Trabajo Fin de Máster

Modelos Predictivos: Ciclo de Vida Familiar

Lidia López Fernández

Máster en Técnicas Estadísticas

Curso 2022-2023

Propuesta de Trabajo Fin de Máster

Título en galego: Modelos Preditivos: Ciclo de Vida Familiar
Título en español: Modelos Predictivos: Ciclo de Vida Familiar
English title: Predictive Models: Family Life Cycle
Modalidad: Modalidad B
Autora: Lidia López Fernández, Universidad de Santiago de Compostela
Directores: Javier Tarrío Saavedra, Universidad de A Coruña; Salvador Naya Fernández, Universidad de A Coruña
Tutor: Juan Manuel Mazaira Gómez, ABANCA
Breve resumen del trabajo: El objetivo del trabajo fin de máster se podrá definir de acuerdo a su perfil enmarcándose siempre dentro de las líneas de trabajo del área de Modelos Predictivos de 'Inteligencia de Clientes'. Se realizará, entre otras, la tarea de: -Construir un modelo predictivo que permita clasificar a los clientes en las diferentes etapas de su ciclo de vida familiar, utilizando para ello su información sociodemográfica y su información transaccional con el banco. A partir del CVF se inferirán diferentes características del núcleo familiar como la capacidad económica.

Don Javier Tarrío Saavedra, Profesor Titular de la Universidad de A Coruña, don Salvador Naya Fernández, Catedrático de la Universidad de A Coruña, y don Juan Manuel Mazaira Gómez, Manager de ABANCA, informan que el Trabajo Fin de Máster titulado

Modelos Predictivos: Ciclo de Vida Familiar

fue realizado bajo su dirección por doña Lidia López Fernández para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal.

En A Coruña, a 5 de junio de 2023.

El director:
Don Javier Tarrío Saavedra

El director:
Don Salvador Naya Fernández

El tutor:
Don Juan Manuel Mazaira Gómez

La autora:
Doña Lidia López Fernández

Declaración responsable. Para dar cumplimiento a la Ley 3/2022, de 24 de febrero, de convivencia universitaria, referente al plagio en el Trabajo Fin de Máster (Artículo 11, Disposición 2978 del BOE núm. 48 de 2022), **el/la autor/a declara** que el Trabajo Fin de Máster presentado es un documento original en el que se han tenido en cuenta las siguientes consideraciones relativas al uso de material de apoyo desarrollado por otros/as autores/as:

- Todas las fuentes usadas para la elaboración de este trabajo han sido citadas convenientemente (libros, artículos, apuntes de profesorado, páginas web, programas,...)
- Cualquier contenido copiado o traducido textualmente se ha puesto entre comillas, citando su procedencia.
- Se ha hecho constar explícitamente cuando un capítulo, sección, demostración,...sea una adaptación casi literal de alguna fuente existente.

Y, acepta que, si se demostrara lo contrario, se le apliquen las medidas disciplinarias que correspondan.

Agradecimientos

A través de este texto me gustaría expresar mi agradecimiento a todas aquellas personas que han hecho este trabajo posible. En primer lugar, agradecer a Javier Tarrío Saavedra y Salvador Naya Fernández de la Universidad de A Coruña por sus continuas recomendaciones y ánimos durante la realización de esta memoria. Por otro lado, agradecer también a todo el equipo de Modelos Predictivos y Prospección de ABANCA, quienes estuvieron siempre dispuestos a ayudarme a través de este proceso. En especial, a mi tutor en la empresa Juan Manuel Mazaira Gómez por su ayuda y orientación continuos y a Jose Piñeiro Abal, por su incondicional apoyo y conocimientos.

Por último, dar las gracias a mis padres, amigos y Javi, quienes me han apoyado durante todo este proceso, de principio a fin.

Índice general

Resumen

Resumen en español

El ciclo de vida familiar es un modelo teórico que describe las diferentes etapas que una familia normalmente atraviesa a lo largo de su vida, asociadas a cambios en la composición del núcleo familiar y las relaciones entre los miembros. Por tanto, también cambiarán las necesidades financieras de la familia a lo largo de las etapas. El objetivo de este trabajo será construir un modelo mediante técnicas de aprendizaje estadístico que nos permita determinar a que etapa del ciclo de vida familiar pertenece cada cliente, para así poder ofrecerle los productos financieros que más se adapten a sus necesidades.

Para ello, se debe recopilar la información que nos permita determinar esta característica de todos los clientes posibles, y a través de ellos y del modelo extender esta determinación al resto de clientes presentes en la entidad, para así poseer la habilidad de clasificar a todos los clientes de la entidad, incluyendo las nuevas altas.

English abstract

The family life cycle is a theoretical model that describes the different stages that a family normally goes through throughout its life, associated with changes in the composition of the family nucleus and the relationships between members. Therefore, the financial needs of the family will also change throughout the stages. The objective of this memory will be to build a model using statistical learning techniques that will allows us to determine which stage of the family life cycle each client belongs to, in order to offer them the financial products that best suit their needs.

For this purpose, the information that allows us to determine this characteristic of all possible clients must be collected, and through it and the model, extend this determination to the rest of the clients present in the entity, in order to have the ability to classify all clients of the entity, including new clients.

Parte I

Antecedentes

Capítulo 1

Introducción al problema

ABANCA *Corporación Bancaria S.A* es una entidad financiera compuesta por varias áreas con distintas funciones. En concreto, el análisis de datos es una parte crucial de ésta que constituye una ayuda fundamental, sobre todo para fines comerciales.
La principal encargada de análisis de datos en ABANCA es el área de Inteligencia de Clientes, conformada por cuatro departamentos: Modelos Predictivos y Propensión, Analítica Avanzada, Investigación Comercial y, finalmente, Gestión de las Relaciones con Clientes (CRM, si tenemos en cuenta sus siglas en inglés).

En la actualidad, las principales funciones y objetivos del departamento de Modelos Predictivos y Propensión son los siguientes:

- Dar soporte a las diferentes unidades de negocio (Empresas, Negocios, Particulares, Personal, Seguros...) y resolver casos de uso, empleando para ello técnicas de analítica avanzada (algoritmos de Machine Learning e Inteligencia Artificial).

- Asegurar la provisión de leads para el cumplimento de los objetivos comerciales del banco y el operador de banca seguros, mediante técnicas de analítica avanzada.

- Contribuir a la eficacia y la eficiencia de las acciones comerciales proactivas con una adecuada selección del target.

- Arquetipos: Identificar factores clave del éxito de estas acciones para orientar la oferta y la argumentación comercial en función de diferentes perfiles con propensión detectados.

- Seleccionar la secuencia de contactos y canales óptima en el momento más adecuado para maximizar la conversión con el mínimo coste y mayor retorno.

Para ello, se lleva a cabo la construcción de diferentes modelos predictivos, a través de la técnica estadística con mejor desempeño según el caso, entre las que se encuentran algunas como árboles de decisión y técnicas de regresión. Entre los actuales modelos empleados en el departamento de Modelos Predictivos y Prospección, se mencionan a continuación los siguientes:

Por un lado, modelos de familias, los cuales se aplicarán a clientes personas físicas de la entidad. Estos modelos tienen como objetivo prever el comportamiento de los clientes en relación a la contratación de productos financieros objetivo de cada modelo, teniendo en cuenta que los clientes que se analizan no tienen este producto contratado, y así poder ofrecer un catálogo de productos que se ajuste a sus necesidades. Pertenecen a esta clase de modelos los siguientes:

- **Ahorro e inversión.**
  Se utiliza el comportamiento pasado de los clientes para poder predecir el comportamiento futuro, lo que permite a la entidad anticiparse a la necesidades de sus clientes y ofrecerles productos de inversión que se ajusten a su situación financiera presente y futura.

- **Consumo y tarjetas.**
  Se estima la probabilidad de que los clientes contraten préstamos al consumo o tarjetas de crédito. De esta manera, se pueden identificar oportunidades de negocio y mejorar la oferta actual.

- **Seguros.**
  Se enfocan en estimar la probabilidad de contratación de diferentes tipos de seguros, incluyendo seguros de hogar, salud, auto, decesos y vida riesgo. Permite entonces mejorar la oferta de seguros, más ajustada a las necesidades del cliente.

- **Planes de pensiones.**
  Se estima la probabilidad de contratación y aportaciones a este tipo de productos financieros, incluyendo altas, reactivados y aportaciones extraordinarias.

- **Abandono.**
  Se estima la probabilidad de abandono por parte de los clientes de la entidad, permitiendo identificar a los clientes insatisfechos y tomar medidas para retenerlos.

Por otro lado, se tienen los siguientes modelos, definiendo al cliente también como empresa:

- **Modelos empresas.**
  Orientados a personas jurídicas, permiten detectar clientes con necesidades de financiación que no las obtengan actualmente en la entidad bancaria. El principal objetivo es la captación de nuevos clientes, identificando los que tengan un mayor atractivo comercial, es decir, mayor rentabilidad, vinculación y solvencia.

- **Aldia Empresas.**
  Se utilizan técnicas de Web Scraping para recoger información online y transformarla en eventos comerciales, los cuales se distribuyen a los gestores comerciales y así realizar acciones comerciales proactivas a clientes potenciales. Por tanto, se tiene como objetivo detectar oportunidades comerciales en tiempo real a través de medios digitales.

- **Redes empresas.**
  Se tiene como objetivo construir una red de relaciones entre empresas, la cual permita identificar oportunidades comerciales como captación o financiación especializada.

- **Clientes valor.**
  Se enfoca en construir una matriz de afinidad de clientes que permita a la entidad realizar procesos como la diferenciación de grupos de clientes, el cálculo de la penetración de cada producto y la definición de la afinidad con cada uno de ellos.

1.1. Motivación

El departamento de Modelos Predictivos y Propensión está trabajando continuamente en la creación de nuevos modelos y la mejora de aquellos ya existentes.
Surge entonces la necesidad de aportar nuevas variables explicativas a estos modelos, que nos ayuden a mejorar la capacidad de predicción.

En este trabajo, nos centraremos en aquellos modelos orientados a clientes personas físicas, aportando una nueva variable explicativa que nos indique a que etapa del Ciclo de Vida Familiar pertenece

cada cliente, con la intención de apreciar con ello una mejoría en la precisión de los modelos.

El ciclo de vida familiar es un modelo teórico que describe las diferentes etapas que una familia normalmente atraviesa a lo largo de su vida. Estas etapas son progresivas, asociadas a cambios en la composición del núcleo familiar y, también, en las relaciones entre sus miembros.

Este concepto fue introducido por primera vez por Paul Glick en el 1947 [1], pero los enfoques más populares y más utilizados en la actualidad son el aportado por Wells y Gubar en 1966 [2] y el aportado por Duval en 1988 [3]. Los dos constan de 8 etapas, pero mientras que el primero se centra en la presencia de los hijos en el hogar y su edad, el segundo presta más atención al estadio evolutivo en el que se encuentran los hijos. Podemos ver las etapas de los dos modelos en la Tabla 1.1.

Wells y Gubar (1966)
Soltería.
Pareja recién casada.
Nido lleno I: Parejas jóvenes con niños menores de 6 años.
Nido lleno II: Parejas jóvenes con niños de más de 6 años.
Nido lleno III: Parejas mayores con hijos dependientes.
Nido vacío I: Ya no hay hijos en el hogar. El o la cabeza de familia sigue trabajando.
Nido vacío II: La persona cabeza de familia se ha retirado/jubilado.
Sobreviviente solitario: Trabajando o retirado.
Duval (1988)
Parejas casadas sin hijos.
Familias en la crianza inicial: Primogénito/a de menos de 30 meses.
Familias con niños en edad preescolar: Primogénito/a entre 2,5 y 6 años.
Familias con niños en edad escolar: Primogénito/a entre 6 y 13 años.
Familias con adolescentes: Primogénito/a entre 13 y 20 años.
Familias como plataforma de lanzamiento:Abandono de todos los hijos del hogar.
Padres de mediana edad: Desde el nido vacío hasta el retiro laboral.
Familia anciana: Desde el retiro laboral hasta el fallecimiento de ambos miembros de la pareja.

Tabla 1.1: Ciclo de vida familiar según diferentes autores.

1.1.1. Ciclo de vida familiar

El ciclo de vida familiar también puede ser analizado desde una perspectiva económica. En general, se considera que los ingresos y gastos de una familia evolucionan a lo largo de su vida. Es este al punto al que queremos llegar, ya que dependiendo de la etapa del ciclo de vida familiar en que se encuentre una persona, tendrá necesidades financieras distintas.

Es aquí donde comienza nuestro interés por el tema, debido a que, como entidad bancaria, es de gran ayuda saber a que clientes debemos ofrecer cada producto y servicio.
Por esta misma razón, tomaremos un ciclo de vida familiar poniendo especial atención al componente económico de cada etapa, y su adaptación a la sociedad actual, ya que los modelos anteriormente explicados en la Tabla 1.1 fueron construidos para un modelo de sociedad distinto al actual. Para ello, tomamos como referencia el modelo de ciclo de vida familiar descrito en el artículo 'Conceptualización de ciclo vital familiar: Una mirada a la producción durante el periodo comprendido entre los años 2002 a 2015' escrito por Moratto et al. [4]. Este se construirá a partir de la base de datos descrita en el Capítulo 3, estando compuesto por las siguientes etapas:

- **Single:** Aquí encontramos a las personas solteras, que no forman parte de un núcleo familiar. En esta etapa, la atención se halla en el trabajo y la diversión. También se tiene muy en cuenta la forma de generar ingresos, ya que se aspira al desarrollo profesional, laboral y social. Se podrá apreciar entonces un gasto alto en moda, ocio y estéticos.
- **Parejas sin hijos:** Se tratan de personas jóvenes, en los cuales no se encuentra un nivel de ingresos alto, pero si adecuado, ya que normalmente las dos personas trabajan. Es común que estén ahorrando para la compra de una vivienda y otros gastos asociados con el matrimonio y la formación de una familia. También pueden estar invirtiendo en educación y formación para mejorar sus perspectivas laborales y así aumentar sus ingresos en el futuro.
- **Parejas con hijos pequeños (0-6 años):** En esta etapa el gasto aumenta significativamente, debido al coste asociado con el nacimiento de un niño: atención médica, pañales, alimentos y ropa. Es común que, aparte de la baja de maternidad y/o paternidad, uno o ambos padres reduzcan sus horas de trabajo o se tomen un tiempo libre para así poder cuidar a los hijos, lo que puede afectar a los ingresos de la familia.
- **Parejas con hijos en edad escolar (6-18 años):** En la etapa escolar, los gastos pueden disminuir en comparación con la etapa anterior, pero es posible que los padres deban invertir en la educación y actividades extracurriculares de sus hijos. A medida que los hijos crecen y se vuelven más independientes, es probable que los ingresos de la familia aumenten según los padres vuelven a trabajar más horas o retoman sus carreras.
- **Parejas con hijos mayores de edad:** En esta etapa, los gastos vuelven a aumentar debido al coste de la educación superior o de ayudar al hijo con ciertos gastos propios para así ayudarlo en la formación de su posterior vida adulta independiente, como por ejemplo en la compra de un coche.
- **Pareja con hijos independizados (nido vacío):** Esta etapa se da en el momento en el que todos los hijos abandonan el hogar familiar, siendo ya independiente económicamente de este. Por tanto, podrán destinar más dinero para el disfrute común, invirtiendo en ocio, viajes y vacaciones. Algunos de ellos ya estarán jubilados, por lo que a parte de tener suficiente patrimonio, también dispondrán de tiempo para gastarlo.
- **Viudo/a:** Se produce cuando una de las dos personas de la pareja fallece. Normalmente, son personas de edad avanzada y por tanto jubiladas, cobrando las pensiones correspondientes. Al ser personas solitarias, es común la presencia de una mascota, invirtiendo dinero en su bienestar. Debido a la edad, aumentará también el gasto en productos de salud preventiva y curativa, y todo tipo de seguros médicos.

1.2. Selección de los datos

Para este trabajo utilizaremos una base de datos de entrenamiento proporcionada por ABANCA, almacenada en la tabla de SQL *mktg_usr.tabla_datos_entrenamiento*, en la que podemos ver más de 28 millones de observaciones reales correspondientes a más de 1500000 clientes pertenecientes a la entidad, definidos por más de 300 variables. Entre ellas, podemos encontrar las siguientes:

- **CLIENTE_ID:** Número de identificación del cliente sobre el que se toma la observación.
- **FECHA:** Fecha en la que fue tomada la observación. En el caso de este trabajo, la fecha será el 31 de enero de 2023.
- **EDAD:** Edad del cliente en años a fecha de observación.
- **AMBITO_CLIENTE:** Ámbito del cliente en función del padrón municipal de 2017. Toma los valores: 1 (Rural), 2 (Semiurbano), 3 (Ciudad), 4 (Gran ciudad), 5 (Área metropolitana), 6 (Desconocido).
- **IMP_NOMINA_SUA:** Media de los importes de nómina e INEM de los últimos cuatro meses.
- **IMP_PENSION_SUA:** Media de los importes de pensión de los últimos cuatro meses.
- **IMP_TRANSFERENCIAS_SUA:** Media de los importes de transferencias de los últimos cuatro meses.
- **VINCULACION_TRANS_CT:** Variable vinculación transaccional del cliente como cualquier titular (indica la recurrencia con la que el cliente movimientos en los productos financieros que tiene contratados en la entidad).
- **VINCULACION_NEG_CT:** Variable vinculación de negocio del cliente como cualquier titular (indica la tenencia de de los distintos productos financieros por parte del cliente en la entidad).
- **MI_CLIENTE_PT_12M:** Suma de los márgenes de intermediación generados por el cliente como primer titular en los últimos 12 meses anteriores a la fecha de observación.
- **MO_CLIENTE_PT_12M:** Suma de los márgenes ordinarios generados por el cliente como primer titular en los últimos 12 meses anteriores a la fecha de observación.
- **IMP_RECIBOS_AGUA_4M_CT:** Media de los importes de recibos de agua en los últimos cuatro meses.
- **IMP_RECIBOS_ALQUILER_4M_CT:** Media de los importes de recibos de alquiler en los últimos cuatro meses.

- **IMP_RECIBOS_COLEGIOS_4M_CT:** Media de los importes de colegios en los últimos cuatro meses.

- **IMP_RECIBOS_COMUNIDAD_4M_CT:** Media de los importes de recibos de comunidades en los últimos cuatro meses.

- **IMP_RECIBOS_ENERGIA_4M_CT:** Media de los importes de recibos de energía en los últimos cuatro meses.

- **IMP_RECIBOS_TELEFONO_4M_CT:** Media de los importes de recibos de teléfono en los últimos cuatro meses.

- **IMP_RECIBOS_SM_4M_CT:** Media de los importes de recibos de seguros médicos u otros en los últimos cuatro meses.

- **VISTA_DMK_CT:** Saldo medio vista en el último mes en los contratos como cualquier titular.

- **PLAZO_DMK_CT:** Suma de los saldos medios en el último mes de imposiciones a plazo fijo (a corto y largo plazo) en los contratos como cualquier titular.

- **ACTIVO_DMK_CT:** Saldo medio de activo en el último mes en los contratos como cualquier titular.

- **PASIVO_DMK_CT:** Saldo medio de pasivo en el último mes en los contratos como cualquier titular.

- **N_SEG_HOGAR:** Variable número de seguros de hogar en la fecha de observación tanto libres como vinculados.

- **N_SEG_VIDA:** Variable Número de seguros de vida libres en la fecha de observación.

