

Aprendizaje Estadístico Supervisado

Natalia da Silva

2024

Ya vimos

- Árboles de decisión
- Métodos de agregación/ensembles (Bagging y Random Forest)

Veremos

- Algunas extensiones de árboles
- Extensiones de Random Forest
- Otro método de agregación, Boosting

Recap Árboles de decisión

- Fifty Years of Classification and Regression Trees ([Loh 2014](#))
- Los modelos basados en árboles consisten en una secuencia de particiones anidadas que dividen el espacio de los predictores donde en cada partición se usa un modelo para predecir la respuesta.
- En cada paso del árbol se establecen decisiones de tipo si pasa A entonces B.
- Los árboles de decisión se pueden agrupar por el número de variables predictoras utilizadas en cada partición de nodos.
- Los árboles que usan sólo una variable en cada partición de los nodos producen particiones paralelas a los ejes, mientras que cuando usan muchas variables para la partición producen particiones oblicuas.

Recap Árboles de decisión

- En teoría, las regiones que define un árbol podrían ser de cualquier forma.
- Nosotros seleccionamos CART (para regresión y clasificación) donde se divide el espacio de los predictores en rectángulos en altas dimensiones.

Recap Árboles de decisión

- Un árbol puede ser visto como un conjunto de reglas de decisión que definen particiones del espacio de las predictoras. El valor esperado de la variable de respuesta Y puede ser definido como:
- Donde representa una partición en el espacio de las predictoras tal que con y .

Si entonces el valor predicho para es .

es calculado diferente si el problema es de regresión o de clasificación

Recap Árboles de decisión

Para problemas de clasificación denota la clase para cada observación. El valor esperado para cuando es la clase más frecuente in la partición , i.e.

Para problemas de regresión el valor predicho es:

Recap Árboles de decisión

Finalmente para un conjunto de datos dato el valor predicho para la respuesta puede ser definido como

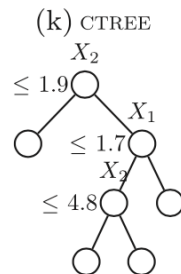
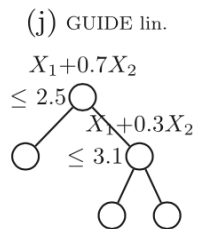
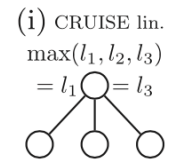
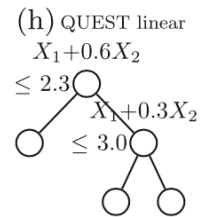
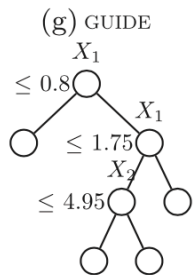
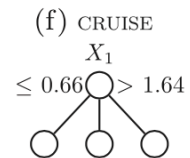
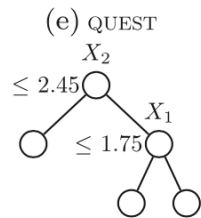
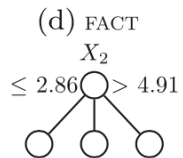
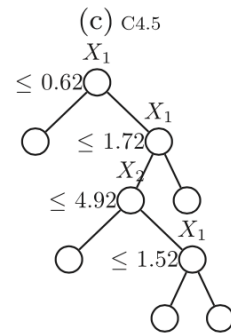
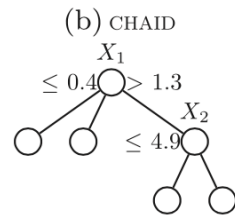
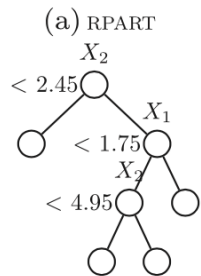
Diferente para regresión y clasificación:

- Regresión: Promedio de la respuesta en la hoja
- Clasificación: Clase más frecuente de la respuesta en la hoja

Variantes de CART

- CHAID, C4.5, FACT, QUEST, CRUISE, GUIDE, CTREE, PPtree, etc.
- Principales diferencias entre los distintos métodos son la forma en que se parte el nodo.
- Algunos usan métodos de kernel, vecino más cercano, particiones lineales en un subconjunto de variables seleccionadas.
- Pueden ser árboles binarios o con particiones múltiples.
- Los árboles que usan una sola variable en la partición del nodo generan particiones paralelas a los ejes, cuando se usan más variables las particiones tienden a ser oblicuas.

Variantes



Variantes

- Chaid (CHi-squared Automatic Interaction Detector aproximación) similar a regresión stepwise para el criterio de selección. Para buscar una variable X para dividir un nodo, este último se divide inicialmente en dos o más nodos hijos, cuyo número depende del tipo de variable. Disponible sólo en software comercial.
- C4.5 para clasificación, si X tiene m valores distintos en un nodo las particiones son en m nodos hijos, un nodo por hijo. Si X es ordenada los nodos se parten en la forma habitual en dos nodos () usa una medida de impureza basada en entropía, ratio de ganancia.

Variantes

- C4.5 es que inducen sesgo en la selección de variables. Si y con y distintos valores es más probable seleccionar la que tiene más valores distintos.
- Hay algunos algoritmos que intentan superar este problema, CRUISE, QUEST, CTREE

Variantes

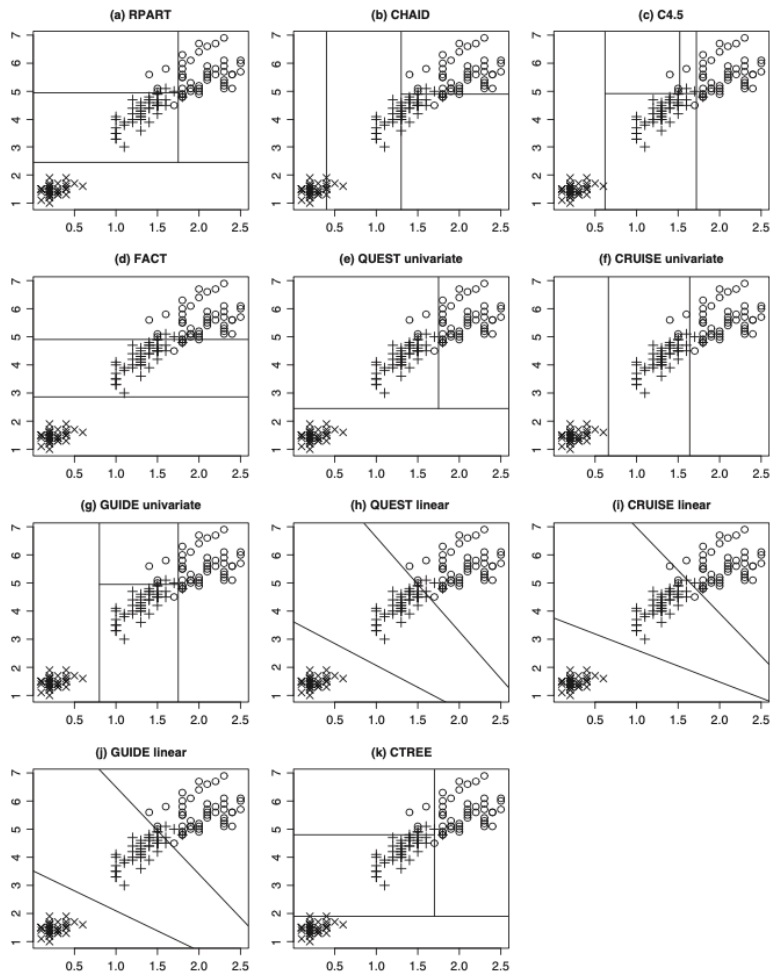


Figure 3. Plots of petal length versus petal width with classification tree partitions; *Setosa*, *Versicolour*, and *Virginica* are marked by triangles, circles, and crosses, respectively.

