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

Tema 7: Modelos. BORRADOR

Fernando San Segundo

Curso 2019-20. Última actualización: 2019-09-29

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI
DEPARTAMENTO DE MATEMÁTICA APLICADA

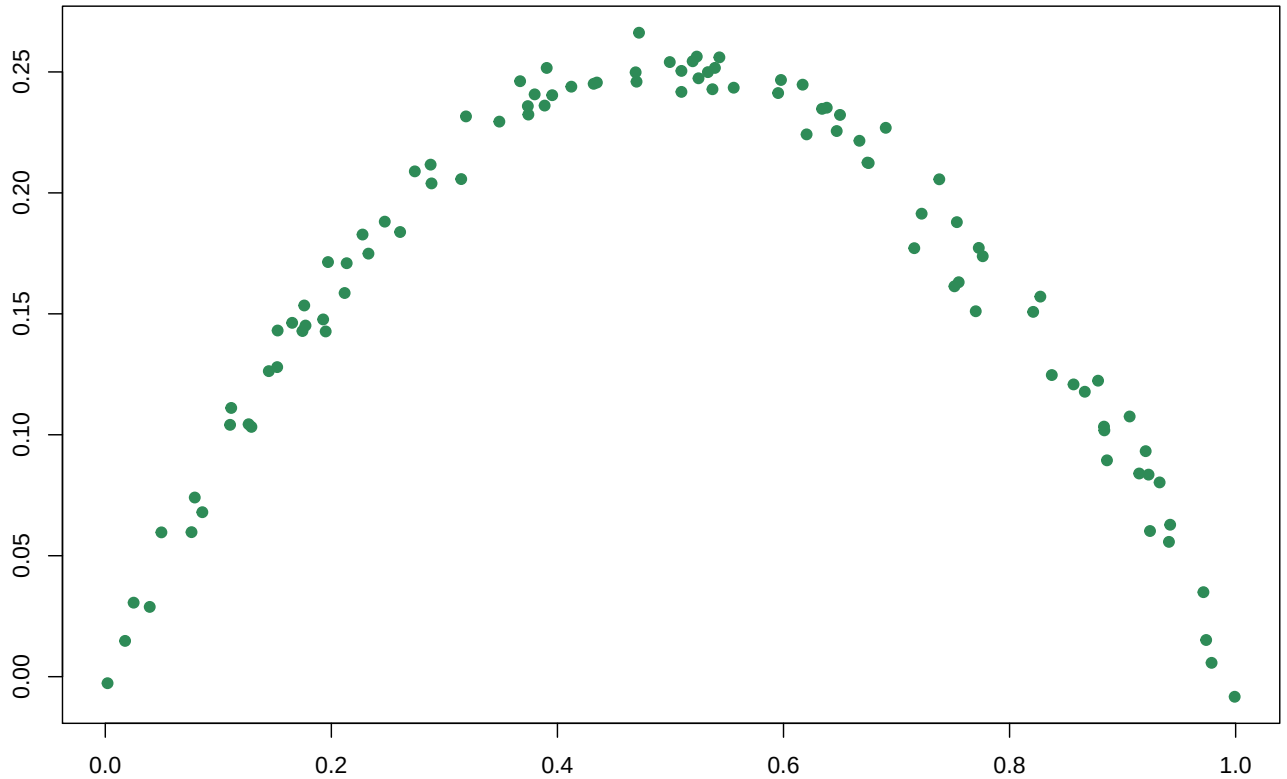
1. Extendiendo el modelo lineal.
2. Modelos lineales con factores. Anova.
3. Modelos lineales generalizados (glm). Regresión Logística.
4. Complementos de R.

Sección 1

Extendiendo el modelo lineal.

Regresión simple con una variable explicativa más allá de las rectas.

- Recuerda que antes hemos visto un conjunto de datos en el que el ajuste lineal mediante una recta resulta inadecuado.

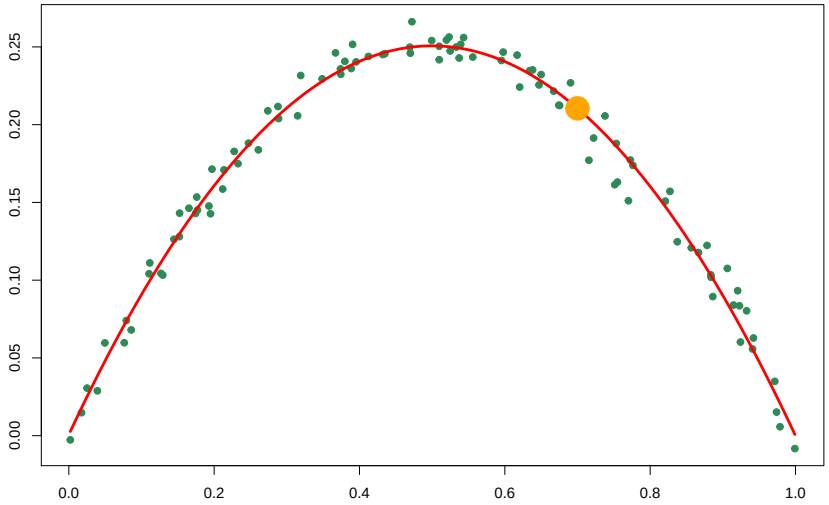


Lo razonable en un caso como este es ajustar una parábola o una curva simular a los datos. En R es muy fácil ajustar curvas de grado más alto con `lm`. Por ejemplo, para una parábola (polinomio de grado 2):

```
modeloParabola = lm(y ~ poly(x, 2))
```

El modelo ya no es una recta.

- Añadimos esa parábola al diagrama de dispersión para que quede claro que ya no estamos ajustando rectas.



Aunque no veamos la ecuación de la parábola, todo funciona casi igual. Por ejemplo podemos predecir valores con `predict`; el punto naranja que aparece destacado corresponde a $x = 0.7$ y el valor de y correspondiente se ha obtenido con (ver código):

```
predict(modeloParabola, newdata = data.frame(x = 0.7))
```

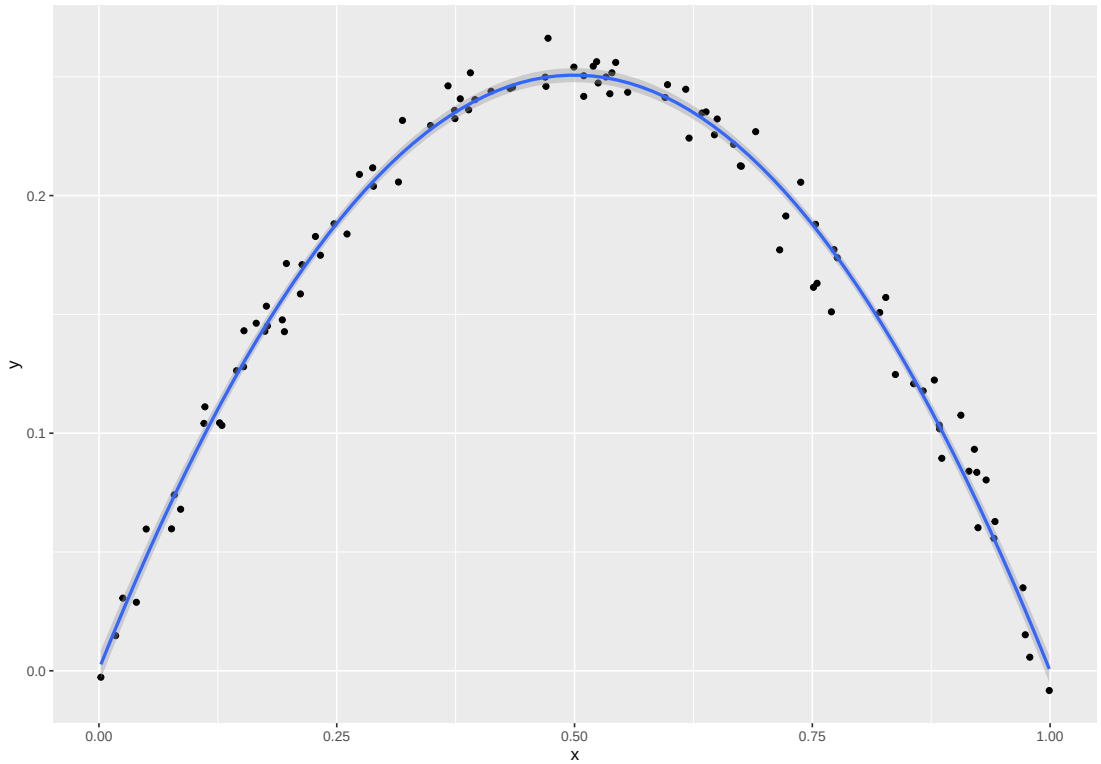
Advertencia: a veces tenemos la tentación de usar un polinomio de grado tres, cuatro, etc. para tratar de *ajustar mejor* los datos. El riesgo de sobreajuste (overfitting) debe estar siempre presente en nuestro trabajo.

Nota: Volviendo sobre la ecuación de la parábola, la interpretación de los coeficientes del modelo no es ahora tan sencilla como en la recta. Por razones técnicas sobre las que volveremos después, R usa *polinomios ortogonales* para hacer el ajuste. Si en el futuro usas mucho la regresión polinómica tendrás que entrar en estos detalles técnicos.

Dibujo con ggplot.

