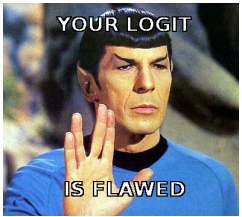
Estadística Aplicada III

Regresión logística y otros métodos en clasificación

Jorge de la Vega Góngora

Departamento de Estadística,
Instituto Tecnológico Autónomo de México



Regresión logística

Regresión logistica

- Los modelos de regresión logística permite ampliar el conjunto de modelos de clasificación.
- La regresión logística se puede utilizar como un modelo adicional de clasificación, cuando:
  - se tienen k variables que pueden incluir variables categóricas o factores, como atributos de las observaciones a clasificar. Estas variables conforman un vector **x**.
  - La variable de respuesta Y sólo toma dos posibles valores, que pueden corresponder a una indicadora de 2 poblaciones. Los valores siempre se pueden identificar como $Y = 0$ o $Y = 1$.
- Estos modelos transforman una combinación lineal de atributos de un elemento de la población a una probabilidad de pertenencia. En este contexto nos interesa estimar $P(Y = 1|\mathbf{x}) = p(\mathbf{x})$.
- La regresión logística es uno de los ejemplos más intuitivos de los *modelos log-lineales*, correspondientes a las distribuciones de la variable de respuesta Bernoulli, Binomial, Poisson, Gamma y multinomial, entre otras posibilidades.

Regresión logística: logit

- Para poder modelar adecuadamente $P(Y = 1|\mathbf{x})$, es conveniente usar la razón de probabilidades o **razón de momios**:

$$\frac{P(Y = 1|\mathbf{x})}{1 - P(Y = 1|\mathbf{x})} = \frac{P(Y = 1|\mathbf{x})}{P(Y = 0|\mathbf{x})} = \frac{p(\mathbf{x})}{1 - p(\mathbf{x})} \in (0, \infty)$$

- Para poder extender el rango de posibles valores de la razón de momios a todo $\mathbb{R}$, se toma el logaritmo de esta razón, que se conoce como **logit**:

$$\log\left(\frac{p(\mathbf{x})}{1 - p(\mathbf{x})}\right) = \text{logit}(p(\mathbf{x}))$$

- Se asume que el logit es una función lineal de los predictores, obteniendo entonces:

$$\text{logit}(p(\mathbf{x})) = \beta'\mathbf{x}$$

En lo que sigue se supondrá que el vector $\mathbf{x}$ es de la forma $\mathbf{x} = (1, x_1, \ldots, x_p)$, incluyendo el valor de la constante en el modelo.

función logística I

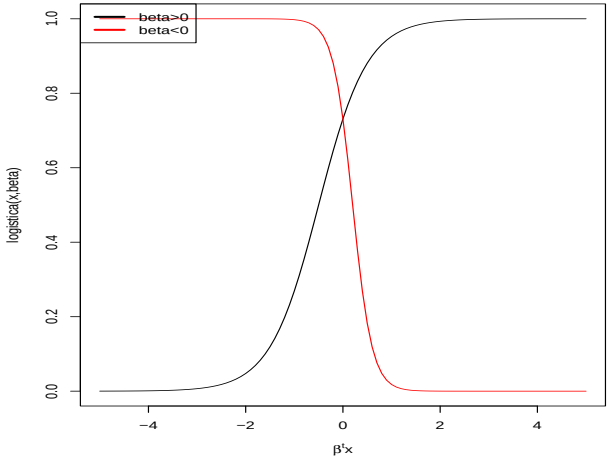
- Una de las ventajas del logit es que podemos expresar de manera directa a la probabilidad de interés despejando de manera directa y obteniendo la función logística:

$$
\begin{aligned}
p(\mathbf{x}) &= \frac{\exp(\beta'\mathbf{x})}{1+\exp(\beta'\mathbf{x})} \\
&= \frac{1}{1+\exp(-\beta'\mathbf{x})}
\end{aligned}
$$

Un ejemplo de la función logistica se grafica a continuación:

```
logistica <- function(x,beta) exp(beta[1]+beta[2]*x)/(1+exp(beta[1]+beta[2]*x))
curve(logistica(x, beta = c(1,2)), from = -5, to = 5,
          ylab = "logistica(x,beta)", xlab = expression(beta^t*x))
curve(logistica(x, beta = c(1,-5)), from = -5, to = 5, add = T, col = "red")
legend("topleft",legend=c("beta>0","beta<0"),
       col=c("black","red"), lty=c(1,1),lwd=3)
```

función logística II



- En este modelo, $\mathrm{E}(Y) = p(\mathbf{x})$ y $\mathrm{Var}(Y) = p(\mathbf{x})(1 - p(\mathbf{x}))$, por lo que no tiene varianza constante.

