UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

UNA INTRODUCCIÓN A ALGUNOS MÉTODOS DE APRENDIZAJE ESTADÍSTICO Y SU APLICACIÓN EN EL SISTEMA BANCARIO

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

ACTUARIO

P R E S E N T A:

BERNARDO WILLIAMS MORENO SÁNCHEZ

DIRECTOR DE TESIS:

M. EN C. FABIOLA CAROLINA CAÑAS MAYA

CIUDAD DE MÉXICO, 2020

Agradecimimientos

A mis padres por su apoyo incondicional a lo largo de la carrera, sin duda son mi pilar y la razón principal de mis logros.

A mis amigos que me retaron intelectualmente y me acompañaron a lo largo de mis estudios universitarios. A compañeros de trabajo y amigos que formé en el banco, principalmente, a la asesora de este proyecto, Fabiola Carolina Cañas Maya, de quien me llevo interminables aprendizajes, gracias por siempre guiarme al camino difícil pero correcto, que al final es el que mayor satisfacción da. A mis sinodales, por su guía y comentarios sobre el trabajo.

A la UNAM por darme lo mejor de dos mundos, la Facultad de Ciencias y el equipo representativo de basquetbol. Al Coach Daniel Gómez León y a todos con quien compartí el deporte representativo, fue un orgullo portar los colores de la universidad dejando todo en la duela cada vez que jugaba, me llevo una familia para toda la vida y aprendizajes tan importantes como los adquiridos en el aula.

Índice general

Agradecimientos **1**

Introducción **1**

1. Bases del aprendizaje estadístico **3**
1.1. Introducción al aprendizaje supervisado . . . 3
1.2. Selección del modelo . . . 5
1.2.1. Algoritmo de selección y evaluación del modelo . . . 6
1.3. Medidas de desempeño . . . 8
1.3.1. Curva ROC y área bajo la curva . . . 10
1.3.2. Curva precisión y exhaustividad y el área bajo la curva . . . 10
1.4. Dilema entre sesgo y varianza . . . 10
1.4.1. Descomposición sesgo-varianza . . . 11

2. Modelos lineales generalizados **14**
2.1. Modelos lineales generalizados . . . 14
2.1.1. Regresión lineal . . . 14
2.1.2. Modelos lineales generalizados . . . 15
2.1.3. Propiedades de la verosimilitud . . . 16
2.1.4. Estimación máxima verosímil . . . 18
2.2. Regresión logística . . . 19
2.2.1. Selección de variables . . . 20

3. Métodos basados en árboles **25**
3.1. Árboles de regresión y de clasificación . . . 25
3.1.1. Árboles de decisión . . . 25
3.1.2. Ajuste de un árbol . . . 28
3.1.2.1. Tratamiento de variables con valores nulos . . . 29
3.1.3. Árboles con pesos modificados . . . 29
3.1.4. Importancia de las variables . . . 30
3.2. Bagging y bosques aleatorios . . . 30
3.2.1. Bagging . . . 30
3.2.2. Bosques aleatorios . . . 31
3.2.3. Importancia de las variables . . . 33
3.3. Boosting adaptativo . . . 33
3.3.1. Relación de boosting adaptativo y un modelo aditivo . . . 35
3.3.2. Uso de la pérdida exponencial . . . 39

3.3.2.1. Problemas de adaboost . . . 40
3.3.3. Importancia de las variables . . . 41
3.4. Boosting del gradiente . . . 41
3.4.1. Optimización mediante descenso del gradiente . . . 42
3.4.1.1. Descenso del gradiente . . . 42
3.4.2. Boosting del gradiente . . . 42
3.4.2.1. Boosting del gradiente en clasificación . . . 43
3.4.3. Hiper-parámetros de boosting del gradiente . . . 45
3.4.4. Importancia de las variables . . . 47

4. Aplicación **48**
4.1. El problema de la retención . . . 48
4.1.1. Definición de abandono de un producto . . . 49
4.2. Tratamiento de los datos . . . 52
4.2.1. Entrenamiento y prueba . . . 52
4.2.2. Balanceo de la muestra de entrenamiento . . . 53
4.3. Proceso de ajuste de los modelos con dependencia temporal . . . 55
4.4. Ajuste regresión logística . . . 55
4.4.1. Selección de variables . . . 56
4.4.2. Resultados clasificación . . . 57
4.5. Ajuste árbol de decisión . . . 58
4.6. Ajuste bagging de árboles de clasificación . . . 60
4.7. Ajuste bosques aleatorios . . . 62
4.8. Ajuste boosting adaptativo . . . 64
4.9. Boosting del gradiente . . . 65
4.10. Selección del modelo final . . . 68

Conclusiones **70**

A. Prueba Zivot-Andrews de cambios estructurales **71**
A.1. Series de tiempo y estacionariedad . . . 71
A.1.1. Prueba Dickey-Fuller . . . 72
A.1.2. Prueba Dickey-Fuller aumentada . . . 73
A.2. Pruebas de cambio estructural de raíces unitarias . . . 75
A.2.1. Prueba Perron . . . 75
A.2.2. Prueba Zivot-Andrews . . . 78

B. Métodos avanzados para el aprendizaje estadístico **79**
B.1. Descomposición sesgo-varianza en un problema de clasificación binario . . . 79
B.2. Descomposición del error general sobre regiones disjuntas . . . 80

Bibliografía **82**

Introducción

El **aprendizaje estadístico** se forma por una serie de herramientas matemáticas y computacionales usadas principalmente para dar sentido a una gran cantidad de información, es decir, para encontrar patrones y entender la estructura subyacente de los datos. El aprendizaje estadístico se utiliza en diversas industrias, por ejemplo en el sector financiero se usa para evaluar el riesgo crediticio de un cliente, mientras que en la medicina se usa para diagnosticar enfermedades. La teoría del aprendizaje estadístico se divide en dos ramas principales que son: **el aprendizaje supervisado** y **el aprendizaje no supervisado**. En el aprendizaje supervisado se busca predecir una variable objetivo por medio de un conjunto de variables de interés que influyen en ésta. Cuando la variable a explicar toma una cantidad finita de valores entonces se dice que es un problema de **clasificación** y cuando la variable objetivo toma valores en los continuos entonces se dice que es un problema de **regresión**. El **aprendizaje no supervisado** es aquel en el cual no se tiene una variable objetivo asociada a un conjunto de variables explicativas, así que no se busca predecirla, sino encontrar una estructura en los datos.
El objetivo de este trabajo es introducir la teoría de algunos métodos de aprendizaje estadístico supervisado de clasificación binaria (la variable objetivo toma únicamente dos valores) y de su aplicación práctica en el sistema bancario.
El primer capítulo de este trabajo introduce las bases teóricas del aprendizaje estadístico supervisado, en pocas palabras, la manera de estimar una función de las variables explicativas a la variable objetivo, con la finalidad de que la estimación generalice a observaciones fuera del conjunto sobre el que se estima, el error que pueda cometer la estimación se descompone en dos principales fuentes: sesgo y varianza.
En el segundo capítulo se abordan los modelos lineales generalizados con un enfoque particular en la regresión logística. La tercera parte introduce los árboles aleatorios, y la manera en que se pueden combinar para disminuir ya sea el sesgo, la varianza o ambos mediante técnicas de ensamble.
Por último en el cuarto capítulo, se desarrolla la aplicación de estos modelos que consiste en identificar de manera preventiva el abandono potencial de clientes en un banco, es decir, clasificar a los clientes en dos grupos:

1. El **grupo de abandono:** clientes que están en riesgo de abandonar el banco en un periodo cercano

2. El **grupo de permanencia:** clientes que no han abandonado el banco y no tienen riesgo inmediato de abandonar.

Se ajustan la regresión logística y los métodos basados en árboles con las técnicas explicadas. Se selecciona el modelo con el mejor desempeño, en términos del área bajo la curva ROC y la bajo la curva ROC precisión y exhaustividad, mostrando ventajas comparativas sobre los demás.

Capítulo 1

Bases del aprendizaje estadístico

Resumen

En este capítulo damos una breve introducción al aprendizaje estadístico. Se cubren las bases teóricas en las que se sustenta la aplicación que se presenta en el capítulo 4.

Se define qué es el aprendizaje estadístico supervisado, también se explica el procedimiento de estimación usado en cada uno de los modelos introducidos en el trabajo y por último se presentan algunas formas de medir el error en un problema de clasificación binario.

1.1. Introducción al aprendizaje supervisado

El **aprendizaje estadístico** es una serie de métodos estadísticos con el objetivo de dar sentido a datos, encontrando patrones en ellos.

- El **aprendizaje supervisado** considera problemas donde hay una variable respuesta asociada a una serie de variables de entrada, estimando una función que pueda pronosticar la variable respuesta.

El conjunto de aprendizaje $\mathcal{L}$ es la colección de pares $\{(x_i, y_i)\}_{i=1}^{N}$. con $x_i \in \mathbb{R}^p$ y $y_i \in \mathbb{R}$. Representa el total de datos disponibles sobre el problema.

Entonces el objetivo es, dados $(x, y) \in \mathcal{L}$ encontrar la función de predicción $\varphi_{\mathcal{L}} : \mathbb{R}^p \longrightarrow \mathbb{R}$ tal que una función de pérdida $L(y, \varphi_{\mathcal{L}}(x))$ **sea mínima** y pueda usarse para predecir usando datos fuera del conjunto de aprendizaje.

- Si y es discreta se dice que es un **problema de clasificación** y $\varphi_{\mathcal{L}}$ es un **clasificador**.

- Si y es continua se dice que es un **problema de regresión** y $\varphi_{\mathcal{L}}$ es un **regresor**.

Si cada $(x_i, y_i) \in \mathcal{L}$ son realizaciones del vector aleatorio (X, Y) Entonces la manera de medir el desempeño de φ globalmente (no solo dentro de $\mathcal{L}$) es con el **error de predicción**.

$$Err\varphi_{\mathcal{L}} = \mathbb{E}_{X,Y}[L(\varphi_{\mathcal{L}}(X), Y)] \tag{1.1}$$

Observación 1.1. Se hace explícita la dependencia de $\varphi_{\mathcal{L}}$ sobre el conjunto de entrenamiento $\mathcal{L}$, pues esta función se estima sobre este conjunto. Notemos que el error de predicción es la esperanza sobre el espacio de probabilidad, no sobre el conjunto de aprendizaje.

Definición 1.2. El **modelo de Bayes** es la función φ_B que resuelve la ecuación 1.1 para cada $x \in \mathbb{R}$, es decir tiene el menor error de predicción. Por ende es el mejor modelo obtenible. Usando la ley de la esperanzas iteradas sobre el error de predicción obtenemos el error residual.

$$Err\varphi_B(x) = \mathbb{E}_{X,Y}[\mathbb{E}_{Y|X}[L(\varphi_B(X), Y)]]$$

Entonces para cada x, la función φ_B es aquella que reduce la esperanza condicional

$$\varphi_B(x) = \arg\min_y \mathbb{E}_{Y|X=x}[L(y, Y)]] \tag{1.2}$$

El modelo de Bayes se puede encontrar de manera analítica, en un problema de clasificación y en un problema de regresión con las siguientes funciones de pérdida, donde $\mathbf{1}$ es una función indicadora.

- Error de clasificación $\quad L(\varphi(x), y) = \mathbf{1}(\varphi(x) \neq y)$

$$\begin{aligned}
\varphi_B &= \arg\min_y \mathbb{E}_{Y|X=x}[\mathbf{1}(Y \neq y)] \\
&= \arg\min_y \mathbb{P}(Y \neq y | X = x) \\
&= \arg\min_y 1 - \mathbb{P}(Y = y | X = x) \\
&= \arg\max_y \mathbb{P}(Y = y | X = x)
\end{aligned}$$

  Es decir, corresponde a clasificar a aquella clase $y \in \{c_1, ..., c_K\}$ que sea más probable.

- Error cuadrático $\quad L(\varphi(x), y) = (\varphi(x) - y)^2$
  De acuerdo a la proposición 1.4 obtenemos

$$\begin{aligned}\varphi_B(x) &= \arg\min_y \mathbb{E}_{Y|X=x}[(Y-y)^2] \\ &= \mathbb{E}_{Y|X=x}[Y]\end{aligned}$$

  Se obtiene, que en términos del error cuadrático medio, el estimador óptimo es la esperanza condicional en x. Aún más, es un estimador insesgado, tomando esperanza $\mathbb{E}[X - \varphi_B] = \mathbb{E}[Y] - \mathbb{E}[Y] = 0$, por esperanzas iteradas.

Observación 1.3. "En problemas prácticos, la probabilidad conjunta de X e Y es desconocida y el modelo de Bayes no se puede encontrar analíticamente" [7, p. 13]. Así que buscaremos modelos que estimen φ_B a su vez estimando el error de predicción 1.1 y buscando reducirlo.

Proposición 1.4. *Sea Y variable aleatoria, tal que $\mathbb{E}[|Y|] < \infty$, y $f(x)$ una función. Entonces $\mathbb{E}_{Y|X=x}[(Y - f(x))^2]$ se minimiza en $f(x) = \mathbb{E}_{Y|X=x}[Y]$.*

$$\begin{aligned}\frac{\partial}{\partial f}\mathbb{E}_{Y|X=x}[(Y-f(x))^2] &= 0 \\ \mathbb{E}_{Y|X=x}[-2(Y-f(x))] &= 0 \\ \textit{finalmente despejando}& \\ f(x) &= \mathbb{E}_{Y|X=x}[Y]\end{aligned}$$

y la segunda derivada es 2*, por lo tanto es el mínimo.*

Observación 1.5. Es importante notar que aunque este trabajo se enfoca solamente en modelos de clasificación, uno de ellos **boosting del gradiente**, que se encuentra en la sección 3.4, usa un modelo de regresión internamente, para ajustar el modelo de clasificación, por esta razón introducimos ambos problemas.

Veremos en la sección 3.1 de árboles de regresión y clasificación que los árboles de decisión siguen estos principios, clasifican a la clase cuya probabilidad estimada sea mayor, y en el caso de regresión la estimación es el promedio de la región correspondiente.

1.2. Selección del modelo

Si conociéramos la distribución verdadera de (X, Y) el modelo Bayes solucionaría el problema de la estimación de φ. En la práctica se desconoce su distribución y resulta más

preciso seleccionar φ de una serie de modelos eligiendo aquel que "menos se equivoque", es decir, debemos estimar el error general del modelo, de la ecuación 1.1. Las dos formas más comunes son:

1. **Validación**
   Si se estima el modelo en el conjunto $\mathcal{L}$, se estima el error sobre otro conjunto ajeno a este, $\mathcal{L}'$ de N' elementos. Esto es, el error estimado esperado del estimador sobre el conjunto de validación es:

$$\bar{E}(\varphi_{\mathcal{L}}, \mathcal{L}') = \frac{1}{N'} \sum_{i=1}^{N'} L(\varphi_{\mathcal{L}}(x_i), y_i) \tag{1.3}$$

2. **Validación cruzada**
   Se parte el conjunto de entrenamiento $\mathcal{L}$ en K conjuntos ajenos por muestreo aleatorio simple denotados $\mathcal{L}_k$, $k = 1, ..., K$. Se ajusta un modelo sobre $\mathcal{L} \backslash \mathcal{L}_k$ y se mide su error sobre $\mathcal{L}_k$ para $k = 1, ..., K$. Se toma el promedio de los K errores. El estimador del error general del modelo es:

$$\widehat{Err}^{CV}(\varphi_{\mathcal{L}}) = \frac{1}{K} \sum_{k=1}^{K} \bar{E}(\varphi_{\mathcal{L} \backslash \mathcal{L}_k}, \mathcal{L}_k) \tag{1.4}$$

Se recomienda usar validación cruzada ya que en este método cada una de las observaciones del conjunto de aprendizaje se usa para la estimación del error general, aunque no siempre es factible su uso, pues requiere la estimación de K modelos, lo cuál puede requerir bastante tiempo computacional, en tal caso, usar validación. Además validación cruzada al ser el promedio de varias validaciones reduce la varianza del error.

1.2.1. Algoritmo de selección y evaluación del modelo

La función φ depende de parámetros y de hiper-parámetros.

- Los **parámetros** son internos del modelo y dependen de los datos proporcionados
- Los **hiper-parámetros** son aquellos que regulan la ejecución del modelo/algoritmo y son dados *a priori* antes de la ejecución sobre datos.

Hacemos notar la dependencia del algoritmo de aprendizaje sobre los hiper-parámetros θ y el conjunto de aprendizaje, con la notación $\mathcal{A}(\theta, \mathcal{L}) = \varphi$. Si θ son la colección de hiper-parámetros que controlan la ejecución del algoritmo de aprendizaje $\mathcal{A}$, entonces buscamos encontrar el valor de los hiper-parámetros que generen un modelo con el mínimo error general.

$$\theta^* = \arg\min_{\theta} Err\mathcal{A}(\theta, \mathcal{L})$$

Con este propósito, es necesario dividir el conjunto de aprendizaje en tres conjuntos ajenos. Si se cuentan con un conjunto de aprendizaje $\mathcal{L}$, se divide en:

- $\mathcal{L}_{\text{Ent}}$: El conjunto de entrenamiento sirve para entrenar el modelo
- $\mathcal{L}_{\text{Val}}$: El conjunto de validación, se le da uso, para escoger los hiper-parámetros que minimicen la estimación del error de predicción
- $\mathcal{L}_{\text{Pru}}$: El conjunto de prueba se usa para tener un conjunto en el cual se mide el desempeño del modelo y que no haya sido usado en la elección de los hiper-parámetros.

Basándonos en metodología planteada en [7], se propone usar el siguiente algoritmo para la selección del modelo y su evaluación.

Selección y evaluación del modelo

1. Dividir el conjunto de aprendizaje en tres $\mathcal{L}_{\text{Ent}}$, $\mathcal{L}_{\text{Val}}$ y $\mathcal{L}_{\text{Pru}}$
2. Hacer la selección del modelo sobre $\mathcal{L}_{\text{Ent}} \cup \mathcal{L}_{\text{Val}}$ usando *a*) **validación** o *b*) **validación cruzada**
   a) Ajustando el modelo sobre $\mathcal{L}_{\text{Ent}}$ y **validando** sobre $\mathcal{L}_{\text{Val}}$ seleccionando los parámetros que produzcan el menor error en la validación.
   $$\hat{\theta}^* = \arg\min_{\theta} \bar{E}(\mathcal{A}(\theta, \mathcal{L}_{\text{Ent}}), \mathcal{L}_{\text{Val}}) \tag{1.5}$$
   b) Usando **validación cruzada** sobre $\mathcal{L}^* = \mathcal{L}_{\text{Ent}} \cup \mathcal{L}_{\text{Val}}$ seleccionando los hiper-parámetros tengan la menor estimación del error.
   $$\hat{\theta}^* = \arg\min_{\theta} \widehat{Err}^{CV}(\varphi_{\mathcal{L}^*}) \tag{1.6}$$
3. Se estima el error del modelo final como
   $$\bar{E}(\mathcal{A}(\theta, \mathcal{L}_{\text{Ent}} \cup \mathcal{L}_{\text{Val}}), \mathcal{L}_{\text{Pru}}) \tag{1.7}$$
4. Se ajusta el modelo final $\mathcal{A}(\theta, \mathcal{L})$ sobre el conjunto completo $\mathcal{L}$

FIGURA 1.1: Algoritmo selección y evaluación del modelo

Se busca que se acumule 60 %, 20 % y 20 %, en entrenamiento, validación y prueba respectivamente, o alguna proporción similar tal que las distribuciones de los grupos sean iguales, por esta razón, los conjuntos son seleccionados aleatoriamente.

Observación 1.6. Podría parecer natural estimar el error final con $\bar{E}(\mathcal{A}(\theta^*, \mathcal{L}_{\text{Ent}}), \mathcal{L}_{val})$ o con $\widehat{Err}^{CV}(\varphi_{\mathcal{L}^*})$ pero el problema es que estas estimaciones del error no son independientes de $\mathcal{L}_{\text{Val}}$, ¡porque se usa este conjunto para la selección de θ^*! Por eso se tiene el conjunto $\mathcal{L}_{\text{Pru}}$ que no se involucra en la selección del modelo, para dar un estimado final del error que no haya sido sesgado por los datos.

1.3. Medidas de desempeño

En esta sección introducimos algunas medidas de desempeño en un problema de clasificación binario que son: exactitud, matriz de confusión, precisión, exhaustividad, especificidad, curva ROC y curva PE. Hasta ahora solo hemos presentado el error de clasificación, es decir $\frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(\varphi(x) \neq y)$. Las medidas de desempeño que se presentan en esta sección se basan en la matriz de confusión, figura 1.2. La matriz de confusión compara las cuatro combinaciones de un clasificador binario con el valor verdadero de la variable respuesta. Si 1 representa la ocurrencia del evento de interés y 0 la no ocurrencia, entonces la diagonal de la matriz son los valores correctamente clasificados y fuera de la diagonal los incorrectamente clasificados.

		Valor verdadero	
		0	1
Valor predicción	0	Verdaderos Negativos (VN)	Falsos Negativos (FN)
	1	Falso Positivos (FP)	Verdaderos Positivos (VP)

FIGURA 1.2: Matriz de confusión

Las medidas de desempeño son las siguientes

La **exactitud** es el la proporción de las estimaciones que son atinadas. La **precisión** de un clasificador binario mide la proporción de los pronósticos que son correctos. La **exhaustividad** o **sensitividad** de un clasificador binario mide la proporción de unos del total de las observaciones que logra pronosticar. La **especificidad** es análoga a la exhaustividad pero para la clase de ceros.

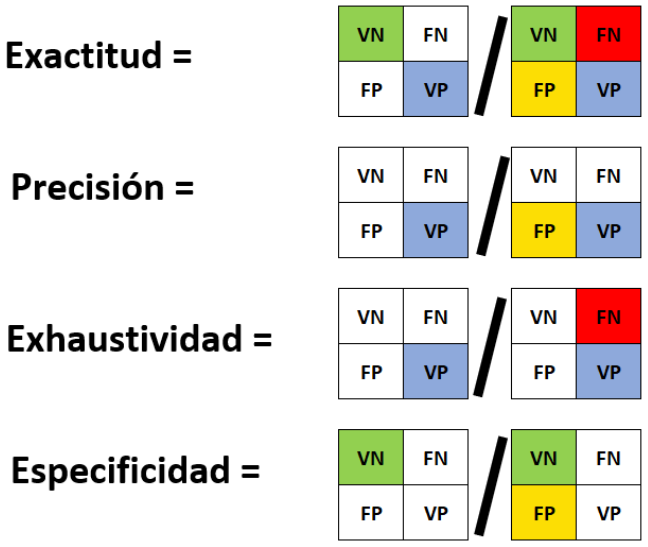


FIGURA 1.3: Medidas de desempeño

Una medida popular que busca un balance entre la precisión y exhaustividad de un clasificador es la **puntuación F1**, esta es la media armónica de estas dos cantidades, $F1 \in (0,1)$. La puntuación F1 es menor para valores muy separados que para valores cercanos, aunque su media aritmética sea la misma. esto da un balance entre la precisión y la exhaustividad.

$$\begin{aligned} F1 &= \left(\frac{Prec + Exh}{2}\right)^{-1} \\ &= 2\frac{Prec \cdot Exh}{Prec + Exh} \end{aligned}$$

Estas medidas son importantes porque si un problema de clasificación binario está sesgado a una clase el error de clasificación no es una medida confiable, supongamos que en la muestra de prueba el 99 % de las observaciones tienen asociada la respuesta $y = 0$ y solo el 1 % tienen asociada la respuesta $y = 1$. Si nuestro clasificador es la constante cero, $\hat{\varphi}(x) = 0$ entonces tendríamos una exactitud del 99 % (error de clasificación del 1 %), pero en realidad no estaríamos capturando ninguna de las observaciones de la clase de interés. En este caso sería mas adecuado tener una exactitud menor, pero capturar mayor porcentaje de los $1's$ en la muestra, o tal vez pronosticar pocos $1's$ pero que estos pronósticos sean atinados. La precisión y la exhaustividad miden justamente esto.

Observación 1.7. Las medidas de desempeño que hemos presentado hasta ahora presuponen que la estimación de la respuesta es un valor de dos clases $\{0,1\}$, pero los modelos de clasificación que se presentan en este trabajo generan como estimación de la clase de unos, una puntuación en $(0,1)$. Podemos fijar un corte fijo $c \in (0,1)$ se obtienen las clases a las cuales pronosticar, la clase de unos si $\hat{\varphi}(x) > c$ y la clase de ceros si

$\hat{\varphi}(x) <= c$. Pero para poder medir el desempeño de los modelos sin depender del punto de corte se presenta la curva ROC y la curva PE.

1.3.1. Curva ROC y área bajo la curva

La curva ROC, en cada punto de corte $c \in (0, 1)$, calcula la proporción de los unos que se clasifican correctamente (exhaustividad) y uno menos la proporción de los ceros que se clasifican correctamente (especificidad). Asigna la misma importancia a los ceros y a los unos que logra capturar el modelo. Al igual que la exactitud, si la muestra esta sesgada a uno de los dos valores, esta métrica se va a ver afectada, pues una de las dos razones va a ser muy alta siempre. Aún con esta consideración la curva ROC es la métrica más usada. Si calcula la integral de esta curva se obtiene el área bajo la curva ROC (ABC ROC), que es la medida que se presenta normalmente, la cual se puede observar en la figura 1.4.

