

Aprendizaje Estadístico Supervisado

Natalia da Silva

2024

Clase pasada

- Árboles de regresión y clasificación

Clase de hoy

- Métodos de agregación:
 - Bagging
 - Random Forest

Recap algunas ventajas de los árboles

Algunas ventajas de los árboles:

- Sencillos de explicar, tal vez más que una regresión lineal.
- Algunos piensan que los árboles están más cerca de la forma en que las personas toman decisiones.
- Pueden ser representados gráficamente y de forma sencilla se puede interpretar lo que hace el método.
- Los árboles pueden usar predictores cualitativos sin tener que crear variables de tipo dummy.

Algunas desventajas de los árboles

Algunas desventajas de los árboles:

- En general no tienen tan buena performance predictiva como otros métodos
- Son poco robustos, pequeños cambios en los datos pueden causar un gran cambio en las estimaciones finales.

Surgen métodos que intentan mejorar estas desventajas. Agregando muchos árboles de decisión logran mejorar sustancialmente la performance predictiva.

Métodos de Agregación

Métodos de agregación

- Métodos de agregación: combina múltiples modelos individuales para construir un modelo de predicción combinando los anteriores.
- En general los modelos individuales que se combinan son llamados **weak learners**, ya que en general tienen predicciones mediocres individualmente.
- Algunos ejemplos bien conocidos son; bagging, boosting, random forest entre otros.
- La diferencia principal entre los métodos de agregación son; el tipo de modelos individuales a ser combinados, las tareas que se asignan a cada modelo y la forma en que estos modelos son combinados.

A este tipo de modelos se les llama *ensembles*.

Los árboles de regresión y clasificación son muy utilizados para esto.

Métodos de agregación

Bagging

- Construir árboles entrenados independientemente
- Árbol individual grande
- Promediar predicciones de cada árbol

Bagging

Bagging: Bootstrap Aggregation

Queremos ajustar un modelo de árbol de regresión (o clasificación) en un conjunto de datos $\mathbf{D} = (\mathbf{Y}, \mathbf{X})$. Llamemos $g_{\mathbf{D}}(\mathbf{x})$ al árbol construido con todos los datos.

Bagging

1. Sortear réplicas bootstraps de los datos \mathbf{D}_b^* , con $b = 1, \dots, B$
2. En cada réplica, ajustar un árbol $g_b(\mathbf{x})$, terminamos con B árboles
3. Agregar las predicciones de los B árboles construidos:

Regresión

$$g(\mathbf{x}) = \frac{1}{B} \sum_b g_b(\mathbf{x})$$

Clasificación $g(\mathbf{x}) = \operatorname{argmax}_{(k \in 1, \dots, K)} \left\{ \sum_b I(g_b(\mathbf{x}) = k) \right\}$

Bagging

Mencionamos que:

- La idea de bagging es promediar modelos ruidosos, aproximadamente insesgados, para reducir la varianza.
- Los árboles de tipo CART son *inestables*, cambios en el conjunto de entrenamiento puede resultar en cambios grandes del modelo final.

Los árboles son **buenos** candidatos para hacer *bagging*.

- capturan interacciones complejas
- tienen poco sesgo (si crecen lo suficiente)

Cada árbol es generado id, la esperanza del promedio es igual a la esperanza de un árbol, entonces el sesgo es el mismo pero hay una posible reducción en la varianza.

Idea general, bosque

(Breiman 2001)

- Método supervisado de agregación (Breiman 2001) ampliamente utilizado (más de 140 mil citas).
- Pueden ser usados tanto para problemas de clasificación como de regresión en una amplia variedad de problemas.
- En regresión, Random Forest (RF) suaviza la estimación promediando en un conjunto de árboles.

Se basa en promediar un conjunto de árboles aleatorizados.

Idea general, bosque

Recordemos el contexto del problema, $Y = f(X) + \epsilon$, el estimador RF de f para el caso de respuesta numérica, se puede expresar como:

$$\hat{f}_{\text{rf}}(\mathbf{x}) = \frac{1}{B} \sum_b T(\mathbf{x}, \Theta_b)$$

donde $T(\mathbf{x}, \theta_b)$ es un árbol *aleatorizado*.

Θ_b representa **dos** fuentes de aleatoriedad:

- Se entrena con una muestra bootstrap (igual que *bagging*)
- Las variables para hacer particiones se seleccionan aleatoriamente

La idea es tener como base la reducción de la varianza lograda por bagging y obtener mejora a través de incorporar una fuente adicional de aleatoriedad.

Idea general: bosque

Consideremos la varianza del *promedio de B variables aleatorias*

- Las variables i.i.d cada una con varianza σ^2 , entonces varianza del promedio es $\frac{1}{B}\sigma^2$.
- Las variables son i.d (no independientes) con correlación ρ positiva, la varianza del promedio es

