Master en Big Data. Fundamentos matemáticos del análisis de datos.

Sesión 4. Poblaciones, muestras y probabilidad. Variables Aleatorias.

Fernando San Segundo

Curso 2020-21. Última actualización: 2020-09-19

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI
DEPARTAMENTO DE MATEMÁTICA APLICADA

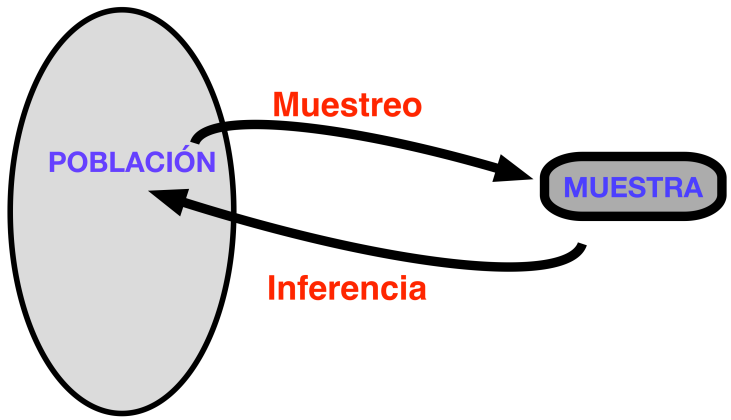
1. Población y muestra.
2. Probabilidad básica.
3. Probabilidad total y Regla de Bayes.
4. Tablas de Contingencia.
5. Variables aleatorias discretas.
6. La distribución binomial.
7. Variables aleatorias continuas.
8. Variables aleatorias normales.
9. Complementos de R: Operaciones con factores, verbos de dplyr.

Section 1

Población y muestra.

Inferencia Estadística.

- El objetivo central de la Estadística es obtener información fiable sobre las características de una **población** a partir de **muestras**. Ese término significa aquí un conjunto de entidades individuales (individuos), no necesariamente seres vivos. La población pueden ser los vehículos matriculados en 2015 o las órdenes de compra recibidas por una empresa cierto mes o las especies de colibrí que visitan un comedero en Costa Rica en los últimos 10 años, etc.
- Muchas veces estudiar toda la población es demasiado difícil, indeseable o imposible. Entonces surge la pregunta de si podemos usar las muestras para *inferir*, o *predecir* las características de la población. ¿Hasta qué punto los datos de la muestra son *representativos*de la población?
- La *Inferencia Estadística* es el núcleo de la Estadística porque da sentido a estas preguntas, las formaliza y responde.

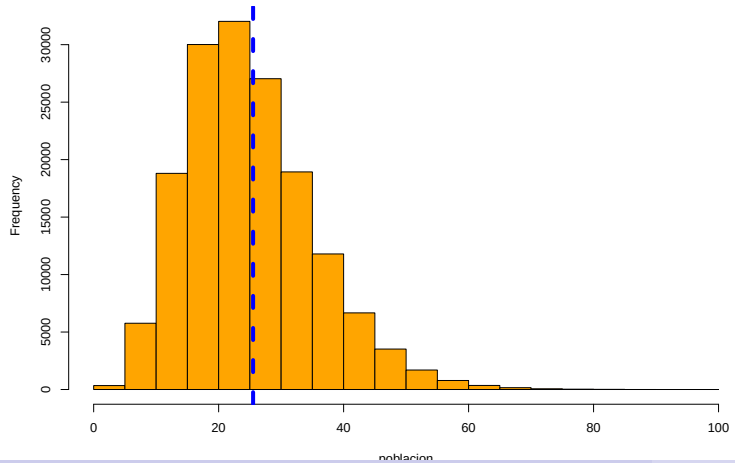


Poblaciones y muestras aleatorias simples con vectores usando R.

- Al estudiar una población nos interesan determinadas características individuales, que pueden cambiar de un individuo a otro y que constituyen las *variables de interés*. Cuando tomamos una muestra obtenemos los valores de esas variables en algunos individuos de la población.
- Para que la muestra sea representativa lo mejor es que sea una **muestra aleatoria simple**: elegimos a los individuos al azar y con remplazamiento (podemos incluir al mismo individuo más de una vez en la muestra).

```
set.seed(2019)
N = 158000
poblacion = as.integer(2 * rchisq(N, df = 13), 0)
```

- Para entenderlo mejor haremos un experimento con R. En este caso vamos a suponer una población de $N = 158000$ individuos. Por ejemplo, los viajeros que pasan por un aeropuerto en un día y sea la variable de interés su edad. El código de esta sesión construye un vector `poblacion` con las edades de los viajeros. Vamos a hacer una pequeña trampa y mostraremos el histograma de las edades. La línea de puntos indica *la media poblacional de la edad.* ¿Cuál crees que es?



Medias muestrales

- Ese es justo el tipo de preguntas que esperamos que responda la Estadística. Aunque en este caso disponemos del vector completo de edades debes tener claro que en los problemas reales no será así. Así que recurrimos a las muestras aleatorias (con remplazamiento), en inglés *random sample (with replacement)*. Por ejemplo, de tamaño 20. En R construimos una de esas muestras así:

```
n = 20
```

```
##  [1] 20 10 18 39 36 29 55 25 30 40 18 44 12 30 18 15 12 22 10 19
```

```
options(width= 70)
```

Esas son las 20 edades $x_1, \ldots, x_{20}$ de los viajeros de la muestra. Para *estimar* la edad media de *todos los viajeros* a partir de estos valores calcularíamos la **media muestral**.

$$\bar{x} = \frac{x_1 + \cdots + x_{20}}{n} = \frac{20 + 10 + \cdots + 19}{20} \approx 25.1 = \text{mean(muestra) en R}$$

- Naturalmente, si tomas otra muestra, su media muestral puede ser otra:

```
(muestra2 = sample(poblacion, n, replace = TRUE))
```

```
##  [1] 16 28 38 18 28 46 18 32 27 16 15 23 18 30 48 23 30 14 23 31
```

```
mean(muestra2)
```

```
## [1] 26.1
```

Muestras buenas y malas.

- Hemos visto que cada muestra produce una media muestral y que esas medias muestrales pueden ser distintas. ¿Cuántas muestras distintas hay? Hay una cantidad inimaginablemente grande:

$$158000^{20} = 9.4003005 \times 10^{103}$$

Para ponerlo en perspectiva, se estima que en el universo hay menos de 10^{40} estrellas. Esta cantidad enorme de muestras, de las que solo hemos visto 2, forman lo que se llama el **espacio muestral** (de tamaño $n = 20$) de este problema.

- Entre esas muestras hay muestras *buenas* y muestras *malas*. ¿Qué queremos decir con esto? Para seguir con nuestro experimento vamos a ordenar *la población completa* por edad y tomemos los 20 primeros valores:

```
(muestra3 = sort(poblacion)[1:20])
```

```
##  [1] 1 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3
```

Hemos llamado `muestra3` a ese vector porque es una más de las muchísimas muestras posibles que podríamos haber obtenido al elegir al azar 20 viajeros. Y si usáramos esta muestra para estimar la media de la población obtendríamos

```
mean(muestra3)
```

```
## [1] 2.5
```

Eso es lo que llamamos una *muestra mala*, poco representativa.

La distribución de las medias muestrales.

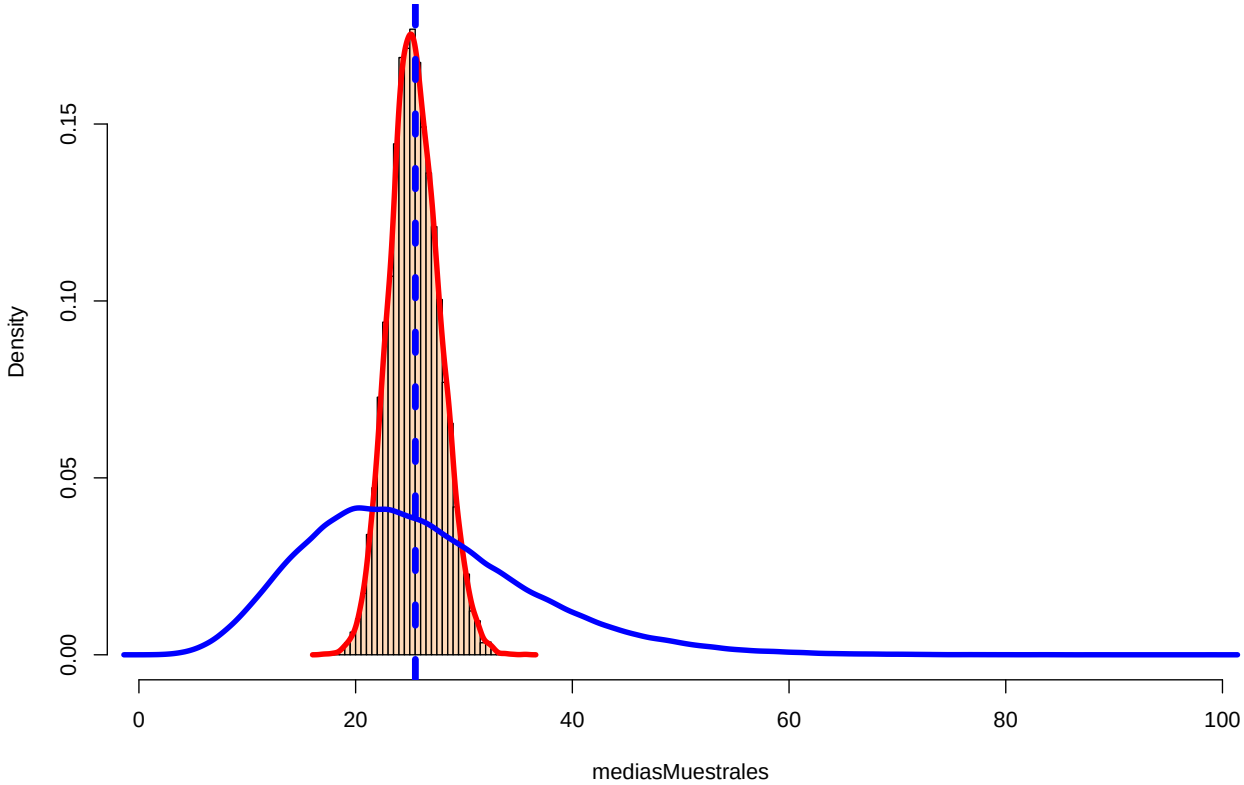
- La última muestra que hemos examinado era muy poco representativa. Pero la pregunta esencial para la estadística es ¿cuál es la relación entre muestras buenas y malas? Al elegir una muestra al azar, ¿cómo de probable es que nos toque una muestra tan mala en lugar de una buena?
- Podemos hacer otro pequeño experimento para explorar el espacio muestral. No podemos repasar todas las muestras una por una para clasificarlas en buenas o malas (eso sería demasiado incluso para R) pero podemos tomar *muchas* muestras aleatorias (pongamos $k = 10000$) y ver como de buenas o malas son (hacemos una *muestra de muestras*). En R es muy fácil hacer esto usando la función `replicate`:

```
k = 10000
# replicate repite k veces los comandos entre llaves y guarda el resultado
# del último comando en el vector mediasMuestrales
mediasMuestrales = replicate(k, {
  muestra = sample(poblacion, n, replace = TRUE)
  mean(muestra)
})
head(mediasMuestrales, 10)
```

```
##  [1] 25.00 28.70 24.85 26.05 25.75 27.15 28.05 25.15 28.40 28.40
```

Se muestran las primeras 10 de las 10000 medias muestrales que hemos obtenido.

- En lugar de examinar una a una esas 10000 medias muestrales vamos a representarlas en un histograma y una curva de densidad. Además, aprovechándonos de que en este caso tenemos acceso a la población completa hemos añadido su curva de densidad:



- Este es posiblemente **el gráfico más importante del curso**. Fíjate en tres cosas:
  - La *media de las medias muestrales* coincide con la media de la población.
  - Prácticamente no hay *muestras malas*. Es *extremadamente improbable* que una muestra elegida al azar sea muy mala.
  - La distribución de las medias muestrales tiene forma de campana (y es muy estrecha).

  Para entender bien estas ideas *necesitaremos aprender más sobre Probabilidad*.

Otra población, mismos resultados.