El punto de referencia para la ABC ROC es del 50 % pues si se tiene un modelo por debajo de está métrica, si se voltean las clasificaciones, es decir el nuevo clasificador es $1 - \hat{\varphi}(x)$, el área del 50 % se verá superada, como se puede apreciar en la figura 1.4. La elección del modelo de acuerdo a esta medida, sería el modelo con la área bajo la curva ROC mayor, en el conjunto de prueba.

1.3.2. Curva precisión y exhaustividad y el área bajo la curva

Así mismo para cada posible corte $c \in (0, 1)$ podemos obtener la precisión y exhaustividad correspondiente y graficarlos en función del corte, esta métrica se enfoca completamente en la capacidad de clasificar correctamente la clase de interés, y es más robusta al desbalance de la muestra. Nuevamente se puede obtener el área bajo la curva (PE ABC) integrando la curva obtenida, lo cual se observa en la figura 1.5. El valor óptimo o mínimo del área bajo la curva depende del problema en el que se trabaja, se sugiere usar más para comparar el desempeño de distintos modelos binarios, a tener un valor mínimo de esta medida como punto de referencia.

1.4. Dilema entre sesgo y varianza

El error de la función de predicción se puede descomponer en una fuente de error irreducible (el error de Bayes), el sesgo y la varianza. El dilema entre sesgo y varianza es la problemática de reducir el error que no proviene del modelo de Bayes atacando alguna de sus dos fuentes: sesgo o varianza, encontrando un balance entre ambas.

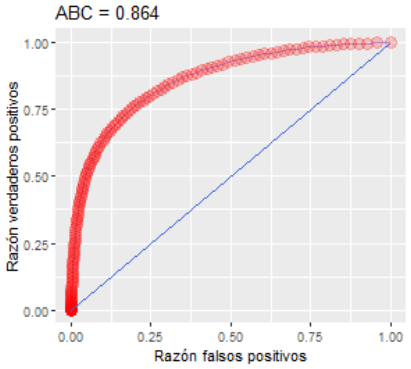


FIGURA 1.4: ABC ROC

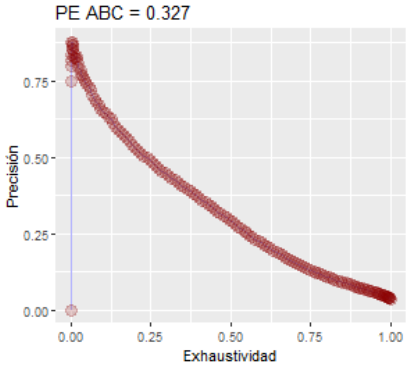


FIGURA 1.5: PE ABC

Si el sesgo es grande la función de predicción no se acerca al verdadero valor de la variable objetivo, y se podría pensar que el modelo no logra capturar la estructura de los datos, se dice entonces que hay **sub-ajuste**.

Si la varianza es grande, las predicciones varían mucho y podemos pensar que el modelo captura demasiada de la estructura de los datos de entrenamiento lo que lleva a que cometa errores en el conjunto de prueba pues no logra generalizar adecuadamente, por eso se dice que hay **sobre-ajuste**.

La complejidad del modelo es el conjunto de hiper-parámetros que controlan la capacidad del modelo para capturar la estructura de los datos. Por ejemplo, en el capítulo 3, la complejidad puede ser la profundidad del árbol, o el parámetro de complejidad. A mayor profundidad del árbol, más nodos tienen el árbol y puede capturar más características de los datos sobre los cuáles se entrena, por lo que puede sobre-ajustar. Si el parámetro de complejidad es pequeño, de igual manera se ajustan árboles de mayor profundidad que por lo mencionado llegan a sobre-ajustar. Al incrementar la complejidad de un modelo esperamos que la varianza suba y el sesgo baje y si bajamos la complejidad de un modelo entonces esperemos que baje la varianza, pero aumente el sesgo. Como se puede observar en la figura 1.6.

1.4.1. Descomposición sesgo-varianza

La descomposición del error en los componentes: irreducible, sesgo y varianza depende del tipo de problema, regresión o clasificación, asimismo depende de la función de pérdida usada. Se presenta el caso mas sencillo, usando pérdida cuadrática y un problema de regresión.

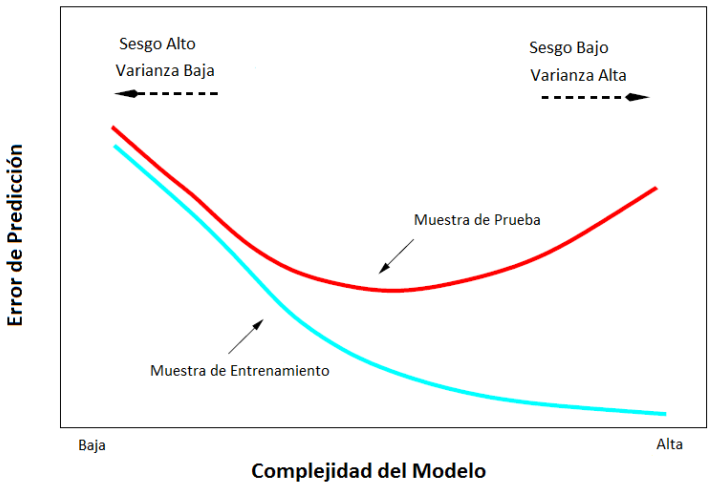


FIGURA 1.6: Dilema sesgo varianza

Consideremos el error general $Err\varphi$ condicionado a $X = x$, esto es:

$$Err\varphi(x) = \mathbb{E}_{Y|X=x}[L(Y, \varphi_{\mathcal{L}}(x))]$$

Como la función de pérdida es la pérdida cuadrática, e introduciendo el modelo de Bayes se obtiene.

$$\begin{aligned} Err\varphi(x) &= \mathbb{E}_{Y|X=x}[(Y - \varphi_B(x) + \varphi_B(x) - \varphi_{\mathcal{L}}(x))^2] \\ &= \mathbb{E}_{Y|X=x}[(Y - \varphi_B(x))^2] + \mathbb{E}_{Y|X=x}[(\varphi_B(x) - \varphi_{\mathcal{L}}(x))^2] \\ &\quad + 2\mathbb{E}_{Y|X=x}[(Y - \varphi_B(x))(\varphi_B(x) - \varphi_{\mathcal{L}}(x))] \\ &= \underbrace{Err\varphi_B(x)}_{\substack{\text{Error residual} \\ \text{irreducible}}} + \underbrace{(\varphi_B(x) - \varphi_{\mathcal{L}}(x))^2}_{\text{Discrepancia con Bayes}} + 0 \end{aligned} \tag{1.8}$$

ya el modelo de Bayes es la esperanza condicional en x $\varphi_B(x) = \mathbb{E}_{Y|X=x}[Y]$.
Más aún, si consideramos la aleatoriedad de $\mathcal{L}$, que surge al escoger un conjunto de aprendizaje de tamaño N, sobre todos los conjuntos de entrenamiento posibles, $\mathbb{E}[\varphi_{\mathcal{L}}(x)]$ es la esperanza de la predicción sobre todos los conjuntos $\mathcal{L}$. Tomando la esperanza de

1.8 obtenemos:

$$\begin{aligned}
&\mathbb{E}_{\mathcal{L}}[(\varphi_B(x) - \varphi_{\mathcal{L}}(x))^2] \\
&\quad = \mathbb{E}_{\mathcal{L}}\left[(\varphi_B(x) - \mathbb{E}[\varphi_{\mathcal{L}}(x)] + \mathbb{E}[\varphi_{\mathcal{L}}(x)] - \varphi_{\mathcal{L}}(x))^2\right] \\
&\quad = \mathbb{E}_{\mathcal{L}}\left[(\varphi_B(x) - \mathbb{E}[\varphi_{\mathcal{L}}(x)])^2\right] + \mathbb{E}_{\mathcal{L}}\left[(\mathbb{E}[\varphi_{\mathcal{L}}(x)] - \varphi_{\mathcal{L}}(x))^2\right] \\
&\qquad + 2\mathbb{E}_{\mathcal{L}}\left[(\varphi_B(x) - \mathbb{E}[\varphi_{\mathcal{L}}(x)])(\mathbb{E}[\varphi_{\mathcal{L}}(x)] - \varphi_{\mathcal{L}}(x))\right] \\
&\quad = \underbrace{(\varphi_B(x) - \mathbb{E}[\varphi_{\mathcal{L}}(x)])^2}_{\text{Sesgo cuadrado}} + \underbrace{\mathbb{E}_{\mathcal{L}}\left[(\mathbb{E}[\varphi_{\mathcal{L}}(x)] - \varphi_{\mathcal{L}}(x))^2\right]}_{\text{Varianza}} + 0
\end{aligned}$$

Juntando todo lo anterior obtenemos la descomposición final del error de predicción general esperado

$$\mathbb{E}_{\mathcal{L}}[Err\varphi_{\mathcal{L}}(x)] = Ruido(x) + Sesgo^2(x) + Var(x) \tag{1.9}$$

donde

$$\begin{aligned}
Ruido(x) &= Err\varphi_B(x) \\
Sesgo^2(x) &= (\varphi_B(x) - \mathbb{E}[\varphi_{\mathcal{L}}(x)])^2 \\
Var(x) &= \mathbb{E}_{\mathcal{L}}\left[(\mathbb{E}[\varphi_{\mathcal{L}}(x)] - \varphi_{\mathcal{L}}(x))^2\right]
\end{aligned}$$

Recordemos que todo es función de x, porque se las estimaciones se realizan sobre las observaciones del conjunto de aprendizaje. Como mencionamos antes, esta descomposición es válida para la función de pérdida cuadrática en un problema de regresión. "En un problema de clasificación se han hecho esfuerzos con la función de pérdida del error de clasificación $\mathbb{E}_{\mathcal{L}}[\mathbb{E}_{Y|X=x}][1(\varphi_{\mathcal{L}}(x) \neq Y)]$ redefiniendo los conceptos de sesgo y varianza para que sean adecuados a un problema de clasificación, destacados son los trabajos de Dietterich y Kong [1995], Breiman [1996], Kohavi et al. [1996], Tibshirani [1996] y Domingos [2000] [7]. Se puede dar otro enfoque a este problema porque usualmente la estimación de los problemas de clasificación no es como tal la clase final, sino que se estima la probabilidad de que la respuesta pertenezca a cierta clase $\hat{P}(Y = c|X = x)$ para $c \in \{c_1, ..., c_K\}$. A este estimado se le puede hacer una descomposición parecida a 1.9 que puede parecer larga y poco intuitiva en comparación a la descomposición de un problema de regresión con mínimos cuadrados. Para más detalles se puede consultar en el Anexo B.1

Capítulo 2

Modelos lineales generalizados

Resumen

En este capítulo se presentan los modelos lineales generalizados, los cuales establecen una relación lineal entre una función invertible de la media de una distribución en la familia exponencial canónica y las variables explicativas. De particular interés para este trabajo es el uso de la distribución Bernoulli para modelar la respuesta, en un problema de clasificación binario, conocido como regresión logística. Se presentan un algoritmo para seleccionar variables (Stepwise) y algunas medidas de bondad de ajuste del modelo.

2.1. Modelos lineales generalizados

2.1.1. Regresión lineal

La regresión lineal es un método supervisado cuyo objetivo es predecir una **variable respuesta** a través de la combinación lineal de las variables explicativas, también llamadas **regresores**. Este método es útil cuando la variable de respuesta toma valores en todos los números reales y existen relaciones lineales entre los regresores y la respuesta. La regresión lineal es uno de los modelos más estudiados y usados debido a su rápida implementación, interpretación y sencillez en la capacidad para determinar la significancia del efecto de los regresores en la respuesta.

Dado un conjunto de entrenamiento $\mathcal{L}_{ent} = \{(x_i, y_i)\}_{i=1}^{N}$ el modelo lineal asume

1. $Y_i|X_i \sim \mathcal{N}(\mu(X_i), \sigma^2)$ independientes
2. $\mu(X_i) = X_i^T\beta$ donde $\mu(X_i) = \mathbb{E}[Y_i|X_i]$

Donde $\beta = (\beta_0, \beta_1, ..., \beta_k)^T$ son los pesos asociados a cada i-ésimo elemento del conjunto de entrenamiento $x_i = [1, x_{i,1}, ..., x_{i,k}]^T$.

Observación 2.1. El modelo lineal asume que las variables respuesta son distribuidas normalmente con la varianza constante y que son independientes entre ellas. Además asume que no haya dependencia lineal entre las variables explicativas, es decir que no haya multicolinealidad. Esta no es la forma más general del modelo lineal, el teorema de Gauss-Markov, menciona condiciones menos rígidas para obtener el mejor estimador insesgado, en términos de la varianza. Pero en la formulación que se presenta, se cumplen las condiciones del teorema de Gauss-Markov.

2.1.2. Modelos lineales generalizados

En contraste con la regresión lineal, los modelos lineales generalizados (MLG) amplían dos supuestos generalizando tanto la distribución de $Y|X$ como la relación entre la respuesta y las variables regresoras. Por lo que la distribución condicional de Y no va a ser necesariamente normal, sino que va a pertenecer a una familia de distribuciones llamada familia exponencial canónica.

Definición 2.2. Dada conjunto de entrenamiento $\mathcal{L}_{ent} = \{(x_i, y_i)\}_{i=1}^N$ un **modelo lineal generalizado** asume

1. $Y_i|X_i \sim$ Familia exponencial canónica independientes
2. Existe una función enlace g invertible, tal que $g(\mu(X_i)) = X_i^T\beta$ donde $\mu(X_i) = \mathbb{E}[Y_i|X_i]$

Los modelos lineales generalizados asumen que la respuesta sigue una distribución en la familia exponencial canónica, independientes. También asume que en las variables respuesta no exista multicolinealidad.

La familia exponencial canónica tiene propiedades que son claves para los MLGs y la estimación de sus parámetros. La forma particular que tiene la distribución provee una manera sencilla de establecer la función de enlace entre la media y los parámetros de la distribución, esto se puede ver en la proposición 2.6.

Definición 2.3. La **familia exponencial canónica** es una familia de distribuciones tal que para $\theta \in \mathbb{R}$ y $\phi \in \mathbb{R}$ la densidad puede ser expresada como

$$f_\theta(y) = exp\left(\frac{y\theta - b(\theta)}{\phi} - c(y, \phi)\right) \tag{2.1}$$

para funciones reales $b(\cdot)$ y $c(\cdot, \cdot)$. La tabla 2.4 muestra tres distribuciones de la familia exponencial canónica, sus rangos y parámetros asociados

Observación 2.4. El parámetro θ puede ser una transformación del parámetro original de la distribución, adecuada para que la densidad tome la forma vista en le ecuación 2.1. En la tabla 2.4, se muestran algunas distribuciones pertenecientes a la familia exponencial canónica, su rango, parámetros y funciones.

Distribución	$\mathcal{N}(\mu, \sigma^2)$	$Pois(\lambda)$	$Ber(p)$
Rango de Y	$(-\infty, -\infty)$	$\mathbb{N} \cup \{0\}$	$\{0, 1\}$
θ	μ	$log\ \lambda$	$log(\frac{p}{1-p})$
$b(\theta)$	$\theta^2/2$	$exp(\theta)$	$log(1 + exp(\theta))$
$c(y, \phi)$	$-\frac{1}{2}(\frac{y^2}{\phi} + log(2\pi\phi))$	-log(y!)	1
ϕ	σ^2	1	1

TABLA 2.1: Familia exponencial canónica.

2.1.3. Propiedades de la verosimilitud

Proposición 2.5. *Sea Y una variable aleatoria con densidad f_θ con log-verosimilitud $l(\theta) = \log\ f_\theta(Y)$ entonces, si $\mathbb{E}[Y^2] < \infty$, se cumplen las siguientes identidades.*

$$\mathbb{E}[\frac{\partial l}{\partial \theta}] = 0 \tag{2.2}$$

$$\mathbb{E}[\frac{\partial^2 l}{\partial \theta^2}] + \mathbb{E}[\frac{\partial l}{\partial \theta}]^2 = 0 \tag{2.3}$$

Demostración. Por ser densidad $\int_{\mathbb{R}} f_\theta(y)dy = 1$, derivando con respecto a θ obtenemos

$$\begin{aligned}
0 &= \frac{\partial}{\partial \theta} \int_{\mathbb{R}} f_\theta(y)dy \\
&= \int_{\mathbb{R}} \frac{\partial}{\partial \theta} f_\theta(y)dy \\
&= \int_{\mathbb{R}} \frac{\partial}{\partial \theta} \exp\left(\log(f_\theta(y))\right) dy \\
&= \int_{\mathbb{R}} f_\theta(y) \frac{\partial}{\partial \theta} \log\ f_\theta(y)dy \\
&= \mathbb{E}[\frac{\partial}{\partial \theta} l(\theta)]
\end{aligned}$$

Derivando respecto a θ nuevamente

$$\begin{aligned}
0 &= \frac{\partial}{\partial\theta}\mathbb{E}[\frac{\partial}{\partial\theta}l(\theta)] \\
&= \frac{\partial}{\partial\theta}\int_{\mathbb{R}} f_\theta(y)\frac{\partial}{\partial\theta}\log\ f_\theta(y)dy \\
&= \int_{\mathbb{R}} \frac{\partial}{\partial\theta}f_\theta(y)\frac{\partial}{\partial\theta}\log\ f_\theta(y)dy \\
&= \int_{\mathbb{R}} f_\theta(y)\frac{\partial^2}{\partial\theta^2}\log\ f_\theta(y)dy + \int_{\mathbb{R}} \frac{\partial}{\partial\theta}\exp(\log\ f_\theta(y))\frac{\partial}{\partial\theta}\log\ f_\theta(y)dy \\
&= \mathbb{E}[\frac{\partial^2}{\partial\theta^2}l(\theta)] + \int_{\mathbb{R}} f_\theta(y)\left(\frac{\partial}{\partial\theta}\log\ f_\theta(y)\right)^2 dy \\
&= \mathbb{E}[\frac{\partial^2}{\partial\theta^2}l(\theta)] + \mathbb{E}\left[(\frac{\partial}{\partial\theta}l(\theta))^2\right]
\end{aligned}$$

□

Proposición 2.6. *Para las distribuciones de la familia exponencial canónica si b es dos veces diferenciable la media y varianza se pueden obtener por:*

$$\mathbb{E}[Y] = b'(\theta) \tag{2.4}$$
$$Var(Y) = b''(\theta)\phi \tag{2.5}$$

Demostración. La log-verosimilitud es

$$l(\theta) = \frac{Y\theta - b(\theta)}{\phi} + c(y,\phi) \tag{2.6}$$

Tomando la derivada parcial $\frac{\partial}{\partial\theta}$

$$\frac{\partial}{\partial\theta}l(\theta) = \frac{Y - b'(\theta)}{\phi} \tag{2.7}$$

usando 2.2 y tomando la esperanza obtenemos

$$\mathbb{E}[Y] = b'(\theta) \tag{2.8}$$

Para derivar la varianza usamos 2.3.

$$\begin{aligned}
\frac{\partial^2}{\partial\theta^2}l(\theta) + (\frac{\partial}{\partial\theta}l(\theta))^2 &= \frac{-b''(\theta)}{\phi} + (\frac{Y - b'(\theta)}{\phi})^2 \\
\frac{Var(Y)}{\phi^2} &= \frac{b''(\theta)}{\phi} \qquad \text{tomando esperanza} \\
Var(Y) &= \phi b''(\theta)
\end{aligned}$$

□

Nuestro objetivo es estimar los parámetros β mediante máxima verosimilitud, para hacer el ajuste del modelo. Para esto necesitamos que los parámetros aparezcan dentro de la especificación de la densidad de la familia exponencial canónica. Gracias a la ecuación 2.8 podemos lograr esto, ya que la función b' nos lleva de $\theta \to \mu$ y el enlace nos lleva de $\mu \to \beta$ como se puede observar en el siguiente diagrama.

$$\theta \xleftrightarrow{b'} \mu \xleftrightarrow{g} \beta$$

El enlace canónico se define como la función enlace que hace el recorrido de θ a β lo más sencillo posible. El enlace canónico es la función $g(\mu) = (b')^{-1}(\mu)$ por lo que $\theta = x_i^T\beta$. La definición formal es la siguiente:

Definición 2.7. El **enlace canónico** es la función g que mapea cada $\mu(x_i)$ al parámetro canónico θ de la familia exponencial canónica esto es

$$g(\mu(x_i)) = \theta \tag{2.9}$$

Por la ecuación 2.8 $\mu(x_i) = b'(\theta)$, el enlace canónico esta dado por:

$$g(\mu(x_i)) = (b')^{-1}(\mu(x_i)) = x_i^T\beta \tag{2.10}$$

Observación 2.8. Si $\phi \geq 0$ entonces $Var(Y) = \phi b''(\theta) > 0$ por lo que $b'' > 0$ entonces b' es creciente y b es convexa. Además $(b')^{-1}$ es también creciente ya que su inverso lo es y en consecuencia el enlace canónico es una función estrictamente creciente.

2.1.4. Estimación máxima verosímil

Dada una muestra de entrenamiento $\mathcal{L}_{ent}$ de tamaño N tal que la distribución de las variables aleatorias Y_i pertenece a la familia exponencial canónica de parámetro canónico θ_i y parámetro de dispersión ϕ, para encontrar los parámetros máximos verosímiles β notemos que

$$\theta_i = (b')^{-1}(\mu(x_i)) \tag{2.11}$$
$$= (b')^{-1}(g^{-1}(x_i^T\beta)) \tag{2.12}$$

entonces definimos la función $h = (b')^{-1} \circ g^{-1} = (g \circ b')^{-1}$, la log-verosimilitud de la muestra es (quitando el componente de la densidad $c(y_i, \varphi)$ ya que no depende de β y

no afecta la optimización)

$$l_N(\beta; y, x) = \sum_{i=1}^{N} \frac{y_i\theta_i - b(\theta_i)}{\phi} \tag{2.13}$$

$$= \sum_{i=1}^{N} \frac{y_i h(x_i^T\beta) - b(h(x_i^T\beta))}{\phi} \tag{2.14}$$

Observación 2.9. Usando el enlace canónico la log-verosimilitud es mas simple

$$l_N(\beta; y, x) = \sum_{i=1}^{N} \frac{y_i x_i^T\beta - b(x_i^T\beta)}{\phi} \tag{2.15}$$

Además la expresión 2.15 es la suma de una función lineal y una función cóncava (observación 2.8) por lo que l_N es estrictamente cóncava lo cual garantiza la existencia de un máximo global. A diferencia de la regresión lineal no es posible dar una forma cerrada de la solución $\hat{\beta}$ que maximice la log-verosimilitud (y en consecuencia la verosimilitud), no obstante se usan algoritmos para aproximar el verdadero valor, tal es el caso del descenso del gradiente que se puede encontrar en la subsección 3.4.1.1.

2.2. Regresión logística

La **regresión logística** consiste en usar la distribución Bernoulli con el enlace canónico en un modelo lineal generalizado. De la tabla 2.4 $b(\theta) = log(1+exp(\theta))$ entonces $b'(\theta) = \frac{exp(\theta)}{1+exp(\theta)}$ y $(b')^{-1}(\mu) = log(\frac{\mu}{1+\mu})$.

1. $Y_i \sim Ber(\mu_i)$ independientes
2. El enlace canónico es

$$g(\mu(x_i)) = \log\left(\frac{\mu(x_i)}{1-\mu(x_i)}\right) = x_i^T\beta \quad i = 1, ..., N \tag{2.16}$$

Al enlace canónico también se le conoce como la función **logit** en este caso.

Observación 2.10. En este modelo no hay homoscedasticidad (varianza constante) ya que cada Y_i tiene media $\mu(x_i)$ y varianza $\mu(x_i)(1-\mu(x_i))$. El enlace canónico mapea la media de la distribución que toma valores en $(0,1)$ a todos los reales. $g : (0,1) \longrightarrow \mathbb{R}$. El uso de la función logit como la función de enlace no es la única opción, en general cualquier función del $(0,1)$ a $\mathbb{R}$ invertible puede ser usada. Por ejemplo, el inverso de la función de distribución de cualquier variable aleatoria continua en $\mathbb{R}$ es candidato de función de enlace. Pero no tendrían las ventajas estadísticas y computacionales que tiene usar la liga canónica [4].

Algunas otras alternativas populares a la regresión logística son el uso de la inversa de la distribución normal estándar i.e. $g(\mu) = \Phi^{-1}(\mu)$ (Regresión Probit) y el uso de la función enlace $g(\mu) = log(-log(1-\mu))$ (Regresión log-log complementaria). El artículo [3] expone de manera muy intuitiva que función de enlace escoger, por ahora diremos que la función de enlace que surge de manera natural y que simplifica los cálculos computacionales es el enlace canónico, en consecuencia es la que se usa para el ajuste realizado en este trabajo.

En la figura 2.1 se muestran las tres funciones: logit, probit y cloglog en el intervalo $(0, 1)$, se puede ver que son bastante similares.

