

# Árboles de clasificación basados en proyecciones y posibles extensiones

Natalia da Silva

Instituto de Estadística-FCEA-UDELAR

natalia.dasilva@fcea.edu.uy - natydasilva.com - @pacocuak

Noviembre-2022



FACULTAD DE  
CIENCIAS ECONÓMICAS  
Y DE ADMINISTRACIÓN



INSTITUTO  
DE ESTADÍSTICA



UNIVERSIDAD  
DE LA REPÚBLICA  
URUGUAY

## Estructura de la charla

- Motivación
- Projection Pursuit
- Descripción de PPtree
- Aspectos menos deseables de PPtree
- Posibles extensiones de PPtree
- Comentarios Finales

# Motivación

- Trabajo previo, PPforest, bosques aleatorios basados en proyecciones. [da Silva et al., 2021], [da Silva et al., 2022a], [da Silva et al., 2022b]
- Clasificador individual de PPforest es PPtree.
- Algunas debilidades de PPtree que se pueden mejorar.
- Mejorar la performance del bosques mejorando los árboles individuales.

## Motivación, Bosques aleatorios

Bosques aleatorios [Breiman, 2001] es un métodos de supervisado de agregación ampliamente utilizado y competitivo respecto a otros. Consiste en combinar árboles individuales incorporando dos conceptos claves:

- Agregación Bootstrap [Breiman, 1996]
- Selección aleatoria de variables [Amit and Geman, 1997], [Ho, 1998] en los árboles individuales.

# Motivación, Bosques aleatorios

Dos propuestas de agregación basadas en árboles:

- Basada en árboles ortogonales
- Basada en árboles oblicuos

## Motivación, PPforest

- PPforest, bosque aleatorio basado en proyecciones para clasificación.
- Clasificador individual PPtree[Lee et al., 2013], usa combinaciones lineales de variables en la partición del nodo, separa las clases teniendo en cuenta la correlación entre las variables.
- Hay algunos aspectos no deseables del clasificador individual que se pueden modificar para mejorar la performance del bosque.

# Projection Pursuit

- PP es una técnica estadística para exploración para encontrar proyecciones de datos que revelen estructuras interesantes [Friedman and Tukey, 1974], [Friedman and Stuetzle, 1981].
- Los algoritmos de PP buscan proyecciones de bajas dimensiones optimizando un índice de proyección que mide si la proyección es útil en algún sentido.
- Algunos índices de PP incorporan información de las clases para el cálculo.

## Indices PP para clasificación

- [Lee et al., 2005] proponen un índice derivado del análisis discriminante lineal útil para exploración en clasificación supervisada.
- [Lee and Cook, 2010] proponen un índice que funciona bien en el contexto de  $n \ll p$  y las variables altamente correlacionados.



## Índice PP, $\mathbb{I}_{LDA}$

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

donde  $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_q]$  es una matriz  $p \times q$  y define una proyección ortonormal de  $p$ -dimensiones a un subespacio  $q$ -dimensional,

$\mathbf{B} = \sum_{g=1}^G n_g (\bar{\mathbf{x}}_g - \bar{\mathbf{x}})(\bar{\mathbf{x}}_g - \bar{\mathbf{x}})^T$  es la suma de cuadrados entre grupos  $p \times p$ .

$\mathbf{W} = \sum_{g=1}^G \sum_{i \in H_g} (\mathbf{x}_i - \bar{\mathbf{x}}_g)(\mathbf{x}_i - \bar{\mathbf{x}}_g)^T$  es la suma de cuadrados al interior de los grupos  $p \times p$  donde  $H_g = \{i | y_i = g, i = 1, \dots, n\}$ ,  $\bar{\mathbf{x}}_g$  es el vector de medias de los grupos y  $\bar{\mathbf{x}}$  vector de medias global. Si el índice LDA tiene valores altos hay una gran diferencia entre clases y no de lo contrario.