- **N_SEG_VIDA_V:** Variable Número de seguros de vida vinculados en la fecha de observación.

- **TJCREDITO:** Variable marca que nos indica la tenencia del servicio tarjeta de crédito o no.

- **TJDEBITO:** Variable marca que nos indica la tenencia del servicio tarjeta de débito o no.

- **IN_DIGITAL:** Grado de digitalidad del cliente, toma los valores: nulos, 0 (no digital) , 1 (digital, tiene banca móvil/electrónica), 2 (digital activo, solo consulta), 3 (digital transaccional).

- **AUTORIZADO_BE:** Suma de los saldos dispuestos y disponibles del cliente en el Banco de España.

- **AUTORIZADO_AB:** Suma de los saldos dispuestos y disponibles del cliente en la entidad.

- **HIPOTECA_PENDIENTE_CT:** Saldo final prestamo garantia real como cualquier titular.

- **GASTO_TPV_CREDITO:** Diferencia entre las imposiciones y abonos en tarjetas de crédito por TPV en el último año.

- **GASTO_TPV_DEBITO:** Diferencia entre las imposiciones y abonos en tarjetas de débito por TPV en el último año.

- **N_FONDOS_GARANTIZADOS_CT:** Número de contratos como cualquier titular de fondos conservadores.

Una vez extraídas las variables pertenecientes a este dataset, se añaden al conjunto de datos aquellas variables que aportan información sobre la tipología de gastos del cliente, las cuales se extraen de diferentes tablas de datos de la entidad, mediante un proceso elaborado y proporcionado por el departamento de Analítica Avanzada.
Entre estas tablas, se encuentran, por ejemplo, las denominadas *VIEWSINQ.TARJETA_PLASTICO_MOVIMIENTO*, *viewsinq.cliente_contrato_rl*, *MKTG_USR.PROD_ESTABLECIMIENTO* y *mktg_usr.aa_subtipo_comercio*.

La información contenida en estas variables será el gasto medio del cliente para cada tipo de gasto durante los últimos 12 meses, así definidos para evitar los efectos de la estacionalidad. Por lo cual, estas variables contendrán la información obtenida desde el 31 de enero de 2022 al 31 de enero del 2023. Se tienen entonces dos variables para cada tipo de gasto, una que contiene información sobre el gasto que realiza el cliente mediante la tarjeta de débito y otra referida a la tarjeta de crédito. Para este trabajo no nos interesa saber de que tipo de tarjeta proviene cada gasto, por lo que se crean variables que sean la suma de las dos variables que refieren cada tipo de gasto. Por tanto, resultan las siguientes variables:

- **GASTO_RESTAURANTES:** Gasto medio del cliente en bares y restaurantes durante los últimos 12 meses.

- **GASTO_SUPERMERCADO:** Gasto medio del cliente en supermercados durante los últimos 12 meses.

- **GASTO_VIAJE:** Gasto medio del cliente en viajes durante los últimos 12 meses.

- **GASTO_VEHICULO:** Gasto medio del cliente en autopistas, gasolineras y automóvil durante los últimos 12 meses.

- **GASTO_OCIO:** Gasto medio del cliente en ocio y entretenimiento durante los últimos 12 meses.

- **GASTO_ESTETICA:** Gasto medio del cliente en belleza y estética durante los últimos 12 meses.

- **GASTO_MODA:** Gasto medio del cliente en moda durante los últimos 12 meses.
- **GASTO_HOGAR:** Gasto medio del cliente en hogar durante los últimos 12 meses.
- **GASTO_VENTAONLINE:** Gasto medio del cliente en venta online durante los últimos 12 meses.
- **GASTO_MULTIMEDIA:** Gasto medio del cliente en contenido multimedia durante los últimos 12 meses.
- **GASTO_SALUD:** Gasto medio del cliente en farmacia, atención hospitalaria, óptica, ortopedia y audiología durante los últimos 12 meses.
- **GASTO_CULTURA:** Gasto medio del cliente en cultura durante los últimos 12 meses.
- **GASTO_DEPORTE:** Gasto medio del cliente en deporte durante los últimos 12 meses.
- **GASTO_CAJEROS:** Gasto medio del cliente en retirada de efectivo en cajeros durante los últimos 12 meses.
- **GASTO_JUGUETES:** Gasto medio del cliente en juguetes durante los últimos 12 meses.
- **GASTO_FUNERARIAS:** Gasto medio del cliente en funerarias durante los últimos 12 meses.
- **GASTO_SERVICIOS:** Gasto medio del cliente en servicios durante los últimos 12 meses.
- **GASTO_VARIOS:** Gasto medio del cliente en gastos varios durante los últimos 12 meses.
- **GASTO_GRANDESSUP:** Gasto medio del cliente en grandes superficies durante los últimos 12 meses.
- **GASTO_CONSTRUCCION:** Gasto medio del cliente en construcción durante los últimos 12 meses.
- **GASTO_SECTORPRIMARIO:** Gasto medio del cliente en el ámbito del sector primario durante los últimos 12 meses.
- **GASTO_FERRETERIA:** Gasto medio del cliente en ferreterías durante los últimos 12 meses.
- **IMPORTE_TOTAL:** Gasto total del cliente detectado por importes en cuenta durante los últimos 12 meses.

Una vez se extraen todas las variables necesarias, se almacenan en una única tabla denominada *mktg_usr.tablon_lidia_gastos_enero*, lo que permite, posteriormente, la fácil extracción de los datos mediante el uso de R.

Se parte entonces de un dataset con 368 variables, en el cual se procederá a hacer limpieza de datos.

El primer paso será eliminar los NA's que se encuentren en las variables. En este trabajo, se sustituirán todos los NA's por el valor 0, ya que, en las variables bancarias afectadas, el desconocimiento del valor de la variable suele indicar que el verdadero valor más probable es 0.

Este proceso se llevará a cabo en R, mediante el siguiente código:

```
1  convertir_na_numero<-function(x,valor){
2          x<-ifelse(is.na(x),valor,x)
3          return(x)
4  }
5
6  valores_nulos<-numeric(ncol(datos))
7
8  for (i in 1:ncol(datos)){
9          valores_nulos[i]<-length(which(is.na(datos[,i])==TRUE))
10 }
11
12 nulos<-valores_nulos[which(valores_nulos>=1)]
13 names(nulos)<-colnames(datos)[which(valores_nulos>=1)]
14
15 nulos
16
17
18 for (i in 1:length(which(valores_nulos>=1))) {
19         datos[,which(valores_nulos>=1)[i]]<-convertir_na_numero(datos[,which(
               valores_nulos>=1)[i]],0)
20 }
```

En él, se define la función **convertir_na_numero**, que convierte los Na's en el valor numérico que se le pida. Seguidamente, se define la variable *valores_nulos*, la cual guardará el número de valores NA's de cada variable del dataset. La variable *nulos* mostrará entonces las variables que poseen NA's y cuantos poseen. Si se piden que se muestre esa variable, una parte de la salida será:

```
        IN_DIGITAL                      AUTORIZADO_BE
        5632                            212262
        N_SEG_VID_V                     HIPOTECA_PENDIENTE_CT
        215269                          625
```

Por último, el bucle descrito en las líneas 18-20 del código reemplazará todos los NA's contenidos en el dataset por el valor 0, mediante la función **convertir_na_numero**, y se eliminarán aquellas variables que, una vez finalizado este paso, resulten en variables con más del 95 % de valores 0, ya que no aportarán información significativa.

Se procederá entonces a eliminar variables altamente correlacionadas. En este trabajo se considerará que dos variables están altamente correlacionadas si la correlación existente entre ellas es mayor que 0,9, siguiendo criterios estadísticos y las recomendaciones de la entidad.

Se realiza la detección de variables altamente correlacionadas mediante el siguiente código de R:

```
nums<-sapply(datos,is.numeric)
corr<-cor(datos[,nums])
#Se utiliza la función definida en el recuadro de código anterior:
corr<-convertir_na_numero(corr,0)

library(caret)

indcorr<-findCorrelation(corr,cutoff=0.9,names = T)
```

El primer paso es identificar las variables numéricas, que será entre las que se comprobará el nivel de correlación. Una vez hechas, se sutituyen aquellas en las que el valor obtenido fue NA por 0. Finalmente, mediante la función **findCorrelation** del paquete *caret* de R, identificamos aquellas con una correlación superior a 0,9. Se pide entonces que imprima *indcorr* por pantalla:

```
 [1] "PASIVO_DMK_PT"             "FB_DMK_CT"               "PASIVO_DMK_CT"
 [4] "INGRESOS_TOTALES"          "VINCULACION_NEG_PT"      "PLANES_DMK_PT"
 [7] "TARJERTAS_DMK_PT"          "ACTIVO_DMK_PT"           "N_PLANES_PT"
[10] "INGRESOS_TOTALES_NUCLEO"   "AUTORIZADO_AB"           "LIMITE_PH"
[13] "PRESTAMO_PENDIENTE_PT"     "NUM_SEG_VIDA_LIBRE"      "PRIMA_NETA_ANUAL_MULT_HOGAR_VINC"
[16] "IMPORTE_TOTAL_CRE"         "FECHA_ESTADO_CIVIL"      "INGRESOS_CLIENTE_REL_NUCLEO"
[19] "SCORE_RAW"                 "TRIAD_LIMITE_PP"         "CAPITAL_PENDIENTE_PTMOS_G"
[20] "N_SEG_VIDA_V"              "N_SEG_HOGAR_V"           "N_SEG_COCHE"
```

Se eliminan entonces un total de 23 variables por estar altamente correlacionadas con otras pertenecientes al dataset.

Se puede comprobar también el nivel de correlación entre variables mediante un *corrplot* o correlograma en R, opción que en este caso no es viable como única forma de comprobación ya que nuestro dataset tiene demasiadas variables como para que la información aportada por un gráfico de este tipo sea intuitiva e interpretable. En cambio, los correlogramas si se podrán utilizar si se coge un subconjunto de un tamaño relativamente pequeño del conjunto de datos, como podemos ver en la Figura 1.1.

Podemos ver entonces la alta correlación presente entre las variables *ACTIVO_DMK_PT* y *CAPITAL_PENDIENTE_PTMOS_GR*, entre *FB_DMK_CT* y *FONDOS_DMK_CT*, y entre *IMP_TRANSFERENCIAS_SUA* e *INGRESOS_TOTALES*.

Una vez finalizada la limpieza y el preprocesado de datos, el conjunto de datos resultante, el cual será usado en este trabajo, está compuesto por un total de 215 variables.

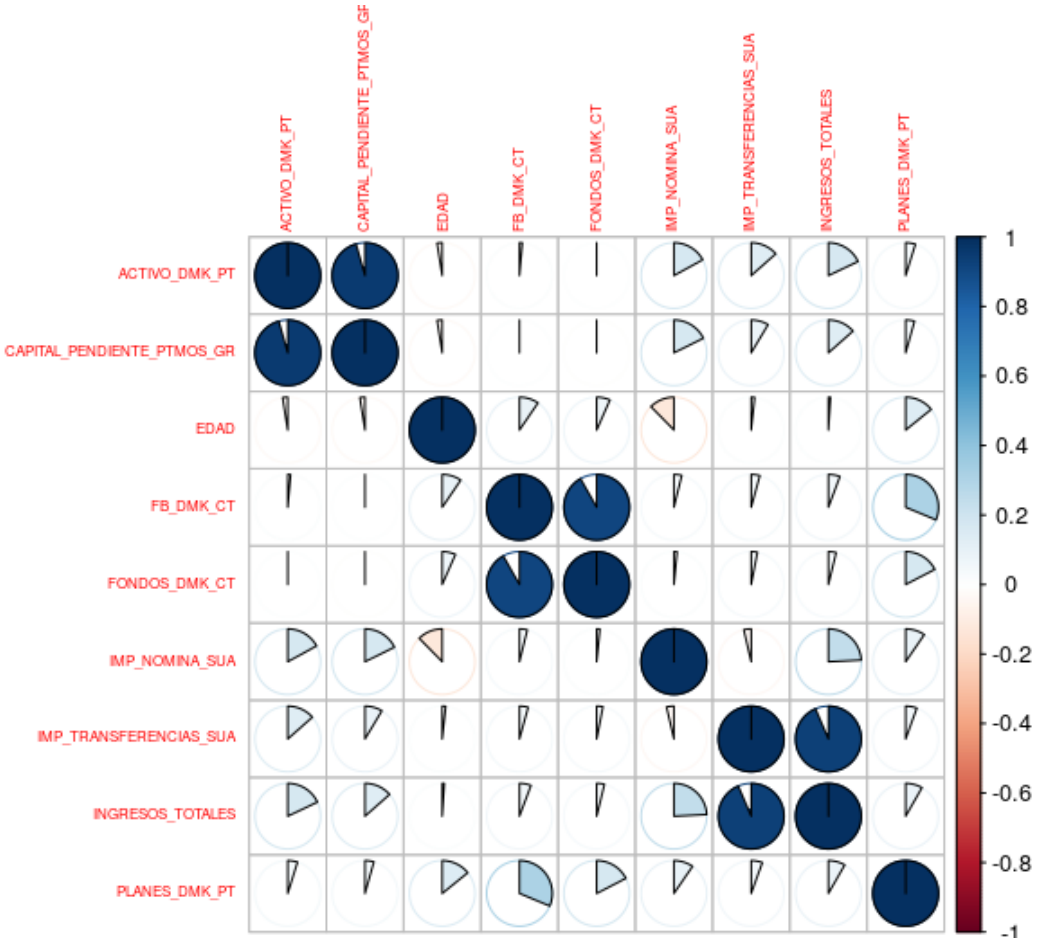


Figura 1.1: Gráfico de correlación para una selección de las variables del conjunto de datos utilizado.

Capítulo 2

Métodos de clustering

El *clustering*, análisis clúster o clasificación no supervisada es un conjunto de técnicas de aprendizaje estadístico no supervisado que consiste en dividir un conjunto de datos dado en grupos según su similaridad, los cuales serán llamados clusters.

Se suponen n objetos, datos u observaciones que necesitan ser agrupados y que pueden ser personas, flores, empresas, países, etc. Si se dispone de p variables predictoras, se puede representar el conjunto de datos a través de una matriz $n \times p$, definiendo cada uno de sus elementos como x_{ij} donde $i = 1, \ldots, n$ y $j = 1, \ldots, p$.

$$\begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n-1,1} & x_{n-1,2} & \cdots & x_{n-1,p} \\ x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{pmatrix} \tag{2.1}$$

La elección del método de *clustering* a utilizar depende tanto de la naturaleza del conjunto de datos como de el objetivo a alcanzar en cada ocasión particular. En este caso, como se ha especificado en la Sección 1.2, se dispone de un conjunto de datos con más de 1500000 clientes de la entidad y 216 variables, lo cual conforma una base de datos con más dimensiones de las habitualmente tratadas con método de *clustering*. Por ello, se utilizará un método diseñado específicamente para grandes conjuntos de datos.

2.1. Clustering Large Applications (CLARA)

El algoritmo CLARA se trata de un método de *clustering* particional, el cual establecerá las categorías en los elementos a partir de un número k de clusters previamente dado. En este tipo de métodos los k grupos deberán de cumplir las condiciones exigidas a una partición:

- Cada grupo debe contener al menos un objeto.
- Cada objeto debe de pertenecer exactamente a un grupo.

Estas condiciones implican que el número de grupos será, como máximo, igual al número de objetos, es decir, $k \leq n$. La segunda condición implica que dos grupos diferentes no pueden tener ningún objeto, en nuestro caso cliente, en común y que los k grupos juntos deben de formar exactamente el conjunto total.

La idea del algoritmo está basada en lo siguiente: Para obtener k clusters, el método selecciona k clientes, los cuales serán denominados representantes. Entonces, los correspondientes clusters son formados asignando cada cliente al representante más cercano. Esta cercanía se mide a través de la distancia escogida, que puede ser, entre otras:

- **Distancia Euclídea:**
  Dados dos clientes i y j, se corresponde con la verdadera distancia geométrica entre los puntos con coordenadas $(x_{i1}, \ldots, x_{ip})$ y $(x_{j1}, \ldots, x_{jp})$,

$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2}.$$

- **Distancia Manhattan:**
  Debe su nombre a que si te encuentras en un punto i de una ciudad como Manhattan, donde todo su plano se rige por una cuadrícula, la distancia más corta al punto j será una aquella en la que sigas una combinación de líneas verticales y horizontales, debido la imposibilidad de salirte de la cuadrícula dada por lo edificios correspondientes. Su expresión es la siguiente:

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{ip} - x_{jp}|.$$

Cabe destacar que las dos distancias mencionadas cumplen las condiciones matemáticas necesarias para considerarse como tal. Para todo cliente i, j y h, se verifica:

- $d(i,j) \geq 0$.
- $d(i,i) = 0$.
- $d(i,j) = d(j,i)$.
- $d(i,j) \leq d(i,h) + d(h,j)$.

No toda selección de representantes lleva a una buena partición, por lo que los representantes deben ser elegidos de forma que estén localizados en el centro de los clusters que definen cada uno. Exactamente, la distancia media del representante a los demás clientes del mismo clúster debe ser minimizada. Por ello al representante se le denominará el medoide de su clúster y esta técnica se denomina k-medoides.

La principal diferencia que presenta CLARA frente a otros métodos de *clustering* es que no almacenará todas las distancias dos a dos entre clientes, ya que esto consumiría memoria del orden de $O(n^2)$, inviable en grandes conjuntos de datos.

El algoritmo seguido por CLARA consiste en los siguientes pasos:

1. Se escoge un número N de muestras en las que dividir el conjunto de datos.
2. Se toma una de las N muestras del conjuntos de datos y se aplica la técnica k-medoides, obteniendo k clusters con un representante cada uno.
3. Cada cliente no perteneciente a la muestra tomada en el paso anterior es asignado al clúster con el representante más cercano a él.
4. Se repite este proceso con cada una de las N muestras seleccionadas, y se escoge la partición del conjunto de datos dada por la muestra que obtenga la mínima distancia media entre cada cliente y su representante.

Para más información sobre el método CLARA se recomienda consultar el Capítulo 3 de Kaufman y Rousseeuw [5].

Se aplica entonces este método sobre el conjunto de datos construido en el Capítulo 3, para poder comprobar la eficacia del método, eligiendo tanto distancia euclídea como Manhattan, representado en las Figuras 2.1 y 2.2, respectivamente. Se toman $N = 50$ y $k = 7$.

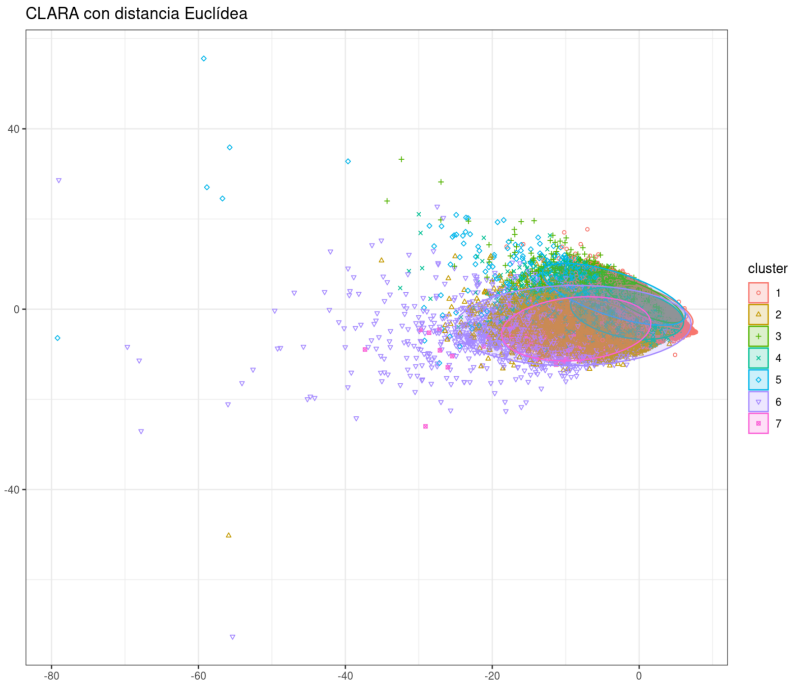


Figura 2.1: Diagrama de dispersión construid a partir de las dos primeras componentes principales que definen a los clientes. Se muestra el resultado de la partición en grupos obtenida mediante la aplicación del método CLARA con distancia euclídea.