- Se puede obtener un dibujo similar con ggplot así:

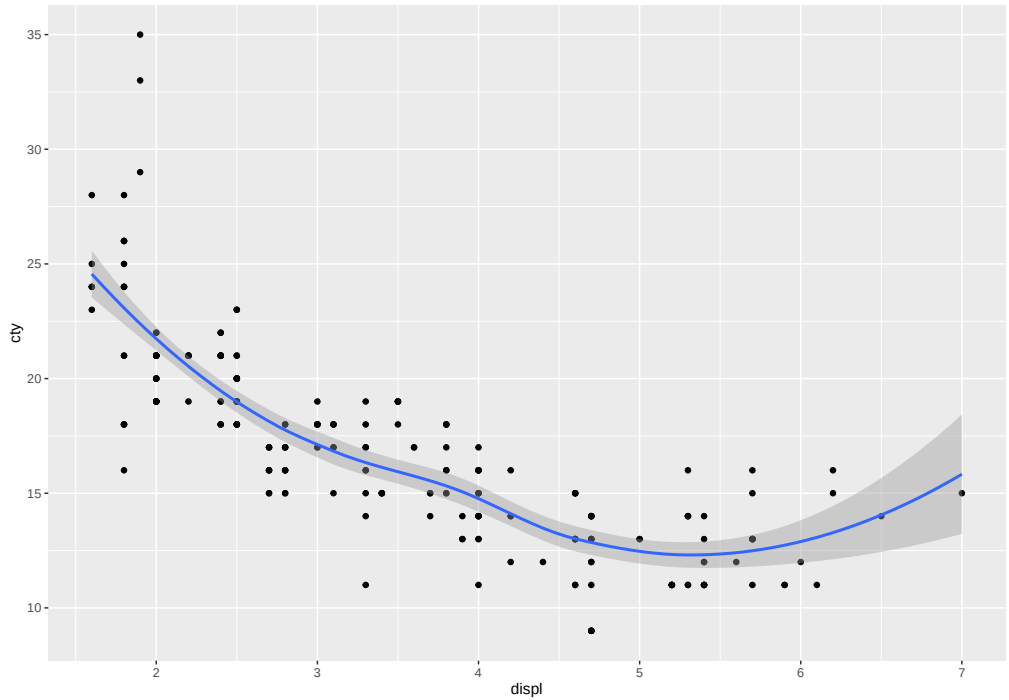
```
library(tidyverse)
datos = data.frame(x, y)
ggplot(datos) +
  geom_point(aes(x, y)) +
  geom_smooth(aes(x, y), method="lm", formula = y ~ poly(x, 2))
```



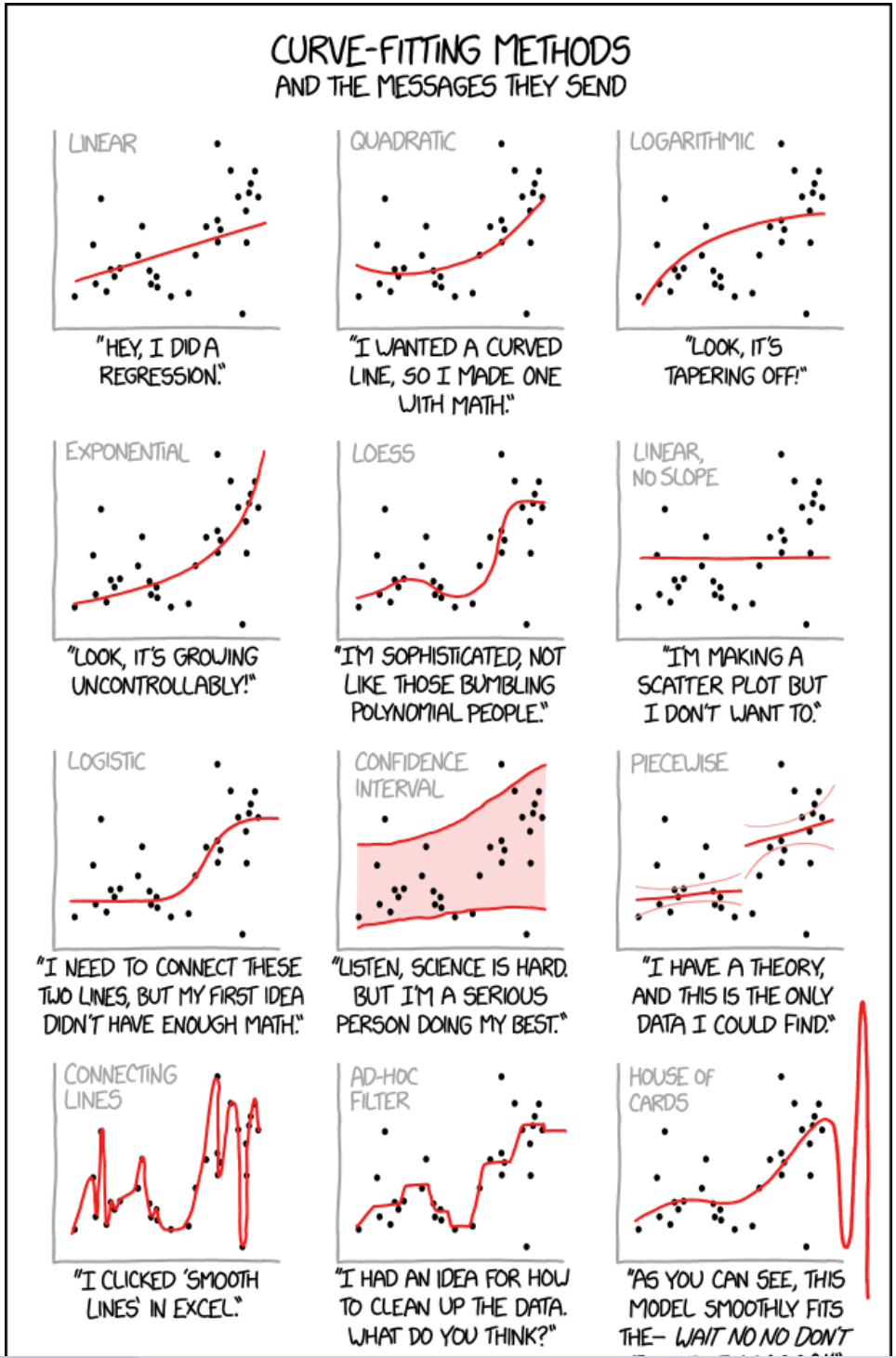
Ajuste de curvas exponenciales, logarítmicas, etc.

- El tipo de curvas que se pueden usar no se agota en los polinomios, desde luego. A veces la curva que mejor se ajusta a unos datos es una exponencial, un logaritmo, etc. Exsiten también las llamadas técnicas de *ajuste local* (loess, ver Wikipedia), que es el que utiliza por defecto `ggplot` en `geom_smooth` para dibujar curvas de tendencia y sus bandas de confianza, como en esta figura.

```
ggplot(mpg) +
  geom_point(aes(displ, cty)) +
  geom_smooth(aes(displ, cty))
```



Siempre hay que ejercer el sentido común a la hora de ajsutar curvas a los datos, para no caer en alguna de las situaciones sobre las que ironiza XKCD en las viñetas de la próxima página.


CURVE-FITTING METHODS
AND THE MESSAGES THEY SEND
LINEAR
"HEY, I DID A REGRESSION."
QUADRATIC
"I WANTED A CURVED LINE, SO I MADE ONE WITH MATH."
LOGARITHMIC
"LOOK, IT'S TAPERING OFF!"
EXPONENTIAL
"LOOK, IT'S GROWING UNCONTROLLABLY!"
LOESS
"I'M SOPHISTICATED, NOT LIKE THOSE BUMBLING POLYNOMIAL PEOPLE."
LINEAR, NO SLOPE
"I'M MAKING A SCATTER PLOT BUT I DON'T WANT TO."
LOGISTIC
"I NEED TO CONNECT THESE TWO LINES, BUT MY FIRST IDEA DIDN'T HAVE ENOUGH MATH."
CONFIDENCE INTERVAL
"LISTEN, SCIENCE IS HARD. BUT I'M A SERIOUS PERSON DOING MY BEST."
PIECEWISE
"I HAVE A THEORY, AND THIS IS THE ONLY DATA I COULD FIND."
CONNECTING LINES
"I CLICKED 'SMOOTH LINES' IN EXCEL."
AD-HOC FILTER
"I HAD AN IDEA FOR HOW TO CLEAN UP THE DATA. WHAT DO YOU THINK?"
HOUSE OF CARDS
"AS YOU CAN SEE, THIS MODEL SMOOTHLY FITS THE– WAIT NO NO DON'T

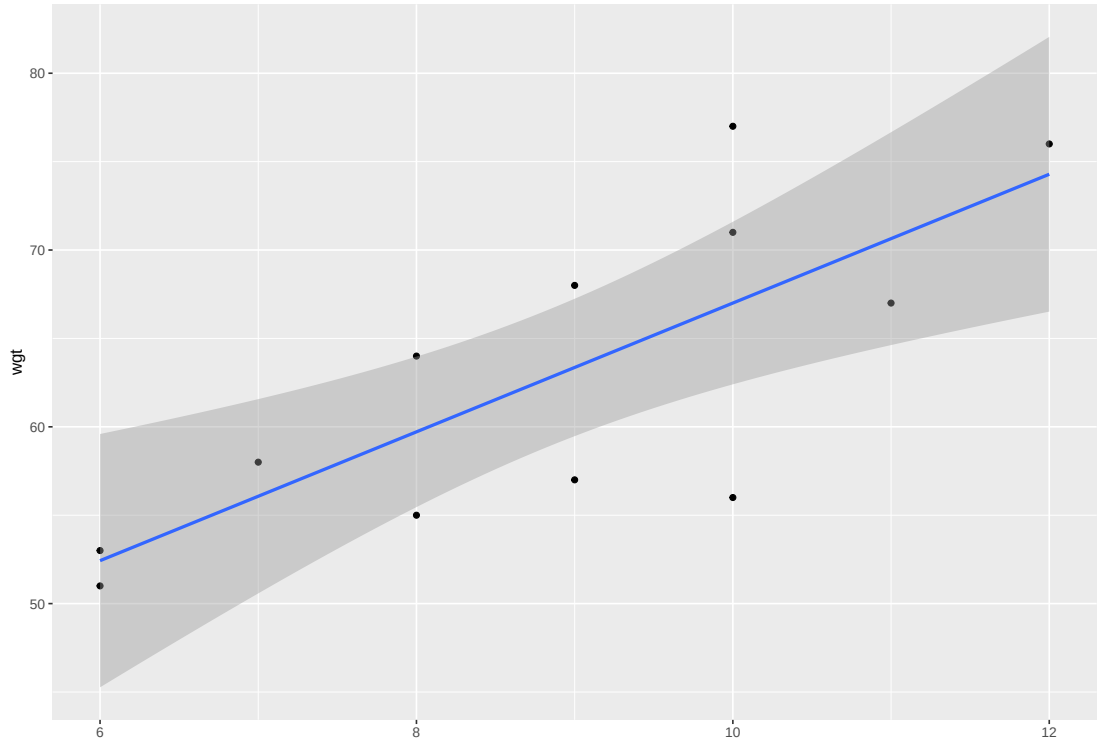
Regresión multivariable.