Estimación del modelo por máxima verosimilitud

- Los estimadores de $\beta = (\beta_0, \beta_1, \dots, \beta_k)$ se pueden obtener por máxima verosimilitud. Para una muestra de n observaciones $y_1, y_2, \dots, y_n \sim$ **Bernoulli** $(p(\mathbf{x}))$ se tiene:

$$P(Y_j = y_j) = p(\mathbf{x})^{y_j}(1 - p(\mathbf{x}))^{1-y_j} \quad j = 1, \dots, n$$

- Entonces la función de verosimilitud es

$$\begin{aligned} L(\beta|\mathbf{x}) &= \prod_{i=1}^{n} p(\mathbf{x}_i)^{y_i}(1 - p(\mathbf{x}))^{1-y_i} \\ &= \prod_{i=1}^{n} \left(\frac{\exp(\beta'\mathbf{x}_i)}{1 + \exp(\beta'\mathbf{x}_i)}\right)^{y_i} \left(1 - \frac{\exp(\beta'\mathbf{x}_i)}{1 + \exp(\beta'\mathbf{x}_i)}\right)^{1-y_i} \\ &= \prod_{i=1}^{n} \left(\frac{\exp(\beta'\mathbf{x}_i)}{1 + \exp(\beta'\mathbf{x}_i)}\right)^{y_i} \left(\frac{1 + \exp(\beta'\mathbf{x}_i) - \exp(\beta'\mathbf{x}_i)}{1 + \exp(\beta'\mathbf{x}_i)}\right)^{1-y_i} \\ &= \prod_{i=1}^{n} \frac{\exp(y_i\beta'\mathbf{x}_i)}{1 + \exp(\beta'\mathbf{x}_i)} = \frac{\exp\left(\sum_{i=1}^{n} y_i\beta'\mathbf{x}_i\right)}{\prod_{i=1}^{n}(1 + \exp(\beta'\mathbf{x}_i))} \end{aligned}$$

- La función $L(\beta|\mathbf{x})$ se tiene que maximizar numéricamente, no tiene una forma analítica cerrada. El método para maximizar este modelo (y otros modelos loglineales) se conoce como *iterative reweighted least squares (IRLS)*. No lo revisaremos en esta clase, y usaremos la solución numérica que da `R`.

Extensión al modelo de regresión binomial I

- Una extensión del modelo logístico surge cuando se tienen replicas de observaciones de la variable de respuesta para el mismo conjunto de predictores.
- Por ejemplo, se puede tener como atributos idénticos `edad`, `género`, `ingreso`, y la persona 1 puede ser un buen sujeto de crédito ($Y_1 = 1$) mientras que la persona 2 es un mal sujeto de crédito ($Y = 0$).
- Supongamos que hay m bloques, cada uno con n_j casos con atributos $\mathbf{x}_j$ fijos, donde $\sum_{j=1}^{m} n_j = n$. Entonces la variable de respuesta Y_j es la suma de las respuestas en esos bloques, por lo que $Y_j \sim \textbf{Bin}\left(n_j, p(\mathbf{x})\right)$. Entonces, bajo el supuesto de independencia de las respuestas, la verosimilitud es ahora:

$$L(\beta|\mathbf{x}) = \prod_{i=1}^{m} \binom{n_i}{y_i} p(\mathbf{x}_i)^{y_i} (1 - p(\mathbf{x}_i))^{n_i - y_i}$$

- Para especificar el modelo binomial en `R`, es necesario representar a los datos en una de tres formas posibles:
  1. Si los datos no están agregados (es decir, la respuesta está en ceros y unos) se aplica el procedimiento usual.

