Árboles de Regresión y Clasificación

Gabriel Martos Venturini
gmartos@utdt.edu

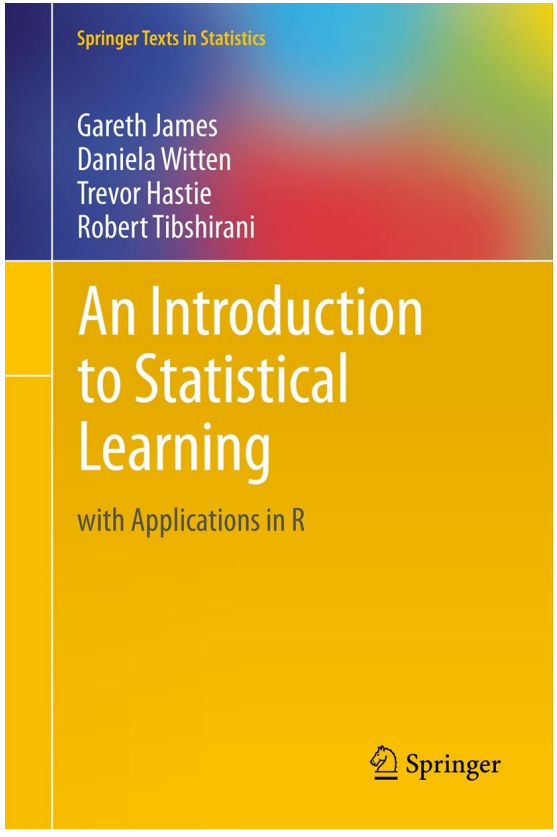
UNIVERSIDAD
TORCUATO DI TELLA

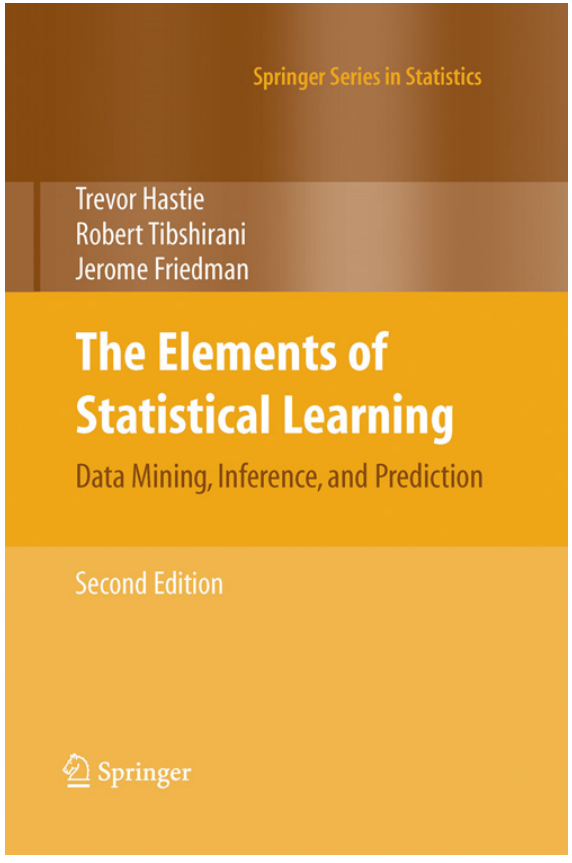
Agenda

Árboles de regresión y clasificación

Taller de trabajo en clase: CHURN en Teleco

Bibliografía recomendada





ISL: Lectura sugerida 8
ESL: Lectura sugerida 9.1 a 9.3.

Agenda

Árboles de regresión y clasificación

- Introducción y contexto
- Árboles de regresión
- CART en R y ejemplos
- Árboles de clasificación
- Consideraciones finales

Taller de trabajo en clase: CHURN en Teleco

Agenda

Árboles de regresión y clasificación

- Introducción y contexto
- Árboles de regresión
- CART en R y ejemplos
- Árboles de clasificación
- Consideraciones finales

Taller de trabajo en clase: CHURN en Teleco

- Desarrollado por matemáticos de Berkeley y Standford (L. Breiman: "**C***lassification* **A***nd* **R***egression* **T***rees*", 1984).