- El modelo de regresión lineal simple con una variable que hemos usado es

$$Y = \beta_0 + \beta_1 X + \epsilon, \qquad \text{con} \qquad \epsilon \sim N(0, \sigma)$$

pero muchas veces vamos a querer estudiar situaciones en que Y depende de más de ua variable explicativa. Como ejemplo usaremos una tabla con datos sobre pesos, alturas y edades de un grupo de niños. Inicialmente pensamos en la relación entre edad y peso.

```
childData = data.frame(
wgt = c(64, 71, 53, 67, 55, 58, 77, 57, 56, 51, 76, 68),
hgt = c(57, 59, 49, 62, 51, 50, 55, 48, 42, 42, 61, 57),
age = c(8, 10, 6, 11, 8, 7, 10, 9, 10, 6, 12, 9))
ggplot(childData) +
      geom_point(aes(age, wgt)) +
      geom_smooth(aes(age, wgt), method="lm")
```



Modelo de regresión lineal con una variable.

- Como ya sabemos, poodemos analizar el modelo de la página anterior con R así:

```
##
## Call:
## lm(formula = wgt ~ age, data = childData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -11.000  -3.911   1.143   4.071  10.000
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  30.5714     8.6137   3.549  0.00528 **
## age           3.6429     0.9551   3.814  0.00341 **
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.015 on 10 degrees of freedom
## Multiple R-squared:  0.5926,   Adjusted R-squared:  0.5519
## F-statistic: 14.55 on 1 and 10 DF,  p-value: 0.003407
```

  Los dos asteriscos que aparecen en la fila age indican que parece haber una relación significativa entre edad y peso.

- Pero al hacer esto no estamos teniendo en cuenta el efecto que la altura puede tener sobre esa relación. Como señala B. Caffo en Regression Models el objetivo principal de la regresión multivariable es *analizar la relación entre una variable explicativa y una respuesta, teniendo en cuenta el resto de las variables.*

Modelo lineal con dos variables.

- La extensión del modelo lineal para incluir dos variables explicativas es formalmente muy sencilla:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon, \qquad \text{con} \qquad \epsilon \sim N(0, \sigma)$$

  Es decir, añadimos un término más para la nueva variable.

- Para obtener un modelo como este en R, en el que añadimos la variable altura `hgt` hacemos:

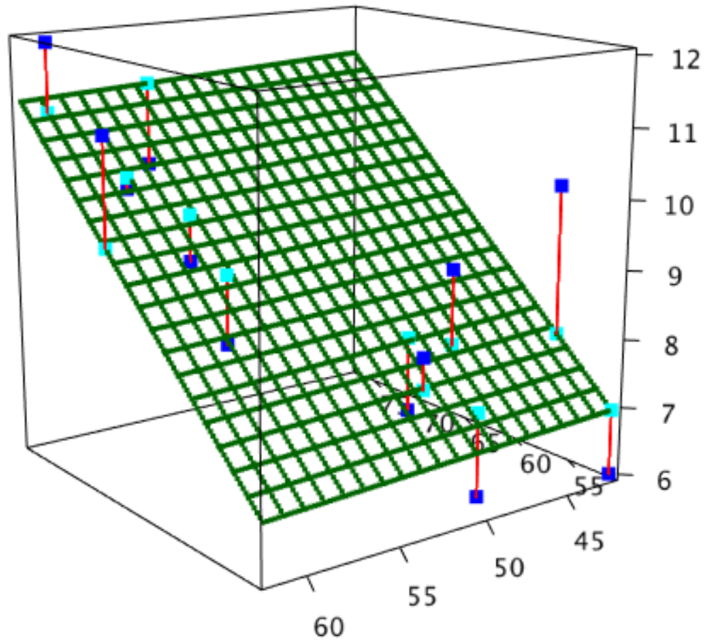
```
modelo2 = lm(wgt ~ age + hgt, data = childData)
summary(modelo2)
```

```
##
## Call:
## lm(formula = wgt ~ age + hgt, data = childData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.8708 -1.7004  0.3454  1.4642 10.2336
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.5530    10.9448   0.599   0.5641
## age           2.0501     0.9372   2.187   0.0565 .
## hgt           0.7220     0.2608   2.768   0.0218 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.66 on 9 degrees of freedom
## Multiple R-squared:   0.78,    Adjusted R-squared:  0.7311
## F-statistic: 15.95 on 2 and 9 DF,  p-value: 0.001099
```

Fíjate en que ahora la edad ya no aparece como significativa y la altura sí, aunque con un p-valor relativamente grande.

Representación del modelo con dos variables.

- Al introducir dos variables explicativas en el modelo la geometría de la situación aumenta de dimensión. Ya no podemos representarla mediante un diagrama de dispersión en el plano, sino que tendríamos que ir a una representación tridimensional como esta:



Puedes explorar esta representación tridimensional en este enlace. El papel de la recta de regresión lo juega ahora un *plano de regresión*, pero las nociones de residuos, valores predichos, etc. siguen teniendo el mismo sentido.

- Aunque vamos a usar ejemplos con dos variables explicativas, está claro que podríamos usar más variables y en esos casos ya no es posible visualizar el modelo así.

Expresión de los coeficientes estimados en términos de residuos.

- En el modelo de dos variables $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$ el valor predicho y el residuo para un par de valores de las variables predictoras (x_{i1}, x_{i2}) son:

$$\hat{y}_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}, \qquad e_i = y_i - \hat{y}_i$$

- Hay una expresión interesante de los *coeficientes estimados* del modelo con dos variables, que facilita su interpretación. Por ejemplo para la estimación de β_1 es:

$$\hat{\beta}1 = \frac{\sum e_i(Y \sim X_2) e_i(X_1 \sim X_2)}{\sum (e_i(X_1 \sim X_2))^2}$$

  donde $e_i(Y \sim X_2)$ representa los residuos del modelo en el que solo usamos X_2 como variable explicativa, mientras $e_i(X_1 \sim X_2)$ son los residuos de un modelo en el usamos X_2 para explicar los valores de X_1. Al considerar los residuos con respecto a X_2 es como si *restáramos* el efecto de esa variable por un lado sobre Y y por otro lado sobre X_1. Así que lo que queda en la estimación es el efecto de X_1 sobre Y *ajustado* respecto de X_2.

- En general en un modelo con p variables explicativas $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$ tendríamos:

$$\hat{\beta}1 = \frac{\sum e_i(Y \sim (X_2 + \cdots + X_p) e_i(X_1 \sim (X_2 + \cdots + X_p))}{\sum (e_i(X_1 \sim (X_2 + \cdots + X_p)))^2}$$

  con expresiones similares para los otros $\hat{\beta}_i$. El caso de $\hat{\beta}_0$ se puede tratar introduciendo una variable auxiliar X_0 que vale 1 en todas las observaciones.

Comprobación con R en los datos del ejemplo. Identidad Anova.

- La estimación de $\beta 1$, el coeficiente de age para el `modelo2` que es `wgt ~ age + hgt` es:

```
modelo2$coefficients
```

```
## (Intercept)         age         hgt
##    6.553048    2.050126    0.722038
```

Para comprobar las expresiones residuales anteriores vamos a crear dos modelos auxiliares en los que analizamos el efecto de `hgt` sobre `wgt` y sobre age por separado:

```
modelo_yx2 =  lm(wgt ~ hgt, data = childData)
modelo_x1x2 =  lm(age ~ hgt, data = childData)
```

y ahora usamos la expresión de $\hat{\beta}_1$ en términos de los residuos de estos dos modelos:

```
sum(residuals(modelo_yx2) * residuals(modelo_x1x2)) / sum(residuals(modelo_x1x2)^2)
```

```
## [1] 2.050126
```

Como puede verse, es el mismo resultado.

- Por otra parte, para estos modelos multivariable sigue siendo verdad la **identidad Anova** que ya vimos en el caso de una variable:

$$\underbrace{\sum_{i=1}^{n}(y_i-\bar{y})^2}_{SStotal} = \underbrace{\sum_{i=1}^{n} e_i^2}_{SSresidual} + \underbrace{\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2}_{SSmodelo}$$

y los términos que aparecen en ella se interpretan exactamente igual.

Interpretación de los coeficientes.