## Indice PP, $\mathbb{I}_{PDA}$

Cuando las variables están altamente correlacionadas  $|\mathbf{A}^T(\mathbf{W} + \mathbf{B})\mathbf{A}|$  en  $\mathbb{I}_{LDA}$  es cercano a cero y no funciona bien.

$$\mathbb{I}_{PDA}(\mathbf{A}, \lambda) = 1 - \frac{|\mathbf{A}^T \mathbf{W}_{PDA}(\lambda) \mathbf{A}|}{|\mathbf{A}^T (\mathbf{W}_{PDA}(\lambda) + \mathbf{B}) \mathbf{A}|} \quad (2)$$

misma notación que en  $\mathbb{I}_{LDA}$ , agregando que  $\lambda \in [0, 1)$  es un parámetro de contracción y usamos una suma de cuadrados al interior de grupos diferente,  $\mathbf{W}_{PDA}(\lambda) = \text{diag}(\mathbf{W}) + (1 - \lambda)\text{offdiag}(\mathbf{W})$ .

## Optimización e $\mathbb{I}_{LDA}$ y $\mathbb{I}_{PDA}$

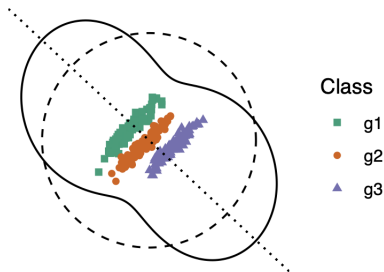
Para los dos índices de proyección seleccionados se puede encontrar el óptimo teórico.

- La proyección q-dimensional que maximiza  $\mathbb{I}_{LDA}$ , son los primeros q vectores propios de  $(\mathbf{W} + \mathbf{B})^{-1}\mathbf{B}$
- La proyección q-dimensional que maximiza  $\mathbb{I}_{PDA}$ , son los primeros q vectores propios de  $(\mathbf{W}_{PDA} + \mathbf{B})^{-1}\mathbf{B}$

## Huber plot, datos simulados

Huber plot [Huber, 1990]

Max: 0.97

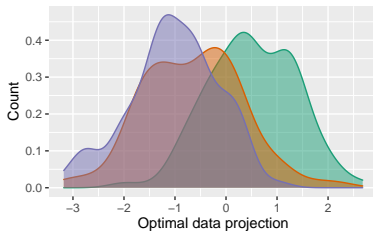
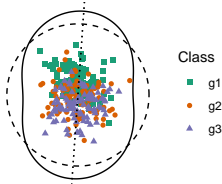


- Muestra el índice PP en todas las posibles direcciones en 2D
- Los índices se calculan usando las proyecciones para  $(\cos\theta, \sin\theta)$
- $\theta = 1^\circ, \dots, 180^\circ$
- Para cada proyección, cálculo el índice en los datos proyectados (línea sólida)
- Círculo punteado es de referencia, mediana de todos los valores del índice.
- Línea punteada corresponde al máximo índice.

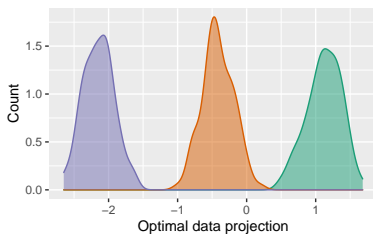
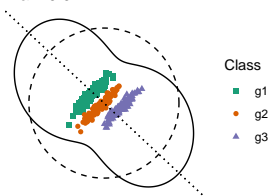
# $\mathbb{I}_{LDA}$ , proyección 1-D

## Huber Plot

Max: 0.35



Max: 0.97



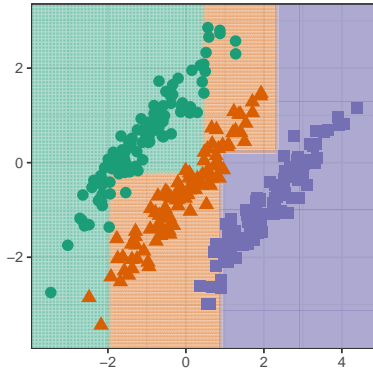
## PPtree: Projection pursuit classification tree

Método supervisado de clasificación

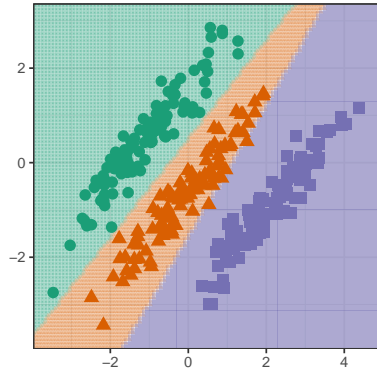
- **PPtree**: Árbol de clasificación basados en proyecciones [Lee et al., 2005] .
- Combina métodos de árboles con reducción de dimensionalidad basada en proyecciones, optimizando un índice de PP.
- Las particiones en **PPtree** se basan en combinaciones lineales de variables por lo que toma en cuenta la correlación entre las variables para separar las clases.