<https://pages.stat.wisc.edu/~loh/research.html>

Projection pursuit Trees

Les voy a contar sobre PPtree que combina concepto de árboles con optimización de índices de proyección Lee et al. (2013)

Projection Pursuit

- PP es una técnica estadística para exploración para encontrar proyecciones de datos que revelen estructuras **interesantes** en algún sentido.
- Los algoritmos de PP buscan proyecciones de bajas dimensiones optimizando un índice de proyección que mide si la proyección es útil/interesante en algún sentido.
- Pocos PP índices incorporan información de las clases para el cálculo.
- Lee et. al. proponen un índice derivado del análisis discriminante lineal útil para exploración en clasificación supervisada.

PP_LDA

$$I_{LDA}(A) = \begin{cases} 1 - \frac{|A^T W A|}{|A^T (W + B) A|} & \text{for } |A^T (W + B) A| \neq 0 \\ 0 & \text{for } |A^T (W + B) A| = 0 \end{cases}$$

donde A es una matriz y define una proyección ortonormal de d -dimensiones a un subespacio d -dimensional, W es la suma de cuadrados entre grupos, B es la suma de cuadrados al interior de los grupos donde μ_k es el vector de medias de los grupos y μ vector de medias global. Si el índice LDA tiene valores altos hay una gran diferencia entre clases y no de lo contrario.

PP_PDA

Cuando las variables est'an altamente correlacionadas en es cercano a cero y no funciona bien.

$$I_{PDA}(A, \lambda) = 1 - \frac{|A^T W_{PDA} A|}{|A^T (W_{PDA} + B) A|}$$

misma notación que en , agregando que es un parámetro de contracción y usamos una suma de cuadrados al interior de grupos diferente, .

Projection pursuit

- PP quiere encontrar todas las proyecciones interesantes.
- No solamente un óptimo global ya que los óptimos locales pueden revelar estructuras interesantes.

PCA como caso particular de PP

- PP selecciono proyecciones de bajas dimensiones en mis datos multivariados optimizando algún índice de proyección “interesante”.
- PCA se puede ver como un caso particular de PP donde el índice de proyección es la varianza de los datos que se maximiza sobre todas las proyecciones.

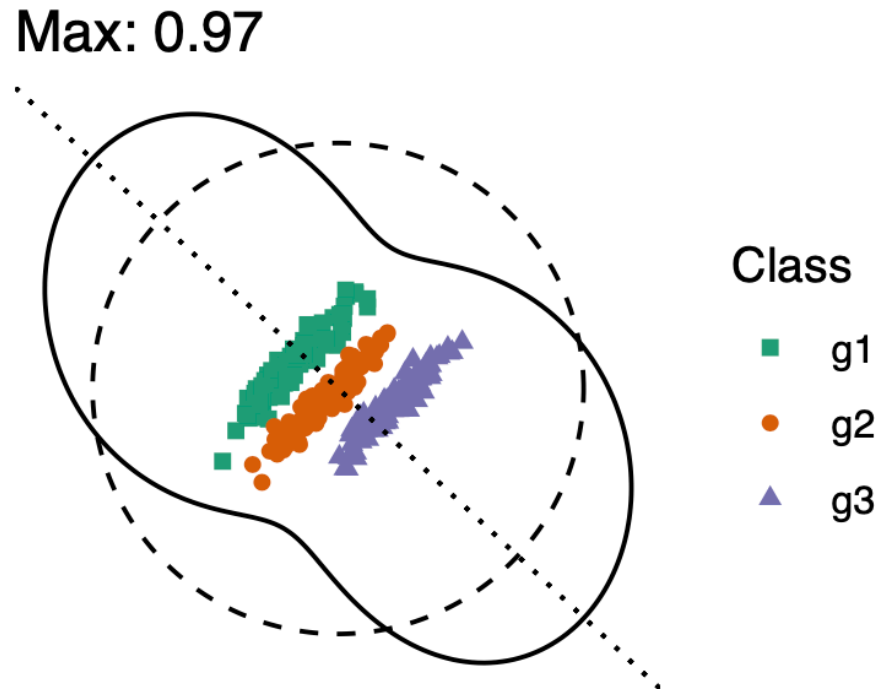
Optimización e y

Para los dos índices de proyección seleccionados se puede encontrar el óptimo teórico.

- La proyección q -dimensional que maximiza , son los primeros q vectores propios de
- La proyección q -dimensional que maximiza , on los primeros q vectores propios de

Huber plot, datos simulados

Huber plot ([huber1990data?](#))

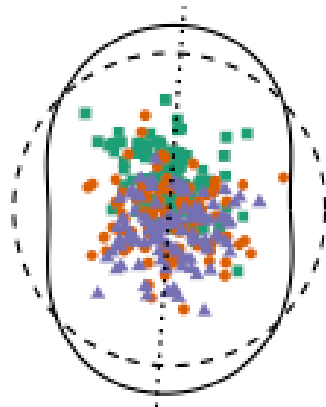


- Muestra el índice PP en todas las posibles direcciones en 2D
- Los índices se calculan usando las proyecciones para
- $=, \dots,$
- Para cada proyección, cálculo el índice en los datos proyectados (línea sólida)
- Círculo punteado es de referencia, mediana de todos los valores del índice. Línea punteada corresponde al máximo índice.