Extensión al modelo de regresión binomial II

2. Si los datos están agregados, es decir, la respuesta es la variable binomial, entonces se deben especificar como pesos las frecuencias, o n_i's para cada caso.
3. se puede pensar a la respuesta como una matriz de dos columnas, donde la primera columna tiene los éxitos y la segunda los fracasos, para cada ensayo.

Estimación de modelo logístico con R: glm

- La principal función para ajustar modelos lineales generalizados es la función `glm`, con argumentos principales:

  `glm(formula,family, data, weights, control),`

  escogiendo familia `binomial`. También se puede especificar una función liga distinta, logit o probit o cloglog de la siguiente manera (logit es default): `family = 'binomial(link=probit)' .`
- La fórmula para especificar el modelo utiliza la misma notación y convenciones que se utiliza para modelos lineales.
- A continuación se especificarán los diferentes comandos que se utilizarán para extraer la información que se comenta del modelo.Supongamos que se ejecuta el modelo:

  ```
  mlogit1 <- glm(y ~ x1 + x2 + x3, family = "binomial", data=datos)
  ```

  Respecto a este modelo se comentarán los comandos necesarios.

Intervalos de confianza asintóticos para los coeficientes

- Cuando el tamaño de muestra n es grande, entonces

$$\hat{\beta} \sim \mathcal{N}_{p+1}\left(\beta, \hat{\mathsf{Var}}(\hat{\beta})\right)$$

donde

$$\hat{\mathsf{Var}}(\hat{\beta}) \approx \left(\sum_{i=1}^{n} \hat{p}(\mathbf{x}_i)(1-\hat{p}(\mathbf{x}_i))\mathbf{x}_i\mathbf{x}_i'\right)^{-1}$$

- La matriz $\hat{\mathsf{Var}}(\hat{\beta})$ se obtiene con el comando `vcov` en `R`.
- De la distribución asintótica anterior se pueden obtener intervalos de confianza para cada uno de los coeficientes:

$$\hat{\beta}_l \pm z_{1-\alpha/2} se(\hat{\beta}_l)$$

- Los errores estándar de los coeficientes se pueden obtener con `sqrt(diag(vcov(mlogit1)))`, o con el comando: `summary(mlogit1)$coeff[,2]`

Pruebas de significancia para regresión logistica I

- En el caso de modelos log-lineales, incluyendo la regresión logística, se puede utilizar una variedad de estadísticas de prueba, incluyendo la prueba LRT (razón de verosimilitides) y la estadística de prueba es típicamente $-2\log(\Lambda)$ donde como siempre $\Lambda = \frac{L(\beta|H_0)}{L(\beta|H_a)}$.
- Usualmente en modelos log-lineales, se utiliza una generalización de la suma de residuales al cuadrado que es un estimador de la variabilidad de los datos con respecto al modelo, conocido como la devianza y corresponde a $-2\log(\Lambda)$. Dependiendo de qué compare Λ, se tienen diferentes devianzas:
  - La *devianza nula*, compara $H_0 : \text{logit}(p(\mathbf{x})) = \beta_0$ versus H_a : el *modelo saturado*, que es el que ajusta perfectamente los datos: $\hat{y} = y$.
  - La *devianza residual*, compara el modelo ajustado contra el modelo saturado. Este caso corresponde en el modelo normal a la suma de residuales al cuadrado, RSS.
- Una aproximación al equivalente del coeficiente de determinación R^2 está dado por:

$$\text{pseudo-}R^2 = \frac{\text{Devianza nula} - \text{Devianza residual}}{\text{Devianza nula}}$$

Pruebas de significancia para regresión logistica II

- La devianza residual tiene distribución aproximada $\chi^2_{(n-k)}$, n es el tamaño de muestra y k es el número total de parámetros en el modelo completo. En la salida de `R` su valor corresponde al concepto: `Residual deviance`, y se puede obtener como `summary(mlogit1)$deviance` o `deviance(mlogit1)`.
- El análisis de varianza permite comparar, como en el caso de regresión lineal, modelos anidados, $M_0 \subset M$. La prueba de hipótesis se puede aproximar de diferentes maneras. En `R`, se puede usar el comando `anova(mlogit1, test = "LRT")` con opciones 'Rao' o 'Chisq' entre otras opciones. En la familia binomial, no se puede usar la prueba F.
- El criterio de información de Akaike es una medida del ajuste que penaliza de acuerdo al número de parámetros ajustados: Si l_M es la log-verosimilitud de un modelo M, entonces

$$AIC = -2l_M + 2k$$

- Otra medida es el criterio de información bayesiano (BIC), que es similar al AIC pero el BIC tiende a penalizar más a modelos con más parámetros:

$$BIC = -2l_M + \log(n)k$$

Hay que recordar que valores menores del AIC o BIC indican un mejor ajuste y puede ser utilizado para comparar modelos que no necesariamente estén anidados.