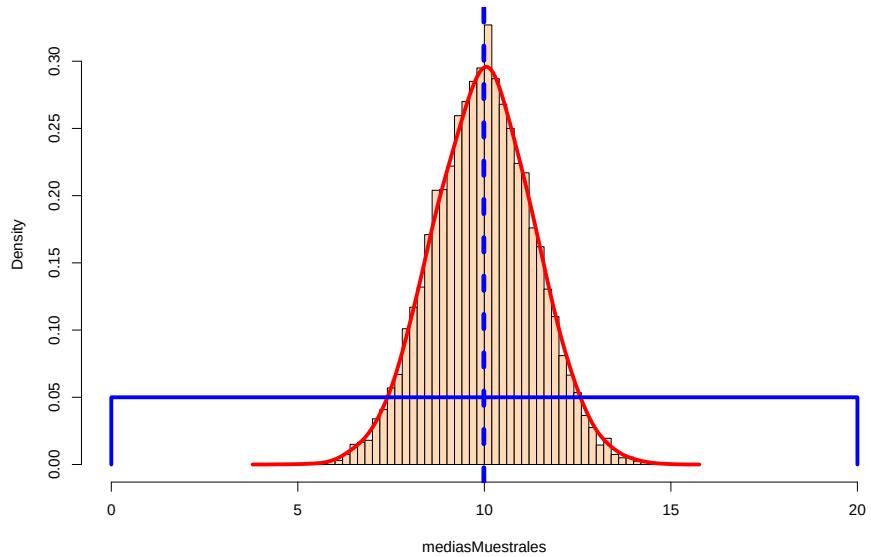
- Pero antes de lanzarnos a la probabilidad vamos a asegurarnos de algo. Puede que te preguntes si la población con la que hemos empezado tenía algo especial. Probemos con otra muy distinta. La población la forman 20000 números elegidos al azar del 0 al 20, siendo todos los valores igual de probables (su curva de densidad es horizontal).

```
poblacion = sample(0:20, 20000, replace = TRUE)
```

Y ahora repetimos el proceso de construcción de medias muestrales usando replicate

```
k = 10000
mediasMuestrales = replicate(k, {
  muestra = sample(poblacion, n, replace = TRUE)
  mean(muestra)
})
```

El gráfico del resultado muestra el mismo comportamiento de las medias muestrales, lo que se conoce como **Teorema Central del Límite**:



Section 2

Probabilidad básica.

- Para entender resultados como el Teorema Central del Límite tenemos que aprender el mínimo vocabulario necesario para poder hablar con precisión sobre la Probabilidad.
- Lo primero de lo que hay que ser conscientes es de que nuestra intuición en materia de probabilidad suele ser muy pobre. Vamos a empezar usando ejemplos de juegos de azar (dados, naipes, etc.) para poder desarrollar el lenguaje, igual que sucedió históricamente.

Experimentos del Caballero de Méré.

- ¿Qué es más probable?

  (a) obtener al menos un seis en cuatro tiradas de un dado, o
  (b) obtener al menos un seis doble en 24 tiradas de dos dados?

- Los jugadores que en el siglo XVIII se planteaban esta pregunta pensaban así:

  (a) La probabilidad de obtener un seis en cada tirada es $\frac{1}{6}$. Por lo tanto, en cuatro tiradas es $\frac{1}{6}+\frac{1}{6}+\frac{1}{6}+\frac{1}{6}=\frac{2}{3}$.

  (b) La probabilidad de un doble seis en cada tirada de dos dados es $\frac{1}{36}$, (hay 36 resultados distintos) y todos aparecen con la misma frecuencia. Por lo tanto, en veinticuatro tiradas será $\frac{24}{36}=\frac{2}{3}$.

  Así que en principio ambas apuestas parecen iguales,

- Vamos a usar R para jugar a estos dos juegos sin tener que jugarnos el dinero. Descarga este fichero de código y ejecútalo.

La paradoja del cumpleaños.

- Otro experimento que puede servir para afianzar la idea de que la probabilidad es poco intuitiva. Si en una sala hay 1000 personas entonces es seguro que hay dos que cumplen años el mismo día. De hecho basta con que haya 367 personas. Si hay menos de ese número, la probabilidad de que dos cumpleaños coincidan disminuye. ¿Cuál es el *menor número de personas* que nos garantiza una probabilidad mayor del 50% de coincidencia?
- Usemos R para averiguar ese número. Repite el experimento varias veces para convencerte..

```
## La paradoja del cumpleaños.
n = 366 # Número de personas en la sala

# Vamos a repetir el experimento N veces (N salas de n personas)
N = 10000
pruebas = replicate(N, {
fechas = sort(sample(1:366, n, replace=TRUE))
max(table(fechas)) # si el máximo es mayor que 1 es que 2 fechas coinciden
})
mean(pruebas > 1) # ¿qué proporción de salas tienen coincidencias?
```

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

Regla de Laplace.

- Fue históricamente el primer resultado que hizo posible calcular probabilidades de una manera sistemática, aunque como veremos no está libre de problemas.
- Vamos a fijar el lenguaje necesario para entender esa regla.
  (a) Estudiamos un experimento aleatorio con *n resultados elementales* posibles (no simultáneos) que además son *equiprobables*; es decir, sus frecuencias relativas son iguales cuando el experimento se repite muchas veces.:

$$\{a_1, a_2, \ldots, a_n, \}$$

  (b) El *suceso aleatorio* A es un {subconjunto del conjunto de resultados elementales}. Por ejemplo, si lanzamos un dado, A puede ser: obtener un número par.

  (c) Los resultados elementales que forman A son los {resultados favorables} a A. Por ejemplo, si lanzamos un dado, los resultados favorables al suceso

$$A = \text{obtener un número par}$$

  son $\{2, 4, 6\}$.

- **Regla de Laplace:** En esas condiciones la probabilidad de A es

$$P(A) = \frac{\text{número de sucesos elementales favorables a } A}{\text{n (número de sucesos elementales posibles)}}$$

Aplicaciones y limitaciones de la Regla de Laplace.

- Con la Regla de Laplace y un poco de Combinatoria (don't panic!) es posible responder a preguntas como estas:
  - ¿Cual es la probabilidad de que la suma de los resultados al lanzar dos dados sea igual a siete?
  - ¿Cuál es la probabilidad de que al tirar tres dados aparezca el seis en uno de los dados (no importa cual), pero sólo en uno de ellos?
  - En un paquete hay 20 tarjetas numeradas del 1 al 20. Se escogen al azar dos tarjetas. ¿Cuál es la probabilidad de que las dos que se han elegido sean la número 1 y la número 20? ¿Hay alguna diferencia entre sacar las dos tarjetas a la vez, o sacarlas consecutivamente sin remplazamiento? ¿Y si es con remplazamiento?
- Pero es necesario entender que la Regla de Laplace *no es una definición de Probabilidad*. En primer lugar porque sería una definición circular. Y en segundo lugar porque no sirve para responder a preguntas sencillas que tienen respuestas intuitivamente obvias como esta:
  - Si elegimos al azar un número real x en el intervalo $[0, 1]$, ¿cuál es la probabilidad de que sea $1/3 \leq x \leq 2/3$? ¿Qué dice (a gritos) la intuición? Y ahora trata de pensar en este problema usando la regla de Laplace. ¿Cuántos casos posibles (valores de x) hay? ¿Cuántos son los casos favorables? Experimenta con este fichero de código R.

La Regla de Laplace no se diseñó para tratar con valores continuos, como el x de este ejemplo. Necesitamos una noción de Probabilidad más general.

Teoría Axiomática de la Probabilidad.

- Los detalles técnicos son complicados pero, simplificando mucho hay tres ingredientes:

  a) Tenemos un *espacio muestral* Ω que es el conjunto de todos los posibles resultados de un experimento.

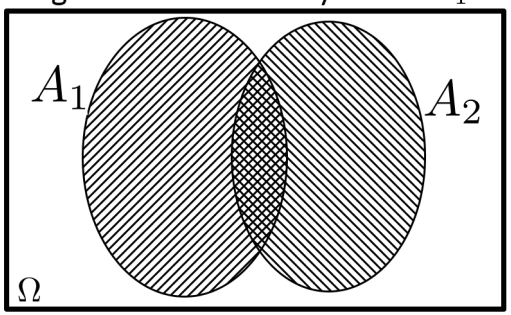
  b) Un *suceso aleatorio* es (casi) cualquier subconjunto de Ω (no *demasiado raro*).

  c) Una *función probabilidad* que representaremos con la letra P que asigna un número $P(A)$ a cada suceso aleatorio A del espacio muestral Ω. La función probabilidad debe cumplir tres propiedades:

     1. $P(\Omega) = 1$.
     2. Sea cual sea el suceso aleatorio A, se tiene $0 \leq P(A) \leq 1$.
     3. Si A_1 y A_2 son dos sucesos aleatorios entonces
     $$P(A_1 \cup A_2) = P(A_1) + P(A_2) - P(A_1 \cap A_2)$$

- Aquí $A_1 \cup A_2$ es la *unión* de sucesos y $A_1 \cap A_2$ la *intersección*, como ilustra el diagrama de Venn (cambia probabilidades por áreas e imagina que el área del rectángulo es 1).

La región doblemente rayada es $A_1 \cap A_2$



Propiedades adicionales.

- En la próxima sesión veremos ejemplos concretos y útiles de como construir esas funciones de probabilidad tanto en casos discretos como continuos.
- La probabilidad del *suceso vacío* $\emptyset$ es 0; es decir $P(\emptyset) = 0$.
- Dos sucesos A_1 y A_2 se llaman *incompatibles* o *disjuntos* si su intersección es vacía; es decir, no pueden ocurrir a la vez. En tal caso:

$$P(A_1 \cup A_2) = P(A_1) + P(A_2)$$

- Dado un suceso aleatorio A, el *suceso complementario* A^c se define como "no ocurre A". Y siempre se cumple que

$$P(A^c) = 1 - P(A).$$

- Si $A \subset B$ (se lee: si A es un subconjunto de B) entonces

$$P(A) \leq P(B)$$

- **Ejercicio:** Calcular la probabilidad de que un número de cuatro cifras tenga alguna repetida. Extra: diseña una simulación con R para comprobar tu resultado.

Probabilidad condicionada.

- El concepto de probabilidad condicionada trata de reflejar los cambios que se producen en el valor de probabilidad $P(A)$ de un suceso cuando tenemos alguna *información adicional (pero parcial)* sobre el resultado de un experimento aleatorio.
- *Ejemplo.* ¿Cuál es la probabilidad de que al lanzar un dado obtengamos un número par? Está claro que es 0.5. Pero y si te dijera, sin revelarte el resultado, que al lanzar el dado hemos obtenido un número estrictamente mayor que 3. ¿Seguirías pensando que esa probabilidad es 0.5?
- Lo que ocurre en situaciones como esa es que queremos calcular la probabilidad de un suceso A *sabiendo con certeza que* ha ocurrido otro suceso B, lo que se denomina probabilidad de A condicionada por B y se representa mediante $P(A \mid B)$. La definición, que se puede justificar con la Regla de Laplace, es:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

- *Ejemplo (continuación):*

$$P(\text{dado par}|\text{sabiendo que dado} > 3) = \frac{P(\text{dado par y a la vez dado} > 3)}{P(\text{dado} > 3)} = \frac{2/6}{3/6} = \frac{2}{3}$$

- El denominado Problema de Monty Hall es un ejemplo famoso de como la información adicional altera nuestra estimación de probabilidades (ver también).

Sucesos independientes.

- El suceso A es independiente del suceso B si el hecho de saber que el suceso B ha ocurrido no afecta a nuestra cálculo de la probabilidad de que ocurra A. Es decir, la independencia significa que $P(A \mid B) = P(A)$. Hay una manera equivalente de escribir esto que deja claro que la independencia es simétrica:

$$A \text{ y } B \text{ son independientes significa que } P(A \cap B) = P(A)\,P(B)$$

- **Nunca confundas sucesos independientes e incompatibles** Los sucesos incompatibles no pueden ser independientes.
- Esta noción de independencia es una abstracción matemática, que raras veces coincidirá en la práctica con nuestra noción intuitiva de que dos fenómenos son independientes. Más adelante tendremos ocasión de profundizar en esta discusión y hablaremos de como medir en casos reales la independencia.

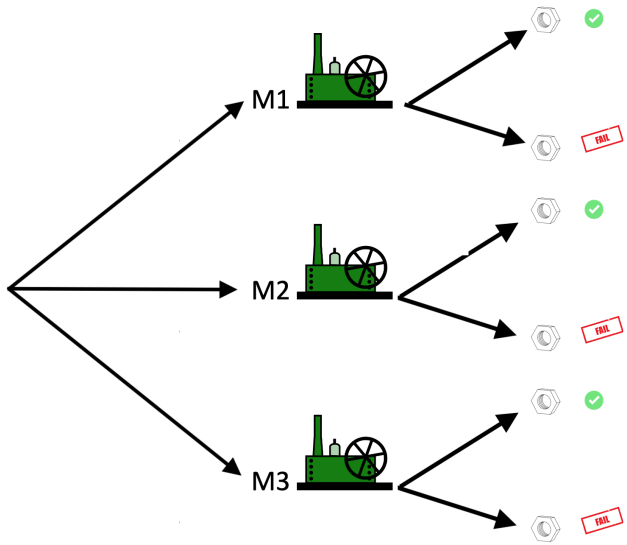
Section 3

Probabilidad total y Regla de Bayes.