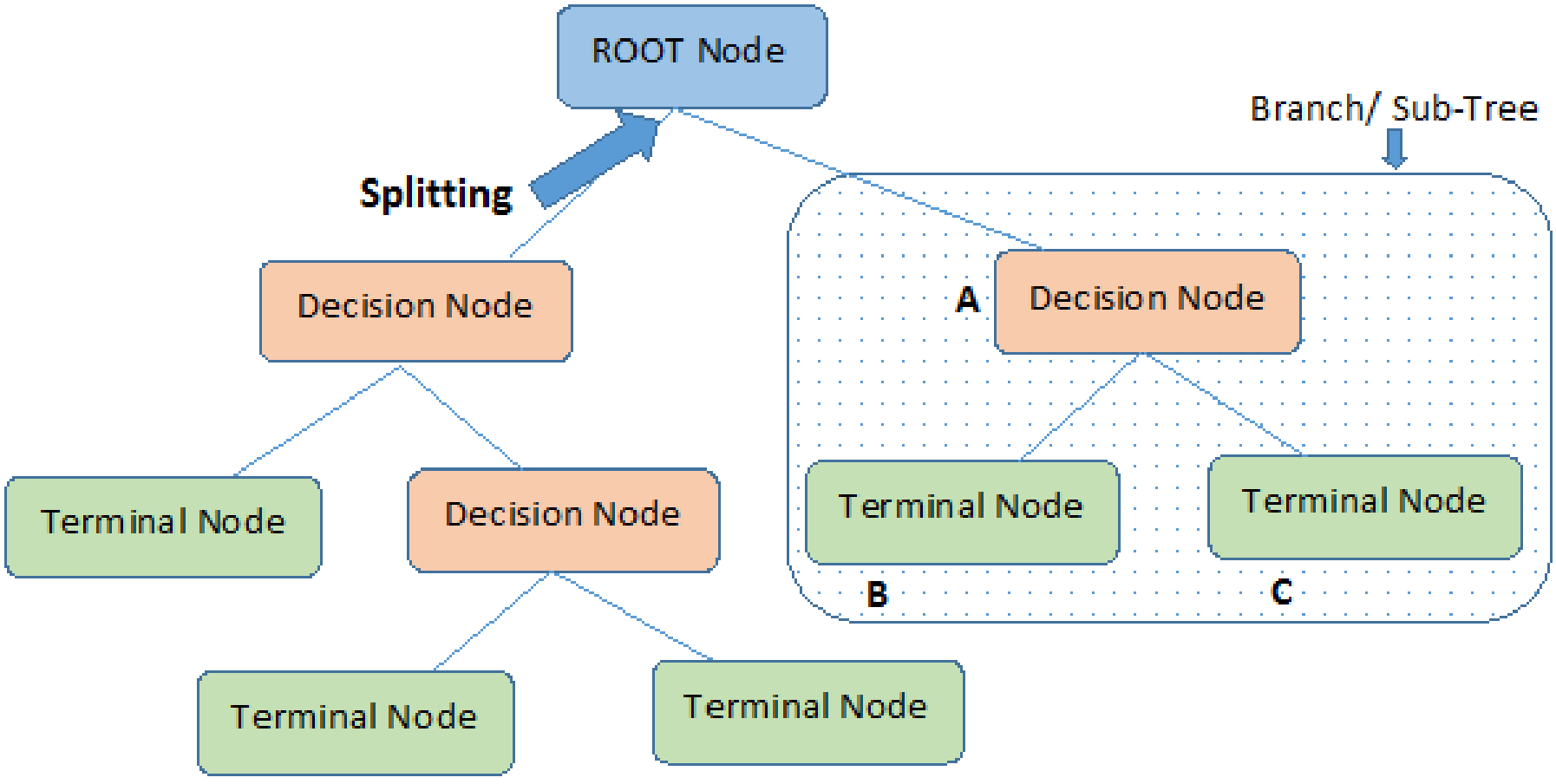


Figure: Visualizamos el modelo a través de un árbol (grafo conexo acíclico).

- En el contexto de un problema de aprendizaje supervisado $Y = f(X) + \varepsilon$, un árbol $T(X; \boldsymbol{\theta})$ es un modelo para $f(X)$.
- Cambio de paradigma: Aprender de los datos por medio de reglas.
  - Nodos de decisión = *parámetros del modelo* ("$\boldsymbol{\theta}$").
  - No hay estructura probabilística explícita en el modelo.
  - **Statistical Modeling: The Two Cultures**; L. Breiman (2001).
- Comencemos la presentación introduciendo la *estructura general* de un modelo de partición recursiva (también conocidos como modelos CART, por Classification and Regression Trees).
- Extensiones: Bagging, Random Forest y Boosting.

Agenda

Árboles de regresión y clasificación

- Introducción y contexto
- Árboles de regresión
- CART en R y ejemplos
- Árboles de clasificación
- Consideraciones finales

Taller de trabajo en clase: CHURN en Teleco

Particiones recursivas

Los modelos CART utilizan los datos de train para partir el espacio de features en (hiper)*rectángulos* **disjuntos** de manera de conseguir respuestas homogéneas en cada región R_i del espacio de features.