Al disponer de tantos datos, no se pueden evaluar bien los resultados gráficamente. Se procede a comparar el tamaño de los clusters dados por este método con el verdadero tamaño de los grupos que pertenecen a cada etapa del ciclo de vida familiar, información que podemos encontrar en la Tabla 3.2.

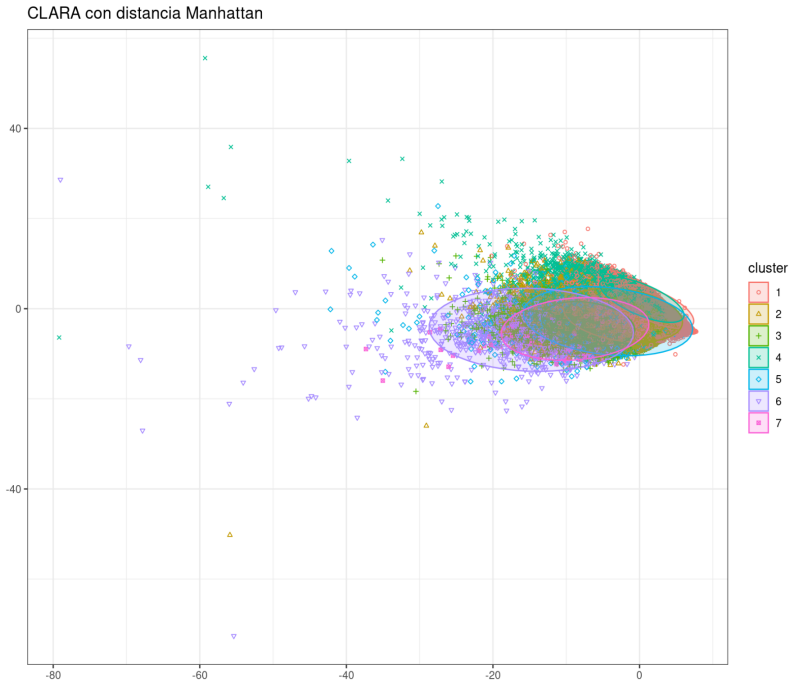


Figura 2.2: Diagrama de dispersión construido a partir de las dos primeras componentes principales que definen a los clientes. Se muestra el resultado de la partición en grupos obtenida mediante la aplicación del método CLARA con distancia Manhattan.

El tamaño de los clusters dados por el método CLARA con distancia euclídea se muestra en la siguiente salida de R:

```
Cluster sizes:           219505 39584 57911 45493 1501 3492 279
```

Estos resultados distan mucha de la realidad, ya que, sólo fijándonos en el primer clúster, éste está formado por casi el doble de clientes que la categoría real más numerosa.

El tamaño de los clusters dados por el método CLARA con distancia Manhattan se muestra en la siguiente salida de R:

```
Cluster sizes:           229365 42990 30100 60562 3165 1278 305
```

Se observa que estamos ante la misma situación que en el caso anterior.

A falta de hacer un estudio más exhaustivo, se concluye entonces que los métodos de *clustering* no son, en principio, adecuados para resolver el problema a tratar en este trabajo, ya que el objetivo no es solo dividir el conjunto de datos en 7 grupos, si no que cada grupo esté dotado de ciertas características que nosotros imponemos, por lo que serán necesarios métodos de aprendizaje supervisado, que se verán en la Parte II de este trabajo.

Parte II

Modelos de clasificación supervisada

Capítulo 3

Construcción de la base de datos adecuada

Como se vió en el Capítulo 2, para la resolución del problema propuesto por la entidad, una técnica de aprendizaje no supervisado no será suficiente. Por ello, se utilizarán técnicas de aprendizaje supervisado, las cuales serán expuestas en el Capítulo 4 (métodos de clasificación). Por lo tanto, nos hallamos ante una base de datos con información de 1500000 clientes, pero no se dispone de la información suficiente de todos ellos para saber a qué etapa del ciclo de vida familiar pertenecen. Por ello, el primer objetivo será seleccionar aquellos clientes para los que sí se tiene la información necesaria y, a partir de ellos, construir un modelo mediante la aplicación de técnicas de aprendizaje supervisado que permitan predecir la respuesta para aquellos clientes cuya información necesaria se desconoce.

3.1. Creación de los leads

Como se comentó previamente en la Sección 1.2, los datos son extraídos mediante el programa SQL a partir de diversas tablas, entre ellas *mktg_usr.tabla_datos_entrenamiento*, pero, para poder aplicar un modelo de clasificación, esto no es suficiente. Se necesita tener una parte de la base de datos totalmente identificada, de forma que se sepa exactamente a qué etapa del ciclo de vida familiar descrito en la Sección 1.1.1 pertenece cada cliente.

Para ello, se crean unos indicadores en SQL (consultas), a los cuales llamaremos 'leads'. Para cada etapa del ciclo de vida familiar crearemos un lead, el cual creará variables nuevas para cada cliente, entre ellas una variable indicadora, que devolverá un 1 si el cliente pertenece a esta etapa y un 0 si no pertenece a ella.

Según se vayan creando los leads, se unirán a la tabla base, *mktg_usr.tabla_datos_entrenamiento*, mediante un LEFT JOIN.

Finalmente, se creará la variable factor *Ciclo_de_vida*, variable respuesta que tomará valores del 0 al 6 dependiendo a que etapa/clúster se detecta que pertenece cada cliente, de la forma expuesta en la Tabla 3.1.

Una vez creada esta variable, se eliminarán de la base de datos todas aquellas variables indicadoras que se obtuvieron a través de los leads y, por tanto, se tendrá una base de datos compuesta por 216 variables.

A su vez, nos quedaremos con una base de datos compuesta por aquellos clientes con un valor de la variable *Ciclo_de_vida* detectado. En este caso, la base de datos resultante estará compuesta por un total de 367765 clientes, distribuidos tal y como se muestra en la Tabla 3.2.

Ciclo_de_vida	
0	Single
1	Pareja sin hijos
2	Pareja con hijos pequeños (0-6)
3	Pareja con hijos en edad escolar (6-18)
4	Pareja con hijos mayores de edad
5	Nido vacío
6	Viudo/a

Tabla 3.1: Valores de la variable *Ciclo_de_vida*.

Número de clientes	
Single	-
Pareja sin hijos	-
Pareja con hijos pequeños (0-6)	-
Pareja con hijos en edad escolar (6-18)	-
Pareja con hijos mayores de edad	-
Nido vacío	-
Viudo/a	-

Tabla 3.2: Número de clientes que pertenecen a cada categoría de la variable *Ciclo_de_vida* (tabla modificada por confidencialidad).

3.1.1. Single

El primer paso será quedarse con aquellos clientes que en todos sus contratos no tienen ningún otro cliente asociado, medida para descartar posibles parejas. Esto se hará cruzando la tabla *mktg_usr.tabla_datos_entrenamiento*, que contiene todos los datos de los clientes, con la tabla *viewsinq.cliente_contrato_rl*, que contiene los datos de todos los contratos, mediante la clave primaria *CLIENTE_ID*.

Imponemos entonces la condición de quedarnos con los clientes en cuyos contratos el número de personas involucrado sea un total de 1, información guardada en la variable n_clientes. Por tanto, la sentencia a imponer en SQL será

$$\text{HAVING Max(n\_clientes)} = 1.$$

Una vez se realiza este paso, nos quedamos con los clientes con domicilio único, es decir, aquellos

que viven solos, no compartiendo la casa con ninguna otra persona.

Seguidamente, eliminamos aquellos cliente compartidos con otras entidades, ya que esto puede significar que la cuenta abierta en ABANCA no es su cuenta principal, lo que dificultaría detectar sus contratos y si dispone de domicilio único. Esta información se almacena en la tabla *mktg_usr.clientes_compartidos_hist*, por lo que volvemos a cruzar por clave primaria *CLIENTE_ID*. Se pide entonces tomar aquellos clientes que tengan un *CLIENTE_ID* nulo en la tabla *mktg_usr.clientes_compartidos_hist*.

Finalmente, se le pide al cliente que tenga una vinculación con el banco suficiente, para así asegurarse de seleccionar a singles reales, y no a clientes con información insuficiente sobre ellos. Para ello, se toma el percentil 40 de la variable *VINCULACION*, y se pide al cliente que supere este valor.

Una vez se termina este proceso, resulta la variable *in_single*, variable indicadora que toma el valor 1 si se detecta que el cliente pertenece al clúster single y 0 si no se detecta.

3.1.2. Parejas sin hijos

El primer paso para detectar una pareja sin hijos será localizar las cuentas compartidas por más de una persona. Se empieza seleccionando todas las relaciones vigentes de los contratos, información que se halla cruzando las tablas *viewsinq.cliente_contrato_rl* y *viewsinq.cliente* mediante la clave primaria *CLIENTE_ID*. Una vez halladas, se cruzan los NUC ('Número de contrato') con todas las relaciones que tiene, mediante un INNER JOIN con la tabla *viewsinq.cliente_contrato_rl* mediante la clave primaria *nuc*, seguido de otro INNER JOIN con la tabla *vistascrme.dmk_cliente* mediante la clave primaria *CLIENTE_ID*.
Se pide entonces que el contrato esté vigente, es decir, que la variable *fecha_baja_relacion* sea nula, y finalmente, que el contrato tenga al menos dos titulares, información guardada en la variable *n_clientes*, mediante la sentencia

$$\text{HAVING Max}(n\_clientes) > 1.$$

El segundo paso será comprobar qu el cliente comparte domicilio con al menos una persona más. Para ello, se obtienen todos los clientes asociados a las direcciones que tienen mas de un cliente, información que se obtiene de un INNER JOIN de las tablas *viewsinq.cliente* y *vistascrme.dmk_cliente* mediante la clave primaria *CLIENTE_ID*.

Finalmente, identificamos aquellos clientes que comparten núcleo financiero, información que se extrae de la tabla *mktg.cliente_nucleo_financiero*.

De este proceso resulta la variable *in_pareja_sin_hijos*, variable indicadora que toma el valor 1 si se detcta el cumplimiento de las tres condiciones especificadas y 0 si no se detecta.

3.1.3. Parejas con hijos pequeños (0-6 años)

En el momento de tener un hijo y de iniciar la baja de maternidad/paternidad, la seguridad social pasa a hacerse cargo del sueldo del cliente. El objetivo será detectar el momento en el que comienzan esas ayudas y detectar todos aquellos clientes que hayan pasado por ese momento en los últimos 6 años. En nuestro caso, aquellos clientes que hayan posado por ese momento entre el 31 de enero de 2017 y el 31 de enero de 2023. Para asegurarnos de la veracidad de este indicador, exigiremos que la madre y el padre sean menores de 45 años en la fecha del primer cobro.

Para detectar estos cobros, se buscarán transferencias en la cuenta del cliente por conceptos clave, entre los que están incluidos 'A.E.A.T.' y 'DIRECCION PROVINCIAL' como nombre del ordenante, y 'SOLICITUD DEDUCCION MATERNIDAD', 'PATERNIDAD', 'MATERNIDAD' y 'NACIMIENTO' como conceptos. Asimismo, buscaremos transferencias o movimientos con conceptos como 'bebe', 'carrito', 'cuna', 'vacuna' o 'bautizo'. Toda esta información podemos extraerla de las tablas *cinf_e.oa01tb01*, *viewsinq.orden_movimiento* y *viewsinq.ra10tb01*.

Obtenemos entonces, resultantes de este proceso, las variables *n_cobros*, que indica el número de cobros detectados por este proceso que recibió el cliente, *primer_cobro* que indica la fecha en la que se efectuó el primer cobro, y *ultimo_cobro*, que indica la fecha en la que se efectuó el último cobro.
Seguidamente se crea la variable *in_hijo_0_6*, variable indicadora que toma el valor 1 si se detectó el primer cobro entre el 31 de enero de 2017 y el 31 de enero de 2023 y 0 si no se detectó.

3.1.4. Parejas con hijos en edad escolar (6-18 años)

El objetivo será encontrar pares únicos padre/madre-hijo, donde la edad del hijo esté comprendida entre los 6 y 18 años, estando los 18 no incluidos. Entonces procedemos con una búsqueda con el objetivo de encontrar pares de personas con los mismos apellidos (los primeros o el segundo y el primero, dependiendo del orden de los apellidos escogidos por los padres), misma dirección, código postal y teléfono fijo. Es imprescindible que sólo uno de ellos sea menor de edad.

Esta información se halla en la tabla *viewsinq.cliente*, por lo que la debemos cruzar con ella misma mediante la clave primaria *CLIENTE_ID*. De esta forma, una tabla nos proporcionará la información del padre/madre, mientras que la otra proporciona los datos del menor.

Resulta entonces la variable *in_hijo_menor*, variable indicadora que toma el valor 1 cuando se detecta que el cliente aparece en uno de los pares únicos buscados y 0 si no se detecta. Agrupando estos resultados por la clave primaria *CLIENTE_ID*, se obtienen las variables *n_hijos_menores*, indicando el número de hijos menores de edad del cliente, *edad_menor_primero*, indicando la edad del hijo mayor con edad entre 6 y 18 años, y *edad_menor_ultimo*, indicando la edad del hijo pequeño con edad entre 6 y 18 años.

Paralelamente, repetiremos el proceso descrito en la Sección 3.1.3, donde esta vez buscaremos que el primer cobro se haya efectuado entre el 31 de enero de 2005 y el 31 de enero de 2017, siempre y cuando la edad de la madre y el padre sea menor de 45 años en el momento del primer cobro. Si ese primer cobro existe y cumple las condiciones estipuladas para cierto cliente, entonces *in_hijo_menor* será igual a 1 para ese cliente.

3.1.5. Parejas con hijos mayores de edad

El objetivo será tomar aquellos contratos formado por dos personas donde una de ellas era representante legal, pero dejó de serlo. Este caso se da en aquellas cuentas que se crean a un menor, donde es necesario que uno de los padres conste como representante legal, y este cumple la mayoría de edad, ya que cuando esto pasa el representante legal deja automáticamente de serlo. Por otro lado, para asegurarnos de que nos hallamos ante un hijo mayor de edad que sigue viviendo en casa, y no ante un caso de nido vacío, imponemos las condiciones de que el hijo sea menor de 27 años y que no reciba nómina.

Para ello, primero se cogen los contratos que están activos a fecha de la observación. Estos se encuentran en la tabla *viewsinq.cliente_contrato_rl*, y se les exige que la variable *fecha_baja_relacion* del

contrato sea nula. Esto significa que la variable con valor nulo debe de ser aquella asociada al hijo, para obtener esta información se realiza un INNER JOIN con la tabla *vistascrme.dmk_cliente* a través de la clave primaria *CLIENTE_ID*, y seleccionamos las observaciones pertenecientes al hijo. Acto seguido, se toman aquellos en los que hubo una fecha de baja relación de un tutor, es decir, se les exige que la variable *fecha_baja_relacion* del representante legal sea no nula. Entonces se vuelve a realizar un INNER JOIN con la tabla *vistascrme.dmk_cliente* a través de la clave primaria *CLIENTE_ID*, esta vez tomando las observaciones relativas al representante legal.

De este proceso resulta la variable *in_hijo_mayor*, variable indicadora que toma el valor 1 si al cliente se le detcta un hijo mayor de edad que siga viviendo en el domicilio familiar y 0 si no se detecta. Agrupando estos resultados por la clave primaria *CLIENTE_ID*, se obtienen las variables *n_hijos_mayores*, que nos indica el número de hijos mayores de edad residentes en el domicilio familiar detectados; *edad_mayor_primero*, que nos indica la edad del hijo mayor con más de 18 años residente en el domicilio familiar, y *edad_mayor_ultimo*, que nos indica la edad del hijo menor con más de 18 años residente en el domicilio familiar.

De nuevo, se repite el proceso descrito en la Sección 3.1.3, donde esta vez buscaremos que el primer cobro se haya efectuado antes del 31 de enero de 2005, siempre y cuando la edad de la madre y el padre sea menor de 45 años en el momento del primer cobro. Si ese primer cobro existe y cumple las condiciones estipuladas para cierto cliente, entonces *in_hijo_mayor* será igual a 1 para ese cliente.

3.1.6. Parejas con hijos independizados (nido vacío)

Como fue especificado en la Sección 1.1.1, la etapa de nido vacío se da cuando todos los hijos abandonan el hogar familiar, es decir, en el momento en el que el último hijo lo abandona.

Para detectar esta situación, el primer paso será seguir el proceso descrito en la Sección 3.1.5, es decir, detectar la existencia de un hijo mayor de edad. Se seguirá este proceso ya que no se tendrá en cuenta el caso de un menor emancipado o independiente económicamente.

El siguiente paso será comprobar que el cliente no tenga hijos menores de edad. Por tanto, se seguirá el proceso relatado en la Sección 3.1.4, es decir, detectar la presencia de un hijo menor de edad, extrayendo la variable *CLIENTE_ID* del menor. Se pedirá entonces que este sea nulo, asegurándonos así de la no existencia de un hijo menor de edad. Por último, para asegurar todo lo posible que los hijos mayores de edad no residan en el domicilio familiar, se les pedirá una nómina mayor o igual a 1500 €.

Resulta entonces la variable *in_nido_vacio*, variable indicadora que toma el valor 1 si se detecta que un cliente pertenece al clúster de nido vacío y 0 si no se detecta. Agrupando estos resultados por la clave primaria *CLIENTE_ID*, se obtienen las variables *n_nido*, indicando el número de hijos que posee el cliente perteneciente al clúster de nido vacío, *edad_nido_primero* y *edad_nido_ultimo*, que indican la edad del hijo mayor y la del pequeño, respectivamente, de un cliente clasificado en el clúster nido vacío.

3.1.7. Viudo/a

Se toma la tabla *viewsinq.cliente*, donde se encuentran los datos personales de los clientes, y se realiza un INNER JOIN con la tabla *mktg.cliente_nucleo_financiero*, donde se encuentran los datos de los clientes que comparten núcleo financiero con el cliente seleccionado, bajo la clave primaria *CLIENTE_ID*. En este caso, compartir núcleo financiero se entenderá como dos personas compartiendo cuenta.

Se toman entonces los datos del cliente que comparta núcleo financiero con un cliente marcado como fallecido, resultando la variable *in_rel_fallecido*, variable indicadora que toma el valor 1 cuando se detecta qu el cliente comparte núcleo financiero con una persona fallecida (y por tanto pertenece al clúster viudo/a) y 0 si no se detecta.