, proyección 1-D

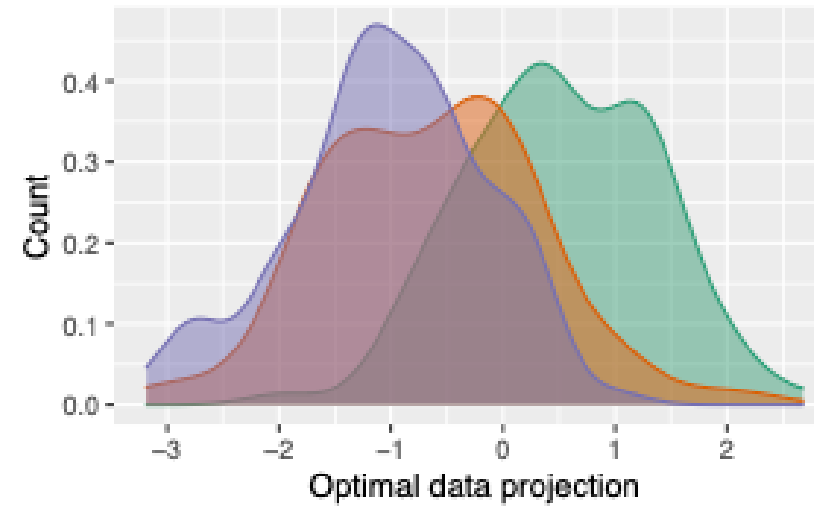
Huber Plot

Max: 0.35

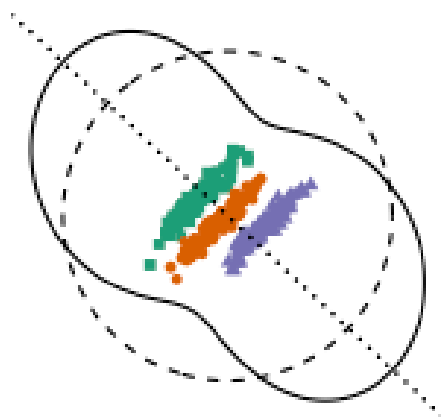


Class

- g1
- g2
- g3

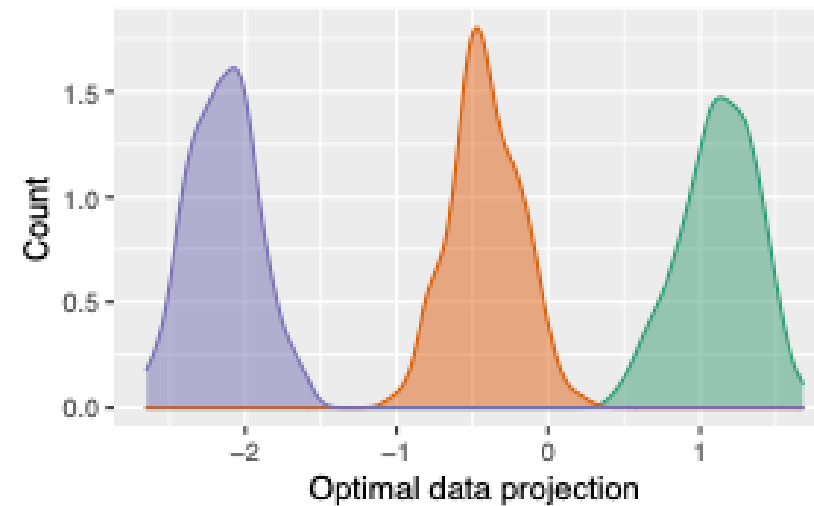


Max: 0.97



Class

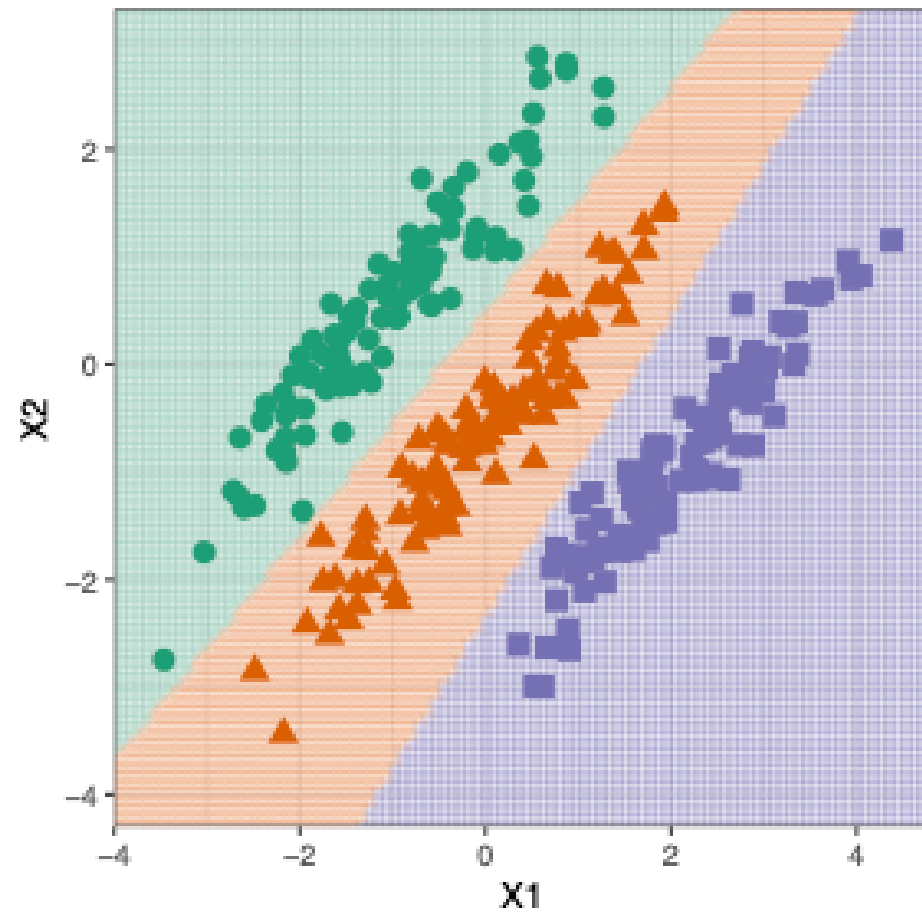
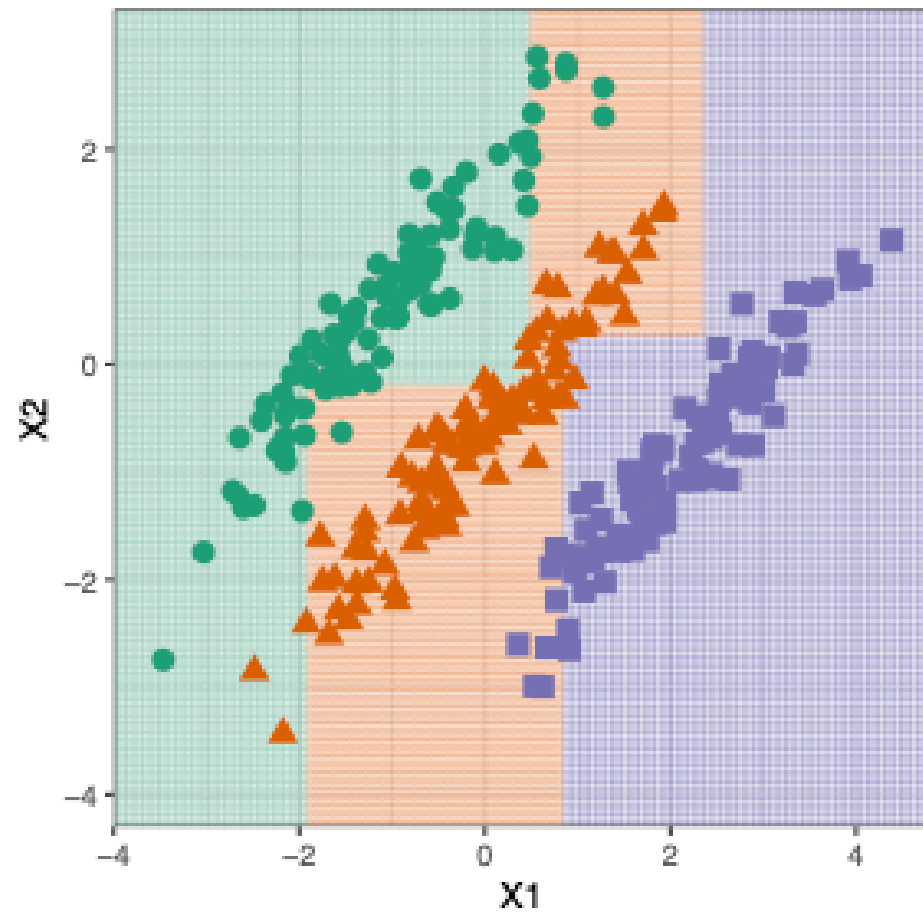
- g1
- g2
- g3



PPtree

- **PPtree** son árboles de clasificación basados en proyecciones.
- Combina métodos de árboles con reducción de dimensionalidad basada en proyecciones.
- Las particiones en **PPtree** se basan en combinaciones lineales de variables por lo que toma en cuenta la correlación entre las variables para separar las clases.

PPtree



PPtree

- Trata los datos siempre como un problema de dos clases.
- Cuando las clases son más de dos el algoritmo usa dos pasos de proyección optimizando un índice de proyección en cada partición del nodo.
- Los coeficientes de proyección en cada nodo representan la importancia de la variables, útiles para explorar como las clases son separadas en cada árbol.