Teorema de la probabilidad total.

- Este resultado sirve para calcular la probabilidad de un suceso A que puede ocurrir a través de uno de entre k mecanismos excluyentes.
- *Ejemplo:* una fábrica produce un tipo de piezas usando tres máquinas. (a) Cada pieza proviene de una y una sola de esas máquinas. (b) Cada una de las máquinas produce una fracción conocida de las piezas y (c) tiene una tasa de piezas defectuosas también conocida. Con esa información queremos calcular la tasa total de piezas defectuosas.

• Sea A el suceso *la pieza es defectuosa*; lo que queremos calcular es $P(A)$. Sean M_1, M_2, M_3 los sucesos *la pieza se fabrica en la máquina 1 o en la 2 o 3 respectivamente.*



- T conocemos las tres probabilidades condicionadas $P(A \mid M_1)$, $P(A \mid M_2)$ y $P(A \mid M_3)$.
- En problemas como este el Teorema de la Probabilidad Total afirma que:

$$P(A) = \underbrace{P(A \mid M_1)P(M_1) + P(A \mid M_2)P(M_2) + P(A \mid M_3)P(M_3)}_{\text{un término para cada máquina / camino}}$$

Teorema de Bayes

- El Teorema de Bayes se usa en situaciones idénticas a la que acabamos de ver, pero sirve para hacer una *pregunta inversa*. *Sabiendo que la pieza es defectuosa*, ¿cuál es la probabilidad de que provenga de la máquina M1 (por ejemplo)? Se trata por tanto de calcular $P(M_1 \mid A)$. Y el resultado es:

$$P(M_1 \mid A) = \frac{P(A \mid M_1)P(M_1)}{P(A \mid M_1)P(M_1) + P(A \mid M_2)P(M_2) + P(A \mid M_3)P(M_3)}$$

  Fíjate en que el denominador es $P(A)$.

- *Ejemplos:*
  - responde a la pregunta con la que se abre esta página.
  - Lo más difícil al usar el Teorema de Bayes suele ser identificar los datos de forma correcta, Un ejemplo típico se ilustra en este problema: *Un hospital tiene dos quirófanos en funcionamiento. En el primero se han producido incidentes en el 20% de sus operaciones y el segundo sólo en el 4%. El número de operaciones es el mismo en ambos quirófanos. La inspección hospitalaria analiza el expediente de una operación, elegido al azar y observa que en esa operación se produjo un incidente. ¿Cuál es la probabilidad de que la operación se realizara en el primer quirófano?*

Jugando con el Teorema de Bayes y R

- Vamos a usar una tabla de datos sobre spam en mensajes de correo electrónico. La tabla se llama `spam` y pertenece a la librería `kernlab`. Instala la librería y carga la tabla con `data(spam)`, La tabla contiene datos sobre varios miles de mensajes de correo. La última columna contiene la clasificación como spam o no spam. Las primeras 48 columnas indican el porcentaje de palabras del mensaje que coinciden con el título de la columna. Aquí se muestra una parte de la tabla:

```
spam[1:4, c(1:10, 58)]
```

```
##   make address  all num3d  our over remove internet order mail type
## 1 0.00    0.64 0.64     0 0.32 0.00   0.00     0.00  0.00 0.00 spam
## 2 0.21    0.28 0.50     0 0.14 0.28   0.21     0.07  0.00 0.94 spam
## 3 0.06    0.00 0.71     0 1.23 0.19   0.19     0.12  0.64 0.25 spam
## 4 0.00    0.00 0.00     0 0.63 0.00   0.31     0.63  0.31 0.63 spam
```

- Con estos datos y usando funciones de R responde a estas preguntas:
  - ¿Cuál es la probabilidad de que un mensaje elegido al azar sea spam?
  - ¿Cuál es la probabilidad de que un mensaje elegido al azar contenga la palabra *order*?
  - *Sabiendo* que un mensaje es spam, ¿cuál es la probabilidad de que contenga la palabra *order*?
  - Y ahora, usando la fórmula de Bayes, vamos a construir el programa antispam más simple del mundo: sabiendo que un mensaje contiene la palabra *order*, ¿cuál es la probabilidad de que sea spam?
- Este método es muy rudimentario, pero cuando aprendas algoritmos de clasificación estudiarás el método Naive Bayes (Bayes ingenuo) que se basa en ideas similares.

Section 4

Tablas de Contingencia.

Tablas de contingencia 2x2

- En el problema anterior nos hemos encontrado con una situación típica en la que hay dos factores binarios. Un factor S con valores *spam* / *no spam* y un factor O, con valores "contiene order/ no contiene order". Al combinarlos hay cuatro casos posibles que podemos representar en una tabla dos por dos.
- Primero usaremos `dplyr` para obtener una tabla en la que solo aparezcan esos dos factores, aprendiendo de paso alguna manipulación adicional:

```
library(tidyverse)
spam = spam %>%
  select(order, type) %>%
  mutate(hasOrder = factor(order > 0, # Creamos el factor hasOrder
                           levels = c(TRUE, FALSE),
                           labels = c("order", "no order")),
         type = relevel(type, ref = "spam"), # Reordenamos los niveles
         -order) # y elminamos el factor order original
```

Ahora podemos obtener la tabla con

```
table(spam$hasOrder, spam$type)
```

```
           spam nonspam
order       555     218
no order   1258    2570
```

Vocabulario adicional para tablas de contingencia 2x2

- El lenguaje de las tablas de contingencia proviene en buena medida del contexto de las pruebas diagnósticas para enfermedades. Esas pruebas no son infalibles: a veces dan como resultado que una persona padece la enfermedad, cuando en realidad no es así. Es lo que se llama un *falso positivo* (FP). En otras ocasiones será al contrario. La prueba dirá que la persona está sana, aunque de hecho está enfermo. Eso es un *falso negativo* (FN). Los resultados correctos, que están en la diagonal principal de la tabla, son los TP (true positives) y los TN (true negatives).
- Por ejemplo, podemos tener una tabla como esta:

		Padecen la enfermedad		
		Enfermo	Sano	Total
Resultado de la Prueba	Positivo	TP = 192	FP = 158	350
	Negativo	FN = 4	TN = 9646	9650
	Total	196	9804	10000

- Vamos a usar este script de R para aprender algo más de lenguaje sobre tablas de contingencia y de como manejarlas con R.
- Una prueba diagnóstica es un *clasificador* de pacientes. Más adelante vamos a encontrar muchos algoritmos clasificadores, porque clasificar es una de las tareas básicas en *Machine Learning*. Veremos entonces que en ese contexto se usa mucho el vocabulario de pruebas diagnósticas.

Section 5

Variables aleatorias discretas.

Modelos teóricos frente a datos empíricos.

- Vamos a proponerte un pequeño experimento mental. Imagínate que lanzamos un dado (honesto, no cargado) un millón de veces y que calculamos las *frecuencias relativas* de cada uno de los valores. ¿Qué números crees que habrá en la segunda fila de esta tabla?

valor del dado	1	2	3	4	5	6
frecuencia relativa	?	?	?	?	?	?

  Esos valores que ves con claridad en tu cabeza son un *modelo* teórico del experimento aleatorio que consiste en lanzar un dado. Y esa es precisamente la idea que trata de captar una variable aleatoria discreta: *un modelo teórico de un experimento aleatorio cuyos resultados son un conjunto discreto de valores.*

- Para describir una variable aleatoria discreta X tenemos por tanto que dar su **tabla (o función) de densidad de probabilidad**: una tabla de valores posibles de X y sus correspondientes probabilidades:

valor de X	x_1	x_2	$\cdots$	x_k
Probabilidad de ese valor $P(X = x_i)$	p_1	p_2	$\cdots$	p_k

  con $p_1 + p_2 + \cdots + p_k = 1$. A veces usaremos *notación funcional* $f(x_i) = P(X = x_i)$.

Ejercicio: usa R para hacer ese experimento y compara los datos empíricos con el modelo.

Media y varianza de distribuciones discretas.

- Una variable aleatoria discreta es un modelo teórico de la distribución de valores en la población. La **media poblacional** o **esperanza** es la media aritmética de dichos valores y se representa con la letra griega μ o con el símbolo $E(X)$. De forma análoga se define la **varianza poblacional** que denotaremos σ^2.
- Nuestro objetivo es utilizar datos muestrales para estimar o inferir los parámetros de una población. Si tenemos una muestra de una variable discreta que toma k valores distintos $x_1, \ldots, x_k$ con frecuencias absolutas $f_1, \ldots, f_k$ respectivamente podemos calcular la *media muestral* haciendo:

$$\bar{x} = \frac{x_1 f_1 + \cdots + x_k f_k}{n} = x_1 fr_1 + \cdots + x_k fr_k$$

donde $fr_1, \ldots, fr_k$ son las *frecuencias relativas* de los valores. Recuerda que las frecuencias relativas son las versiones *empíricas* de las probabilidades *teóricas*. Por eso la media poblacional μ (teórica) se calcula así a partir de la tabla de probabilidades:

$$\boxed{\mu = x_1 p_1 + \cdots + x_k p_k}$$

Una razonamiento similar conduce a esta expresión para la *varianza poblacional*

$$\boxed{\sigma^2 = (x_1 - \mu)^2 p_1 + \cdots + (x_k - \mu)^2 p_k}$$

- **Ejercicio:** usa R para calcular μ y σ^2 para un dado.

Usando `sample` con variables aleatorias discretas.

- **Ejercicio:** Dada esta tabla de densidad de probabilidad de una variable aleatoria X:

valor de X	0	1	2	3
Probabilidad de ese valor $P(X = x_i)$	$\frac{64}{125}$	$\frac{48}{125}$	$\frac{12}{125}$	$\frac{1}{125}$

  usa R para calcular μ, σ^2 y también σ, la desviación típica poblacional.

- Hasta ahora hemos usado `sample` para fabricar muestras en las que todos los elementos del vector eran equiprobables. Pero también podemos simular muestras de una población como la que describe el modelo teórico X usando la opción `prob` así (fíjate en que no hace falta *normalizar las probabilidades*):

```
muestra = sample(0:3, size = 10, replace = TRUE, prob = c(64, 48, 12, 1))
```

- **Ejercicio:**
  (1) Simula una muestra de tamaño 1000 de esta variable. ¿Cuál crees que es la mejor manera de representar gráficamente esa muestra?

  (2) Combina `sample` con `replicate` para simular cien mil muestras de tamaño 10. Estudia la distribución de las medias muestrales como hemos hecho en ejemplos previos.

Operaciones con variables aleatorias.

- Imagina que la variable aleatoria X representa el gasto en seguro del hogar y la variable Y el gasto en seguro del automóvil. Si queremos calcular el gasto total en ambos seguros tenemos que pensar en la *variable suma* $X + Y$. De la misma forma, a veces queremos multiplicar una variable por un número y, en general, vamos a pensar en combinaciones de la forma $a\,X + b\,Y$ donde a y b son coeficientes numéricos.
- La media $E(X + Y)$ de la variable $a\,X + b\,Y$ se calcula a partir de las medias de X e Y usando la misma combinación

$$a\,\mu_X + b\,\mu_Y$$

- Para la varianza las cosas son más complicadas, porque involucran la noción de *independencia*, que discutiremos después. Informalmente, X e Y son independientes si la información sobre el valor de X no afecta a la tabla de probabilidades de los valores de Y. La **covarianza** de X e Y es:

$$\text{cov}(X, Y) = E((X - \mu_X)(Y - \mu_Y))$$

  y en general $\sigma^2(a\,X + b\,Y) = a^2\,\sigma_X^2 + b^2\,\sigma_Y^2 + 2\,a\,b\,\text{cov}(X, Y)$
- **Cuando X e Y son independientes** se tiene $\text{cov}(X, Y) = 0$ y por tanto:

$$\sigma^2(a\,X + b\,Y) = a^2\,\sigma_X^2 + b^2\,\sigma_Y^2$$

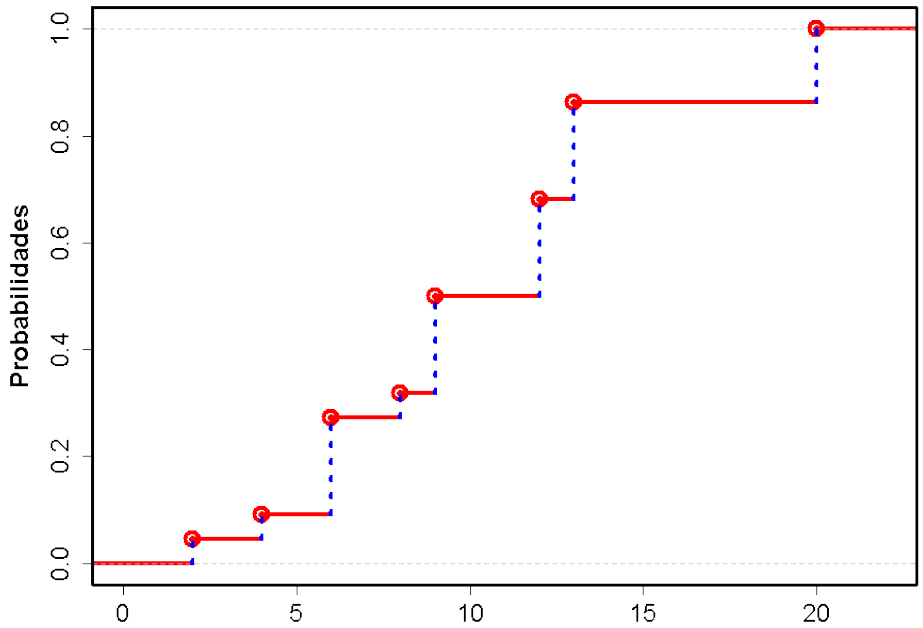
Función de distribución de una variable aleatoria discreta.