Regla de clasificación con regresión logística

- En los modelos de regresión logística usualmente se supone que los costos de clasificación errónea son unitarios y las probabilidades iniciales son iguales.

Regla de clasificación logística

Asigna $\mathbf{x}_0$ a Π_1 ($Y = 1$) si

$$\text{logit}(\hat{p}(\mathbf{x}_0)) = \hat{\beta}' \mathbf{x}_0 > 0$$

La regla es equivalente a asignar $\mathbf{x}_0$ a Π_1 si la razón de *momios* es mayor que 1:

$$\frac{\hat{p}(\mathbf{x}_0)}{1 - \hat{p}(\mathbf{x}_0)} = \exp(\hat{\beta}' \mathbf{x}_0) > 1$$

Ejemplo: datos de Salmón I

- Los datos cuentan con una variable adicional que identifica el sexo del salmón y que no se consideró en los ejercicios previos; en esta ocasión se tomará en cuenta, para ver si es posible obtener una mejor clasificación.
- La respuesta `lugar` está codificada como 1 si el salmón es de Alaska o 2 si es de Canadá. La variable se considerará de tipo *factor* en el modelo. Los predictores serán `genero` que también es un factor, y `rio` y `mar`.

Ejemplo: datos de Salmón II

```r
options(scipen=9) #para ver números sin notación científica.
#lugar y género son factores, lugar es la variable de respuesta.
salmon <- read.table("https://raw.githubusercontent.com/jvega68/EA3/master/datos/J%26W/T11-2.DAT",
                     colClasses = c("factor", "factor", "numeric", "numeric"))
names(salmon) <- c("lugar", "genero", "rio", "mar")
mlogit1 <- glm(lugar ~ . , data = salmon, family = "binomial")
summary(mlogit1)


Call:
glm(formula = lugar ~ ., family = "binomial", data = salmon)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.78657    6.29358   0.602 0.547403
genero2      0.28156    0.83383   0.338 0.735614
rio          0.12642    0.03570   3.541 0.000398 ***
mar         -0.04865    0.01457  -3.339 0.000842 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 138.629  on 99  degrees of freedom
Residual deviance:  38.674  on 96  degrees of freedom
AIC: 46.674

Number of Fisher Scoring iterations: 7
```

- El p-value asociado a la devianza nula se obtiene de χ^2_{99} lo que da un valor de 0.0053. Por lo tanto, el modelo es significativo.

Ejemplo: datos de Salmón III

- Los resultados muestran que el género del salmón no es relevante para clasificación. Reestimando el modelo sin género, obtenemos:

```
mlogit2 <- update(mlogit1,.~. - genero) #quitamos género
summary(mlogit2)


Call:
glm(formula = lugar ~ rio + mar, family = "binomial", data = salmon)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.92484    6.31518   0.621 0.534275
rio          0.12605    0.03586   3.515 0.000439 ***
mar         -0.04854    0.01452  -3.342 0.000831 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 138.629  on 99  degrees of freedom
Residual deviance:  38.788  on 97  degrees of freedom
AIC: 44.788

Number of Fisher Scoring iterations: 7
```

Ejemplo: datos de Salmón IV

- Con este modelo estimado, podemos aplicar la regla de clasificación $\mathbf{x} \in \Pi_1$ si $\widehat{\beta}'\mathbf{x} \geq 0$ y calcular la matriz de confusión. Obtenemos los valores de $\widehat{\beta}'\mathbf{x}$ del componente `linear.predictors` del modelo:

```
matriz_conf <- table(salmon$lugar, mlogit2$linear.predictors > 0)
matriz_conf
```

```
   FALSE TRUE
  1    46    4
  2     3   47
```

```
(APER <- 100*(matriz_conf[1,2] + matriz_conf[2,1])/sum(matriz_conf))
```

```
[1] 7
```

```
# No cambio significativo con el modelo estimado con LDA.
```

