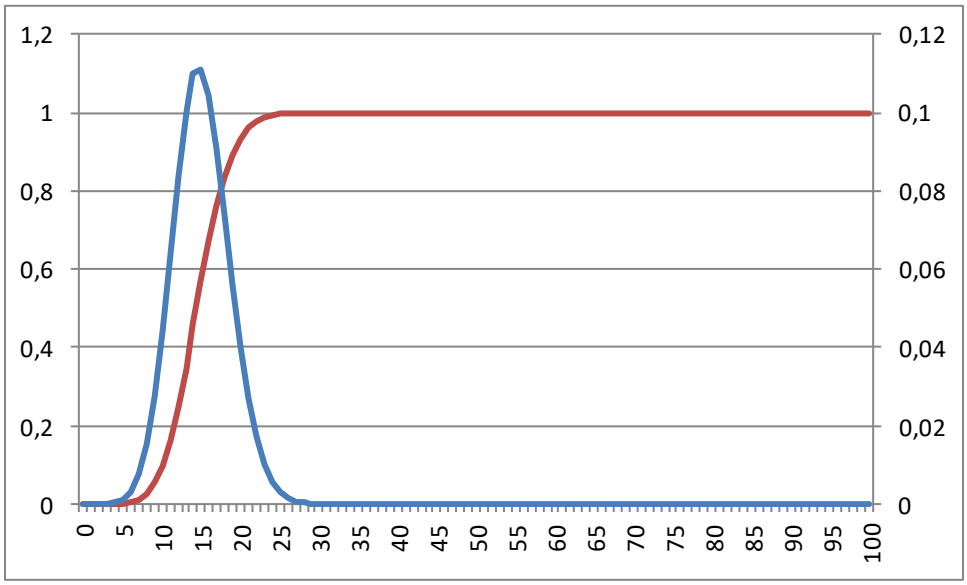
En estos casos resulta perfectamente posible aplicar la distribución binomial. En el que nos ocupa:

- P(2 con sobrepeso, 3 sin sobrepeso) = distr.binom(2;5;0.15;falso) = 13.82%
- P(3 con sobrepeso, 2 sin sobrepeso) = distr.binom(3;5;0.15;falso) = 2.44%
- P(4 con sobrepeso, 1 sin sobrepeso) = distr.binom(4;5;0.15;falso) = 0.22%
- P(5 con sobrepeso, 0 sin sobrepeso) = distr.binom(5;5;0.15;falso) = 0.007%

Hay dos formas en que los porteños podríamos diferenciarnos del resto de la humanidad, con una tasa de sobrepeso mayor o menor. Nosotros queremos asegurarnos que la muestra que obtuvimos esté dentro del 90% más probable, o sea, ni dentro del 5% menos probable por demasiado delgados ni dentro del 5% menos probable por demasiado gordos.

Ahora miramos la situación. Si los porteños tuviéramos la misma tasa de sobrepeso que el resto de la humanidad la probabilidad de encontrarnos con 2 en 5 sería del 13.82% esto no se encuentra dentro del 5% superior ni del 5% inferior por lo que no tendríamos argumentos para rechazar la hipótesis nula.

En cambio si la cantidad de porteños con sobrepeso hubiera sido 3, 4 o 5 lo que hubiera implicado una probabilidad del 2.44%+0.22%+0.007% = 2.667% lo que si se hubiera encontrado por debajo del 5% y nos habríamos visto forzados a rechazar la hipótesis nula y asumir, muy a nuestro pesar, que los porteños tendemos a tener más personas con sobrepeso que el total de la humanidad.



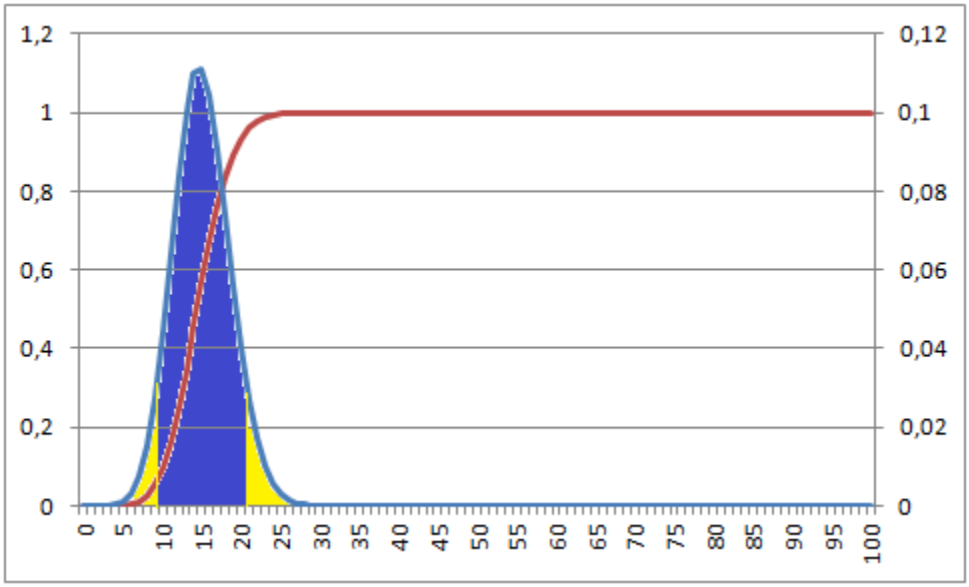
La línea azul muestra la probabilidad de encontrar exactamente tantas personas con sobrepeso en una muestra de 100 como se indica sobre el eje X. Esto lo asociamos a probabilidad puntual y podemos reproducirlo calculando distr.binom(x;100;,15;falso)

La línea roja muestra la probabilidad de encontrar hasta la cantidad de personas que se muestran sobre el eje X en una muestra de 100. Esto lo llamamos probabilidad acumulada y podemos reproducirlo calculando distr.binom(x;100;,15;falso)

El primer 5% de probabilidad lo acumulamos cuando la línea roja cruza el nivel del 5% (sobre el eje vertical izquierdo) lo que tiene lugar para X = 10 (aproximadamente)

El último 5% de probabilidad lo acumulamos recién cuando la línea roja cruza el 95% (sobre el eje vertical izquierdo) lo que recién ocurre para X = 21 (aproximadamente)

Entonces, con el planteo hecho, si hubiéramos tomado una muestra de 100 porteños al azar estaríamos en condiciones de rechazar la hipótesis nula si la muestra hubiera contenido 10 personas o menos con sobrepeso o bien 21 o más.

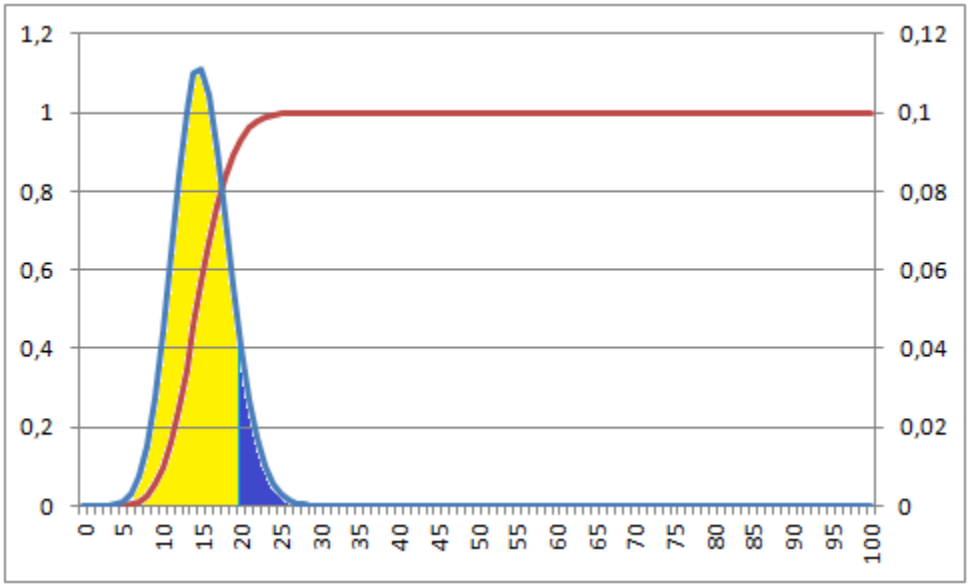


En amarillo tenemos las zonas de rechazo a ambos lados del máximo mientras que en azul tenemos las zonas de aceptación.

Vemos que al preguntar si la incidencia del sobrepeso de los porteños es o no la misma que la del resto de la humanidad dejamos parte de la zona de rechazo para el caso en que los porteños resulten demasiado delgados y parte para la posibilidad de que resulten demasiado gordos.

Es importante ser muy cuidadoso con el planteo. Si en vez de preguntarnos, en general, si la probabilidad de sobrepeso de los porteños es la misma o no de la población mundial nos hubiéramos preguntado si era, por ejemplo mayor el planteo hubiera sido ligeramente diferente.

Cualquier valor de la curva roja por debajo del 90% nos hubiera llevado a aceptar la hipótesis nula. En el caso que nos ocupa con 19 personas con sobrepeso o menos nos hubiéramos conformado y nos hubiera bastado con 20 o más para rechazar la hipótesis nula y abrazar la hipótesis alternativa.



Queda pintada de color azul la zona de rechazo de la hipótesis nula mientras que queda pintada de amarillo la zona de aceptación.

Vemos que al preguntar si los porteños tienen mayor incidencia de sobrepeso que el resto de la humanidad la zona de rechazo queda sólo de un lado.

Ejercicio 3.8
Un apostador empedernido lo contrata para saber si le están haciendo trampa con los dados.

Para eso le pasa una serie de 1.000 tiradas del dado en cuestión:

Usted contabiliza las frecuencias y descubre que:

Cara	Veces que salió
1	170
2	235
3	163
4	105
5	168
6	159

Como el apostador es muy cauto le pide que le responda con un nivel de confianza del 5% ¿Qué le puede decir?

Análisis descriptivo y exploratorio de datos

Análisis exploratorio

Una vez que se tiene un conjunto de datos ya limpio (por ejemplo habiendo decidido qué hacer con los 'Not Available' o NAs), lo primero que se debe hacer es realizar un análisis exploratorio.

Este consiste en estudiar la estructura general de los datos, cuántas variables tiene el set de datos, sus dimensiones, sus clases, para luego empezar a explorar las relaciones entre esas variables (cuál depende de cuál, cómo dependen, etc.).

El análisis que haremos dependerá directamente de qué querramos hacer con los datos, es decir, si nuestro propósito es generar una visualización, o desarrollar un modelo predictivo, o simplemente presentar un informe descriptivo que resuma los datos.