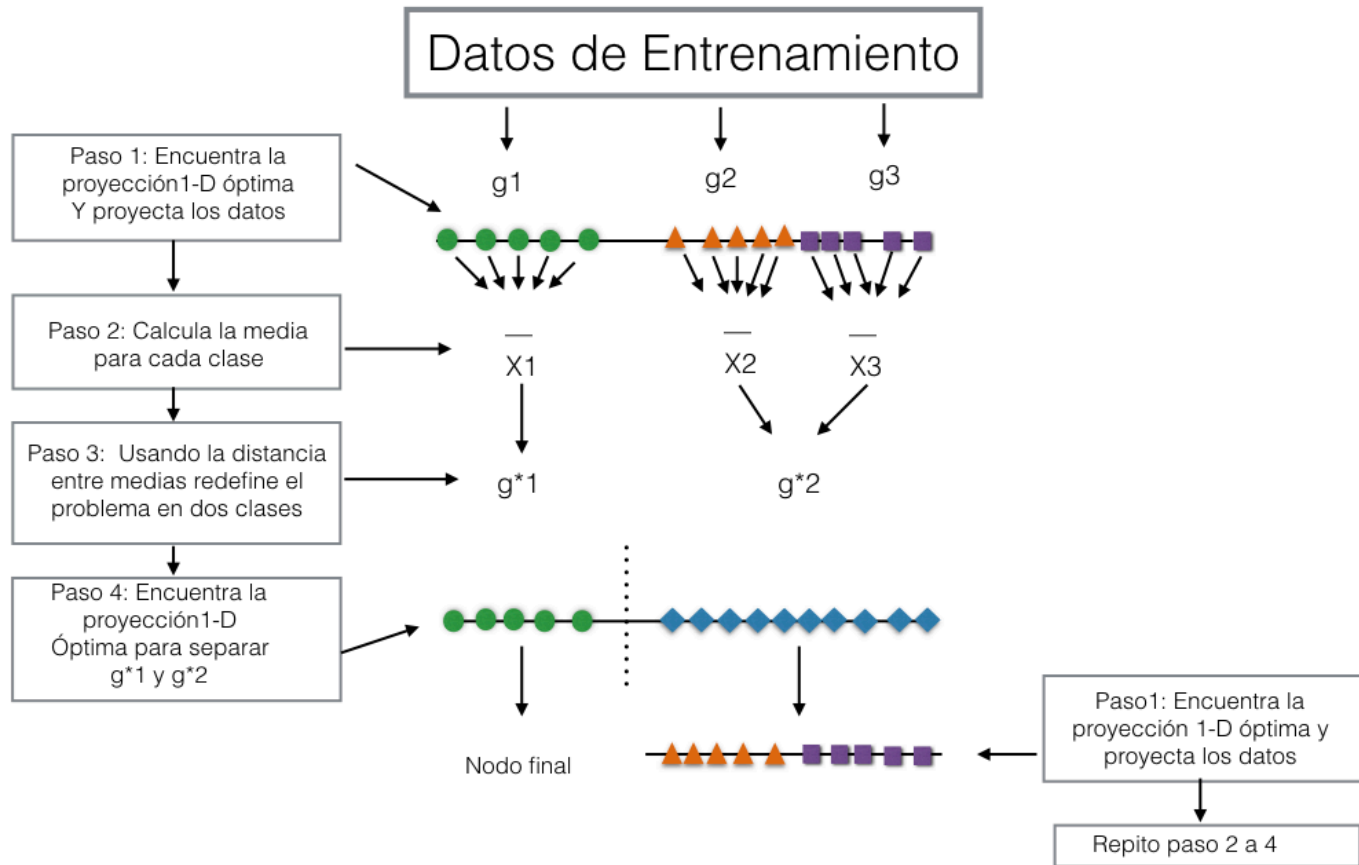
Algoritmo de PPtree

1. Optimiza un índice de proyección para encontrar una proyección óptima, , para separar todas las clases obteniendo los datos proyectados para todo .
2. En los datos proyectados, redefine el problema en uno de dos clases usando la distancia de la media entre clases asigna una nueva etiqueta, o para cada observación, generando una nueva variable para la clase. Notar que esto requiere juntar dos o más grupos en uno, tal que los nuevos grupos y posiblemente tengan más de una clase original.
3. Encuentra una proyección óptima , usando para separar un problema de dos clases y . La mejor separación de y es determinada en este paso proporcionando la regla de decisión para el nodo.
4. si entonces asigna al nodo izquierdo y al nodo derecho,

donde es el vector de medias . El valor de corte en los datos proyectados se calcula como (defecto), donde y son las medias de grupos de los datos proyectados. Hay 8 reglas para calcular .

5. Para cada grupo todos los pasos previos son calculados y repetidos hasta que y tienen solo una clase de la clase original. La profundidad del árbol es como mucho la cantidad de clases.

Ilustración del algoritmo



La profundidad del PPtree es como mucho el número de clases.

PPtree, 8 reglas de corte

$$C = \frac{1}{2}\bar{\mathbf{x}}_1 + \frac{1}{2}\bar{\mathbf{x}}_2$$

$$C = \frac{n_1}{n_1+n_2}\bar{\mathbf{x}}_1 + \frac{n_2}{n_1+n_2}\bar{\mathbf{x}}_2$$

$$C = \frac{s_1}{s_1+s_2}\bar{\mathbf{x}}_1 + \frac{s_2}{s_1+s_2}\bar{\mathbf{x}}_2$$

$$C = \frac{s_2/\sqrt{n_2}}{s_1/\sqrt{n_1}+s_2/\sqrt{n_2}}\bar{\mathbf{x}}_1 + \frac{s_1/\sqrt{n_1}}{s_1/\sqrt{n_1}+s_2/\sqrt{n_2}}\bar{\mathbf{x}}_2$$

$$C = \frac{1}{2}\mathbf{x}_1^{med} + \frac{1}{2}\mathbf{x}_2^{med}$$

$$C = \frac{n_1}{n_1+n_2}\mathbf{x}_1^{med} + \frac{n_2}{n_1+n_2}\mathbf{x}_2^{med}$$

$$C = \frac{IQR_2}{IQR_1+IQR_2}\mathbf{x}_1^{med} + \frac{IQR_1}{IQR_1+IQR_2}\mathbf{x}_2^{med}$$

$$C = \frac{IQR_2/\sqrt{n_2}}{IQR_1/\sqrt{n_1}+IQR_2/\sqrt{n_2}}\mathbf{x}_1^{med} + \frac{IQR_1/\sqrt{n_1}}{IQR_1/\sqrt{n_1}+IQR_2/\sqrt{n_2}}\mathbf{x}_2^{med}$$

Características

- PPtree produce árboles sencillos
- Si existe una banda lineal en los datos, PPtree produce un árbol sin error de clasificación.
- En cada nodo el PPtree separa dos clases usando combinaciones lineales, el número de clases es el mismo que el número de nodos finales por lo que la profundidad del árbol es como mucho $G-1$.
- No necesita ser podado.
- En cada nodo, es posible hacer histogramas de los datos proyectados que puede servir para estudiar la separación. Entonces la razón de la mala clasificación puede ser analizada en cada nodo.
- Los coeficientes de proyección pueden ser usados para medir la contribución de las variables (los datos deben estar estandarizados)

Implementación

```
1 library(PPtreeViz)
2 library(help=PPtreeViz)
3 data(iris)
4 Tree.result <- PPTreeclass(Species~.,data = iris,"LDA")
5 Tree.result
```

```
=====
Projection Pursuit Classification Tree result
=====
```