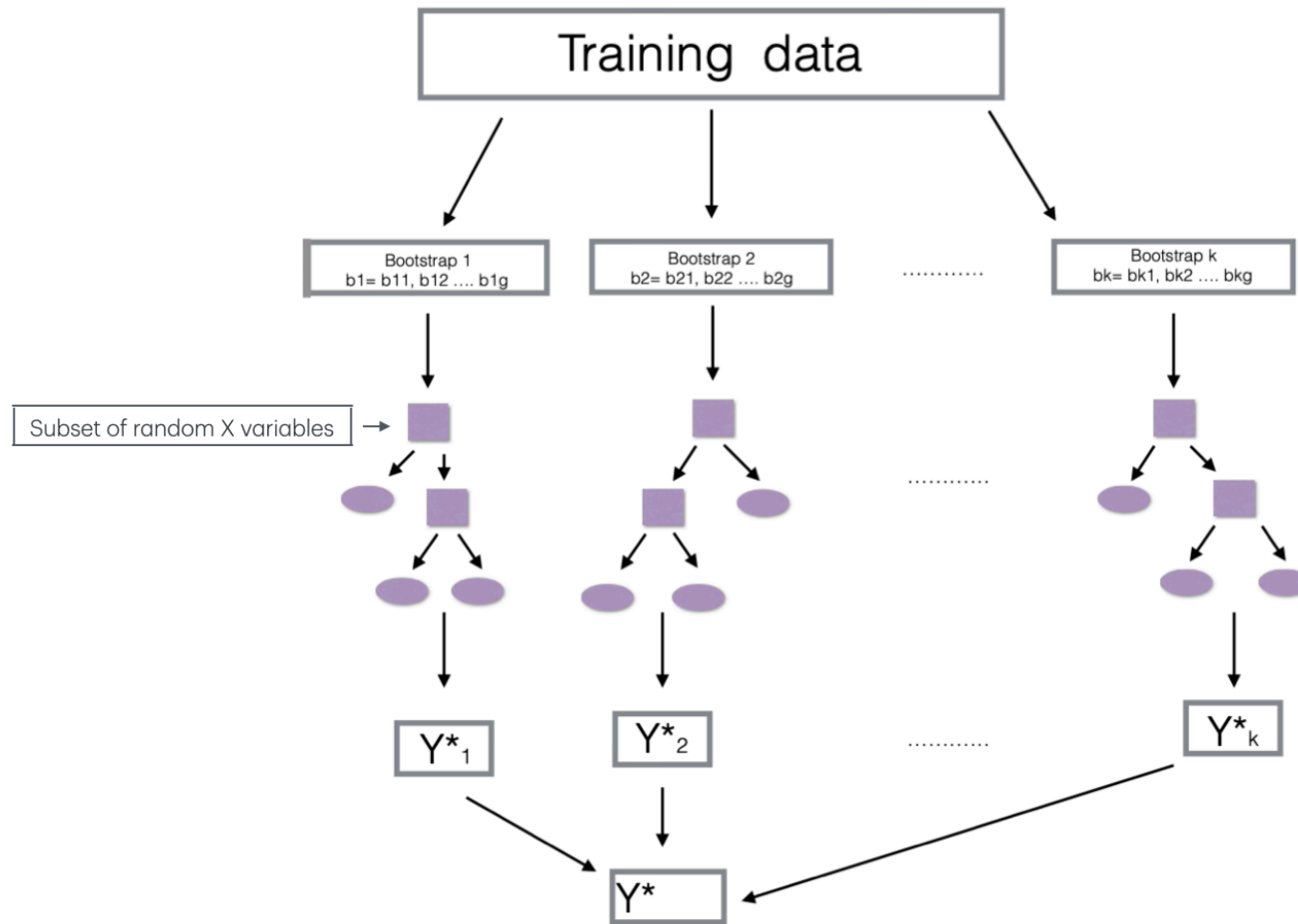
$$\rho\sigma^2 + \frac{(1 - \rho)\sigma^2}{B}$$

Esto se vincula con las fuentes de aleatoriedad en RF:

- muestras bootstrap: aumentar B reduce el segundo componente
- selección de predictores: busca reducir ρ

Bosque aleatorio: algoritmo

Diagrama bosque



Algoritmo Bosque

Obtengo un conjunto de árboles $\{T_b\}_1^B$. Para $b \in 1, 2, \dots, B$

1. Selecciono muestra bootstrap Z^b de tamaño N
2. Con los datos Z^b , estima árbol T_b , tipo CART, con dos modificaciones:
 - en cada nodo, solo se consideran m variables seleccionadas al azar
 - el árbol crece hasta el final sin podar

Luego:

Regresión
$$\hat{f}_{rf} = \frac{1}{B} \sum_{b=1}^B T(x, \theta_b)$$

Clasificación
$$\hat{f}_{rf} = \underset{g}{\operatorname{argmax}} \frac{1}{B} \sum_{b=1}^B I(T(x, \theta_b) = g)$$

Parámetros auxiliares

Parámetros básicos: B , m y tamaño de las hojas.

Reducir m hace que se reduzca la correlación entre pares de árboles y reducir la varianza del promedio. Pero al mismo tiempo hace que cada árbol individual sea peor.

Valores por defecto:

- Clasificación, defecto \sqrt{p} y los nodos tienen un tamaño mínimo de 1.
- Regresión, defecto para m es $p/3$ y el mínimo tamaño de nodo es 5.
- El valor de B depende de la implementación computacional.

Muestras OOB

- Cada muestra bootstrap tiene asociado un conjunto de datos *fuera de la muestra*, llamamos este conjunto OOB_b .
- Para cada observación $z_i = (x_i, y_i)$, se puede hacer una predicción utilizando **unicamente** los árboles T_b donde $z_i \in \text{OOB}_b$.

Definimos:

$$\text{errOOB} = \frac{1}{N} \sum_i (y_i^{\text{oob}} - y_i)^2$$

Sobreajuste

- Cuando el número de variables es grande pero pocas son relevantes, RF es probable que no funcione bien si se selecciona un m chico.
- En cada partición sería pocas las chances de tener alguna variable relevante.
- RF se promociona como un método que no sobreajusta, incrementar B no causa un sobreajuste

Importancia y selección de variables

Importancia de las variables Medida que indica que tan importante es la variable para el método predictivo.

En cada partición de los árboles, la mejora en el criterio de partición es la medida de importancia de esa variable y se acumula en todos los árboles en el bosque.

RF también usa las observaciones OOB para otra medida de importancia que permite medir la fortaleza de cada variable para predecir (permuted importance variable).

Medida de importancia permutada

En cada árbol $T_b(\mathbf{x}, \theta_b)$ se obtiene una medida del error individual con los datos OOB_b

$$VI_j = \frac{1}{B} \sum_b \text{erOOB}_b^* - \text{erOOB}_b$$

erOOB_b^* es el error en $T_b(\mathbf{x}, \theta_b)$ luego de permutar aleatoriamente los valores del predictor X_j .

- La importancia de X_j es mayor cuanto mayor sea el incremento en el error debido a permutar sus valores

Matriz de proximidad

- Contiene la información de que tan cercanas/similares son dos observaciones en el bosque.
- Para cada árbol, todo par de observaciones OOB que comparten el mismo nodo terminal incrementan su proximidad en uno.
- La matriz de proximidades puede ser utilizada para imputar datos faltantes.

La idea es poder visualizar usando la matriz de proximidades las observaciones que son cercanas según el bosque.

Propiedades estadísticas

¿Qué podemos decir del bosque estimado $\hat{f}(x)$ para hacer inferencia?

Más allá de tasa de error y su convergencia, hay muchos trabajos dedicados a estudiar propiedades estadísticas del método.