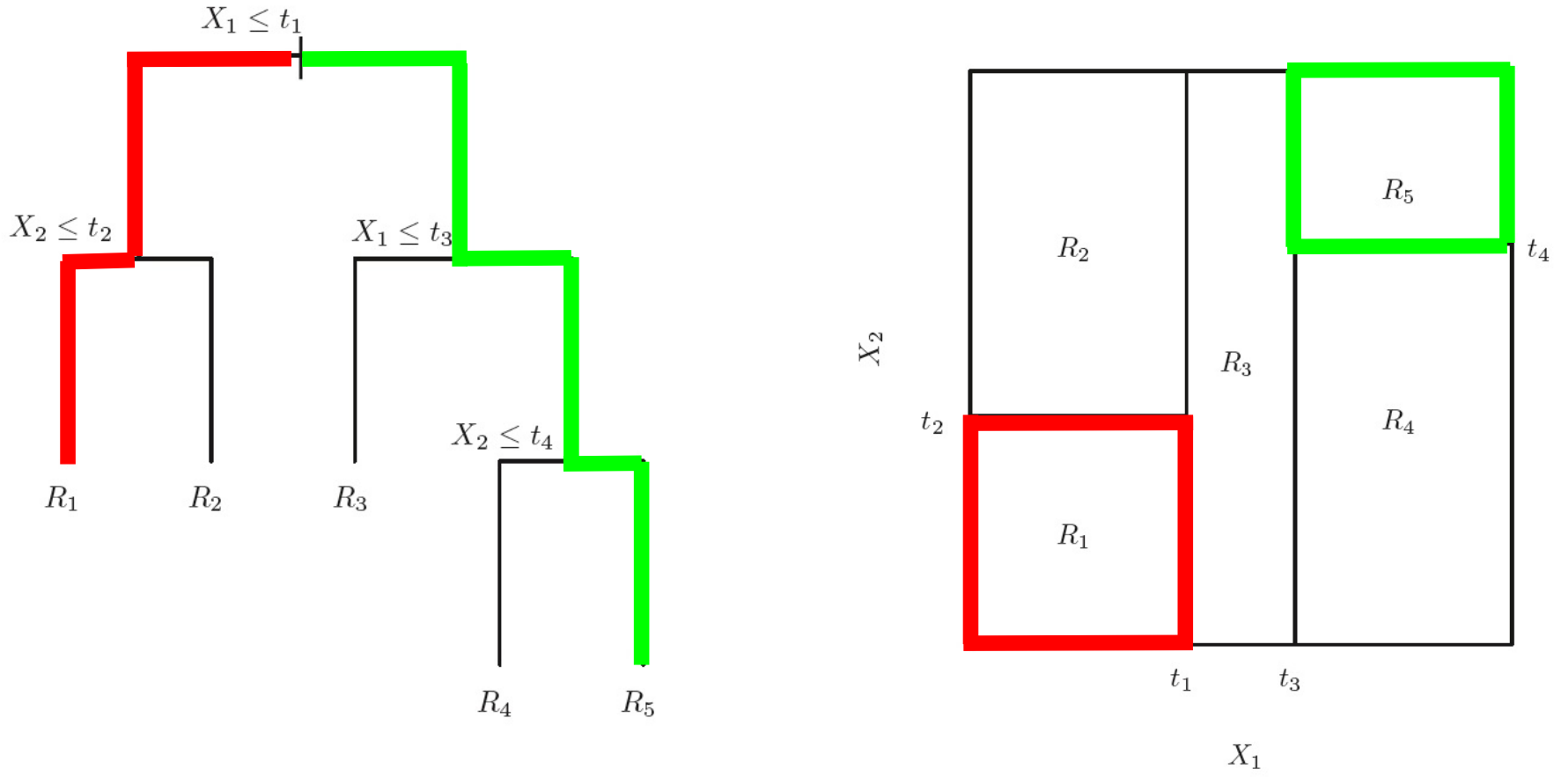
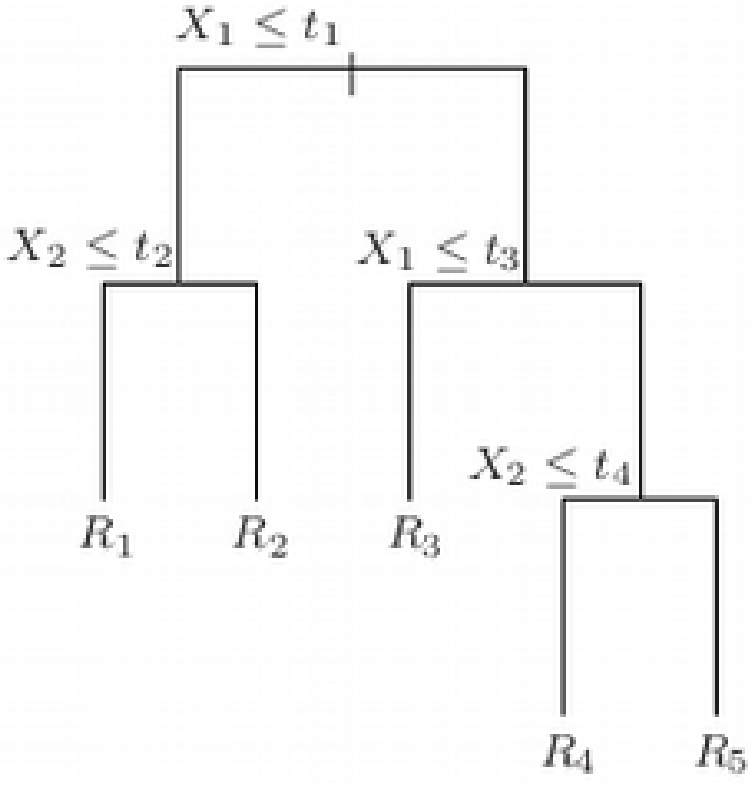
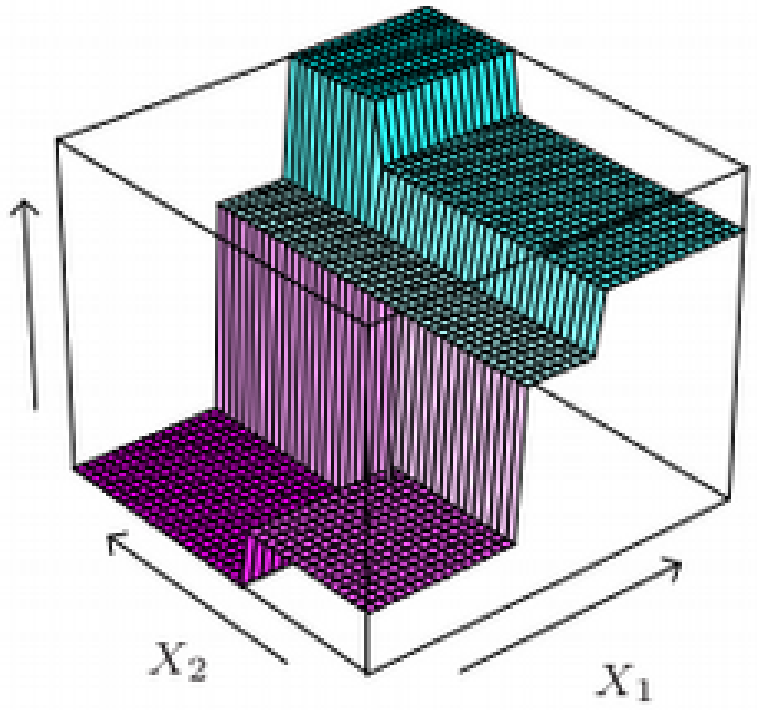


Figure: (X_1, X_2, Y) son continuos y el árbol tiene 5 nodos terminales.