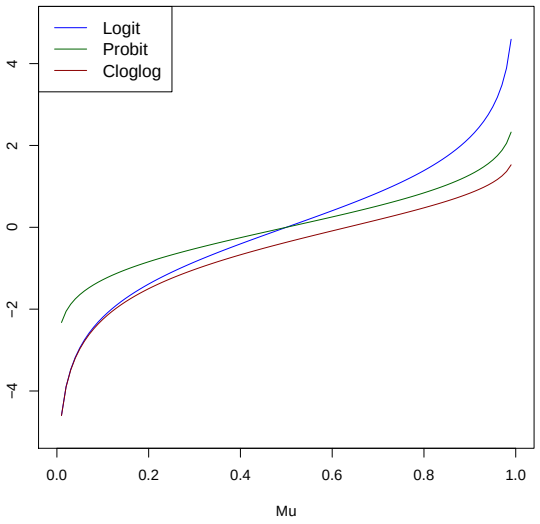


FIGURA 2.1: Logit, Probit y Cloglog

2.2.1. Selección de variables

La verosimilitud correspondiente a la regresión logística es

$$L(\beta) = \prod_{i=1}^{n} \mu(x_i)^{y_i}(1 - \mu(x_i))^{(1-y_i)} \tag{2.17}$$

con $\mu_i = g^{-1}(x_i) = (1 + exp(-x_i^T\beta))^{-1}$.
Si las predicciones $\hat{y}_i = \varphi(x_i)$ son perfectas, es decir coinciden siempre con los valores reales y_i diremos que el modelo está **saturado** y al hacer los cálculos la verosimilitud obtenida $L(\beta) = 1$.

Observación 2.11. Esta versosimilitud se obtiene si la respuesta sigue una distribución Bernoulli, es decir Y toma solo dos valores$\{0, 1\}$, con probabilidad $P(Y_i = 1|X_i = x_i) = \mu(x_i)$ y por lo tanto $P(Y_i = 0|X_i = x_i) = 1 - \mu(x_i)$. En este trabajo solamente se usa este caso, pero se puede generalizar a que Y_i siga una distribución binomial, por ejemplo

si en lugar de clasificar en dos clases se quisiera clasificar en más clases, o si se sumaran varias variables Bernoulli en un problema de conteo.

Definición 2.12. Devianza: Sea L la verosimilitud del modelo y L^* la verosimilitud del modelo saturado. Una medida del ajuste del modelo es compararlo contra el modelo saturado en un cociente.

$$\begin{aligned} D &= -2log(\frac{L(\beta)}{L^*(\beta)}) \\ &= -2log(L(\beta)/1) \\ &= -2log(L(\beta)) \end{aligned}$$

Si el modelo ajustado se aproxima al saturado entonces $D \to 0$, en cambio si el modelo ajustado está por debajo del valor del modelo saturado D se hace cada vez mayor, así que buscamos encontrar valores de D lo más pequeños posibles.

En general, esta misma técnica se usa para comparar modelos anidados, y probar si un modelo es distinto de otro. Recordemos que cada x_i es un vector de $p = k + 1$ entradas, $[x_{i,0}, x_{i,1}.., x_{i,k}]$, donde usualmente la primer entrada es constante. Por ejemplo, si $x_{i,0}$ es constante, la prueba de hipótesis para probar que alguno de los parámetros ajustados, diferentes al intercepto, es distinto de cero es: $H_0 : \beta_1 = ... = \beta_k = 0$ vs $H_1 : \beta_i \neq 0$ si $i = 1, .., k$ se puede probar con la estadística

$$G = -2log(\frac{L(\beta)}{L^0(\beta)}) \tag{2.18}$$

donde L es la densidad conjunta del modelo con p regresores y L^0 es la densidad conjunta del modelo que contiene solamente el intercepto β_0.
Bajo H_0, G se distribuye chi-cuadrada de $k + 1 - 1 = k$ grados de libertad.

$$G \overset{\alpha}{\sim} \chi^2_{(k)} \tag{2.19}$$

Observación 2.13. La validez de la prueba antes mencionada requiere de supuestos rígidos, el libro [8], capítulo 2, advierte sobre los supuestos que deben sostenerse, para que las distribución asintótica de la desviación sea cierta. En caso de pruebas de hipótesis formales se usan comúnmente los cocientes de verosimilitud, por encima de la prueba de Wald por ejemplo. También se usa la desviación como una medida de comparación de modelos, ya que a menor desviación es preferible el modelo. El coeficiente de Akaike es la desviación penalizada por el número de parámetros, así buscando evitar sobre-ajuste, este es: $AIC = 2p + D$, esto se puede ver en la definición 2.16.

Definición 2.14. Información de Fisher Si l_N es la log-verosimilitud para una muestra aleatoria de tamaño N y H_{l_N} es la matriz Hessiana $H_{i,j} = \frac{\partial^2}{\partial\theta_i\theta_j} l_N(\theta)$ entonces la información de Fisher es

$$I(\theta) = \mathbb{E}[-H_{l_N}] \tag{2.20}$$

Proposición 2.15. *Dada una regresión logística si derivamos dos veces $l_N(\theta)$ con respecto a β obtenemos*

$$\begin{aligned} I(\beta) &= \mathbb{E}[-H_{l_N}(\beta)] && (2.21)\\ &= \mathbb{E}[\mathbf{X}^T\mathbf{V}\mathbf{X}] && (2.22)\\ &= \mathbf{X}^T\mathbf{V}\mathbf{X} && (2.23) \end{aligned}$$

donde H_{l_N} es la matriz Hessiana, $\mathbf{X} = [x_1, ..., x_N]^T$ y $\mathbf{V}$ es la matriz dada por

$$\mathbf{V} = \begin{bmatrix} \mu_1(1-\mu_1) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \mu_n(1-\mu_n) \end{bmatrix}$$

Demostración. Como la distribución pertenece a la familia exponencial canónica como podemos ver en 2.4 la log-verosimilitud es

$$\begin{aligned} l_N(\beta) &= \sum_{i=1}^{N} \left(y_i\theta_i - log(1+e^{\theta_i})\right) && (2.24)\\ &= \sum_{i=1}^{N} \left(y_i x_i^T\beta - log(1+e^{X_i^T\beta})\right) && (2.25) \end{aligned}$$

Entonces el gradiente es

$$\nabla l_N(\beta) = \sum_{i=1}^{N} \left(y_i x_i - \frac{e^{x_i^T\beta}}{1+e^{x_i^T\beta}} x_i\right) \tag{2.26}$$

Y finalmente usando el hecho de que

$$\begin{aligned} \frac{\partial\nabla l_N(\beta)}{\partial\beta_j} &= \sum_{i=1}^{N} \frac{(1+e^{x_i^T\beta})e^{x_i^T\beta}x_{i,j} - e^{x_i^T\beta}e^{x_i^T\beta}x_{i,j}}{(1+e^{x_i^T\beta})^2} x_i && (2.27)\\ &= \sum_{i=1}^{N} \frac{e^{x_i^T\beta}x_{i,j}}{1+e^{x_i^T\beta}} x_i && (2.28) \end{aligned}$$

La matriz Hessiana queda

$$H_{l_N}(\beta) = -\sum_{i=1}^{N} \frac{e^{x_i^T\beta}}{(1+e^{x_i^T\beta})^2} x_i x_i^T \tag{2.29}$$

$$= -\mathbf{X}^T\mathbf{V}\mathbf{X} \tag{2.30}$$

ya que $\mu_i(1+\mu_i) = \frac{e^{x_i^T\beta}}{(1+e^{x_i^T\beta})^2}$
Observemos que en H_{l_N} no hay componente aleatorio y_i, entonces $\mathbb{E}[H_{l_N}] = H_{l_N}$. O sea se puede usar la matriz observada o la esperada. □

Una propiedad interesante de los estimadores máximo verosímiles es que bajo ciertas condiciones se distribuyen asintóticamente normal, esto es

$$\hat{\beta} \overset{\alpha}{\sim} \mathcal{N}(\beta, I(\beta)^{-1}) \tag{2.31}$$

este resultado no será probado en este trabajo pero una prueba puede ser revisada en la sección 10.5 de [13]. Por este resultado la matriz de varianzas y convarianzas del estimador máximos verosímiles $\hat{\beta}$ puede ser estimado por $I(\hat{\beta})^{-1}$

$$Var(\hat{\beta}) = I(\hat{\beta})^{-1} \tag{2.32}$$

este resultado es de particular interés para la construcción de una prueba de significancia para cada estimador $\hat{\beta}$.

La **Prueba de Wald** *contrasta las hipótesis $H_0 : \beta_i = 0$ contra $H_1 : \beta_i \neq 0$ con el estadístico*

$$W_j = \frac{\hat{\beta}_j}{sd(\hat{\beta}_j)} \quad j = 0, 1, ..., p \tag{2.33}$$

La desviación estándar se obtiene usando la ecuación 2.32 y el estadístico de Wald sigue una distribución $W_j \sim \mathcal{N}(0,1)$ bajo la hipótesis nula

Definición 2.16. Criterio de información de Akaike (AIC) es una herramienta para la selección de modelos que penaliza la log-verosimilitud l_N por el número de parámetros involucrados en el modelo. Si $\hat{\beta}$ es el estimador máximo verosímil de los parámetros β entonces el **AIC** es:

$$AIC = 2p - 2l(\beta) \tag{2.34}$$

Un método iterativo para seleccionar modelos con AIC es el **método stepwise**. El algoritmo 2.2, muestra los pasos a seguir.

Observación 2.17. Una desventaja con el método Stepwise es que se basa en puntajes para la selección del modelo, no con pruebas formales. También existen los algoritmos forward y backward, forward empieza sin variables y la va introduciendo aquella que minimiza el AIC, hasta que no haya una que mejore. Backward hace lo contrario, empieza con todas las variables y quita de una en una, hasta no poder mejorar el AIC. Stepwise puede empezar también con cero variables e ir introduciendo y quitando, y la solución no necesariamente es la misma. También, notemos que stepwise no recorre todas las combinaciones posibles de variables, y no garantiza encontrar el mínimo global del AIC, probar todas las combinaciones puede ser costoso computacionalmente pues requiere probar las 2^p combinaciones. También se pueden usar otros coeficientes además del AIC, como el BIC el cuál penaliza más fuertemente la complejidad del modelo, este es $BIC = p\log(N) - 2l(\beta)$

Stepwise

El **algoritmo stepwise** sigue los siguientes pasos

1. Se empieza con el modelo original de p variables
2. para $i \in \{1, ..., p\}$ ajustamos un nuevo modelo quitando (cuando es posible) la i-ésima variable y calculamos el AIC.
3. para $i \in \{1, ..., p\}$ ajustamos un nuevo modelo agregando (cuando es posible) la i-ésima variable y calculamos el AIC.
4. Actualizar el modelo seleccionando aquel modelo de *2* y *3* que tenga el AIC mas bajo
5. Repetir los pasos *2, 3, 4* hasta que el AIC no se pueda reducir mas.

FIGURA 2.2: Algoritmo Stepwise

Capítulo 3

Métodos basados en árboles

Resumen

Los árboles de decisión son métodos no-paramétricos, intuitivos y de rápido ajuste. Por si solo, un árbol de decisión es susceptible a sobre-ajustar, pero si se combinan combinar secuencias de árboles llamados ensambles, se reduce reducen la varianza. Algunos ensambles además de reducir la varianza, reducen el sesgo del modelo tomando en consideración la minimización de la función de pérdida. Los árboles son modelos no lineales por lo que se pierde la interpretación sencilla del efecto de las variables sobre la respuesta, es por esto que surgen medidas de importancia de las variables explicativas sobre la respuesta.

3.1. Árboles de regresión y de clasificación

Los métodos de ensamble que se presentan en este capítulo usan tanto árboles de clasificación como árboles de regresión, es por eso que se presenta la teoría de ambos tipos.

3.1.1. Árboles de decisión

Dado un conjunto de entrenamiento $\mathcal{L}$, un árbol es un modelo que parte el conjunto en T regiones disjuntas $\{R_t\}_{t=1}^{T}$ tal que la predicción $\varphi(x) = \hat{y}_t \mathbb{1}(x \in R_t)$ es la constante $\hat{y}_t$ en cada partición.
El error general del modelo se puede descomponer en las regiones disjuntas R_t, en el apéndice B.2 se encuentra la prueba.

$$Err\varphi_{\mathcal{L}} = \mathbb{E}_{X,Y}[L(\varphi(X), Y)] \tag{3.1}$$

$$= \sum_t \mathbb{P}(X \in R_t)\mathbb{E}_{X,Y|t}[L(\hat{y}_t, Y)] \tag{3.2}$$

Entonces minimizar el error general corresponde a minimizar el error en cada una de las regiones R_t, esto se hace aproximando el modelo de Bayes. Sobre cada región terminal el estimador $\hat{y}_t$ es la clase más frecuente en la región (clasificación) o el promedio de las variables respuesta en la región (regresión).
La partición del espacio de variables que genera las regiones R_t, se hace un proceso tal que dado el espacio muestral completo, genera cortes sucesivos en las variables, estos cortes se eligen de acuerdo a las medidas de impureza.

Definición 3.1. La medida de impureza sobre la región R_t es la función i tal que:
Para regresión: si N_t es el número de observaciones en R_t

- $i(t) = \sum_{x_i \in R_t} (y_i - \hat{y}_t)^2$ con $\hat{y}_t = \sum_{x_i \in R_t} y_i / N_t$

Para clasificación: si se tienen K clases y la probabilidad estimada de pertenecer a la clase k en la región R_t es $\hat{p}_{tk} = \frac{1}{N_t} \sum_{i: x_i \in R_t} \mathbb{1}(y_i = k)$ y $k(t) = \arg\max_k \hat{p}_{tk}$, las tres medidas más comunes de impureza son:

1. Error de Clasificación: $i(t) = 1 - \hat{p}_{tk(t)}$
2. Gini: $i(t) = \sum_{k \neq l} \hat{p}_{tk} \hat{p}_{tl}$
3. Entropía cruzada: $i(t) = -\sum_{k=1}^{K} \hat{p}_{tk} log(\hat{p}_{tk})$

Se puede ver en la figura 3.1 las tres medidas de impureza para un problema de clasificación binario. Notemos que alcanzan su máximo dentro el intervalo $(0, 1)$ en $p = 1/2$ y son simétricas respecto a este punto. Esto implica que si queremos reducir la impureza, debemos encontrar valores de p cercanos a 0 o a 1, y en consecuencia, la proporción de la clase cambie.

Definición 3.2. Dado que una región R_t tiene medida de impureza $i(t)$ podemos partir R_t en dos nuevas regiones disjuntas R_{t_L} y R_{t_R}. El **decrecimiento en impureza** es la reducción en la medida de impureza ponderado por la proporción de observaciones en cada nueva región respectiva

$$\Delta i(t) = i(t) - \frac{N_{t_L}}{N_t} i(t_L) - \frac{N_{t_R}}{N_t} i(t_R) \tag{3.3}$$

donde N_{t_L} es el número de observaciones en el nodo hijo t_L y N_{t_R} es el número de observaciones en el nodo hijo t_R.

Definición 3.3. Un **árbol** es una gráfica $G = (V, A)$ (una colección de vértices y aristas), en la que cualesquiera dos vértices (nodos) se conectan por un único camino.

Definición 3.4. Un **árbol con raíz** es una gráfica dirigida, donde todos los caminos parten de un único nodo, llamado **raíz**.

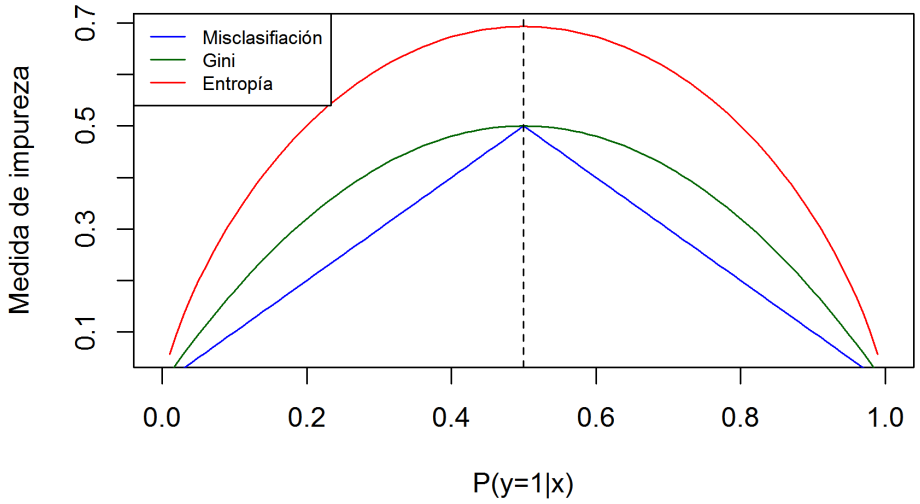


FIGURA 3.1: Medidas de impureza en clasificación

Definición 3.5. Si existe una arco (arista con dirección) del nodo t_1 al nodo t_2 entonces decimos que t_1 es **padre** del nodo t_2, o que t_2 es **hijo** de t_1.

Definición 3.6. En un árbol con raíz un nodo es llamado **interno** si tiene al menos un hijo, y es llamado **terminal** si no tiene ningún hijo.

Definición 3.7. Un **árbol binario** es un árbol con raíz donde todos los nodos internos tienen exactamente dos hijos.

Para los modelos que se presentan en este trabajo se usan árboles binarios para problemas de regresión y problemas de clasificación de dos clases o de más clases. En un árbol binario cada nodo t está asociado a una región en el espacio de variables. El nodo raíz t_0 corresponde al espacio entero de variables. Los nodos internos se etiquetan con un corte s_t que divide el espacio de variables respecto a una única variable en dos nuevas regiones disjuntas, cada una de estas nuevas regiones esta asociada a un nodo hijo. El algoritmo para el ajuste de un árbol de decisión binario se encuentra en la figura 3.2.

Observación 3.8. Los árboles son robustos ante valores extremos. El pronóstico generado por el árbol es el promedio de las observaciones en el nodo hoja o la clase más común en la hoja, así que no se sesga por observaciones explicativas de gran magnitud. Además los cortes hechos sobre las variables no dependen de la magnitud de las variables, solo del ordenamiento de ellas si la variable es continua o de las clases si la variable es categórica.

FIGURA 3.2: Algoritmo ajuste árbol de decisión

Árbol de decisión

El algoritmo para el ajuste de un árbol binario, basado en [7] cap. 3, es el siguiente:

1. Inicia el árbol con un nodo raíz t_0
2. Si se cumple un criterio de frenado global o particular a cada nodo, entonces en cada nodo terminal $t \in T$ la predicción es $\varphi(x) = \hat{y}_t \mathbb{1}(x \in R_t)$, en otro caso ir a 3.
3. Para cada nodo terminal $t \in T$ que no cumpla el criterio de frenado se encuentra el punto de corte s_j^* sobre cada variable x_j que maximiza el decrecimiento en impureza en t.
$$s_j^* = \arg\max_{s \in \mathbb{R}} \Delta i(s, t)$$
Se selecciona el corte s_j^* sobre la variable que maximice el decrecimiento en impureza.
4. Con la nueva partición $\{x_j \leq s_j^*\}$ y $\{x_j > s_j^*\}$ genera los nodos hijos izquierdo y derecho t_L y t_R y se agregan al árbol.

Observación 3.9. La convergencia del algoritmo está acotada por el tamaño de la base de entrenamiento, por el crecimiento en nodos del árbol y por el número de variables. Para más detalle consultar el artículo en [5].

3.1.2. Ajuste de un árbol

Los tres **criterios de frenado** más comunes son: la **profundidad** del árbol, el número mínimo de observaciones en cada nodo terminal y el **parámetro de complejidad**.

Definición 3.10. La **profundidad del árbol** es el número de pasos máximos del nodo raíz a cualquiera de los nodos terminales.

Definición 3.11. El **parámetro de complejidad**, basado en [2] cap. 9, es el valor α que regula al costo de complejidad. Si T es el conjunto nodos terminales y $\alpha \geq 0$ el costo de complejidad es:
$$C_\alpha = \sum_{t \in T} N_t i(t) + \alpha |T|$$

Seleccionaremos el árbol con menor costo de complejidad para cierto valor α. El valor óptimo del hiper-parámetro α se puede encontrar por validación o validación cruzada $\alpha^* = \arg\min_\alpha \widehat{Err}^{CV}(\varphi)$. Dado que al variar α de 0 a ∞ se obtiene una única secuencia de árboles contenidos uno en otro, cada uno de menos nodos, a este proceso suele llamarse **podar** un árbol. Cuando se ajusta un árbol de decisión usualmente se deja fijo un valor

del número mínimo de observaciones en cada nodo terminal, y se elige α por validación cruzada, α a su vez regula el número de niveles del árbol por lo que no es necesario regular este parámetro. Por último notemos que contrario a la intuición si el parámetro de complejidad es pequeño entonces el árbol ajustado es de gran tamaño y si el parámetro de complejidad es grande entonces el árbol resultante es de pocos niveles.

3.1.2.1. Tratamiento de variables con valores nulos

Los árboles a diferencia de otros métodos de aprendizaje supervisado permiten incorporar en el modelo datos con valores faltantes. La técnica para tratar con observaciones nulas es: en cada nodo del árbol se obtiene la regla de decisión de la variable que realiza el mayor decrecimiento de impureza pero también se almacena la segunda variable que mejor decrece la impureza, la tercera variable que mejor decrece la impureza y así sucesivamente. Entonces cuando el algoritmo genera predicciones sobre muestras de datos con una observación faltante la regla de decisión al nodo hijo se realiza con la primer variable que no tenga un valor nulo de acuerdo al decrecimiento en impureza. A las variables que no son la partición principal en el nodo se les conoce como **variables sustitutas**. En el capítulo 4 de Aplicación se presenta la base de datos a tratar que tiene variables con observaciones nulas.

3.1.3. Árboles con pesos modificados

Algunas veces se da mayor ponderación a observaciones que son de mayor interés para la clasificación o la regresión. Es por ello que se pueden ajustar árboles de decisión con observaciones ponderadas, en particular para el algoritmo de ensamble boosting se usan pesos modificados secuenciales. Dada una serie de pesos positivos $\{w_i\}_{i=1}^{N}$ asignados a cada observación en el conjunto de entrenamiento, el algoritmo 3.2 se modifica ligeramente, cada una de las probabilidades estimadas en la región R_t se convierte en

$$\hat{p}_{t,k} = \frac{\sum_{x_i \in Rm} w_i \mathbb{1}(y_i = k)}{\sum_{x_i \in R_t} w_i} \tag{3.4}$$

como se hizo anteriormente la clase más probable es $k(t) = \arg\max_k \hat{p}_{tk}$. El resto del algoritmo de ajuste de un árbol de decisión se mantiene igual.
Los pesos son relativos a la importancia que dará el algoritmo a clasificar las observaciones correctamente. Entre más alto sea el peso, el algoritmo sesgará su ajuste a tener estas observaciones estimadas correctamente, y lo inverso si el peso es bajo.

3.1.4. Importancia de las variables

Dada la naturaleza no lineal de los árboles de decisión, es de interés tener una medida de que variables son las que tienen mayor impacto en las estimaciones que genera el modelo. Una manera, es a través de la importancia de las variables. La importancia de una variable l se define como el acumulado en la reducción en la medida de impureza, sobre los nodos internos del árbol. Esto es, dado un solo árbol φ con T nodos terminales, la importancia de la $l-$ésima variable denotada $\mathcal{I}_l(\varphi)$ es:

$$\mathcal{I}_l(\varphi) = \sum_{t=1}^{T-1} \Delta i(t) \mathbb{1}(l = v(t)) \tag{3.5}$$

Consideraciones sobre la ecuación 3.5

- Si un árbol tiene T nodos terminales tuvieron que haber $T-1$ particiones anteriores en el espacio de las variables para generar los nodos terminales. Cada partición tiene asociada un nodo interno por lo que existen $T-1$ nodos internos. Por esta razón la ecuación 3.5 considera la suma sobre los $T-1$ nodos internos.
- $\Delta i(t)$ es el decrecimiento en la medida de impureza
- $v(t)$ es la variable que genera la mayor medida de impureza sobre el nodo t, es decir la variable con la cuál se realiza el corte.

Notemos que si una variable no entra en ninguno de los cortes entonces tendrá importancia de cero.

3.2. Bagging y bosques aleatorios

Definición 3.12. Dada un conjunto de observaciones $\mathcal{L}$, una **muestra bootstrap** del conjunto es una muestra aleatoria con reemplazo del mismo número de observaciones que el conjunto original.

3.2.1. Bagging

Bagging o **bootsrap agregado** genera una secuencia de M árboles $\{\varphi_m\}_{m=1}^{M}$ cada uno estimado sobre una muestra bootstrap. Menciona Brieman en su artículo: "Cada árbol del ensamble se crece a su máxima extensión sin podarlo."[1] Las predicciones en regresión y clasificación son:

- Regresión:
  $\varphi(x) = \frac{1}{M}\sum_{m=1}^{M}\varphi_m(x)$ donde φ_m es la predicción de cada árbol del ensamble

- Clasificación:
  $\varphi(x) = C_k$ donde C_k es la clase más frecuente

En el caso de clasificación si se quiere estimar la probabilidad de alguna clase, se estima con la proporción de "votos" del comité de M árboles que eligen dicha clase. Otra alternativa es tomar el promedio de las probabilidades estimadas de cada clase en cada árbol. Mencionan Hastie et al. en su libro: "En general se ha encontrado que esta segunda estrategia genera mejores estimaciones que la primer estrategia y con menor varianza." [2, p. 283].