- Teoría asintótica (convergencia, normalidad)
- Varianzas y/o errores standard
- Intervalos de predicción (cobertura)

(Biau and Scornet 2016)

Convergencia y normalidad asintótica

- Difícil de trabajar con RF original.
- Modificación más común: hacer sub-muestras en vez de bootstrap.
- Wager, Hastie, and Efron (2014), Mentch and Hooker (2016)

Algunos resultados se basan en

$$\hat{f}_{\text{rf}}(\mathbf{x}) = \frac{1}{B} \sum_b T(\mathbf{x}, \Theta_b) = \sum_{i=1}^n w_i(\mathbf{x}) Y_i$$

RF vistos como promedios ponderados

Consideramos UN árbol aleatorizado

- $\ell = 1 \dots L$ las hojas del árbol (nodos terminales) y $R_\ell \subseteq S(X)$ la región que define ℓ .
- x es un nuevo dato, queremos predecir y
- La predicción de un solo árbol $T_b(x, \theta)$ para x es el promedio de las observaciones Y_i en la región $R_{\ell(x, \theta)}$.

Podemos expresar la predicción para x como

$$T_b(x, \theta) = \sum_{i=1}^n w_i(x, \theta) Y_i \quad w_i(x, \theta) = \frac{1_{\{X_i \in R_{\ell(x, \theta)}\}}}{\#\{j: X_j \in R_{\ell(x, \theta)}\}}$$

RF vistos como promedios ponderados

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

$$\begin{aligned}\hat{f}_{\text{rf}}(\mathbf{x}) &= \frac{1}{B} \sum_b T(\mathbf{x}, \Theta_b) \\ &= \frac{1}{B} \sum_b \sum_{i=1}^n w_i(\mathbf{x}, \Theta_b) Y_i \\ &= \sum_{i=1}^n \left(\frac{1}{B} \sum_b w_i(\mathbf{x}, \Theta_b) \right) Y_i \\ &= \sum_{i=1}^n w_i(\mathbf{x}) Y_i\end{aligned}$$

donde $w_i(\mathbf{x}) = \frac{1}{B} \sum_{t=1}^B w_i(\mathbf{x}, \theta_t)$.

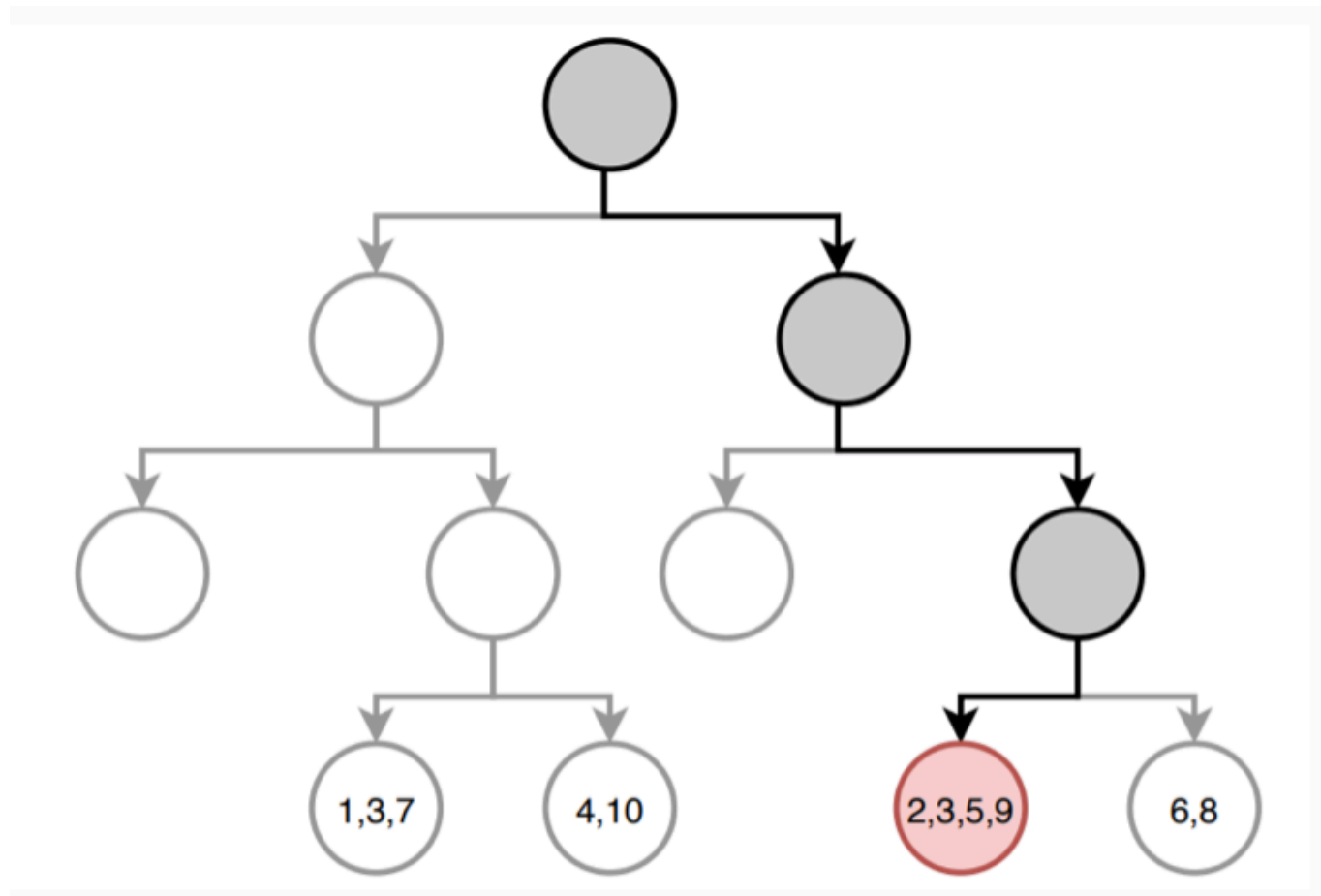
Notar que $w_i(\mathbf{x}, \theta)$ suman 1 en cada árbol y $w_i(\mathbf{x})$ suma 1 en el bosque.