Modelos de partición recursiva

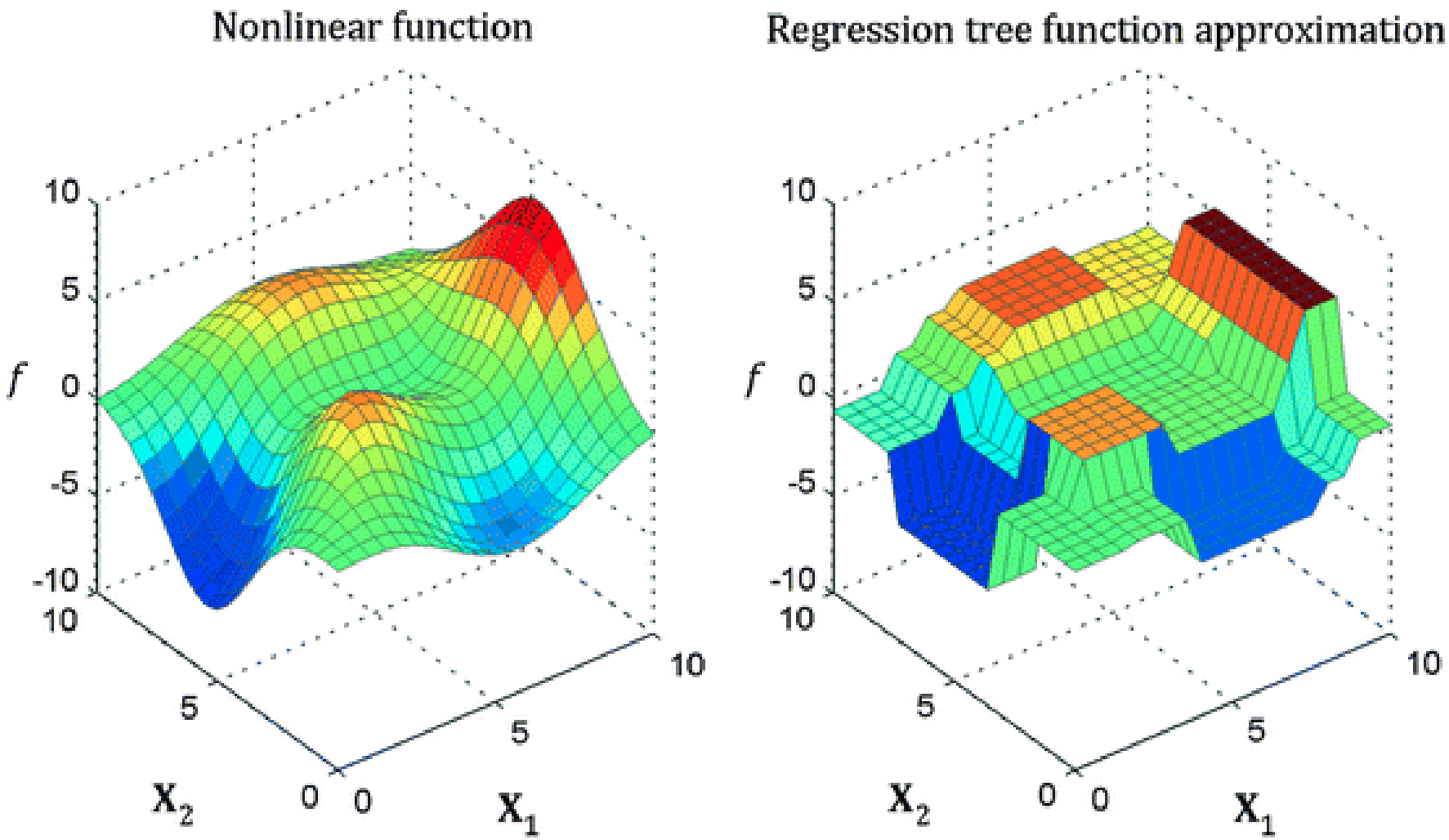
- Instancias relativas al *nodo terminal* R_i serán predichas con la media (mediana) dentro de R_i, cantidad que denotamos como c_i.





$$\widehat{f}(\mathbf{x}) = c_1 \mathbb{I}(\underbrace{x_1 \leq t_1 \& x_2 \leq t_2}_{\text{interacciones}}) + \cdots + c_5 \mathbb{I}(x_1 > t_3 \& x_2 > t_4) = \sum_{m=1}^{5} c_m \mathbb{1}_{[R_m]}(\mathbf{x})$$

Intuición respecto del modelo



- Aproximación continua a trozos de la verdadera f. El algoritmo aprenderá las particiones en el 'espacio de features' con datos.

Aprendizaje de reglas/parámetros

- Restricción computacional:
  - No es factible explorar de forma exhaustiva todas las posibles combinaciones de reglas asociadas al espacio de covariables.
- Binary & Greedy algorithm:
  - Particiones binarias recursivas.
  - Particiones hechas en una etapa anterior no admiten revisión.
- Métricas de aprendizaje (funciones de riesgo):
  - Regresión: Suma de cuadrados residuales.
  - Clasificación: Tasa de error, índice de Gini, Entropía.
- Cada regla (nodo de decisión) puede ser interpretado como un **parámetro** del modelo (complejidad = tamaño del árbol).
- Discutimos el mecanismo de aprendizaje de reglas (regresión).

Dada $S_n = \{(y_1, x_{11}, x_{21}), \ldots, (y_n, x_{1n}, x_{2n})\}$ llamamos:

$$R_L(j,s) = \{(y, x_1, x_2) \in S_n | x_j \leq s\} \text{ y } R_D(j,s) = \{(y, x_1, x_2) \in S_n | x_j > s\},$$

para $j = 1, 2$ y $s \in \mathbb{R}$. Definimos la función de **riesgo empírico** como:

$$\text{RSS}(j, s, \widehat{y}_L, \widehat{y}_D, S_n) = \Big[\sum_{i \in R_L} (y_i - \widehat{y}_L)^2 + \sum_{i \in R_D} (y_i - \widehat{y}_D)^2 \Big]$$

El algoritmo optimiza esta cantidad en cada nodo del árbol:

$$\min_{j,s,\widehat{y}_L,\widehat{y}_D} \text{RSS}(j, s, \widehat{y}_L, \widehat{y}_D) = \min_{j,s} \Big[\min_{\widehat{y}_L} \sum_{i \in R_L} (y_i - \widehat{y}_L)^2 + \min_{\widehat{y}_D} \sum_{i \in R_D} (y_i - \widehat{y}_D)^2 \Big]$$

Para todo (j, s), resulta óptimo predecir con la media en R_L y R_D:

$$\min_{j,s,\widehat{y}_L,\widehat{y}_D} \text{RSS}(j, s, \widehat{y}_L, \widehat{y}_D) = \min_{j,s} \Big[\sum_{i \in R_L} (y_i - \overline{y}_L(j,s))^2 + \sum_{i \in R_D} (y_i - \overline{y}_D(j,s))^2 \Big]$$

- (j^*, s^*) se determinan de manera heurística (ESL: § 9.2.4).
- Repetimos el procedimiento en los nodos terminales R_L y R_D.