Al explorar los datos, especialmente cuando tenemos muchos datos y estos son complejos, siempre debemos acordarnos de nuestro propósito, de qué buscamos contar con los datos, porque esto nos va a ayudar a definir en qué enfocarnos y llevar adelante el proceso en un tiempo razonable.

En todo el proceso de exploración (e incluso también cuando hacemos minería de datos y modelos predictivos), veremos que generar gráficos resulta imprescindible para ver qué estamos haciendo y poder detectar relaciones y patrones que de otra forma sería imposible encontrar en la maraña de datos.

Por ejemplo si utilizamos el set de datos *iris* incluido en R, lo primero que podemos hacer es ver la estructura que este objeto tiene en R. Para ello utilizamos el comando **str()**.

Veamoslo: primero cargamos el set iris:

```
library(datasets)
data(iris)

str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species: Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1
1 1 1 ...
```

Esta función muestra la estructura interna del objeto. En particular, este conjunto de datos corresponde a mediciones en centímetros de las variables longitud y ancho del sépalo y longitud y ancho de pétalo, para 50 flores de 3 especies de iris. Las especies son Iris setosa, versicolor, y virginica.

Ahora bien este conjunto de datos tiene una clase determinada, para poder saber cual es usamos el comando **class()**

```
class(iris)
```

```
## [1] "data.frame"
```

Como era de esperar nuestro conjunto de datos es un *dataframe*, pero veamos cuantas filas (mediciones) tiene nuestro dataframe. Para ello utilizamos el comando **nrow()**

```
nrow(iris)
```

```
## [1] 150
```

Análogamente para saber la cantidad de columnas usamos **ncol()**

```
ncol(iris)
```

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

Siguiendo con la exploración de este dataset, veamos cuáles son los nombres de las variables.

```
names(iris)
```

```
##  [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"
## [5] "Species"
```

También es importante ver las clases de cada variable, ya que podemos encontrar las variables categóricas que serán útiles en nuestro análisis. Por ejemplo, podemos ver qué clase tiene la variable 'Sepal.Length':

```
class(iris[,"Sepal.Length"])

## [1] "numeric"
```

Cuando se trabaja con variables numéricas queremos ver que rango de valores abarca las mismas, y para ello usamos la función **range()**, mientras que con **length()** podemos saber cuántos datos hay:

```
range(iris[,"Sepal.Length"])

## [1] 4.3  7.9

que nos dice que la variable “Sepal.Length” del set iris toma valores
comprendidos entre 4.3 y 7.9, y

length(iris[,"Sepal.Length"])

## [1] 150
```

que nos dice que tenemos 150 datos.
Como habíamos mencionado, el conjunto *iris* consta de datos de 3 especies de flores. Podemos corroborar eso con los comandos **nlevels()** (nos dice el número de valores posibles de cada variable) y **levels()** (nos da la lista de estos valores posibles de cada variable). A los valores posibles de una variable categórica los llamamos niveles, y de ahí viene el nombre de estos dos comandos. Por ejemplo,

```
nlevels(iris[,"Species"])

## [1]      3
```

que nos dice que la variable (categórica) tiene tres posibles valores, y

```
levels(iris[,"Species"])

## [1] "setosa"     "versicolor"   "virginica"
```

que nos dice cuáles son estos niveles de la variable 'Species' del set de datos *iris*.

Pasemos ahora a una función muy útil a la hora de explorar datos (y también hacer reportes): la función **table**. Para el ejemplo que estamos viendo, la información sobre la variable 'Species' puede ser expresada mediante una tabla:

```
table(iris[,"Species"])

##
##     setosa     versicolor        virginica
##         50             50               50
```

que nos dice que hay 50 datos (observaciones) para cada uno de los niveles de la variable 'Species'. Siguiendo con las tablas, supongamos que queremos saber cuántas veces aparece

cada valor posible de una variable númerica. Pongamos como ejemplo la variable 'Sepal.Length' del set *iris*.

Recordemos que para acceder a estos valores, usamos el operador **$**. En este caso, los valores de la variable 'Sepal.Length' del set *iris* se acceden mediante '**iris$Sepal.Length**'. Tenemos entonces:

```
table(iris$Sepal.Length)

##
## 4.3 4.4 4.5 4.6 4.7 4.8 4.9  5   5.1 5.2  5.3 5.4  5.5  5.6  5.7   5.8  5.9   6
## 1   3   1   4   2   5   6    10  9   4    1   6    7    6    8     7    3     6
## 6.1 6.2 6.3 6.4 6.5 6.6 6.7  6.8 6.9 7    7.1 7.2  7.3  7.4  7.6  7.7   7.9   6
## 6   4   9   7   5   2   8    3   4   1    1   3    1    1    1    4     1     7
```

que nos dice que el valor 4.3 aparece 1 vez, el valor 4.4 aparece 3 veces, y así siguiendo.

Este ejemplo de table es interesante porque de paso repasamos algo esencial en R, que aparece muchas veces. Notemos que hemos usado el comando **table()** pasándole como argumento tanto una variable categórica como una variable numérica, y *nunca tuvimos que indicarle a R de qué tipo de variable se trataba*.

Es decir, la mayoría de los comandos de R detectan automáticamente el tipo de los datos de entrada, y operan en consecuencia.

Otros dos comandos útiles para explorar el contenido de objetos son el **head** y **tail**, que nos dan la primera o la última parte del objeto, respectivamente:

```
head(iris)

##        Sepal.Length   Sepal.Width   Petal.Length   Petal.Width   Species
## 1            5.1           3.5                1.4                  0.2
setosa
## 2            4.9           3.0                1.4                  0.2
setosa
## 3            4.7           3.2                1.3                  0.2
setosa
## 4            4.6           3.1                1.5                  0.2
setosa
## 5            5.0           3.6                1.4                  0.2
setosa
## 6            5.4           3.9                1.7                  0.4
setosa

tail(iris)

##        Sepal.Length  Sepal.Width  Petal.Length  Petal.Width     Species
## 145           6.7          3.3                5.7                  2.5
```


```
virginica
## 146          6.7         3.0          5.2          2.3
       virginica
## 147          6.3         2.5          5.0          1.9
       virginica
## 148          6.5         3.0          5.2          2.0
       virginica
## 149          6.2         3.4          5.4          2.3
       virginica
## 150          5.9         3.0          5.1          1.8
       virginica
```

Pasaremos ahora a dar una introducción al análisis exploratorio de la estadística de los datos, tan importante para comprender la naturaleza de los datos y cómo se relacionan las variables. Luego reforzaremos la discusión con un ejemplo práctico de regresión lineal.

Análisis Estadístico

Este tipo de análisis emplea técnicas estadísticas para interpretar datos y describir sus características. Por ejemplo, dada una población, calcular su edad media, distribución de frecuencias de dicha edad, y desviación típica. Se pretende facilitar un retrato de dicha población que permita definirla con precisión.

Los análisis realizados con esta finalidad suelen ser catalogados como muy sencillos y de escasa dificultad. Esto no debe hacer que los despreciemos, dado que, con un esfuerzo moderado, facilitan información muy útil e imprescindible para la comprensión de la realidad de dicha población. Además, nos permiten entender mejor los datos y cómo se relacionan las variables, cosas que son imprescindibles para definir bien qué modelos y visualizaciones serán los más apropiados para analizar y mostrar estos datos.

Debe tenerse en cuenta un detalle importante: los análisis descriptivos restringen sus conclusiones a la población analizada. Es decir, si se estudia un grupo de 30 personas, los valores de edad media, peso, etc. serán válidos únicamente para esas 30 personas. Si se desea extrapolar la información recogida a un grupo mayor, será necesario utilizar la estadística inferencial.

Ahora realicemos un análisis estadístico sobre el conjunto de datos *iris* para la variable *Sepal.Length*. Calculemos su mediana, valor medio y desviación estándar:

```
median(iris[,"Sepal.Length"])

## [1] 5.8

mean(iris[,"Sepal.Length"])

## [1] 5.843333
```


```
sd(iris$Sepal.Length)
```

```
## [1] 0.8280661
```

Si quisieramos hacer un análisis más rápido podemos usar la función **summary**:

```
summary(iris)
```

```
##     Sepal.Length       Sepal.Width        Petal.Length       Petal.Width
##   Min.    :4.300    Min.    :2.000    Min.    :1.000    Min.    :0.100
##   1st Qu.:5.100    1st Qu.:2.800    1st Qu.:1.600    1st Qu.:0.300
##   Median :5.800    Median :3.000    Median :4.350    Median :1.300
##   Mean    :5.843    Mean    :3.057    Mean    :3.758    Mean    :1.199
##   3rd Qu.:6.400    3rd Qu.:3.300    3rd Qu.:5.100    3rd Qu.:1.800
##   Max.    :7.900    Max.    :4.400    Max.    :6.900    Max.    :2.500
##                                                           Species
##                     setosa                                   :50
##                                                       versicolor:50
##                                          virginica                :50
##
##
##
```

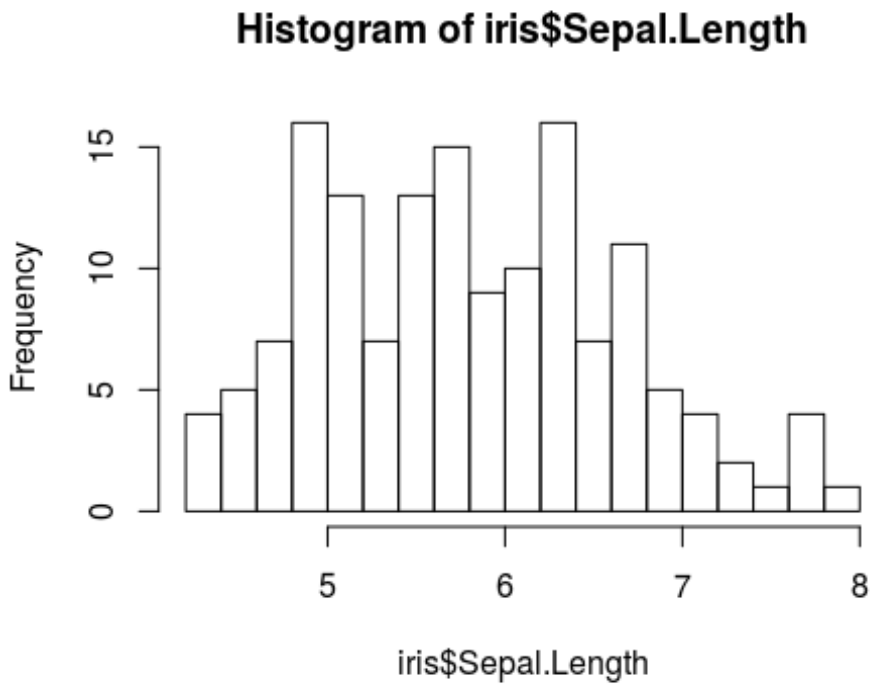
Este es un comando muy conveniente, porque no sólo nos da la mediana y valor medio sino además los cuartiles, el mínimo y máximo, y no sólo de una variable sino de todas las que aparecen en el set de datos. Notemos que también nos da los niveles de las variables categóricas (como 'Species'), y el número de observaciones de cada variable categórica (50 observaciones de 'Species' igual a setosa, 50 de 'Species' igual a versicolor, y 50 de 'Species' igual a virginica).