Otros modelos de clasificación

Árboles de decisión/clasificación (CART)

- Los árboles de decisión son diagramas en forma de árboles, en donde cada decisión es un *nodo* y éstas decisiones se pueden anidar. La predicción final se realiza en una *hoja* del árbol, o *nodo terminal*.
- CART: *Classification and Regression Trees* (Breiman, et. al. 1984)
  - **Respuesta categórica**: *Classification Tree*. Tratan de predecir las probabilidades de las clases generadas por la variable de respuesta categórica en las hojas del árbol. Por ejemplo, probabilidades de lluvia, probabilidades de default, preferencias de géneros cinematográficos, etc.
  - **Respuesta Continua**: *Regression Tree*. Tratan de predecir un valor medio para la respuesta en las hojas del árbol, como cantidad promedio de lluvia o la tasa esperada de default en crédito.
- Los modelos CART son un ejemplo de modelo no paramétrico de clasificación, y permite mucho mayor flexibilidad que los modelos paramétricos.

CART: Solución y algoritmos I

- El número posible de árboles de decisión crece exponencialmente con respecto al número de predictores p. De hecho No existe un algoritmo eficiente que determine el árbol de decisión óptimo para un problema (el problema es np-completo)
- La falta de un algoritmo solución óptimo global implica la existencia de varias heurísticas que compiten para construir árboles de decisión con estrategias que son localmente óptimas.
- Diferentes algoritmos llevarán a diferentes soluciones, incluso para los mismos datos.
- Considerando sólo árboles que se particionan recursivamente y que parten un nodo sólo en dos ramas. El algoritmo recursivo básico se conoce como algoritmo de Hunt, basado en la estrategia de *divide y vencerás*:
  - Dado que estamos en cierto nivel del árbol de decisión y que $D_t = (y_t, \mathbf{X}_t)$ es el conjunto de observaciones que están asociadas al nodo t y que $\{y_1, y_2\}$ son las clases disponibles para la variable de respuesta.
  - Si todos los casos en D_t pertenecen a una sola clase, entonces t es un nodo hoja etiquetado como esa clase.
  - Si D_t tiene casos en las dos clases, se dividen los casos en dos nodos hijos de tal forma que la pureza del nuevo conjunto de nodos exceda algún umbral.

CART: Solución y algoritmos II

- La pureza de un conjunto de casos se puede medir de diferentes modos:
  - Coeficiente de Gini: $Gini(t) = 1 - \sum_{i=1}^{2} w_i(t)^2$
  - Entropía: $En(t) = -\sum_{i=1}^{2} w_i(t) \log_2 w_i(t)$

  donde $w_i(t)$ es la fracción de casos que pertenecen a la clase i en el nodo t.
- En la práctica es necesario aplicar validación cruzada para no sobreajustar el modelo.

Herramientas en R I

- En R, se pueden utilizar los paquetes `tree` y `rpart`. El paquete `partykit` tiene varias funciones para trabajar con árboles de decisión.
- CHAID (Chi-square automatic interaction detection), Kass 1980, es un procedimiento de clasificación que en particular se *concentra tanto respuesta como predictores categóricos*. Puede considerar más de dos nodos en una partición. Se puede instalar en R desde otro repositorio (no es oficial):

```
install.packages("CHAID", repos="http://R-Forge.R-project.org")
```

- El paquete `randomForest` combina la información de varios métodos de clasificación para crear un *ensamble de métodos*.Tiene dos limitantes: no se pueden considerar datos perdidos y tiene un límite de 32 niveles por variable categórica. Una alternativa es la función `cforest` del paquete `party`, que no tiene ese límite, aunque puede generar problemas de memoria y tiempo.
- Otros métodos no paramétricos incluyen *Support Vector Machines (SVM)* que cubre clasificadores lineales y no lineales. Busca transformaciones de los datos para encontrar separaciones a través de variedades. Se pueden encontrar funciones en los paquetes `kernlab` y `e1017`. La referencia es Karatzoglou (2006), que hace referencia a otros paquetes. También las Redes Neuronales, que se pueden analizar con los paquetes `nnet` y `neuralnet`.