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

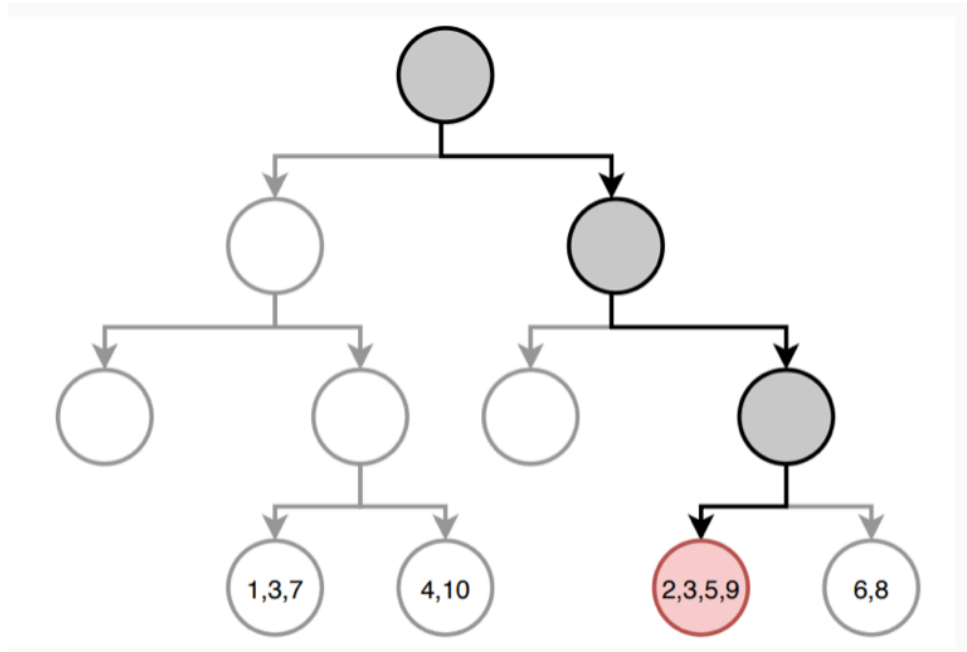
Un solo árbol

Predicción con un árbol de regresión para x : 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

Usando la notación de ponderadores



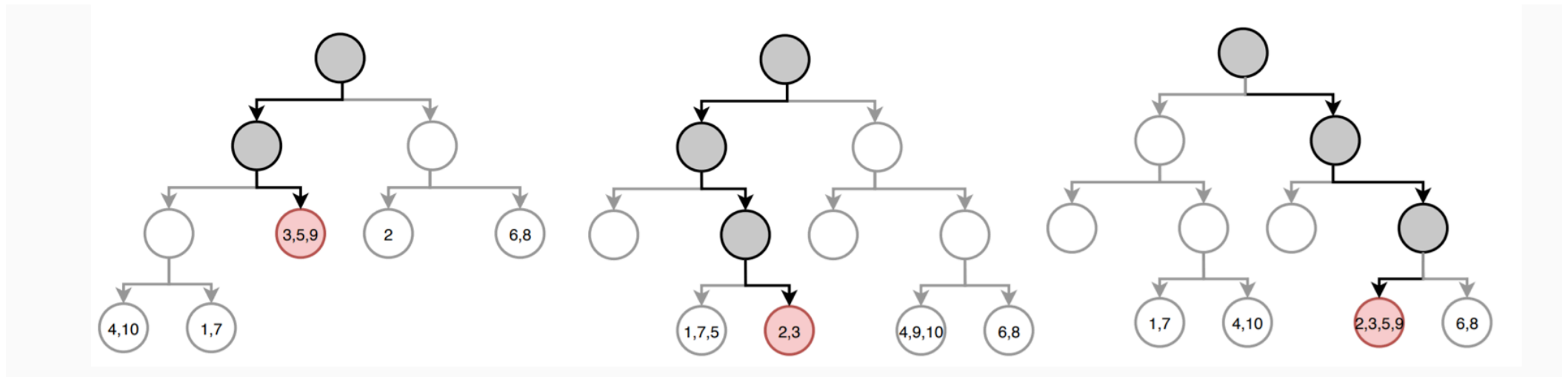
- id: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
- Y: 10,18,24,8,2,9,16,10,20,14
- El valor predicho por el árbol es la media de la variable respuesta Y de las observaciones con id: 2, 3, 5, 9.
- $\hat{f}_{\text{tr}}(\mathbf{x}) = \sum_{i=1}^{10} w_i(\mathbf{x}, \theta) Y_i$
- $\mathbf{w} = (0, \frac{1}{4}, \frac{1}{4}, 0, \frac{1}{4}, 0, 0, 0, \frac{1}{4}, 0)$
- $\hat{f}_{\text{tr}}(\mathbf{x}) = \mathbf{w} * \mathbf{Y}^T$

Predicción un árbol de regresión

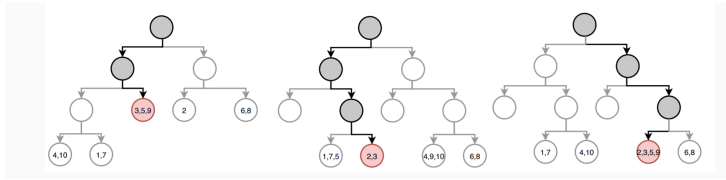
$$\hat{f}_{\text{tr}}(\mathbf{x}) = \mathbf{w} * \mathbf{Y}^T = \left(0 \quad \frac{1}{4} \quad \frac{1}{4} \quad 0 \quad \frac{1}{4} \quad 0 \quad 0 \quad 0 \quad \frac{1}{4} \quad 0\right) \begin{pmatrix} 10 \\ 18 \\ 24 \\ 8 \\ 2 \\ 9 \\ 16 \\ 10 \\ 20 \\ 14 \end{pmatrix}$$

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
- $W_{arbol1} = (0, 0, \frac{1}{3}, 0, \frac{1}{3}, 0, 0, 0, \frac{1}{3}, 0)$
- $W_{arbol2} = (0, \frac{1}{2}, \frac{1}{2}, 0, 0, 0, 0, 0, 0, 0)$
- $W_{arbol3} = (0, \frac{1}{4}, \frac{1}{4}, 0, \frac{1}{4}, 0, 0, 0, \frac{1}{4}, 0)$

- $\bar{W} = \frac{1}{3}(W_{arbol1} + W_{arbol2} + W_{arbol3})$

$$= (0, \frac{1}{4}, \frac{13}{36}, 0, \frac{7}{36}, 0, 0, 0, \frac{7}{36}, 0)$$

Bosque con tres árboles

$$\hat{f}_{\text{rf}}(\mathbf{x}) = \bar{\mathbf{w}} * \mathbf{Y}^T = \left(0 \quad \frac{1}{4} \quad \frac{13}{36} \quad 0 \quad \frac{7}{36} \quad 0 \quad 0 \quad 0 \quad \frac{7}{36} \quad 0\right) \begin{pmatrix} 10 \\ 18 \\ 24 \\ 8 \\ 2 \\ 9 \\ 16 \\ 10 \\ 20 \\ 14 \end{pmatrix}$$