Podemos comenzar a realizar un análisis gráfico de los datos, que siempre resulta de suma utilidad. Los gráficos que más usaremos son el histograma, el llamado QQ-Plot para ver si los datos se distribuyen según una distribución gaussiana (también llamada normal), el diagrama de cajas, y los gráficos de dispersión (scatter plot) que nos ayudan a encontrar relaciones entre los datos.

Es importante remarcar que realizaremos una descripción muy detallada de los gráficos que se pueden hacer con R, y cómo hacerlos, en la unidad que sigue. Por ahora, tomemos estos gráficos como introducción y práctica para lo que viene, y como una manera rápida de entender los datos en un análisis exploratorio, dejando los detalles de los gráficos para la siguiente unidad.

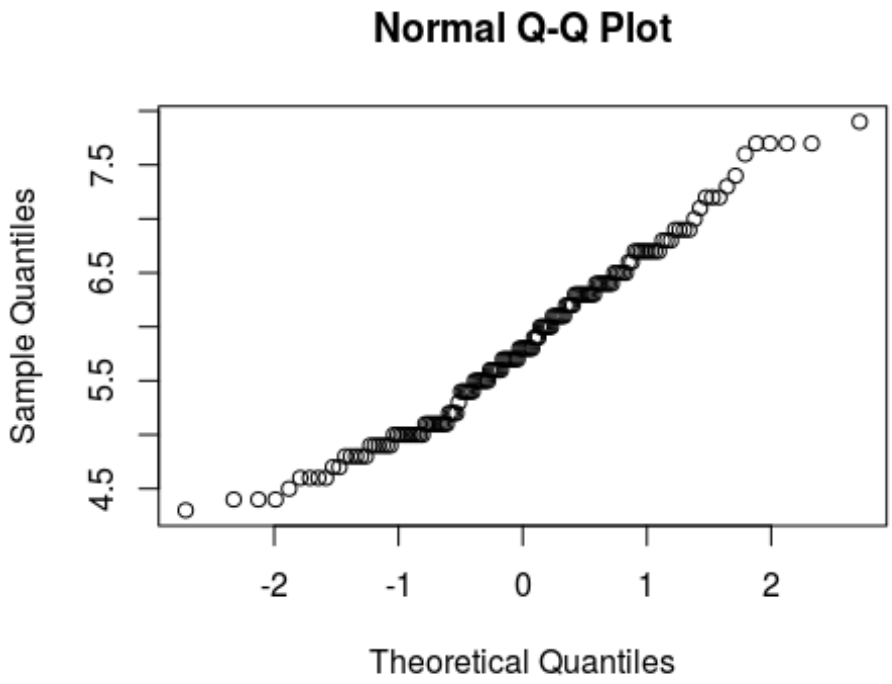
Veamos cómo se distribuyen los valores de la variable 'Sepal.Length' del set *iris* (como siempre, accedemos a través de '**iris$Sepal.Length**'). Para ello, usaremos el comando **hist** que realiza un histograma de los datos:

```
hist(iris$Sepal.Length,breaks=20)
```



Con la opción '*breaks*' indicamos el número de *bins* o subdivisiones de la variable sobre la cual queremos hacer el histograma, y en este caso usamos 20. Prueben de cambiar este valor y ver qué pasa.
Veamos si esta variable, 'Sepal.Length', se distribuye según una distribución normal. Del histograma parecería que se acerca bastante. Para ello, es útil graficar el llamado QQ-Plot. Mientras más puntos de la curva QQ-Plot caigan sobre una línea recta, más se aproxima la distribución de los datos a una normal.

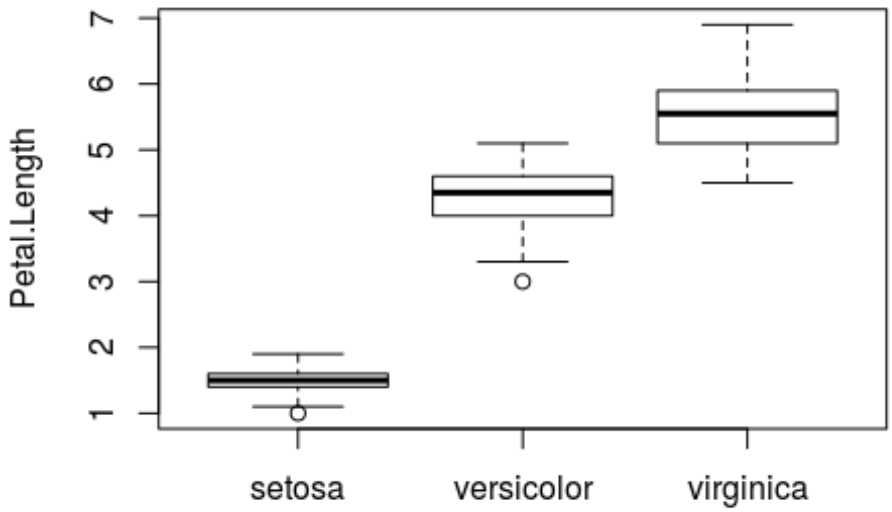
```
qqnorm(iris$Sepal.Length)
```



Vemos que 'Sepal.Length' se distribuye casi como una normal. Saber esto es útil si fuera necesario profundizar el análisis estadístico, ya que sabríamos qué tests estadísticos podemos usar y cuáles no.

Continuando con el estudio de las propiedades estadísticas de un conjunto de datos, veremos ahora cómo construir un diagrama de cajas en R. Este diagrama es sumamente útil porque nos permite ver 'la forma' de la distribución de los datos, el número de casos extremos, visualizar los cuartiles, el valor medio y la mediana. El comando apropiado es el **boxplot()**:

```
boxplot(Petal.Length ~ Species, data =iris, ylab = "Petal.Length",
varwidth = TRUE)
```

Con esta sintaxis le indicamos que queremos el diagrama de cajas de la variable 'Petal.Length' para los tres tipos de especies de flores, dadas por la variable 'Species'.

Notemos el uso del operador '~', que volveremos a encontrar varias veces. La opción 'data = iris' indica que las variables deben sacarse de ese set de datos, mientras que con la opción 'ylab' definimos la etiqueta del eje y. Por último, con la opción 'varwidth' le pedimos que nos muestre la varianza de los datos.

Este tipo de gráfico sirve, además de lo indicado más arriba, para determinar rápidamente si se puede decir si existe diferencia estadística en el valor de una variable entre distintas poblaciones.

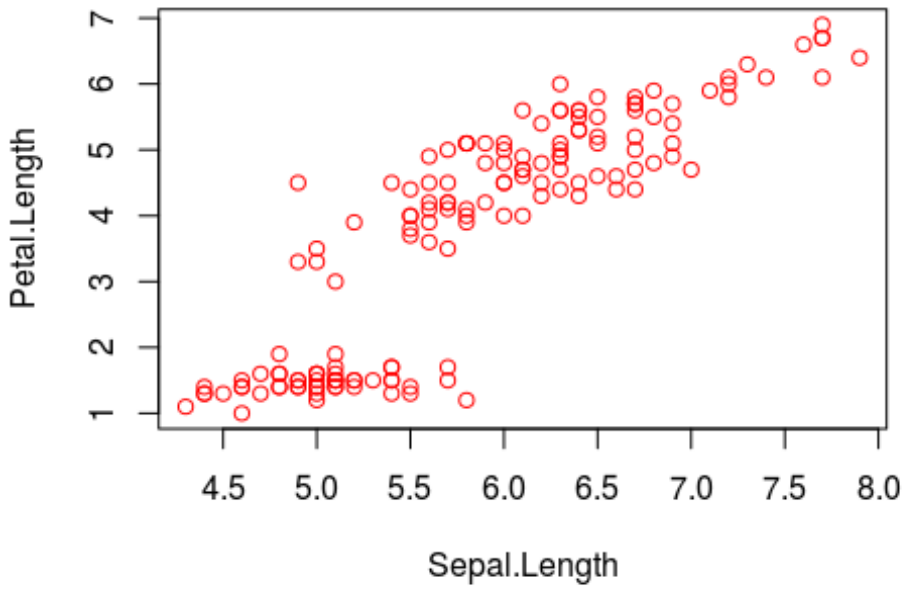
En el ejemplo que estamos viendo, como las 'cajas' correspondientes a las tres especies de flores no se solapan entre sí, podemos asegurar que existen diferencias (estadísticamente) significativas en el valor medio de 'Petal.Length' entre las tres especies.

Esto es muy importante, pues implica, por ejemplo, que la variable 'Petal.Length' es una buena candidata para usar si quisiéramos agrupar las flores por especie a través de una técnica de minería de datos conocida como agrupamiento o *clustering*.

Veremos un ejemplo de esto al final de esta unidad, y seguiremos discutiendo técnicas de minería de datos en las próximas unidades. Esta información también sería crucial si quisiéramos desarrollar un modelo predictivo que, dadas las características de una flor, nos dijera a qué especie pertenece. Con el box-plot, ya sabríamos que 'Petal.Length' es una muy buena variable predictora (es decir, conociendo el valor de 'Petal.Length', podemos *casi casi* saber a qué especie corresponde la flor).

Si queremos ver cómo se relacionan dos variables, podemos usar el comando **plot** (este comando lo discutiremos con mucho detalle en la unidad que sigue).