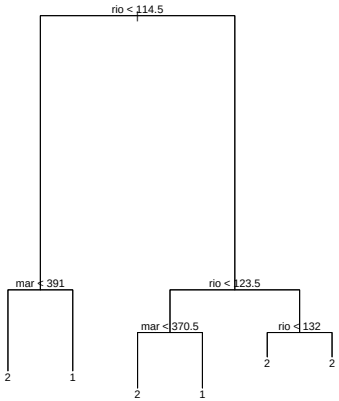
Ejemplo CART: salmón I

En el siguiente ejemplo ponemos la restricción de que el número mínimo de observaciones en las hojas del árbol son 5 observaciones:

Ejemplo CART: salmón II

```
library(tree)
arbol <- tree(lugar ~ ., data = salmon, split = "gini", mincut = 5)
summary(arbol)


Classification tree:
tree(formula = lugar ~ ., data = salmon, split = "gini", mincut = 5)
Variables actually used in tree construction:
[1] "rio" "mar"
Number of terminal nodes:  6
Residual mean deviance:  0.2488 = 23.39 / 94
Misclassification error rate: 0.05 = 5 / 100

arbol

node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 100 138.600 1 ( 0.50000 0.50000 )
   2) rio < 114.5 46  27.180 1 ( 0.91304 0.08696 )
     4) mar < 391 6   7.638 2 ( 0.33333 0.66667 ) *
     5) mar > 391 40   0.000 1 ( 1.00000 0.00000 ) *
   3) rio > 114.5 54  45.300 2 ( 0.14815 0.85185 )
     6) rio < 123.5 14  19.120 2 ( 0.42857 0.57143 )
      12) mar < 370.5 7   0.000 2 ( 0.00000 1.00000 ) *
      13) mar > 370.5 7   5.742 1 ( 0.85714 0.14286 ) *
     7) rio > 123.5 40  15.880 2 ( 0.05000 0.95000 )
      14) rio < 132 10  10.010 2 ( 0.20000 0.80000 ) *
      15) rio > 132 30   0.000 2 ( 0.00000 1.00000 ) *

plot(arbol)
text(arbol,digits=2)
```

Ejemplo CART: salmón III

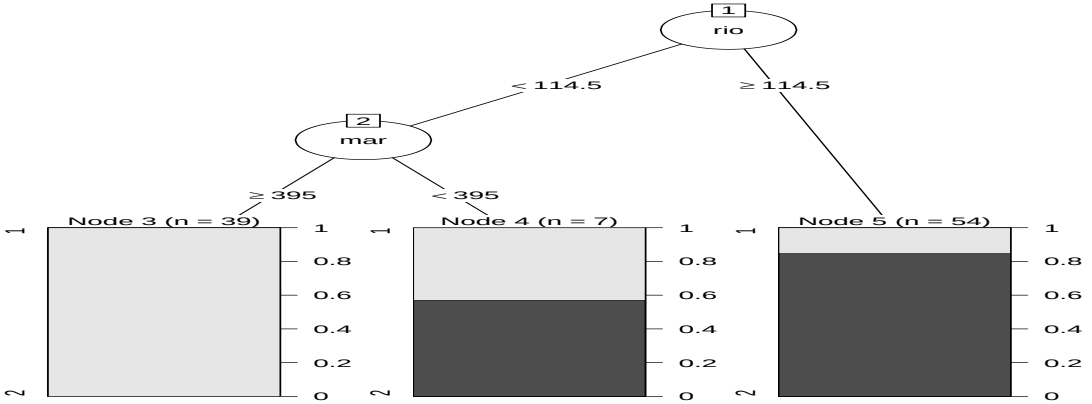


Ejemplo con `rpart`: salmón I

```
library(rpart)
library(partykit)
arbol <- rpart(lugar ~ ., data = salmon)
# summary(arbol) #salida muy larga para un chunk
arbol

n= 100

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 100 50 1 (0.50000000 0.50000000)
  2) rio< 114.5 46  4 1 (0.91304348 0.08695652)
    4) mar>=395 39  0 1 (1.00000000 0.00000000) *
    5) mar< 395 7  3 2 (0.42857143 0.57142857) *
  3) rio>=114.5 54  8 2 (0.14814815 0.85185185) *
```

```
plot(as.party(arbol))
```