3.2.2. Bosques aleatorios

En bagging, los árboles generados en cada muestra bootstrap incluyen las mismas variables por lo que hacen cortes similares. Esto hace que los árboles generen predicciones parecidas. Quisiéramos que en cada iteración del algoritmo el árbol generado aportara predicciones distintas a los demás árboles. Los **bosques aleatorios** al igual que en bagging, generan M árboles sobre muestras bootstrap, pero a diferencia de bagging en cada nodo de cada árbol ajustado se seleccionan aleatoriamente $b \leq p$ variables ($b = p$ es bagging) para reducir la correlación entre las predicciones de los árboles generados, ya que al tomar variables distintas en cada nodo de cada árbol los cortes van a ser distintos y así tendremos árboles con predicciones menos correlacionadas que en bagging.

Bagging obtiene el promedio de una secuencia de modelos de alta varianza (árboles profundos) con el objetivo de reducir la varianza. Notemos que los árboles generados son idénticamente distribuidos (pues provienen de la misma muestra de entrenamiento) por lo que la esperanza de una predicción $\mathbb{E}_{XY}[\varphi(x)]$ es la misma que la esperanza del promedio $\mathbb{E}_{XY}[\frac{1}{M}\sum_{m=1}^{M}\varphi_m(x)]$, así que la única mejora posible es reducir la varianza. Si la varianza individual de las predicciones de cada árbol es σ^2, si los árboles fueran i.i.d. entonces la varianza sería $\frac{\sigma^2}{M}$. Pero en realidad los árboles son idénticamente distribuidos pero no independientes.

Si la correlación entre cualesquiera dos árboles es $\rho > 0$ (una prueba más detallada se encuentra en [7]). Entonces la varianza de la predicción es

$$Var(\frac{1}{M}\sum_{m=1}^{M}\varphi_m(x)) = \frac{1}{M^2}(M(M-1)\rho\sigma^2 + M\sigma^2) \tag{3.6}$$

$$= \rho\sigma^2 + \frac{1-\rho}{M}\sigma^2 \tag{3.7}$$

Para valores de M grandes el primer término se mantiene constante y el segundo término tiende a cero, esto quiere decir que la correlación entre árboles limita la reducción de

varianza que logra el promedio de árboles obtenidos en muestras aleatorias con remplazo. Los bosques aleatorios reducen la correlación entre árboles seleccionando aleatoriamente b de las p variables, por lo que se tienen una mejora en la reducción de varianza.

FIGURA 3.3: Algoritmo ajuste bosque aleatorio

Bosques Aleatorios

El algoritmo para ajuste de un bosque aleatorio es:
Para m=1 a M

1. Obtener una muestra bootstrap de los datos de entrenamiento, del tamaño de la base de entrenamiento. Se inicia el $m-$ésimo árbol con un nodo raíz.
2. Para cada nodo terminal del árbol φ_m
   - *a*) Seleccionamos aleatoriamente $1 \leq b \leq p$ variables
   - *b*) Si se cumple algún criterio de frenado global o particular a cada nodo terminal, obtener las predicciones del $m-$ésimo árbol $\varphi_m(x)$.
   - *c*) Si el nodo terminal no cumple el criterio de frenado se hace el corte que maximiza el decrecimiento en impureza, usando solamente las b variables seleccionadas.
   - *d*) Con la nueva partición se generan los nuevos nodos hijos izquierdo y derecho que se agregan al árbol φ_m y se mueve al siguiente nodo terminal.

Las predicciones en regresión y clasificación son respectivamente:

- $\varphi(x) = \frac{1}{M}\sum_{m=1}^{M}\varphi_m(x)$
- $\varphi(x) = C_k$: donde C_k es la clase mas frecuente

La elección del hiper-parámetro b, el número de variables seleccionadas aleatoriamente, se puede encontrar por validación o por validación cruzada, aunque los bosques aleatorios tienen un método particular para hacer validación que es más conveniente a validación cruzada de $K-$capas, ya que elimina la necesidad de estimar K modelos. Este tipo de validación se conoce como validación fuera de la bolsa.

Definición 3.13. Si un ensamble se construye sobre una secuencia de modelos estimados sobre una muestra bootstrap una estimación del error general es el **error fuera de la bolsa**. En el caso de un bosque aleatorio si M^{-i} es el numero de árboles que en su muestra *bootstrap* no incluyen a la observación (x_i, y_i) entonces el **error fuera de la**

bolsa es

$$\widehat{Err}^{FB}(\varphi_B) = \frac{1}{N}\sum_{(x_i,y_i)\in\mathcal{L}} L\left(\frac{1}{M^{-i}}\sum_{m=1}^{M^{-i}}\varphi_m(x), y_i\right)$$

Entonces la selección del hiper-parámetro b es aquel que minimice el error fuera de la bolsa $b^* = \arg\min_b \widehat{Err}^{FB}(\varphi_B)$.

3.2.3. Importancia de las variables

Para un bosque aleatorio φ la importancia de la $l-$ésima variable es el promedio de la importancia de la variable sobre los elementos del ensamble $\{\varphi_m\}_{m=1}^M$.

$$\mathcal{I}_l(\varphi) = \frac{1}{M}\sum_{m=1}^{M}\mathcal{I}_l(\varphi_m) \tag{3.8}$$

3.3. Boosting adaptativo

Boosting es una técnica de ensamble al igual que bosques aleatorios. Aunque boosting nace primero para problemas de clasificación, se puede extender a problemas de regresión. En esta sección vamos a ver el algoritmo adaboost uno de los algoritmos más populares usado para problemas de clasificación binarios. La idea principal de boosting es combinar muchos clasificadores "débiles" para formar un "comité" poderoso. Se entiende que un clasificador es débil, si la tasa de error de clasificación supera por un margen pequeño a la tasa de error de una clasificación aleatoria. A diferencia de bosques aleatorios en boosting cada clasificador subsecuente aprende a clasificar correctamente las observaciones en las cuales los clasificadores anteriores fallaron mediante un proceso progresivo.

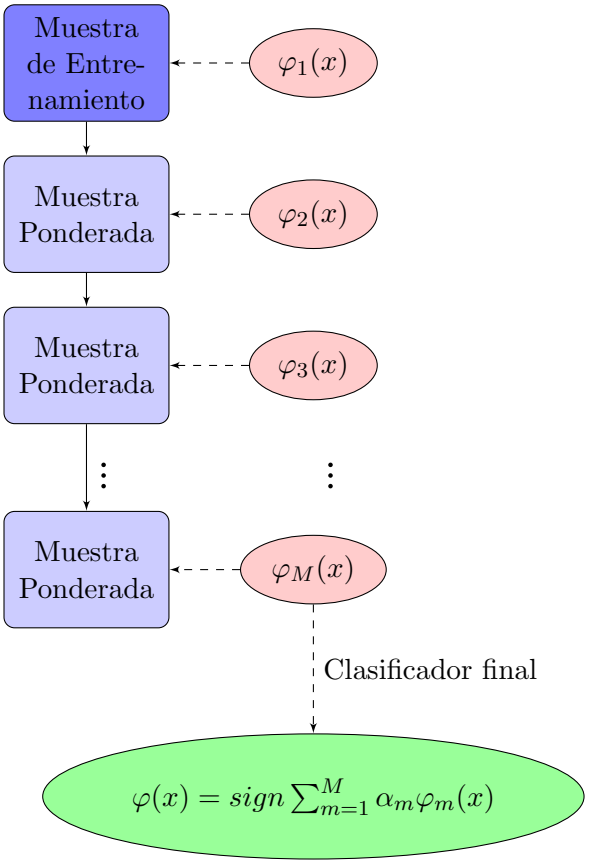
Por simplicidad de esta sección supongamos que la variable respuesta toma valores en $\{-1, 1\}$. Si φ es un clasificador binario $\varphi(x) \in \{-1, 1\}$ entonces boosting busca ajustar clasificadores débiles en secuencia, en cada paso de la secuencia se da mayor peso a las observaciones que fueron clasificadas incorrectamente en los pasos anteriores. Obteniendo los clasificadores $\varphi_1, ..., \varphi_M$ de los cuales se obtiene el clasificador final, si $sign(x)$ es la función signo esto es:

$$\varphi(x) = sign\sum_{m=1}^{M}\alpha_m\varphi_m(x) \tag{3.9}$$

y la función signo,

$$sign(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases} \tag{3.10}$$

FIGURA 3.4: Diagrama de adaboost



La figura 3.4 muestra gráficamente el proceso. Los pesos $\alpha_1, .., \alpha_M$ se obtienen de tal manera que dan mayor ponderación a clasificadores más exactos. Y la muestra de entrenamiento se va modificando de la siguiente manera: En el primer paso a cada observación x_i de la muestra se le asigna un peso equitativo de $w_i = 1/N$ y se ajusta un primer clasificador $\varphi_1(x)$.

En cada m-ésima iteración posterior se modifican los pesos w_i de tal forma que las observaciones clasificadas incorrectamente en la iteración anterior $\varphi_{m-1}(x)$ reciben un mayor peso. Así este método reduce el sesgo del problema, contrario a bosques aleatorios que solo reducían la varianza.

FIGURA 3.5: Algoritmo adaboost

Adaboost

El algoritmo es el siguiente:

1. Iniciamos $w_i = 1/N$ para $i = 1, ..., N$
2. Para m=1 a M
   - *a*) Ajustamos el clasificador φ_m a las observaciones con pesos w_i
   - *b*) Obtenemos el error del clasificador

$$\overline{err}_m = \frac{\sum_{i=1}^{N} w_i \mathbb{1}(y_i = \varphi_m(x_i))}{\sum_{i=1}^{N} w_i}$$

   - *c*) Obtenemos el peso del clasificador

$$\alpha_m = log(\frac{1 - \overline{err}_m}{\overline{err}_m})$$

   - *d*) Actualizar los pesos de las observaciones $i = 1, .., N$

$$w_i = w_i exp\left(\alpha_m \mathbb{1}(y_i \neq \varphi_m(x_i))\right)$$

3. Finalmente el clasificador final es:

$$\varphi(x) = sign \sum_{m=1}^{M} \alpha_m \varphi_m(x)$$

3.3.1. Relación de boosting adaptativo y un modelo aditivo

Para entender de fondo el algoritmo 3.5, más allá de los pasos que sigue, se establece boosting adaptativo como una aproximación a la solución de un integrante de una familia de modelos llamados modelos aditivos progresivos. Viendo boosting desde esta perspectiva, este genera una secuencia de modelos que reducen recursivamente una función de pérdida exponencial, esta es $L(y, f(x)) = e^{-yf(x)}$. Esta función de pérdida facilita los cálculo y da una solución cerrada al problema de minimización, esto se explica más a detalle en la subsección 3.3.2, estás características generan los pesos w_i que se ven en el algoritmo 3.5. Suponer una función de pérdida de este tipo encuentra limitaciones, principalmente la pérdida exponencial la cuál puede ser no la más adecuada al problema, esto ve más a detalle en la subsección 3.3.2.1. En la subsección 3.4.1 extendemos los modelos aditivos progresivos con ciertas modificaciones, a cualquier función de pérdida diferenciable.

Vamos a establecer la relación entre adaboost y una familia de modelos conocidos

como los modelos aditivos progresivos. Los **modelos aditivos progresivos** aproximan una función $f(x)$ de interés por medio de una combinación lineal de funciones $\sum_{m=1}^{M} \beta_m f_m(x)$. Donde $f(x)$ minimiza una función de pérdida.

$$L(y, f(x)) \tag{3.11}$$

Esta serie de modelos tienen la característica que en cada m-ésima iteración los parámetros que aparecieron en las iteraciones pasadas se dejan fijos, y solo se busca optimizar los parámetros asociados a la iteración actual, esto se puede ver en el algoritmo 3.6.

FIGURA 3.6: Modelos aditivos progresivos

Modelos aditivos progresivos

El algoritmo es el siguiente:

1. Iniciamos $f_0(x) = 0$
2. Para m=1 a M
   a) Se estiman los parámetros (β_m, γ_m), que minimizan la pérdida
   $$(\beta_m, \gamma_m) = \arg\min_{\beta,\gamma} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \beta b(x_i; \gamma))$$
   b) Se actualiza la estimación
   $$f_m(x) = f_{m-1}(x) + \beta_m b(x; \gamma)$$
3. Finalmente la estimación es: $f_M(x) = \sum_{m=1}^{M} \beta_m b(x; \gamma_m)$

Vamos probar la equivalencia entre adaboost y un modelo aditivo progresivo donde la función de pérdida es exponencial, es decir:

$$L(y, f(x)) = \exp(-yf(x)) \tag{3.12}$$

Proposición 3.14. *Adaboost aproxima mediante un algoritmo de aprendizaje la solución de un modelo aditivo progresivo con función de pérdida exponencial.*

Demostración. Supongamos que nos encontramos en la *m-ésima* iteración del algoritmo

3.6, es decir, se estimaron los clasificadores $\varphi_1(x), ..., \varphi_{m-1}(x)$ en la base de entrenamiento y buscamos resolver

$$(\beta_m, \varphi_m) = \arg\min_{\beta,\varphi} \sum_{i=1}^{N} \exp(y_i(f_{m-1}(x_i) + \beta\varphi(x_i))) \tag{3.13}$$

$$= \arg\min_{\beta,\gamma} \sum_{i=1}^{N} w_i^{(m)} \exp(-\beta y_i \varphi(x_i)) \tag{3.14}$$

donde $w_i^{(m)} = exp(-y_i f_{m-1}(x_i))$.
La solución de la ecuación 3.14 se encuentra en dos pasos. Para cualquier $\beta > 0$ el valor óptimo de la ecuación 3.14 es aquel clasificador de parámetros γ que se equivoque menos en términos de la pérdida exponencial ponderada por los pesos $w_i^{(m)} > 0$. Al ser la pérdida exponencial una función estrictamente decreciente, el mínimo sobre los valores de entrenamiento se va a alcanzar en el clasificador tal que

$$\varphi_m(x) = \arg\min_{\varphi} \sum_{i=1}^{N} w_i^{(m)} \mathbb{1}(y_i \neq \varphi(x)) \tag{3.15}$$

Ya teniendo el clasificador φ_m podemos encontrar el valor óptimo de β. La ecuación 3.14 se puede formular como

$$\begin{aligned} & e^{-\beta} \sum_{y_i=\varphi(x_i)} w_i^{(m)} + e^{\beta} \sum_{y_i\neq\varphi(x_i)} w_i^{(m)} \\ & = e^{-\beta} W^+ + e^{\beta} W^- \end{aligned}$$

para $W^+ = \sum_{i=1}^{N} \mathbb{I}(y_i = \varphi(x_i)) w_i^{(m)}$ y $W^- = \sum_{i=1}^{N} \mathbb{1}(y_i \neq \varphi(x_i)) w_i^{(m)}$ Derivando esta ecuación respecto a β e igualando a cero

$$\begin{aligned} -W^+ e^{-\beta} + W^- e^{\beta} &= 0 \\ \frac{W^+}{W^-} &= \frac{e^{\beta}}{e^{-\beta}} \\ \frac{1}{2} \log \frac{W^+}{W^-} &= \beta \end{aligned}$$

Además la segunda derivada respecto a β es

$$W^+ e^{-\beta} + W^- e^{\beta}$$

que es estrictamente positiva, por lo que $\beta_m = \frac{1}{2}\log\frac{W^+}{W^-}$. Además

$$\begin{aligned}\frac{W^+}{W^-} &= \frac{\sum_{y_i=\varphi(x_i)} w_i^{(m)} / \sum_{i=1}^N w_i}{\sum_{y_i\neq\varphi(x_i)} w_i^{(m)} / \sum_{i=1}^N w_i} \\ &= \frac{1-\overline{err}_m}{\overline{err}_m}\end{aligned}$$

así

$$\beta_m = \frac{1}{2}\log\frac{1-\overline{err}_m}{\overline{err}_m} \tag{3.16}$$

Si encontramos el clasificador $\varphi_m(x)$ que reduzca el error ponderado con los pesos, podemos encontrar el peso β_m asociado a este clasificador y podemos actualizar la estimación de

$$f_m(x) = f_{m-1}(x) + \beta_m\varphi_m(x)$$

que es lo equivalente a actualizar los pesos

$$w_i^{(m+1)} = w_i^{(m)} e^{-\beta_m y_i \varphi_m(x_i)}$$

Para llegar a la actualización de pesos del algoritmo 3.5 usamos la identidad

$$y_i\varphi_m(x_i) = 2\mathbb{1}(y_i \neq \varphi_m(x_i)) - 1$$

substituyendo en la expresión anterior

$$w_i^{(m+1)} = w_i^{(m)} e^{\alpha_m \mathbb{1}(y_i\neq\varphi_m(x_i))} e^{-\beta_m}$$

donde $\alpha_m = 2\beta_m = \log\frac{1-\overline{err}_m}{\overline{err}_m}$ es la cantidad en 3.5. El factor $e^{-\beta_m}$ es una constante que multiplica a cada uno de los pesos, por esta razón se puede omitir sin causar ninguna discrepancia. Por lo tanto se actualizan los pesos de las observaciones

$$w_i^{(m+1)} = w_i^{(m)} e^{\alpha_m \mathbb{1}(y_i\neq\varphi_m(x_i))}$$

□

El desarrollo presentado, sigue en gran medida el artículo de Rojas [11], el cuál es muy intuitivo y amigable al lector.

En la práctica, bajo el modelo aditivo progresivo, deberíamos encontrar el clasificador $\varphi_m(x)$ que resuelva la ecuación 3.15, debido a la imposibilidad de encontrar el clasificador perfecto, este clasificador se estima con los hiper-parámetros que fija el modelo. Se eligen árboles de clasificación cada uno de ellos φ_m se estima sobre observaciones con pesos

modificados, esto se puede ver en la subsección 3.1.3. Técnicamente boosting no resuelve un modelo aditivo progresivo, sino que aproxima su solución mediante un clasificador con observaciones ponderadas.

3.3.2. Uso de la pérdida exponencial

Dos razones para la elección de la pérdida exponencial como función de pérdida son:

1. Las computaciones iterativas del estimador $f_m(x)$ tienen una forma sencilla y son intuitivas en el sentido de ir cambiando los pesos de las observaciones donde los clasificadores se han equivocado anteriormente.

2. Tomar el signo de $sign \sum_{m=1}^{M} \alpha_m \varphi_m(x)$ es adecuado para el clasificador final y si se requiere obtener la probabilidad de caer en la clase 1, se puede obtener con $p = \frac{1}{e^{2f(x)}-1}$, esto se puede ver en la proposición 3.15.

Proposición 3.15. *Sea Y una variable aleatoria en $\{-1, 1\}$ entonces si $p = \mathbb{P}(Y = 1)$ la pérdida esperada $\mathbb{E}_{Y|X=x}[e^{f(x)y}]$ se minimiza en $f^*(x) = 1/2 \log(p/(1-p))$.*

Demostración. Si p es $\mathbb{P}(Y = 1|X = x)$, entonces la esperanza de $e^{Yf(x)}$ es

$$\mathbb{E}_{Y|X=x}[e^{-f(x)Y}] = e^{-f(x)}p + e^{f(x)}(1-p)$$

Derivando esta expresión respecto a $f(x)$ e igualando a cero obtenemos

$$\begin{aligned} -e^{-f(x)}p + e^{f(x)}(1-p) &= 0 \\ e^{f(x)}(1-p) &= -e^{-f(x)}p \\ e^{2f(x)} &= \frac{p}{1-p} \\ f(x) &= \frac{1}{2}\log\frac{p}{1-p} \end{aligned}$$

También la segunda derivada respecto a $f(x)$ es $e^{-f(x)}p + e^{f(x)}(1-p)$ que es estrictamente mayor que cero para cualquier x y $p \in (0, 1)$. Así que:

$$f^*(x) = \frac{1}{2}\log\frac{p}{1-p} \tag{3.17}$$

es la función que reduce la esperanza de la pérdida exponencial. Si resolvemos para p obtenemos $p = \frac{1}{e^{2f(x)}-1}$.

□

Observación 3.16. Por otro lado, al ser boosting equivalente a un modelo aditivo progresivo con pérdida exponencial, su objetivo es reducir la pérdida aproximando $f(x)$ con $\sum_{m=1}^{M} \alpha_m \varphi_m(x)$. La función $f^*(x)$ es el enlace canónico, el cual es simétrico respecto a $p = 1/2$, positivo para $p > 1/2$ y negativo para $p < 1/2$, por lo que tomar el signo de $\sum_{m=1}^{M} \alpha_m \varphi_m(x)$ para dar el clasificador final es fijar un corte de $p = 1/2$. El estimador de la probabilidad de la clase 1 es $1/(e^{2\sum_{m=1}^{M} \alpha_m \varphi_m(x)} - 1)$. Estimando esta probabilidad podemos generar un corte que sea adecuado para las particularidades del problema, no estamos limitados a un corte de $1/2$. Si se requiere generar un mayor número de pronósticos de la clase de unos se fija un corte bajo, o si se requiere obtener menos pronósticos de la clase de interés se fija un corte alto.

3.3.2.1. Problemas de adaboost

El principal problema de boosting adaptativo es su restricción a una función de pérdida exponencial. La pérdida exponencial $L(y, f(x)) = \exp(-yf(x))$ tiene un margen de clasificación dado por $-yf(x)$, ya que, si se clasifica correctamente a una observación $-yf(x) < 0$ la pérdida es pequeña, pero si se clasifica incorrectamente una observación entonces $-yf(x) > 0$ la pérdida crece, pero crece a un ritmo exponencial, como se puede ver en la figura 3.7. Si en el algoritmo 3.4 existe una observación que se este clasificando incorrectamente varias veces su peso aumentará exponencialmente y el algoritmo puede perder desempeño por enfocarse en estas observaciones, especialmente si se trata de observaciones atípicas.

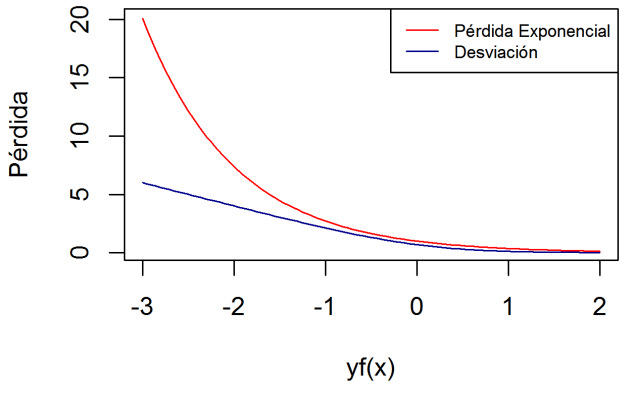


FIGURA 3.7: Funciones de Pérdida

La segunda función en la gráfica 3.7, es una función de pérdida que se usa comúnmente en problemas de clasificación, el negativo del logaritmo de la verosimilitud de una distribución Bernoulli, que es la función de pérdida en la regresión logística, solo que

ahora la respuesta está en $\{-1, 1\}$ así que la función de enlace cambia un poco y es $f(x) = \frac{1}{2}\log\frac{p}{1-p}$ que coincide con 3.17. La función de pérdida es:

$$-l(y, f(x)) = -\frac{y+1}{2}\log p - \frac{1-y}{2}\log(1-p)$$

Otra manera equivalente de escribir la menos log-versimilitud es:

$$-\,l(y, f(x)) = \log(1 + exp(-2yf(x))) \tag{3.18}$$

Esta equivalencia se da al evaluar las funciones en $y = 1$ y $y = -1$. Para $y = 1$ toman el valor $-log(p)$ y en $y = -1$ toman el valor $-log(1-p)$. Así que para cualquier $y \in \{-1, 1\}$ y $p \in (0, 1)$ las expresiones son equivalentes.

Si comparamos la pérdida exponencial con la log-verosimilitud, ambas asignan mayor peso cuando $yf(x) < 0$, es decir, clasificamos incorrectamente una observación. Pero la pérdida exponencial tiene un crecimiento exponencial para valores negativos y la log-verosimilitud tiene un comportamiento asintóticamente lineal en los mismos valores, derivado de esto, la pérdida exponencial da mayor importancia a observaciones clasificadas incorrectamente por un mayor margen que la log-verosimilitud, esto puede llevar a un sesgo a estas observaciones.

3.3.3. Importancia de las variables

En boosting la importancia cuadrada de la variable l es el promedio de las importancia cuadrada de la variable sobre los elementos del ensamble

$$\mathcal{I}_l^2(\varphi) = \frac{1}{M}\sum_{m=1}^{M}\mathcal{I}_l^2(\varphi_m) \tag{3.19}$$

La importancia de la variable l es la raíz cuadrada de la importancia cuadrada.

3.4. Boosting del gradiente

En esta sección se presenta un método de ensamble que usa cualquier función de pérdida, no solo la pérdida exponencial usada en adaboost. **Boosting del gradiente** ataca el problema de varianza con el promedio ponderado y el problema de sesgo reduciendo la pérdida iterativamente, además permite mayor flexibilidad que adaboost en la elección de la función de pérdida. Antes de presentar el algoritmo de boosting del gradiente, se presenta la técnica de optimización de una función de pérdida mediante descenso del

gradiente, porque el algoritmo incorpora descenso del gradiente en la generación del ensamble.