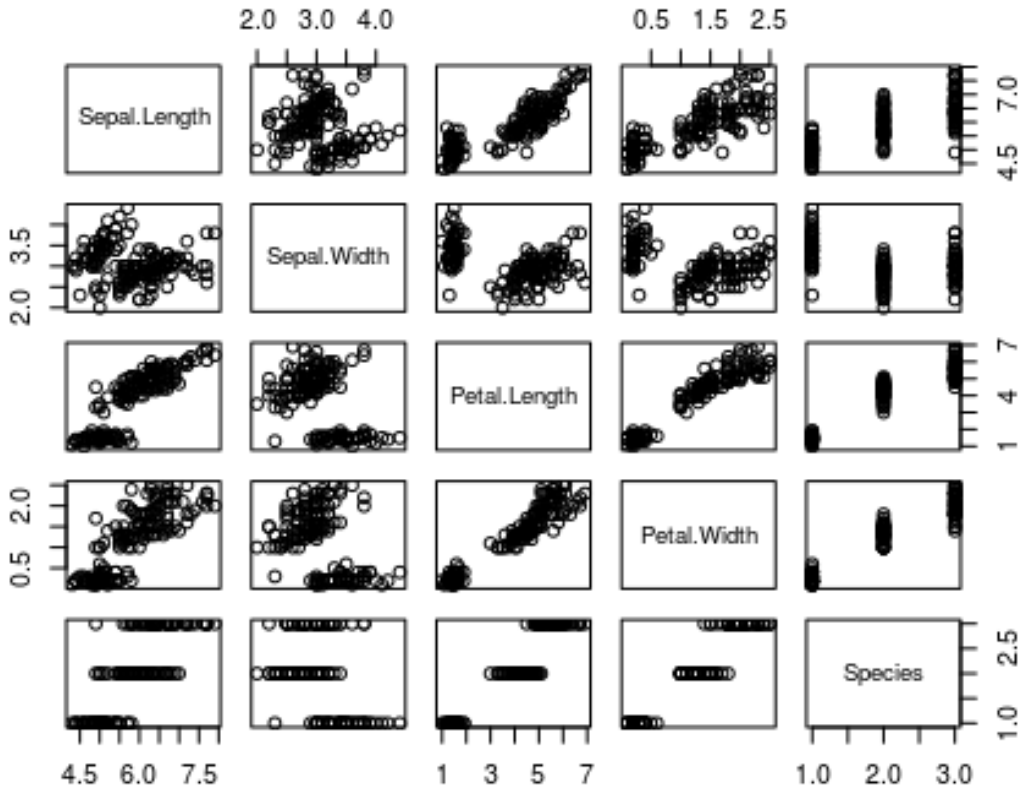
```
plot(Petal.Length ~ Sepal.Length, data = iris, col = 'red')
```



Con este comando le estamos diciendo a R que grafique la variable 'Petal.Length' en función de la variable 'Sepal.Length'. Con la opción 'data = iris', le estamos indicando que saque las variables del set *iris*. Con la opción 'col' le indicamos de qué color queremos los puntos graficados.

Podríamos hacer esto para todos los pares de variables del set *iris*, pero existe un comando realmente útil que hace exactamente esto: el comando **pairs:**

```
pairs(iris)
```



Notemos que el argumento de esta función es un set de datos (en este caso *iris)*), pero también podría ser un subconjunto del set de datos.

Vemos que **pairs** nos muestra gráficos de dispersión para todos los pares de variables del conjunto de datos de entrada. Esto nos permite ver rápidamente la relación entre las variables: de qué otras variables dependen -o no- y cómo dependen (linealmente, exponencialmente, etc).

Esto nos ayuda muchísimo para definir los modelos que se ajustan mejor a la naturaleza de los datos. Por ejemplo, si queremos predecir una variable y notamos que depende linealmente de otra, entonces un modelo de regresión lineal será una opción razonable.

Como se ve de la figura anterior, este es el caso para la dependencia de 'Petal.Length' con respecto a 'Petal.Width' por ejemplo.

Pasaremos ahora a una introducción a la regresión lineal, pero antes describiremos brevemente algunas de las funciones de distribución con las que cuenta R, y algunos tests estadísticos que podemos hacer que resultan útiles a la hora de explorar datos.

Distribuciones

En R cada distribución de probabilidades tiene cuatro funciones, las cuales tienen la misma raiz para el nombre pero con un prefijo que identifica la función que realizan.

- p para "probabilidad", esta función da la función de distribución acumulada (c. d. f.)
- q para "quantile", que es la inversa de lo anterior c. d. f.
- d para "density", función de densidad de probabilidades (p. f. or p. d. f.)
- r para "random": es para crear variables aleatorias con la distribución especificada

Ahora listemos algunas de las funciones de distribución más usadas, notando que en R hay muchísimas más!:

Nombre	Probabilidad	Percentil	Densidad	Random
Normal	pnorm	qnorm	dnorm	rnorm
Binomial	pbinom	qbinom	dbinom	rbinom
Poisson	ppois	qpois	dpois	rpois
Student t	pt	qt	dt	rt
Exponential	pexp	qexp	dexp	rexp
Uniform	punif	qunif	dunif	runif
Chi-Square	pchisq	qchisq	dchisq	rchisq

Las primeras distribuciones de la tabla son las mas conocidas. Por ejemplo para la distribución normal podemos calcular el área debajo de un valor de x para alguna distribución normal:

```
pnorm(27.4, mean=50, sd=20)
```

```
## [1] 0.1292381
```

Recordemos que en R podemos obviar, si queremos, los nombres de los parámetros (en este caso 'mean' y 'sd'):

```
pnorm(27.4, 50, 20)
```

```
## [1] 0.1292381
```

donde el primer argumento es el valor de x, el segundo argumento es el valor medio, y el tercero es la desviación estándar.

Gráficos

En esta unidad daremos un panorama de los distintos gráficos disponibles en R. Cabe destacar que las posibilidades de R en cuanto a visualización de datos son muy amplias, y que aquí nos enfocaremos en los gráficos más usuales. Una vez que se llega a manejar estos gráficos, resulta mucho más sencillo realizar visualizaciones más avanzadas y personalizar los gráficos que discutiremos en esta unidad y otros.

Conceptos básicos

Los gráficos son una parte muy importante para el científico de datos ya que no sólo sirven para visualizar (gráficos analíticos) sino que también permiten explorar (gráficos exploratorios) diversas características de los datos e inferir relaciones entre ellos, que usualmente son muy difíciles de ver en los datos crudos.

Por ejemplo, con los gráficos analíticos se pueden mostrar comparaciones, mostrar causalidad, mostrar datos multivariados, e integrar evidencia (números, imágenes, diagramas, anotaciones).

Por su parte, con los gráficos exploratorios se puede entender propiedades de los datos y sus relaciones, encontrar patrones y tendencias, sugerir estrategias para encontrar, validar y mejorar modelos estadísticos y predictivos,y comunicar resultados en forma clara y entendible a colegas y clientes: algo indudablemente importante!

Algunos de los tipos de gráficos que se pueden realizar en una dimensión son:

- Líneas
- Boxplot (diagrama de cajas)
- Histogramas
- Gráficos de densidad
- Gráficos de barras

Mientras que en 2 dimensiones o más los más comunes son:

- Múltiples líneas
- Scatterplot (gráfico de dispersión)
- Bubbles (diagrama de burbujas)
- Heatmap (mapa de calor)

Hay que tener presente que cada problema y cada tipo de dato (o datos) se presta mejor a uno u otro gráfico, y para el científico de datos es crucial comprender las `prestaciones' de cada tipo de visualización en función de los datos y del problema a examinar.

De esta forma, es posible maximizar la claridad y rápida comprensión de la información contenida en los datos y en los resultados que obtenemos de nuestros análisis cuando los presentamos a terceros.

Trataremos de ir enfatizando este punto en esta unidad y en las que siguen a medida que vayamos analizando algunos ejemplos.

En lo que sigue veremos ejemplos de varios de los tipos de gráficos mencionados más arriba, prestando especial atención a las generalidades (sintaxis típica de funciones de visualización en R), y a cómo manejar las opciones de los mismos.

Sistemas de graficación

En R existen tres sistemas básicos de graficación que deben usarse por separado en cada desarrollo. Es decir, una vez que se elige el sistema para cierta tarea (por ejemplo, realizar dos gráficos sobre cierto conjunto de datos) se debe continuar con el mismo, porque de otra forma los sistemas interfieren entre sí ya que incluyen algunas funciones que poseen el mismo nombre pero son distintas. Entonces, como regla general recomendamos no mezclar los sistemas de graficación .

Los tres sistemas de graficación son:

Sistema Base

Es el sistema que por defecto comienza al iniciar R y en el cual cada elemento de un gráfico es puesto uno a uno mediante una serie de comandos. Este es el sistema que utilizaremos en este curso introductorio por ser el más adecuado para aprender la mayor cantidad de comandos.

Una vez que se maneja este sistema, no es complicado manejar los otros sistemas. El sistema base está contenido en las siguientes librerías: graphics y grDevices . Vale aclarar que, al ser el sistema base el default, no hace falta cargarlo al iniciar R.

Veamos un ejemplo sencillo para empezar: el comando típico de graficación plot . A modo ilustrativo, cargaremos el set de datos llamado cars que contiene datos de distancia de frenado (en pies) en función de la velocidad (en millas por hora).

Recordemos que para eso debemos `levantar' la librería datasets que contiene los conjuntos de datos (entre ellos el cars), utilizando para ello el comando library(datasets). Luego cargamos el set de datos utilizando el comando data(cars) .

```
library(datasets)
data(cars)
```

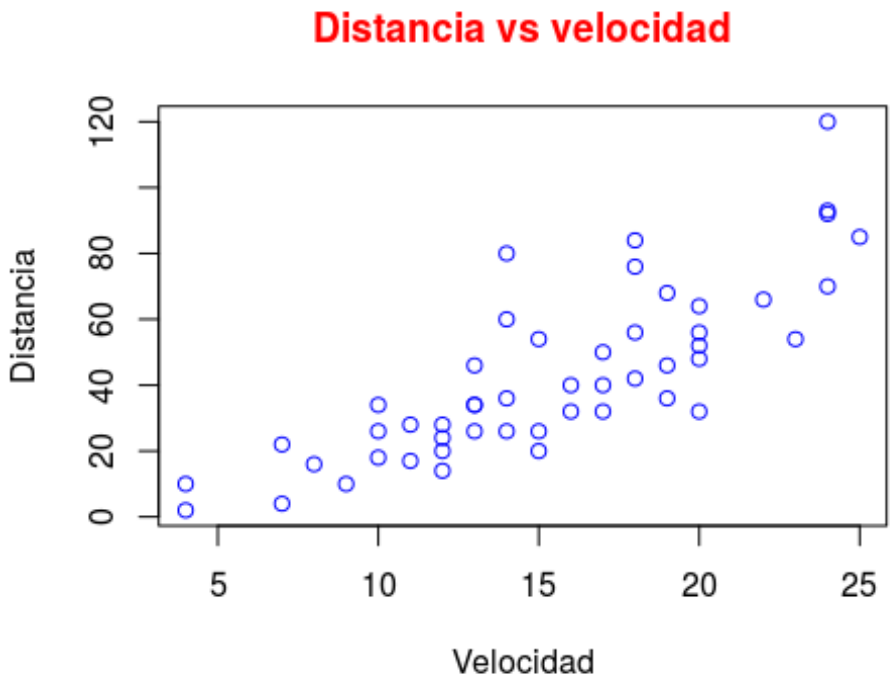
Con los datos cargados, podemos graficar con plot . Antes, hagamos un head para ver los primeros datos y conocer el nombre de las columnas:

```
head(cars)
```

```
##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10
```

Vemos que el dataframe cars contiene dos columnas, una con la velocidad ('speed') y la otra con la distancia ('dist'). Para graficar, basta escribir:

```
with(cars,plot(speed,dist,xlab = "Velocidad",
       ylab = "Distancia",
       main = "Distancia vs velocidad",
       col.main="red",col="blue"))
```



Hay varios comentarios para hacer sobre lo que acabamos de escribir. Primero, notamos que el comando plot(x,y) recibe al menos dos argumentos, x e y, que son la serie de datos que queremos como eje x y como eje y.