Ejemplo con `rpart`: salmón II



Naïve Bayes I

- El objetivo es inferir una respuesta binaria $y \in \{0, 1\}$ de un caso de la información que se provee de las variables atributo $\mathbf{x}_0$
- En esta aplicación, usualmente se supone que $\mathbf{x}$ es un vector de variables categóricas o factores.
- **Idea**: buscar en el conjunto de entrenamiento para encontrar casos que correspondan con $\mathbf{x}_0$ y usar la respuesta más frecuente para deducir el valor de y. **Problema**: Puede haber pocos o ningún caso que corresponda a $\mathbf{x}_0$.
- Si no se tienen datos para estimar $P(y = 1|\mathbf{x})$, se puede usar el teorema de Bayes para obtener:

$$P(y = 1|\mathbf{x}) = \frac{P(\mathbf{x}|y = 1)\pi_1}{P(\mathbf{x}|y = 1)\pi_1 + P(\mathbf{x}|y = 0)\pi_0}$$

Esta solución puede ser difícil de implementar por la dificultad de estimar $P(\mathbf{x}|y = 1)$ y $P(\mathbf{x}|y = 0)$. Para π_i se pueden usar las proporciones del conjunto de entrenamiento.

- El enfoque Bayesiano ingenuo supone que, condicional en la respuesta, los predictores son independientes. Bajo este supuesto, se puede escribir:

$$P(y=1|\mathbf{x}) = \frac{\left[\prod_{i=1}^{k} P(X_i|y=1)\right] P(y=1)}{\left[\prod_{i=1}^{k} P(X_i|y=1)\right] P(y=1) + \left[\prod_{i=1}^{k} P(X_i|y=0)\right] P(y=0)}$$

  Este modelo puede ser implementado ya que usualmente hay suficiente información en el conjunto de entrenamiento para estimar las probabilidades marginales $P(X_i|y=0)$ y $P(X_i|y=1)$ para $i = 1, \ldots, k$.

- En los paquetes `e1071` y `klaR` implementan funciones con el mismo nombre `naiveBayes`.

Ejemplo. [Naïve Bayes para datos de Titanic]

Para tomar un conjunto de datos con predictores categóricos, tomaré los datos del Titanic. Como los datos están dados en frecuencias, hay que descomponerlos en casos individuales.

Naïve Bayes III

```r
# naive bayes
library(e1071)

data ("Titanic")
titanic <- as.data.frame(Titanic)
expande.freqs <- rep.int(seq_len(nrow(titanic)),titanic$Freq)
titanic <- titanic[expande.freqs, -5]

nb <- naiveBayes(Survived ~ .,data=titanic)
#nb #no se imprime por su longitud
clases <- predict(nb,titanic)
table(clases,titanic$Survived)
```

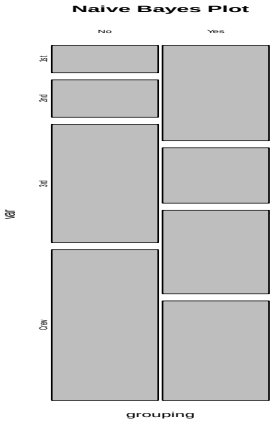
```
clases   No  Yes
   No  1364  362
   Yes  126  349
```

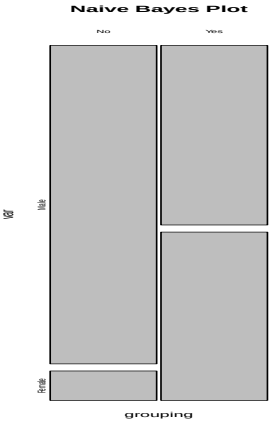
```r
library(klaR)
```

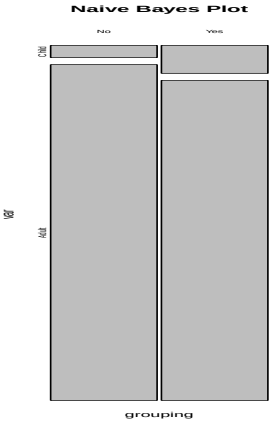
```
Loading required package: MASS
```

```r
nb.klaR<- klaR::NaiveBayes(Survived~.,data=titanic)
par(mfrow=c(1,3))
plot(nb.klaR)
```

Naïve Bayes IV







□