Overfitting

- Optimización naive: El árbol crece para minimizar la RSS hasta tener 1 observación en cada uno de sus n nodos terminales.
  - El error en la muestra de entrenamiento es 0, pero a la hora de predecir lo haremos muy mal! (un modelo demasiado ajustado a los datos particulares de la muestra de entrenamiento).
- Early stopping: Para evitar el overfitting, una estrategia razonable de entrenamiento consiste en restringir el crecimiento del árbol.
  - Particionar un nodo sólo cuando el decremento en la RSS que se produce al partirlo es más grande que un cierto umbral predefinido.
  - Permitir particionar un nodo sólo cuando la cantidad de observaciones en éste es superior a cierta cantidad predefinida.
- Estas estrategias son poco eficaces en la práctica. Conviene utilizar técnicas de regularización (similar al planteo de Lasso).

- Llamemos $|T|$ a la cantidad de nodos terminales del árbol T:

$$\mathrm{RE}_\alpha(T) = \underbrace{\sum_{t=1}^{|T|} \sum_{i \in R_t} (y_i - c_i)^2}_{\text{Bias}} + \alpha \underbrace{|T|}_{\text{Var.}},$$

- $|T|$ mide la complejidad del modelo ("cantidad de parámetros").

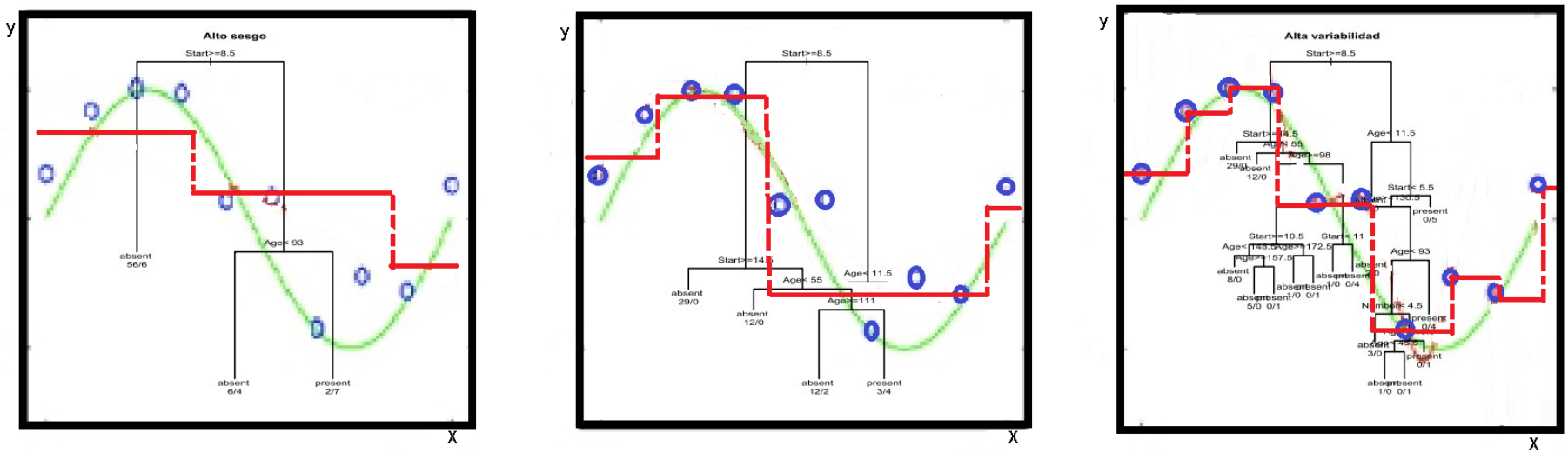


Figure: En azul $\{x_i, y_i\}_{i=1}^n$, en verde f y en rojo $\hat{f}$. Izquierda α grande (underfitting), derecha α pequeño (overfitting) y medio α adecuado.

- ¿Cómo elegimos α en la práctica?

Weakest link pruning

- Denotemos con $T_n(x, \theta)$ al árbol maximal.
  - Un árbol muy extenso con eventualmente 1 dato en cada nodo terminal; en este caso $|T_n| = n$ (en la práctica $|T_n| \ll n$).
- Consideremos $\mathcal{M} = \{T_n, T_{n-1}, \ldots, T_0\}$ la secuencia de árboles que se forman al colapsar sucesivamente los nodos de decisión de T_n en un orden tal que se produce (sucesivamente) el menor incremento en la RSS hasta alcanzar el nodo raíz (T_0).
- Definamos $T_\alpha \equiv \arg\min_T \mathrm{RE}_\alpha(T)$ (ver definición en slide anterior), se puede demostrar que $T_\alpha \in \mathcal{M}$ para todo $\alpha \geq 0$.
  - No tenes que aprender un modelo *nuevo* para cada posible α.
- Utilizamos VC para aprender α y seleccionar $T^* \equiv T_{\alpha^*}$.
  - Por ejemplo como indica la próxima slide.

Algorithm 8.1 *Building a Regression Tree*

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α.

3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:

   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.

   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

   Average the results for each value of α, and pick α to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of α.

Figure: Fuente: ISL capítulo 8.1.

- El paquete `rpart` tiene implementada esta rutina para la calibración de α parametrizado con el nombre cp $\in (0, 1)$.