En nuestro caso son 'speed' como eje x, y 'dist' como eje y. Segundo, vemos que hay algunos parámetros opcionales como son la etiqueta de los ejes x e y, dados por xlab y ylab respectivamente, el título y los colores.

Además, notamos la presencia del comando with() . Esto le indica a R que queremos utilizar el set cars , y por eso en el comando plot que sigue podemos usar las variables 'speed' y 'dist' sin explicitar su origen (no necesitamos indicar su origen porque ya estamos usando with(cars,...)).

Podríamos haber procedido de otra forma. Por ejemplo, podríamos haber usado el operador $ para indicar variables de un dataframe (en nuestro caso, cars). Es decir:

```
plot(cars$speed,cars$dist,xlab = "Velocidad",
     ylab = "Distancia",
     main = "Distancia vs velocidad",
     col.main="red",col="blue")
```

Sistema Lattice

Este es un sistema condicionante, es decir que cada gráfico es creado con una sóla función, sin la necesidad de agregar elementos uno por uno. Este sistema es útil en el caso que se deban hacer muchos gráficos en la pantalla.

Para inicializarlo es necesario llamarlo con la función library(lattice) , pero a su vez necesita que la librería grid esté instalada.

Sistema ggplot2

Este es un sistema que combina los dos sistema anteriores, es decir que cada gráfico es creado con una sóla función, pero a su vez se pueden agregar elementos mediante comandos. Para inicializarlo es necesario llamarlo con la función library(ggplo2) .

Creando gráficos

Habiendo discutido los sistemas de graficación, y recordando que nos enfocaremos en el sistema base, pasaremos a ver cómo crear gráficos.

Hay dos fases para crear un gráfico:

Inicializar un nuevo gráfico

Anotar un gráfico ya existente, es decir, agregarle propiedades, por ejemplo, la etiqueta de los ejes o el título.

Llamando a las funciones de graficación, por ejemplo plot o hist , el dispositivo gráfico (conocido como device) se activa y se dibuja un gráfico en el dispositivo. Para simplificar, podemos pensar que el dispositivo no es más que una nueva ventana donde irá a parar el gráfico.

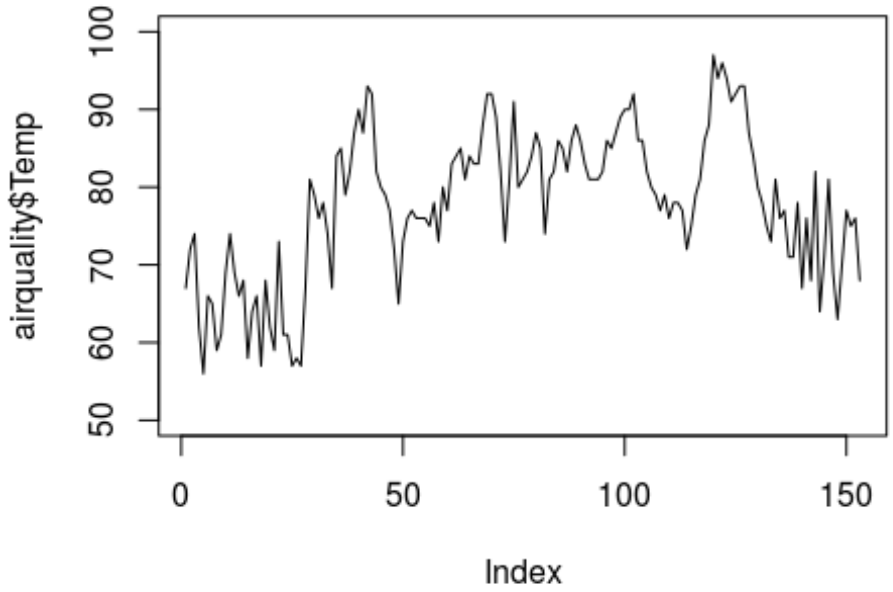
Pero hay que recordar que, en realidad, el dispositivo es un objeto y por lo tanto podría ser pasado, por ejemplo, a una función para otro tipo de procesamiento más avanzado. Algo de esto veremos cuando querramos guardar un gráfico en un archivo (en realidad estaremos mandando el dispositivo a un archivo).

Si los argumentos de la funcion plot no son especificados, entonces se usan los argumentos definidos por defecto; esta función tiene muchos argumentos que permiten que agreguemos título, nombres a los ejes, colores, ticks de los ejes, etc.

En particular, el sistema base tiene muchos parámetros que pueden ser cambiados y ajustados, los mismos se encuentran documentados y pueden ser vistos con el comando ?par . No discutiremos todas las opciones de cada gráfico porque son demasiadas y la intención es fomentar que investiguen y prueben distintos comandos y opciones.
Veamos algunos comandos para generar distintos tipos de gráficos. Para cada uno cargaremos sets de datos usando nuestros conocidos comandos library(datasets) y with. Notamos también que, para ejemplificar la flexibilidad de R y lograr que se familiaricen con distintas formas de hacer lo mismo, a veces usaremos with y otras veces usaremos otras alternativas (por ejemplo el operador $).

- Gráfico de líneas o line plot:

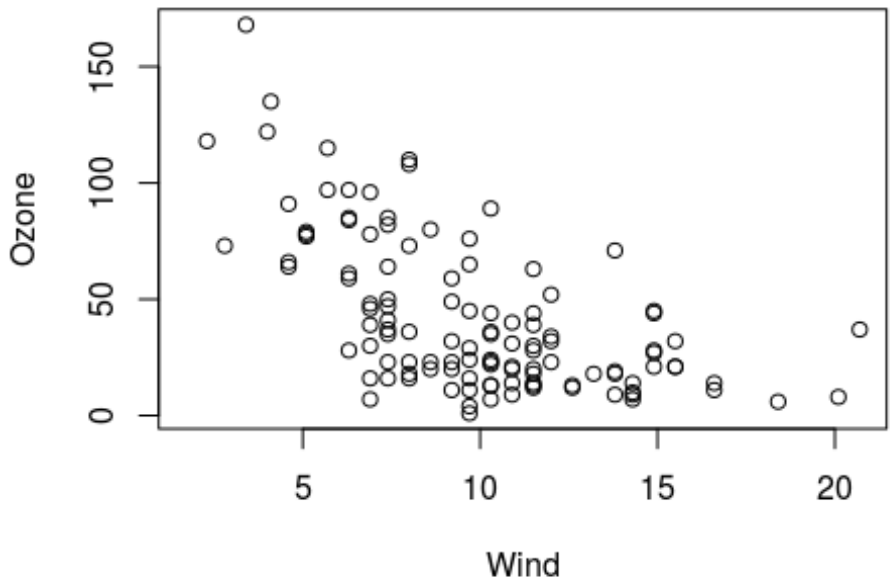
```
library(datasets)
plot(airquality$Temp, ylim=c(50,100), type="l")
```

Notemos aquí la presencia del parámetro type , definido como “l" (línea) en este ejemplo.

- Diagrama de dispersión o scatter plot:

```
library(datasets)
with(airquality, plot(Wind, Ozone))
```

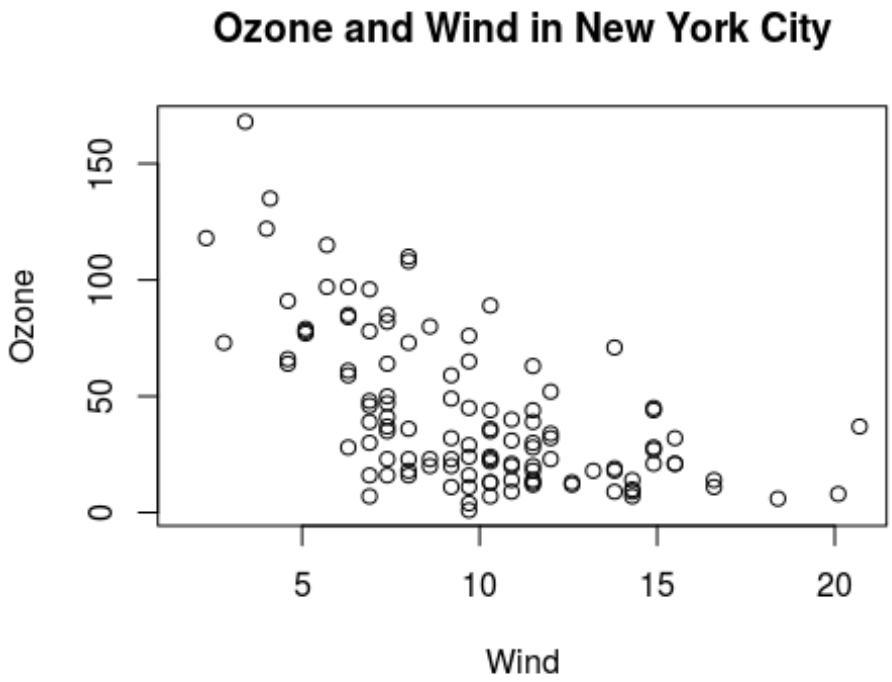


Antes de seguir con otros tipos de gráficos, aprovecharemos los diagramas de dispersión para ilustrar la incorporación de anotaciones, cambios de color de los objetos y gráficos múltiples.

Estos ejemplos nos permiten familiarizarnos con los comandos usados para personalizar y "tunear" los gráficos que hacemos.