- Las expresión del modelo de regresión múliple permite interpretar de forma sencilla los coeficientes β_i. Por ejemplo β_1 es el incremento medio en el valor de Y esperado al aumentar en una unidad el valor de X_1 *manteniendo el resto de las variables explicativas constantes (controlando respecto a esas variables)*.
- En el ejemplo la edad está en años, la altura en pulgadas y el peso en libras. Así que por ejemplo el valor estimado $\hat{\beta}_2 \approx 0.722$ significa que por cada pulgada adicional de altura *pero manteniendo la edad constante* el peso aumenta en 0.722 libras.
- Vamos a comprobarlo usando predict. Creamos una tabla de tres datos de entrada con edad constante y alturas espaciadas por una pulgada:

```
nuevosDatos = data.frame(age = c(9, 9, 9), hgt = c(52, 53, 54))
(pesosPredichos = predict(modelo2, newdata = nuevosDatos))
```

```
##        1        2        3
## 62.55016 63.27220 63.99424
```

Y ahora vamos a ver las diferencias entre elementos sucesivos del vector de resultados (eso es precisamente lo que hace la función `diff`)

```
diff(pesosPredichos)
```

```
##        2        3
## 0.722038 0.722038
```

Las diferencias son precisamente el valor de $\hat{\beta}_2$, como esperábamos.

Temas para seguir avanzando.

- Apenas hemos rozado la superficie de los modelos de regresión múltiple y nos dejamos pendientes muchos aspectos, de los que aquí vamos a destacar solo algunos por su especial relevancia.
- No hemos discutido la inferencia para este tipo de modelos, aunque es una generalización natural de lo que vimos en el caso de una variable. En particular con `lm` es sencillo obtener e interpretar inferencias tales como intervalos de confianza para los β_i, intervalos de predicción para valores de Y, contrastes de hipótesis sobre esos coeficientes, etc. Como muestra:

```
confint(modelo2)
```

```
##                    2.5 %    97.5 %
## (Intercept) -18.20587071 31.311967
## age          -0.07002526  4.170278
## hgt           0.13205592  1.312020
```

- Para que esas inferencias estén bien fundadas es necesario comprobar en primer lugar que se verifican las hipótesis del modelo. Y para ello se usan herramientas de diagnóstico análogas a las que vimos en el caso de una variable: representaciones gráficas de los residuos que sirven para analizar la independencia, homogeneidad de la varianza, posibles puntos influyentes, etc.
- Otro tema interesante del que aun no hemos tenido ocasión de tratar es la posible presencia de *interacción* entre las variables predictoras. Pero como lo vamos a discutir en el caso de Anova, dejamos esto aparcado por el momento.
- Para profundizar en todos estos aspectos recomendamos consultar las fuentes que aparecen en la sección de Referencias. El tema seguramente más relevante por novedoso de los que nos quedan por tratar es el tema de la selección de modelos. Lo discutiremos a continuación.

Selección de modelos.

- ¿Qué modelo es mejor para representar las relaciones entre variables en `childData`? ¿El modelo inicial `wgt ~ age` o el modelo con dos predictores `wgt ~ age + hgt`? Una manera de responder consiste en comparar la fracción de la variabilidad total (recuerda *SStotal*) explicada por cada uno de los dos modelos. Esto se lleva a cabo mediante una tabla Anova, que en R puede obtenerse mediante:

```
anova(modelo2)
```

```
## Analysis of Variance Table
##
## Response: wgt
##           Df Sum Sq Mean Sq F value    Pr(>F)
## age        1 526.39  526.39 24.2419 0.0008205 ***
## hgt        1 166.43  166.43  7.6646 0.0218070 *
## Residuals  9 195.43   21.71
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

  Podemos interpretar esta tabla así: los tres asteriscos de la primera fila muestran que el modelo1 `wgt ~ age` es preferible al *modelo nulo* (que no usa variables explicativas y predice un valor constante `mean(wgt)`). A continuación el asterisco de la segunda fila dice que, con un p-valor mayor en este caso, el modelo2 `wgt ~ age + hgt` es preferible al modelo1.

- **Advertencias:** aunque pueda parecer igual, para R no es lo mismo `wgt ~ age + hgt` que `wgt ~ hgt + age`. Prueba a definir un modelo3 con `wgt ~ hgt + age` y luego haz `anova(modelo3)` para ver la diferencia. Además, en este caso las cosas sencillas porque los modelos están *anidados* (uno se obtiene del otro añadiendo una variable). Pero en el caso general la selección de modelos es *mucho más complicada*.

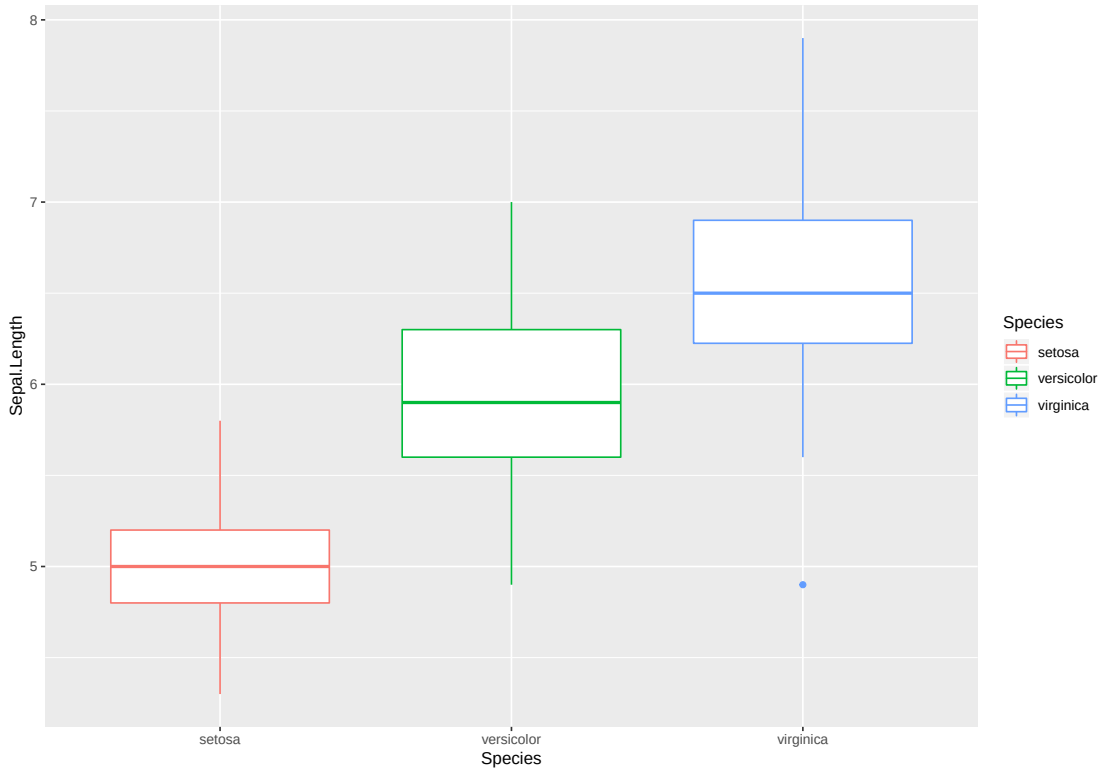
Sección 2

Modelos lineales con factores. Anova.

Un ejemplo

- Aunque la motivación inicial del modelo de regresión lineal está en la relación $C \sim C$ entre dos variables continuas, los métodos que se desarrollaron allí se pueden extender a otro tipo de situaciones. En particular es fácil extenderlos al caso $C \sim F$ en el que la variable respuesta es una variable continua pero queremos utilizar como variable explicativa un factor.
- Para centrar las ideas vamos a empezar por un ejemplo muy sencillo. Usando la tabla `iris` vamos a estudiar la relación entre la variable respuesta continua `Sepal.Length` y el factor `Species`. Empezamos con una figura de boxplots por nivel que ilustra esa relación.

```
ggplot(iris) +
  geom_boxplot(aes(x = Species, y = Sepal.Length, color=Species))
```



El modelo lineal para $C \sim F$.

- Para describir una relación de tipo $C \sim F$ con notación similar a la de la regresión vamos a seguir llamando Y a la variable respuesta (en el ejemplo la variable continua `Sepal.Length`). Para la variable predictora (el factor `Species`) vamos a hacer algo un poco más complicado. Puesto que la variable tiene tres niveles, vamos a usar dos *variables índice* auxiliares (en inglés se las denomina con poco acierto *dummy variables*), a las que llamaremos X_1 y X_2. Esas variables sólo toman los valores 0 y 1 y se tiene esta tabla de valores:

	Variables	
Species	X_1	X_2
setosa	0	0
versicolor	1	0
virginica	0	1

- Con estas variables la ecuación del modelo lineal te debería resultar familiar:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon, \qquad \text{con} \qquad \epsilon \sim N(0, \sigma)$$

  enseguida volveremos sobre el significado de los β_i de esta ecuación.

- ¡No te preocupes, R se encarga de todo esto automáticamente! Basta con escribir:

```
modelo = lm(Sepal.Length ~ Species, iris)
```

  y R se encarga de definir las variables auxiliares X_1, X_2 adecuadas.

Interpretación de los coeficientes del modelo.

- El modelo que estamos construyendo predice como respuesta para cada nivel del factor X la media de Y en ese nivel del factor. Es decir, que si las medias de $Y = \texttt{Sepal.Length}$ en cada uno de los tres niveles de $X = \texttt{Species}$ son, respectivamente μ_1, μ_2 y μ_3 entonces al usar la ecuación del modelo

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon, \qquad \text{con} \quad \epsilon \sim N(0, \sigma)$$

  con una observación del nivel i, debemos obtener como respuesta $Y = \mu_i$.

- Pero recuerda que si la observación es de la especie `setosa` entonces $X_1 = X_2 = 0$. Y al sustituir esto junto con $Y = \mu_1$ en la ecuación se obtiene

$$\beta_0 = \mu_1$$

- Si la observación es de la especie `versicolor` entonces $X_1 = 1, X_2 = 0$. Sustituyendo esto con $Y = \mu_2$ en la ecuación vemos que ha de ser:

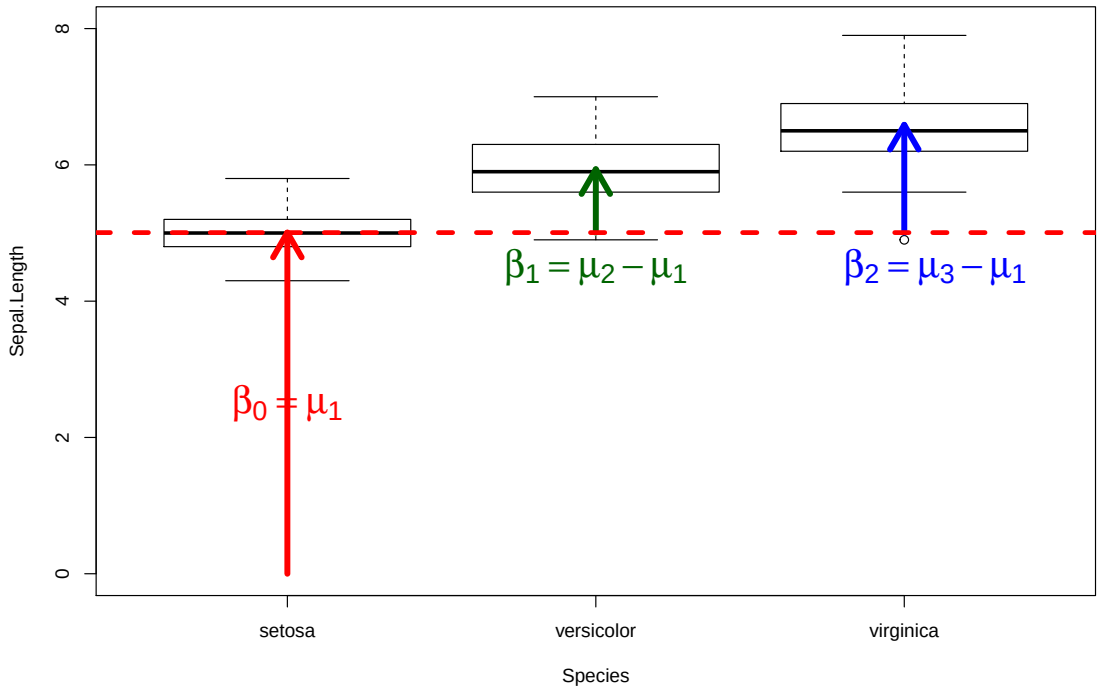
$$\beta_1 = \mu_2 - \mu_1$$

  De forma análoga, para una observación de la especie `virginica` se obtiene

$$\beta_2 = \mu_3 - \mu_1$$

Representación gráfica de los coeficientes del modelo.

- Podemos visualizar esos coeficientes en un diagrama de boxplots paralelos mediante las flechas de colores que se muestran:



Como indica el diagrama en este modelo la media μ_1 del primer nivel (`setosa`) se usa como *nivel de referencia*; es decir, como término indepiente o *intercept* β_0 del modelo. Los dos coeficientes *beta*$_1 = \mu_2 - \mu_1$ y $\beta_2 = \mu_3 - \mu_1$ representan las diferencias entre las medias de esos niveles y el nivel de referencia.

Estimando los coeficientes del modelo.

- A poco que lo pensemos debería estar claro que la estimación de los valores μ_i se obtiene mediante las medias muestrales $\bar{Y}_1$, $\bar{Y}_2$ y $\bar{Y}_3$ de Y en cada uno de los niveles del factor X. En R:

```
(medias = aggregate(Sepal.Length ~ Species, iris, FUN = mean)[,2])
```

```
## [1] 5.006 5.936 6.588
```

Ahora calculamos $\bar{y}_1, \quad \bar{y}_2 - \bar{y}_1, \quad \bar{y}_3 - \bar{y}_1$

```
c(medias[1], medias[2] - medias[1], medias[3] - medias[1])
```

```
## [1] 5.006 0.930 1.582
```

y comparamos con los coeficientes del modelo que se obtiene con `lm`:

```
modelo = lm(Sepal.Length ~ Species, iris)
(coefs = modelo$coefficients)
```

```
##       (Intercept) Speciesversicolor  Speciesvirginica
##             5.006             0.930             1.582
```

que confirma nuestra interpretación de los coeficientes del modelo.

Anova de una vía. Utilizando el modelo para hacer inferencia.

- Un modelo lineal para $C \sim F$ como el que acabamos de describir se conoce clásicamente como *modelo Anova de una vía (one-way Anova)*. La palabra Anova se debe a que también en este caso se tiene una identidad Anova similar a la que vimos en la regresión. Si definimos el *residuo* de una observación como la diferencia $e_i = y_i - \hat{y}_i$ entre el valor observado y la estimación del valor predicho por el modelo, esa relación Anova se escribe de hecho exactamente igual:

$$\underbrace{\sum_{i=1}^{n}(y_i - \bar{y})^2}_{SStotal} = \underbrace{\sum_{i=1}^{n} e_i^2}_{SSresidual} + \underbrace{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}_{SSmodelo}$$

  Lo único que cambia es la interpretación; aquí la estimación del valor predicho $\hat{y}_i$ es la media muestral $\bar{y}_i$ correspondiente a un nivel del factor predictor X. A partir de esta relación se puede proceder de manera muy parecida a como lo hacíamos en el caso de la regresión.

- El modelo que estamos usando es el que se emplearía por ejemplo para hacer un contraste de la hipótesis nula de que no hay diferencia entre las medias de los tres niveles:

$$H_0 = \{\mu_1 = \mu_2 = \mu_3\}$$

  ¡Atención! La hipótesis alternativa aquí no es "las tres medias son diferentes" sino "al menos hay dos medias que son diferentes."

Información sobre el modelo con R.

- Si hacemos:

```
(sumModelo = summary(modelo))
```

```
##
## Call:
## lm(formula = Sepal.Length ~ Species, data = iris)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6880 -0.3285 -0.0060  0.3120  1.3120
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         5.0060     0.0728  68.762  < 2e-16 ***
## Speciesversicolor   0.9300     0.1030   9.033 8.77e-16 ***
## Speciesvirginica    1.5820     0.1030  15.366  < 2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5148 on 147 degrees of freedom
## Multiple R-squared:  0.6187,   Adjusted R-squared:  0.6135
## F-statistic: 119.3 on 2 and 147 DF,  p-value: < 2.2e-16
```

obtenemos mucha información sobre este modelo. Queremos destacar el valor del coeficiente de correlación 0.6187 que nos dice que el modelo con el factor `Species` explica el porcentaje correspondiente de la variabilidad total en `Sepal.Length`.

De nuevo, esto es solo el principio.

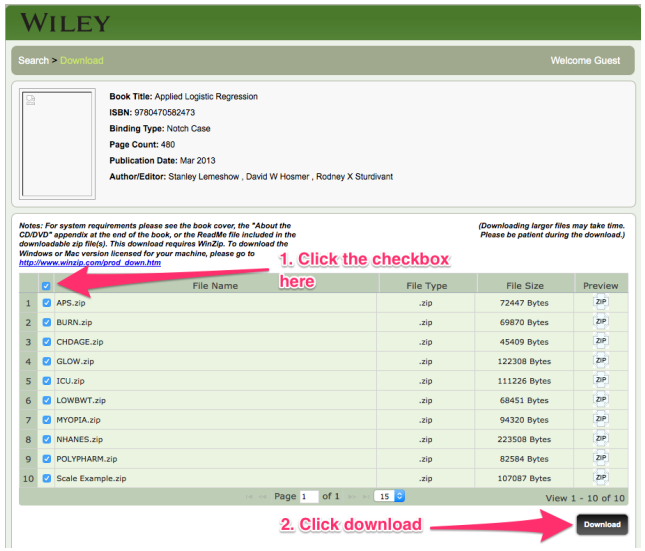
- Como dijimos en el caso de la regresión multivariable, apenas nos hemos asomado a la puerta de los modelos lineales. Todos los temas que adelantamos allí están presentes también cuando las variables explicativas (o algunas de ellas) son factores.
- En particular, debemos ocuparnos de la inferencia sobre los coeficientes del modelo, de verificar que se cumplen las hipótesis del modelo (diagnósticos del modelo), etc.
- La formulación del modelo que hemos empleado (que es la que R usa por defecto) es adecuada para la hipótesis nula de igualdad de medias que hemos discutido antes. Supongamos que rechazamos esa hipótesis. Eso significa que al menos hay dos medias diferentes. La pregunta es evidente ¿cualés? Una manera de hacer esto es comparar dos a dos las medias de cada uno de los niveles. Cuando hay tres niveles, eso no es un problema. Pero si el factor tuviera, por ejemplo, 8 niveles, entonces el número de comparaciones dos a dos sería 28. Y ya hemos visto que hacer tantas comparaciones puede producir errores de tipo I por puro azar.
- Además, es fácil comprender que existen extensiones naturales de estos modelos a situaciones en las que interviene más de un factor como variable explicativa. En ese tipo de modelos (de doble vía si intervienen dos factores, etc.) el análisis es mucho más delicado y choca de lleno con el tema del Diseño Experimental, del que no hemos hablado.

Sección 3

Modelos lineales generalizados (glm). Regresión Logística.

Relaciones del tipo $F \sim C$ para factores binarios.

- En este apartado vamos a estudiar un modelo adecuado para relaciones del tipo F C, en las que usamos una variable continua X como predictor de una variable respuesta Y de tipo factor, que inicialmente suponemos binario (con dos niveles).
- Para guiarnos en la discusión vamos a utilizar un conjunto de datos disponible en el sitio web de la editorial Wiley y procedente del libro *Applied Logistic Regression* de S. Lemeshow (Hosmer Jr, Lemeshow, and Sturdivant 2013). Introduce en el campo de búsqueda adecuado el ISBN del libro que es: 9780470582473 haz clic en *Search* y después haz clic en el enlace con ese número que aparecerá . A continuación marca la casilla para seleccionar todos los ficheros y después haz clic en *Download* como indica la figura. Descargarás entonces un fichero zip con todos los ficheros de datos necesarios. Nosotros vamos a usar uno de los ficheros que contiene, llamado `CHDAGE.txt` con datos sobre la existencia de enfermedad coronaria en un grupo de pacientes. Asegúrate de colocar ese fichero en la subcarpeta `datos` de tu directorio de trabajo.



Descripción del problema.

- Después de leer los datos a un data.frame de R que vamos a llamar CHDdata, lo exploramos:

```
CHDdata = read.table("../datos/CHDAGE.txt", header = TRUE)
head(CHDdata)
```

```
##    ID AGE CHD
## 1   1  20   0
## 2   2  23   0
## 3   3  24   0
## 4   5  25   1
## 5   4  25   0
## 6   7  26   0
```

Como puede verse hay tres variables: `ID` es simplemente un identificador y no la usaremos; `AGE` es la edad con valores enteros y `CHD` (de *coronary heart disease*) es un factor codificado como una variable binario que toma los valores 0 o 1 para indicar, respectivamente, la ausencia o presencia de enfermedad coronaria.

- Recuerda que siempre conviene empezar explorando los datos. En este caso usamos `summary`:

```
summary(CHDdata)
```

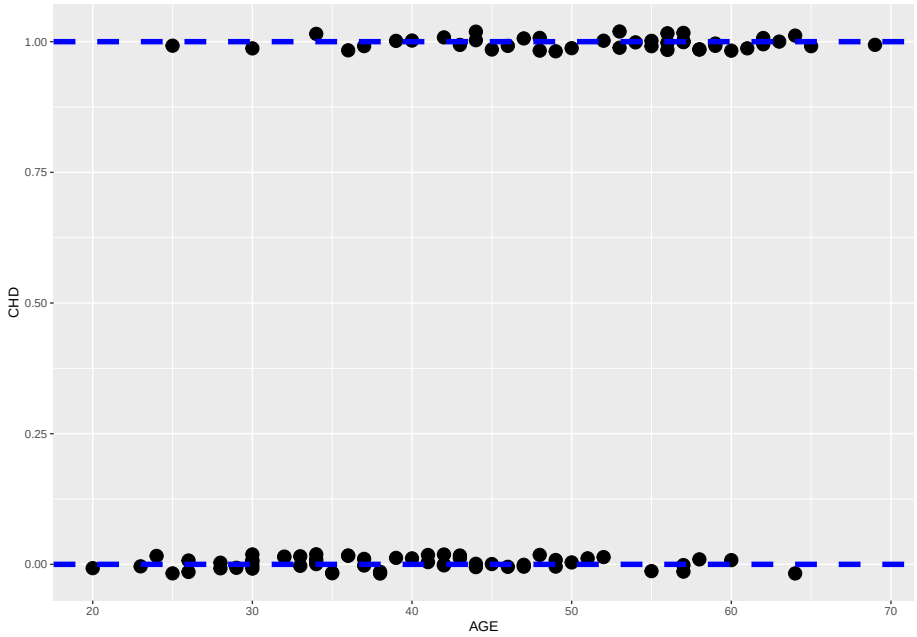
```
##        ID              AGE              CHD        
##  Min.   :  1.00   Min.   :20.00   Min.   :0.00  
##  1st Qu.: 25.75   1st Qu.:34.75   1st Qu.:0.00  
##  Median : 50.50   Median :44.00   Median :0.00  
##  Mean   : 50.50   Mean   :44.38   Mean   :0.43  
##  3rd Qu.: 75.25   3rd Qu.:55.00   3rd Qu.:1.00  
##  Max.   :100.00   Max.   :69.00   Max.   :1.00
```

En particular comprobamos que no hay datos ausentes.

Representando los datos.

- Para entender lo que queremos hacer vamos a usar una representación gráfica similar al dotplot del que hablamos al principio del tema 6.

```
ggplot(CHDdata) +
  geom_point(aes(x = AGE, y = CHD, size=2),
             show.legend=FALSE, position = position_jitter(w = 0, h = 0.02)) +
  geom_hline(yintercept = 0:1, linetype = "dashed", color = "blue", size = 2)
```



Aunque Y es binario hemos *"agitado"* los puntos verticalmente para mejorar la visualización. Fíjate en que a medida que $X = AGE$ aumenta hay más valores de $Y = CHD$ iguales a 1. Como cabría esperar, la presencia de enfermedades coronarias aumenta con la edad. Esa es la *tendencia o señal* que estamos tratando de detectar o cuantificar expresándola a través de algún tipo de modelo.

Construcción del modelo.

- En una situación como esta no podemos usar un modelo lineal de regresión del tipo

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

  porque la respuesta Y no es continua, *solo toma dos valores*.

- La idea astuta que nos va a permitir avanzar en este caso es una que ya nos hemos encontrado antes. Cuando tratamos con una variable continua podemos agruparla en clases (en R con cut) para tratarla como un factor ordenado. Definimos unas franjas de edad así:

```
(AGEbreaks = c(20, seq(from = 30, to = 60, by = 5), 70))
```

```
## [1] 20 30 35 40 45 50 55 60 70
```

  Los intervalos son de 5 en 5 años salvo el primero y el último porque tenemos pocos datos en los extremos. Usamos estos valores para cortar las edades y añadimos esa información a los datos:

```
CHDdata$AgeGroup = cut(CHDdata$AGE, breaks = AGEbreaks, right = FALSE)
```

  Hagamos una tabla de contingencia de CHD frente al grupo de edad:

```
(tabla1 = as.matrix(table(CHDdata$CHD, CHDdata$AgeGroup)))
```

```
##
##      [20,30) [30,35) [35,40) [40,45) [45,50) [50,55) [55,60) [60,70)
##    0       9      13       9      10       7       3       4       2
##    1       1       2       3       5       6       5      13       8
```

  Ahí está de nuevo, visible, la señal. En la segunda fila de la tabla los valores aumentan claramente de izquierda a derecha (y en la primera ocurre al revés).

Pensando en términos de probabilidades.

- Vamos a calcular la suma por columnas de esta tabla (es una tabla de frecuencias):

```
(sumaColumnas = colSums(tabla1))
```

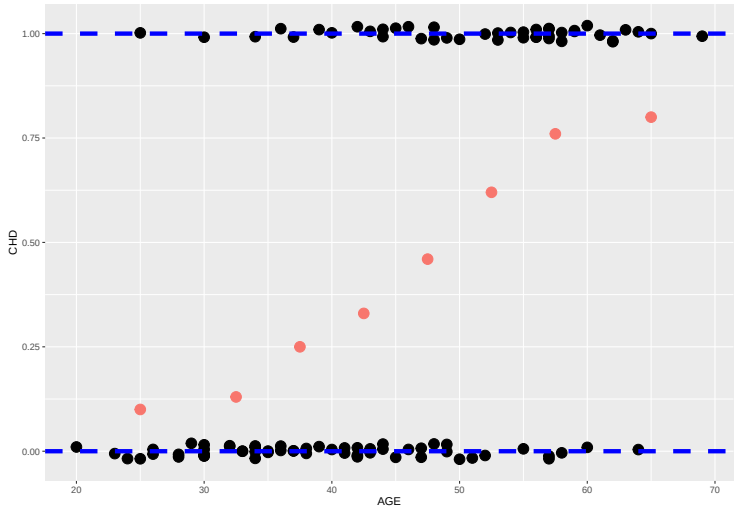
```
## [20,30) [30,35) [35,40) [40,45) [45,50) [50,55) [55,60) [60,70)
##      10      15      12      15      13       8      17      10
```

Y dividimos la segunda fila de la `tabla1` anterior por estas sumas (redondeada a dos cifras). :