3.4.1. Optimización mediante descenso del gradiente

Dado un conjunto de aprendizaje $\mathcal{L}$ de tamaño N y una función de pérdida L, la función evaluada en φ sobre la muestra de entrenamiento es:

$$L(\varphi) = \sum_{i=1}^{N} L(y_i, \varphi(x_i)) \tag{3.20}$$

nuestro objetivo es minimizar $L(\varphi)$ con respecto a φ. Existen diversas técnicas de optimización que aproximan la solución en un proceso iterativo de sumas de componentes $\{h_m\}_{m=1}^{M}$.Dado un valor inicial $h_0 = \varphi_0$, para optimizar en cada $m = 1, ...M$, la función $\varphi_m = h_m + \varphi_{m-1}$ se deja fijo φ_{m-1} y solo se "mueve" h_m, obteniendo en la última iteración $\varphi_M = \sum_{m=1}^{M} h_m$. De estás técnicas, una de las más sencillas y populares es **descenso del gradiente** .

3.4.1.1. Descenso del gradiente

Esta técnica de optimización escoge $h_m = -\rho_m g_m$ donde ρ_m es un escalar y g_m es el gradiente de $L(\varphi)$ evaluado en $\varphi = \varphi_{m-1}$. Cada entrada del gradiente es

$$g_{im} = \left[\frac{\partial L(y_i, \varphi(x_i))}{\partial \varphi(x_i)}\right]_{\varphi(x_i)=\varphi_{m-1}(x_i)} \tag{3.21}$$

El *"tamaño del paso"* ρ_m es la solución a

$$\rho_m = \arg\min_{\rho}\{L(\varphi_{m-1} - \rho g_m)\} \tag{3.22}$$

por último se actualiza $\varphi_m = \varphi_{m-1} - \rho_m g_m$.

Observación 3.17. $-g_m$ es la dirección de máximo decrecimiento de $L(\varphi)$ en $\varphi = \varphi_{m-1}$, ya que el gradiente es la dirección de máximo crecimiento de la función.

3.4.2. Boosting del gradiente

Si nuestro objetivo fuera solamente minimizar $L(y, \varphi(x))$ en el conjunto de entrenamiento entonces, el descenso del gradiente sería una técnica que reduciría la función de pérdida para las observaciones de este conjunto. El problema es que el gradiente solo se

estima sobre las observaciones de entrenamiento y nuestra meta final es extender el modelo a observaciones no vistas. Una solución es introducir en cada iteración del descenso del gradiente un árbol de regresión, que sea lo más parecido al gradiente por mínimos cuadrados; los árboles de regresión generan pronósticos para observaciones fuera del conjunto de entrenamiento, entonces generalizan el descenso del gradiente a observaciones con las cuales no se realiza el descenso, teniendo así una solución aproximada. Esto se puede ver en el algoritmo 3.8.

Tipo de problema	Función de pérdida	$-\frac{\partial L(\varphi(x_i),y_i)}{\partial \varphi(x_i)}$
Regresión	$1/2(y_i - \varphi(x_i))^2$	$y_i - \varphi(x_i)$
Regresión	$\lvert y_i - \varphi(x_i)\rvert$	$sign\{y_i - \varphi(x_i)\}$
Clasificación	Menos	Para cada clase $k = 1, .., K$
	log-verosimilitud	$-\mathbb{1}(y_i = k) + p_k(x_i)$

TABLA 3.1: Gradientes para funciones comunes de pérdida

3.4.2.1. Boosting del gradiente en clasificación

En un problema de multi-clasificación de K clases, es común usar la función de pérdida dada por la log-verosimilitud de una distribución multinomial. A continuación extendemos la idea de la regresión logística en un problema muli-clase y obtenemos el gradiente de la pérdida para poder establecer el algoritmo de descenso del gradiente. La menos log-verosimilitud es proporcional a:

$$p_1^{x_1} \dots p_{K-1}^{x_{K-1}} (1 - \sum_{k=1}^{K-1} p_k)^{x_K} \tag{3.23}$$

con $x_j \in \{0, 1\}$ y $x_k = \mathbb{1}(y = k)$ y p_k depende de x, en realidad es $p_k(x) = \mathbb{P}(Y = k|x)$ pero para hacer la notación más sencilla, escribiremos solo p_k para $k = 1, ..., K$
Para extender la idea de la regresión logística a K clases, introducimos las funciones $f_1(x), f_2(x), ..., f_K(x)$ como

$$f_k(x) = \log \frac{p_k}{p_K} \tag{3.24}$$

No hay necesidad de estimar p_K pues $p_K = 1 - \sum_{l=1}^{K-1} p_k$. Resolviendo para p_k

$$p_k = p_K e^{f_k(x)} \tag{3.25}$$

y sumando sobre $k = 1, ..., K$

$$\sum_{k=1}^{K} p_k = p_K \sum_{k=1}^{K} e^{f_k(x)} \tag{3.26}$$

$$1/(\sum_{k=1}^{K} e^{f_k}) = p_K \tag{3.27}$$

por lo tanto

$$p_k = \frac{e^{f_k(x)}}{\sum_{j=1}^{K} e^{f_j(x)}} \tag{3.28}$$

Entonces la menos log-verosimilitud en cada pareja (x, y) es:

$$-\sum_{k=1}^{K} \mathbb{1}(y = k) \log p_k(x) \tag{3.29}$$

$$= -\sum_{k=1}^{K} \mathbb{1}(y = k) \log \frac{e^{f_k(x)}}{\sum_{j=1}^{K} e^{f_j(x)}} \tag{3.30}$$

$$= -\sum_{k=1}^{K} \mathbb{1}(y = k) f_k(x) + \log \left(\sum_{k=1}^{K} \mathbb{1}(y = k) \sum_{j=1}^{K} e^{f_j(x)} \right) \tag{3.31}$$

$$= -\sum_{k=1}^{K} \mathbb{1}(Y = k) f_k(x) + \log \left(\sum_{j=1}^{K} e^{f_j(x)} \right) \tag{3.32}$$

Así la derivada respecto a cada $f_k(x)$ de la log-verosimilitud en cada punto (x_i, y_i) es:

$$\mathbb{1}(y = k) - \frac{e^{f_k(x_i)}}{\sum_{j=1}^{K} e^{f_j(x)}} \tag{3.33}$$

finalmente se obtiene la expresión de la tabla anterior

$$-\mathbb{1}(y_i = k) + p_k(x_i) \tag{3.34}$$

Observación 3.18. En un problema de clasificación binario, en cada paso del descenso del gradiente, basta con estimar únicamente la probabilidad de la clase de interés, pues tomando el complemento se obtiene la probabilidad de la otra clase. En un problema multi-clase se deben ajustar en cada paso del descenso del gradiente K árboles, uno asociado para cada uno de las probabilidades $p_1, p_2, ..., p_K$.

Observación 3.19. El número de nodos terminales J en cada árbol estimado en 3.8 y 3.9 es fijo, pero podría ser que cada árbol estimado sea de distinto tamaño. Recordemos de 3.1.2 que una posible estrategia para escoger el tamaño de un árbol es ajustar un árbol grande y después podarlo por validación cruzada de acuerdo al criterio de complejidad. Esta estrategia tiene dos problemas principales:

FIGURA 3.8: Algoritmo Boosting del gradiente en un problema de regresión

Boosting del gradiente en un problema de regresión

El algoritmo es el siguiente:

1. Iniciamos $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$ la mejor constante que ajusta a todos los datos.
2. Para m=1 a M
   a) Para $i = 1, ..., N$
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$$
   b) Ajustar un árbol de regresión usando r_{im}, $i = 1, ..., N$ como respuesta, obteniendo las regiones terminales R_{jm} para $j = 1, ..., J$
   c) Para cada región R_j, $j = 1, ..., J$ encontramos
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$$
   d) Actualizar $f_m(x)$ a
   $$f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J} \gamma_{jm} \mathbb{1}(x \in R_{jm})$$
3. Finalmente el estimador es:
   $$\varphi(x) = f_M(x)$$

1. Es computacionalmente más costoso que árboles de tamaño fijo.
2. Presupone que cada árbol es el modelo final, o el único modelo. Lo que lleva a generar árboles demasiado grandes y tener un problema de sobre-ajuste.

3.4.3. Hiper-parámetros de boosting del gradiente

Hasta ahora solo hemos mencionado dos hiper-parámetros de boosting del gradiente que son el número de nodos finales de cada árbol J, y el número de iteraciones M. Existen otros dos hiper-parámetros usados comúnmente, que son: **contracción** y **submuestreo**, con estos hiper-parámetros se busca un ajuste más rápido del modelo y controlar el tamaño del paso del descenso del gradiente. En resumen estos son los hiper-parámetros de regularización

- **Número de nodos finales en Árbol:** $J \in \mathbb{N}$. (controla la profundidad del árbol)

FIGURA 3.9: Algoritmo Boosting del gradiente en un problema de clasificación

Boosting del gradiente en un problema de clasificación

El algoritmo es el siguiente:

1. Iniciamos $f_0(x) = 0$ para $k = 1, ..., K$
2. Para m=1 a M
   a) Para $i = 1, ..., N$ y para $k = 1, .., K$

$$r_{ikm} = -\left[\frac{\partial L(y_i, f_k(x_i))}{\partial f_k(x_i)}\right]_{f_k = f_{k,m-1}}$$

   b) Ajustar un árbol de regresión usando r_{ikm}, $i = 1, ..., N$ como respuesta, obteniendo las regiones terminales R_{jkm} para $j = 1, ..., J$

   c) Para cada región R_j, $j = 1, ..., J$ encontramos

$$\gamma_{jkm} = \arg\min_{\gamma_k} \sum_{x_i \in R_{jkm}} L(y_i, f_{k,m-1}(x_i) + \gamma)$$

   d) Actualizar $f_{km}(x)$ a

$$f_{km}(x) = f_{k,m-1}(x) + \sum_{j=1}^{J} \gamma_{jkm} \mathbb{1}(x \in R_{jkm})$$

3. Finalmente el estimador de f_k es

$$f_k(x) = f_{kM}(x)$$

   y la probabilidad de cada clase estimada es:

$$p_k(x) = \frac{e^{f_k(x)}}{\sum_{j=1}^{K} e^{f_j(x)}}$$

- **Contracción:** $\nu \in (0, 1)$ reduce el peso del siguiente clasificador del ensamble

$$\varphi_m(x) = \varphi_{m-1}(x) + \nu \gamma_{jm} \mathbb{1}(x \in R_{jm})$$

- **Sub-Muestreo:** $\eta \in (0, 1)$ es la proporción de la muestra entrenamiento en cada paso del algoritmo.
- **Número de árboles generados en el ensamble:** $M \in \mathbb{R}$

El parámetro de contracción controla los tamaños de los pasos del descenso del gradiente, si ν es pequeña entonces el algoritmo tardará más iteraciones en aproximar el valor óptimo de la función de pérdida. Y el sub-muestreo selecciona una fracción de la muestra de entrenamiento lo cual implica que el algoritmo ajusta más rápidamente en cada

iteración.
Según el algoritmo 1.1 los valores de los hiper-parámetros, J, ν, η, M, que minimicen el error de validación se encuentran por medio de validación o validación cruzada.

Aunque el proceso recomendado en la aplicación es el siguiente: Se prueban valores fijos de J, ν, η por validación cruzada mientras que M se elige mediante un criterio de frenado, esto es, cuando no se logre una disminución en la estimación del error después de cierto número de iteraciones del algoritmo.

3.4.4. Importancia de las variables

La importancia de las variables es análoga a boosting adaptativo excepto en el problema multiclase. La importancia cuadrada de la variable l es el promedio de la importancia cuadrada de la variable sobre los elementos del ensamble.

$$\mathcal{I}_l^2(\varphi) = \frac{1}{M} \sum_{m=1}^{M} \mathcal{I}_l^2(\varphi_m) \tag{3.35}$$

Para un modelo de clasificación de K clases la importancia depende de la k-ésima clase.

$$\mathcal{I}_{lk}^2 = \frac{1}{M} \sum_{m=1}^{M} \mathcal{I}_l^2(T_{mk}) \tag{3.36}$$

La importancia cuadrada total se puede tomar como el promedio de todas las clases.

$$\mathcal{I}_l^2 = \frac{1}{K} \sum_{k=1}^{K} \mathcal{I}_{lk}^2 \tag{3.37}$$

Por último la **importancia** de la variable $\mathcal{I}_l$ es la raíz cuadrada de la importancia cuadrada.

Capítulo 4

Aplicación

Resumen

En este capítulo se presenta la aplicación de los modelos explicados en los dos capítulos anteriores. El problema afrontado es la anticipación al abandono de cierto producto de una institución financiera. Se propone un esquema recursivo que introduce dependencia temporal en cada uno de los modelos, incorporando el pronóstico de abandono del tiempo anterior como variable explicativa. Se encuentra que esta técnica genera mejor desempeño que no incluir la predicción del mes anterior, encontrando el modelo que obtiene un desempeño superior.

4.1. El problema de la retención

Uno de los problemas más grandes que enfrentan las instituciones financieras es el abandono de sus clientes. Aún si una institución mantiene un aumento sostenido en el número de nuevos usuarios, este crecimiento puede verse mermado por el número de clientes que dejan la institución. Adicionalmente es menos costoso la retención de un cliente que la atracción de uno nuevo, por el costo que tiene el proceso de incorporación a una institución, así que es de gran importancia para las instituciones mantener a sus clientes activos. Detectar los patrones indicativos de la deserción de un cliente antes de que suceda, es de alto valor para una institución financiera. En la figura 4.1, se puede notar que la frecuencia de abandonos mensuales es un poco más baja que la frecuencia de originación de clientes, pero es marginal la diferencia que hay entre ellas, si se lograra reducir la tasa de pérdida de clientes se lograría tener una tasa neta de clientes nuevos por encima de la actual.

Esta aplicación se centra en identificar clientes de la institución que detendrán el uso de cierto producto de interés, dada una ventana de anticipo de a lo más tres meses, con

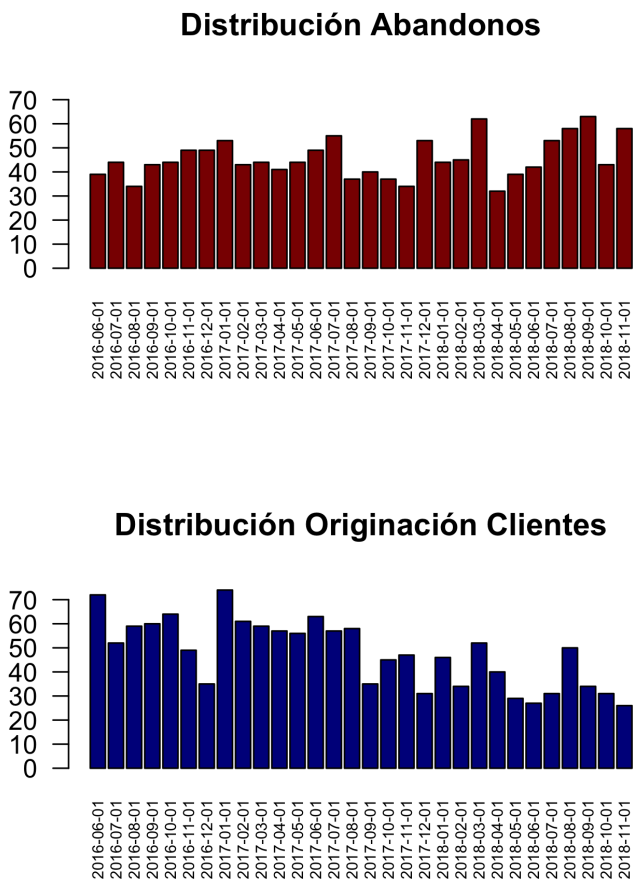


FIGURA 4.1: Distribución abandono y originación de clientes

el objetivo de lograr la intervención operativa preventiva sobre de ellos. El ejercicio se traduce en un problema de clasificación binaria donde las clases son:

- **La clase de abandono:** Clientes que abandonan el producto de interés con una ventana de anticipo de 1-3 meses.
- **La clase de no abandono** El complemento de la clase de abandono, clientes que no abandonan el producto en una ventana de 1-3 meses.

4.1.1. Definición de abandono de un producto

Para definir el abandono del producto de interés se unifican dos conceptos:

1. Dejar de usar completamente el producto (saldo cero) por al menos tres meses seguidos, sin retornar posteriormente a usar el producto. Se toma esta definición, porque si se analiza la frecuencia de la intermitencia máxima del uso del producto para cada cliente, se encuentra que las intermitencias mayores a tres meses ocurren

en menos del 90 % de los casos, como se puede ver en la tabla 1. Esto quiere decir, que si un cliente lleva tres o más meses sin hacer uso del producto, esperamos que vuelva a usarlo en aproximadamente el 10 % de los casos.

Intermitencia máxima (meses)	Porcentaje	Porcentaje Acumulado
0	75.43	75.43
1	8.84	84.27
2	3.48	87.75
3	2.06	89.81
4	1.38	91.19
5	1.09	92.28
6	0.79	93.07
7	0.56	93.63
8	0.56	94.19
9	0.53	94.73
10	0.46	95.19
11	0.39	95.57
12	0.35	95.92
>12	4.07	100

TABLA 4.1: Frecuencia intermitencia máximas en series de tiempo

2. La segunda definición es a través de una herramienta econométrica conocida como **cambios estructurales**. Los cambios estructurales son pruebas de hipótesis sobre quiebres en la media o en la tendencia de una serie de tiempo. El cambio estructural es aquel quiebre que sea lo "más significativo" sobre la serie de tiempo, es decir, donde haya mayor evidencia para rechazar la hipótesis nula de que no haya un cambio estructural. Si el cambio encontrado es significativo a la baja, y no regresa al nivel anterior de la serie de tiempo, decimos que hay un abandono por cambios estructurales. La idea de estas pruebas es lograr encontrar todos aquellos clientes que no han dejando el uso completo del producto, pero si han disminuido significativamente el uso respecto a su capacidad. Se puede ver un ejemplo de un cambio estructural en la figura 4.2. La prueba usada en este trabajo es cambio estructural sobre la media y/o sobre la tendencia de la serie de tiempo, en las cuales el punto de quiebre se determina dentro de la misma prueba. Esta es la prueba de Zivot-Andrews. A más detalle para que la prueba detecte un cambio estructural, se necesita que la serie sea estacionaria según Dickey-Fuller, para esto la prueba se realiza no sobre la serie original sino sobre promedios móvil de orden tres sobre ella. Se recomienda consultar el anexo A para mas detalles.

Para marcar un cambio estructural pedimos tres condiciones:

a) El cambio estructural sea significativo con una significancia del 5 %.

b) El promedio de la serie antes del cambio estructural sea mayor al promedio posterior al cambio estructural.

c) El promedio de los tres meses anteriores al cambio sean mayores al promedio de los últimos tres meses de la serie.

La condición *b*) es para encontrar los cambios estructurales a la baja y la condición *c*) para garantizar que la serie no haya regresado a los niveles anteriores en los últimos meses de información.

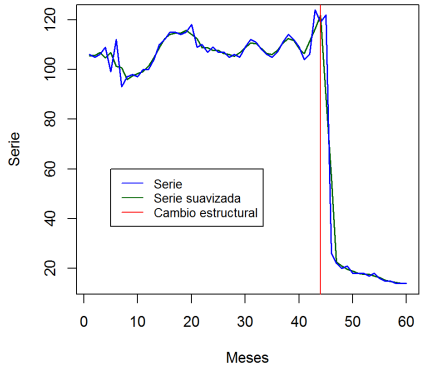


FIGURA 4.2: Ejemplo Cambio Estructural

Si se quiere ahondar a fondo sobre el funcionamiento de los cambios estructurales consultar el Anexo A. La fecha de abandono para cada cliente es la fecha de cambio estructural con las condiciones mencionadas, o en su caso, la última fecha de uso del producto antes de un periodo sin saldos de al menos tres meses continuos.

$$\text{Fecha abandono} = min(\text{Fecha último uso},\ \text{Fecha cambio estructural})$$

El proceso para general la variable objetivo es:

1. Filtrar las series de tiempo con la condición de fecha <= fecha abandono si para ese cliente se tiene la fecha de abandono.

2. Marcamos la bandera de abandono ($y = 1$) sobre los últimos tres meses de información de estos clientes antes del cambio estructural o el abandono, el resto de observaciones se marcan con la clase adicional ($y = 0$).

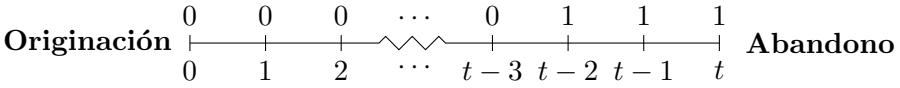


FIGURA 4.3: Bandera de abandono

4.2. Tratamiento de los datos

Para generar el conjunto de variables explicativas se considera una mezcla de variables internas y externas. Por protección de datos internos de la institución financiera, no se puede dar la descripción de las variables internas y se cambian los nombres originales de estas por `var_int`. Las variables externas se han dejado con su nombre original.

En los métodos basados en árboles no es necesario hacer limpieza de valores extremos ni tampoco quitar valores nulos de las variables explicativas, como se mencionó en la subsección 3.1.2.1. En regresión logística se limpian los valores extremos de las variables reemplazando los valores por encima del cuantil 95 % con el cuantil 95 % o por debajo del 5 % con el cuantil 5 % y los valores faltantes se cambian con cero, con la mediana de la variable o con la clase más frecuente si es categórica. Por ejemplo, si la variable con observaciones faltantes es el saldo de un producto, cambiamos los valores nulos con cero, porque denota la no tenencia del producto. Si la variable es la antigüedad del cliente, las observaciones faltantes se cambian con la mediana, porque no es adecuado rellenar con antigüedad cero. Y si la variable es categórica, las observaciones nulas se rellenan con la clase más frecuente.

Observación 4.1. Cabe destacar que la base de datos no contiene variables categóricas que sean fijas en el tiempo, estas se cambian por variables que son longitudinales, por ejemplo, la región en la que se encuentra el usuario. Estas variable es fija, por lo que el modelo puede capturar algún patrón o comportamiento que ocurrió en el pasado y puede sesgar a decir que ese mismo comportamiento ocurrirá en el futuro. Así, no se incluye la región geográfica como variable, pero se incluyen variables como el `itaee`, el cuál es un indicador de actividad económica estatal, que lleva información de la zona geográfica dependiente del tiempo.

4.2.1. Entrenamiento y prueba

El conjunto de entrenamiento es un periodo de 18 meses, y el conjunto de prueba es el mes siguiente al conjunto de entrenamiento. Este proceso se repite 12 veces, así obteniendo un conjunto de prueba final de 12 meses. Así se miden efectos estacionales que puedan afectar el desempeño del modelo.

Variable	Descripción
var_int	Variable interna confidencial individual por cliente
var_int_bma3	Promedio simple de últimas 3 observaciones variable interna
var_int_pdif3	Cambio porcentual variable interna (3m)
var_int_pdif3	Cambio porcentual variable interna (12m)
var_int_cv12	Coeficientes de variación de últimas 12 observaciones
asegurados	# Asegurados entidad federativa
attr_lag_3m	Bandera de abandono en 1,2 o 3 meses
energia	Energía eléctrica, agua y de gas por estado
igae	Indicador global de actividad económica por sector de actvidad
ind_sin_var7	Indicador de si no se está activa var_int7
ing_comer_men	Índice de ingresos del comercio al por menor por estado
itaee	Indicador trimestral de la actividad economica estatal
manufactura	Actividad industrial manufactureras por estado
n_mes_ult_var_int	# meses desde último uso de variable interna correspondiente
sow_atm	Participación de Institución Financiera en ATM's por estado
sow_suc	Participación de Institución Financiera ens sucursales por estado

Tabla 4.2: Variables incluidas en modelo

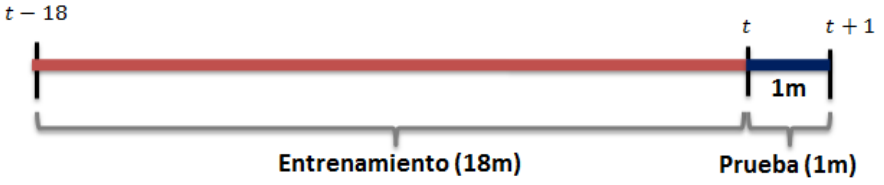


Figura 4.4: Línea de tiempo: entrenamiento y prueba

4.2.2. Balanceo de la muestra de entrenamiento

La base completa con la que se trabaja tiene una proporción de 3.2 % de unos (observaciones marcadas con bandera de abandono) y de 96.8 % de ceros (observaciones sin bandera de abandono). **Balancear una muestra de entrenamiento** consiste en cambiar la proporción de la clase de interés ($y = 1$) con el fin de que el algoritmo de aprendizaje no sesgue sus predicciones a la clase complemento ($y = 0$). En este ejercicio, el balanceo consiste en dejar todos los renglones cuya variable respuesta es uno y seleccionar aleatoriamente el 20 % de las observaciones en las cuales ($y = 0$), creando así una nueva base de entrenamiento. Con este modelo entrenado sobre una base balanceada, se hace inferencia sobre la base de final que no está balanceada. En la tabla 4.3 se compara el desempeño de una regresión logística ajustada sobre una muestra de entrenamiento balanceada y sobre una muestra de entrenamiento no balanceada. En el artículo [6] se dan otras alternativas de balanceo de muestras y se justifica su uso y las limitaciones que tienen. Es preferible balancear con un sub-muestreo del 20 % por las razones que se encuentran en los puntos a continuación.