Varianzas e intervalos de predicción

- Sexton, J., Lake, P. 2009, proponen usar Monte Carlo y variantes para evaluar incertidumbre en predicciones de RF
- Wager, S., et.al., 2014, usa las muestras OOB para estimar la varianza de $\hat{f}_{\text{rf}}(\mathbf{x})$
- Intervalos de predicción se pueden construir con los percentiles Meinshausen, N., 2006. Distribución empírica de los errores OOB (Zhang, H., et.al. 2019)

Estos son posibles papers para presentar

Variantes de Random Forest

- Random survival forests (Ishwaran, H., et.al. 2008)
- Multivariate random forests (Segal, M., Xiao, Y., 2011)
- Enriched random forests (Amaratunga, D., et.al. 2008)
- Quantile regression forests (Meinshausen, N., 2006)
- PPforest (da Silva, N. et.al. 2021)
- y más

Ejemplo: Abandono 1ero 2016-2017

Modelos de clasificación para el abandono

- Variable de respuesta categórica con dos niveles (abandona, no abandona).
- Variables explicativas: Contexto sociocultural, sexo, extra edad fuerte, extra edad leve, inasistencias relativas, centro educativo, departamento.
- Problema: los datos están muy desbalanceados sólo un 7% abandonan
- Distintas estrategias usando bosques aleatorios (RF)

Código para ajustar un Bosque aleatorio:

```
1 rfab_uw <- randomForest(  
2   Abandono ~ nro_doc_centro_educ+inas_rel+cl2+Sexo+EE_Fuer  
3   data = dat2  
4   )
```

Performance de clasificadores


Existen muchas medidas para evaluar clasificadores, las más básicas

- Matriz de confusión
- Sensibilidad y Especificidad
- Curva ROC
- Área debajo de la curva

Vemos las medidas en el ejemplo de abandono

Observado vs predicho, matriz de confusión

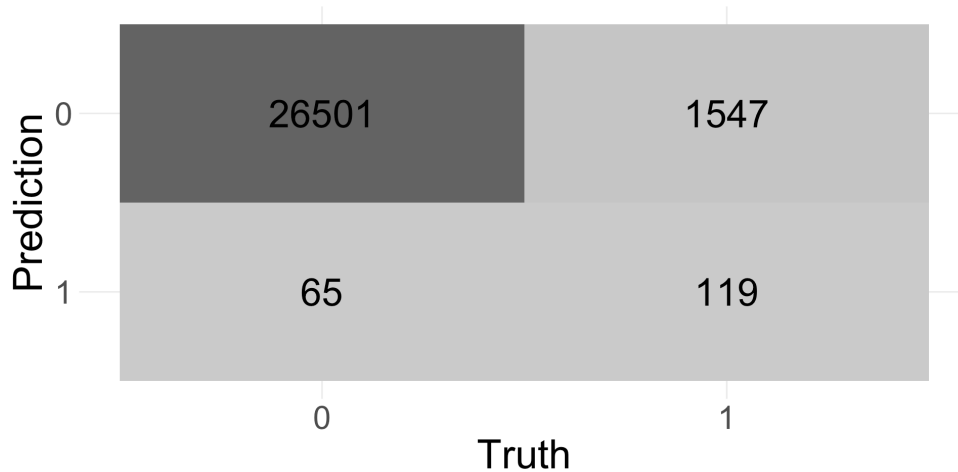
Matriz de confusión para dos clases (Yes, No)

		Observado	
		0	1
Predicho	0	 TN	 FN
	1	 FP	 TP

Las celdas de true positive (TP) y true negative (TN) contienen el número de casos correctamente predichos como **1** o **0**. Las celdas de false positive (FP) y false negative (FN) representan el número de casos que son predichos incorrectamente de **1** y **0**.

RF base - Matriz de confusión

```
1 # Predicciones del Bosque
2 rfPred.uw <- predict(rfab_uw, type='prob') |>
3   data.frame() |>
4   set_names(nm = paste0('prob', 0:1)) |>
5   mutate( trueAB = dat2$Abandono, predAB = predict(rfab_uw
6
7 # Matriz de confusión
8 cm <- conf_mat(data=rfPred.uw, truth = 3, estimate = 4)
```



- 65 obs son clasificadas erroneamente como 1 (FP)
- 1547 obs son clasificadas erroneamente como 0 (FN)

Sensibilidad y Especificidad

A partir de la Matriz de confusión se obtienen *medidas de ajuste*.

- **Sensibilidad** del modelo, es la proporción de verdaderos positivos (TP) que son correctamente identificados por el modelo (tasa de verdaderos positivos).
- **Especificidad** es la tasa de verdaderos negativos (TN), la proporción de negativos que el modelo predice correctamente.

$$\text{Sensibilidad} = \frac{TP}{TP+FN}$$

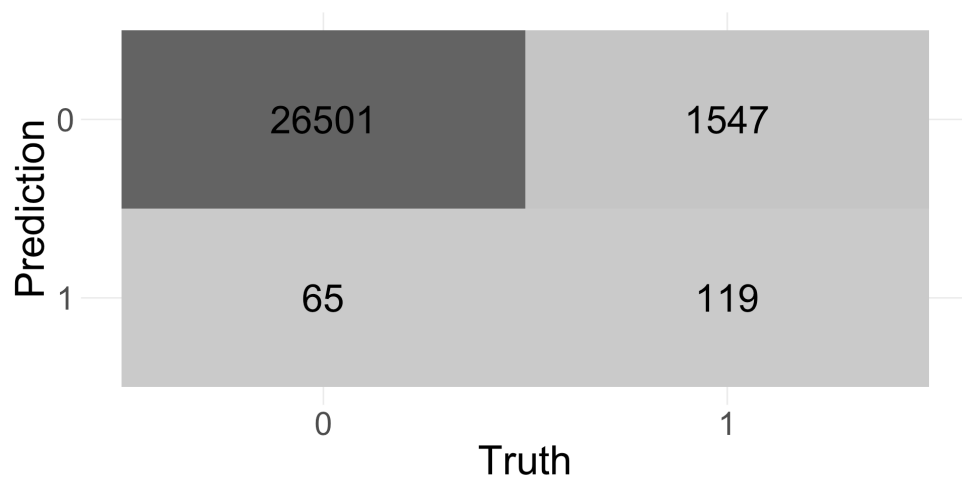
$$\text{Especificidad} = \frac{TN}{TN+FP}$$

- Sensibilidad y especificidad están acotados entre 0 y 1. Valores altos implican mejor performance.
- Si ajustamos el modelo para incrementar la Sensibilidad es probable que caiga también la especificidad.