3.2. Análisis de variables

Esta sección se ha omitido por motivos de confidencialidad. En ella se hizo un análisis de las variables *EDAD*, *IMP_NOMINA_SUA*, *IMP_RECIBOS_BASICOS*, *PLAZO_DMK_CT*, *IN_DIGITAL* y *TJCREDITO*.

Capítulo 4

Modelos de clasificación

Con la finalidad de desarrollar las diferentes técnicas de modelización en el desarrollo de este Capítulo, se representa como X al conjunto total de los datos definidos en la Sección 1.2 más la variable *Ciclo_de_vida* definida en la Sección 3.1, y como $X_1, \ldots, X_p$ al conjunto de variables explicativas. El objetivo es poder predecir la variable respuesta en función de los valores de las variables explicativas que definen a los clientes.

En este caso, la variable objetivo Y indica la probabilidad de pertenecer a cada una de las etapas del ciclo de vida familiar explicado en la Sección 1.1.1. Es decir, la variable respuesta Y tomará 7 valores, siendo cada uno de ellos la probabilidad de que el cliente pertenezca a la respectiva etapa del ciclo de vida familiar mencionado.

Se agruparán por tanto las predicciones en $Y = 0$, $Y = 1$, $Y = 2$, $Y = 3$, $Y = 4$, $Y = 5$ e $Y = 6$, valores de la variable *Ciclo_de_vida* que se pueden ver en la Tabla 3.1, dependiendo de a qué etapa se le asigne una probabilidad más alta. Por tanto, se llevará a cabo un proceso de clasificación supervisada. En este tipo de aprendizaje, el método se entrena con un histórico de datos y, en base al comportamiento a lo largo del tiempo de las observaciones, aprende a asignar la etiqueta de la variable objetivo adecuada a un nuevo valor. Nos encontramos, por tanto, ante un problema donde se aplicarán modelos de clasificación multiclase.

En este capítulo, aparte de los libros y artículos que serán mencionados a lo largo de él, se seguirán los apuntes de la asignatura Aprendizaje Estadístico del Máster de Técnicas Estadísticas impartido por la USC, la UDC y la UVigo, escritos por Fernández-Casal et al. [7].

4.0.1. Partición de los datos

Antes de aplicar un método de clasificación supervisada, se debe previamente separar el conjunto de datos original en subconjuntos de datos de menor tamaño, los cuales serán utilizados con dos distintos fines: entrenamiento y validación.
El subconjunto de datos de entrenamiento se emplea para estimar los parámetros del modelo, mientras que el subconjunto de datos de validación se utiliza para comprobar el comportamiento predictivo del modelo estimado.

Generalmente, se divide el conjunto de datos de forma que el 70 % de ellos sean destinados al entrenamiento y el 30 % a validación, pero esto puede llevar a generar una dependencia entre los dos conjuntos en la implementación de algunos modelos, concretamente para los métodos *Bagging* y *Boosting*, provocando que las predicciones obtenidas no sean lo suficientemente fiables. Esto se da debido

a que los métodos mencionados requieren establecer hiperparámetros para moldear su estructura, por lo que se recomienda separar la muestra de entrenamiento en dos, las cuales servirán para entrenar el modelo y para estimar los hiperparámetros necesarios (entrenamiento y test), y validar el resultado en una muestra totalmente nueva (validación).

Cada registro de la base de datos debe aparecer en uno de los tres subconjuntos. Para ello, la división se realiza a partir de la variable *CLIENTE_ID*, variable de la base de datos que contiene un identificador único para cada cliente de la entidad, consiguiendo así que no se repita el mismo individuo en muestras diferentes y se mitigue el efecto de la dependencia. Para separar el conjunto de datos, se utilizará un procedimiento de muestreo aleatorio, simple o estratificado.

Lo idóneo para entrenar un modelo es que el conjunto de datos de entrenamiento sea independiente del conjunto de datos utilizados para la validación, por tanto en este trabajo se separarán los datos en tres grupos:

- Entrenamiento, 60 %.
- Test, 20 %.
- Validación, 20 %.

Se realizará la predicción sobre la muestra de validación, la cual será independiente de la muestra de entrenamiento y la muestra de test. La comparación del resultado obtenido a partir de la muestra de validación permite validar el modelo en término de error de predicción.

4.1. Modelos de regresión

Los modelos de regresión comprenden un extenso campo en el aprendizaje estadístico. Son un conjunto de modelos que tratan de explicar y predecir el valor de la variable respuesta Y en función de los valores que adopten los diferentes predictores o variables explicativas $(X_1, \ldots, X_p)$ significativos a la hora explicar Y. Se define entonces un modelo de regresión como un problema matemático en el cual existe una relación entre una variable respuesta nombrada como Y, y una o varias variables explicativas o covariables, identificadas como X.

En la actualidad existe una gran variedad de algoritmos predictivos, pero entre ellos la regresión logística sigue siendo un método, aunque clásico, robusto. Este constituye un método de regresión útil para resolver problemas de clasificación binaria. En este trabajo nos encontramos ante un problema de clasificación multiclase, por lo que se usará regresión logística multinomial, que generaliza la regresión logística a problemas multiclase.

4.1.1. Modelo de regresión lineal múltiple

El modelo de regresión lineal múltiple es una extensión del modelo lineal simple, utilizado para expresar la dependencia de la variable respuesta Y con respecto a las variables explicativas $X_1, \ldots, X_n$ de forma lineal. La función de regresión, definida como la media de la variable respuesta condicionada a los valores de X de la forma:

$$f(X) = \mathbb{E}(Y|X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j, \tag{4.1}$$

donde $\beta = (\beta_1, \ldots, \beta_p)^T$ es el conjunto de parámetros desconocidos. Es decir, la variable Y se explica en función de su media condicionada a X más un error no observable ε. Por tanto, el modelo de regresión lineal múltiple se podrá formular como:

$$Y = f(X) + \varepsilon = \beta_0 + \sum_{j=1}^{p} X_j \beta_j + \varepsilon, \tag{4.2}$$

siendo ε normalmente una variable aleatoria gaussiana de media 0 y varianza fija σ^2, es decir, $\varepsilon \sim N(0, \sigma^2)$.

Si se reescribe la Ecuación 4.2 de forma matricial, se obtiene que X se corresponde con una matriz de diseño de dimensión $N \cdot (p+1)$, donde su i-ésima fila, siendo $i = 1, \ldots, N$, se corresponde con la i-ésima observación. Su primera columna estará formada por unos y la j-ésima columna, con $j = 1, \ldots, p+1$, recogerá los valores de la variable explicativa X_{j-1}.

Por tanto, denotando por Y al vector de observaciones de la respuesta, por β al vector de parámetros y por ε al vector de errores, se puede escribir la Ecuación 4.2 de la forma $Y = X\beta + \varepsilon$, siendo su descomposición la siguiente:

$$\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_{N-1} \\ Y_N \end{pmatrix} = \begin{pmatrix} 1 & X_{1,1} & X_{1,2} & \cdots & X_{1,p} \\ 1 & X_{2,1} & X_{2,2} & \cdots & X_{2,p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_{N-1,1} & X_{N-1,2} & \cdots & X_{N-1,p} \\ 1 & X_{N,1} & X_{N,2} & \cdots & X_{N,p} \end{pmatrix} \cdot \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{p-1} \\ \beta_p \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_{N-1} \\ \varepsilon_N \end{pmatrix}, \tag{4.3}$$

en la que $Y \in \mathcal{M}_{N\times 1}$, $X \in \mathcal{M}_{N\times(p+1)}$ y $\beta \in \mathcal{M}_{(p+1)\times 1}$. De esta forma, Y y X serán ahora un vector y una matriz de valores conocidos dados por los datos muestrales, respectivamente.

Para este modelo se asumen las siguientes hipótesis:

- **Linealidad:** La función de regresión dada en la Ecuación 4.2 debe ser lineal.
- **Homocedasticidad:** La varianza del error será constante, sea cual sea el valor de las variables explicativas,
$$\text{Var}(\varepsilon \sim (X_1, \ldots, X_p)) = \sigma^2.$$
- **Normalidad:** El error sigue una distribución normal con media cero y varianza constante,
$$\varepsilon \sim N(0, \sigma^2).$$
- **Independencia:** Los residuos del modelo, variables aleatorias que representan los errores, $\hat{\varepsilon_1}, \ldots, \hat{\varepsilon_N}$ deben ser independientes entre si.

Este modelo se ajusta mediante el método de mínimos cuadrados, minimizando así la suma de los cuadrados de los residuos:

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} (y_i - \beta_0 - X_{1,i}\beta_1 - \cdots - X_{p,i}\beta_p) \tag{4.4}$$

4.1.2. Modelo lineal generalizado

El modelo lineal generalizado constituye una extensión del modelo lineal múltiple, en la que se permite que la distribución de las Y_i condicionada por las X_i no sea necesariamente normal, sino una distribución exponencial con esperanza condicionada $\mu_i = \mathbb{E}[Y_i|X_i]$, $i = 1, \ldots, N$.

Como se puede ver en Wood [8], en los modelos lineales generalizados la distribución de las $Y_i|X_i$ pertenece a la familia exponencial y su función de densidad admite la expresión:

$$f(Y_i|\theta_i, \xi) = \exp\left\{\frac{Y_i\theta_i - b(\theta_i)}{a(\xi)} + c(Y_i, \xi)\right\}, \tag{4.5}$$

donde θ_i se denomina parámetro canónico de localización, ξ es el parámetro de escala y, $a(\cdot)$, $b(\cdot)$ y $c(\cdot)$ son funciones que determinan el tipo de familia exponencial.

Al no seguir una distribución normal, no se cumplen las hipótesis especificadas para un modelo de regresión lineal. Por tanto, se introduce una función de enlace, o función *link*, que transforma el modelo ajustado de la escala lineal a la escala original por la que los datos están realmente distribuidos. Observando el modelo de regresión lineal explicado en la Sección 4.1.1, se determina que estamos ante un caso particular de modelo lineal generalizado en el que la distribución de la variable respuesta es normal, por lo que se cumplen las hipótesis mencionadas y la función *link* aplicada sería la identidad.

La función *link* $g(\cdot)$, monótona y diferenciable, relaciona el predictor lineal $\eta_i = X_i^T\beta$ con la media de la siguiente forma:

$$\eta_i = g(\mu_i) = X_i^T\beta, \quad i = 1, \ldots, N. \tag{4.6}$$

Si se denota por $h(\cdot)$ a la inversa de la función *link*, es decir $h = g^{-1}$, entonces

$$\mu_i = h(\eta_i) = h(X_i^T\beta), \quad i = 1, \ldots, N. \tag{4.7}$$

Por lo tanto, un modelo de regresión lineal generalizado está especificado por el tipo de distribución de la familia exponencial seleccionada, que viene dada por la función *link*, y por la matriz de diseño X.

Para cada familia exponencial existe una función link que iguala al parámetro natural θ_i con el predictor lineal η_i, de forma que

$$\theta_i = g(\mu_i) = \eta_i = X_i^T\beta. \tag{4.8}$$

Ahora, visto el modelo lineal generalizado de forma superficial, se explicará la regresión logística, pensada para una variable objetivo binaria. Acto seguido, se explicará como llegar a partir de ella a la regresión logística multinomial, adaptándola así a una variable objetivo multiclase.

Regresión logística

El modelo logístico surge ante la necesidad de tomar una decisión ante una respuesta binaria. Es un modelo muy utilizado en campos como la medicina y la bioestadística, donde el ejemplo más común será aplicar este modelo para determinar si un paciente padece una enfermedad o no en función de una serie de variables explicativas relacionadas con esta.

Se considera la variable respuesta Y binaria, donde generalmente toma los valores 0 y 1, representando fracaso y éxito respectivamente. Por tanto, la distribución de esta variable será una Bernouilli, donde la media será la probabilidad de éxito, es decir,

$$\mathbb{E}(Y_i|X = X_i) = \mathbb{P}(Y_i = 1|X = X_i), \tag{4.9}$$

$$\mathrm{Var}(Y_i|X = X_i) = \mathbb{P}(Y_i = 1|X = X_i) \cdot [1 - \mathbb{P}(Y_i = 1|X = X_i)], \tag{4.10}$$

representando Y_i el valor de la variable respuesta para el individuo i y $X_i \in \mathbb{R}^p$, $p \in \mathbb{N}$ e $i = 1, \ldots, N$ el vector dado por los valores de las variables explicativas.

Vemos entonces la imposibilidad de representar la media de Y mediante un modelo de regresión lineal, ya que se incumplirían las hipótesis básicas del modelo. La primera en incumplirse sería la linealidad, ya que si se pretende expresar la media de Y como función lineal las predicciones no estarán, en la mayoría de los casos, comprendidas en el intervalo $[0, 1]$, condición necesaria ya que la media es una probabilidad de éxito, tal y como fue definida en la Ecuación 4.9. Seguidamente, se observa que claaramente no estamos ante un modelo homocedástico, ya que el valor de la varianza depende de cada observación, tal y como viene dado en la Ecuación 4.10. Por último, no se cumplirá la hipótesis de normalidad, ya que se nos presenta una distribución Bernouilli. Es decir, la única hipótesis factible sería la de independencia.

Por tanto, se considera un modelo para la probabilidad de éxito condicionada a cada valor de la variable explicativa, siendo esta

$$\pi(X) = \mathbb{P}(Y_i = 1|X = X_i).$$

Para poder considerar un modelo lineal, necesitamos una función link $g(\cdot)$, que aplicándola sobre $\pi(X)$, transforme el intervalo $[0, 1]$ en la recta real. En la regresión logística la función elegrida es la función *logit*:

$$g(p) = \log\left(\frac{p}{1-p}\right), \quad p \in [0, 1].$$

Si tomamos como p la probabilidad de éxito, esta función se basará en aplicar un logaritmo natural a la *odds*, cociente entre la probabilidad de éxito y la probabilidad de fracaso:

$$\mathrm{Odds}(Y) = \frac{\mathbb{P}(Y = 1)}{\mathbb{P}(Y = 0)}.$$

Por tanto, en el modelo de regresión logística se da la siguiente igualdad:

$$\log\left(\frac{\pi(X, \beta)}{1 - \pi(X, \beta)}\right) = X^T\beta.$$

Para representar el modelo como la probabilidad de éxito, será necesario invertir la función *logit*, acción posible gracias a la correspondencia biunívoca y creciente entre los intervalos $[0, 1]$ y $(-\infty, \infty)$, siendo las dos funciones derivables con derivadas continuas. La inversa de la función *logit* $g(\cdot)$ es

$$h(X) = \frac{e^X}{1 + e^X},$$

y por tanto la probabilidad de éxito es expresada por el modelo de la forma

$$\pi(X, \beta) = h(X, \beta) = \frac{e^{X^T\beta}}{1 + e^{X^T\beta}} > 0, \tag{4.11}$$

estrictamente positiva, por serlo también $e^{X^T\beta}$.

Regresión logística multinomial

Como fue definida anteriormente, la regresión logística multinomial generaliza la regresión logística a problemas multiclase. En esta, el objetivo será modelar la probabilidad

$$\pi_k(X) = \mathbb{P}(Y_i = k|X = X_i), \quad k = 1, \ldots, K,$$

siendo K el número total de clases definidas por la variable objetivo Y.

Para llegar al modelo de regresión logística multinomial, se puede pensar en él como un conjunto de regresiones logísticas binarias independientes entre si. Es decir, para K posibles clases, se toma una de ellas como 'pivote', y acto seguido se ejecutan $K-1$ modelos de regresión logística binaria independientes tomando de referencia este pivote.
Si se toma como pivote la primera clase definida por la variable Y, las $K-1$ ecuaciones de regresión son de la forma:

$$\log\left(\frac{\mathbb{P}(Y_i = k|X = X_i)}{\mathbb{P}(Y_i = 1|X = X_i)}\right) = \log\left(\frac{\pi_k(X, \beta_k)}{\pi_1(X, \beta_1)}\right) = X^T\beta_k, \quad k > 1. \tag{4.12}$$

Esta formulación se conoce como *arl* ('Additive logratio transform'), utilizada comúnmente en el análisis de datos composicionales.

Exponenciando ambos lados de la Ecuación 4.12 y despejando llegamos a que

$$\pi_k(X, \beta_k) = \pi_1(X, \beta_1)\exp\{X^T\beta_k\}, \quad k > 1.$$

Como las K probabilidades deben sumar uno, es decir,

$$\sum_{k=1}^{K} \pi_k(X, \beta_k) = 1,$$

se tiene

$$\begin{aligned}\pi_1(X, \beta_1) =& 1 - \sum_{k=2}^{K} \pi_k(X, \beta_k) = 1 - \sum_{k=2}^{K} \pi_1(X, \beta_1)\exp\{X^T\beta_k\} \\ \Rightarrow& \pi_1(X, \beta_1) = \frac{1}{1 + \sum_{k=2}^{K} \exp\{X^T\beta_k\}}.\end{aligned} \tag{4.13}$$

Procediendo de la misma forma con la que se obtuvo la Ecuación 4.13, obtenemos las probabilidades para las demás clases:

$$\pi_k(X, \beta_k) = \frac{\exp\{X^T\beta_k\}}{1 + \sum_{k=2}^{K} \exp\{X^T\beta_k\}}, \quad k = 2, \ldots, K. \tag{4.14}$$

Para más información sobre la regresión logística multinomial, consultar Engel [9] y Menard [10].

4.2. Árboles de decisión

Los árboles de decisión son métodos de clasificación supervisados que ofrecen una manera simple de interpretación, ya que dividen el espacio definido por las variables predictoras en un número reducido de regiones, por ejemplo, rectángulos en el caso bidimensional. Estas regiones se caracterizan por agrupar observaciones similares en términos de la variable respuesta. Para realizar predicciones, generalmente se emplea la media o la moda de las observaciones de entrenamiento dentro de la región a la que pertenece cada observación a predecir. A pesar de su enfoque conceptualmente sencillo, estos árboles poseen un gran poder predictivo. Esta idea fue presentada por Breiman et al., en el libro 'Classification and regression trees' [11], y ha ido experimentando evoluciones a lo largo del tiempo, dando lugar a técnicas más complejas y eficientes como los métodos Bagging y Boosting.

Los árboles de decisión son de los métodos más utilizados para predecir el valor de una variable respuesta. Dependiendo de la naturaleza de dicha variable, estos árboles se clasifican en dos categorías: árboles de regresión, cuando la variable respuesta es continua, y árboles de clasificación, cuando la variable respuesta es discreta.

En ambas categorías, el objetivo consiste en construir un modelo en forma de árbol. Es decir, se comienza con el conjunto completo de datos y, mediante divisiones binarias, se va dividiendo el conjunto a lo largo de ramificaciones en las variables explicativas. Este proceso continúa hasta que se cumple una condición de parada, momento en el cual el árbol deja de ramificarse y los datos restantes en ese nodo se denominarán terminales.