- La función de distribución F_X de una variable aleatoria X (discreta o continua) se define así para cualquier número k:

$$F_X(k) = P(X \leq k)$$

  Para una variable aleatoria la función de distribución juega un papel similar al de una *frecuencia relativa acumulada*, respondiendo a la pregunta ¿qué probabilidad hay de obtener un valor menor o igual que k?

- La gráfica de la función de distribución de una variable aleatoria discreta típica tiene este aspecto:



Section 6

La distribución binomial.

Variables aleatorias de Bernouilli.

- Son probablemente las variables aleatorias discretas más sencillas de todas. Una variable aleatoria X es de tipo Bernouilli con parámetro p si su tabla de valores y probabilidades es:

Valor de X:	1	0
Probabilidad de ese valor:	p	$q = 1 - p$

- Por ejemplo, la variable $X =$"número de seises al lanzar un dado una vez" es una variable de tipo Bernouilli, con $p = \frac{1}{6}$, $p = \frac{5}{6}$. Para representar esto decimos que $X \sim Bernouilli(p)$ (el símbolo $\sim$ se lee *"es de tipo..."*).
- Los valores 1 y 0 se denominan, arbitrariamente, **éxito y fracaso** respectivamente.
- La media de una variable $X \sim Bernouilli(p)$ es $\mu = p$
(porque $1 \cdot p + 0 \cdot q = p$)
- Su varianza es $\sigma^2 = p \cdot q$
(porque $(1-p)^2 \cdot p + (0-p)^2 \cdot q = q^2 p + p^2 q = pq(p+q) = pq$).
- Las variables aleatorias de Bernouilli son útiles porque las usaremos como piezas para construir variables más complicadas, como la binomial que vamos a ver a continuación.

Variable aleatoria binomial.

- **Ejemplo:** Lanzamos un dado 11 veces. Es importante entender que *el experimento no es una tirada sino 11 tiradas* del dado. Definimos la variable X así:

$$X = \textit{número de veces que obtenemos un 6 en esas 11 tiradas}$$

- Esta situación tiene las siguientes características:

  (1) Un **experimento básico**, lanzar un dado se **repite** n **veces** (en el ejemplo $n = 11$).

  (2) Las repeticiones del experimento básico son **independientes** entre sí.

  (3) Cada repetición del experimento sólo puede terminar de una de estas dos maneras: en **éxito (success)** (en el ejemplo, sacar un 6) que se representa con el valor 1; o un **fracaso (failure)** (no sacar un 6) que se representa con el valor 0.

  (4) La **probabilidad de éxito** en cada repetición se denomina p y la de fracaso es $q = 1 - p$. En el ejemplo $p = 1/6, q = 5/6$.

  (5) . La variable X es la **suma del número de éxitos en las** n **repeticiones independientes**.

- **Definición de la variable aleatoria binomial.**
  Una variable aleatoria discreta X que reúne esas características es una variable aleatoria binomial de parámetros n y p, y escribiremos $X \sim B(n, p)$.

Ejemplo:

- Vamos a ver un ejemplo de variable binomial. Para ello usaremos la variable `prevalentHyp` de la tabla `fhs` que hemos usado en sesiones previas. Esa variable vale 1 si el paciente hipertenso y 0 en caso contrario. Para insistir en la arbitrariedad de la elección mantenemos esos valores y definimos como *éxito* el hecho de que el paciente sea hipertenso.
- **Ejercicio:** carga esa tabla de valores y comprueba que si elegimos un paciente al azar, la probabilidad de éxito (de que sea hipertenso) es $p \approx 0.3106$.
- Para definir una variable binomial vamos a elegir al azar $n = 7$ pacientes y nos preguntamos por el número X de hipertensos que hay entre esos siete.
- **Ejercicio:**

  (a) ¿Qué valores puede tomar X?

  (b) Escribe código en R para extraer una muestra de 7 pacientes (con remplazamiento) y contar cuántos de ellos son hipertensos (es decir, para calcular X en esa muestra).

  (c) Usa replicate para fabricar 50000 de esas 7-muestras. Llama `X` al vector de 50000 valores de la variable que se obtiene y haz una tabla de frecuencias relativas de valores de `X`.

Densidad de probabilidad en la binomial.

- Las frecuencias relativas de la última tabla son aproximaciones empíricas a las probabilidades teóricas de la binomial que vamos a calcular a continuación.
- Dada una variable $X = B(n, p)$ la probabilidad de obtener k éxitos es:

$$P(X = k) = \binom{n}{k} p^k \, q^{(n-k)}$$

donde el *número combinatorio* es:

$$\binom{n}{k} = \frac{\overbrace{n\,(n-1)\,(n-2)\cdots(n-k+1)}^{k \text{ factores}}}{k!}$$

y $k! = k \cdot (k-1) \cdot (k-2) \cdot \cdots \cdot 2 \cdot 1$ es el factorial de k.

- **Media y varianza de una variable binomial.** Una variable $X \sim B(n, p)$ es la suma de n variables de Bernouilli independientes (recuerda, que toman valores 0 o 1). Usando los resultados generales sobre variables aleatorias se obtiene:

$$\text{Si } X \sim B(n, p) \text{ entonces } \mu = n\,p, \quad \sigma^2 = n\,p\,q.$$

La binomial con R.

- Para calcular probabilidades concretas como $P(X = 3)$ en R usamos la función `dbinom` (suponiendo que ya has definido p):

```
dbinom(x = 3, size = 7, prob = p)
```

```
## [1] 0.2369079
```

Con `dbinom` podemos calcular a la vez *todas* las probabilidades de la variable binomial (mostramos tres cifras significativas):

```
signif(dbinom(x = 0:7, size = 7, prob = p), digits = 3)
```

```
## [1] 0.074000 0.233000 0.315000 0.237000 0.107000 0.028900 0.004330 0.000279
```

Compara estos valores, que son predicciones teóricas, con las frecuencias relativas empíricas que hemos obtenido tomando muestras.

- La función de distribución $F(k) = P(X \leq k)$ de una binomial se calcula en R con:

```
signif(pbinom(q = 0:7, size = 7, prob = p), digits = 3)
```

```
## [1] 0.074 0.307 0.623 0.860 0.967 0.995 1.000 1.000
```

- Además R permite simular valores aleatorios de la variable binomial mediante:

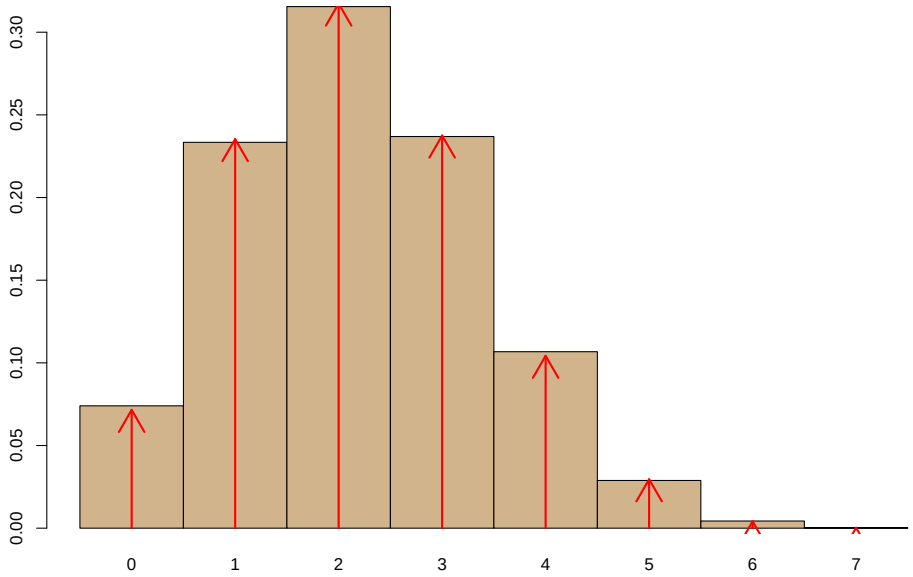
```
rbinom(n = 25, size = 7, prob = p)
```

```
##  [1] 2 5 2 1 2 3 1 1 1 4 3 3 1 1 3 3 3 3 4 0 3 3 2 3 2
```

Representación gráfica de la variable binomial.

- Para visualizar la tabla de densidad de probabilidad de una variable binomial con n moderado lo mejor es utilizar un diagrama de barras como este que muestra como se *distribuye* la probabilidad sobre los valores de 0 a n (hemos reducido a 0 es espacio entre barras por razones que pronto quedarán claras).

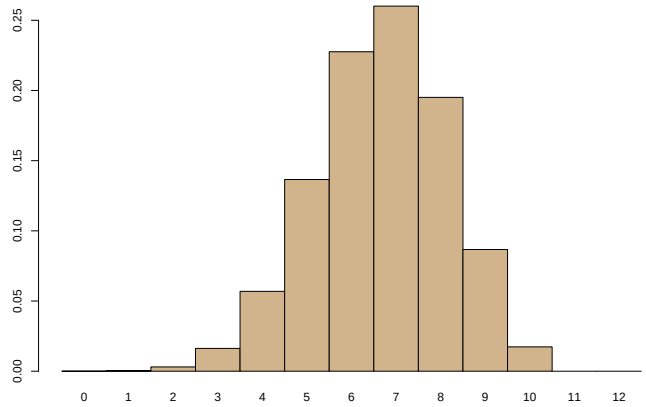
```
probabilidades = dbinom(x = 0:7, size = 7, prob = p)
bp = barplot(probabilidades, space = 0, col="tan", names.arg = 0:7)
```



Las flechas rojas representan las frecuencias relativas (¡empíricas!) de la muestra de miles de valores de X que hemos construido antes. Como puedes ver el acuerdo entre las predicciones de la teoría que representa el modelo de la variable binomial y los valores empíricos de la muestra es muy alto.

El zoo de las binomiales.

- Vamos a fijarnos en la forma de las distribuciones binomiales para distintos valores de n y p. Empezaremos por pensar en valores moderados de n (como 10) y de p (ni cerca de 0, ni cerca de 1). La siguiente figura muestra, a modo de ejemplo la distribución binomial $B(12, \frac{2}{3})$.



En general nos vamos a encontrar con **tres tipos de distribuciones binomiales**:

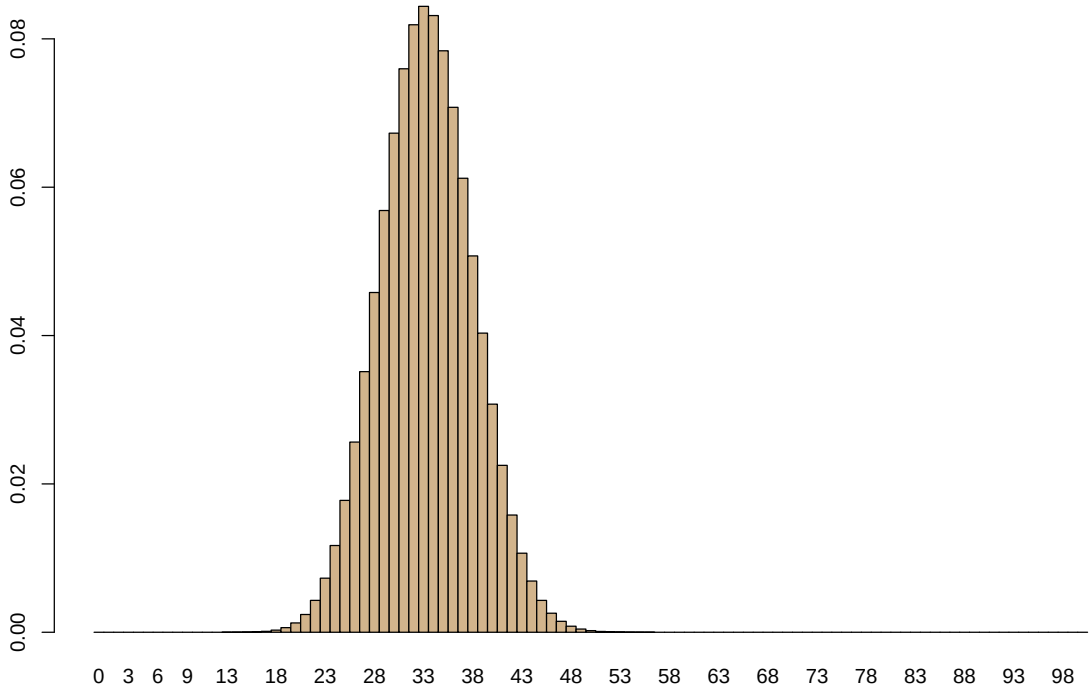
(1) Binomiales con n **pequeño**, como la de la anterior figura. En esos casos usamos la binomial directamente para calcular probabilidades.