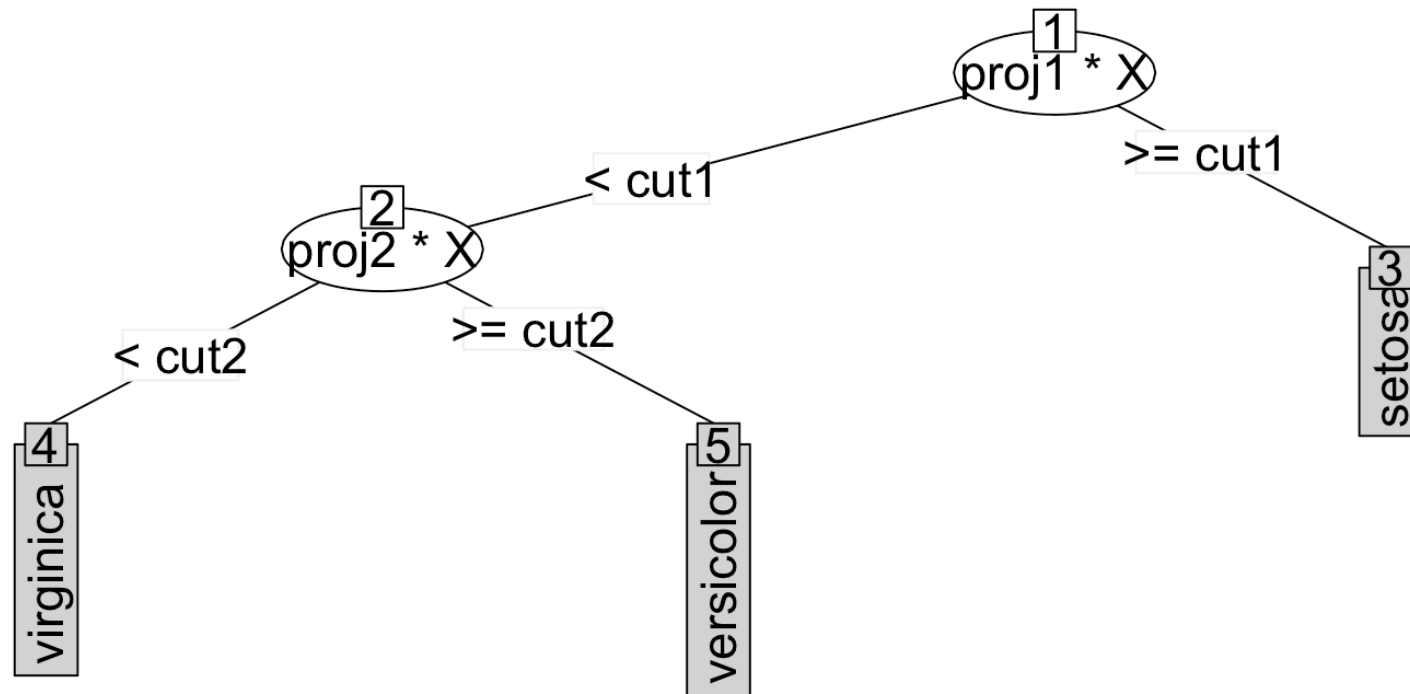
```
1) root
  2) proj1*X < cut1
    4)* proj2*X < cut2 -> "virginica"
    5)* proj2*X >= cut2 -> "versicolor"
  3)* proj1*X >= cut1 -> "setosa"
```

```
Error rates of various cutoff values
-----
```

Implementación

```
1 plot(Tree.result)
```

Projection Pursuit Classification Tree



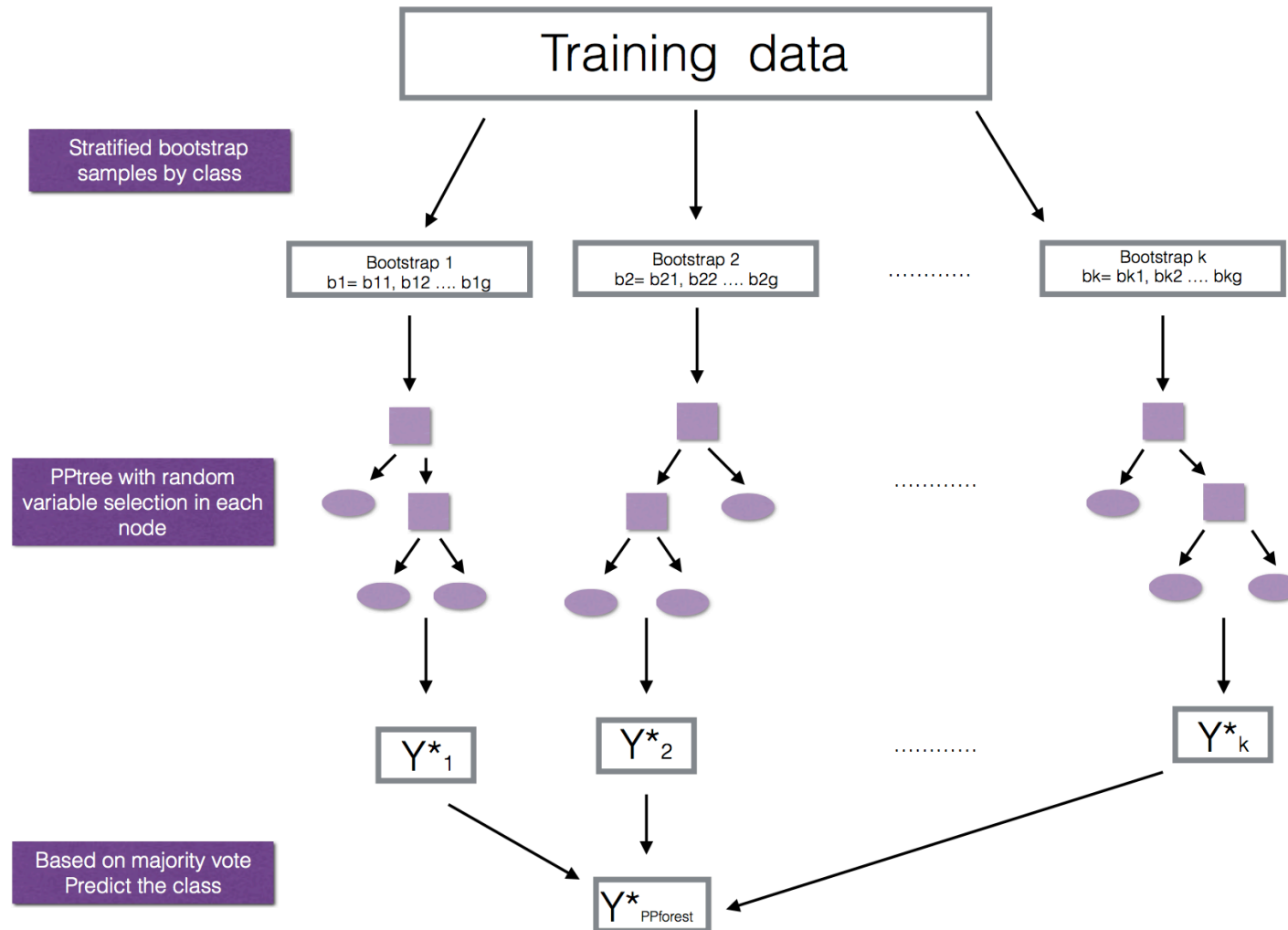
Extensión PPforest

daSilva, Cook, and Lee (2021)

Para problemas de **clasificación**, usa árboles de proyección **PPtree** en vez de CART.