```
(probs = signif(tabla1[2, ] / sumaColumnas, 2))
```

```
## [20,30) [30,35) [35,40) [40,45) [45,50) [50,55) [55,60) [60,70)
##    0.10    0.13    0.25    0.33    0.46    0.62    0.76    0.80
```

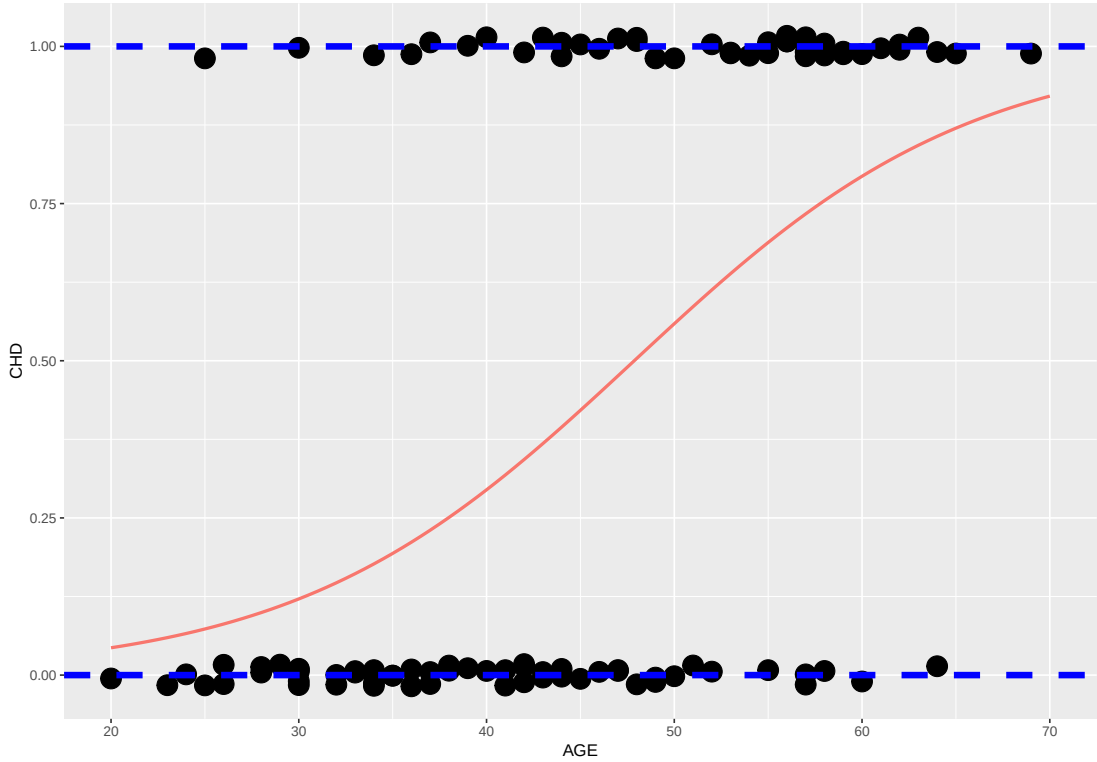
Esto permite pensar en los datos en términos de probabilidades. Por ejemplo, el porcentaje de pacientes de 45 a 50 años con enfermedad coronaria es el 46 % y asciende al 76 % de 55 a 60 años. ¡Esa es la idea clave! Vamos a añadir esto al gráfico (ver código de la sesión):



Los puntos rojos indican la probabilidad de $Y = 1$ para cada intervalo de edades.

El modelo es una curva con forma de s.

- Los puntos rojos de la gráfica previa insinuan una curva con forma de s (*curva sigmoidal*) de izquierda a derecha, como esta (puedes ver el código para ver como la hemos dibujado, pero solo se entenderá después de aprender un poco más):



Esa curva representa el modelo que buscábamos para este tipo de situaciones. Es muy importante recordar que la coordenada vertical de los puntos de esa curva son **probabilidades condicionadas**. Es decir, un punto (x_0, p_{x_0}) de esa curva representa:

$$p_0 = P(Y = 1|X = x_0)$$

Curvas logísticas. Ajuste mediante la verosimilitud.

- Una vez entendido esto, el siguiente paso es más técnico. La familia de curvas sigmoidales que vamos a usar (puedes pensar en ellas como un sustituto de las rectas de regresión) es esta:

Curvas logísticas.
Dados dos números cualesquiera β_0, β_1 la curva logística correspondiente es:

$$f(x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}.$$

Puedes familiarizarte con las propiedades de esta familia de curvas en este enlace.

- Ahora se trata de elegir los valores de β_0 y β_1 que producen *la mejor curva logística posible* para ajustarla a nuestros datos. Esto debe recordarte a lo que hicimos cuando buscábamos la mejor recta de regresión posible. Allí usamos el método de mínimos cuadrados, partiendo de una expresión de los residuos del modelo. Pero ahora no podemos hacer esto, por razones técnicas (la estructura de error del problema no sigue una distribución normal como ocurría en la regresión, sino una binomial). El método que se emplea es uno de los más importantes en Estadística, con un alcance muy general. Se basa en la función **verosimilitud (likelihood)**. En un sentido muy amplio la verosimilitud de un modelo con ciertos parámetros (como β_0, β_1) se define como:

$$\mathcal{L}(\text{modelo con parámetros}) = P(\text{datos} \,|\, \text{dados los valores de los parámetros})$$

y el método para determinar β_0 y β_1 en regresión logística consiste en elegir los que maximizan la función de verosimilitud (por razones técnicas se minimiza el logaritmo de $\mathcal{L}$, que se denomina *loglikelihood*).

Ajustando un modelo logístico con R. Función `glm`.

- La función verosimilitud es computacionalmente mucho más complicada que el error cuadrático medio que usábamos en la regresión. Así que no vamos a dar fórmulas para la estimación de β_0 y β_1 como hacíamos allí. Dejaremos que R se encargue de hacer la estimación por nosotros. Para ello se usa, en lugar de `lm`, la función `glm` de *generalized linear models*. Volveremos después sobre este nombre. En nuestro ejemplo esto se hace con:

```
(glmCHD = glm(CHD ~ AGE, family = binomial, data = CHDdata))
```

```
##
## Call:  glm(formula = CHD ~ AGE, family = binomial, data = CHDdata)
##
## Coefficients:
## (Intercept)          AGE
##     -5.3095       0.1109
##
## Degrees of Freedom: 99 Total (i.e. Null);  98 Residual
## Null Deviance:     136.7
## Residual Deviance: 107.4   AIC: 111.4
```

El formato de la llamada a la función `glm` y de la salida es similar a lo que ya conocemos, salvo por el argumento `family = binomial` que es que le indica a `glm` que queremos un modelo logístico. En particular obtenemos *estimaciones* $\hat{\beta}_0$ y $\hat{\beta}_1$ de los parámetros del modelo logístico:

```
coefficients(glmCHD)
```

```
## (Intercept)         AGE
##  -5.3094534   0.1109211
```

Ahora es un buen momento para volver a examinar el código que genera esta figura y tratar de entenderlo.

Usando el modelo logístico para predecir.

- Con el modelo logístico y la función `predict` podemos hacer predicciones de forma muy parecida a lo que hacíamos en el caso de un modelo lineal. Por ejemplo, ¿cuál es la probabilidad de padecer una enfermedad coronaria (probabilidad de $Y = 1$) que predice el modelo para un paciente de $X = 32$ años?

```
edadPredecir = data.frame(AGE = 32)
(probCHD = predict(glmCHD, newdata = edadPredecir, type = 'response'))
```

```
##         1
## 0.1467932
```

es decir, un 15 %.

El argumento `type = 'response'` que hemos incluido en la llamada a `predict` es el que permite obtener una probabilidad. Si hubiéramos usado `type = 'link'` habríamos obtenido como respuesta el valor

$$\hat{\beta}_0 + \hat{\beta}_1 x_0$$

donde, en este ejemplo, $x_0 = 32$. Compruébalo:

```
predict(glmCHD, newdata = edadPredecir, type = 'link')
```

```
##         1
## -1.759977
```

```
coefficients(glmCHD)[1] + coefficients(glmCHD)[2] * 32
```

```
## (Intercept)
##   -1.759977
```

El valor $\hat{\beta}_0 + \hat{\beta}_1 x_0$ se denomina *log-odds* de x_0.

Odds y log-odds en el contexto de la regresión logística.

- Es una forma de entender la probabilidad común en el mundo anglosajón y especialmente en el contexto de las apuestas deportivas. Si cuelves a pensar en la Regla de Laplace recordarás que era:

$$P(A) = \frac{\text{núm. de sucesos elementales favorables a } A}{\text{núm. total de sucesos elementales}}.$$

  En la misma situación los odds de A (a mi me gusta traducirlo por *psoibilidades*, pero na hay una versión establecida en español) se calculan como:

$$O_A = \frac{\text{núm. de sucesos elementales favorables a } A}{\text{núm. de sucesos elementales } \textbf{contrarios} \text{ a } A}.$$

  así que por ejemplo una probabilidad de $\frac{3}{11}$ se convierte fácilmente en unos odds de $\frac{3}{11-3} = \frac{3}{8}$ (apuestas 3 a 8). En general los odds correspondientes a una probabilidad p son:

$$O_p = \frac{p}{1-p}$$

- ¿Por qué es relvante esto en el contexto de la regresión logística? Pues porque el modelo se puede escribir

$$P(Y = 1|X = x) = p = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}.$$

  y si llamamos $w = \beta_0 + \beta_1 x$ y despejamos se obtiene:

$$w = \ln\left(\frac{p}{1-p}\right) = \ln(O_p)$$

  Por eso decimos que $\beta_0 + \beta_1 x$ son los **log-odds** correspondientes a x.

Interpretación del coeficiente β_1 del modelo de regresión logística.

- Ahora ya podemos interpretar β_1. Si aumentamos en una unidad el valor de x entonces los log-odds $\beta_0 + \beta_1 x$ aumentan precisamente en β. A partir de aquí podemos transformar este resultado para traducirlo en términos de probabilidades.
- Esto ayuda a entender porque `predict` ofrece la opción de obtener el resultado en términos de *log-odds*. En algunos contextos esa cantidad es preferible porque su relación matemática con la variable predictora X es más simple.
- Es muy importante darse cuenta de que la relación sencilla de los valores de X es con los log-odds, no con las probabilidades. Fíjate en lo que pasa cuando consideramos varios valores consecutivos de la edad y miramos los valores predichos de los log-odds y de las probabilidades:

```
edades = data.frame(AGE = 50:55)
probabilidades = predict(glmCHD, newdata = edades, type = 'response')
diff(probabilidades)
```

```
##          2          3          4          5          6
## 0.02714072 0.02662822 0.02597167 0.02518590 0.02428792
```

```
logOdds = predict(glmCHD, newdata = edades, type = 'link')
diff(logOdds)
```

```
##         2         3         4         5         6
## 0.1109211 0.1109211 0.1109211 0.1109211 0.1109211
```