(2) Binomiales con n **grande y** p **moderado (ni cerca de 0, ni cerca de 1)**. De estas hablaremos en el resto de este tema.

(3) Binomiales con n **grande y** p **no moderado (cerca de 0 o cerca de 1)**. Hablaremos de ellas más adelante al discutir la *Distribución de Poisson*.

Binomiales con n grande y p moderado.

- Vamos a ver ahora lo que sucede cuando n es grande y mantenemos p moderado (sin acercarlo al 0 o al 1). La siguiente figura muestra un diagrama de barras para la binomial $B(100, 1/3)$:



  Es un diagrama de barras. Pero es evidente que empieza a adivinarse una curva.

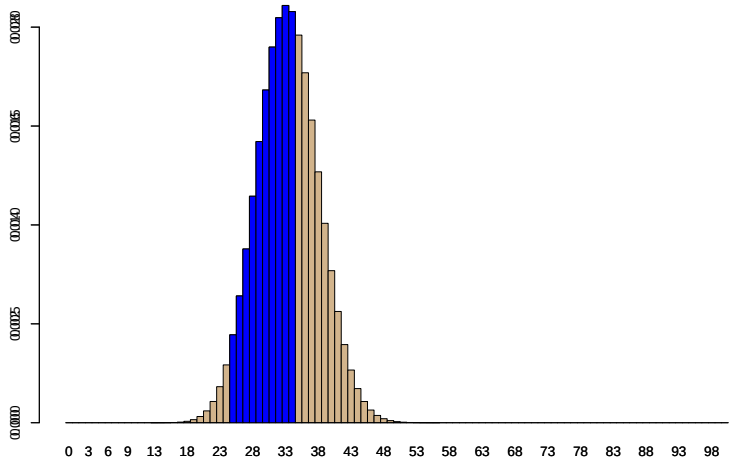
- Y no es una curva cualquiera. Abraham de Moivre descubrió que se trata de la misma curva normal que nos hemos encontrado ya al hablar de la distribución de las medias muestrales. ¿Por qué es útil esa curva?

Cálculos de probabilidad en binomiales con n muy grande.

- La binomial que aparece en la anterior figura es $X \sim B(n = 100, p = 1/3)$. Vamos a suponer que queremos calcular esta probabilidad:

$$P(25 \leq X \leq 35) = P(X = 25) + P(X = 26) + \cdots + P(X = 34) + P(X = 35)$$

Calcular la probabilidad de ese intervalo equivale a calcular el área sombreada.



Para calcular esa suma de términos hay que calcular, por ejemplo, el término:

$$P(X = 29) = \binom{100}{29} \left(\frac{1}{3}\right)^{29} \left(\frac{2}{3}\right)^{71}$$

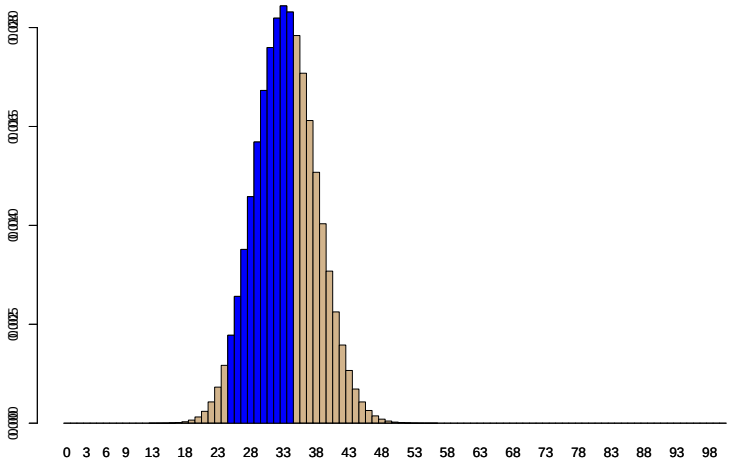
Pero

$$\binom{100}{29} = \frac{100!}{29!\ 71!} = \frac{100 \cdot 99 \cdot 98 \cdots 73 \cdot 72}{29 \cdot 28 \cdot 27 \cdots 2 \cdot 1} = 1917353200780443050763600$$

¡Y esto es solo uno de los términos! La curva normal ofrece una alternativa.

Otra vez la discusión "*discreto frente a continuo*".

- Si volvemos a pensar en la anterior figura del cálculo $P(25 \leq X \leq 35)$ en una $X \sim B(n = 100, p = 1/3))$

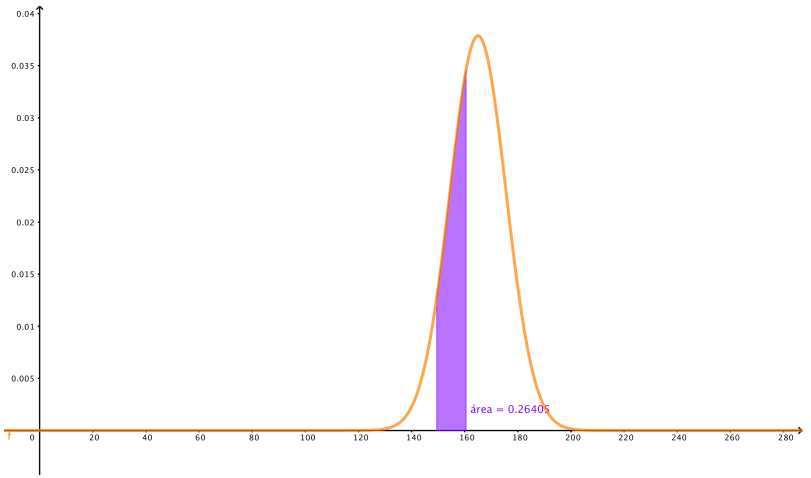


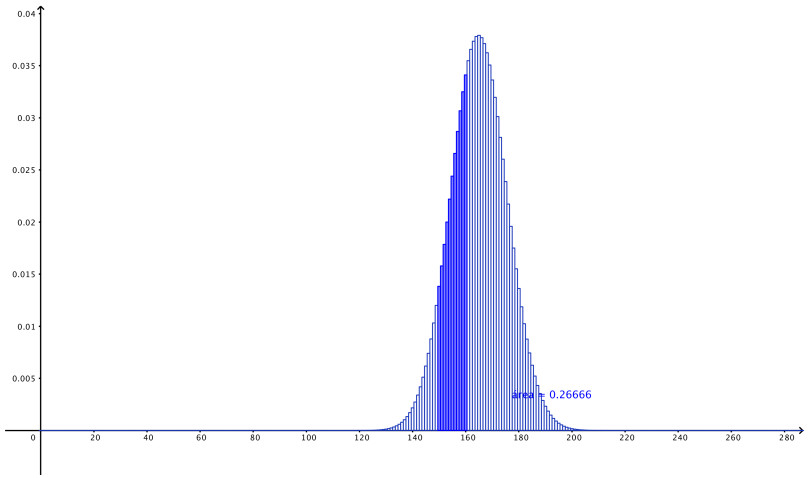
veremos que cada uno de los valores, cada una de las barras que forman ese gráfico, tiene un *peso individual* muy pequeño. Lo que importa es el *área conjunta*. Porque si la variable X puede tomar valores desde 0 hasta 100 entonces en la mayoría de las aplicaciones la diferencia entre $X = 65$ y $X = 66$ será *muy poco relevante*.

- Esta discusión recuerda a la que ya tuvimos al distinguir entre variables discretas y continuas. Cuando una variable toma muchos valores distintos y la diferencia entre valores individuales no es relevante, muchas veces es mejor considerarla continua. **No nos interesa la probabilidad de un valor concreto, sino la de un intervalo**.

Una solución alternativa.

- La curva normal describe muy aproximadamente el perfil de la distribución binomial. Así que para calcular la probabilidad de un intervalo, que es la suma de las áreas de los rectángulos sobre ese intervalo, podemos **aproximarla por el área bajo la curva normal en ese mismo intervalo**.





Las dos figuras muestran las dos formas de trabajar para calcular $P(a \leq X \leq b)$:

- a la izquierda calculamos (de forma exacta) $\sum_{k=a}^{b} P(X = k) = \sum_{k=a}^{b} \binom{n}{k} p^k q^{(n-k)}$.
- a la derecha, si la curva normal es $y = f(x)$, *aproximamos* esa probabilidad mediante

$$P(a \leq X \leq b) \approx \text{área bajo la gráfica de } f = \int_a^b f(x)\,dx.$$

- Si crees que integrar es *complicado*, ¡recuerda cómo son los números combinatorios!

Section 7

Variables aleatorias continuas.

Función de densidad de probabilidad continua.

- Vamos a profundizar en esa idea de usar la integral de una función para calcular la probabilidad de un intervalo. No nos sirve cualquier función, pero basta con que se cumplan dos condiciones.
- Una función $f(x)$ es una **función de densidad continua** si posee estas características:
  - Es no negativa: $f(x) \geq 0$ para todo x; es decir, f no toma valores negativos.
  - El área total bajo la gráfica de f es 1:

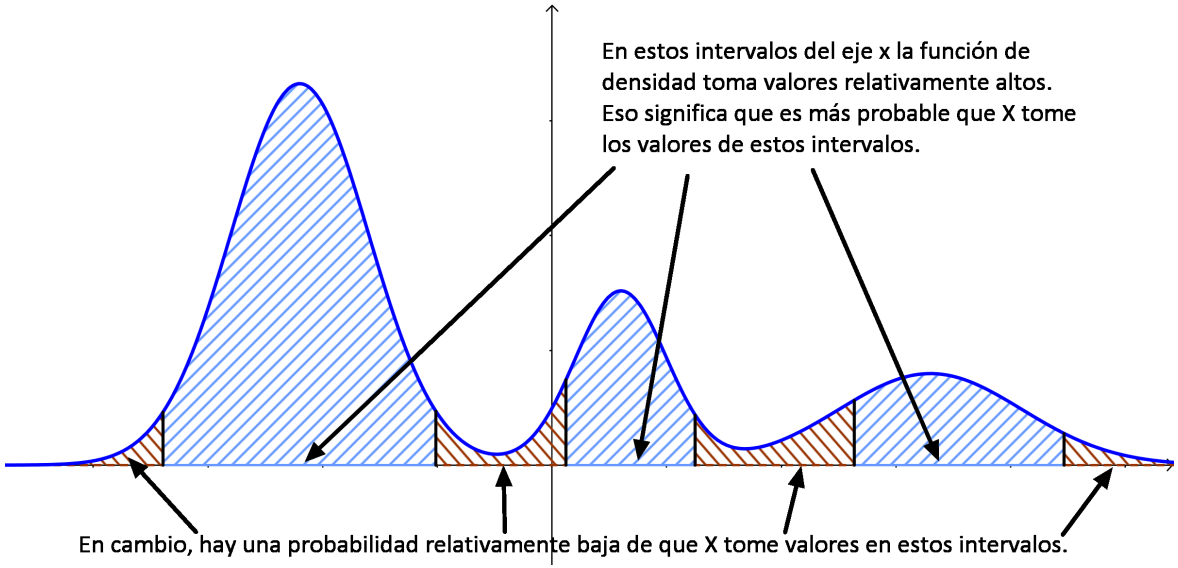
$$\int_{-\infty}^{\infty} f(x)\, dx = 1$$

- Si tenemos una función $f(x)$ con las propiedades que acabamos de ver, entonces diremos que f define una **variable aleatoria continua** X con función de densidad f.
- En tal caso la probabilidad de que el valor de X pertenezca a cualquier intervalo (a, b) se **define así**

$$P(a \leq X \leq b) = \text{área bajo la gráfica de } f = \int_a^b f(x)\, dx.$$

Interpretación de la función de densidad.