- Usa bootstrap estratificado para atacar desbalance
- Usa combinaciones lineales para separar las clases
- Aprovecha la correlación entre las variables para separar grupos

PPforest diagrama



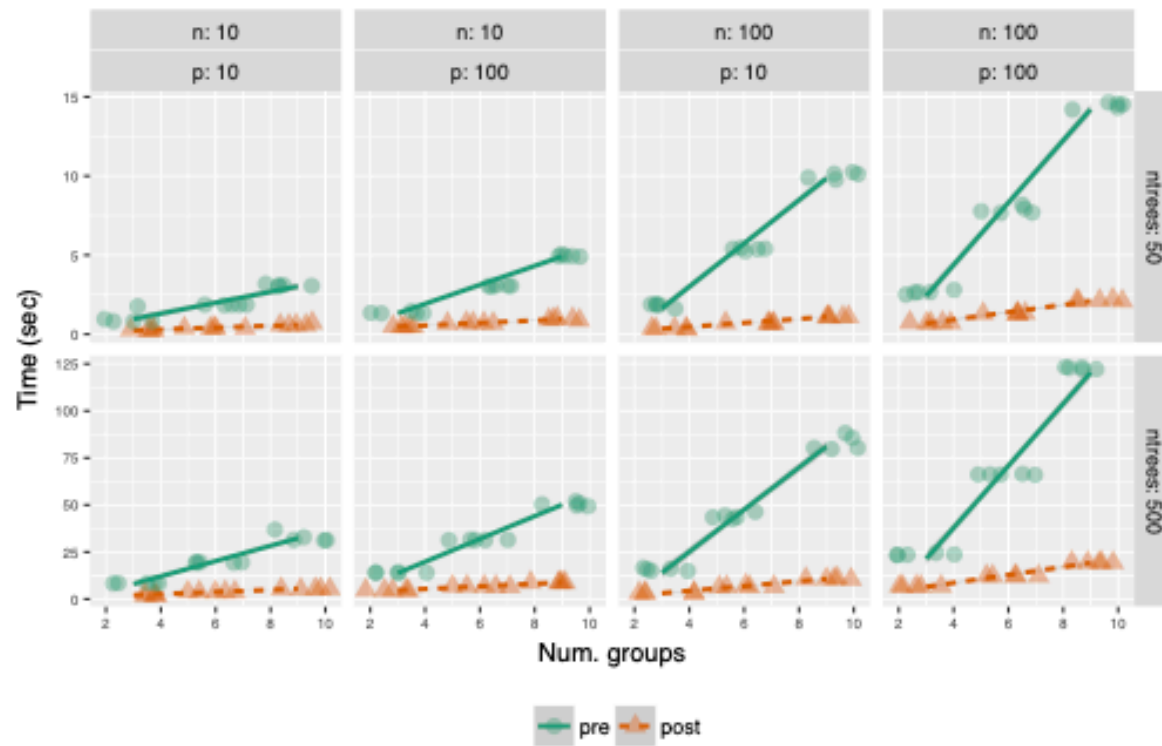
PPforest: Implementación

- Implementado en un paquete de R **PPforest**
- Versión inicial de **PPforest** enteramente en R.

Dos estrategias para optimizar el código fueron empleadas:

- Funciones principales en Rcpp (integración de R con C++)
- Paralelización

PPforest: Tiempo de procesamiento



PPforest implementación

```
1 library(PPforest)
2 pprf.crab<- PPforest(data = crab, class = 'Type',
3   std = FALSE, size.tr = 1, m = 200, size.p = .5,
4   PPmethod = 'LDA' , parallel = TRUE, cores = 2, rule=1)
5 pprf.crab
```

Call:

```
PPforest(data = crab, class = "Type", std = FALSE, size.tr = 1,
m = 200, PPmethod = "LDA", size.p = 0.5, parallel = TRUE,      cores
= 2, rule = 1)
```

Type of random forest: Classification

Number of trees: 200

No. of variables tried at each split: 2

OOB estimate of error rate: 6%

Confusion matrix:

```
BlueFemale BlueMale OrangeFemale OrangeMale class.error
```

APP

vineo

Quantile RF

Meinshausen (2006)

Para problemas de regresión.

- En RF aproximamos como un promedio ponderado de las observaciones de Y .
- El promedio ponderado podría aproximar bien no solo la esperanza condicional sino toda la distribución condicional, en esto se enfoca QRF

De forma similar a RF se puede aproximar como un promedio ponderado de las observaciones

RF vistos como promedios ponderados

En el usamos el promedio de la pedicción de B árboles, donde .

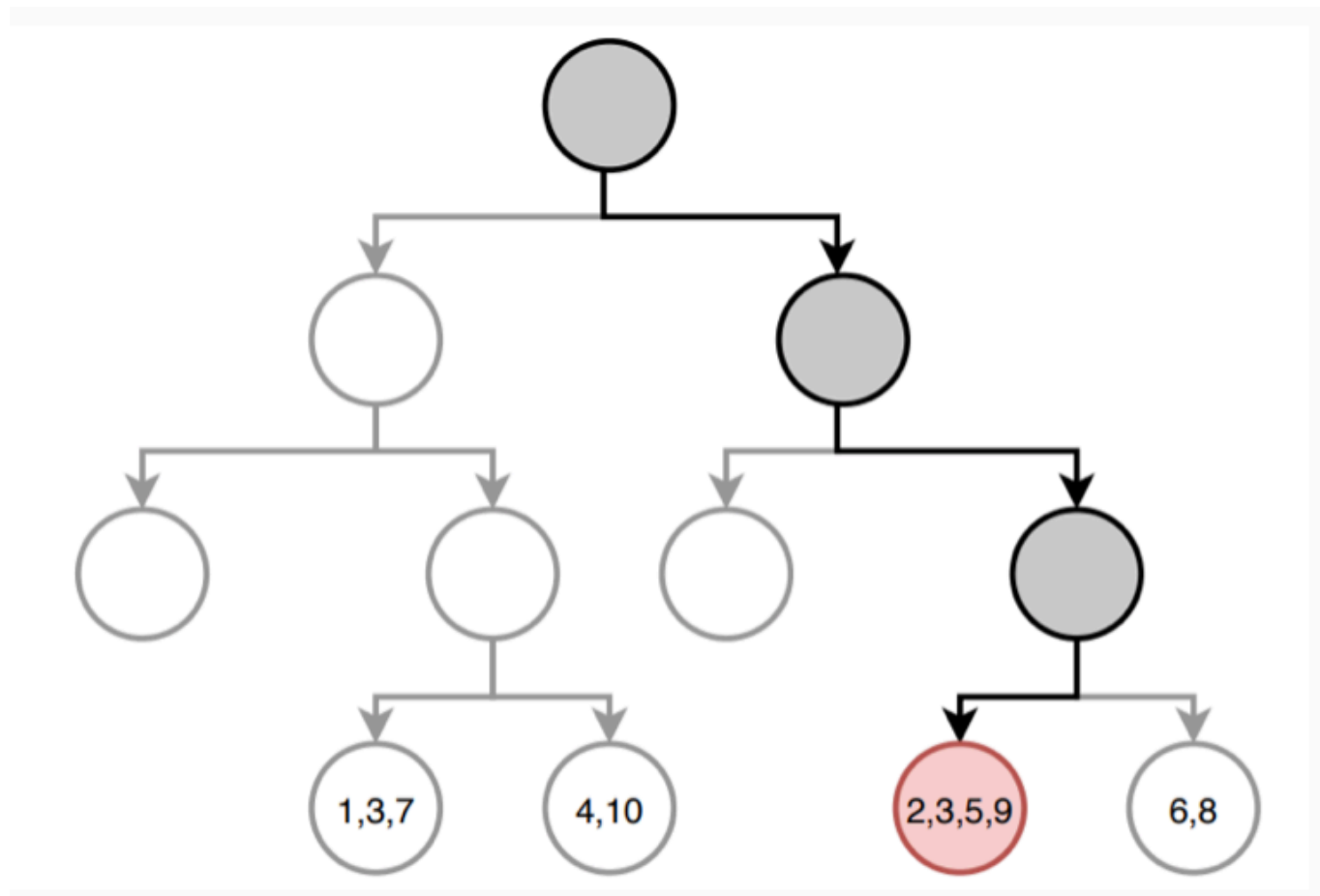
Notar que suman 1 en cada árbol y suma 1 en el bosque.