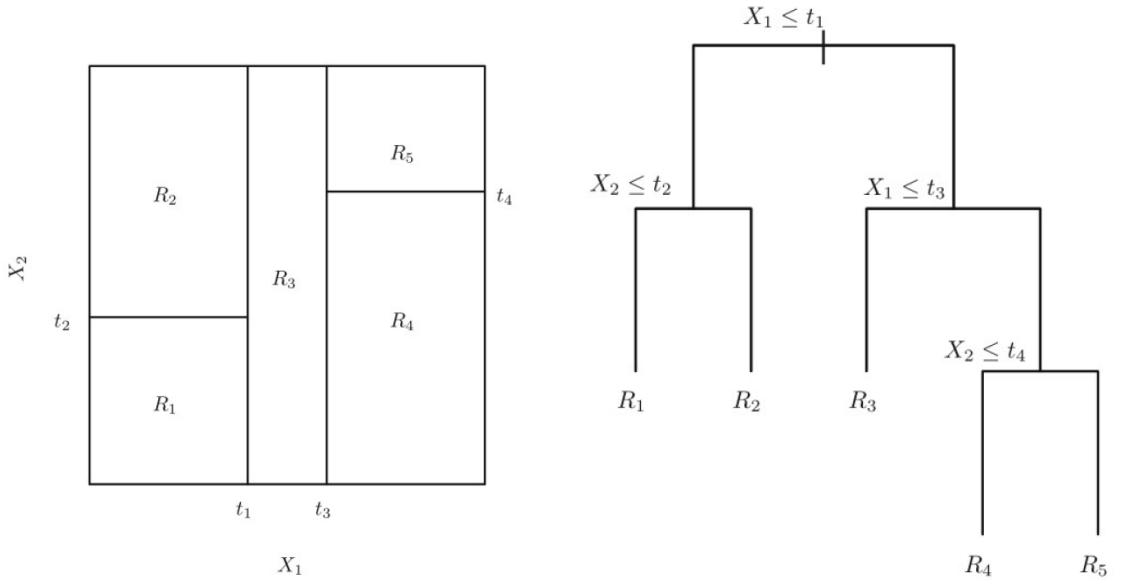
En resumen, cada decisión tomada en el modelo implica una bifurcación, ramificándose en dos posibles direcciones. Aquellos puntos donde se determina que dirección tomar en base a los distintos valores del predictor son denominados nodos internos. Los segmentos que conectan los nodos internos son denominados como ramas. Los nodos finales y en los que ya no existen más divisiones son los ya definidos, que se denotan como $(R_1, R_2, \ldots, R_j)$. Este proceso se puede ver visualmente en la Figura 4.1b.

4.2.1. Metodología CART

A continuación, se explica un método ampliamente conocido tanto para árboles de regresión como para árboles de clasificación, denominado *Classification and Regression Trees* (CART), desarrollado por un grupo de matemáticos de las universidades de Berkeley y Stanford (Breiman, Friedman, Olshen y Stone) a mediados de la década de 1980. Esta técnica propone una segmentación de la base de datos, generando una estructura de árbol compleja, es decir, árboles con una mayor profundidad. El objetivo del proceso de particionamiento recursivo es alcanzar nodos terminales que sean homogéneos. Sin embargo, lograr una homogeneidad completa en los nodos terminales es raramente posible en el análisis de datos reales. Por lo tanto, el criterio de finalización de CART se basa en maximizar la homogeneidad de las variables en los nodos terminales.

Una medida cuantitativa de la homogeneidad es la impureza, que se define como la relación entre el número de sujetos que cumplen una determinada característica en un nodo y el número total de sujetos en dicho nodo. Se selecciona así el punto de corte que conduce a la mayor disminución de la impureza, lo que permite obtener descendientes homogéneos en la variable respuesta. Por lo tanto, es fundamental conocer las distintas condiciones de parada y los criterios de división utilizados.

Para comprender la idea principal de los árboles de decisión, consideremos un ejemplo específico de un problema de regresión que involucra una variable respuesta continua Y y las variables explicativas $X^T = \{X_1, X_2\}$, tomando cada una de ellas valores dentro del intervalo unitario. La idea intuitiva que subyace el algoritmo es dividir el espacio de variables predictoras como se puede ver en la Figura 4.1a.



(a) Secciones del espacio predictor que genera un árbol de decisión James et al. [12].

(b) Árbol de decisión representado en forma de árbol binario James et al. [12].

Figura 4.1: Metodología CART.

En cada elemento de esta partición, se modela la variable respuesta Y utilizando una constante diferente. De esta manera, se elige la variable y el punto de corte que generan un mejor ajuste, lo que puede conducir a una subdivisión adicional de una o ambas regiones en dos o más subregiones. Este proceso continúa hasta que se cumple una condición de finalización. Por ejemplo, en este caso, se selecciona inicialmente el punto de corte $X_1 = t_1$, dividiendo la región $X_1 \leq t_1$ en $X_2 = t_2$ y la región $X_1 > t_1$ en $X_1 = t_3$. Finalmente, la región $X_1 = t_3$ se divide nuevamente en $X_2 = t_4$. Por tanto, el resultado final de este proceso será una partición compuesta por cinco regiones R_i, siendo en este caso $i = 1, 2, \ldots, 5$.

Surge entonces el modelo de regresión, representado en la Figura 4.1b en forma de árbol binario, realiza una partición del espacio de covariables en hiper-rectángulos R_j y la predicción viene dada por

$$\hat{f}(X) = \sum_{j=1}^{J} c_j \mathbb{I}_{\{X \in R_j\}},$$

donde c_j es la constante del valor estimado de todas las observaciones pertenecientes a la región R_j y J el número total de regiones formadas. Concretamente, en el caso tratado representado en la Figura 4.1, $J = 5$.

4.2.2. Árboles de regresión

En los árboles de regresión, se considera un conjunto de p variables explicativas y una variable respuesta Y para cada una de las N observaciones, que denotamos como (X_i, Y_i) para $i = 1, 2, \ldots, N$, donde $Xi = (X_{i,1}, \ldots, X_{i,p})$. El objetivo estadístico consiste en establecer una relación entre la variable objetivo y las variables explicativas de manera que permita predecir Y a partir de los valores de X. En otras palabras, se busca estimar la esperanza condicionada de Y a $X = X_i$, es decir,

$$\mathbb{E}(Y|X = X_i).$$

El algoritmo de árboles de decisión toma decisiones de forma automática sobre las variables y los puntos de división, de la misma forma que sobre la topología o estructura que debe seguir el árbol.

Observando la Figura 4.1b, es fácil apreciar que las regiones R_j se corresponden con los nodos terminales del árbol. Cuando se trata de una variable respuesta continua, es común utilizar el criterio de mínimos cuadrados como medida de impureza en un nodo o región R_j. Este criterio se define como

$$\text{IMP}(R_j) = \sum_{i \in R_j} (Y_i - \hat{f}(X_i))^2,$$

donde $\hat{f}(X_i)$ representa el promedio de los valores de los Y_i pertenecientes a la región R_j. Nótese el abuso de notación presente en $i \in R_j$, que se refiere a las observaciones $i \in N$ tales que $X_i \in R_j$.

Sin embargo, encontrar todas las posibles divisiones para el conjunto de variables en función de mínimos cuadrados no es computacionalmente viable. Por ello, se recurre al *Recursive Binary Splitting*, un algoritmo de partición basado en ir realizando particiones binarias que, en cada iteración, trata de hallar una variable explicativa X_j y un punto de corte t tal que las observaciones halladas en cada una de las regiones,

$$R_1(X_j, t) = \{X|X_j \leq t\} \quad \text{y} \quad R_2(X_j, t) = \{X|X_j > t\},$$

obtengan la máxima reducción viable de la impureza. Es decir, se seleccionan j y t de forma que se minimice la siguiente expresión:

$$\sum_{i \in R_1} (Y_i - \hat{f}_{R_1}(X_i))^2 + \sum_{i \in R_2} (Y_i - \hat{f}_{R_2}(X_i))^2.$$

A diferencia del problema original, este se soluciona rápidamente. Se repetirá el proceso en cada una de las dos regiones R_1 y R_2 sucesivamente hasta alcanzar un criterio de parada.

4.2.3. Árboles de clasificación

La variable respuesta Y de un problema de clasificación, la cual toma los valores $1, 2, \ldots, K$, etiquetas que identifican las K categorías posibles. El proceso a seguir será casi idéntico al seguido para los árboles de regresión, explicado en la Sección 4.2.2, salvo algunas diferencias.

La primera de ellas será sobre la denominada media aritmética $\hat{f}(X_i)$, que era la usada para representar a los valores pertenecientes a una misma región. En este caso, se utilizará la categoría modal de la región, es decir, la clase con mayor número de participantes dentro de esa misma región. Esta será el valor que se dará en la predicción a los registros que caigan en la misma.

A continuación, se nos presenta la siguiente diferencia. No será posible utilizar el error cuadrático medio como medida de impureza para realizar la división. Por ello, será necesario adoptar otra medida adecuada para el contexto actual.

Fijada una región j, se denota por $\hat{p}_{j,k}$, con $k = 1, \ldots, K$ a la proporción de observaciones de la muestra de entrenamiento en dicha región que pertenecen a la categoría k. Es decir, dada una región R_j con N_j observaciones de X_i, se tiene

$$\hat{p}_{j,k} = \frac{1}{N_j} \sum_{X_i \in R_j} \mathbb{I}_{\{Y_i = k\}}.$$

Las tres medidas de impureza más utilizadas en el contexto de los árboles de clasificación son las siguientes:

- **Error de clasificación:**
$$\text{IMP}(R_j) = 1 - \max_k(\hat{p}_{j,k}).$$
- **Índice de Gini:**
$$\text{IMP}(R_j) = \sum_{k=1}^{K} \hat{p}_{j,k}(1 - \hat{p}_{j,k}).$$
- **Entropía:**
$$\text{IMP}(R_j) = -\sum_{k=1}^{K} \hat{p}_{j,k} \log(\hat{p}_{j,k}).$$

El error de clasificación se fundamenta en el porcentaje de observaciones que no pertenecen a la clase predominante en un nodo determinado. El índice de Gini es una métrica que evalúa con qué frecuencia un elemento seleccionado al azar se identifica erróneamente; el objetivo es obtener un valor bajo, ya que esto indica una mayor pureza en el nodo. Este enfoque de división se aplica en diversas técnicas, como el Random Forest, de acuerdo con Breiman [13]. Por otro lado, la entropía es una forma de cuantificar la dispersión de cada clase. Si una división presenta valores de una sola clase, se considera una división pura y, por lo tanto, tiene un valor de cero. Sin embargo, cuando la frecuencia de cada clase es equitativa, la entropía alcanza su valor máximo, que es 1. Por ende, cuanto menor sea el valor de la entropía, mejores resultados se obtendrán.

Aunque la proporción de errores de clasificación resulta ser la medida de error más intuitiva, en la práctica solo se utiliza durante la etapa de poda. Cabe destacar que en el cálculo de esta medida solo se toma en cuenta la proporción máxima estimada, $\max_k(\hat{p}_{j,k})$, mientras que en las medidas alternativas se consideran las proporciones $\hat{p}_{j,k}$ correspondientes a todas las categorías. Durante la fase de crecimiento, se emplea indistintamente el índice de Gini o la entropía.

Cuando nos interesa evaluar el error no solo en una única región, sino en varias (ya sea al dividir un nodo en dos o al considerar todos los nodos terminales), se suman los errores de cada región, teniendo en cuenta el peso determinado por el número de observaciones en cada una de ellas.

4.2.4. Como evitar el sobreajuste

Los árboles de regresión y clasificación tienen una importante limitación común: si no se restringe el número de decisiones, tienden a ajustarse completamente a las observaciones de entrenamiento, creando un nodo terminal por cada observación, teniendo así un 100 % de precisión en el conjunto de datos de entrenamiento. Esto se conoce como sobreajuste u overfitting. Para controlar este problema, existen dos estrategias:

- **Parada temprana o early stopping**
  Se utiliza un conjunto de hiperparámetros para limitar el crecimiento del árbol y el número de nodos que se crean. Estre estos hiperparámetros se incluyen establecer un número mínimo de observaciones para generar una división, establecer un límite máximo de divisiones para la rama más larga, limitar el número máximo de nodos terminales o exigir una reducción mínima del error para que se lleve a cabo una bifurcación. Las implementaciones y algoritmos pueden incluir otras medidas adicionales para controlar los árboles. Sin embargo, la desventaja de esta estrategia es que no considera las decisiones que podrían tomarse después de la división actual, lo que dificulta alcanzar el árbol óptimo.

- **Podado o tree pruning**
  En esta estrategia, se genera un árbol inicialmente limitado en tamaño y luego se evalúa en busca de la estructura óptima que se mantenga robusta y logre un bajo error de prueba, buscando así llegar al sub-árbol óptimo que proporciona el mejor rendimiento.

4.3. Métodos Bagging y Boosting

Tanto el *Bagging* como el *Boosting* son dos enfoques generales utilizados para reducir la varianza en métodos de aprendizaje estadístico. La premisa básica implica la combinación de múltiples métodos de predicción simples (denominados como débiles), que poseen una capacidad predictiva limitada, con el objetivo de obtener un método de predicción potente y robusto. Estas técnicas se aplican tanto a problemas de regresión como de clasificación.

Por ejemplo, los árboles de decisión, que son predictores débiles y rápidos de generar, son muy empleados en estas técnicas. El procedimiento consiste en construir múltiples modelos (creciendo numerosos árboles) y luego combinarlos para generar predicciones, ya sea mediante promedio o por consenso.

Sin embargo, los árboles de decisión a menudo enfrentan desafíos en el equilibrio entre el sesgo y la varianza. Como se vió en la Sección 4.2, a medida que los árboles se vuelven más complejos, es decir, a medida que aumentan las ramificaciones, se reduce el sesgo debido a una mayor similitud con el conjunto de entrenamiento. Sin embargo, esto también aumenta la varianza, ya que el árbol tiende a no ser lo suficientemente preciso en las predicciones futuras (sobreajuste). Construir árboles más simples no es una opción viable, ya que no representarían adecuadamente la combinación de variables, lo que resultaría en un sesgo significativo al no asemejarse a las predicciones del conjunto de entrenamiento, y una baja varianza debido a la simplicidad de la estructura. Se puede ver la representación visual de esta situación en la Figura 4.2.

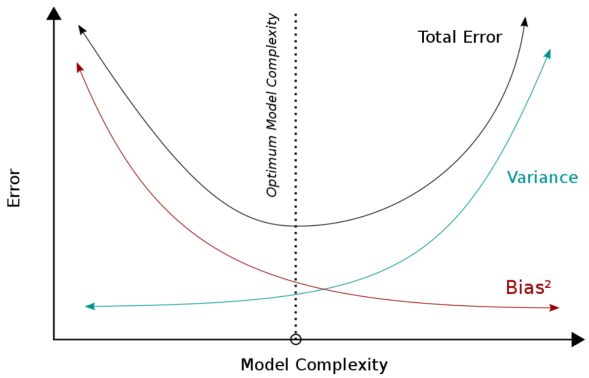


Figura 4.2: Equilibrio entre sesgo y varianza Fernández-Casal et al. [7].

Se introducen entonces los métodos de ensamblado, o *ensemble*, basados en encontrar un equilibrio entre el sesgo y la varianza del modelo, a través de la creación de múltiples árboles.
En el método *Bagging*, los modelos empleados tendrán muy poco sesgo pero mucha varianza, combinando múltiples modelos de este tipo se consigue reducir la varianza sin apenas aumentar el sesgo. En cambio, en el método *Boosting* los modelos empleados poseen muy poca varianza pero mucho sesgo, de forma que ajustando secuencialmente muchos modelos se llega a reducir el sesgo.

En esta sección se explicará el *Bagging*, mientras que el *Boosting* será detallado en la Sección 4.4.

4.3.1. Bagging

A partir de la década de los años 90, surgieron los métodos *ensemble* (o métodos combinados), los cuales se caracterizan por combinar múltiples modelos predictivos con el fin de mejorar las predicciones. Uno de los primeros métodos combinados en ser utilizado fue el *Bagging* (*Bootstrap Aggregation*), presentado por Breiman [13]. Este enfoque busca reducir la varianza, basándose en la combinación del *bootstrap* junto con un modelo de regresión o clasificación, como por ejemplo, un árbol de decisión.

La idea subyacente es simple. Si se cuenta con numerosas muestras de entrenamiento, cada una de ellas puede utilizarse para entrenar un modelo, y luego combinar todas estas predicciones para obtener una predicción final. Mediante este proceso se generan tantas predicciones como modelos y, por ende, tantas predicciones como muestras de entrenamiento. El procedimiento consiste en promediar o tomar consenso de todas las predicciones obtenidas, lo cual presenta dos beneficios clave: simplifica la solución y reduce significativamente la varianza.

No obstante, en la práctica, por lo general se dispone únicamente de una única muestra de entrenamiento, $Z = \{(X_1, Y_1), \ldots, (X_N, Y_N)\}$. Aquí es donde el *bootstrap* entra en juego, una técnica especialmente útil para estimar varianzas y que, en este contexto, se utiliza para reducirlas. Se generan cientos o miles de muestras *bootstrap* Z_b^*, con $b = 1, \ldots, B$, a partir de la muestra de entrenamiento original Z y, posteriormente, cada una de estas muestras se emplea como conjunto de entrenamiento individual (conjunto de datos *bootstrap*), de forma que, para cada una, se genera la predicción $\hat{f}_b^*(X)$.

Para modelos con baja variabilidad inherente, como la regresión lineal, el uso del bagging puede resultar poco interesante, ya que existe un margen limitado para la mejora de rendimiento. No obstante, este método adquiere una gran relevancia en el caso de los árboles de decisión, dado que un árbol con una profundidad considerable (sin podar) presenta una alta variabilidad: incluso una pequeña modificación en los datos de entrenamiento puede generar un árbol completamente distinto al anterior, lo cual se percibe como una desventaja. Por esta razón, el *Bagging* se ajusta perfectamente a este contexto.

En el caso de los árboles de regresión, se generan B árboles sin podar y se calcula la media de las predicciones, es decir,

$$\hat{f}_{bag}(X) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b^*(X). \tag{4.15}$$

En cambio, en los árboles de clasificación, se utiliza el criterio de voto mayoritario, reemplazando el promedio por la moda. En este, cada modelo tiene el mismo peso y por tanto contribuye con un voto al total. Además, la proporción de votos para cada categoría se interpreta como una estimación de su probabilidad, denotada como $\hat{p}_k^*$ para cada clase k. Por tanto, la predicción en este caso será el clasificador

$$\hat{G}_{bag}(X) = \arg\max_k \hat{p}_k^*(X) \tag{4.16}$$

Una ventaja adicional del *Bagging* radica en su capacidad para estimar directamente el error de predicción, sin requerir la utilización de un conjunto de prueba o la aplicación de técnicas como la validación cruzada o, de nuevo, remuestreo, obteniendo un resultado similar a lo que se obtendría con dichos métodos. Se sabe que una muestra *Bootstrap* contiene múltiples observaciones repetidas y, en promedio, utiliza aproximadamente dos tercios de los datos. Un dato que no se utiliza en la construcción un árbol se denomina dato out-of-bag (OOB) , que constituyen el tercio restante, resultado que viene dado por el límite:

$$\lim_{N\to\infty}\left(1-\frac{1}{N}\right)^{N}=e^{-1}\approx 0{,}36.$$

De esta manera, para cada observación, se pueden utilizar los árboles para los cuales esa observación es OOB (aproximadamente un tercio de los árboles construidos) para generar una única predicción para ella. Repitiendo este proceso para todas las observaciones, se obtiene una medida del error.

Una elección a tener en cuenta es la cantidad de muestras bootstrap a utilizar (o, en otras palabras, cuántos árboles se deben construir). En realidad, esto se trata de una aproximación Monte Carlo, por lo que generalmente se examina la convergencia del error OOB al aumentar el número de árboles. Si parece haber convergencia con unos pocos cientos de árboles, el nivel de error no variará significativamente al incrementar el número. Por lo tanto, aumentar considerablemente la cantidad de árboles no mejora las predicciones, aunque tampoco aumenta el riesgo de sobreajuste. Los costes computacionales incrementan con la cantidad de árboles, pero la construcción y evaluación del modelo se pueden paralelizar fácilmente (aunque puede requerir una gran cantidad de memoria si el conjunto de datos es muy grande). Por otro lado, si el número de árboles es demasiado pequeño, es posible obtener pocas o incluso ninguna predicción OOB para algunas de las observaciones en el conjunto de entrenamiento.