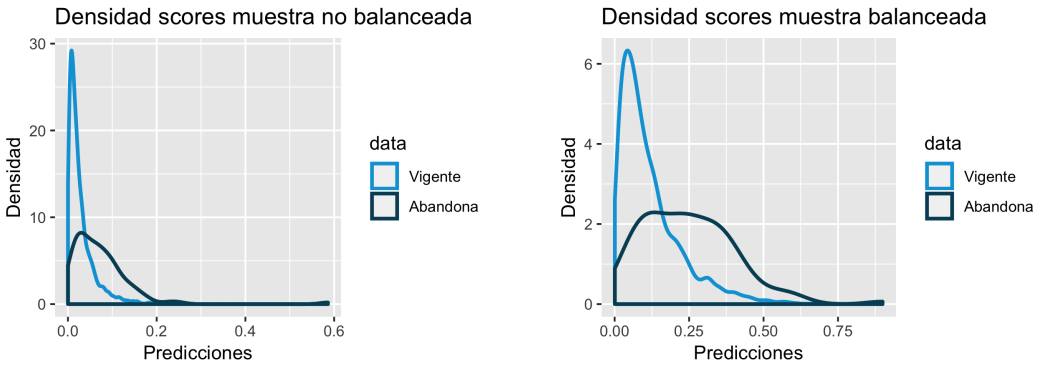


FIGURA 4.5: Muestra balanceada vs no balanceada

- **Muestra no balanceada:** Las predicciones se sesgan a la clase de ceros, aun así mantienen un desempeño un poco superior.
- **Muestra balanceada:** Aunque tiene un desempeño similar, pero un poco inferior al modelo no balanceado, en términos de **ABC** y de **PE**, los cuales se ven en la subsección 1.3.1, muestra dos ventajas principales:

  a) Al sub-muestrear los renglones con ceros se reduce el tamaño de la muestra considerablemente, lo cual hace más rápido el ajuste del modelo.

  b) Los pronósticos no están sesgados a la clase de ceros como el modelo entrenado sobre la base original, por lo que para un mismo punto de corte para ambos modelos, se generan mayor número de pronósticos para el modelo entrenado sobre la base balanceada, como se puede ver en la figura 4.5.

Por estas razones, decidimos ajustar todos los modelos subsecuentes sobre muestras balanceadas por sub-muestreo del 20 % de la clase de ceros. Para una empresa es de interés poder mantener el desempeño del modelo, y poder generar un número más grande de pronósticos para poder tener un mayor campo de acción de la iniciativa, claro sin perder credibilidad del modelo es decir que se reduzca considerablemente precisión del modelo, es importante tener un balance entre estas dos métricas.

Regresión logística con stepwise		
Muestra	Balanceada	No Balanceada
ROC_ABC	75.4 %	77.0 %
PE_ABC	13.3 %	14.4 %
Precisión	20.0 %	50 %
Exhaustividad	8.0 %	0.9 %
F1 (corte 50 %)	11.4 %	1.7 %

TABLA 4.3: Comparación muestra balanceada y no balanceada

4.3. Proceso de ajuste de los modelos con dependencia temporal

Dada la naturaleza temporal del ejercicio, cada uno de los modelos se ajusta mediante un proceso recursivo, que introduce una dependencia temporal con el mes anterior. Esta dependencia temporal se logra con un proceso recursivo que añada a cada observación de la base de entrenamiento (cuando es posible) la probabilidad de de abandono estimada del mes anterior. Entonces se traslada la información del tiempo anterior al tiempo actual, si se ve la tabla 4.13b encontramos que este método incrementa el desempeño de los modelos ajustados. El proceso de ajuste se encuentra en el algoritmo 4.6.

A continuación se va a dar el proceso de ajuste de cada uno de los métodos tratados en este trabajo, que es la selección de los hiper-parámetros, la importancia o siginificancia de sus variables dependiendo el caso y su desempeño en la base de prueba. Los modelos ajustados son:

- Regresión logística
- Árbol de clasificación
- Bagging de árboles
- Bosques aleatorios
- Boosting adaptativo con árboles
- Boosting del gradiente con árboles

4.4. Ajuste regresión logística

Se ajusta una regresión logística con función de enlace canónico sobre el ejercicio de abandono. La selección de variables se hace mediante el método stepwise, con el siguiente procedimiento de ajuste:

1. Ajustar un modelo considerando todas las variables. (Incluyendo la predicción del mes anterior).
2. Hacer una selección de las variables con el **algoritmo stepwise**.
3. Diagnóstico del modelo.
4. Métricas de desempeño clasificación.

Proceso de ajuste de los modelos inspirado $AR(1)$

El proceso de ajuste es el siguiente:

1. Estimamos un modelo inicial sobre los 18 meses del conjunto de entrenamiento balanceado
2. Se estiman los pronósticos de abandono $\hat{y}_{t,i}$ sobre el conjunto de entrenamiento total, donde t es el tiempo e i el individuo.
3. En cada tiempo t del conjunto de entrenamiento original, se agrega una nueva variable $\hat{y}_{t-1,i} \in (0, 1)$, la probabilidad estimada del mes anterior del mismo cliente, cuando es posible, en otro caso se deja como valor nulo.
4. Se balancea el nuevo conjunto de entrenamiento y se vuelve a estimar el modelo.
5. Las predicciones del nuevo modelo del último mes, se agregan a la base de prueba generando las predicciones sobres este, y por último se mide el desempeño del modelo.

FIGURA 4.6: Proceso de ajuste de los modelos inspirado $AR(1)$

La selección de variables con el algoritmo stepwise solo se realiza para el primer mes del conjunto de prueba, por restricciones computacionales, es decir el algoritmo tarda más de 8 horas en terminar su ajuste, sobre 74 variables y al rededor de 600 mil observaciones, estas es el tamaño de entrenamiento para el modelo usado en la aplicación real en el sistema bancario, consultar la observación 4.2. Para los siguientes 11 meses se usan las mismas variables seleccionadas por el algoritmo stepwise en el primer mes.

4.4.1. Selección de variables

En el primer mes del conjunto de prueba se ajusta un modelo logístico completo con un total de 74 variables explicativas. El modelo seleccionado por stepwise selecciona 28 variables. Los dos modelos son bastantes parecidos en términos de desempeño pero el segundo modelo incluye menos variables y es más parsimonioso, por lo que es preferible, como se puede ver en tabla 4.4. Notemos que hay una variable que no es significativa, aunque esta muy cerca de ser al 10 %, al final se deja porque incluirla o quitarla no afectaba el desempeño del modelo. La estructura de las correlaciones de las variables finales se puede ver en la figura 4.7, recordemos que en este enfoque estamos es de mayor importancia generar el modelo que genere las mejores predicciones de abandono, por encima de tener una interpretación clara de las variables independientes en la respuesta, así que la multicolinealidad no nos es de tanta relevancia.

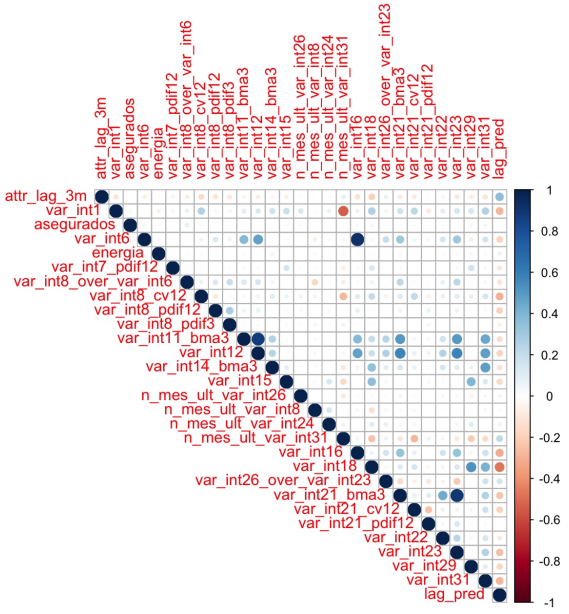


FIGURA 4.7: Correlograma regresión logística

Regresión logística en primer mes		
Muestra	Completa	Stepwise
ROC_ABC	75.4 %	75.6 %
PE_ABC	13.3 %	13.0 %
Núm. Variables	74	28
AIC	9,299	9,232

TABLA 4.4: Comparación modelo completo y stepwise

En la tabla 4.5 se ve el resumen del ajuste realizado por el modelo seleccionado por el stepwise. Se muestra cada variable, su parámetro asociado en la relación lineal y el nivel de significancia. También se da el valor del coeficiente de Akaike final.

4.4.2. Resultados clasificación

Ahora mostramos el desempeño de la clasificación durante el periodo de un año. Viendo la tabla 4.6 podemos ver que el desempeño varía por mes, esto quiere decir hay cierta estacionalidad del desempeño del modelo. Hay estacionalidad que no viene implícita en el modelo, por ejemplo en diciembre la gente abandona más. Si se realizan predicciones de cierto mes, esperar un rendimiento parecido a los años anteriores en ese mes.

	Variable dependiente:
	attr_lag_3m
var_int1	-0.018^{***} (0.006)
asegurados	0.00000^{*} (0.00000)
var_int6	0.011^{***} (0.002)
energia	0.003 (0.002)
var_int7_pdif12	-0.002^{**} (0.001)
var_int8_over_var_int6	-0.00002^{***} (0.00001)
var_int8_cv12	-0.044^{***} (0.007)
var_int8_pdif12	-0.004^{***} (0.001)
var_int8_pdif3	-0.005^{***} (0.001)
var_int11_bma3	0.00000^{***} (0.00000)
var_int12	-0.00000^{**} (0.00000)
var_int14_bma3	0.00000^{**} (0.00000)
var_int15	-0.008^{**} (0.004)
n_mes_ult_var_int26	0.014^{*} (0.007)
n_mes_ult_var_int8	-0.068^{***} (0.020)
n_mes_ult_var_int24	0.032^{***} (0.011)
n_mes_ult_var_int31	-0.009^{***} (0.003)
var_int16	-0.0004^{***} (0.0001)
var_int18	-0.336^{***} (0.052)
var_int26_over_var_int23	0.004^{***} (0.001)
var_int21_bma3	0.00000^{**} (0.00000)
var_int21_cv12	-0.045^{**} (0.019)
var_int21_pdif12	-0.0002^{**} (0.0001)
var_int22	-0.00000^{*} (0.00000)
var_int23	-0.00000^{**} (0.00000)
var_int29	0.216 (0.138)
var_int31	-0.002^{***} (0.001)
lag_pred	3.175^{***} (0.261)
Constant	-0.893^{***} (0.267)
Observaciones	11,990
Log-verosimilitud	-4,586.918
Coeficiente Inf. Akaike	9,231.836
Nota:	*p<0.1; **p<0.05; ***p<0.01

TABLA 4.5: Resumen ajuste regresión logísitica stepwise

TABLA 4.6: Desempeño Regresión Logísitica a través de 12m

Mes	ROC_ABC	PE_ABC	Precisión	Exhaustividad	F1	R^2
1	76 %	13 %	20 %	8 %	11 %	21 %
2	70 %	11 %	15 %	6 %	8 %	21 %
3	70 %	11 %	18 %	5 %	8 %	20 %
4	72 %	12 %	19 %	7 %	11 %	19 %
5	75 %	9 %	18 %	12 %	14 %	19 %
6	78 %	9 %	13 %	8 %	10 %	19 %
7	78 %	14 %	19 %	10 %	13 %	20 %
8	80 %	11 %	10 %	6 %	8 %	19 %
9	82 %	9 %	13 %	13 %	13 %	20 %
10	77 %	7 %	12 %	6 %	8 %	20 %
11	80 %	10 %	15 %	13 %	14 %	21 %
12	83 %	9 %	10 %	10 %	10 %	21 %
Promedio	77 %	10 %	15 %	9 %	11 %	20 %

4.5. Ajuste árbol de decisión

Se ajusta un árbol de decisión con la técnica propuesta en el algoritmo 3.2. Sobre cada una de las 12 iteraciones del ajuste del modelo, una para cada mes, se sigue el siguiente procedimiento:

1. Ajustar un árbol de clasificación profundo, con el único criterio de frenado que cada nodo terminal tenga al menos 20 observaciones.

2. Sobre el árbol ajustado se hace validación cruzada de 10-capas, y se elige aquel criterio de complejidad que minimice el error estimado por validación cruzada, se puede ver en la figura 4.8, en esta figura el error por validación cruzada ha sido escalado empezando en 1 para su fácil lectura. La validación cruzada se usa únicamente para la elección de los hiper-parámetros, es decir se eligen los hiper-parámetros que minimicen el error estimado por validación cruzada, a diferencia de regresión logística, en la cual se eligen los hiper-parámetros por el método stepwise.

3. Se estiman las probabilidades de abandono y se rezagan un mes, así se incluye como variable explicativa al modelo y se repiten los pasos 1 y 2.

4. Se genera con este último modelo los pronósticos sobre la base de prueba.

Al repetirse el proceso 12 veces, una para cada mes de la base de prueba se obtienen los resultados en la tabla 4.7.

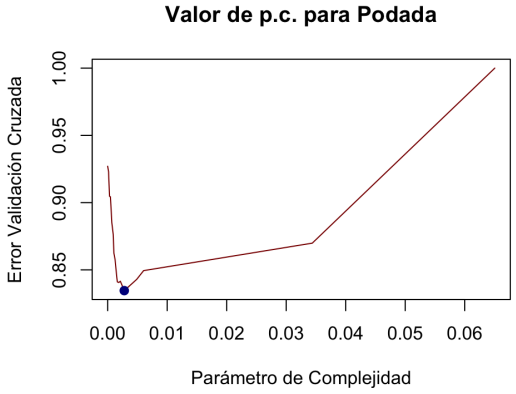


FIGURA 4.8: Elección parámetro de complejidad por validación cruzada

TABLA 4.7: Desempeño árbol de clasificación a través de 12m

Mes	ROC_ABC	PE_ABC	Precisión	Exhaustividad	F1
1	70 %	14 %	23 %	36 %	28 %
2	64 %	14 %	23 %	21 %	22 %
3	62 %	12 %	23 %	24 %	23 %
4	69 %	16 %	28 %	30 %	29 %
5	70 %	9 %	13 %	35 %	19 %
6	70 %	8 %	11 %	27 %	16 %
7	68 %	8 %	13 %	29 %	18 %
8	72 %	9 %	14 %	30 %	19 %
9	71 %	5 %	8 %	25 %	12 %
10	62 %	6 %	10 %	22 %	14 %
11	64 %	7 %	13 %	30 %	18 %
12	72 %	5 %	7 %	15 %	10 %
Promedio	68 %	9 %	16 %	27 %	19 %

Por último se calcula la importancia de las variables como el decrecimiento en la medida de impureza gini en cada uno de los nodos del árbol. En este trabajo se elige gini porque es la más común en aplicaciones [2]. En la figura 4.9 se visualiza la importancia de las variables, en particular la gran importancia que tiene la predicción generada el mes anterior por el árbol de clasificación, bastante por encima de las demás variables.

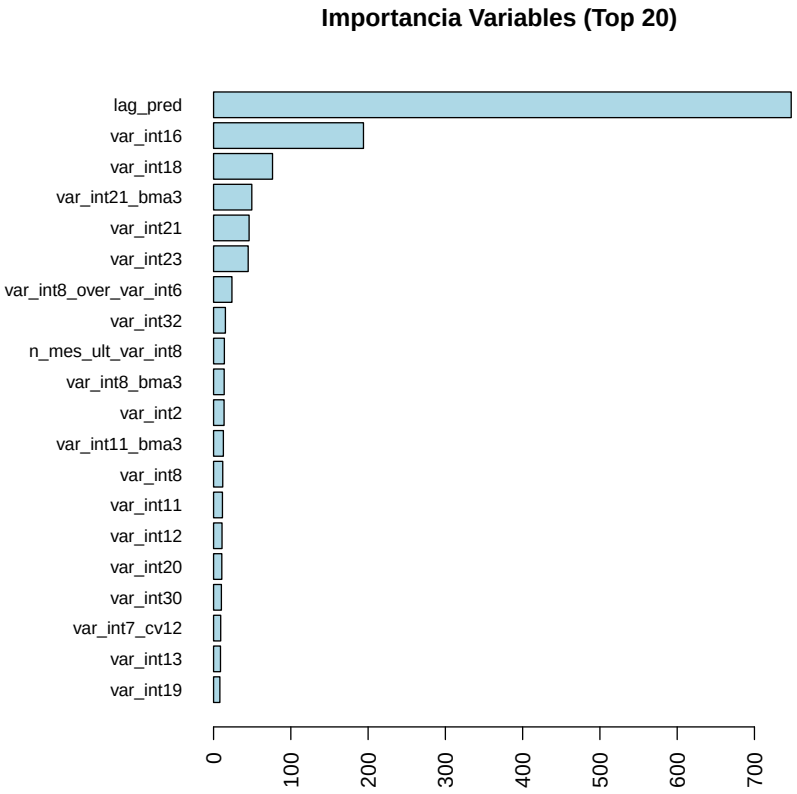


FIGURA 4.9: Importancia variables árbol de decisión

4.6. Ajuste bagging de árboles de clasificación

Recordemos que bagging es un caso particular de bosques aleatorios en donde se ajusta cada árbol sobre una muestra boostrap. En el algoritmo implementado se tiene como único hiper-parámetro el número de árboles que entran al ensamble. Así que no nos tenemos que preocupar por encontrar el parámetro de complejidad óptimo, cada árbol ajustado es el árbol máximo posible, la idea es que aunque individualmente sobre-ajusten, el promedio mitigue la varianza.

El hiper-parámetro del número de árboles en el ensamble se escoge mediante el error fuera de la bolsa, este se puede Ver en figura 4.10. Se usa el primer mes de la base de

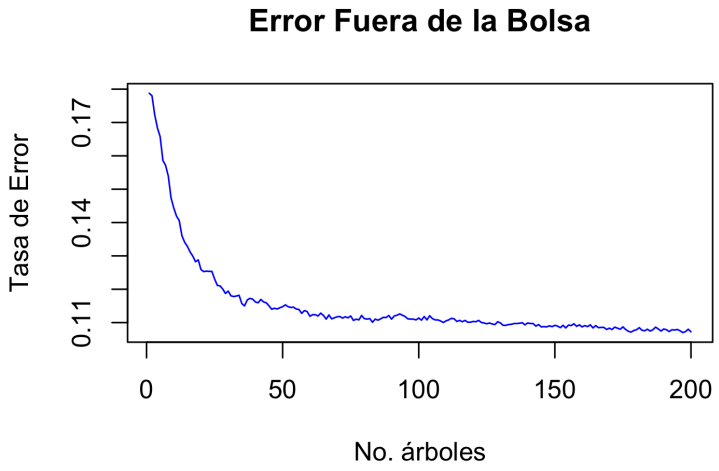


FIGURA 4.10: Error fuera de la muestra bagging

prueba como validación, escogiendo 200 iteraciones, es decir 200 árboles generados sobre muestras bootstrap.

La importancia de las variables es el promedio de la medida de impureza sobre los árboles del ensamble, esta se puede ver en la figura 4.11. La variable más importante es la predicción del mes anterior.

En la tabla 4.8, se puede ver el desempeño del modelo en cada mes de la base de prueba. Se supera de manera considerable el desempeño de un solo árbol de clasificación, visto en la sección previa. El método bagging logra reducir la varianza del problema, y así lograr un mejor desempeño, pues recordemos que esté método solamente ataca la varianza.

TABLA 4.8: Desempeño bagging de árboles a través de 12m

Mes	ROC_ABC	PE_ABC	Precisión	Exhaustividad	F1
1	92 %	60 %	69 %	52 %	59 %
2	83 %	46 %	72 %	31 %	43 %
3	92 %	64 %	75 %	48 %	59 %
4	92 %	72 %	77 %	65 %	71 %
5	91 %	57 %	63 %	50 %	56 %
6	91 %	46 %	53 %	43 %	48 %
7	88 %	51 %	59 %	44 %	50 %
8	92 %	61 %	79 %	52 %	63 %
9	92 %	50 %	51 %	46 %	48 %
10	83 %	38 %	55 %	34 %	42 %
11	96 %	62 %	65 %	60 %	63 %
12	100 %	100 %	78 %	100 %	88 %
Promedio	91 %	59 %	66 %	52 %	57 %

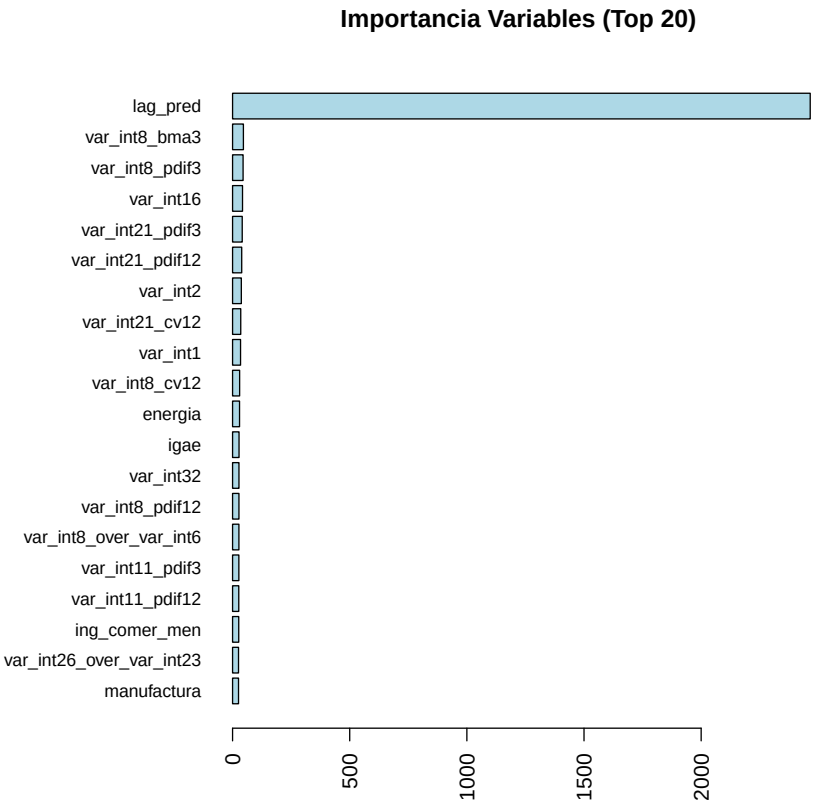


FIGURA 4.11: Importancia variables bagging

4.7. Ajuste bosques aleatorios

Los bosques aleatorios, a diferencia de bagging buscan reducir la correlación entre los árboles seleccionando b de las p variables en cada nodo de cada árbol, ajustado sobre una muestra boostrap, entonces los hiper-parámetros son el número de árboles que entran al ensamble y el número de variables incluidas en cada nodo. Recordemos que al igual que en la técnica anterior, cada árbol se hace crece a gran profundidad sin criterio de frenado. En la figura 4.12 se ve el error de clasificación fuera de la bolsa en relación al número de iteraciones y el número de variables seleccionadas en cada nodo del árbol. Se elige $b = 16$ variables seleccionadas y 300 árboles en el ensamble.

La importancia de las variables al igual que en bagging es el cambio promedio sobre los árboles en la medida de impureza gini, esta se puede ver figura 4.13. Por el cifrado que hacemos a los nombres de las variables no da mucha luz de cuales son las variables que tienen mayor peso en la clasificación, pero podemos notar que las 20 más importantes son todas variables internas, excepto por la predicción del mes anterior.

La tabla 4.9 muestra el desempeño del modelo en cada mes de la base de prueba, el desempeño es ligeramente más alto que bagging en el área bajo la curva precisión

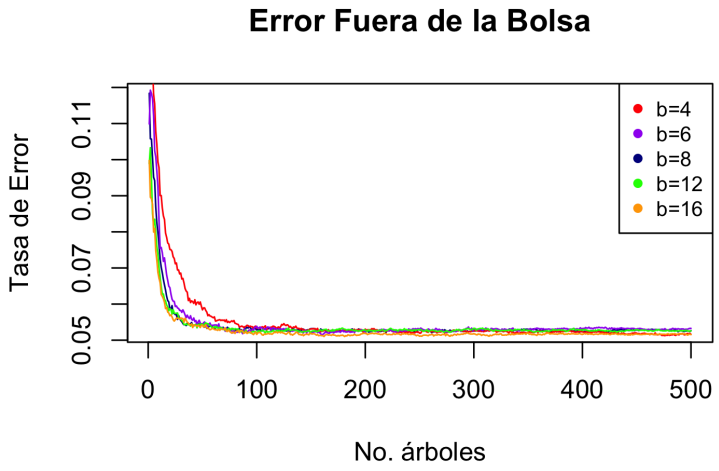


FIGURA 4.12: Error fuera de la muestra para distinto número de variables seleccionadas en bosques aleatorios

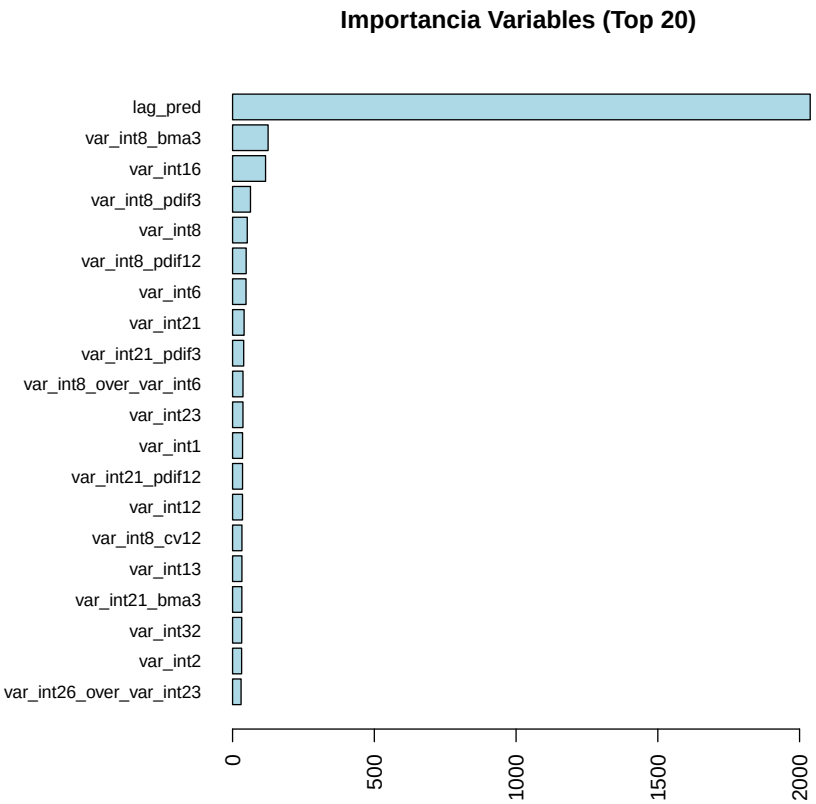


FIGURA 4.13: Importancia variables bosque aleatorio

exhaustividad, de 59 % a 63 %.