Ejemplo con 10 observaciones

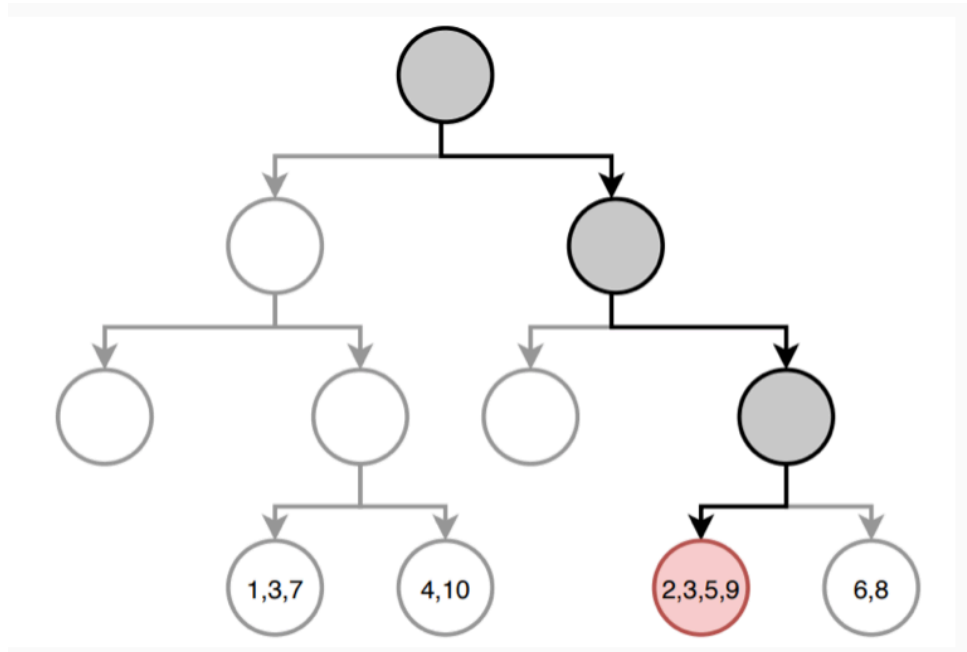
id	1	2	3	4	5	6	7	8	9	10
Y	10	18	24	8	2	9	16	10	20	14
X

Un solo árbol

Predicción con un árbol de regresión para : el camino hasta llegar al nodo final está resaltado. En cada nodo terminal se detalla el índice de las observaciones de entrenamiento que forman parte de la hoja.



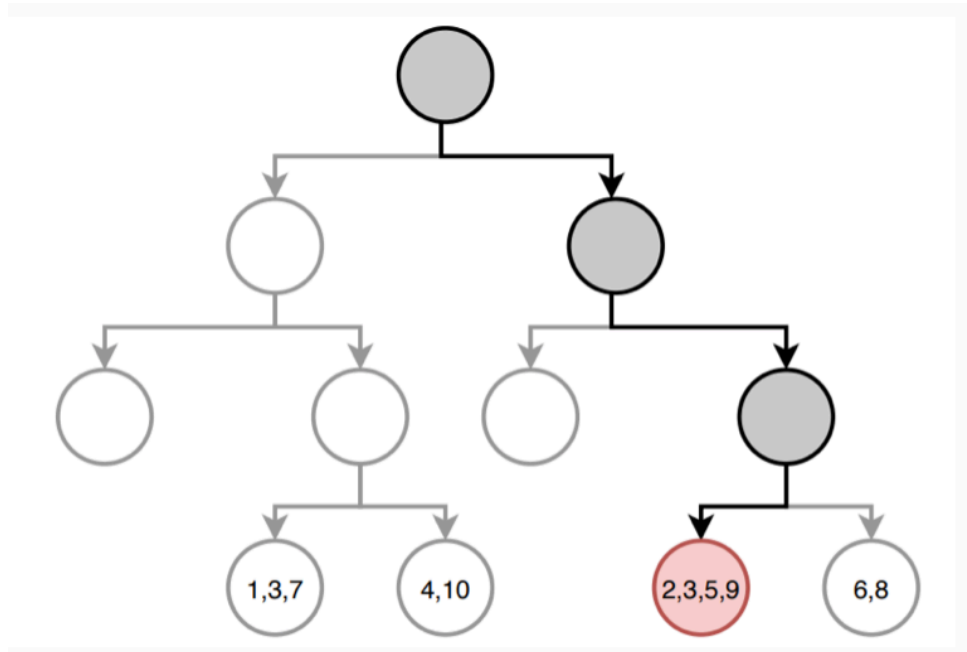
Predicción un árbol de regresión



- id: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
- Y: 10,18,24,8,2,9,16,10,20,14
- El valor predicho por el árbol es la media de la variable respuesta Y de las observaciones con id: 2, 3, 5, 9.
-

Predicción un árbol de regresión

Usando la notación de ponderadores

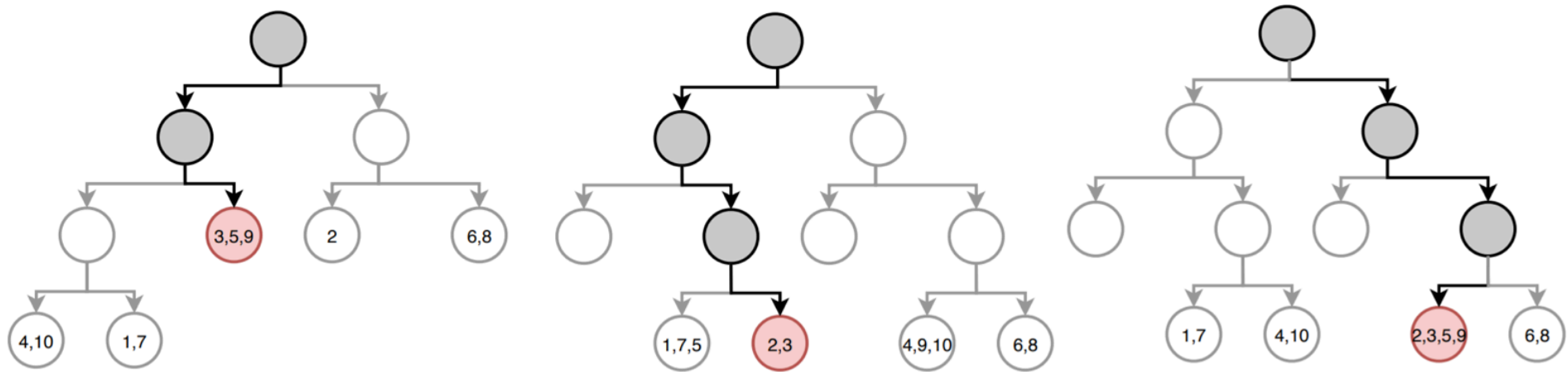


- id: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
- Y: 10,18,24,8,2,9,16,10,20,14
- El valor predicho por el árbol es la media de la variable respuesta Y de las observaciones con id: 2, 3, 5, 9.
-
-
-

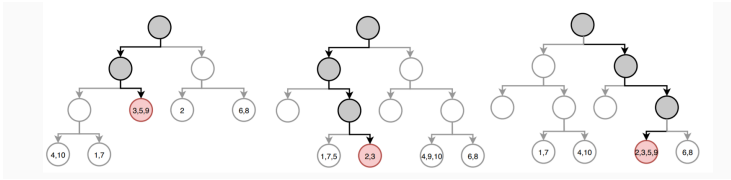
Predicción un árbol de regresión

Predicción del bosque

Predicción con random forest: en cada árbol, el camino hasta llegar al nodo final está resaltado. En cada nodo terminal se detalla el índice de las observaciones de entrenamiento que forman parte de él.