- La siguiente figura muestra una función de densidad y la forma de interpretar sus valores. Recuerda que los valores de la función no son probabilidades. Las probabilidades son *áreas*.



  Por eso decimos que una de estas funciones define una *distribución* (una forma de repartir) la probabilidad. Las variables discretas tienen una tabla de valores y probabilidades. Ahora tenemos la función f para hacer el mismo trabajo. Es la versión teórica de las curvas de densidad que aprendimos a dibujar para describir los datos de una muestra.

- Otra observación importante y que al principio resulta paradójica es que sea cual sea x_0 se cumple $P(X = x_0) = 0$.

Media y varianza de una variable aleatoria continua.

- Recuerda que para un variable aleatoria discreta con valores $x_1, \ldots, x_k$ y probabilidades $p_1, \ldots, p_k$ era:

$$\mu = E(X) = \sum_{i=1}^{k} x_i \cdot p_i$$

$$\sigma^2 = Var(X) = \sum_{i=1}^{k} (x_i - \mu)^2 \cdot p_i$$

- Para una variable aleatoria continua con densidad $f(x)$ se tiene:

$$\mu = E(X) = \int_{-\infty}^{\infty} x \cdot f(x)\, dx$$

$$\sigma^2 = Var(X) = \int_{-\infty}^{\infty} (x - \mu)^2 \cdot f(x)\, dx$$

- El paso de discreto a continuo se consigue cambiando el sumatorio por una integral y la probabilidad p_i por el *diferencial de probabilidad* $dp = f(x)\, dx$ (ver la Sección 5.4.2 de (San Segundo and Marvá 2016)).

La distribución uniforme.

- La *distribución uniforme* en el intervalo $[a, b]$ es un ejemplo sencillo pero muy importante de variable aleatoria continua. Se usa cuando ninguna parte del intervalo es más probable que otra del mismo tamaño.
- Su función de densidad es constante en el intervalo $[a, b]$ y vale 0 fuera:

$$f(x) = \begin{cases} \dfrac{1}{b-a} & \text{si } a \leq x \leq b \\ 0 & \text{en otro caso} \end{cases}$$

  A veces se dice que en esta distribución *todos los puntos de* $[a, b]$ *son igual de probables*. Pero en cualquier distribución continua, uniforme o no, la probabilidad de un punto es 0

- La media de la variable uniforme es como cabía esperar $\mu = \dfrac{a+b}{2}$

  y su desviación típica es $\sigma^2 = \dfrac{(b-a)^2}{12}$
- En R usaremos la función `runif` para generar puntos aleatorios con distribución uniforme.
- **Ejercicio:** ejecuta varias veces `runif(10, min = 5, max = 15)` para ver como funciona.

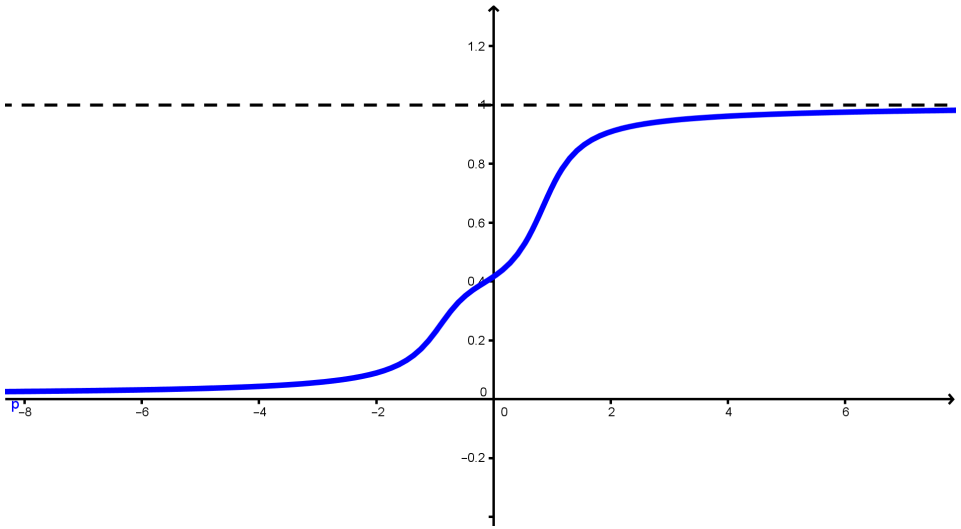
Función de distribución.

- La función de distribución de una variable aleatoria X (¡discreta o continua!) es:

$$F(k) = P(X \leq k)$$

Para una variable continua esto se traduce en $F(k) = \int_{-\infty}^{k} f(x)\,dx$

- Vimos que la gráfica típica de la función de distribución de una variable discreta tiene forma de escalera. Para una variable continua la gráfica típica de la función de distribución es una rampa como esta:



- Lo que hace que F sea a menudo más útil que f es esta **propiedad** :

$$P(a < X < b) = F(b) - F(a)$$

Section 8

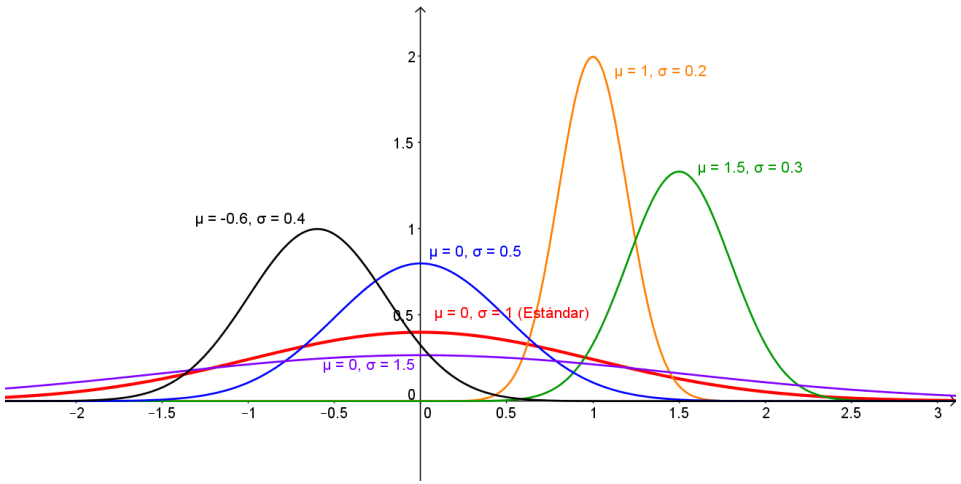
Variables aleatorias normales.

La curva normal.

- Hemos hablado ya varias veces de la curva normal. En realidad hay toda una familia de curvas normales, cuya ecuación es

$$f_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

  Aunque todas tienen forma acampanada, cada elección de valores de μ y σ produce una curva normal distinta: μ determina el centro de la distribución (que es simétrica) y σ controla cómo de estrecha y alta o ancha y baja es la campana. La figura muestra varias curvas normales para varios valores de μ y σ.



- Una variable aleatoria continua X con función de densidad $f_{\mu,\sigma}(x)$ es una **variable normal** y escribiremos $X \sim N(\mu, \sigma)$. La media de la normal $N(\mu, \sigma)$ es μ y su varianza es σ^2 (algunos libros usan $N(\mu, \sigma^2)$, cuidado).

Distribuciones normales en R. La función pnorm.

- La función pnorm permite calcular en R la función de distribución de una variable normal $X \sim N(\mu, \sigma)$.

$$P(X < b) = \texttt{pnorm(b, mean = mu, sd = sigma)}$$

- Si X es de tipo $N(10, 2)$ y queremos calcular la probabilidad de una **cola izquierda** $P(X < 10.5)$ usaríamos

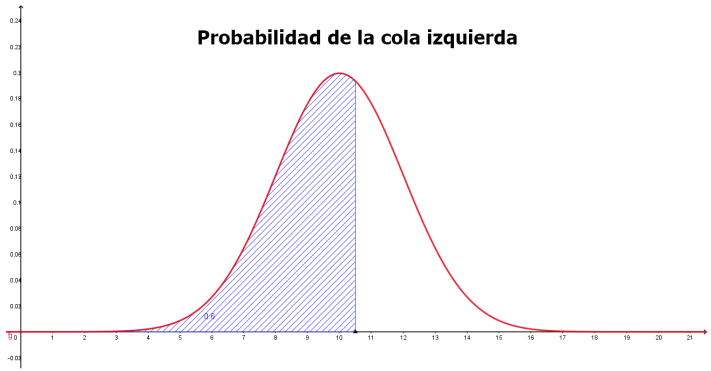
```
pnorm(10.5, mean=10, sd=2)
```

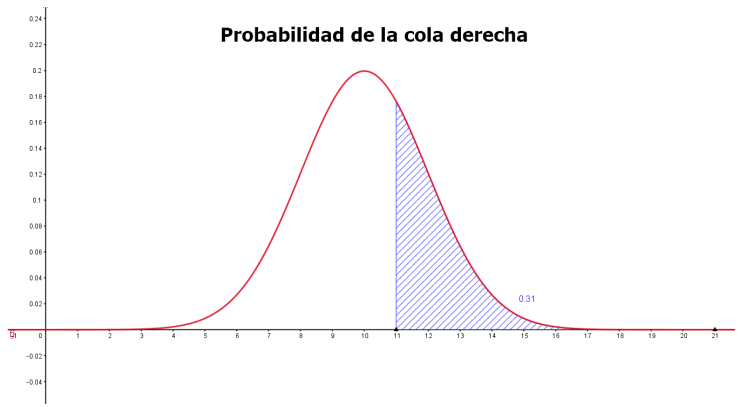
```
## [1] 0.5987063
```

- Si lo que queremos calcular es una **cola derecha** $P(X > 11)$ usaríamos una de estas dos opciones equivalentes:

```
  1 - pnorm(11, mean=10, sd=2)
  pnorm(11, mean = 10, sd = 2, lower.tail = FALSE)
```

```
## [1] 0.3085375
```



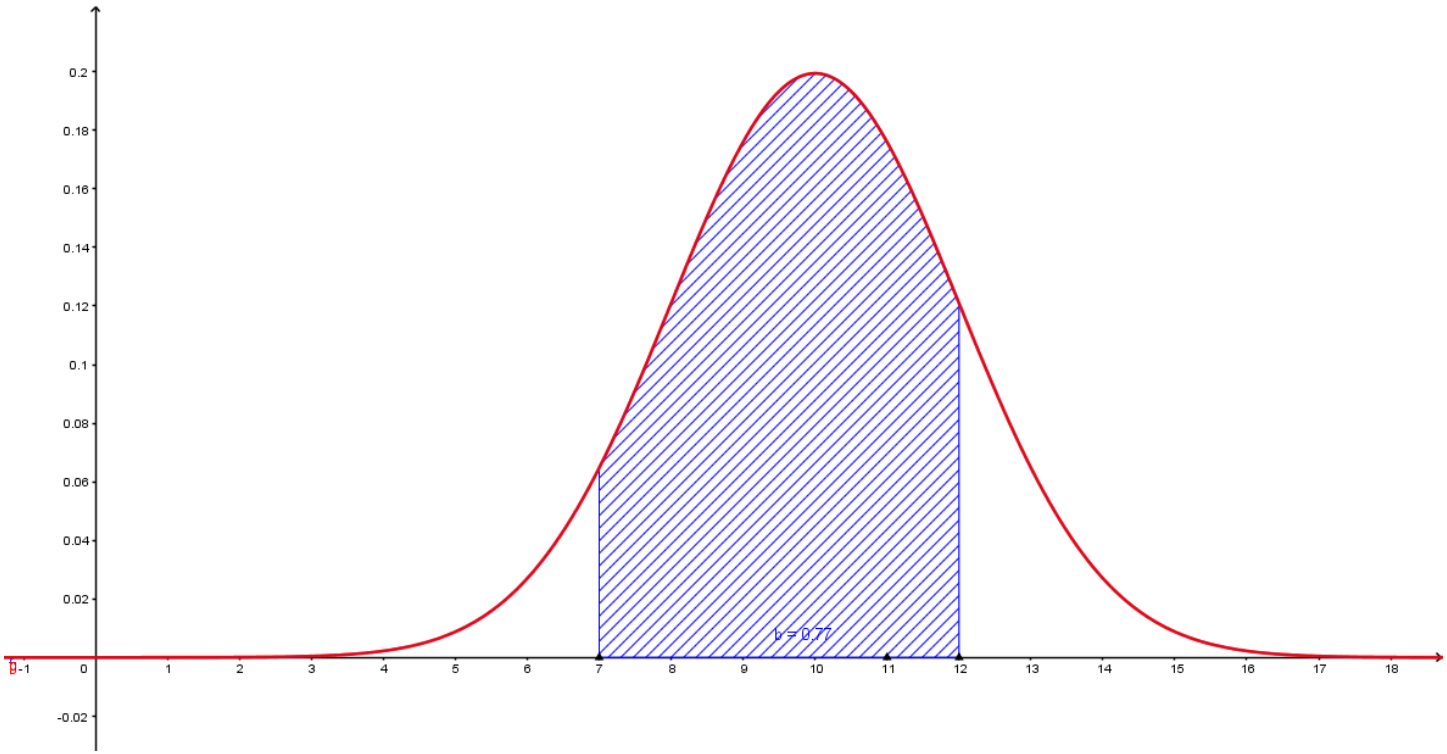


Probabilidad de un intervalo con `pnorm`.