Aunque, al utilizar *Bagging*, se mejora significativamente la capacidad de predicción en comparación con los modelos de árbol de decisión, se asume el coste de perder interpretabilidad. No obstante, existen métodos para calcular la importancia de los predictores. Por ejemplo, al fijar un predictor y una medida de error, se puede medir la reducción del error cada vez que se realiza una división que utiliza ese predictor específico en cada uno de los árboles. Al promediar esta medida en todos los árboles *Bagging*, se obtiene una medida global de importancia:, donde un alto valor en la reducción del error indica que el predictor es importante.

4.3.2. Random Forest

Los bosques aleatorios, o *random forest*, son una variante del método de *Bagging*, especialmente diseñados para trabajar con árboles de decisión. El *Bagging* utiliza muestras *Bootstrap*, generando un elemento de aleatoriedad para así provocar árboles diferentes. Sin embargo, en ocasiones estos árboles no son lo suficientemente distintos.

Es común que los árboles tengan estructuras similares en la parte superior y que se vayan diferenciando a medida que se profundiza en ellos. Este fenómeno se conoce como correlación entre árboles y ocurre cuando el árbol es un buen modelo para describir la relación entre los predictores y la respuesta, y cuando un predictor en particular es altamente relevante y se encuentra en los cortes iniciales. Esta correlación entre árboles se traduce en una correlación entre sus predicciones.

El promedio de variables altamente correlacionadas reduce de forma mucho menor la varianza que si se promedian variables no correlacionadas. Para solucionar esto, se introduce aleatoriedad en el proceso de construcción de los árboles para eliminar la posible correlación entre ellos. Se han propuesto diferentes enfoques, y uno de los más destacados es la idea de Dietterich [14] de introducir aleatoriedad

en la selección de variables de cada corte. Por otro lado, Breiman [15] propuso un algoritmo unificado llamado bosques aleatorios, que combina el *Bagging* con la metodología CART, explicada en la Sección 4.2.1 .

En los bosques aleatorios, durante la construcción de cada árbol del bosque, se realizan cortes binarios y se debe seleccionar una variable predictora en cada corte. La modificación introducida consiste en seleccionar, de entre todas las p variables predictoras, al azar $m < p$ predictores antes de realizar cada corte, siendo m los candidatos para corte. El hiperparámetro m puede seleccionarse utilizando técnicas habituales, como elegir $m = \sqrt{p}$ para problemas de clasificación o $m = \frac{p}{3}$ para problemas de regresión. El número de árboles en el bosque también puede considerarse un hiperparámetro, aunque a menudo se establece experimentalmente para lograr convergencia. En general, se requieren más árboles en los bosques aleatorios en comparación con el *Bagging*.

En términos de eficiencia computacional, los bosques aleatorios son más eficientes que el *Bagging*, ya que la construcción de cada árbol es más rápida al evaluar solo unos pocos predictores en cada corte. Aunque se necesitan más árboles en total, el tiempo de construcción total es menor.

Se describe entonces el algoritmo utilizado por el *Random forest* para árboles tanto de regresión como de clasificación:

Algoritmo *Random Forest*

El primer paso será, para $b = 1, \ldots, B$, construir B muestras *Bootstrap* $Z_1^*, \ldots, Z_B^*$ independientes mediante remuestreo. Acto seguido, se crean, de forma independiente, B árboles *Random Forest*, uno para cada una de las muestras, repitiendo de forma recursiva los pasos detallados a continuación para cada nodo del árbol hasta alcanzar el tamaño mínimo de nodo $n_{\text{mín}}$.

- Seleccionar m variables al azar de entre las p posibles, siguiendo alguno de los criterios detallados anteriormente.
- Se elige la mejor variable y el mejor punto de división entre los m posibles.
- Se divide el nodo en dos regiones.

Posteriormente, se realizan , de cada uno de los B árboles creados, las predicciones individuales , es decir, los estimadores $\hat{f}_1^*(X), \ldots, \hat{f}_b^*(X)$. Se calcula entonces el estimador de predicción *Bagging* definido en la Ecuación 4.15 para el caso de un árbol de regresión y en la Ecuación 4.16 en el caso de un árbol de clasificación.

Para más información acerca del *Random forest* y sus características, consultar el Capítulo 15 de Hastie et al. [16].

Hiperparámetros del modelo

Es de suma importancia asegurarse de ajustar adecuadamente los valores prefijados (hiperparámetros) antes de proceder a la construcción del árbol utilizando este método. Los hiperparámetros más importantes que se requieren son el número de variables candidatas que se evalúan en cada bifurcación y el número de árboles individuales que conforman el algoritmo. Aun así, también existen otros hiperparámetros adicionales que deben tenerse en cuenta:

- **Número de árboles B (*ntree*)**
  Tiene un impacto directo en la precisión de la predicción, por tanto el objetivo es encontrar un

número óptimo de árboles que minimice el error, ya que emplear un exceso de árboles puede generar un alto costo computacional. Por defecto, en el método Random Forest se considera $B = 500$ como valor estándar.

- **Número mínimo de variables para cada bifurcación (*mtry*)**
  Determina la cantidad de variables predictoras, m, seleccionadas aleatoriamente en cada árbol. Al reducir el valor de *mtry*, disminuye la correlación entre los árboles, ya que en cada nodo se reducen las posibilidades de selección de variables candidatas para la bifurcación. Sin embargo, una disminución excesiva de este valor puede reducir la precisión de cada árbol individual.
  Se recomienda utilizar los siguientes valores de referencia para *mtry*: $\sqrt{p}$ para problemas de clasificación y $\frac{p}{3}$ para problemas de regresión. En la práctica, la elección de este hiperparámetro depende del problema específico, por lo tanto, es recomendable realizar simulaciones con diferentes valores.

- **Muestras mínimas (*min_rows*)**
  Establece el número mínimo de muestras requeridas en los nodos terminales. Es importante encontrar un equilibrio entre el sesgo y la varianza. En general, se recomienda utilizar un valor pequeño: por defecto, se emplea *min_rows*= 1 para clasificación y *min_rows*= 5 para regresión.

- **Profundidad máxima (*max_depth*)**
  Determina la profundidad máxima del árbol.

- **Tamaño de la muestra *Bootstrap* (*sample_rate*)**
  Especifica la proporción de la muestra utilizada para entrenar cada árbol. Valores más pequeños generarán árboles con una menor correlación entre ellos. Sin embargo, esto también supondrá una reducción de cada árbol individual. Toma valores en $(0, 1]$ y su valor por defecto es 0,6320000291.

4.4. Boosting

El *Boosting* es una técnica de modelización de aprendizaje lento en la que se combinan muchos modelos obtenidos mediante un método con poca capacidad predictiva para, impulsados, dar lugar a un mejor predictor. Los árboles de decisión pequeños (construidos con poca profundidad) resultan perfectos para esta tarea, al ser realmente malos predictores (weak learners), fáciles de combinar y generarse de forma muy rápida. Es decir, se ajustan de manera individual B árboles, los cuales presentan un sesgo considerable pero una baja varianza, asegurándose de que cada nuevo árbol no cometa los mismos errores que el anterior, mejorando así en cada iteración.

A diferencia del Bagging, no implica la creación de múltiples versiones del conjunto de entrenamiento, sino que se trabaja secuencialmente utilizando siempre el mismo conjunto de datos de entrada y manipulando los pesos de los datos para generar modelos distintos.

La idea del *Boosting* fue desarrollada por Valiant [17] y Kearns y Valiant [18], pero encontrar una implementación efectiva fue una tarea difícil que no se resolvió satisfactoriamente hasta que Freund y Schapire [19] presentaron el algoritmo *AdaBoost*, que rápidamente se convirtió en un éxito.

El algoritmo *AdaBoost* está planteado como una solución mediante ensamblado a un problema de clasificación binaria. Por ello, no se entrará en detalle en este algoritmo, ya que los métodos necesitados en este trabajo requieren clasificación multiclase. Si se desea más información sobre este algoritmo, se recomienda el Capítulo 10 de Hastie et al. [16].

4.4.1. Gradient Boosting

El método de *Gradient Boosting* es una extensión del algoritmo *AdaBoost* que permite la utilización de cualquier función de costo, siempre y cuando esta sea diferenciable. Esta flexibilidad ha permitido la aplicación del *Boosting* a una variedad de problemas, como regresión y clasificación multiclase.

Diversos algoritmos de regresión, incluyendo los árboles de decisión, se enfocan en minimizar funciones de residuos, como la suma de los cuadrados del error residual (SSE), el error cuadrático medio (MSE) o la raíz del error cuadrático medio (RMSE). En lugar de reducir el error ajustando los errores del árbol anterior, como ocurre en AdaBoost, el Gradient Boosting optimiza secuencialmente la función de pérdida. Se detiene cuando alcanza un nivel aceptable o deja de mejorar al aplicarse a un conjunto de datos de validación. Esta adaptación específica del Gradient Boosting minimiza la función de pérdida del error cuadrático medio (SSE), utilizando el gradiente como el error residual.

A diferencia de otros enfoques, en ocasiones es necesario considerar diferentes funciones de pérdida, como el error absoluto medio (MAE), que es menos sensible a valores atípicos. El término "Gradient Boosting Machine"surge de la capacidad de generalizar este procedimiento a funciones de pérdida que no se basan en SSE.

El Gradient Boosting se clasifica como un algoritmo iterativo de descenso de gradientes, un método de optimización altamente versátil para encontrar soluciones óptimas en una amplia gama de problemas.

Sea $L(Y, f(X))$ la función de pérdida asociada a la función de predicción $f(X)$ sobre Y a partir de los valores de X, el algoritmo *Gradient Boosting* para problemas de regresión se describe como sigue:

1. Se inicializa el modelo con un valor constante:
$$f_0(X) = \arg\min_{\gamma} \sum_{i=1}^{N} L(Y_i, \gamma).$$
2. Para $b = 1, \ldots, B$:
   2.1 Para $i = 1, \ldots, N$ se calcula
   $$r_{ib} = -\left[\frac{\partial L(Y_i, f(X_i))}{\partial f(X_i)}\right]_{f=f_{b-1}}.$$
   2.2 Se ajusta un árbol de regresión para los residuos r_{ib}, dando lugar a las regiones terminales R_{ib}, siendo $j = 1, \ldots, J_b$.
   2.3 Para $j = 1, \ldots, J_b$ se calcula
   $$\gamma_{jb} = \arg\min_{\gamma} \sum_{X_i \in R_{jb}} L(Y_i, f_{b-1}(X_i) + \gamma).$$
   2.4 Se actualiza $f_b(X) = f_{b-1}(X) + \sum_{j=1}^{J_b} \gamma_{jb} \mathbb{I}_{\{X \in R_{jb}\}}$.
3. Se obtiene $\hat{f}(X) = f_B(X)$.

En un problema de clasificación el algoritmo seguirá los mismos pasos descritos, repitiendose los pasos 2 y 3 para cada una de las K clases, donde se calculará para cada iteración el residuo

$$r_{ikb} = Y_i - p_k(X_i), \text{ con } p_k(X) = \frac{e^{f_k(X)}}{\sum_{l=1}^{K} e^{f_l(X)}}.$$

Finalmente, en el paso 3 se obtendrá $f_{kB}(X)$, $k = 1, \ldots, K$.

4.4.2. Extreme Gradient Boosting

El *Extreme Gradient Boosting*, tambien conocido como *XGBoost*, es una de las implementaciones más populares y eficientes de *Boosting* en la actualidad, propuesto por Chen y Guestrin [20].

El *XGBoost* toma como función objetivo la función de pérdida, igual que el *Gradient Boosting*, explicado en la Sección 4.4.1. Sin embargo, en este algoritmo se introducen divisiones adicionales dentro de cada árbol creado para reducir así la función objetivo, es decir, la función de pérdida. Al contrario que en otros modelos, cada nodo del árbol tiene una puntuación asociada. Estas puntuaciones de predicción de cada árbol se suman para obtener una puntuación final, lo que permite que los árboles se complementen entre sí y generen un modelo con una estructura más eficiente. Es un metodo más complejo que el anterior que, entre otras modificaciones, utiliza una función de pérdida con una penalización por complejidad y, para evitar el sobreajuste, regulariza utilizando la hessiana de la función de pérdida (necesita calcular las derivadas parciales de primer y de segundo orden), e incorpora parámetros de regularización adicionales para evitar el sobreajuste.

Para entrenar el modelo, al igual que otros modelos *Boosting*, se utiliza una estrategia aditiva donde se fija lo aprendido hasta el momento y se agrega un nuevo árbol en cada paso. De esta manera, se puede expresar el valor de predicción en cada paso de la siguiente forma:

$$\begin{aligned}\hat{f}_0(X_i) =& 0,\\ \hat{f}_1(X_i) =& \hat{f}_0(X_i) + f_1(X_i) = f_1(X_i),\\ &\vdots\\ \hat{f}_B(X_i) =& \sum_{b=1}^{B} f_b(X_i) = \hat{f}_{B-1}(X_i) + f_B(X_i).\end{aligned}$$

Se define entonces

$$L^{(b)}(f) = \sum_{i=1}^{N} L(Y_i, \hat{f}_b(X_i)) + \sum_{b=1}^{B} \Omega(f_b), \tag{4.17}$$

siendo $f_{b-1}(X_i)$ la salida de las combinaciones de los otros $b-1$ árboles dada la overvación i e $\Omega(f_b)$ es un termino que penaliza la complejidad asociada al árbol b.

En un contexto en el que la función dada en la Ecucación 4.17 es convexa y dos veces diferenciable, se puede realizar una expansión de Taylor de segundo orden para resolver:

$$L^{(b)} \approx \sum_{i=1}^{N} \left[L(Y_i, \hat{f}_b(X_i)) + g_i f_b(X_i) + \frac{1}{2} h_i f_b(X_i)^2 \right] + \sum_{b=1}^{B} \Omega(f_b), \tag{4.18}$$

donde

$$g_i = \frac{\partial}{\partial z} L(Y_i, z)\Big|_{z=\hat{f}_{b-1}(X_i)} \quad , \quad h_i = \frac{\partial^2}{\partial^2 z} L(Y_i, z)\Big|_{z=\hat{f}_{b-1}(X_i)}$$

se corresponden con las derivadas de primer y segundo orden de la función de pérdida.

La complejidad de cada árbol b, anteriormente denotada como $\Omega(f_b)$, se define como

$$\Omega(f_b) = \gamma B + \frac{1}{2}\lambda ||\omega||^2.$$

Sustituyendo en la versión simplificada de la Ecuación 4.18, se obtiene el valor de la función objetivo para cada árbol b:

$$\tilde{L}^{(b)} = \sum_{b=1}^{B} \left[\left(\sum_{i \in \mathbb{I}_b} g_i \right) \omega_b + \frac{1}{2} \left(\sum_{i \in \mathbb{I}_b} h_i + \lambda \right) \omega_b^2 \right] + \gamma B, \tag{4.19}$$

donde ω_b representa la puntuación en cada nodo y se elige de forma que su valor óptimo se corresponde con

$$\omega_b^* = -\frac{\sum_{i\in\mathbb{I}_b} g_i}{\sum_{i\in\mathbb{I}_b} h_i + \lambda}.$$

Finalmente, se cuantifica la disminución en el valor de la función objetivo al producirse la ramificación mediante la Ecuación 4.20.

$$\begin{aligned} L_{\text{split}} &= \frac{1}{2}\left[\frac{\left(\sum_{i\in\mathbb{I}_L} g_i\right)^2}{\sum_{i\in\mathbb{I}_L} h_i + \lambda} + \frac{\left(\sum_{i\in\mathbb{I}_R} g_i\right)^2}{\sum_{i\in\mathbb{I}_R} h_i + \lambda} - \frac{\left(\sum_{i\in\mathbb{I}_L+R} g_i\right)^2}{\sum_{i\in\mathbb{I}_L+R} h_i + \lambda}\right] - \gamma \\ &= \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G_L+R^2}{H_L+R+\lambda}\right] - \gamma. \end{aligned} \tag{4.20}$$

En esta, se nos presenta la puntuación del nodo de la izquierda (L), del de la derecha (R) y la del nodo original (L+R). Tambien se encuentra la regularización en el nodo adicional, γ.

Hiperparámetros del modelo

- **Número de iteraciones (*nrounds*)**
  Número de árboles considerados para construir el modelo.

- **Tasa de aprendizaje (*eta*)**
  Magnitud con valor en el intervalo $[0, 1]$ en la que se reduce el peso de la estimación proporcionada en cada árbol para que el proceso sea más eficaz y no haya sobreajuste.
  Un valor elevado provoca que se llegue más rápido al mínimo de la función objetivo, pero aumenta el riesgo de sobreajuste, mientras que un valor más bajo aumentará la lentitud, causando incluso que no se llegue a alcanzar el mínimo. El valor establecido por defecto es 0,3.

- **Reducción de la pérdida mínima (*gamma*)**
  Establece la pérdida mínima necesaria para realizar la siguiente partición en un nodo del árbol de decisión.
  Toma valores en el intervalo $[0, \infty)$, donde valores más altos causarán modelos más conservadores. El valor establecido por defecto es 0.

- **Pureza mínima en los nodos (*min_child_weight*)**
  Criterio de parada que establece, a partir de la hessiana, la impureza mínima necesaria para poder continuar realizando divisiones de los nodos del arbol.
  Toma valores en el intervalo $[0, \infty)$ y su valor establecido por defecto es 1.

- **Peso delta máximo (*max_delta_step*)**
  Normalmente no es necesario, pero puede ser de utilidad en ciertos casos en los que las diferentes clases están extremadamente desequilibradas.
  Toma valores en el intervalo $[0, \infty)$

- **Profundidad máxima de los árboles (*max_depth*)**
  Establece el número máximo de nodos de bifurcación de los árboles de decisión usados en el entrenamiento, es decir, su profundidad. Valores elevados llevan a un modelo más propenso al sobreajuste, ya que las divisiones pierden relevancia.
  Toma valores en el intervalo $[0, \infty)$ y su valor establecido por defecto es 6.

- **Fracción de observaciones (*subsample*)**
  Proporción de la submuestra utilizada en cada paso para construir los árboles de decisión. Si

se toma el valor 0,5 se muestrea al azar la mitad de los datos de entrenamiento, evitando así el sobreajuste.
Toma valores en el intervalo $(0, 1]$ y su valor establecido por defecto es 1.

- **Ratio de variables predictoras por árbol (*colsample_bytree*)**
  Fracción de variables predictoras a utilizar por cada árbol.
  Toma valores en el intervalo $(0, 1]$ y su valor establecido por defecto es 1.

- **Ratio de variables predictoras por nivel (*colsample_bylevel*)**
  Proporción de variables predictoras a utilizar para cada nivel.
  Toma valores en el intervalo $(0, 1]$ y su valor establecido por defecto es 1.

- **Términos de regularización (*lambda* y *alpha*)**
  Términos de regularización que controlan la penalización de tipo L2 y L1, respectivamente. Un aumento en su valor provocará un modelo más conservador.
  Sus valores por defecto son 1 y 0 respectivamente.