Criterios de parada para construir T_n

En la práctica hacemos que[1] $|T_n| \ll n$ y $\mathcal{M}$ no resulta demasiado extenso. Criterios habitualmente utilizados en la práctica:

- Controlar cantidad mínima de observaciones en nodos terminales.
  - En `rpart` usamos `rpart.control(minbucket)`.
- Restringir la cantidad de nodos terminales de T_n.
  - En `rpart` usamos `rpart.control(maxdepth)`.
- No se permiten partir nodos que no produzcan reducciones del riesgo empírico superiores a cierto umbral mínimo de referencia.
  - En `rpart` usamos `rpart.control(cp)`.
- Restringir la creación de nodos de decisión cuando se producen dos sub–grupos asimétricos (no disponible en `rpart`).
- Explorar `rpart.control` de la librería `rpart` para más opciones.

[1] No tiene mucho sentido considerar modelos tan extensos, que con alta probabilidad hacen overfitting. De esta forma reducimos el costo computacional.

Midiendo la contribución de cada co–variable

- Para un árbol T de regresión con N_T nodos de decisión, se calcula la importancia de cada una de las p–covariable como:

$$\mathcal{I}_j^2(T) = \sum_{i=1}^{N_T} \widehat{\nabla \text{RE}}_i^2 \mathbb{1}_{[v_i]}(j), \text{ para } j = 1, \ldots, p.$$

- v_i indica que variable se utilizó para hacer el corte en el nodo i de T, con $i = 1, \ldots, N_T$ (barremos los nodos de decisión del árbol).
  - $\mathbb{1}_{[v_i]}(j) = 1$ si el nodo de decisión i de T se utilizó el feature j.
- En clasificación $\widehat{\nabla \text{RE}}_i$ es una estimación de la disminución de Gini o Entropía del modelo después del corte estimado en el nodo i.
- Esta estimación será aún más relevante cuando *ensamblemos* modelos de árbol (Bagging, Random Forest, Boosting Machines).

Agenda

Árboles de regresión y clasificación

- Introducción y contexto
- Árboles de regresión
- CART en `R` y ejemplos
- Árboles de clasificación
- Consideraciones finales

Taller de trabajo en clase: CHURN en Teleco

Algoritmo CART en R.

```
library(rpart)
rpart(formula, data, weights, subset,
                method, parms, control, cost)
```

- formula & data: caracterización del modelo y origen de los datos.
- weights: importancia de cada observación (1/n por defecto).
- subset: subconjunto de filas para entrenamiento.
- method: "anova" (regresión) y "class" (clasificación).
- params: función de pérdida (EC y Gini por defecto).
- control: parámetros de gestión del crecimiento del árbol.
  - cp: cambio mínimo en la función de riesgo para particionar.
  - minsplit & minbucket: número mínimo de observaciones en nodo antes de cortar y despúes de cortar respectivamente.
- cost: peso de cada variable en las reglas de corte.

Caso de estudio (regresión)

Agenda

Árboles de regresión y clasificación

- Introducción y contexto
- Árboles de regresión
- CART en R y ejemplos
- Árboles de clasificación
- Consideraciones finales

Taller de trabajo en clase: CHURN en Teleco

Árboles de **clasificación**

- $Y \in \{y_1, \ldots, y_C\}$, es decir que Y asume valores en un conjunto de $C \geq 2$ categorías diferentes (si $C > 2$, problema multiclase).
- Sobre el aprendizaje de reglas del modelo:
  - Medimos el riesgo empírico en cada nodo de decisión y cada nodo terminal con: Tasa de Error, el Índice de Gini ó la Entropía.
  - A continuación hacemos referencia a los nodos terminales; sin embargo las mismas métricas se emplean en los nodos de decisión a la hora de determinar con que feature y en que nivel del mismo resulta más adecuado hacer el corte (estimar la regla).
- El árbol T proveerá en cada nodo terminal R_t una estimación $\widehat{P}(Y = y_c | X \in R_t) = \widehat{p}(t, c)$, con $c = 1, \ldots, C$ y $t = 1, \ldots, |T|$.

$\widehat{p}(t, c)$ = Porcentaje de observaciones para las que $y = y_c$ en el nodo R_t.

- El riesgo empírico total del árbol T se computa agregando los riesgos en cada nodo terminal (ponderando por su peso relativo):

$$\text{RE}(T) = \sum_{t=1}^{|T|} \frac{n_t}{n} \text{RE}_t,$$

donde RE_t y n_t indican el riesgo empírico y número de observaciones en el nodo terminal t. Medidas de impureza:

  - **Tasa de Error**: Proporción de observaciones mal clasificadas

$$\text{RE}_t = 1 - \max\{\hat{p}(t,1), \ldots, \hat{p}(t,C)\}.$$

  - **Gini**: Varianza en cada nodo terminal

$$\text{RE}_t = \sum_{c \neq c'} \hat{p}(t,c)\hat{p}(t,c') = \sum_{c=1}^{C} \hat{p}(t,c)(1 - \hat{p}(t,c)).$$

  - **Entropía**: Divergencia entre la distribución de las clases en nodo

$$\text{RE}_t = -\sum_{c=1}^{C} \hat{p}(t,c) \log \hat{p}(t,c)$$

Comparativa de métricas de riesgo

Entropía = Medida de incertidumbre (Teoría de la Información).