TABLA 4.9: Desempeño bosque aleatorio a través de 12m

Mes	ROC_ABC	PE_ABC	Precisión	Exhaustividad	F1
1	92 %	68 %	81 %	55 %	65 %
2	84 %	49 %	78 %	32 %	46 %
3	93 %	68 %	85 %	36 %	51 %
4	94 %	78 %	85 %	63 %	73 %
5	91 %	59 %	73 %	47 %	57 %
6	92 %	54 %	63 %	37 %	47 %
7	92 %	59 %	73 %	45 %	56 %
8	93 %	61 %	76 %	46 %	57 %
9	93 %	55 %	70 %	44 %	54 %
10	83 %	41 %	61 %	34 %	44 %
11	97 %	62 %	69 %	51 %	59 %
12	100 %	100 %	81 %	100 %	90 %
Promedio	92 %	63 %	75 %	49 %	58 %

4.8. Ajuste boosting adaptativo

El método de boosting adaptativo reduce el sesgo de las predicciones, recursivamente aprendiendo de los errores de las iteraciones con base en una función de pérdida exponencial, que a su vez reduce la varianza al tomar un promedio ponderado de varias predicciones.

Como boosting no se estima sobre una muestra bootstrap no se puede estimar el error fuera de la bolsa, y al ser más tardado el ajuste del modelo, ya que validación cruzada de K-capas requiere estimar K distintos modelos, resulta consumir mucho tiempo, así que se ajusta el modelo por validación normal.

Se usa el primer mes de la muestra de prueba como el conjunto de validación, y sobre este se eligen los hiper-parámetros que reduzcan el error por validación, los cuales se dejan fijos para los siguientes ajustes del modelo. En la figura 4.14 vemos el error de clasificación para distintos valores de la profundidad del árbol y número de iteraciones del algoritmo. Se elige una profundidad de 3 y 70 iteraciones. En la parte superior de la figura, se ve el error en entrenamiento de las tres opciones y en la figura 4.15 se tiene la importancia de las variables.

El desempeño del modelo es más bajo que bagging y que bosques aleatorios, lo cual sugiere que el ejercicio tiene un error más grande de varianza que de sesgo.

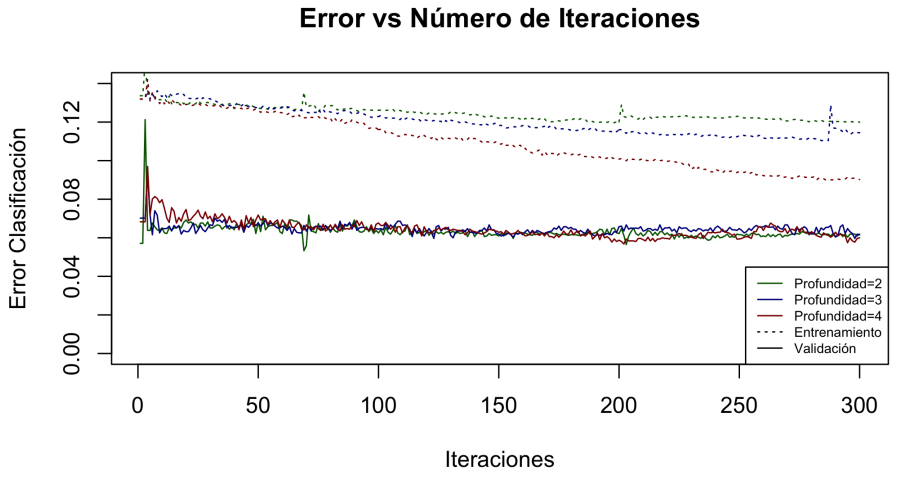


FIGURA 4.14: Error en validación para distinta profundidad de los árboles en AdaBoost

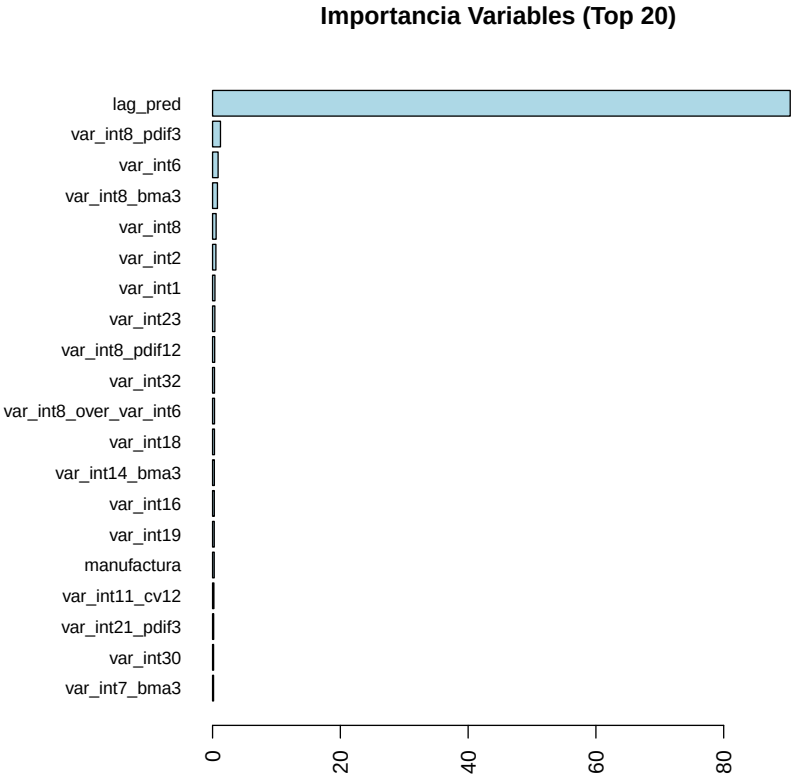


FIGURA 4.15: Importancia variables adaboost

4.9. Boosting del gradiente

Boosting del gradiente minimiza la función de pérdida (log-verosimilitud binomial) por medio de descenso del gradiente, en cada iteración se ajusta un árbol de regresión para

TABLA 4.10: Desempeño boosting adaptativo a través de 12m

Mes	ROC_ABC	PE_ABC	Precisión	Exhaustividad	F1
1	85 %	19 %	25 %	36 %	30 %
2	76 %	16 %	19 %	27 %	22 %
3	80 %	17 %	21 %	23 %	22 %
4	79 %	13 %	16 %	33 %	22 %
5	82 %	10 %	11 %	32 %	16 %
6	85 %	15 %	15 %	34 %	21 %
7	84 %	11 %	13 %	29 %	18 %
8	84 %	12 %	16 %	37 %	22 %
9	85 %	9 %	14 %	46 %	21 %
10	77 %	9 %	9 %	25 %	13 %
11	82 %	7 %	11 %	25 %	15 %
12	90 %	9 %	10 %	33 %	15 %
Promedio	83 %	12 %	15 %	32 %	20 %

poder extender el modelo no solo a las observaciones de entrenamiento sino a todo el espacio muestral.

Los hiper-parámetros ajustados son la profundidad de los árboles, la contracción y el número de iteraciones. Estos se escogen por validación cruzada en la base de entrenamiento del primer mes. En la figura 4.16 vemos el error de clasificación para distintos valores de los parámetros. También en la tabla 4.11 se ve el error mínimo de cada una de las opciones. Basándonos en ésto se selecciona una profundidad máxima de 7, $\nu = 0.5$ y 230 iteraciones. Se probó también el parámetro de sub- muestreo pero se encontró que no generaba alguna mejora considerable, por lo que se excluye apriori.

		Profundidad Máxima				
		4	5	6	7	8
Contracción (ν)	0.01	2.97 %	3.03 %	3.00 %	2.97 %	2.90 %
	0.05	2.90 %	3.00 %	2.84 %	2.74 %	2.74 %
	0.5	3.00 %	2.94 %	2.87 %	2.68 %	3.10 %
	1	3.41 %	3.29 %	3.77 %	3.51 %	3.61 %

TABLA 4.11: Error clasificación mínimo para distinta profundidad y contracción con 230 iteraciones

El desempeño del modelo es más alto que el adaboost, pero no logra ser tan alto como bosques aleatorios. Para este ejercicio en particular bosques aleatorios es aquel que logra tener el mejor desempeño, tanto en $AUC_{Bosque} = 92\,\% > AUC_{BoostGrad} = 91\,\%$ y $F1_{Bosque} = 57\,\% > F1_{BoostGrad} = 50\,\%$. Estas medidas se encuentran en la tabla 4.12.

Observación 4.2. Por cuestiones de sensibilidad de la información, los datos sobre los que se realizo el completo de este trabajo no consisten el total de la base de datos de la institución financiera, sino un muestreo aleatorio de alrededor del 10 % de los clientes, así que los resultados de este trabajo difieren sobre los obtenidos con el total de la base de datos de la institución financiera. En este último caso se obtiene que el modelo con el

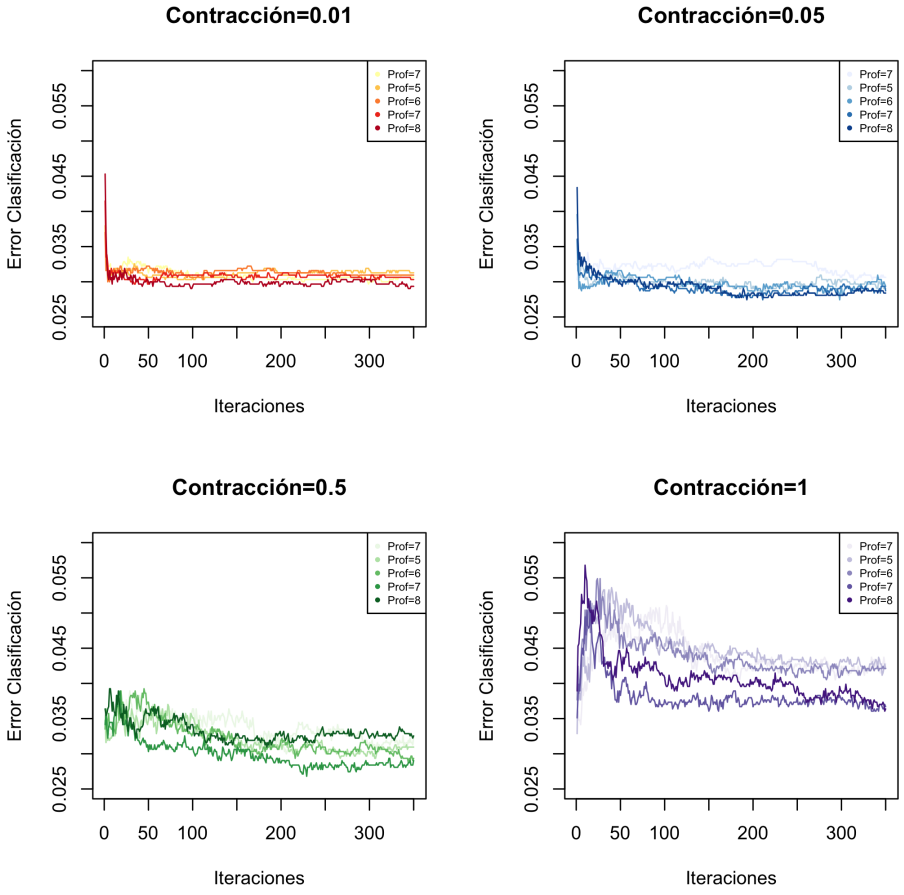


FIGURA 4.16: Error de clasificación en validación cruzada para distinta profundidad de los árboles y distinta contracción

mejor desempeño es el boosting del gradiente. Creemos que al tener menos observaciones se puede tener un problema de sobre-ajuste, en el cual bosques aleatorios logra reducir más la varianza y en consecuencia obtener mejor desempeño.

TABLA 4.12: Desempeño boosting del gradiente a través de 12m

Mes	ROC_ABC	PE_ABC	Precisión	Exhaustividad	F1
1	93 %	62 %	63 %	53 %	58 %
2	86 %	44 %	57 %	34 %	43 %
3	91 %	52 %	55 %	42 %	48 %
4	93 %	70 %	64 %	67 %	65 %
5	92 %	46 %	45 %	50 %	47 %
6	92 %	37 %	39 %	45 %	42 %
7	90 %	47 %	55 %	38 %	45 %
8	90 %	52 %	50 %	48 %	49 %
9	91 %	48 %	41 %	48 %	44 %
10	84 %	28 %	40 %	33 %	36 %
11	95 %	49 %	42 %	47 %	45 %
12	100 %	95 %	66 %	100 %	80 %
Promedio	91 %	52 %	51 %	50 %	50 %

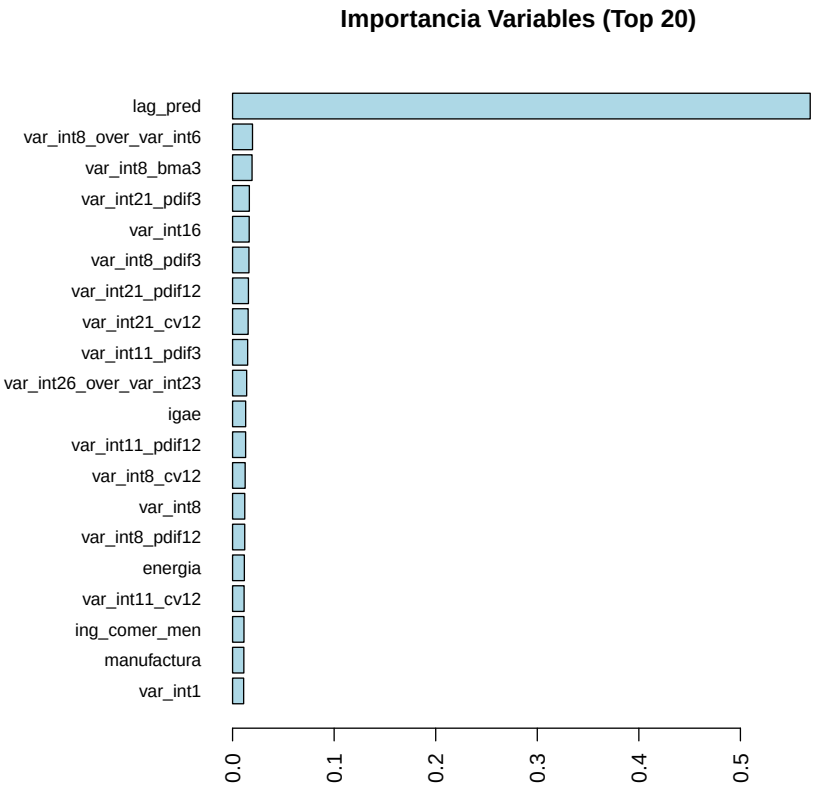


FIGURA 4.17: Importancia variables boosting del gradiente

4.10. Selección del modelo final

A continuación se muestra la tabla con las medidas promedio de cada uno de los métodos propuestos, y se añade la misma tabla pero sin incluir la predicción del mes anterior en las variables. Los modelos que incluyen el estimado del mes anterior superan en su desempeño a los modelos que no lo incluyen excepto en el árbol de clasificación. Bagging, bosques aleatorios y boosting del gradiente muestran mejoras sustanciales en la PE ABC y la puntuación F1. El árbol de clasificación y boosting adaptativo son los que muestran menos cambio en incluir la predicción del mes anterior.

En este ejercicio nuestro modelo campeón es el bosque aleatorio, superando a los demás en ROC ABC, PE ABC y puntuación F1.

Observación 4.3. Usualmente no se presenta, por el trabajo extra que presenta, pero para mejorar la comparación de modelos en la tabla 4.13, se puede incluir el intervalo de confianza estimado para cada una de las métricas, que se calcula generando estimaciones de estas por bootstrap y después encontrando los cuantiles a cierto nivel de confianza, y comparar estos intervalos sobre la métricas con la predicción del mes anterior y la

Mes	ROC_ABC	PE_ABC	Precisión	Exhaustividad	F1
Reg. log.	77 %	10 %	15 %	9 %	11 %
Árbol clas.	68 %	9 %	16 %	27 %	19 %
Bagging	91 %	58 %	67 %	50 %	56 %
Bosque alea.	92 %	62 %	73 %	49 %	57 %
Adaboost	83 %	12 %	15 %	32 %	20 %
Boost. grad.	91 %	52 %	51 %	50 %	50 %

(A) Métricas **con** variable predicción mes anterior

Mes	ROC_ABC	PE_ABC	Precisión	Exhaustividad	F1
Reg. log.	73 %	6 %	21 %	5 %	8 %
Árbol clas.	69 %	8 %	13 %	22 %	17 %
Bagging	83 %	17 %	20 %	23 %	22 %
Bosque alea.	85 %	23 %	36 %	22 %	27 %
Adaboost	81 %	12 %	15 %	24 %	18 %
Boost. grad.	91 %	25 %	33 %	28 %	30 %

(B) Métricas **sin** variable predicción mes anterior

TABLA 4.13: Métricas de desempeño promedio mensual

métricas sin la predicción del mes anterior. Recordemos también que en esta tabla no se incluye el error de clasificación de cada modelo porque la base es sumamente desbalanceada y está métrica no aporta ninguna luz del desempeño del modelo, ya que de hecho disminuye si se contrasta con un modelo donde todos los pronósticos son la clase mayoritaria.

Conclusiones

En este trabajo se presentó la teoría de algunos métodos de aprendizaje estadístico supervisado de clasificación binaria además de su aplicación práctica en el sistema bancario.

Vimos al aprendizaje estadístico supervisado como un problema de minimización de una función de pérdida. Se revisaron los modelos lineales generalizados con un enfoque particular en la regresión logística, y se desarrolló la teoría de métodos basados en árboles con técnicas de ensamble. Se aplican estos métodos en un problema temporal, obteniendo resultados variantes en el tiempo. Encontramos que el esquema recursivo inspirado en un $AR(1)$ logra mejores medidas de desempeño, si se incluye la probabilidad del mes anterior y que los bosques aleatorios tienen un desempeño superior a los demás modelos presentados en este trabajo.

Se piensa que incluir la probabilidad del mes anterior como una variable adicional al modelo mejora algunos modelos y otros no, por la razón de que modelos que son muy susceptibles a sobre ajustar como los árboles de decisión, al darle una importancia muy grande a esta variable se sesgan demasiado a lo que esta variable pronostica. Métodos como la regresión logística y bosques aleatorios controlan el sobre-ajuste, entonces incorporar esta variable mejora el desempeño. Habría que darle una justificación matemática a este argumento.

Cabe destacar que existen otros métodos de aprendizaje supervisado que no se probaron en este trabajo como son las máquinas de soporte vectorial y las redes neuronales. De particular interés son, los modelos secuenciales de redes neuronales, que incorporan dependencia temporal. Sería atractivo probar si estos modelos generan resultados superiores a los vistos en este trabajo. Por ejemplo, seguir una arquitectura de una red como la que se usa en sistemas de detección de palabras de activación (ejemplos: “Hey Siri”, “Alexa”), podría generar buenos resultados al ser sistemas que funcionan para bases altamente des-balanceadas con dependencia temporal [14].

Apéndice A

Prueba Zivot-Andrews de cambios estructurales

A.1. Series de tiempo y estacionariedad

Definición A.1. Una **serie de tiempo** $\{X_t\}_{t\in T}$ es un proceso estocástico a tiempo discreto donde T es un conjunto de índices a lo más numerable.
Si $\mathbb{E}[X_t^2] < \infty$ definimos las funciones

1. Función de medias $\mu_t = \mathbb{E}[X_t]$
2. Función de autocovarianzas $\gamma(t,s) = \mathbb{E}[(X_t - \mu_t)(X_s - \mu_s)]$
3. Función de autocorrelaciones $\rho(t,s) = \frac{\gamma(t,s)}{\sqrt{\gamma(t,t)}\sqrt{\gamma(s,s)}}$

Definición A.2. Una serie de tiempo $\{X_t\}_{t\in T}$ de segundo momento finito $\mathbb{E}[|X_t|^2] < \infty$ es **débilmente estacionaria** si su primer y segundo momentos son invariantes en el tiempo, $\forall t, s \in T$ entonces $\mu_t = \mu$ y $\gamma(t,s) = \gamma(t+h, s+h)$.

Definición A.3. Un proceso $\{Y_t\}$ es llamado de **diferencia estacionaria**, si al diferenciar el proceso $\nabla Y_t = Y_t - Y_{t-1}$ resulta ser estacionario. También se le conoce como un proceso con una **raíz unitaria** ya que el polinomio de rezagos $\phi(z) = 1 - \phi z$ tiene raíz $z = 1$ o $z = -1$ pues $|\phi| = 1$.

Definición A.4. Un proceso es llamado de **tendencia estacionaria** si basta con quitarle la tendencia para que el proceso sea estacionario.

Una serie de tiempo de la forma $Y_t = \phi Y_{t-1} + \varepsilon_t$ con $\mathbb{E}[\varepsilon_t] = 0$ y $\mathbb{E}[\varepsilon_t^2] = \sigma^2$ llamada autoregresiva de orden 1 o AR(1) tiene la propiedad de ser estacionario si $|\phi| < 1$ y de diferencia estacionario si $\phi = 1$. Esto motiva la prueba Dickey-Fuller.

A.1.1. Prueba Dickey-Fuller

Sea $\{X_t\}$ un proceso $AR(1)$, $X_t = \phi X_{t-1} + \epsilon_t$, entonces

$$\begin{aligned} X_t - X_{t-1} &= \phi X_{t-1} - X_{t-1} + \varepsilon_t \\ \nabla X_t &= (\phi - 1)X_{t-1} + \varepsilon_t \\ \nabla X_t &= c_0 X_{t-1} + \varepsilon_t \end{aligned}$$

Si $c_0 = \phi - 1$ y la prueba asociada sería $\phi = 1$ si $c_0 = 0$ contra $|\phi| < 1$ si $c_0 < 0$

$$\begin{aligned} H_0 : c_0 = 0 \quad \text{vs} \quad H_1 : c_0 < 0 \\ \text{“Raíz unitaria”} \quad \text{vs} \quad \text{“No hay raíz unitaria.”} \end{aligned}$$

Basado en lo anterior hay tres versiones de la prueba, c_0 se estima por mínimos cuadrados el estadístico de prueba es $\tau = \frac{\hat{c_0}}{sd(\hat{c_0})} = \frac{\hat{\phi}-1}{S_{\hat{\phi}}}$ y B_t es el movimiento Browniano.