- **Estrategia de crecimiento del árbol (*grow_policy*)**
  Método escogido para el crecimiento del árbol. El establecido por defecto, *depthwise*, escoge una forma equilibrada eligiendo nodos de poca profundidad para realizar las divisiones. Por otro lado, *lossguide* escogerá los nodos con mejor ganancia, independientemente de su nivel.

4.4.3. LightGBM

Con el objetivo de mejorar los algoritmos ya existentes, Ke et al. [21] proponen el software *LightGBM* (*Light Gradient Boosted Machine*), en colaboración con Microsoft.

Una de las principales ventajas del *LightGBM* es el tiempo de computación necesario, muy inferior en promedio al de otros algoritmos como lo puede ser el *XGBoost*. Concretamente, muestran que *LightGBM* puede llegar a ser hasta 20 veces más rápido que otros algoritmos de *Gradient Boosting*.

El algoritmo *LightGBM* se distingue de manera fundamental de otros algoritmos en la forma en que se desarrollan los árboles de decisión. En contraste con el *XGBoost*, que hace crecer los árboles de manera gradual por niveles, *LightGBM* selecciona los nodos que generan la mayor ganancia, lo que resulta en árboles, aunque más precisos, menos equilibrados y con mayor riesgo de sobreajuste. Aunque *XGBoost* ha incorporado este nuevo método en los últimos años, solo se aplica al algoritmo de búsqueda de particiones basado en histogramas.

Para resolver problemas de eficiencia, *LightGBM* propone reducir tanto el número de observaciones como el de variables predictoras consideradas en cada iteración para seleccionar el punto de corte correspondiente. Esta estrategia se basa en dos nuevas técnicas llamadas GOOS (*Gradient-based One-Side Sampling*) y EFB (*Exclusive Feature Bundling*).

La técnica GOOS se fundamenta en que las observaciones con gradientes de la función de pérdida más altos son las que más influyen en el aumento de la ganancia al realizar un corte en el árbol de decisión. Por tanto, con el objetivo de reducir el número de observaciones a considerar sin perder información relevante para la ganancia, se propone un muestreo en el cual se mantienen las observaciones con gradientes más altos y se selecciona una muestra aleatoria de aquellas con gradientes más bajos. Luego, se ajusta el error asociado a estas últimas observaciones para que tengan un peso equivalente al de todas las observaciones con gradientes bajos.

Supongamos que O representa el conjunto de datos de entrenamiento en un nodo específico del árbol. La ganancia que se obtiene al realizar una partición utilizando la variable predictora j-ésima en el punto de corte d para dicho nodo se puede medir de forma precisa mediante el término $V_{j|O}(d)$, que se define como sigue:

$$V_{j|O}(d) = \frac{1}{n_O}\left(\frac{\left(\sum_{\{x_i\in O: x_{ij}\leq d\}} g_i\right)^2}{n^j_{l|O}} + \frac{\left(\sum_{\{x_i\in O: x_{ij}> d\}} g_i\right)^2}{n^j_{r|O}}\right),$$

donde $n_O = \sum_{i=1}^n \mathbb{I}_{\{x_i\in O\}}$ es el número de observaciones original del nodo, $n^j_{l|O} = \sum_{i=1}^n \mathbb{I}_{\{x_i\in O: x_{ij}\leq d\}}$ es el número de observaciones del nodo izquierdo después de la bifurcación del nodo original, de la misma forma $n^j_{r|O} = \sum_{i=1}^n \mathbb{I}_{\{x_i\in O: x_{ij}> d\}}$ es el número de observaciones del nodo derecho y g_i es el opuesto del gradiente de la función de pérdida correspondiente evaluado en la predicción proporcionada por el modelo para la observación i-ésima.
La técnica GOOS propone la aproximación $\tilde{V}_j(d)$, resultante de el siguiente procedimiento:

1. Ordenar las observaciones según el valor absoluto de su gradiente asociado.
2. Tomar un conjunto A de observaciones de forma que sea equivalente a la proporción a de aquellas observaciones totales con los valores anteriores más altos.
3. Tomar una proporción b de observaciones del conjunto complementario a A, formando así un nuevo conjunto B.
4. Obtener una aproximación de la ganancia dada al realizar una partición en el punto de corte d con la variable predictora j-ésima mediante:

$$\tilde{V}_j(d) = \frac{1}{n_O}\left(\frac{\left(\sum_{\{x_i\in A: x_{ij}\leq d\}} g_i + \frac{1-a}{b}\sum_{\{x_i\in A: x_{ij}\leq d\}} g_i\right)^2}{n^j_{l|O}(d)} + \frac{\left(\sum_{\{x_i\in A: x_{ij}> d\}} g_i + \frac{1-a}{b}\sum_{\{x_i\in B: x_{ij}> d\}} g_i\right)^2}{n^j_{r|O}(d)}\right).$$

Cabe destacar que la aproximación $\tilde{V}_j(d)$ comete un error que converge a 0 cuando $n \to \infty$, es decir, cuanto mayor sea el número de datos, más precisa será la aproximación.

Por otro lado, la técnica EFB se basa en que, generalmente, los predictores rara vez son distintos de cero simultáneamente, aunque haya un número elevado de variables explicativas en el problema. Se tomarán entonces las variables excluyentes y se tratará de agruparlas, para así poder conseguir un número efectivo de variables predictoras menor.
Así se reduce este problema al denominado de coloración de grafos, tomando las variables predictoras como vértices y añadiendo una arista por cada par de predictores no mutuamente excluyentes. Acto seguido, se resuelve a través de un algoritmo voraz. Una vez determinadas las variables a agrupar, la técnica EFB propone combinarlas a través de un algoritmo basado en histogramas.

Hiperparámetros del modelo

- **Número de iteraciones (*num_iterations*)**
  Número de árboles considerados para construir el modelo.
- **Profundidad máxima de los árboles (*max_depth*)**
  Establece el número máximo de nodos de bifurcación de los árboles de decisión usados en el

entrenamiento, es decir, su profundidad. Valores elevados llevan a un modelo más propenso al sobreajuste, ya que las divisiones pierden relevancia.
Toma valores en el intervalo $[0, \infty)$ y su valor establecido por defecto es 6.

- **Número máximo de hojas por árbol (*num_leaves*)**
  Número máximo de nodos terminales permitidos para cada árbol de decisión generado.
  Toma valores en el intervalo $(1, 131072]$ y su valor establecido por defecto es 31.

- **Número mínimo de observaciones en la hoja de un árbol (*min_data_in_leaf*)**
  Número mínimo de observaciones que debe de tener cada nodo terminal de un árbol de decisión generado.
  Toma valores en el intervalo $[0, \infty)$ y su valor establecido por defecto es 20.

- **Ganancia mínima para la partición (*min_gain_to_split*)**
  Ganancia mínima que debe obtenerse para poder realizar una partición sobre cualquier nodo de un árbol de decisión.
  Toma valores en el intervalo $[0, \infty)$ y su valor establecido por defecto es 0.

- **Tasa de aprendizaje (*learning_rate*)**
  Magnitud con valor en el intervalo $[0, 1]$ en la que se reduce el peso de la estimación proporcionada en cada árbol para que el proceso sea más eficaz y no haya sobreajuste.
  Un valor elevado provoca que se llegue más rápido al mínimo de la función objetivo, pero aumenta el riesgo de sobreajuste, mientras que un valor más bajo aumentará la lentitud, causando incluso que no se llegue a alcanzar el mínimo. El valor establecido por defecto es 0,3.

- **Términos de regularización (*lambda_l1* y *lambda_l2*)**
  Términos de regularización que controlan la penalización de tipo L1 y L2, respectivamente. Un aumento en su valor provocará un modelo más conservador.
  Toman valores en el intervalo $[0, \infty)$ y su valor establecido por defecto es 0.

4.5. Validación y evaluación de los modelos

Una vez construidos los modelos, surge la necesidad de evaluar sus resultados, comprobando así la fiabilidad y precisión de estos. Existen varias métricas que se podrían usar, que serán explicadas en esta sección.

Como la variable objetivo en un problema de clasificación es categórica, para evaluar un modelo sobre un conjunto de datos particular, lo habitual es considerar las predicciones asociadas a las observaciones y construir una tabla de contingencia frente a las categorías reales que es lo que se conoce en este contexto como matriz de confusión.

	Predicción	
Observación	$\hat{G} = k$	$\hat{G} \neq k$
$Y = k$	Verdaderos positivos (TP_k)	Falsos negativos (FN_k)
$Y \neq k$	Falsos positivos (FP_k)	Verdaderos negativos (TN_k)

Tabla 4.1: Matriz de confusión asociada a la categoría k.

Para ello, al encontrarnos en el contexto de una variable objetivo con más de dos clases, se adoptará un enfoque denominado *One vs All* (uno contra todos). Así se puede ver en la matriz de confusión representada en la Tabla 4.1 los siguientes conceptos, dada una categoría k con $k = 1, \ldots, K$:

- **Verdaderos positivos:** Número de observaciones de la categoría k que el modelo ha clasificado en ella.
- **Falsos negativos:** Número de observaciones pertenecientes a la categoría k que el modelo ha clasificado incorrectamente.
- **Verdaderos negativos:** Número de observaciones no pertenecientes a la categoría k que el modelo ha clasificado en una categoría diferente a k.
- **Falsos positivos:** Número de observaciones no pertenecientes a la categoría k que fueron clasificadas como pertenecientes a la clase k.

Tomando estos valores presentes en la matriz de confusión se forman diversas métricas que permiten validar un modelo de clasificación multiclase, siendo dos de las más utilizadas:

- **Sensibilidad** (TPR_k)
  Dada una categoría k, proporción de las observaciones pertenecientes a ella correctamente clasificadas:
  $$\mathrm{TPR}_k = \frac{\mathrm{TP}_k}{\mathrm{TP}_k + \mathrm{FN}_k}.$$
- **Especificidad** (TNR_k)
  Dada una categoría k, proporción de observaciones no pertenecientes a ella que fueron clasificadas como tal:
  $$\mathrm{TNR}_k = \frac{\mathrm{TN}_k}{\mathrm{TN}_k + \mathrm{FP}_k}.$$

Otra medida por categorías será la puntuación F_1 mediante la media armónica de la sensibilidad y el valor predictivo positivo, es decir, para cada categoría k tenemos la expresión:

$$\mathrm{F}_{1k} = \frac{2\mathrm{TP}_k}{2\mathrm{TP}_k + \mathrm{TP}_k + \mathrm{FN}_k}.$$

Aunque estas medidas por categorías aportan información valiosa, lo ideal sería disponer de métricas que permitan una visión global del modelo.
Por un lado, se tiene la precisión global (ACC) que viene dada la proporción de observaciones clasificadas de forma correcta, es decir, por el cociente entre la suma de verdaderos positivos asociadas a cada una de las K categorías del modelo entre el total de observaciones:

$$\mathrm{ACC} = \frac{\sum_{k=1}^{K} \mathrm{TP}_k}{n}.$$

Por otro lado, existen dos enfoques principales para abordar esta cuestión, conocidos como macro-promedios y micro-promedios. Los macro-promedios se caracterizan por calcular una métrica específica para cada una de las categorías individuales y luego realizar un promedio de dichas métricas. Sin embargo, los micro-promedios se basan en la extracción de valores de la matriz de confusión para calcular una única medida global.

De esta manera, es posible definir las medidas de sensibilidad macro-promedio y sensibilidad micro-promedio de la siguiente forma:

$$\mathrm{Sensibilidad}_{macro} = \frac{1}{K}\sum_{k=1}^{K} \mathrm{TPR}_k, \qquad \mathrm{Sensibilidad}_{micro} = \frac{\sum_{k=1}^{K} \mathrm{TP}_k}{\sum_{k=1}^{K}(\mathrm{TP}_k + \mathrm{FN}_k)}.$$

Se puede seguir el mismo proceso para la especificidad:

$$\text{Especificidad}_{macro} = \frac{1}{K}\sum_{k=1}^{K}\text{TNR}_k, \qquad \text{Especificidad}_{micro} = \frac{\sum_{k=1}^{K}\text{TN}_k}{\sum_{k=1}^{K}(\text{TN}_k + \text{FP}_k)}.$$

Es esencial tener en cuenta que existe diferencia entre la interpretación de los micro-promedios y los macro-promedios. Los macro-promedios se calculan al tratar a todas las categorías de manera igualitaria, mientras que los micro-promedios se calculan al considerar todas las observaciones por igual. En el caso de los macro-promedios, se favorece a los modelos que tienen un buen funcionamiento en todas las categorías, lo cual es relevante si todas las categorías poseen igual importancia. Sin embargo, surge un problema cuando se enfrenta a un desequilibrio en los datos, ya que el macro-promedio tiende a generar una medida demasiado optimista o pesimista dependiendo de la categoría mayoritaria.

Para resolver este inconveniente, se podría establecer un intervalo de confianza para el macro-promedio. No obstante, lo más común es calcularlo mediante un promedio ponderado de todas las categorías, asignando un peso correspondiente a cada una de ellas en función de su prevalencia en el conjunto de datos. De esta manera, se otorga importancia a cada categoría en proporción a su frecuencia en los datos, permitiendo una evaluación más equilibrada del rendimiento del modelo. Si se desea más información sobre este tema, ver Jurafkasky y Martin [22].

Acto seguido, se explicará la métrica utilizada en este trabajo para la comparación de modelos, la curva ROC. La curva ROC (*Receiver Operating Characteristic*), es una metodología desarrollada para analizar un sistema de decisión. Consiste en una representación gráfica, en la que contrastamos la sensibilidad con la especificidad para un sistema de clasificación, comparando el rendimiento de los distintos modelos.

En ella se representa el valor obtenido restándole la especifidad a la unidad, representada en el eje de abcisas, frente a la sensibilidad, en el eje de ordenadas. Lo ideal será una curva cercana al vértice superior izquierdo, que significaría un modelo con alta sensibilidad y especificidad. Se puede ver un ejemplo de curva ROC en la Figura 4.3.

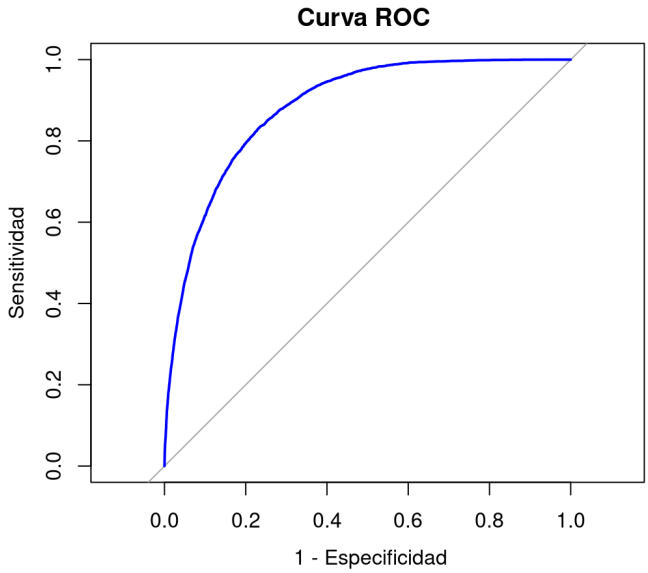


Figura 4.3: Ejemplo de curva ROC.

Se define estonces el AUC (*Area under the curve*) como el área bajo la curva ROC, que estima la capacidad del modelo para discriminar entre 'exitos' y 'fracasos':

$$\text{AUC} = \int_0^1 \text{ROC}(p)dp.$$

En Breiman et al. [11], se muestra que el AUC está muy relacionado con el índice de Gini. Por definición, el AUC varía entre 0 y 1, donde 1 representa un modelo perfecto sin errores de clasificación, y 0,5 equivale a un modelo que no distingue entre las categorías correspondientes. Por lo tanto, si el modelo es efectivo, el valor del AUC será mayor a 0,5, y se considera que es un buen modelo a partir de valores cercanos a 0,7.

En cuanto a la comparación de modelos, se establece que un modelo A es uniformemente mejor que otro modelo B cuando se cumple la siguiente desigualdad:

$$\text{ROC}_A(p) \geq \text{ROC}_B(p), \quad \forall p \in (0,1),$$

lo que es sencillo ver que implica

$$\text{AUC}_A \geq \text{AUC}_B.$$

Cabe destacar que el inverso no es cierto en general, por lo que obtener un mayor AUC no implica que el modelo vaya a ser mejor.

Si bien es cierto que la curva ROC y el AUC son métricas diseñadas para evaluar un problema de clasificación binaria, se pueden adaptar a un problema de clasificación multiclase de la misma forma que las otras métricas anteriormente explicadas.

Se adopta el enfoque *One vs. All* y se calcula la curva ROC y el AUC para cada una de las K categorías del modelo, a través de las medidas de sensibilidad y especificidad calculadas para cada categoría. Adicionalmente, se aplican las estrategias de macro-promedio y micro-promedio anteriormente explicadas calculando el AUC en base a la curva ROC construida con las sensibilidades y especificidades macro-promedio y micro-promedio, respectivamente.

Para más información sobre la curva ROC y el AUC consultar Alonzo y Pepe [23] y Fawcett [24].

Capítulo 5

Resultados

En este capítulo se comentarán los resultados obtenidos al realizar la predicción de a qué etapa del ciclo de vida familiar pertenece cada cliente mediante varias metodologías de aprendizaje estadístico vistas en el Capítulo 4, en concreto se aplican: Regresión logística multinomial (Sección 5.1), *Random Forest* (Sección 5.2), *XGBoost* (Sección 5.3) y *LightGBM* (Sección 5.4).

Acto seguido, se validarán estos modelos mediante diversas técnicas introducidas en la Sección 4.5 sobre el conjunto de validación, y finalmente, se identificará cual será el modelo con un mejor desempeño de entre los vistos, además de evaular si es un modelo satisfactorio para la entidad en la Sección 5.5.

Como fue comentado en la Sección 4.0.1, se dividirá nuestro conjunto de datos en tres muestras: Entrenamiento 60 %, Test 20 % y Validación 20 %, quedando el conjunto de datos como se muestra en la Tabla 5.1.

Conjunto de datos:	**Entrenamiento**	**Test**	**Validación**
Clientes	-	-	-
Singles (%)	- %	- %	- %
Sin hijos (%)	- %	- %	- %
Hijos 0-6 (%)	- %	- %	- %
Hijos 6-18 (%)	- %	- %	- %
Hijos >18 (%)	- %	- %	- %
Nido vacío (%)	- %	- %	- %
Viudos (%)	- %	- %	- %

Tabla 5.1: Número de clientes y proporción de cada etapa del ciclo de vida familiar en cada conjunto de la partición de los datos (tabla modificada por confidencialidad).

Finalmente, comentar que el software utilizado para llevar todo este proceso a cabo ha sido R, mediante el uso de diferentes librerías disponibles. En concreto, para la aplicación de método *Random Forest* se ha utilizado la librería **h2o**[25], que dispone de una plataforma de código abierto que permite construir modelos de aprendizaje automático a partir de grandes cantidades de datos (*Big Data*) de forma distribuida y escalable mediante el paradigma *MapReduce*. Para el *XGBoost* y el *LightCBM* fueron utilizadas las librerías de código abierto **xgboost** [26] y **lightgbm** [27], respectivamente. Por úlitmo, para el modelo de regresión logística multinomial fue usada la librería **mnet** [28].