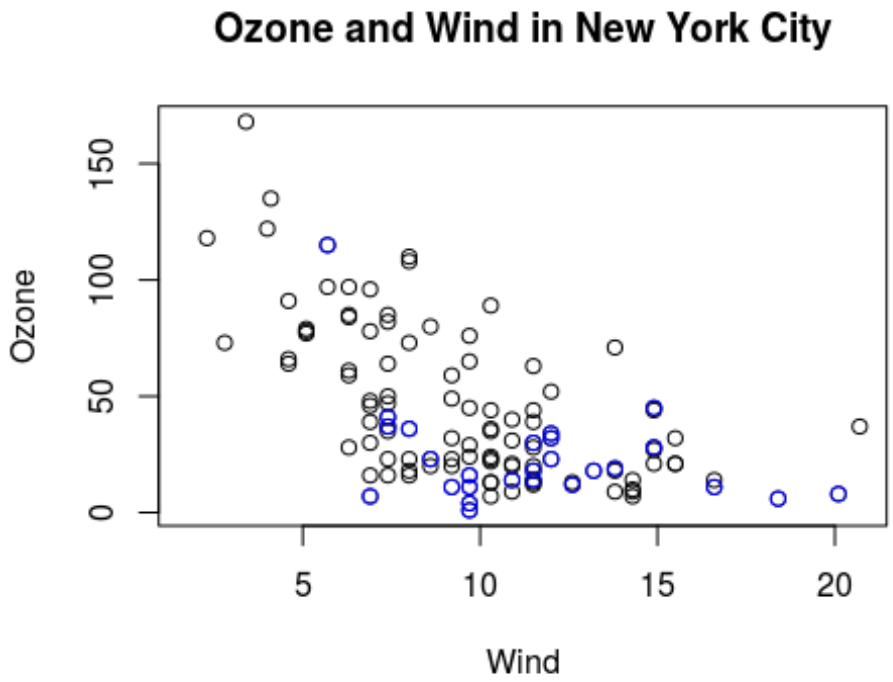
- Gráfico con título:

```
library(datasets)
with(airquality,                    plot(Wind,                    Ozone))
title(main = "Ozone and Wind in New York City")
```

- Gráfico con una selección de datos en otro color:

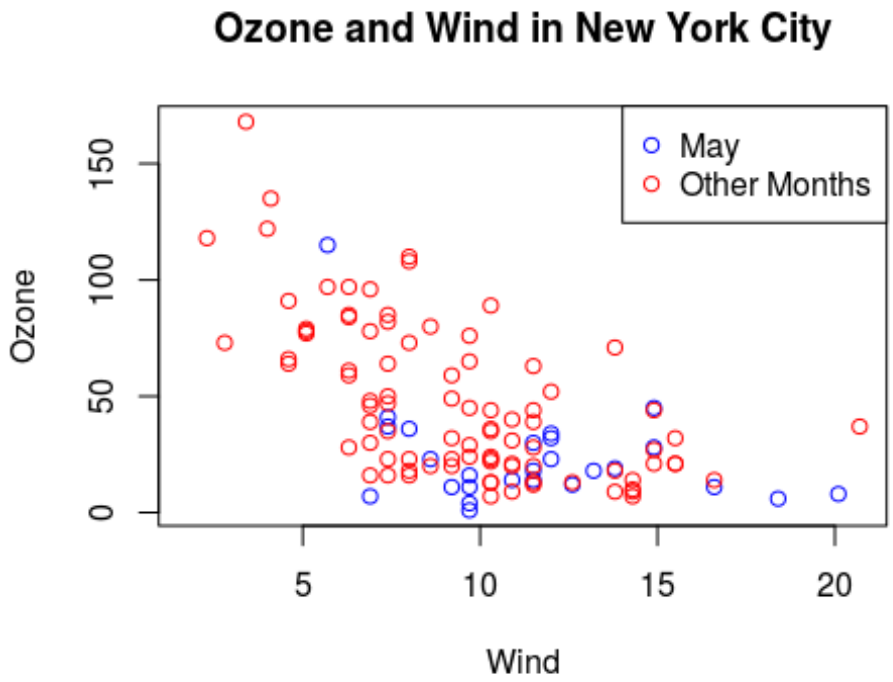
```
with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in New York
City"))
with(subset(airquality, Month == 5), points(Wind, Ozone, col = "blue"))
```



Notamos aquí el uso del comando subset para elegir un subconjunto de los datos. En este caso, del dataframe airquality nos quedamos con aquellos datos que corresponden al quinto mes, y ponemos esos puntos con color azul.

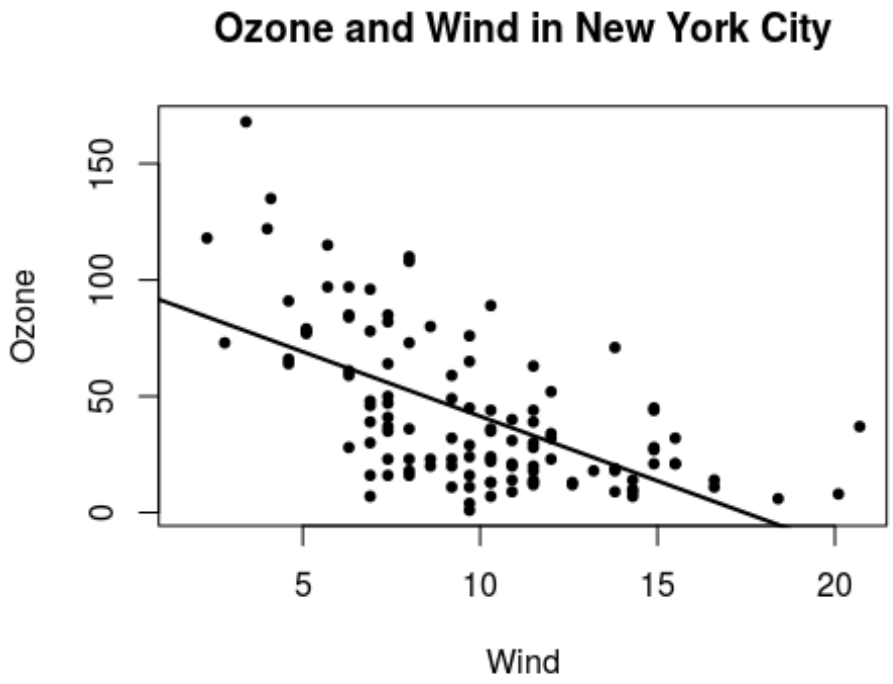
Siguiendo con este ejemplo, si quisiéramos poner en rojo todos los puntos que no corresponden al quinto mes usaríamos algo parecido pero con el operador != ('disinto de'). Aprovechamos además para incluir una legenda con legend .

```
with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in New York
City",                    type                    =                    "n"))
with(subset(airquality, Month == 5), points(Wind, Ozone, col = "blue"))
with(subset(airquality, Month != 5), points(Wind, Ozone, col = "red"))
legend("topright", pch = 1, col = c("blue", "red"), legend = c("May",
"Other Months"))
```

- Incluimos una línea de regresión como las que veremos en la unidad 4:

```
with(airquality, plot(Wind, Ozone, main = "Ozone and Wind in New York
City",                   pch                   =                   20))
model        <-        lm(Ozone        ~        Wind,        airquality)
abline(model, lwd = 2)
```

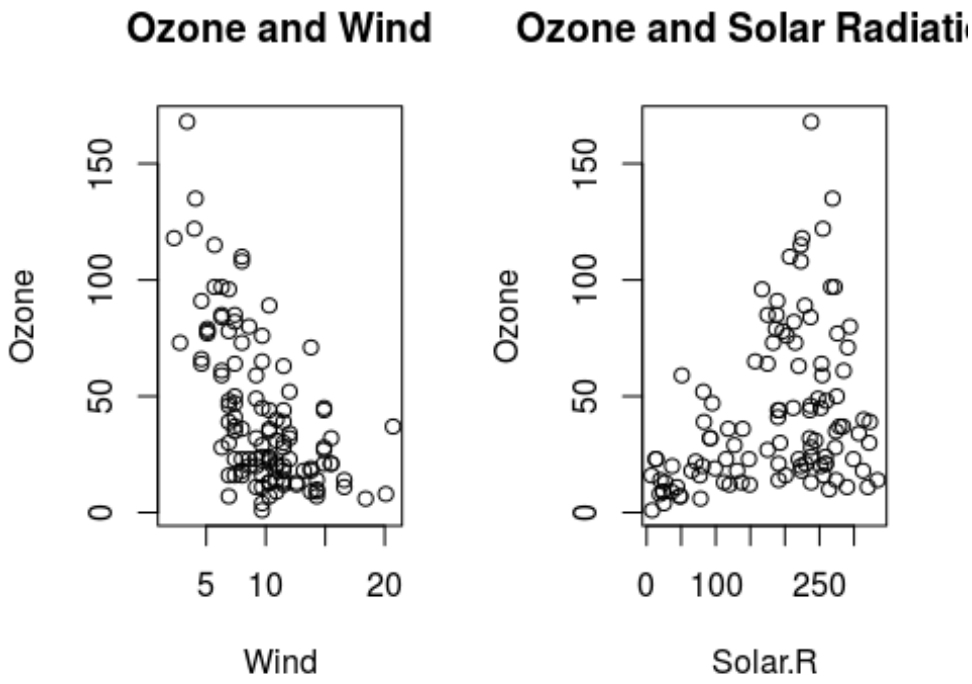
Utilizamos el comando lm (que viene de 'linear model') para calcular una regresión lineal de la variable 'Ozone' en función de la variable 'Wind' (notemos el uso de Ozone \~ Wind que ya vimos).

El segundo argumento de la función lm corresponde al dataframe utilizado, es decir 'airquality' en este caso. El resultado de lm lo guardamos en la variable 'model'.

Luego, agregamos al gráfico creado antes la línea de regresión, usando para ello el comando abline .