RF: sensibilidad y especificidad

$$\text{Sensibilidad: } \frac{TP}{TP+FN}$$

$$\text{Especificidad: } \frac{TN}{TN+FP}$$



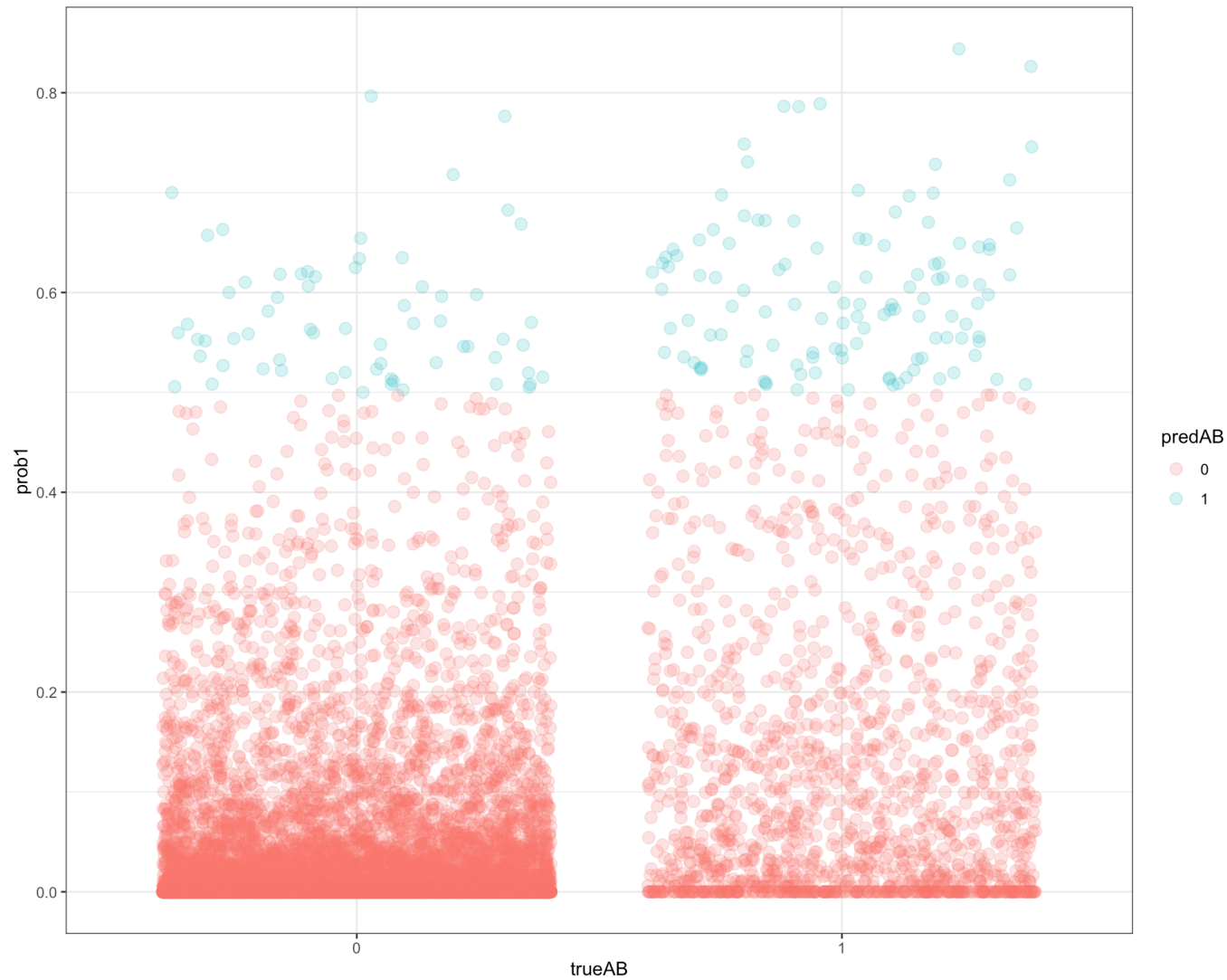
```
# A tibble: 3 × 2
  .metric .estimate
  <chr>    <dbl>
1 accuracy 0.943
2 sens     0.0714
3 spec     0.998
```

- 94% correctamente clasificados (precisión)
- Solo 7% de los 1s correctamente clasificados (sensibilidad)
- Casi todos los 0s son correctamente clasificados (especificidad)