## CART vs PPtree, datos simulados

Rpart error 7.3 %



PPtree error 0.7 %



# PPtree

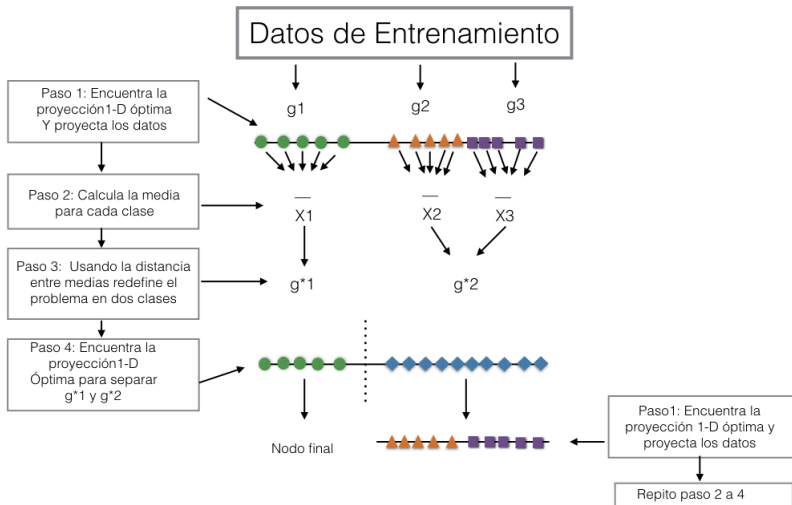
- Aunque el problema sea de clases múltiples, el método lo transforma en un problema de dos clases.
- Cuando las clases son más de dos el algoritmo usa dos pasos de proyección optimizando un índice de proyección en cada partición del nodo.
- Los coeficientes de proyección en cada nodo representan la 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  $1 - D$  óptima,  $\mathbf{a}_1^*$ , para separar todas las clases obteniendo los datos proyectados  $z_i = \mathbf{a}_1^{*T} \mathbf{x}_i$  para todo  $i$
- 2 En los datos proyectados, re-define el problema en uno de dos clases comparando distancias entre medias y asigna una nueva etiqueta,  $g_1^*$  o  $g_2^*$  para cada observación, generando una nueva variable  $y_i^*$  para la clase ( $g_1^*$  y  $g_2^*$ ).
- 3 Encuentra una proyección  $1 - D$  óptima  $\mathbf{a}_1^{**}$ , usando  $\{(\mathbf{x}_i, y_i^*)\}_{i=1}^n$  para separar  $g_1^*$  y  $g_2^*$ . La mejor separación y la regla de decisión para el nodo, si  $\mathbf{a}_1^{**T} \bar{\mathbf{x}}_{g_1^*} < c$  entonces asigna  $g_1^*$  al nodo izquierdo y en otro caso  $g_2^*$  al derecho, donde  $\bar{\mathbf{x}}_{g_1^*}$  es la media de  $g_1^*$ .
- 4 Para cada grupo, todos los pasos previos son repetidos hasta que  $g_1^*$  y  $g_2^*$  tienen una sola clase de la clase original.