- Si queremos calcular la probabilidad de un **intervalo**, como $P(7 < X < 12)$ lo expresamos como una diferencia:

```
pnorm(12, mean=10, sd=2) - pnorm(7, mean=10, sd=2)
```

```
## [1] 0.7745375
```

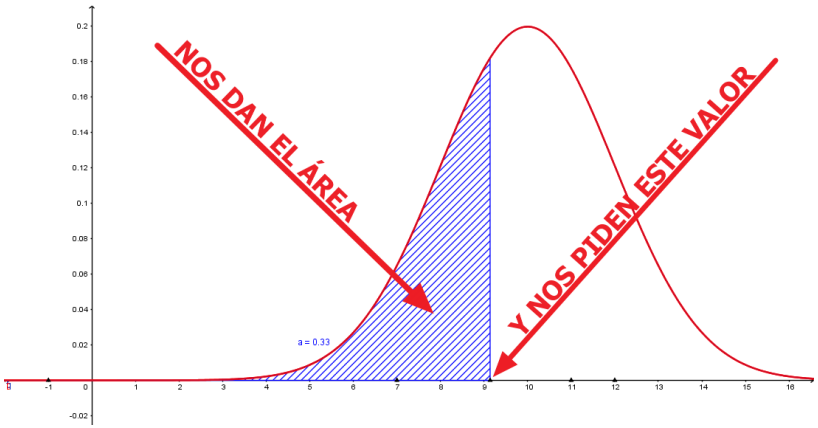


Problema inverso de probabilidad. La función `qnorm`.

- El *problema inverso* de probabilidad es este: dada una probabilidad p ¿cuál es el valor b tal que $P(X < b) = p$? Es un problema *muy importante para la Inferencia Estadística*.
- **Ejemplo** (problema inverso de cola izquierda) dada una distribución normal de tipo $N(10, 2)$ ¿cuál es el valor k para el que se cumple $P(X \leq k) = \frac{1}{3}$? En R lo obtenemos así:

```
qnorm(p = 1/3, mean = 10, sd=2)
```

```
## [1] 9.138545
```



- **Ejercicio importante:** dada una normal $N(0, 1)$, ¿cuál es el valor k para el que se cumple $P(X \geq k) = 0.025$? Volveremos a encontrar esta pregunta cuando hablemos de intervalos de confianza.

Otras funciones para trabajar con normales: `rnorm` y `dnorm`.

- La función `rnorm` es *muy útil para simulaciones*. Sirve para fabricar una muestra con n valores de una variable $X \sim N(\mu, \sigma)$ mediante:

```
muestra = rnorm(n, mean = mu, sd = sigma)
```

  ¡Atención! `mean(muestra)` no es `mu` y `sd(muestra)` no es `sigma`. ¿Ves por qué? Cuando queramos conseguir eso usaremos la función `mvrnorm` de la librería `MASS`.

- **Ejercicio:** genera vectores `x1` e `y1` cada uno con 1000 valores de una normal $N(0, 1)$. Luego ejecuta este código.

```
ggplot(data.frame(x1, y1)) +
  geom_point(mapping = aes(x1, y1), col="red")
```

  Ahora genera `x2` e `y2` cada uno con 1000 valores de una distribución uniforme en $N(0, 1)$ y ejecuta ese código cambiando `x1` e `y1` por `x2` e `y2`. ¿Ves la diferencia?

- La función `dnorm` es la función de densidad de la variable normal. Es decir, su valor es la altura de la curva normal y *no se debe interpretar directamente en términos de probabilidad*. Sirve casi exclusivamente para dibujar esa curva.

- Cuando conozcamos otras distribuciones verás que para todas ellas existen funciones similares a estas. Por ejemplo, para la distribución exponencial existen `pexp`, `qexp`, `rexp`, `dexp`. El sufijo `exp` identifica la distribución y el prefijo `p`, `q`, etc. identifica la función.

Tipificación y normal estándar Z.

- **Tipificación:** Si X es una variable aleatoria normal de tipo $N(\mu, \sigma)$, entonces la variable que se obtiene mediante la transformación de tipificación:

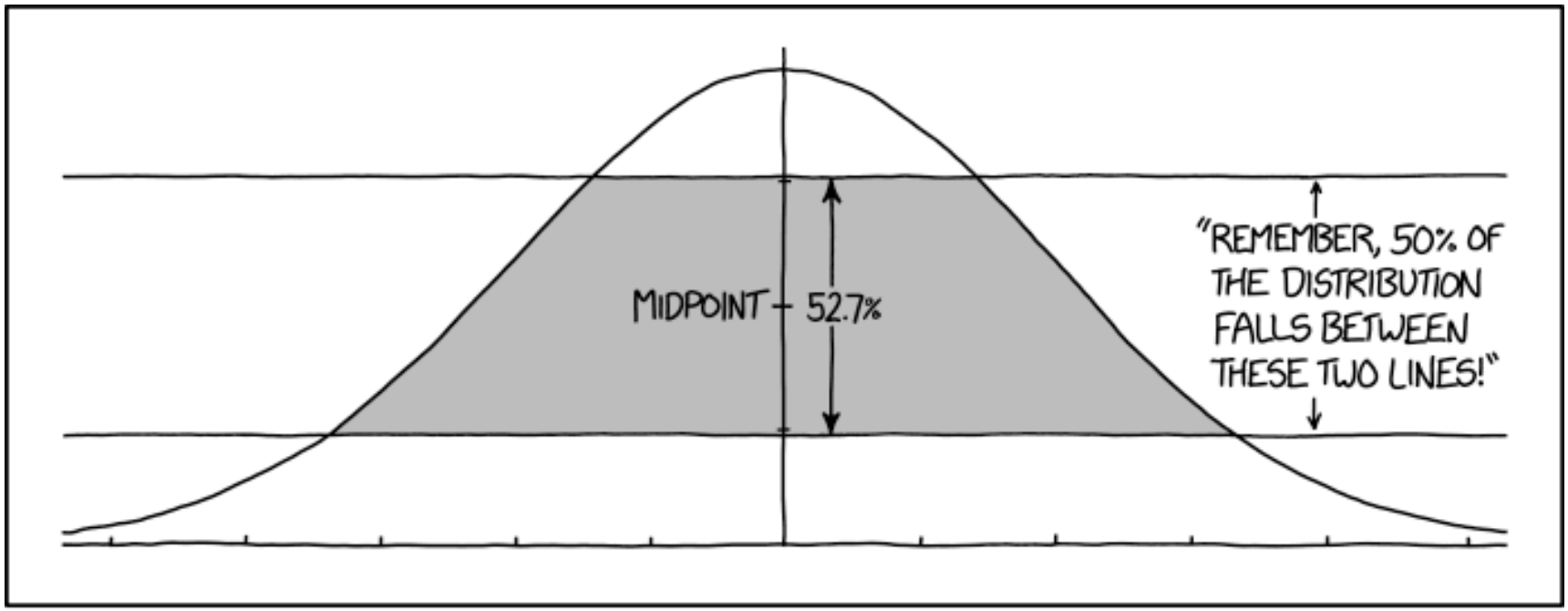
$$Z = \frac{X - \mu}{\sigma}$$

es una variable normal de tipo $N(0, 1)$, la **normal estándar** a la que siempre llamaremos Z. La tipificación permite reducir cualquier observación de una normal $N(\mu, \sigma)$ a una *escala universal* que nos proporciona la distribución Z.

- **Regla 68 - 95 - 99.** Una consecuencia de lo anterior es que si X es una variable normal de tipo $N(\mu, \sigma)$ entonces **siempre** se cumplen estas aproximaciones:

$$\begin{cases} P(\mu - \sigma < X < \mu + \sigma) \approx 0.683, \\ P(\mu - 2\sigma < X < \mu + 2\sigma) \approx 0.955 \\ P(\mu - 3\sigma < X < \mu + 3\sigma) \approx 0.997 \end{cases}$$

- **Ejercicio:**
(a) comprueba estos resultados para, por ejemplo, la normal $N(0, 1)$ y la normal $N(40, 3.6)$.
(b) Tenemos una variable $X \sim N(123, 17)$ y observamos el valor 168. ¿Como de *raro* es este valor? Tipifícalo para responder.
(c) Ejecuta `scale(168, center = 123, scale = 17)`



HOW TO ANNOY A STATISTICIAN

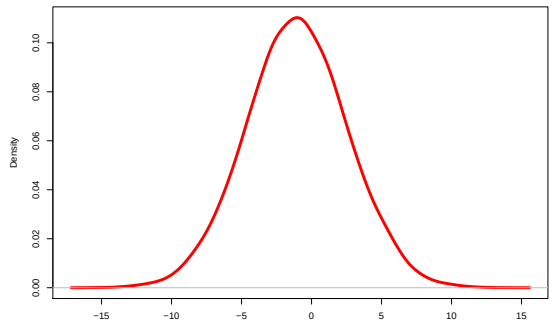
Suma (y mezcla) de normales independientes.

- Si $X_1 \sim N(\mu_1, \sigma_1)$ y $X_2 \sim N(\mu_2, \sigma_2)$ son variables **normales independientes**, su suma es **de nuevo una variable normal** de tipo

$$N\left(\mu_1 + \mu_2, \sqrt{\sigma_1^2 + \sigma_2^2}\right).$$

Insistimos, la novedad es que *la suma de dos normales independientes sigue siendo normal*. Ejecuta este código para ver un ejemplo

```
set.seed(2019)
pob1 = rnorm(30000, mean = -3, sd = 1)
pob2 = rnorm(30000, mean = 2, sd = 0.5)
pobSuma = 3 * pob1 + 4 * pob2
plot(density(pobSuma, adjust = 1.6), main="", lwd=5, col="red", xlab="")
```



Este resultado se generaliza a la suma de k variables normales independientes, que dan como resultado una normal de tipo $N\left(\mu_1 + \cdots + \mu_k, \sqrt{\sigma_1^2 + \cdots + \sigma_k^2}\right)$.

La **mezcla** de variables normales es un proceso completamente distinto. A menudo da como resultado distribuciones bimodales o multimodales.

Section 9

Complementos de R: Operaciones con factores, verbos de dplyr.

Operaciones básicas con factores.

- Podemos crear un factor a partir de un vector de strings con la función `factor`:

```
(ardeida = factor(c("martinete", "garzaReal", "avetorillo", "garzaReal",
                    "cangrejera", "martinete", "martinete"), ))
```

```
## [1] martinete  garzaReal  avetorillo garzaReal  cangrejera martinete
## [7] martinete
## Levels: avetorillo cangrejera garzaReal martinete
```

Dos detalles sobre la salida: fíjate en la ausencia de comillas y en que el orden de los niveles es alfabético. Si quieres otro orden (por ejemplo para las tablas de frecuencia) puedes hacerlo explícito:

```
(ardeida = factor(c("martinete", "garzaReal", "avetorillo", "garzaReal",
                    "cangrejera", "martinete", "martinete"),
levels = c("garzaReal", "martinete",  "cangrejera", "avetorillo")))
```

```
## [1] martinete  garzaReal  avetorillo garzaReal  cangrejera martinete
## [7] martinete
## Levels: garzaReal martinete cangrejera avetorillo
```

- El factor puede ser *ordenado* si además incorporamos la opción `ordered = TRUE`. *No se debe confundir con el uso de levels* para fijar un orden "estético" de los niveles. En un factor ordenado el orden aporta información *relevante* sobre los niveles.