Visualizando la performance del modelo

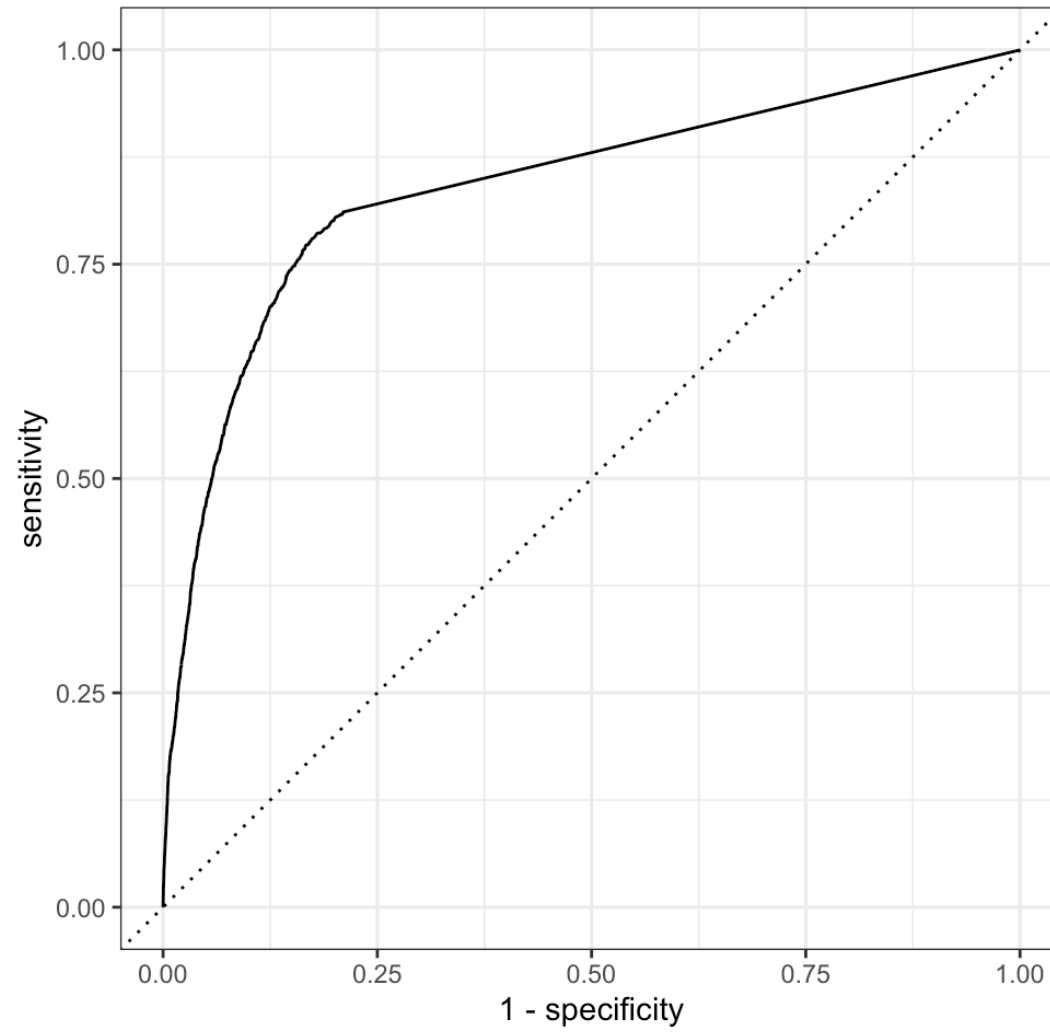
- Durante el proceso de clasificación los algoritmos estiman la probabilidad de que cada observación pertenezca a una clase particular. Conocida como propensiones y ellas se calculan teniendo en cuenta un valor de corte.
- En general para un problema de dos clases el punto de corte es 0.5.
- Es posible usar otros puntos de corte lo que afectará el cálculo de la sensibilidad y la especificidad.
- Entender como cambia la sensibilidad y la especificidad de un clasificador cuando cambia el punto de corte nos da un mejor entendimiento de la performance del modelo.
- Este balance se resume en la **Curva ROC**

Punto de corte



```
# A tibble: 3 × 2
  .metric .estimate
  <chr>    <dbl>
1 accuracy 0.943
2 sens      0.071
3 spec      0.998
```

RF: Curva ROC



Datos desbalanceados

- Los algoritmos de ML no tienen mucha información de la clase minoritaria para tener buenas predicciones en la clase minoritaria.
- Los algoritmos son guiados por la precisión, minimizan el error global donde la clase minoritaria tiene poco peso.
- Los algoritmos de ML asumen que las clases están balanceadas.
- También asumen que los errores obtenidos de diferentes clases tienen el mínimo costo.

Random Forest, datos desbalanceados

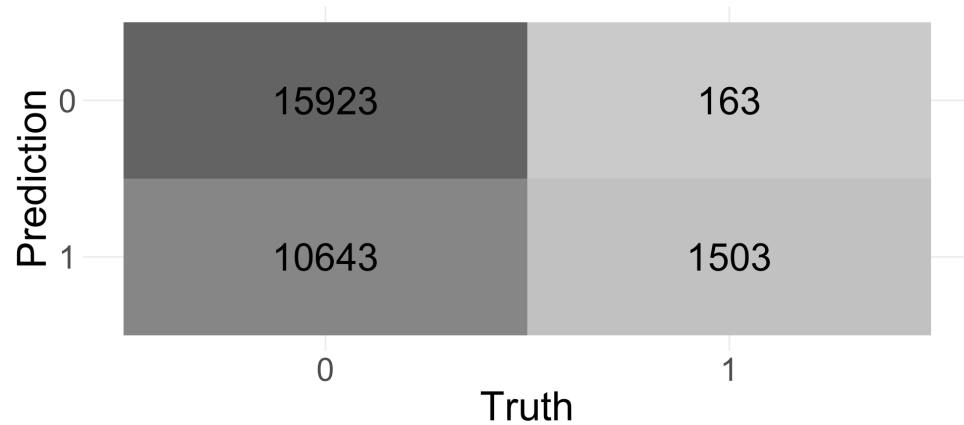
- Podemos incorporar pesos a las clases y penalizar el error de clasificación en la clase minoritaria (WRF)
- Combinar técnicas de muestreo y métodos de agregación. Sub muestrea la clase mayoritaria y crece los árboles con muestras más balanceadas (BRF).

Random Forest ponderado, WRF

- Aprendizaje sensible al costo
- Debemos incorporar mayor penalidad al error de clasificación en la clase minoritaria.
- En WRF los pesos se traducen en dos lugares: usa pesos para encontrar las particiones de cada árbol. En los nodos terminales de cada árbol se ponderan las clases y se usa voto mayoritario ponderado.

Random Forest ponderado, WRF

```
1 rfab_w15<-randomForest(Abandono~ nro_doc_centro_educ+inas_  
2                           cl2+ Sexo+ EE_Fuerte,  
3                           classwt = c(1, 5),  
4                           data = dat2)
```



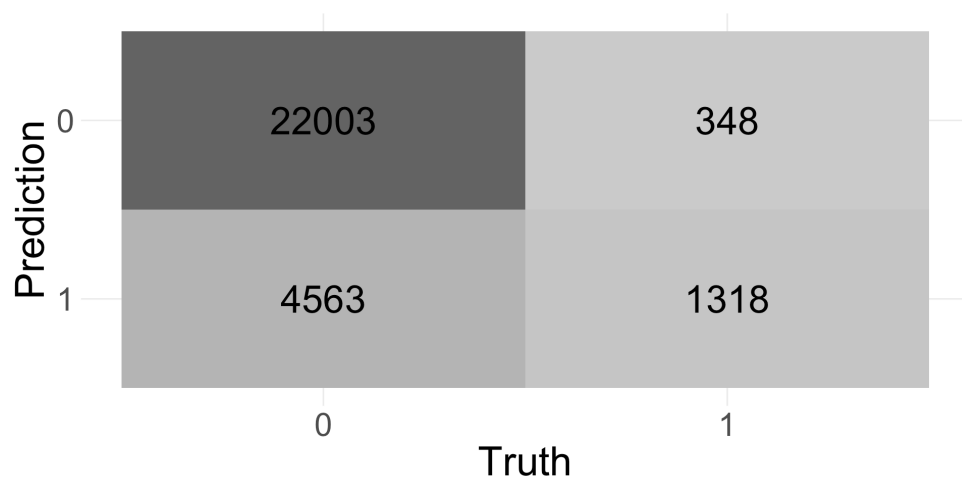
```
# A tibble: 3 × 2  
  .metric .estimate  
  <chr>    <dbl>  
1 accuracy 0.617  
2 sens     0.902  
3 spec     0.599
```