- Gráficos múltiples:

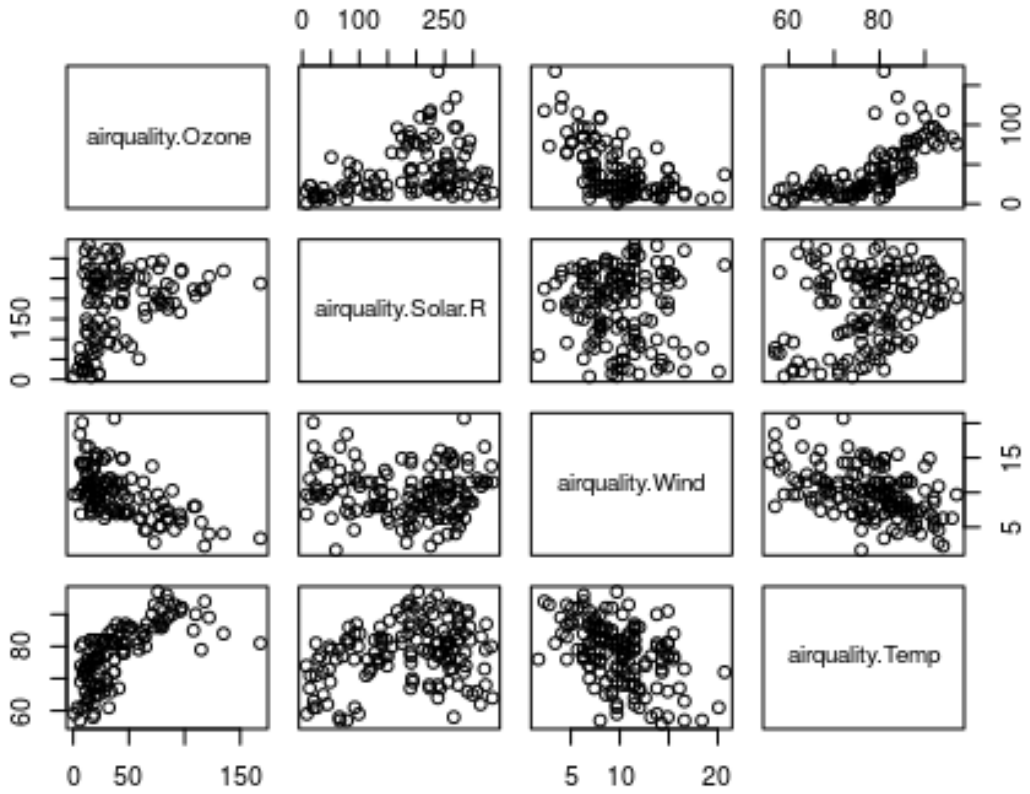
```
par(mfrow = c(1, 2))
with(airquality, {
    plot(Wind, Ozone, main = "Ozone and Wind")
    plot(Solar.R, Ozone, main = "Ozone and Solar Radiation")
})
```



Es de destacar aquí el uso del comando par y dentro del mismo el parámetro mfrow . Lo que hace mpar es separar el dispositivo (o ventana del gráfico) en filas y columnas. En este caso, a través de mfrow = cpar(1,2) le decimos que use una fila y dos columnas. En otras palabras, el display del gráfico estará separado en dos, con un gráfico al lado del otro. Dentro de cada subgráfico, se utilizan los comandos habituales (como si estuviéramos graficando en un sólo display).

- Gráfico para visualizar correlaciones:

```
pairs(data.frame(airquality$Ozone, airquality$Solar.R, airquality$Wind,
airquality$Temp))
```

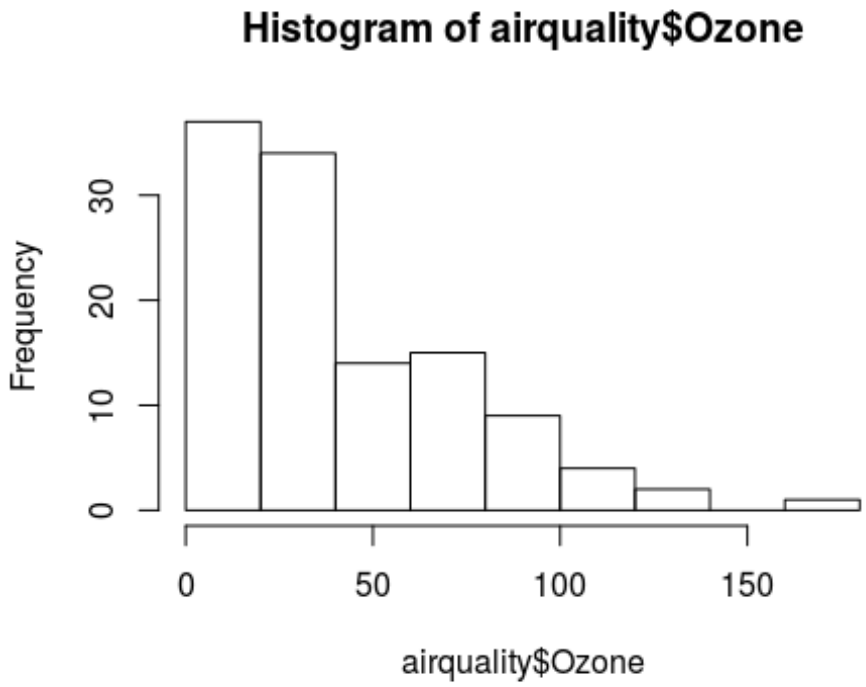
Aquí utilizamos la función pairs , aplicada a las variables '`Ozone', '`Solar.R', '`Wind', y '`Temp' del set 'airquality'.

Esta función realiza gráficos de dispersión para cada par de variables, permitiendo así visualizar muy rápidamente el grado de correlación entre variables. Volveremos sobre estos gráficos en la unidad 4 donde veremos las correlaciones.

- Histograma:

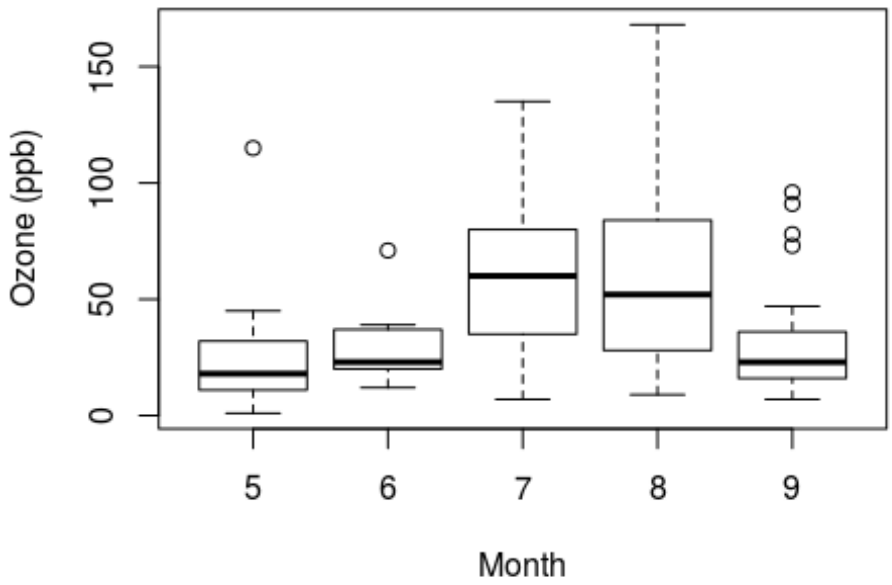
Por ejemplo, del dataframe 'airquality' queremos hacer un histograma de la variable 'Ozone'. Tenemos entonces

```
library(datasets)
hist(airquality$Ozone)
```



- Diagrama de cajas o box plot:

```
library(datasets)
airquality <- transform(airquality, Month = factor(Month))
boxplot(Ozone ~ Month, airquality, xlab = "Month", ylab = "Ozone (ppb)")
```



Notemos que en este caso utilizamos en comando transform para definir el factor en el diagrama de cajas. Definimos la variable `Month' como factor. Recordemos que en R el término factor indica variables categóricas, es decir, define los niveles de una variable categórica, o en otras palabras, los valores discretos que la misma puede tomar. Por ejemplo, para la variable` Month' los niveles son los meses del a~no.

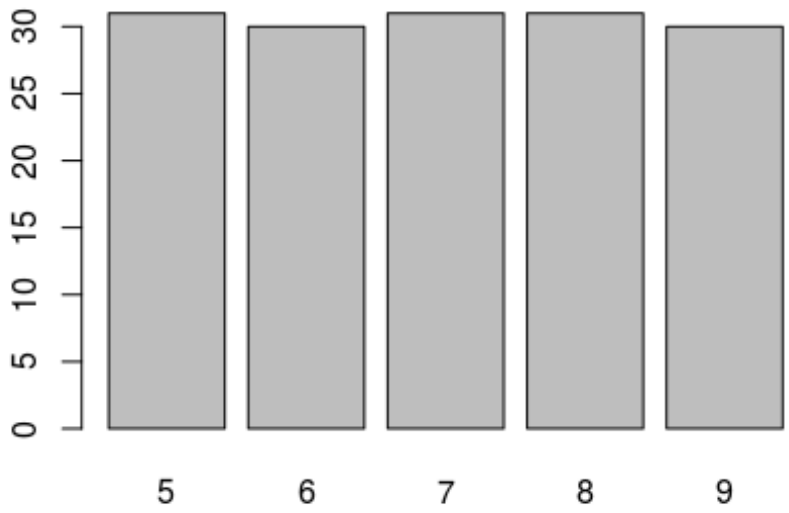
Veremos otro ejemplos de variables categóricas y sus factores a continuación. Como comentario general, notamos que los diagramas de cajas son sumamente útiles al investigar las propiedades estadísticas de un conjunto de datos en función de un factor. Con un vistazo obtenemos una idea acertada de la distribución de los datos y sus percentiles, si existen casos extremos (outliers), y si el factor en cuestión es estadísticamente significativo.

Por ejemplo, y simplificando un poco, si las cajas para dos valores del factor no se solapan quiere decir que el factor es significativo, en el sentido de que se puede asegurar que la variable en estudio depende de ese factor.

- Gráficos de barras o bar charts:

Si una variable es considerada por R como un factor, el resultado por default del comando plot es construir un gráfico de barras. Por ejemplo, sigamos con el dataset `airquality'. Si queremos ver cuántos registros hay para cada mes, podemos usar

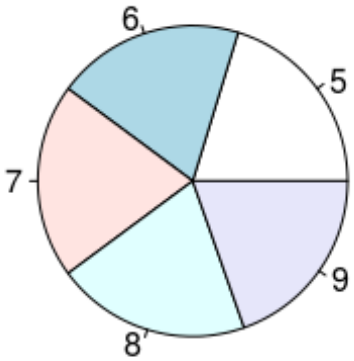
```
plot(airquality$Month)
```



- Gráfico de tortas o Pie:

Análogamente al caso de gráfico de barras, podemos hacer un gráfico de tortas

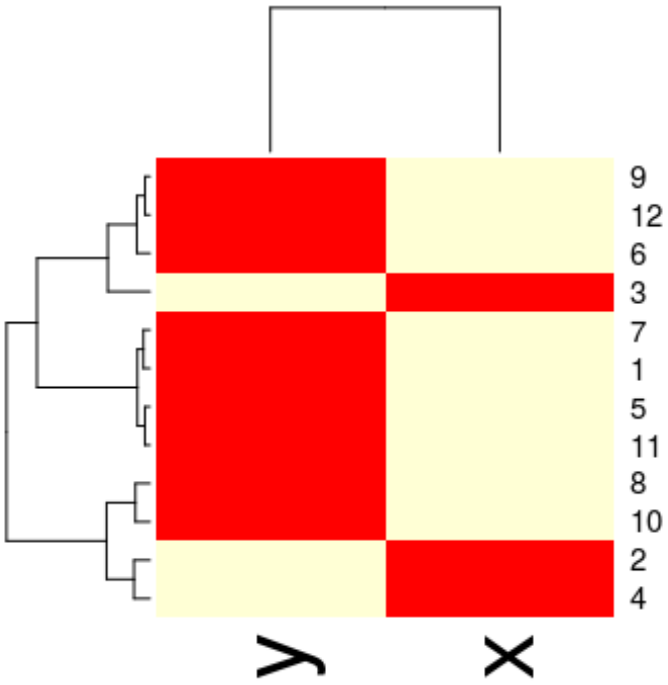
```
pie(summary(airquality$Month))
```



Notemos que en este caso hay que incluir un summary previamente.