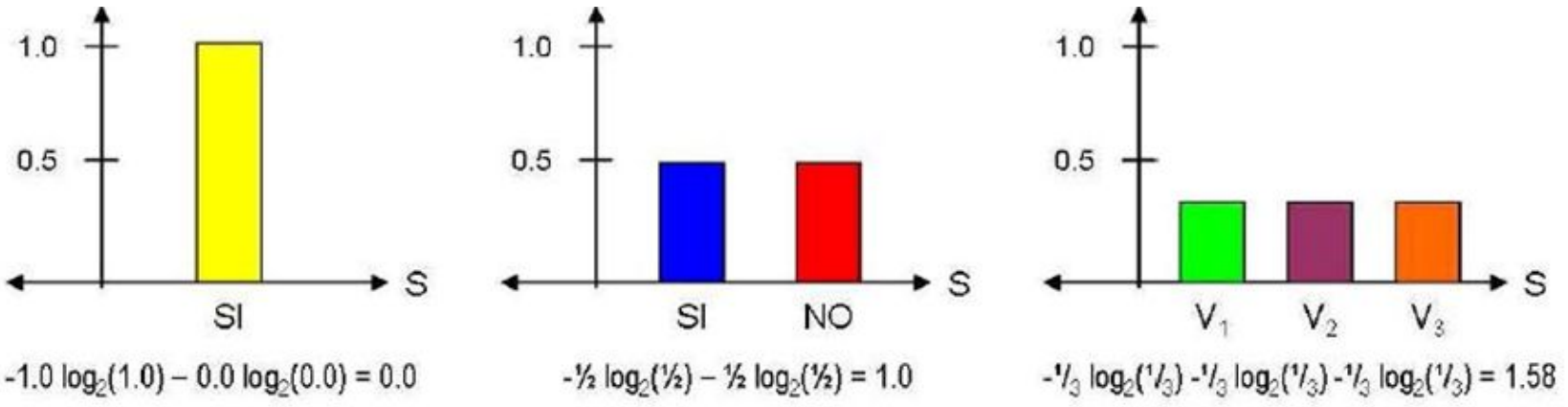
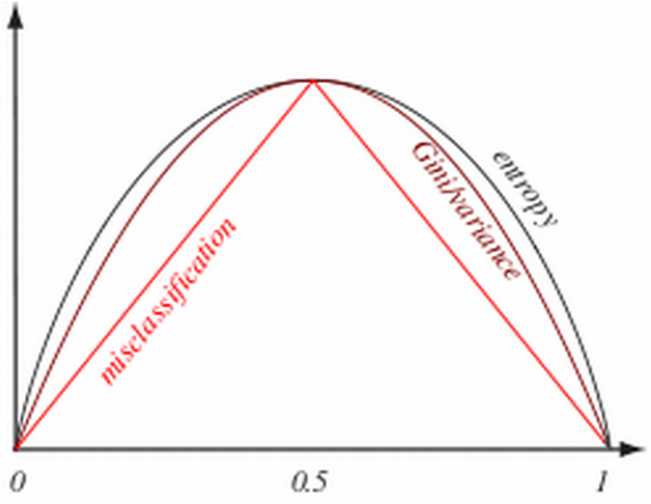


Figure: Valores de la entropía para 3 distribuciones diferentes.

- Gini o Entropía para hacer crecer el árbol y tasa de error para podar el árbol.



Ejemplo

- Consideremos $Y \in \{0, 1\}$, para cada $X \in (X_1, \ldots, X_p)$ y $s \in \mathbb{R}$:

Nodo	Aciertos	Errores	Total
$A_L(X \leq s)$	$a(L, s)$	$e(L, s)$	$n(L, s)$
$A_R(X > s)$	$a(R, s)$	$e(R, s)$	$n(R, s)$
Total	$A(s)$	$E(s)$	$n(\cdot, s)$

Table: Matriz de confusión para la regla "$X \leq s$".

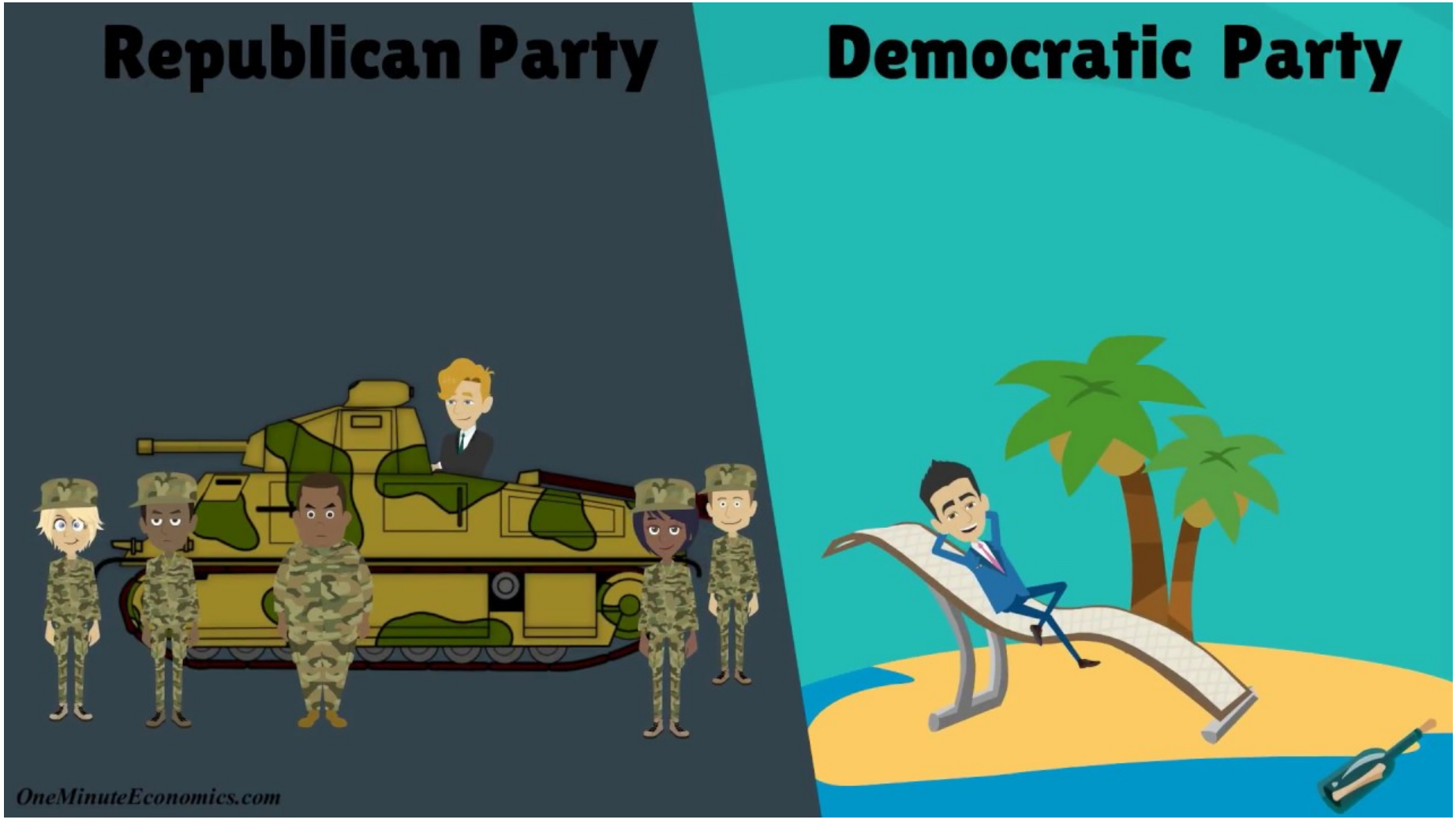
- Llamemos $p(j, s) = n(j, s)/n(\cdot, s)$ con $j = L, R$ y definamos