Random Forest, balanceado (BRF)

- Saco muestras bootstrap para la clase minoritaria y aleatoriamente con reposición, selecciono el mismo número de casos para la clase mayoritaria.
- Ajusto CART para cada muestra bootstrap y selecciono aleatoriamente un subconjunto de variables para la partición de cada nodo
- Basado en voto mayoritario obtengo la predicción del bosque

Random Forest, balanceado (BRF)

```
1 rfab_b <- randomForest(Abandono~nro_doc_centro_educ+inas_r  
2                       cl2 + Sexo + EE_Fuerte,  
3                       strata = dat2$Abandono,  
4                       sampsize =rep(1666,2), data = dat2)
```

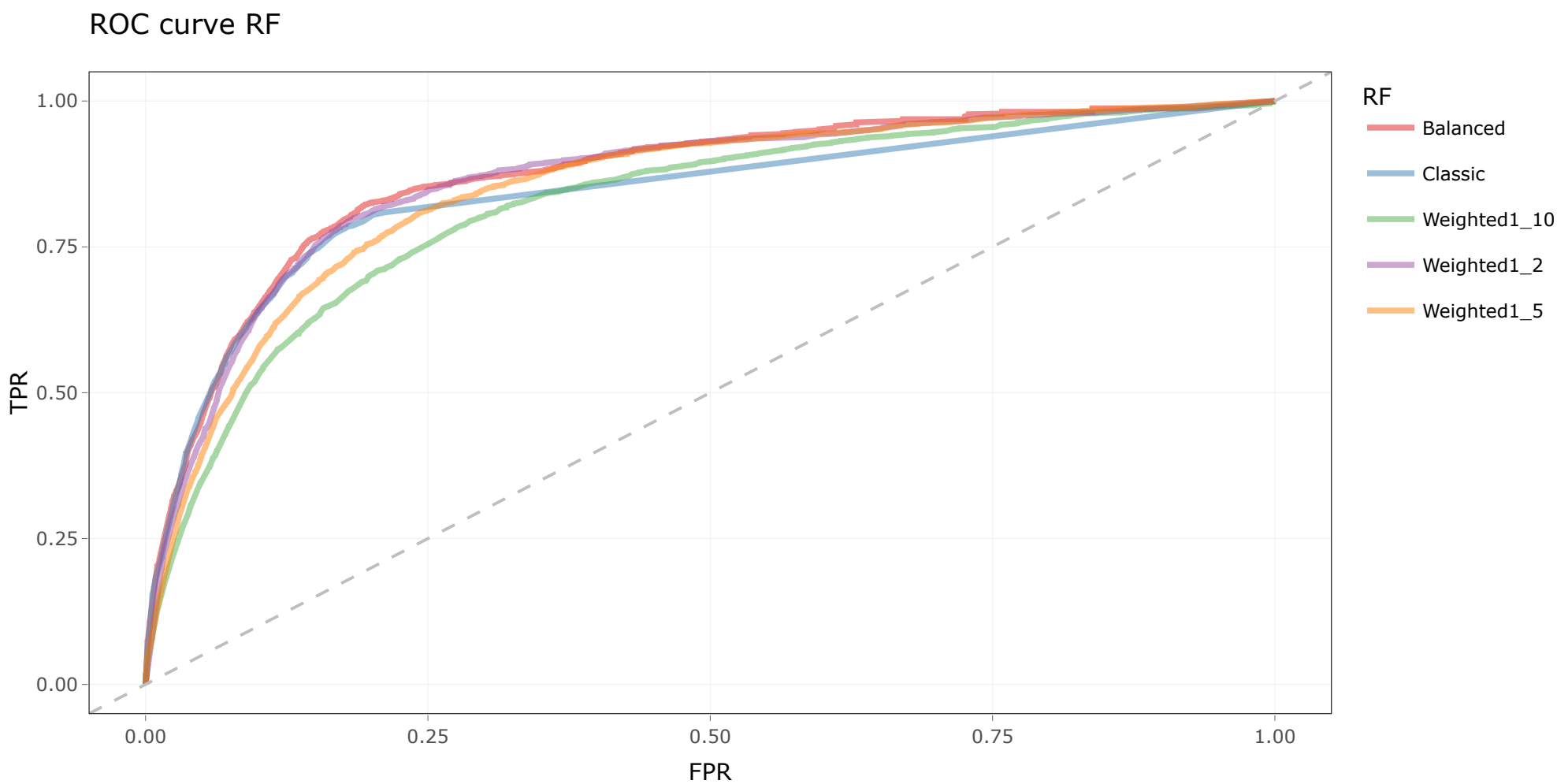


```
# A tibble: 3 × 2  
  .metric .estimate  
  <chr>    <dbl>  
1 accuracy 0.826  
2 sens      0.791  
3 spec      0.828
```

Random Forest, ROC

- Comparar la performance, uso ROC (Receiver operating characteristic curve)
- ROC es una representación gráfica que muestra el trade off entre los TPR y FPR para todos los puntos de corte

Random Forest, ROC



Comentarios Finales

Datos desbalanceados:

- Importante usar modelos que tomen en cuenta esta característica
- Seleccionar medidas de performance apropiadas para evaluar los modelos
- Seleccionar modelos en base a nuestro interés en el problema particular
- En nuestro caso reducir los Falsos negativos (que abandone y clasificarlo erróneamente)

Referencias

- Biau, Gérard, and Erwan Scornet. 2016. "A Random Forest Guided Tour." *Test* 25 (2): 197–227.
- Breiman, Leo. 2001. "Random Forests." *Machine Learning* 45 (1): 5–32.
- Mentch, Lucas, and Giles Hooker. 2016. "Quantifying Uncertainty in Random Forests via Confidence Intervals and Hypothesis Tests." *The Journal of Machine Learning Research* 17 (1): 841–81.
- Wager, Stefan, Trevor Hastie, and Bradley Efron. 2014. "Confidence Intervals for Random Forests: The Jackknife and the Infinitesimal Jackknife." *The Journal of Machine Learning Research* 15 (1): 1625–51.