- Mapa de calor o heatmap:

```
set.seed(1234)
par(mar = c(0, 0, 0, 0))
x <- rnorm(12, mean = rep(1:3, each = 4), sd = 0.2)
y <- rnorm(12, mean = rep(c(1, 2, 1), each = 4), sd = 0.2)
dataFrame <- data.frame(x = x, y = y)
set.seed(143)
dataMatrix <- as.matrix(dataFrame)[sample(1:12), ]
heatmap(dataMatrix)
```

Para este ejemplo, utilizamos a modo ilustrativo dos variables aleatorias que creamos con la función rnorm . Con esas dos variables construimos un dataframe con el comando series data.frame y la transformamos a una matriz con la función as.matrix . Es necesario transformar el dataframe a una matriz porque la función heatmap toma una matriz como argumento y no reconocería un dataframe.

Este tipo de gráfico es realmente útil cuando tenemos variables categóricas con muchos niveles (factores). En estos casos, la visualización por medio de gráficos estándar como gráficos de barras no resulta satisfactoria, no porque no se pueda realizar, sino más bien porque los gráficos resultantes quedan muy cargados y difíciles de leer. En cambio, los mapas de calor permiten una visualización mucho más clara.

Para este tipo de gráficos es preciso contar con dos variables categóricas que jugarán el rol de los ejes x e y, y una matriz de valores numéricos donde la fila corresponde a la primer variable categórica y las columnas corresponden a la segunda variable categórica.

Para fijar ideas, podemos pensar un ejemplo sencillo. Imaginemos que tenemos un set de datos con el número de ventas por mes y por sucursal (ambas son variables categóricas). Tenemos datos de un año y de 15 sucursales, algo difícil de acomodar en gráficos de barras y otros gráficos estándar. El heatmap resulta una solución muy adecuadada. Veamoslo, y de paso usemos este caso como ejemplo para practicar.

Creemos primero una secuencia de meses, y una secuencia de sucursales (es decir, numeramos las sucursales de la 1 a la 15, y algo similar con los meses del 1 al 12, claro). Para eso usamos la función seq :

```
meses<-seq(1,12)
suc<-seq(1,15)
```

Luego, crearemos una matriz de 12 filas (los meses) por 15 columnas (las sucursales), donde el valor del elemento sea el número de ventas (inventadas). Para eso, usamos el comando matrix :

```
A<-matrix(sample(1:30,12*15,replace=TRUE),nrow=12,ncol=15)
```

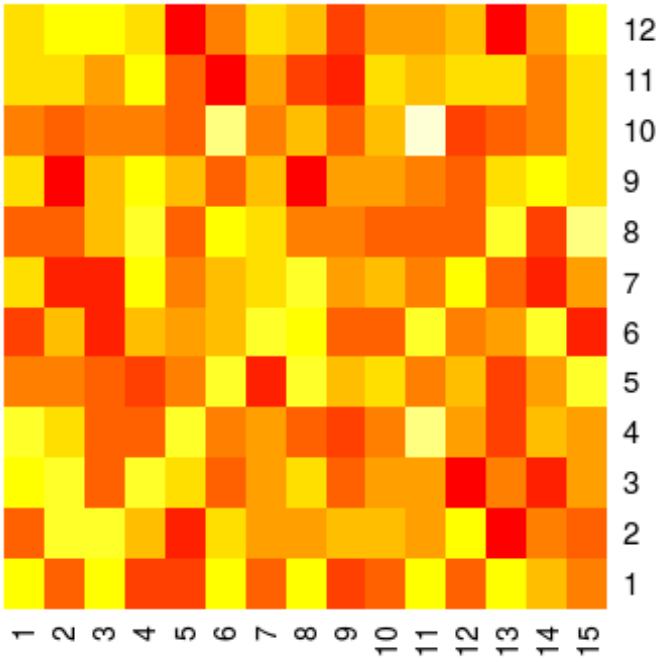
Como valores de los elementos de la matriz 'A' (dados por el primer argumento de la función matrix), usamos un entero aleatorio del 1 al 30. Esto lo hacemos con el comando

sample(1:30,12**15,replace=TRUE)* .

Este comando selecciona *12*15 elementos del conjunto dado por el primer argumento, que en este caso es una secuencia del 1 al 30. Como hay más elementos a seleccionar que elementos en el conjunto de orígen, debemos permitir que los números que se extraen puedan repetirse, de ahí la opción replace=TRUE .

Ya tenemos todo para hacer el heatmap. Notemos que el color de cada elemento en el heatmap será proporcional al número de ventas.

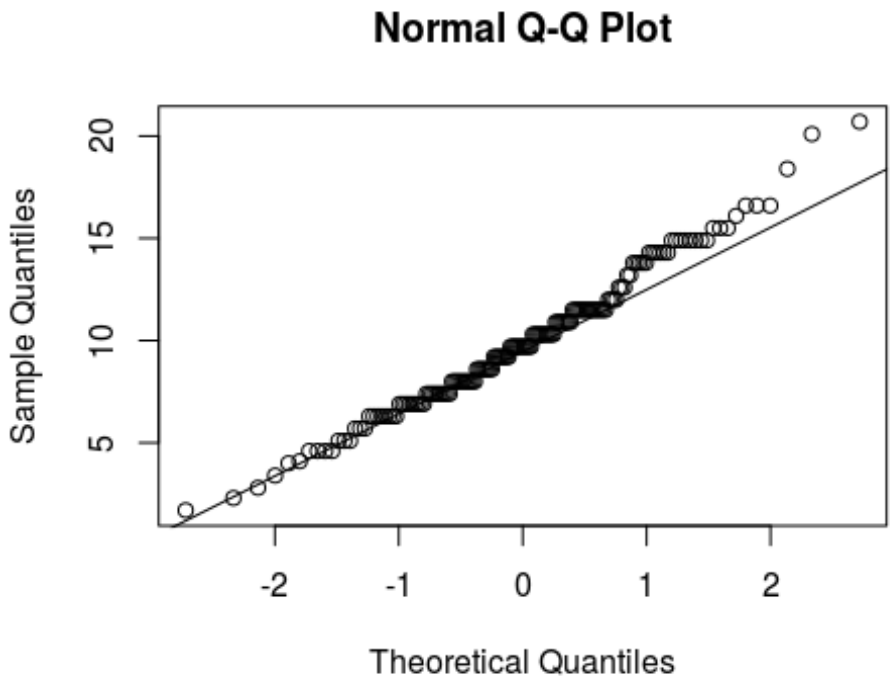
```
heatmap(A,Rowv=NA,Colv=NA)
```

Las opciones Rowv y Colv seteadas en `'NA' sirven para que R no agrupe ni las filas ni las columnas (y preserve el orden original dado en la matriz de entrada 'A')`.

- Gráfico QQ de probabilidad normal:

```
qqnorm(airquality$Wind)
qqline(airquality$Wind)
```

Utilizamos el comando qqnorm para crear el gráfico normal. En este caso, lo aplicamos a la variable `'Wind' dentro del dataset 'airquality'` que venimos utilizando. A continuación, le agregamos una línea recta a 45 grados, con el comando qqline .

Este gráfico es muy utilizado para entender si un conjunto de datos sigue o no una distribución normal. Si los puntos caen sobre la línea de 45 grados, entonces los datos siguen (exactamente) una distribución normal. Las desviaciones respecto de la línea recta indican desviaciones respecto de una distribución normal.

Hemos mostrado un buen panorama de los gráficos disponibles en R, pero como indicamos anteriormente las capacidades de visualización en R son enormes y sólo hemos explorado una porción. Por completitud y para el lector curioso, mencionamos algunos otros tipos de gráficos que resultan muy útiles en las situaciones adecuadas: gráficos de burbujas (muy útiles cuando tenemos 3 o 4 variables numéricas), dendogramas (muy usados en minería de texto y en análisis de agrupamiento), gráficos de mosaicos, correlogramas, gráficos 3D, entre otros.

Veremos algunos ejemplos de estos gráficos en las próximas unidades dedicadas a la minería de datos.

Exportando gráficos

Para terminar, mostraremos un ejemplo de como exportar nuestros gráficos a archivos. Existen varias formas de realizar esto; aquí mostraremos un ejempo sencillo a modo ilustrativo. Dejamos como ejercicio investigar otras formas de exportar gráficos en distintos formatos (png, bmp, jpg, etc.).

Como ejemplo, exportaremos el mapa de calor que realizamos anteriormente a un archivo pdf. El procedimiento para exportar a otros tipos de archivos es muy similar, sólo cambia la función que controla al dispositivo (en este ejemplo la función es la pdf):

```
pdf(file = "ejemplo_heatmap.pdf")   ## Creamos archivo y abrimos device
heatmap(A,Rowv=NA,Colv=NA)
dev.off()  ## Cerramos device
```

El proceso se divide en tres partes. Primero, se abre el dispositivo gráfico. En este caso, se trata de un dispositivo tipo pdf, y por eso utilizamos la función pdf , pasándole como argumento el nombre del archivo de destino. Luego, se crea el gráfico, en este caso es un heatmap pero podría ser cualquier tipo de gráfico (plot, hist, etc.).

Notemos que como tenemos abierto un dispositivo, no veremos el gráfico en pantalla; el gráfico queda “atrapado" en el dispositivo hasta que este se cierre. Por último, cerramos el dispositivo gráfico con dev.off() . En ese momento, el gráfico es realmente exportado al archivo.

Es importante cerrar el dispositivo porque si quedara abierto cualquier gráfico que creemos posteriormente iría a parar a ese dispositivo (lo cual sería un problema: por un lado no lo veríamos en pantalla, y por el otro, podría sobreescribir el gráfico que ya habíamos hecho si luego cerramos el dispositivo).

En las unidades que siguen continuaremos utilizando varios de los gráficos que describimos aquí y algunos otros que simplemente mencionamos, usándolos como complemento esencial de los análisis de minería de datos que llevaremos adelante.

Ejercicio opcional

- Elegir un dataset y uno o dos tipos de gráficos que se ajusten a los tipos de datos. Realizar los gráficos en dos paneles (uno al lado del otro), exportar a png y enviar.