```
coefficients(glmCHD)[2]
```

```
##       AGE
## 0.1109211
```

Como puedes ver al aumentar la edad en un año el incremento de las probabilidades (calculado con `diff`) no es constante, pero el de los log-odds sí y coincide con el coeficiente estimado $\hat{\beta}_1$.

La idea detrás del modelo lineal generalizado (glm).

- La idea clave en la construcción del modelo de regresión logística de una variable es la de pasar de pensar en los valores de Y a pensar en las probabilidades $p = P(Y = 1|X = x)$; *el modelo predice probabilidades*. Pero todavía se puede dar un paso más y mirar el modelo desde una perspectiva más general.
- Para hacer esto, debemos recordar que la *variable respuesta condicionada* $(Y|X = x)$ se puede entender como una variable binaria que toma los valores 1 (con probabilidad p) y 0 (con probabilidad q). Es decir, que es una variable de Bernouilli. Y la media de esa variable de Bernouilli es precisamente $\mu_{Y|X=x} = p$. Así que dando un paso más podemos pensar que en lugar de predecir la probabilidad *el modelo predice medias de* Y (una media para cada valor de X).
- Pero si recordamos la expresión del modelo de regresión lineal:

$$Y = \beta_0 + \beta_1 X + \epsilon, \qquad \text{con} \quad \epsilon \sim N(0, \sigma)$$

al tomar la media de la variable respuesta condicionada $(Y|X = x)$ se obtiene, teniendo en cuenta que $\mu_\epsilon = 0$:

$$\mu_{Y|X=x} = \beta_0 + \beta_1 x = \hat{Y}(x)$$

es decir, la media $\mu_{Y|X=x}$ coincide con el valor que predice el modelo. Otra vez: *el modelo predice medias de* Y (una media para cada valor de X).

Formalizando el glm.

- Podemos convertr esa idea en una definición. Un **modelo lineal generalizado** consta de tres componentes:

  1. Una **función de distribución** de probabilidad f con una distribución conocida para modelizar la variable respuesta condicionada. Técnicamente, la distribución f debe ser de la *familia exponencial*, una familia muy amplia de variables aleatorias que incluye la normal, la binomial, etc.
  2. Un **predictor lineal** $\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$ que depende de las variables predictoras $X_1, \ldots, X_p$ (pueden ser continuas o factores usando variables índice).
  3. Una **función de enlace (link function)** g que sirve para lo que esperamos del modelo: *predecir medias de* Y (una media para cada valor de X).

- *Ejemplo 1:* en el modelo de regresión lineal simple la distribución f que usamos es la de normal $N(0, \sigma)$, el predictor lineal es $\beta_0 + \beta_1 X$ mientras que la función de enlace es la identidad $g(\mu) = \mu$. Por eso este modelo es el más simple, por la sencillez del enlace.

- *Ejemplo 2:* en el modelo de regresión logística con una variable la distribución f es la Bernouilli con probabilidad $p = \mu_{Y|X=x}$ que hemos visto, el predictor lineal vuelve a ser $\beta_0 + \beta_1 X_1$ mientras que la función de enlace es $g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$, la función log-odds que es la inversa de la transformación logística.

- Se pueden definir otros modelos lineales generalizados cambiando estas componentes para por ejemplo modelizar una relación $Y \sim X$ donde Y es una variable de tipo Poisson (regression de Poisson) o para modelizar una variable respuesta de tipo factor politómico con más de dos niveles (regresión multinomial). Ver Wikipedia y Regression Models de B.Caffo.

Sección 4

Complementos de R.

Familia apply.

- La función `apply` sirve para aplicar una función a una tabla *marginalmente* (por filas o por columnas). Tabla aquí significa una estructura tabular, como un data.frame o matriz. Por ejemplo, para calcular un intervalo de confianza para cada columna de una matriz:

```
options(width = 80)
set.seed(2019)
M = matrix(rnorm(100 * 5), ncol = 5)
head(M)
```

```
##            [,1]        [,2]       [,3]       [,4]        [,5]
## [1,]  0.7385227 -0.84505019  0.7208450 -0.1601367 -1.25634410
## [2,] -0.5147605  0.85792776 -0.3946306 -0.1565010  0.25845334
## [3,] -1.6401813 -0.68360647  0.9826626  0.6516824  2.20375446
## [4,]  0.9160368 -0.01069453 -1.4043020  0.5355833 -0.04794335
## [5,] -1.2674820 -1.40416454  0.8002034  0.1941661 -2.23715410
## [6,]  0.7382478  1.39177661 -0.7507425  0.2994167 -0.02410486
```

```
apply(M, MARGIN = 2,
      FUN = function(x)t.test(x, alternative = "two.sided")$conf.int)
```

```
##            [,1]        [,2]        [,3]       [,4]       [,5]
## [1,] -0.2529956 -0.36644266 -0.32040990 -0.2474999 -0.3094948
## [2,]  0.1063276  0.02600377  0.06556297  0.1446696  0.1423253
```

La función que hemos aplicado es una *función anónima*, definida ad hoc dentro de apply y el resultado es una matriz que contiene en cada columna el intervalo de confianza correspondiente.

- **Notas:** En los últimos años la implementación los bucles for de R han mejorado mucho y ya no puede decirse que `apply` sea mucho más rápido. Con `apply` se sonigue código más compacto, pero menos legible. La librería `dplyr`es una alternativa recomendable frente a muchos usos tradicionales de apply. Veremos a continuación las funciones más destacadas de la familia `apply`.

lapply y sapply

- La función `lapply` es similar, pero en este caso operando sobre listas. Es decir, `lapply` recorre los elementos de la lista, aplica una función a cada uno de ellos y produce como salida una lista con los resultados. Por ejemplo, aquí aplicaremos `lapply` para calcular la dimensión de los elementos de una lista que contiene data.frames, matrices o tablas.

```
L = list(A = iris, B = matrix(1:12, nrow = 3),
         C = table(mpg$cyl), D = iris)
lapply(L, FUN = dim)
```

```
## $A
## [1] 150   5
##
## $B
## [1] 3 4
##
## $C
## [1] 4
##
## $D
## [1] 150   5
```

- Por su parte `sapply` actúa como `lapply` pero trata de simplificar el resultado para producir como salida si es posible un vector o matriz en lugar de una lista. Una situación típica es cuando queremos aplicar una función a cada elemento de un vector. Por ejemplo, vamos a fabricar muestras de tamaño 4 de varias binomiales el mismo $p = 1/3$ pero con tamaño distinto:

```
(muestras = sapply(4:8, function(k)rbinom(n = 4, size = k, prob = 1/3 )))
```

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

Como se ve no obtenemos una lista de muestras sino una matriz.

tapply

- La función `tapply` solía usarse para aplicar funciones (como la media) a columnas de un data.frame según los valores de un factor en otra columna. Por ejemplo en la tabla `mpg` la media del consumo urbano en cada clase de vehículo se obtiene con:

```
tapply(mpg$cty, INDEX = mpg$class, FUN = mean)
```

```
##    2seater    compact    midsize    minivan     pickup subcompact        suv
##   15.40000   20.12766   18.75610   15.81818   13.00000   20.37143   13.50000
```

aunque hay alternativas más claras como `aggregate`

```
aggregate(cty ~ class, data = mpg, FUN = mean)
```

```
##        class      cty
## 1    2seater 15.40000
## 2    compact 20.12766
## 3    midsize 18.75610
## 4    minivan 15.81818
## 5     pickup 13.00000
## 6 subcompact 20.37143
## 7        suv 13.50000
```

o más preferiblemente `dplyr` con `group_by` y `summarize`.

```
mpg %>% group_by(class) %>% summarize(mean(cty))
```

```
## # A tibble: 7 x 2
##   class      `mean(cty)`
##   <chr>            <dbl>
## 1 2seater           15.4
## 2 compact           20.1
## 3 midsize           18.8
## 4 minivan           15.8
## 5 pickup            13
## 6 subcompact        20.4
## 7 suv               13.5
```

Fechas en R. Lubridate.

- Las fechas (y horas) son un tipo de dato muy frecuente pero a menudo difícil de gestionar, por los diferentes formatos que se usan. En el ecosistema del `tidyverse` disponemos de la librería `lubridate` para facilitar ese tipo de operaciones. Recomendamos la lectura del Capítulo 16 de *R for Data Science* o el Capítulo 8 de (Boehmke 2016).
- Una de las operaciones básicas es la conversión de una fecha en formato texto al tipo de datos fecha que usa `lubridate`. Existen diversas funciones de conversión para los formatos más habituales. En este ejemplo, después de examinar el texto hemos optado por la función `dmy_hm` (de *day-month-year_hour_minute*):

```
library(lubridate)
fechaTexto = "21-07-1969 02:56UTC"
(fecha = dmy_hm(fechaTexto))
```

```
## [1] "1969-07-21 02:56:00 UTC"
```

De la misma forma, existen funciones como `ydm`, `hms`, etc. para extraer la información necesaria de la mayoría de formatos de texto que encontrarás.

Acceso a las componentes de una fecha y operaciones con fechas.

- Una vez que disponemos de una fecha en el formato correcto, también podemos extraer de ella las partes que la componen:

```
day(fecha)
```

```
## [1] 21
```

```
month(fecha)
```

```
## [1] 7
```

```
hour(fecha)
```

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

- También es muy fácil hacer operaciones con fechas. Por ejemplo, ¿qué fecha se obtiene si sumamos 100 días a la fecha de ejemplo que venimos usando?

```
fecha + days(100)
```

```
## [1] "1969-10-29 02:56:00 UTC"
```

¿Y si le sumamos 1000 horas y 42 minutos?

```
fecha + hours(1000)  + minutes(42)
```

```
## [1] "1969-08-31 19:38:00 UTC"
```

El 29 de octubre de ese mismo año se envio el primer mensaje a través de Arpanet, el precursor de Internet. ¿Cuántos días separan ambas fechas?

```
  fecha2 = dmy_hm("29-10-1969 00:00")
  fecha2 - fecha
```

```
## Time difference of 99.87778 days
```

Algunos recursos online para seguir aprendiendo R .

Cursos online (gratuitos y comerciales).

- Coursera & John Hopkins Data Science Specialization.
- DataCamp, cursos de pago, impartidos en algunos casos por expertos como el propio Hadley Wickham.

Referencias para la sesión

Enlaces

Bibliografía

Boehmke, Bradley C. 2016. *Data Wrangling with R*. Springer. https://doi.org/10.1007/978-3-319-45599-0.

Hosmer Jr, David W, Stanley Lemeshow, and Rodney X Sturdivant. 2013. *Applied logistic regression*. Vol. 398. John Wiley & Sons.