$$H(j, s) = \frac{a(j, s)}{n(j, s)} \log\left(\frac{a(j, s)}{n(j, s)}\right) + \frac{e(j, s)}{n(j, s)} \log\left(\frac{e(j, s)}{n(j, s)}\right) \text{ para } j = L, R$$

- Buscamos $X \in (X_1, \ldots, X_p)$ y $s \in \mathbb{R}$ que:

$$\min_{X,s} p(L, s)H(L, s) + p(R, s)H(R, s).$$

Caso de estudio (clasificación)



Agenda

Árboles de regresión y clasificación

- Introducción y contexto
- Árboles de regresión
- CART en R y ejemplos
- Árboles de clasificación
- Consideraciones finales

Taller de trabajo en clase: CHURN en Teleco

Modelos Lineales vs Árboles

- Los modelos lineales plantean:

$$E[Y|X_1, \ldots, X_p] \approx \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p.$$

- En clasificación, el modelo logístico plantea:

$$E[Y|X_1, \ldots, X_p] \approx \text{Logistic}(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p).$$

- Los CART se pueden interpretar como modelos *locales y no lineales* para la media condicional $E(Y|X_1, \ldots, X_p)$, donde:

$$E(Y|X_1, \ldots, X_p) \approx \sum_{i=1}^{|T|} w_i \mathbb{1}_{[R_i]}(X_1, \ldots, X_p),$$

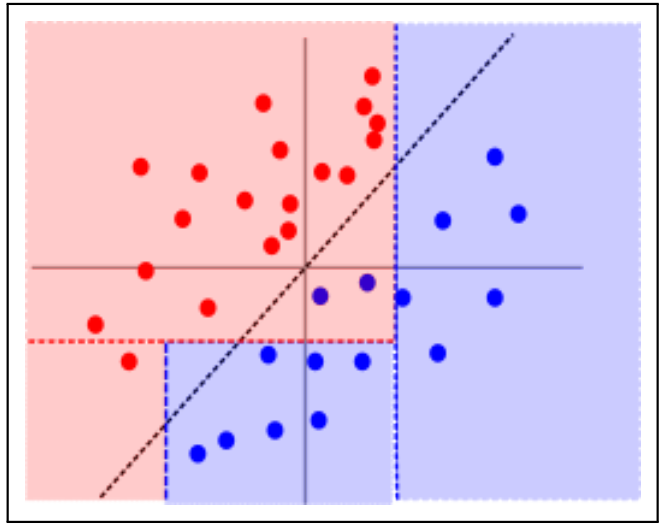
donde w_i es una estimación local de $E[Y|X_1, \ldots, X_p]$ que construimos utilizando las observaciones del nodo terminal R_i que a su vez no depende de forma lineal de los datos.

Ventajas de los árboles

- Interpretabilidad: Se pueden representar gráficamente, resultan intuitivos y comprensibles (parámetros = reglas).
- Pueden combinarse variables cuantitativas y cualitativas sin demasiado pre–proceso de datos.
- Outliers (datos extremos) y datos perdidos no afectan considerablemente la estimación de las reglas del modelo de árbol.
  - Estrategia de reglas subrogadas para datos perdidos.
- No hay especificaciones fuertes en la modelización:
  - La colinealidad, autocorrelación, y/o la heterocedasticidad no afectan al mecanismo de aprendizaje de reglas.
- Ingeniería de atributos: Tramear features continuos, incluír transformaciones, interacciones, componentes principales, etc.

Sobre las limitaciones de este modelo

- Poco efectivo para aproximar relaciones lineales.



- Modelo "sensibles" a los datos de entrenamiento:

$$\text{Bias}^2 + \textbf{Var} + \sigma^2$$

- ... sin embargo existen métodos de "agregación" (ensamble) que logran reducir la variabilidad de estos modelos y los vuelven competitivos: Bagging, Random Forest, Boosting Machines.

Agenda

Árboles de regresión y clasificación

Taller de trabajo en clase: CHURN en Teleco

CHURN en Telecomunicaciones

TRAIN: matriz de 3.333 filas (cada una representa un cliente) y 20 columnas (variables).

Variables predictorias (19): state, accountlength, area_code international_plan(yes/no), voice_mail_plan(yes/no), number_vmail_messages, total_day_minutes, total_day_calls, total_day_charge, total_eve_minutes, total_eve_calls, total_eve_charge, total_night_minutes, total_night_calls, total_night_charge, total_intl_minutes, total_intl_calls, total_intl_charge and number_customer_service_calls.

Variables a predecir: churn (yes/no).

Consignas:

- Construye un modelo de árbol para predecir la fuga de clientes utilizando los datos de TRAIN.
  - Tienes disponible un fichero `R` para cargar datos de train y test.
  - Estima el tamaño adecuado del árbol utilizando VC.
- Utilizando como benchmark el modelo de regresión logística de la sesión anterior y sobre el conjunto TEST, computa:
  - Los errores tipo I (falsos negativos) y II (falsos positivos) de ambos modelos.
  - Computa el AUC de ambos modelos.
  - Reflexiona sobre las que consideras ventajas y desventajas de cada uno de los dos modelos anteriores.