## Ilustración del algoritmo



## PPtree, 8 reglas de corte

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

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

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

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

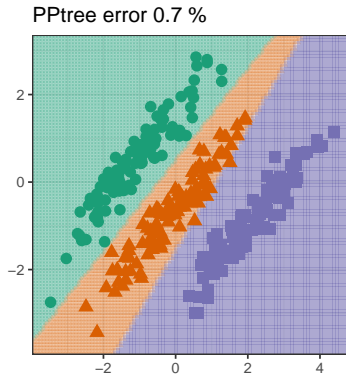
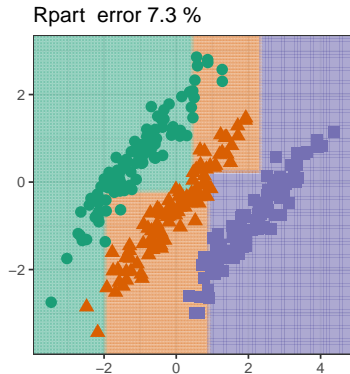
$$c = \frac{1}{2}\mathbf{x}_1^{med} + \frac{1}{2}\mathbf{x}_2^{med}$$

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

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

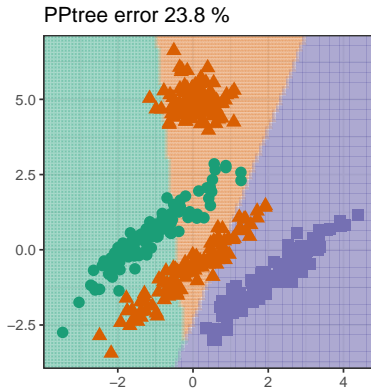
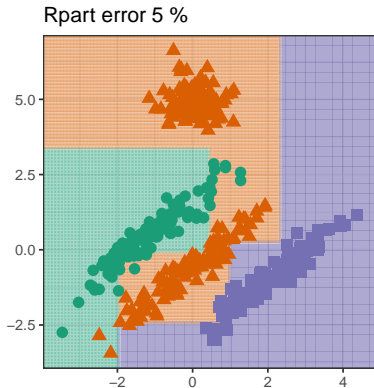
$$c = \frac{IQR_2/\sqrt{n_2}}{IQR_1/\sqrt{n_1}+IQR_2/\sqrt{n_2}}\mathbf{x}_1^{med} + \frac{IQR_1/\sqrt{n_1}}{IQR_1/\sqrt{n_1}+IQR_2/\sqrt{n_2}}\mathbf{x}_2^{med}$$

## Aspectos menos deseables: I



PPtree define una banda entre el naranja y el violeta que es muy cercana al grupo ya que la primer partición usa información del naranja y verde para calcular la media que define el punto de corte.

## Aspectos menos deseables: II



Los naranjas no pueden separarse por una única partición lineal y PPtree no puede modelar esto porque una clase es asignada solamente a un nodo terminal.

## Extensión: PPtree

Hay dos formas que el algoritmo ha sido modificado:

- Regla de decisión.
- Permitiendo particiones múltiples por grupos.

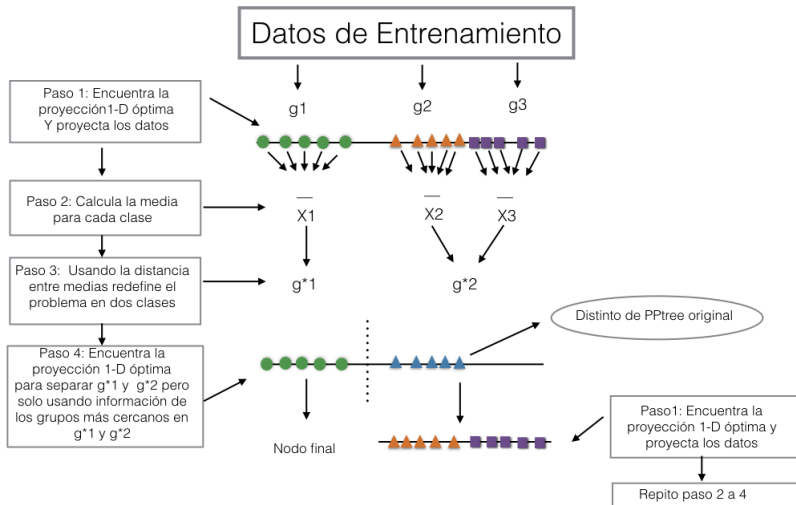
Las modificaciones implican:

- Subdividir super-clase para producir valor de corte más apropiado.
- Para incrementar el número de particiones por grupos modifíco la selección de la partición y debemos definir reglas de parada adicionales.