1. Si el proceso no tiene tendencia ni media $\mu = 0$ i.e. $X_t = \phi X_{t-1} + \epsilon_t$

$$\begin{aligned} H_0 : \nabla X_t = c_0 X_{t-1} + \varepsilon_t \text{ con } c_0 = 0 \text{ vs} \\ H_1 : c_0 < 0 \end{aligned}$$

$$\tau \xrightarrow{d} \frac{\frac{1}{2}[(B_1)^2 - 1]}{[\int_0^1 B_t dt]^{1/2}}$$

2. Si el proceso no tiene tendencia pero si tiene media y $\mathbb{E}[X_t] = \mu$ i.e. $X_t = \mu + \phi X_{t-1} + \epsilon_t$

$$\begin{aligned} H_0 : \nabla X_t = \mu + c_0 X_{t-1} + \varepsilon_t \text{ con } c_0 = 0 \text{ vs} \\ H_1 : c_0 < 0 \end{aligned}$$

$$\tau_\mu \xrightarrow{d} \frac{\frac{1}{2}[(B_1)^2 - 1] - B_1 \int_0^1 B_t dt}{[\int_0^1 B_t^2 dt - (\int_0^1 B_t dt)^2]^{1/2}}$$

3. Corresponde al estadístico encontrado de una serie de tiempo que presenta media distinta de cero y una tendencia lineal i.e. $X_t = a + bt + \phi X_{t-1} + \varepsilon_t$. Esta prueba compara: *diferencia estacionaria* vs *tendencia estacionaria*

$$\begin{aligned} H_0 : \nabla X_t = a + bt + c_0 X_{t-1} + \varepsilon_t \text{ con } c_0 = 0 \text{ y } b = 0 \text{ vs} \\ H_1 : X_t = a + X_{t-1} + \varepsilon_t \text{ i. e. } c_0 < 0 \end{aligned}$$

$$\tau_\mu \xrightarrow{d} \frac{\int_0^1 B_t^* dB_t}{[\int_0^1 (B_t^*)^2 dt]^{1/2}}$$

$$B_t^* = B_t - (4 - 6t) \int_0^1 B_s ds - (12t - 6) \int_0^1 s B_s ds$$

A.1.2. Prueba Dickey-Fuller aumentada

Esta prueba extiende la prueba Dickey-Fuller a series de tiempo de la forma $X_t = \phi_1 X_{t-1} + ... + \phi_p X_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + ... + \theta_q \varepsilon_{t-q}$ con $\mathbb{E}[\varepsilon_t] = 0$ y $\mathbb{E}[\varepsilon_t^2] = \sigma^2$ llamadas ARMA(p,q). Nuevamente se pueden plantear tres pruebas de hipótesis, construyendo los estadísticos τ, τ_μ y τ_t que están asociados al proceso sin media, al proceso con media y al proceso con media y tendencia respectivamente. Se prueba que la distribución de estos estadísticos vuelve a ser la misma. La prueba se puede encontrar en [12]. En este trabajo solo se presenta el caso de un proceso ARMA con media sin tendencia.
Sea $\{X_t\}$ con media μ y se plantea el siguiente contraste de hipótesis

$$H_0 : \Phi(B)\nabla X_t = \Theta(B)\epsilon_t$$
$$vs$$
$$H_1 : \Phi(B)(1-\rho B)(X_t - \mu) = \Theta(B)\epsilon_t \text{ si } \mid \rho \mid < 1$$

Si el proceso $ARMA(p,q)$ ajustado al ∇X_t es causal y estacionario entonces bajo H_0 tenemos que hay raíz unitaria. Viendo la hipótesis alternativa notamos que si $\rho = 1$ se regresa a la hipótesis nula y si $\mid \rho \mid < 1$ entonces el proceso sin diferenciar ya es estacionario.

Suponiendo que el proceso diferenciado bajo H_0 es invertible podemos encontrar el polinomio infinito $AR(\infty)$ dado por

$$\Pi(z) = \frac{\Phi(z)}{\Theta(z)} \tag{A.1}$$

y podemos rescribir la prueba como

$$H_0 : \Pi(B)\nabla X_t = \epsilon_t$$
$$vs$$
$$H_1 : \Pi(B)(1-\rho B)(X_t - \mu) = \epsilon_t \text{ si } \mid \rho \mid < 1$$

Se aproxima $\Pi(B)$ por un polinomio finito de orden m y se usa la siguiente descomposición de polinomios

Proposición A.5. *Descomposición Polinomios Finitos*
Sea $A(B) = 1 - a_1 B - a_2 B^2 - \cdots - a_m B^m$ *si* $\sum_{i=1}^{m} a_i \neq 1$ *entonces* $A(B)$ *admite la siguiente descomposición*

$$A(B) = A(1)B + C(B)(1-B) \tag{A.2}$$

donde $C(B)$ *es el polinomio de orden* $m-1$ *dado por*

$$C(B) = 1 - c_1 B - c_2 B^2 - ... - c_{m-1} B^{m-1}$$

con $c_j = \sum_{i=j+1}^{m} a_i$

Entonces aproximando $\Pi(B)(1-\rho B)$ *por un polinomio finito de orden* m*, tenemos por A.2 que*

$$\Pi(B)(1-\rho B) = (1-\Pi_1-\Pi_2-...-\Pi_m)(1-\rho)B \tag{A.3}$$
$$+ (1-c_1B-...-c_{m-1}B^{m-1})(1-B) \tag{A.4}$$

Así el modelo de H_1 *es*

$$(1-\Pi_1-\Pi_2-...-\Pi_m)(1-\rho)B(X_t-\mu)+ \tag{A.5}$$
$$+ (1-c_1B-...-c_{m-1}B^{m-1})\nabla(X_t-\mu) = \epsilon_t \tag{A.6}$$

haciendo $c_0 = -(1-\Pi_1-\Pi_2-...-\Pi_m)(1-\rho)$ *y* $c = -\mu c_0$ *se escribe como*

$$-c_0X_{t-1} - c + \nabla X_t - c_1\nabla X_{t-1} - ... - c_{m-1}\nabla X_{t-m+1} = \epsilon_t \tag{A.7}$$

y tenemos la construcción final

$$\nabla X_t = c + c_0X_{t-1} + c_1\nabla X_{t-1} + ... + c_{m-1}\nabla X_{t-m+1} + \epsilon_t \tag{A.8}$$

Se tiene una regresión de ∇X_t *en términos de ella misma retrasada cada vez un periodo mas hasta* $m-1$ *retrasos. Se vuelve a estimar por mínimos cuadrados* $\hat{c}_0$ *y se desea probar* $\hat{c}_0 = 0$ *que es equivalente a* $\hat{\rho} = 1$*. Nuevamente se encuentra la distribución límite de un estadístico tipo t dado por*

$$\tau = \frac{\hat{c_0}}{sd(\hat{c_0})} \tag{A.9}$$

y además las hipótesis son

H_0 : *"Hay raíz unitaria"*

vs

H_1 : *"El proceso sin diferenciar ya es estacionario"*

Notemos que para $c, c_1, \ldots, c_{p-1}$ *se puede usar una prueba t-usual para ver si son significativos pero para* c_0 *se usa la distribución asintótica. En general se sugiere usar* $m = [n^{1/4}] + 1$ *para escoger hasta que término truncar el polinomio infinito* $\Pi(z)$*.*

A.2. Pruebas de cambio estructural de raíces unitarias

La prueba de Perron y la prueba de Zivot-Andrews son una generalización de la prueba Dickey-Fuller aumentada que introducen funciones indicadoras que marcan un cambio en la estructura de la serie de tiempo, ya sea en la media o en la tendencia. En la prueba Perron se debe determinar de antemano (exógena) el punto de cambio de la prueba, en la prueba Zivot-Andrews el punto de cambio estructural se determina dentro de la prueba (endógena). Los cambios estructurales que se van a probar

- *Cambio en el nivel (media) de la serie*
- *Cambio en la tendencia de la serie*
- *Una combinación de ambas*

Los cuales se ilustran en la Figura A.1.

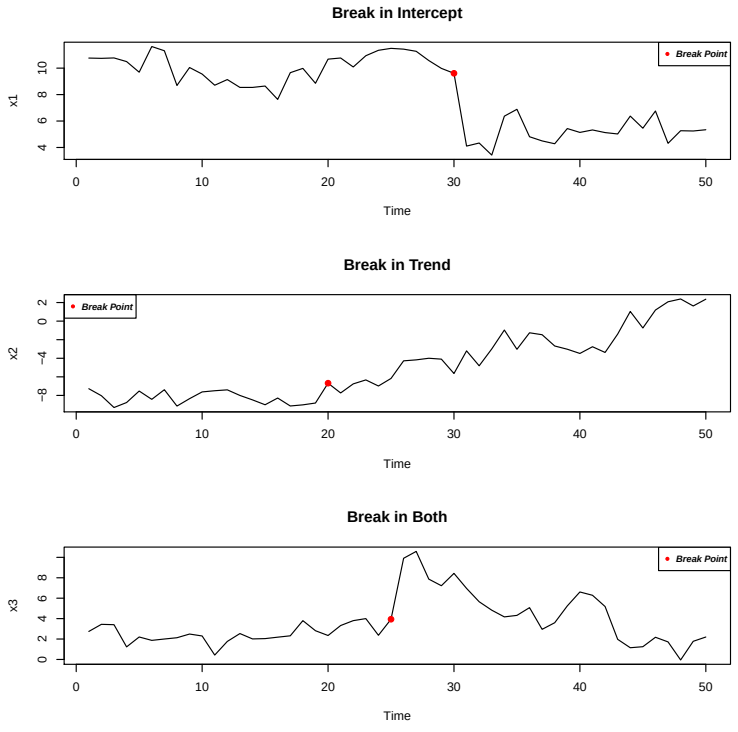


FIGURA A.1: Diferentes tipos de cambios estructurales

A.2.1. Prueba Perron

Pierre Perron desarrolla una metodología para probar la presencia de un cambio en la estructura de la serie en un momento **fijo** *del tiempo determinado apriori. Su metodología esta basado en lo previamente hecho por* **Dickey-Fuller** *considerando además los*

dos posibles cambios estructurales y la combinación de ambos como se presentan en la Figura A.1.En [10] se permite la presencia de cambio estructural tanto en la hipótesis nula como en la alternativa.
Notemos que el contraste de hipótesis es

$$H_0 : \textit{Diferencia estacionaria y cambio estructural}$$
$$\textit{vs}$$
$$H_1 : \textit{Tendencia estacionaria y cambio estructural .}$$

En la prueba de Cambio en intercepto *d y $(a_2 - a_1)$ representan el tamaño del salto de la media en H_0 y H_1 respectivamente.*
En la prueba de Cambio en tendencia *$(a_2 - a_1)$ y $(b_2 - b_1)$ son la magnitud del cambio en la tendencia en H_0 y H_1 respectivamente.*
En la prueba de Cambio en intercepto y tendencia *d y $(a_2 - a_1)$ son la magnitud del salto de la media y del cambio en la tendencia bajo H_0. Y $(a_2 - a_1)$ y $(b_2 - b_1)$ representan el tamaño del salto de la media y la magnitud del cambio en la tendencia bajo H_1.*

A) Cambio en Intercepto

$$H_0 : \ X_t = a + d\mathbb{1}_{[t=t^*+1]} + X_{t-1} + \epsilon_t \ \textit{vs} \ H_1 : \ X_t = a_1 + bt + (a_2 - a_1)\mathbb{1}_{[t>t^*]} + \epsilon_t$$

B) Cambio en Tendencia

$$\begin{aligned} H_0 : \ X_t &= a_1 + (a_2 - a_1)\mathbb{1}_{[t>t^*]} + X_{t-1} + \epsilon_t \ \textit{vs} \\ H_1 : \ X_t &= a + b_1 t + (b_2 - b_1)(t - t^*)\mathbb{1}_{[t>t^*]} + \epsilon_t \end{aligned}$$

C) Cambio en Intercepto y Tendencia

$$\begin{aligned} H_0 : \ X_t &= a_1 + d\mathbb{1}_{[t=t^*+1]} + (a_2 - a_1)\mathbb{1}_{[t>t^*]} + X_{t-1} + \epsilon_t \ \textit{vs} \\ H_1 : \ X_t &= a_1 + b_1 t + (a_2 - a_1)\mathbb{1}_{[t>t^*]} + (b_2 - b_1)(t - t^*)\mathbb{1}_{[t>t^*]} + \epsilon_t \end{aligned}$$

Pierre P. propone errores ARMA, $\epsilon_t \sim ARMA(p, q)$ en sus planteamientos de hipótesis.

Hace uso de prueba tipo Dickey-Fuller aumentada de raíces unitarias para construir el estadístico de los tres modelos anteriores.
Tomando $\theta = (a_2 - a_1)$, $\gamma = (b_2 - b_1)$ y $\hat{\epsilon}_t = \hat{Y}_t - \hat{a} - \hat{b}_1 t - \hat{\gamma}(t - t^*)\mathbb{1}_{[t>t^*]}$

a) $\nabla X_t = a + \theta\mathbb{1}_{[t>t^*]} + bt + d\mathbb{1}_{[t=t^*+1]} + c_0 X_{t-1} + \sum_{i=1}^{k} c_i \nabla X_{t-i} + \epsilon_t$

b) $Y_t = a + b_1 t + \gamma(t - t^*)\mathbb{1}_{[t>t^*]} + \epsilon_t$
$\nabla\hat{\epsilon}_t = c_0\hat{\epsilon}_{t-1} + \sum_{i=1}^{k} c_i \nabla\hat{\epsilon}_{t-i} + \eta_t$

c) $\nabla X_t = a + \theta \mathbb{1}_{[t>t^*]} + bt + \gamma(t-t^*) + d\mathbb{1}_{[t=t^*+1]} + ...$
$... + c_0 X_{t-1} + \sum_{i=1}^{k} c_i \nabla X_{t-i} + \epsilon_t$

Una vez que tenemos el proceso $\{X_t\}$ *dividido en una media, una tendencia lineal, funciones indicadoras y una combinación lineal de diferencias de términos rezagados de* X_t *como en a) y c), podemos probar la significancia de los coeficientes por mínimos cuadrados y de esta manera tomar la decisión sobre la hipótesis nula.*
En b) la construcción es un poco distinta, primero se realiza una regresión de $Y_t = a - b_t - \gamma(t-t^*)\mathbb{1}_{[t>t^*]} + \epsilon_t$, *de aquí se estiman los errores* $\hat{\epsilon}_t$ *y sobre estos se hace la prueba de Dickey Fuller. Se aplica un procedimiento distinto a a) y c) para que no haya problemas para construir la distribución límite asintótica del estadístico de prueba* $\tau_b(\lambda)$. *La construcción de las tres distribuciones límite para cada uno de los casos* $\tau_i(\lambda)$ *para* $i = a, b, c$, *rebasan el nivel de este trabajo. Se puede encontrar la expresión de las distribuciones [10] y su construcción*
El número natural k *se determina de la misma manera que se hizo en ADF. Y el estadístico de prueba* $\tau_i(\lambda)$ *para* $i = a, b, c$ *corresponde a*

$$\tau_i(\lambda) = \frac{\hat{c_0}(\lambda)}{sd(\hat{c_0}(\lambda))} \quad para\ i = a, b, c \tag{A.10}$$

$$\lambda = t^*/n$$

A diferencia de la **Ecuación A.9** *el estadístico* $\tau_i(\lambda)$ *para* $i = a, b, c$ *depende de* λ *del momento del punto de cambio estructural; se hace este énfasis porque la distribución límite es distinta para cada* λ. *Pierre P. encontró las distribuciones límites bajo la hipótesis nula para a),b) y c) y tabuló los valores críticos para una secuencia de* $\lambda \in (0, 1)$.

Entonces la prueba para A, B y C se reduce a

A) $H_0 : \theta = 0,\ b = 0\ y\ c_0 = 0\ vs$
$H_1 : \theta \neq 0\ b \neq 0\ y\ c_0 < 0$

B) $H_0 : b_1 = 0,\ \gamma = 0\ y\ c_0 = 0\ vs$
$H_1 : b_1 \neq 0,\ \gamma \neq 0\ y\ c_0 < 0$

C) $H_0 : \theta = 0,\ b = 0\ ,\ \gamma = 0\ y\ c_0 = 0\ vs$
$H_1 : \theta \neq 0,\ b \neq 0\ ,\ \gamma \neq 0\ y\ c_0 < 0$

Notemos que haciendo esta modificación para b) solo se permite el cambio estructural bajo la hipótesis alternativa.
El trabajo de Pierre P. es criticada por dos razones principales, la primera es que la

prueba raramente rechaza la hipótesis nula, por lo que tiene una potencia baja. La segunda razón es que la prueba puede estar sesgada por las ideas o creencias de quien vaya a aplicar la prueba tenga de antemano, pues se determina antes de aplicar la prueba el punto donde se cree que hay cambio estructural.

A.2.2. Prueba Zivot-Andrews

Zivot y Andrews en su artículo publicado en 1992 [15], critican la prueba de Pierre P., principalmente el supuesto de cambios estructurales determinados de manera exógena. Z. y A. proponen determinar el momento de cambio estructural de manera endógena, tomando el ínfimo sobre λ de los estadísticos tipo t de Pierre P. Se realizan las tres mismas pruebas de hipótesis de cambio en media, tendencia o ambos, con la pequeña diferencia de solo permitir cambio estructural en la hipótesis alternativa. Las distribuciones asintóticas de los estadísticos de prueba son diferentes, la región de rechazo es más negativa que Perron. La construcción de los estadísticos, de la distribución, y tablas de los valores se pueden encontrar en [15]. Esta fue la prueba con la cual se determinan los quiebres estructurales en media y tendencia para las series de tiempo del producto de interés en la aplicación.

Apéndice B

Métodos avanzados para el aprendizaje estadístico

B.1. Descomposición sesgo-varianza en un problema de clasificación binario

"Aunque se han hecho varios esfuerzos por proponer descomposiciones basadas en el error general de la función de pérdida de clasificación $\mathbb{E}_{\mathcal{L}}\left[\mathbb{E}_{Y|X=x}[\mathbb{1}(Y=\varphi_{\mathcal{L}}(x))]\right]$ *redefiniendo los conceptos de sesgo varianza, ninguno ha logrado un esquema simple y satisfactorio como en regresión" [7].*

Todos modelos que se estudian en este trabajo, además de estimar un clasificador $\varphi(x)$*, estiman las probabilidades de cada clase* $\hat{\mathbb{P}}_{\mathcal{L}}(Y=c|X=x)$*, de cuales se obtiene el clasificador como:* $\varphi_{\mathcal{L}}(x)=\arg\max_c \hat{\mathbb{P}}_{\mathcal{L}}(Y=c|X=x)$*. Siguiendo el desarrollo presentado en el capítulo 4 de [7], se descompone el error general en una parte irreducible, y en otra que depende del clasificador* $\varphi_{\mathcal{L}}(x)$*. (Todas las probabilidades se calculan sobre la distribución de la v.a.* Y *condicionada a* $X=x$*, se omite la notación para hacer más sencilla la lectura). Supongamos que el problema es de clasificación binario.*

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{L}}\left[\mathbb{E}_{Y|X=x}[\mathbb{1}(Y=\varphi_{\mathcal{L}}(x))]\right] \\
&\quad = \mathbb{P}_{\mathcal{L}}(\varphi_{\mathcal{L}}(x)\neq Y) \\
&\quad = 1-\mathbb{P}_{\mathcal{L}}(\varphi_{\mathcal{L}}(x)= Y) \\
&\quad = 1-\mathbb{P}_{\mathcal{L}}(\varphi_{\mathcal{L}}(x)=\varphi_B(x))\mathbb{P}(\varphi_B(x)=Y)-\mathbb{P}_{\mathcal{L}}(\varphi_{\mathcal{L}}(x)\neq\varphi_B(x))\mathbb{P}(\varphi_B(x)\neq Y) \\
&\quad = \mathbb{P}(\varphi_B(x)\neq Y)+\mathbb{P}(\varphi_B(x)\neq Y)\mathbb{P}_{\mathcal{L}}(\varphi_{\mathcal{L}}(x)\neq\varphi_B(x))-\mathbb{P}_{\mathcal{L}}(\varphi_{\mathcal{L}}(x)\neq\varphi_B(x))\mathbb{P}(\varphi_B(x)\neq Y) \\
&\quad = \mathbb{P}(\varphi_B(x)\neq Y)+\mathbb{P}_{\mathcal{L}}(\varphi_{\mathcal{L}}(x)\neq\varphi_B(x))\left(2\mathbb{P}(\varphi_B(x)=Y)-1\right)
\end{aligned}
$$

El primer término es el error irreducible de Bayes, y el segundo término es el error cometido por el modelo ajustado, en comparación con Bayes. Dado que el clasificador asigna a la clase de mayor probabilidad, y se tiene un problema de clasificación binario, se tiene la igualdad: $\mathbb{P}_{\mathcal{L}}(\varphi_{\mathcal{L}}(x) \neq \varphi_B(x)) = \mathbb{P}_{\mathcal{L}}(\hat{\mathbb{P}}_{\mathcal{L}}(Y = \varphi_B) < 0.5)$.

Si la probabilidad estimada de cada clase sigue una distribución normal de media $\mu_{\hat{p}} = \mathbb{E}_{\mathcal{L}}[\hat{\mathbb{P}}_{\mathcal{L}}(Y = \varphi_B)]$ *y varianza* $\sigma^2_{\hat{p}} = Var_{\mathcal{L}}(\hat{\mathbb{P}}_{\mathcal{L}}(Y = \varphi_B))$ *entonces*

$$\mathbb{P}_{\mathcal{L}}(\hat{\mathbb{P}}_{\mathcal{L}}(Y = \varphi_B) < 0.5) = \Phi\left(\frac{1/2 - \mu_{\hat{p}}}{\sigma_{\hat{p}}}\right)$$

- *Si* $\mu_{\hat{p}} > 1/2$ *y la varianza de la predicción tiende a cero* $\sigma_{\hat{p}} \to 0$ *entonces* $\Phi\left(\frac{1/2-\mu_{\hat{p}}}{\sigma_{\hat{p}}}\right) \to 0$ *y el error general tiende al error de Bayes*
- *Por otro lado Si* $\mu_{\hat{p}} < 1/2$ *y la varianza de la predicción tiende a cero* $\sigma_{\hat{p}} \to 0$ *entonces* $\Phi\left(\frac{1/2-\mu_{\hat{p}}}{\sigma_{\hat{p}}}\right) \to 1$ *el error general es máximo.*

Por esta razón buscamos modelos que sean "buenos" ($\mu_{\hat{p}} > 1/2$) *, que mantengan el sesgo y que logren reducir la varianza. En consecuencia estos modelos reducen el error general y mejoran el desempeño. Dado que los modelos de ensamble reducen el sesgo, varianza o ambos es natural que logren tan buenos resultados.*

B.2. Descomposición del error general sobre regiones disjuntas

Siguiendo las notas de David Nualart [9], dado un espacio de probabilidad $(\Omega, \mathcal{F}, \mathbb{P})$ *se define la probabilidad condicional de un evento* A *dado otro evento* B *tal que* $\mathbb{P}(B) > 0$ *como*

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)} \tag{B.1}$$

El mapeo $A \mapsto \mathbb{P}(A|B)$ *define una nueva medida de probabilidad en la* σ*-álgebra* $\mathcal{F}$*. La esperanza de una variable aleatoria* X *respecto a la medida de probabilidad condicional se define como esperanza condicional de* X *dado un evento* B*, y se puede calcular como:*

$$\mathbb{E}[X|B] = \frac{1}{\mathbb{P}(B)}\mathbb{E}[X\mathbb{1}_B] \tag{B.2}$$

De este resultado, si se tiene un partición del espacio muestral $\{R_t\}_{t=1}^T$ *tal que* $R_t \cap R_s = \emptyset$ *si* $s \neq t$*. Entonces como* $X = \sum_{t=1}^T X\mathbb{1}_{R_t}$ *se obtiene* $\mathbb{E}[X] = \sum_{t=1}^T \mathbb{E}[X\mathbb{1}_{R_t}]$*. Aplicando*

el resultado B.2 sobre cada elemento de la suma obtenemos:

$$\mathbb{E}[X] = \sum_{t=1}^{T} \mathbb{P}(R_t)\mathbb{E}[X|R_t] \tag{B.3}$$

Por este resultado se puede descomponer el error general como se muestra en la ecuación 3.2.

Bibliografía

[1] *Leo Breiman. Random forests.* Machine Learning, *45(1):5–32, 2001. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.*

[2] *Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman.* The elements of statistical learning: data mining, inference, and prediction, 2nd Edition. *Springer series in statistics. Springer, 2009. ISBN 9780387848570.*

[3] *UKevin S. Van Horn. Which link function—logit, probit, or cloglog?* Bayesium Analytics, *August 2015. URL hhttp://bayesium.com/.*

[4] *Momo (https://stats.stackexchange.com/users/8413/momo). What is the difference between a link function and a canonical link function for glm? Mathematics. URL https://stats.stackexchange.com/questions/40876/what-is-the-difference-between-a-link-function-and-a-canonical-link-function. Accesed: 2020-01-23.*

[5] *Lina Huang. How do i calculate the time complexity of a decision tree (machine learning algorithm)? Mathematics. URL https://www.quora.com/How-do-I-calculate-the-time-complexity-of-a-decision-tree-machine-learning-algor Accesed: 2020-02-04.*

[6] *Sotiris Kotsiantis, D. Kanellopoulos, and P. Pintelas. Handling imbalanced datasets: A review.* GESTS International Transactions on Computer Science and Engineering, *30:25–36, 11 2005.*

[7] *Gilles Louppe.* Understanding Random Forests from Theory to Practice. *PhD thesis, University of Liège, 2014.*

[8] *P. McCullagh and J. A. Nelder.* Generalized Linear Models. *Chapman amd Hall / CRC, London, 1989.*

[9] *David Nualart. Stochastic processes lecture notes.*

[10] Pierre Perron. The great crash, the oil price shock, and the unit root hypothesis. Econometrica*, 57(6):1361–1401, 1989. ISSN 00129682, 14680262. URL* http://www.jstor.org/stable/1913712*.*

[11] Raúl Rojas. Adaboost and the super bowl of classifiers a tutorial introduction to adaptive boosting. 2009.

[12] SAID E. SAID and DAVID A. DICKEY. Testing for unit roots in autoregressive-moving average models of unknown order. Biometrika*, 71(3):599–607, 12 1984. ISSN 0006-3444. doi: 10.1093/biomet/71.3.599. URL* https://doi.org/10.1093/biomet/71.3.599*.*

[13] Pranab K. Sen, Julio M. Singer, and Antonio C. Pedroso de Lima. Regression Models*, page 293–337. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2009. doi: 10.1017/CBO9780511806957.011.*

[14] Sarang Zambare. Detecting wake-words in speech. https://mc.ai/detecting-wake-words-in-speech/*, february 2019. Accessed: 2019-09-16.*

[15] Eric Zivot and Donald W. K. Andrews. Further evidence on the great crash, the oil-price shock, and the unit-root hypothesis. Journal of Business and Economic Statistics*, 10(3):251–270, 1992. ISSN 07350015. URL* http://www.jstor.org/stable/1391541*.*