Bosque con tres árboles



- id: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 Y:
10,18,24,8,2,9,16,10,20,14

•

•

•

•

Bosque con tres árboles

Quantile RF

QRF propone estimar la distribución acumulada como:

Estimación empírica de la función de distribución

Algoritmo:

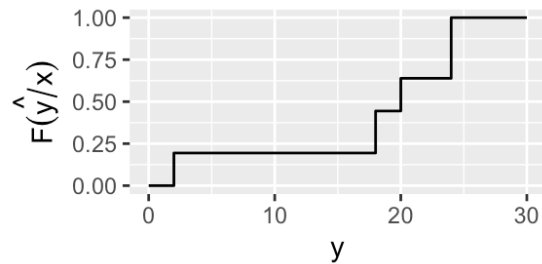
- Mismo algoritmo que RF pero en cada hoja de cada árbol se queda con todas las observaciones no sólo con el promedio
- Para cada se recorren todos los árboles y se calcula los pesos y su promedio
- Estimo la función de distribución como se definió anteriormente para todo

Y se estima el cuantil condicional como

Quantile RF

Bosque con tres árboles

: 2, 8, 9, 10, 10, 14, 16, 18, 20, 24



Una vez obtenida la estimación de la distribución, se puede calcular cualquiera de sus cuantiles.

Ensembles de Árboles

Why do tree-based models still outperform deep learning on typical tabular data? ([grinsztajn2022tree?](#))

Ya mencionamos dos variantes principales:

Bagging (vimos)

- Construir árboles independientes
- Árbol individual grande: poco sesgo, mucha varianza
- Promediar predicciones de cada árbol

Boosting (veremos)

- Construir árboles de forma secuencial
- Árbol individual chico: mucho sesgo, poca varianza
- Agregar las predicciones de cada árbol

Boosting

Boosting de árboles

- es la estructura del árbol: formada por
- es el valor de las hojas del árbol,

Boosting de árboles

Construir un conjunto de árboles:

- Los árboles son entrenados *secuencialmente*.
- Cada modelo sucesivo aprende de los errores del árbol anterior.
- **AdaBoost**: Adaptive Boosting (Freund & Schapire, 1997)
- Greedy function approximation: a **gradient boosting machine** (Friedman, 2001)

Gradient boosting

A partir de un conjunto de datos , una de las formas básicas de aplicar **gbm** es:

1. Iniciar , calcular
2. Para , se repite:
 - Obtener en base a con nodos terminales
 - Actualizar modelo:
 - Actualizar residuos:
3. El modelo resultante es:

Ejemplo de juguete: datos

los datos ...

```
1 d <- data.frame(  
2   altura = c(1.6, 1.6, 1.5, 1.8, 1.5, 1.4),  
3   color  = c('A', 'V', 'A', 'N', 'V', 'A'),  
4   sexo   = c('M', 'F', 'F', 'M', 'M', 'F'),  
5   peso   = c(88, 76, 56, 77, 73, 57)  
6 )  
7  
8 d
```

	altura	color	sexo	peso
1	1.6	A	M	88
2	1.6	V	F	76
3	1.5	A	F	56
4	1.8	N	M	77
5	1.5	V	M	73
6	1.4	A	F	57

Ejemplo de juguete: datos

En un primer paso, estimamos todos los valores con el promedio

```
1 mean(d$peso)
```

```
[1] 71.16667
```

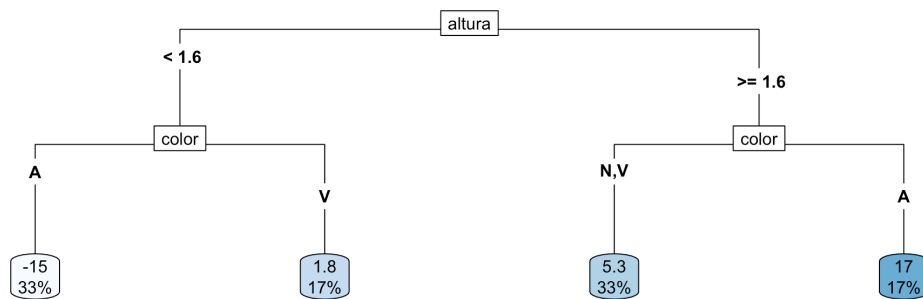
```
1 d$peso - mean(d$peso)
```

```
[1] 16.833333 4.833333 -15.166667 5.833333 1.833333 -14.166667
```

Ejemplo de juguete: Paso 1

```
1 library(rpart)
2 library(rpart.plot)
3 d$rr<- d$peso - mean(d$peso)
4
5 tt1 <- rpart( rr ~ altura+color+sexo, data=d,
6               control = rpart.control(minsplit = 1, minbuck
```

```
1 rpart.plot(tt1, type = 5)
```

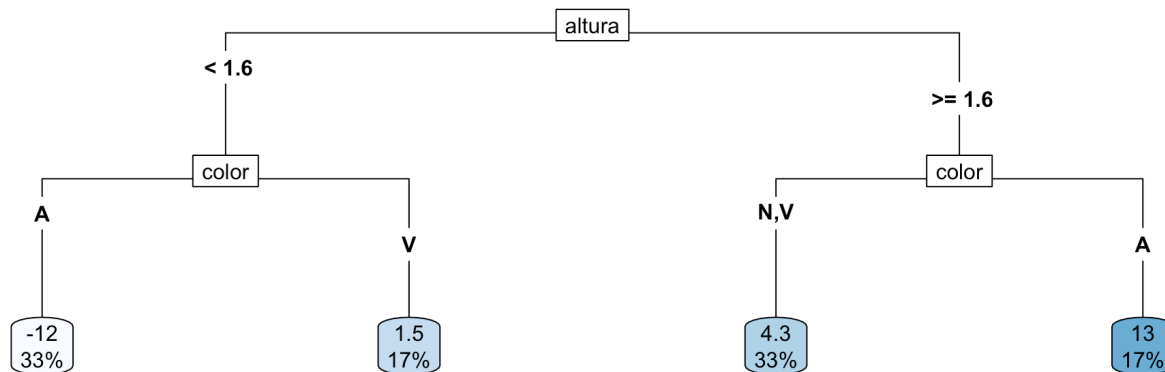


¿Cómo cambia la predicción del primer dato?

Ejemplo de juguete: Paso 2

```
1 d$rr <- d$rr - 0.2*predict(tt1)
2
3 tt2 <- rpart( rr ~ altura+color+sexo, data=d,
4               control = rpart.control(minsplit = 1, minbuck

1 rpart.plot(tt2, type = 5)
```



¿Cómo cambia la predicción del primer dato?

Ejemplo de juguete: Predicción

¿Como se predice un valor nuevo con nuestro Boosting de 2 árboles?

```
1 nd <- data.frame(altura=1.7, color='A', sexo='F')
```

El predictor resultante con 2 árboles y es:

```
1 mean(d$peso) + 0.2*predict(tt1, nd) + 0.2*predict(tt2, nd)
```

1

77.22667

Referencias

- daSilva, Natalia, Dianne Cook, and Eun-Kyung Lee. 2021. "A Projection Pursuit Forest Algorithm for Supervised Classification." *Journal of Computational and Graphical Statistics* 30 (4): 1168–80.
- Lee, Yoon Dong, Dianne Cook, Ji-won Park, and Eun-Kyung Lee. 2013. "PPtree: Projection Pursuit Classification Tree." *Electronic Journal of Statistics* 7: 1369–86.
- Loh, Wei-Yin. 2014. "Fifty Years of Classification and Regression Trees." *International Statistical Review* 82 (3): 329–48.
- Meinshausen, Nicolai. 2006. "Quantile Regression Forests." *Journal of Machine Learning Research* 7 (Jun): 983–99.