## Extensión I: subdividiendo clases para producir mejores bandas

- La primer modificación se enfoca en el cuarto paso del algoritmo original.
- En vez de combinar clases en una super clase, unicamente las dos clases más cercanas son usadas para determinar la partición.

## PPtree extensión I





## Extensión II: permitiendo particiones múltiples por grupos

- La modificación introduce una nueva forma de seleccionar el valor de la partición basado en la impureza del grupo resultante.
- $E(s) = -\sum_{j=1}^G p_{js} \log(p_{js})$  donde  $p_{js}$  es la proporción de puntos de la clase  $j$  en el subconjunto  $s$  y  $G$  el número de clases.
- $E(s)$  grande indican grupos mezclados, impuros.
- $E(s) = 0$  grupo puro.

## PPtree extension II

- Para determinar la calidad de la partición se debe considerar el lado izquierdo y derecho de la partición.
- $E(s_L, s_R) = \frac{n_L}{n_L + n_R} E(s_L) + \frac{n_R}{n_L + n_R} E(s_R)$
- $E(s_L, s_R)$  se calcula para cada posible partición y la que tiene la mínima impureza determina el corte.
- Esta modificación cambia dramáticamente el algoritmo de PPtree con el objetivo de flexibilizarlo a clasificador no lineal permitiendo muchas particiones por clase.
- Necesitamos determinar reglas de parada.

## PPtree extension II

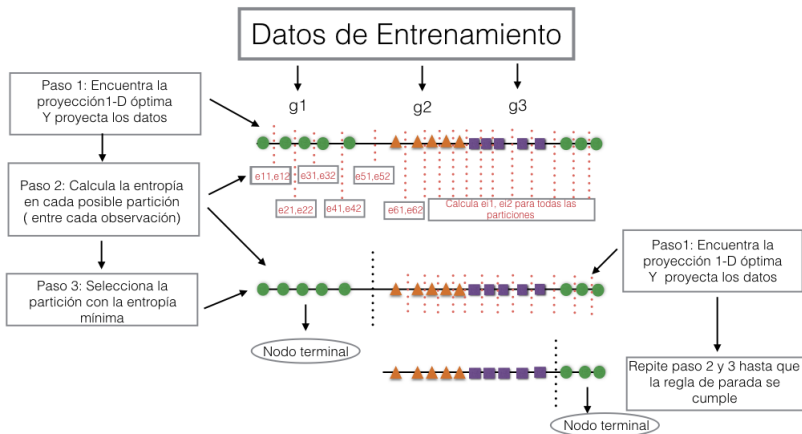
- 1 Optimiza un índice de proyección para encontrar la proyección  $1 - D$  óptima  $\mathbf{a}_1^*$  para separar todas las clases en los datos.
- 2 Con los datos proyectados calcula la entropía para cada posible partición. Las posibles particiones son definidas entre cada valor proyectado.
- 3 Selecciona la mejor partición que minimiza  $E(s_L, s_R)$ .
- 4 Repite los pasos anteriores hasta que la regla de parada se satisface.

## PPtree extensión II regla de parada

La regla de parada controla cuando el crecimiento del árbol debe parar. Las siguientes reglas de parada son usadas:

- Si un nodo es puro; todos los casos son de la misma clase en un nodo.
- Si el tamaño del nodo es menor a un valor determinado  $n_S$ .
- Si la reducción de la entropía de una partición es menos que un valor especificado  $ent_S$ .

## Ilustración de la extensión II



## Comparación de algoritmos

- Comparamos los métodos usando shiny [Chang et al., 2015] que nos permite incluir datos simulados y definir las bandas de cada algoritmo de forma interactiva.
- Se incluyen distintos métodos para simular datos 2D y se muestran las bandas definidas por los distintos algoritmos.