Más funciones que generan factores

- Ya hemos visto que el resultado de `cut` es un factor ordenado cuyos niveles son los intervalos en que se divide el recorrido de la variable.
- La función `gl` sirve para generar factores a medida y es un complemento para otras funciones como `rep`. Un ejemplo en el que fabricamos un factor con tres niveles y 4 repeticiones:

```
gl(n = 3, k = 4, labels = c("piedra", "papel", "tijera"))
```

```
##  [1] piedra piedra piedra piedra papel  papel  papel  papel  tijera
## [10] tijera tijera tijera
## Levels: piedra papel tijera
```

- A veces, por cuestiones de diseño del experimento o del conjunto de datos, querremos que los niveles del factor aparezcan intercalados.

```
gl(n = 3, k=1, length = 30, labels = c("piedra", "papel", "tijera"))
```

```
##  [1] piedra papel  tijera piedra papel  tijera piedra papel  tijera
## [10] piedra papel  tijera piedra papel  tijera piedra papel  tijera
## [19] piedra papel  tijera piedra papel  tijera piedra papel  tijera
## [28] piedra papel  tijera
## Levels: piedra papel tijera
```

- Puedes consultar el Capítulo 7 de (Boehmke 2016), el Capítulo 12 (15 en la versión online) de (Wickham and Grolemund 2016), o el Capítulo 6 de (Matloff 2011).

Matrices

- Para trabajar sobre un ejemplo concreto, vamos a colocar los números del 1 al 36 en una matriz de R, llamada `M`, de 4 filas y 9 columnas (diremos que es una matriz 4×9). El código es este:

```
(M = matrix(1:36, nrow=4) )
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
## [1,]    1    5    9   13   17   21   25   29   33
## [2,]    2    6   10   14   18   22   26   30   34
## [3,]    3    7   11   15   19   23   27   31   35
## [4,]    4    8   12   16   20   24   28   32   36
```

Fíjate en que R rellena la matriz columna por columna. Para rellenar por filas:

```
(M = matrix(1:36, nrow=4, byrow = TRUE) )
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
## [1,]    1    2    3    4    5    6    7    8    9
## [2,]   10   11   12   13   14   15   16   17   18
## [3,]   19   20   21   22   23   24   25   26   27
## [4,]   28   29   30   31   32   33   34   35   36
```

- Podemos usar `dim` para cambiar las dimensiones de la matriz:

```
dim(M) = c(3, 12)
M
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
## [1,]    1   28   20   12    4   31   23   15    7    34    26    18
## [2,]   10    2   29   21   13    5   32   24   16     8    35    27
## [3,]   19   11    3   30   22   14    6   33   25    17     9    36
```

Operaciones con matrices.

- Podemos convertir matrices en vectores aplicándoles la función `c`. Esta operación es útil para modificar el orden de los elementos de un vector (especialmente de uno que no hemos generado nosotros). Por ejemplo, dado este vector:

```
v = c(1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3)
```

  Si queremos intercalar los unos y doses hacemos:

```
Mv = matrix(v, nrow=3, byrow = TRUE)
(v = c(Mv))
```

```
##  [1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```

- **Funciones matriciales**. Muchas funciones actúan sobre las matrices vectorialmente (elemento a elemento). Pero hay funciones que tienen en cuenta la estructura matricial. La función `t` traspone matrices (intercambia filas y columnas). Las funciones `rowSums` y `colMeans` calculan sumas o medias por filas o columnas como indican sus nombres. Las funciones `cbind` y `rbind` unen (pegan) matrices compatibles por filas o columnas.
- Cuando veamos funciones de la familia `apply` volveremos sobre esto. Y si alguna vez necesitas Álgebra Matricial, el Capítulo 12 de (Boehmke 2016) o el libro (Braun and Murdoch 2016) pueden ser útiles.

Verbos de dplyr

- Esta sección pretende ser una invitación a la lectura del Capítulo 5 de (Wickham and Grolemund 2016) y desde luego no aspira a sustituir esa lectura.
- Aunque ya hemos visto algunos ejemplos de `dplyr` en acción, vamos a recopilar aquí de forma más sistemática, los elementos básicos de la transformación de datos con esa librería. En esencia, la mayoría de las operaciones se organizan en torno a una familia de verbos. Los principales son:
  - `select`
  - `filter`
  - `mutate`
  - `arrange`
  - `summarize`
  - `group_by` En las próximas páginas vamos a ver ejemplos de uso de estos verbos. Los tres primeros han aparecido ya, así que nos detendremos un poco más en los nuevos.

select para elegir columnas

- En esta y en las siguientes páginas vamos a usar la tabla 'gapminder, así que empezamos cargándola. Además vamos a ver los nombres de las variables que la componen:

```
library(gapminder)
names(gapminder)
```

```
## [1] "country"   "continent" "year"      "lifeExp"   "pop"
## [6] "gdpPercap"
```

- Ahora vamos a usar select para elegir las columnas de lifeExp y gdpPercap.

```
gapminder %>%
  select(lifeExp, gdpPercap) %>%
  head(3)
```

```
## # A tibble: 3 x 2
##   lifeExp gdpPercap
##     <dbl>     <dbl>
## 1    28.8      779.
## 2    30.3      821.
## 3    32.0      853.
```

Fíjate en que hemos usado head para ver los primeros elementos de la tabla.

Otras posibilidades de `select`

- De la misma forma que `12:20` representa un conjunto consecutivo de números podemos usar `:` para seleccionar un conjunto consecutivo de columnas *por sus nombres*. Y si usamos `-` estaremos excluyendo una columna:

```
gapminder %>%
  select(continent:pop, -year) %>%
  names()
```

```
## [1] "continent" "lifeExp"   "pop"
```

Asegúrate de que entiendes por qué se incluyen específicamente esas columnas.

- Además podemos usar una serie de funciones auxiliares que permiten elegir las columnas cuyos nombres cumplan cierto patrón. Esas funciones incluyen: `contain`, `starts_with`, `ends_with`, `matches`, `one_of` (pero hay más). Por ejemplo:

```
gapminder %>%
  select(starts_with("c")) %>%
    names()
```

```
## [1] "country"   "continent"
```

Este tipo de funciones auxiliares son muy útiles cuando estemos *limpiando conjuntos sucios* de datos antes del análisis.

`filter` para elegir filas.

- La función `filter` realiza selección por filas en una tabla. Por ejemplo, para ver las observaciones correspondientes a España:

```
gapminder %>%
filter(country == 'Spain') %>%
head(4)
```

```
## # A tibble: 4 x 6
##   country continent  year lifeExp      pop gdpPercap
##   <fct>   <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Spain   Europe     1952    64.9 28549870     3834.
## 2 Spain   Europe     1957    66.7 29841614     4565.
## 3 Spain   Europe     1962    69.7 31158061     5694.
## 4 Spain   Europe     1967    71.4 32850275     7994.
```

- Además de `filter` existen otras funciones que permiten seleccionar por filas. Por ejemplo aquí usamos `top_n` (mira la chuleta de dplyr para ver más posibilidades):

```
gapminder %>%
filter(year == "1997") %>%
top_n(3, gdpPercap)
```

```
## # A tibble: 3 x 6
##   country       continent  year lifeExp      pop gdpPercap
##   <fct>         <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Kuwait        Asia       1997    76.2  1765345    40301.
## 2 Norway        Europe     1997    78.3  4405672    41283.
## 3 United States Americas   1997    76.8 272911760    35767.
```

mutate para crear nuevas variables.

- Usemos mutate para añadir una columna que calcule el gdp (en millones de dolares) multiplicando pop por gdpPercap. Aprovechamos para usarsample_n, emparentada confilter‘:

```
gapminder %>%
  mutate(gdp = pop * gdpPercap / 10^6) %>%
  filter(year == 1982) %>%
  sample_n(4)
```

```
## # A tibble: 4 x 7
##   country     continent  year lifeExp       pop gdpPercap     gdp
##   <fct>       <fct>     <int>   <dbl>     <int>     <dbl>   <dbl>
## 1 Algeria     Africa     1982    61.4  20033753     5745. 115097.
## 2 New Zealand Oceania    1982    73.8   3210650    17632.  56611.
## 3 Cuba        Americas   1982    73.7   9789224     7317.  71627.
## 4 India       Asia       1982    56.6 708000000      856. 605852.
```

- Hay otras funciones relacionadas con mutate, como add_column, rename, etc.
- Si quieres aplicar una función a todos los elementos de una columna puedes usar mutate_at. Por ejemplo, para calcular el logaritmo en base 10 del gdp, ejecuta:

```
gapminder %>%
  mutate(gdp = pop * gdpPercap / 10^6) %>%
  mutate_at("gdp", log10) %>%
  head(4)
```

summarize y group_by para describir los datos

- Vamos a ver como usar summarize para explorar nuestros datos. En un primer ejemplo sencillo vamos a calcular la longitud media de los pétalos en la tabla iris:

```
iris %>%
  summarise(mediana = median(Petal.Length), desvMediana = mad(Petal.Length))
```

```
##   mediana desvMediana
## 1    4.35     1.85325
```

Por cierto ¿como harías esto con R básico? Busca información sobre la familia de funciones apply de R (por ejemplo en el Capítulo 21 de (Wickham and Grolemund 2016), o las Secciones 3.3 y 4.4 de (Matloff 2011).)

- Eso está bien, pero sabemos que iris contiene datos de tres especies y lo natural es preguntar si hay diferencias *significativas* (volveremos pronto sobre esa palabra) entre las longitudes de los pétalos de cada una de esas especies. Así que queremos calcular las medias por especie, que son *medias agrupadas*. Ahí es donde interviene group_by:

```
iris %>%
  group_by(Species) %>%
  summarise(mediana = median(Petal.Length), desvMediana = mad(Petal.Length))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 3 x 3
##   Species    mediana desvMediana
##   <fct>        <dbl>       <dbl>
## 1 setosa         1.5       0.148
## 2 versicolor     4.35      0.519
## 3 virginica      5.55      0.667
```

Grupos con más de un factor.

- En el ejemplo anterior hemos agrupado las observaciones de la tabla `iris` usando únicamente el factor `Species`. Pero no es necesario limitarse a un único factor. Por ejemplo en la tabla `mpg`

```
   mpg %>%
     group_by(manufacturer, cyl) %>%
     summarise(urbano = mean(cty), n = n()) %>%
 head(8)
```

```
## `summarise()` regrouping output by 'manufacturer' (override with `.groups` argument)
```

```
## # A tibble: 8 x 4
## # Groups:   manufacturer [3]
##   manufacturer   cyl urbano     n
##   <chr>        <int>  <dbl> <int>
## 1 audi             4   19.1     8
## 2 audi             6   16.4     9
## 3 audi             8   16       1
## 4 chevrolet        4   20.5     2
## 5 chevrolet        6   17.7     3
## 6 chevrolet        8   13.6    14
## 7 dodge            4   18       1
## 8 dodge            6   15      15
```

- **Ejercicio:** ¿qué cambia si usas el orden inverso `group_by(manufacturer, cyl)` en el anterior código?
- **Ejercicio:** piensa qué hace la función `n()` en este código ()

Funciones que podemos usar con `summarize`

- Para que podamos usar una función dentro de `summarize` tiene que ser una función vectorial (que actúa sobre una columna de la tabla, vista como vector) cuyo resultado sea un valor simple (como un número o un booleano). Te recomendamos consultar la discusión de la Sección 5.6.4 de (Wickham and Grolemund 2016).
- Una de las funciones más útiles de ese tipo es la función `count`. Fíjate en el resultado de este código y compáralo con el anterior.

```
mpg %>%
  group_by(manufacturer) %>%
  count(cyl) %>%
  head(8)
```

```
## # A tibble: 8 x 3
## # Groups:   manufacturer [3]
##   manufacturer   cyl     n
##   <chr>        <int> <int>
## 1 audi             4     8
## 2 audi             6     9
## 3 audi             8     1
## 4 chevrolet        4     2
## 5 chevrolet        6     3
## 6 chevrolet        8    14
## 7 dodge            4     1
## 8 dodge            6    15
```

Observa en particular que no hemos necesitado agrupar por `cyl`.

- La Sección 5.7.1. de (Wickham and Grolemund 2016) describe otras operaciones interesantes que podemos hacer usando `group_by`.

Referencias para la sesión

Enlaces

- Código de esta sesión

Bibliografía

Boehmke, B. C. (2016). *Data Wrangling with R* (p. 508). Springer. https://doi.org/10.1007/978-3-319-45599-0

Braun, J., & Murdoch, D. J. (2016). *A first course in statistical programming with R, 2nd ed.* (p. 215). Cambridge University Press. https://doi.org/10.1017/CBO9781316451090

Matloff, N. S. (2011). *The art of R programming : tour of statistical software design* (p. 373). No Starch Press. https://doi.org/10.1080/09332480.2012.685374

San Segundo, F., & Marvá, M. (2016). *PostData 1.0.* (p. 616). Lulu.com. http://www.lulu.com/shop/fernando-san-segundo-and-marcos-marv%7B/'%7Ba%7D%7D/postdata-10/paperback/product-22855863.html

Wickham, H., & Grolemund, G. (2016). *R for data science: import, tidy, transform, visualize, and model data*. O'Reilly Media, Inc.