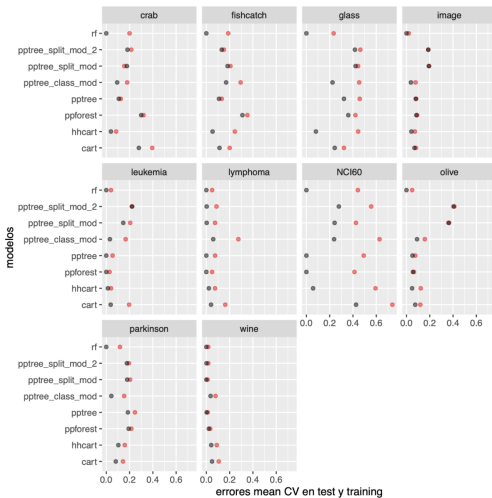
## Comparación de algoritmos

- Hay tres pestañas que controlan los distintos tipos de datos simulados.
- 1 mixtura multivariada con igual matriz de var-cov para los grupos y distinta media.
- Incluye uno de los grupo con outliers.
- Simulación de mixturas con el paquete 'MixSim'.

- Panel 1: Simulación básica de 3 clases  
<https://player.vimeo.com/video/222613204>
- Panel 2: Simulación con Outliers  
<https://player.vimeo.com/video/222613230>
- Panel 3: Simulaciones con MixSim pkg  
<https://player.vimeo.com/video/222613251>



## Con datos reales, resultados preliminares



## Paquete PPtreeExt

- Paquete en R PPtreeExt en etapa de desarrollo
- Disponible en <https://github.com/natydasilva/PPtreeExt>






## Comentarios Finales

- Se presentaron dos posibles modificaciones en el algoritmo de PPtree para hacerlo más flexible.
- shiny nos permite construir una herramienta para explorar resultados primarios en la modificación del algoritmo con datos simulados
- Los resultados primarios muestran un algoritmo de clasificación más flexibles utilizando combinaciones de variables
- Implementado en un paquete en R en desarrollo.

## Trabajo futuro

- Comparar la performance de PPtree con sus posibles extensiones con datos simulados y reales
- Con los nuevos árboles extender PPforest
- ....

## References I

-  Amit, Y. and Geman, D. (1997).  
Shape quantization and recognition with randomized trees.  
*Neural computation*, 9(7):1545–1588.
-  Breiman, L. (1996).  
Bagging predictors.  
*Machine Learning*, 24(2):123–140.
-  Breiman, L. (2001).  
Random forests.  
*Machine learning*, 45(1):5–32.
-  Chang, W., Cheng, J., Allaire, J., Xie, Y., and McPherson, J. (2015).  
shiny: Web application framework for R, R package version 0.11.
-  da Silva, N., Cook, D., and Lee, E.-K. (2021).  
A projection pursuit forest algorithm for supervised classification.  
*Journal of Computational and Graphical Statistics*, 30(4):1168–1180.

## References II



da Silva, N., Cook, D., and Lee, E.-K. (2022a).  
Interactive graphics for visually diagnosing forest classifiers in *Computational Statistics*, *Aceptado*.



da Silva, N., Cook, D., and Lee, E.-K. (2022b).  
*PPforest: Projection Pursuit Classification Forest*.  
R package version 0.1.3.



Friedman, J. H. and Stuetzle, W. (1981).  
Projection pursuit regression.  
*Journal of the American statistical Association*, 76(376):817–823.



Friedman, J. H. and Tukey, J. W. (1974).  
A projection pursuit algorithm for exploratory data analysis.  
*IEEE Transactions on computers*, 100(9):881–890.

## References III



Ho, T. K. (1998).

The random subspace method for constructing decision forests.  
*Pattern Analysis and Machine Intelligence, IEEE Transactions on*,  
20(8):832–844.



Huber, P. (1990).

Data analysis and projection pursuit.  
In *Technical report PJH-90-1*. Dept. of Mathematics, Massachusetts  
Institute of Technology Cambridge, MA.



Lee, E.-K. and Cook, D. (2010).

A projection pursuit index for large  $p$  small  $n$  data.  
*Statistics and Computing*, 20(3):381–392.



Lee, E.-K., Cook, D., Klinke, S., and Lumley, T. (2005).

Projection pursuit for exploratory supervised classification.  
*Journal of Computational and Graphical Statistics*, 14(4):831–846.

## References IV



Lee, Y. D., Cook, D., Park, J.-W., and Lee, E.-K. (2013).  
PPtree: Projection pursuit classification tree.  
*Electronic Journal of Statistics*, 7:1369–1386.