5.1. Regresión logística multinomial

En esta Sección se muestran los resultados correspondientes al modelo de regresión logística multinomial, explicada en la Sección 4.1.2.
La primera intención fue construir el modelo con todas las variables del conjunto de datos, y luego aplicar un método que permitiera quedarnos con aquellas de más importancia, como el AIC o el BIC. Sin embargo, el conjunto de datos con el que trabajamos está compuesto por 216 variables, número que el modelo no es capaz de soportar.

Por tanto, inicialmente se construirá el modelo con las 10 variables con más importancia según el modelo *XGBoost*, construido en la Sección 5.3.

Posteriormente se aplica el criterio del AIC, definido como

$$\mathrm{AIC} = -2\log \mathrm{Lik} + kn,$$

donde $\log \mathrm{Lik}$ es el valor de la log-verosimilitud de modelo, k es un término de penalización del exceso de parámetros, y n es el número de parámetros del modelo. El criterio AIC consistirá en quedarse con el modelo con el AIC más bajo, decidiendo si quitar alguna variable del modelo y cuales quitar.

Una vez aplicado el criterio del AIC, nos devuelve el modelo inicial, por lo que la ausencia de alguna de las variables no mejorará el modelo. Si aplicamos la función *summary* al modelo, se muestra la siguiente salida:

```
Coefficients:
(Intercept)    Variable 1    Variable 2       Variable 3    Variable 4  Variable 5   Variable 6
1    4.261963 -0.06486917  -0.0003916495   -0.0001049797 -9.073063e-06   1.216939  1.395342e-04
2    3.006402 -0.06282683  -0.0003614055   -0.0001130545 -2.613153e-06   1.372945  1.429229e-04
3   -1.004770  0.07274739  -0.0004872500    0.0005611863 -4.773922e-06   1.324594  1.497956e-04
4   -8.778551  0.21539948  -0.0006125829    0.0006148598 -5.999749e-06   1.444647  1.355182e-04
5  -16.732809  0.32192155  -0.0006220637    0.0008281012 -4.385855e-06   1.554483  8.047646e-05
6  -21.919933  0.41032104  -0.0011522610    0.0016202300 -3.417595e-06   1.467977  1.030193e-04
            Variable 7          Variable 8             Variable 9                Variable 10
1           -1.2129462          0.01332086            2.456491e-05              -0.0039024438
2           -0.8153737          0.01205865            2.450411e-05              -0.0025200488
3           -0.9605778          0.01264350            1.901867e-05              -0.0007439805
4           -1.0346316          0.01279766            1.364240e-05               0.0036263688
5           -1.0432131          0.01317586            1.621662e-05               0.0036331481
6           -0.9101388          0.01229494           -1.416001e-06              -0.0036730819

Std. Errors:
(Intercept)    Variable 1      Variable 2       Variable 3    Variable 4  Variable 5
1 1.010554e-05 0.0004079970   1.165100e-05    1.010229e-04 5.648468e-07 1.057139e-05
2 9.767664e-06 0.0003924826   9.197013e-06    7.122437e-05 2.897834e-07 1.167537e-05
3 7.630962e-06 0.0003129618   8.190339e-06    4.613864e-05 2.617673e-07 6.167264e-06
4 7.593504e-06 0.0003450556   9.929725e-06    4.866322e-05 3.043785e-07 7.144447e-06
5 8.221906e-06 0.0004751097   1.579937e-05    5.344839e-05 3.898375e-07 9.898393e-06
6 8.082214e-06 0.0004996790   2.537836e-05    5.173127e-05 4.265399e-07 8.703581e-06
    Variable 6          Variable 7          Variable 8               Variable 9
1  1.071581e-05         2.114903e-05        0.0001293494             4.319507e-06
2  8.762724e-06         3.395988e-05        0.0001301588             3.826007e-06
```

```
3  8.294128e-06         4.529315e-05         0.0001206393             3.687168e-06
4  8.980173e-06         2.458867e-05         0.0001261352             3.886495e-06
5  1.375744e-05         2.079466e-05         0.0001351991             4.529672e-06
6  1.590265e-05         1.914100e-05         0.0001853132             5.679471e-06
                Variable 10
1               0.0002964455
2               0.0002652320
3               0.0002336695
4               0.0002544518
5               0.0003228460
6               0.0004436244

Residual Deviance: 534099
AIC: 534231
```

En cada fila k se encuentran los valores de las estimaciones de lo parámetros β_k asociados a cada una de las variables, lo que nos permitiría calcular los términos $\hat{\pi}_k(X, \beta_k)$ para $k = 1, \ldots, 7$, según las Ecuaciones 4.13 y 4.14. Cabe destacar que en este modelo se ha tomado la categoría Single de referencia.

A continuación, se muestra en la Tabla 5.2 la matriz de confusión dada por el modelo sobre el conjunto de Validación.

RLM	Observación						
Predicción	**Single**	**Sin hijos**	**Hijos 0-6**	**Hijos 6-18**	**Hijos >18**	**Nido vacío**	**Viudo**
Single	9961	988	3318	4515	844	53	17
Sin hijos	0	949	598	377	4	1	1
Hijos 0-6	1240	713	1471	353	2	0	1
Hijos 6-18	2039	3135	3688	17242	6292	428	232
Hijos >18	1802	1	27	2169	5345	1458	754
Nido vacío	0	0	0	23	78	160	110
Viudo	78	0	6	144	416	340	2535

Tabla 5.2: Matriz de confusión obtenida mediante la aplicación del método de clasificación Regresión logística multinomial.

Las métricas obtenidas a partir de esta matriz de confusión serán las usadas para evaluar el modelo. En concreto, se obtiene un AUC para el modelo global de 0,8301, indicando que estamos ante un modelo relativamente bueno. Se representan en la Figura 5.1 las curvas ROC correspondientes a cada una de las categorías de la variable objetivo, *Ciclo_de_vida*.

Se puede ver que la curva ROC más próxima al vértice superior izquierdo es la correspondiente a la categoría de Viudo, con un AUC de 0,9688, por lo que será la categoría mejor clasificada por el modelo. Esto ya se puede saber a simple vista en la matriz de confusión, donde se ve que muy poca proporción de las observaciones correspondientes a la categoría Viudo están clasificadas incorrectamente.

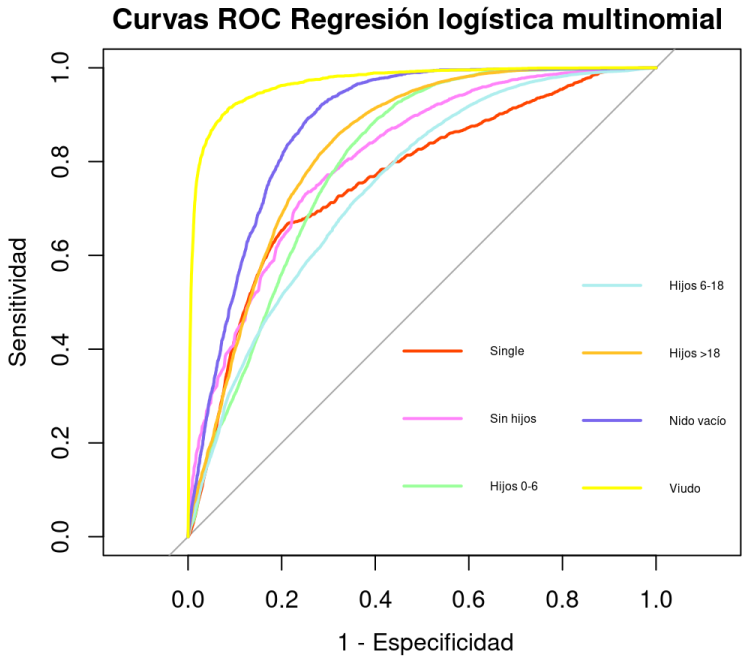


Figura 5.1: Curvas ROC por categoría para la regresión logística multinomial.

Para el resto de categorías se tiene un AUC de 0,7629 para los singles, de 0,8076 para las parejas sin hijos, de 0,7977 para las parejas con hijos de 0 a 6 años,de 0,7482 para las parejas con hijos de 6 a 18 años, de 0,8299 para las parejas con hijos mayores de edad y de 0,8783 para el nido vacío.

5.2. Random Forest

En esta sección se muestran los resultados correspondientes a la metodología *Random Forest*, explicada en detalle en la Sección 4.3.2. Para la construcción de este, los hiperparámetros considerados fueron *m_try*, tomando valores de 14 a 20; *min_rows*, de 1 a 200; *sample_rate*, de 0,5 a 1 y *max_depth*, de 1 a 9. Finalmente, el hiperparámetro *ntrees* tendrá asignado su valor por defecto, 500.

Se realizaron pruebas con un total de 150 combinaciones entre los posibles valores dados para los hiperparámetros, tratando de encontrar la que obtenga mejores resultados, evaluados en el conjunto de Test. Dicha configuración óptima se muestra en la Tabla 5.3.

Una vez entrenado el modelo *Random Forest* con los hiperparámetros mostrados en la Tabla 5.3, se muestran en la Figura 5.2 las 20 variables con mayor importancia tomadas por el modelo.

A continuación, se muestra en la Tabla 5.4 la matriz de confusión dada por el modelo sobre el conjunto de Validación, aquel que no se usó ni para la selección de hiperparámetros ni para el entrenamiento del modelo.

Las métricas obtenidas a partir de esta matriz de confusión serán las usadas para evaluar el modelo. En concreto, se obtiene un AUC para el modelo global de 0,8832, lo que parece indicar que estamos ante un modelo bastante bueno. A continuación, se representa en la Figura 5.3 las curvas ROC para cada categoría definida por la variable objetivo *Ciclo_de_vida*.

Hiperparámetro	**Valor**
ntrees	500
m_try	20
min_rows	1
sample_rate	0,66
max_depth	7

Tabla 5.3: Configuración óptima de hiperparámetros para el *Random Forest*.

CONFIDENCIAL

Figura 5.2: Variables con mayor importancia para el *Random Forest*.

Random Forest	Observación						
Predicción	**Single**	**Sin hijos**	**Hijos 0-6**	**Hijos 6-18**	**Hijos >18**	**Nido vacío**	**Viudo**
Single	11900	722	2598	2302	834	136	89
Sin hijos	6	1019	100	48	1	0	0
Hijos 0-6	128	218	682	160	3	0	0
Hijos 6-18	2718	3784	5729	20842	7623	651	494
Hijos >18	459	0	16	1387	4097	1191	500
Viudo	49	0	3	108	294	254	2409

Tabla 5.4: Matriz de confusión obtenida mediante la aplicación del método de clasificación *Random Forest*.

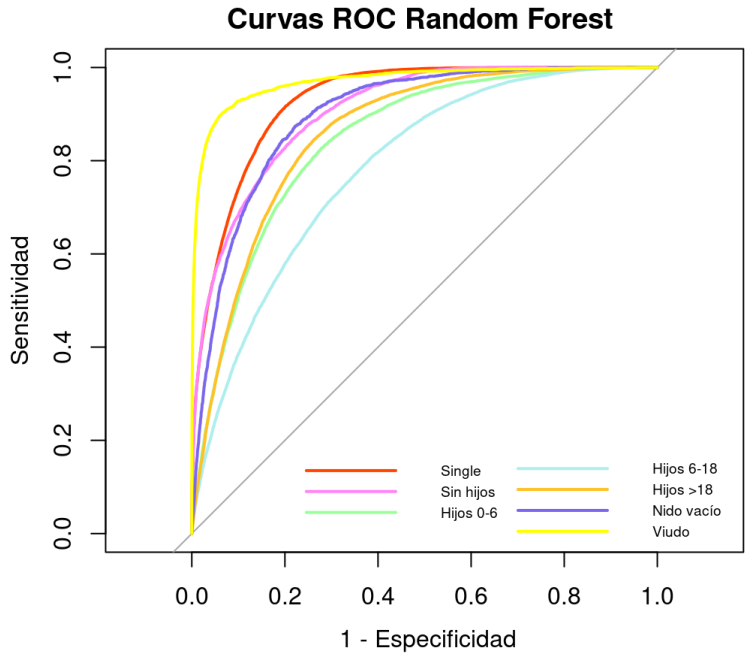


Figura 5.3: Curvas ROC por categoría para el *Random Forest*.

En ella podemos ver que las curvas ROC más próximas al vértice superior izquierdo son las pertenecientes a las categorías de Single y Viudo, por lo que será las categorías que mejor estima el modelo, con un AUC de 0,9302 y 0,9706, respectivamente.

Asimismo, podemos ver que las curvas ROC pertenecientes al resto de categorías, aunque no tanto como las anteriormente mencionadas, están próximas al vértice superior izquierdo. En concreto se tienen los AUC de 0,9067 para las parejas sin hijos, de 0,8452 para las parejas con hijos de 0 a 6 años, de 0,7847 para las parejas con hijos de 6 a 18 años, de 0,8589 para las parejas con hijos mayores de edad y de 0,8983 para el nido vacío.

Este caso es un claro ejemplo de como se comporta el AUC cuando existen clases desbalanceadas, ya que, como se puede observar en la matriz de confusión referente a este modelo (Tabla 5.4), no es capaz de predecir la categoría nido vacío. En cambio, se le asocia un AUC de 0,8983, lo que indicaría una capacidad de predicción bastante buena sobre esta clase. Por ello, es importante no restringirse sólo a una métrica, ya que todas tienen sus ventajas y desventajas, y complementarla o bien con otras métricas o bien observando la matriz de confusión del modelo.

Por los resultados obtenidos y lo anteriormente explicado, no se considerará este modelo a la hora de seleccionar el mejor en la Sección 5.5.

5.3. XGBoost

En esta sección se muestran los resultados correspondientes a la metodología *XGBoost*, explicada en detalle en la Sección 4.4.2. Para la construcción de este los hiperparámetros considerados fueron *nrounds*, tomando valores desde 10 a 500; *eta*, desde 0,001 a 0,1; *gamma*, de 0 a 1; *min_child_weigth*, de 30 a 1000; *max_delta_step*, de 0 a 10; *max_depth*, de 1 a 10; *subsample*, *colsample_bytree* y *colsample_bylevel*, los tres hiperparámetros en rangos de 0,3 a 1. Por último, los hiperparámetros *lambda*, *alpha* y *grow_policy* mantienen sus valores por defecto.

Se realizaron pruebas con un total de 150 combinaciones posibles entre los valores dados para los hiperparámetros, con el fin de encontrar aquella que diera mejores resultados, evaluando estos sobre la muestra de Test. Dicha configuración óptima se muestra en la Tabla 5.5.

Una vez entrenado el modelo *XGBoost* con los hiperparámetros mostrados en la Tabla 5.5, se muestran en la Figura 5.4 las 20 variables con mayor importancia tomadas por el modelo.

A continuación, se muestra en la Tabla 5.6 la matriz de confusión dada por el modelo sobre el conjunto de Validación, aquel que no se usó ni para la selección de hiperparámetros ni para el entrenamiento del modelo.

Las métricas obtenidas a partir de esta matriz de confusión serán las usadas para evaluar el modelo. En concreto, se obtiene un AUC para el modelo global de 0,9199, lo que indica que estamos ante un modelo muy bueno. A continuación, se representa en la Figura 5.5 las curvas ROC para cada categoría definida por la variable objetivo *Ciclo_de_vida*.

En ella podemos ver que las curvas ROC más próximas al vértice superior izquierdo son las pertenecientes a las categorías de Single y Viudo, por lo que será las categorías que mejor estima el modelo, con un AUC de 0,9743 y 0,9854, respectivamente. Esto era algo que ya se puede ver a simple vista en la matriz de confusión representada en la Tabla 5.6, donde se ve que las observaciones pertenecientes a estas dos categorías se califican incorrectamente pocas veces.

Hiperparámetro	**Valor**
nrounds	100
eta	0,1
gamma	0,5
min_child_weigth	30
max_delta_step	3
max_depth	9
subsample	0,3
colsample_bytree	0,75
colsample_bylevel	0,8

Tabla 5.5: Configuración óptima de hiperparámetros para el *XGBoost*.

Figura 5.4: Variables con mayor importancia para el *XGBoost*.

XGBoost	Observación						
Predicción	**Single**	**Sin hijos**	**Hijos 0-6**	**Hijos 6-18**	**Hijos >18**	**Nido vacío**	**Viudo**
Single	13707	272	1524	1587	777	104	91
Sin hijos	18	2920	552	645	36	1	7
Hijos 0-6	367	908	3885	1466	19	0	4
Hijos 6-18	773	1684	3101	17749	4313	195	134
Hijos >18	202	2	41	3199	7309	1542	416
Nido vacío	7	0	1	35	138	196	38
Viudo	46	0	4	121	313	248	2857

Tabla 5.6: Matriz de confusión obtenida mediante la aplicación del método de clasificación *XGBoost*.

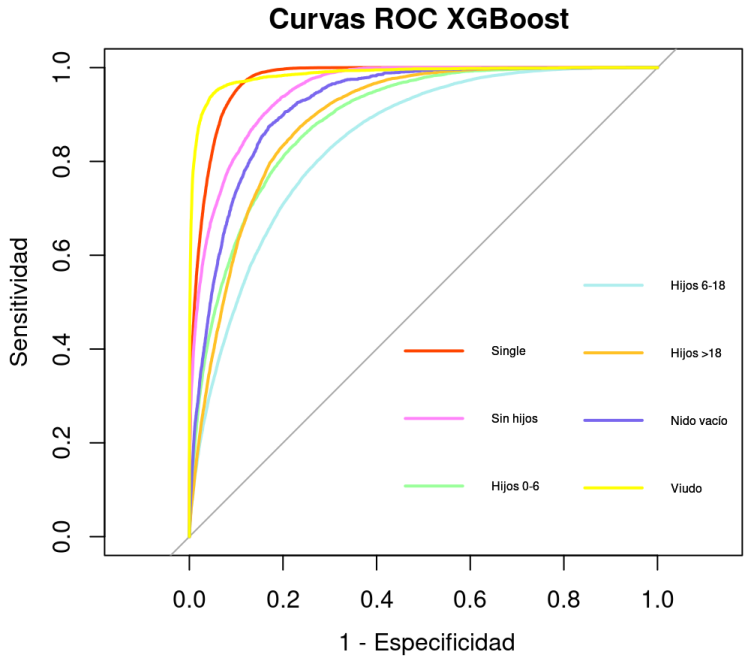


Figura 5.5: Curvas ROC por categoría para el *XGBoost*.

Asimismo, podemos ver que las curvas ROC pertenecientes al resto de categorías, aunque no tanto como las anteriormente mencionadas, están próximas al vértice superior izquierdo. En concreto se tienen los AUC de 0,9487 para las parejas sin hijos, de 0,8886 para las parejas con hijos de 0 a 6 años, de 0,8419 para las parejas con hijos de 6 a 18 años, de 0,8905 para las parejas con hijos mayores de edad y de 0,9211 para el nido vacío.

5.4. LightGBM

En esta sección se muestran los resultados correspondientes a la metodología *LightGBM*, explicada en detalle en la Sección 4.4.3. Para la construcción de este los hiperparámetros considerados fueron *num_iterations*, tomando valores de 10 a 500; *learning_rate*, de 0,0001 a 0,1; *min_gain_to_split*, de 0 a 10; *num_leaves*, de 30 a 1000; *max_depth*, de 1 a 10 ; *lambda_l1* y *lambda_l2* de 0 a 10. Por último, el hiperparámetro *min_data_in_leaf* tendrá asignado su valor por defecto, 20.

Se realizaron pruebas con un total de 150 combinaciones entre los posibles valores dados para los hiperparámetros, tratando de encontrar la que obtenga mejores resultados, evaluados en el conjunto de Test. Dicha configuración óptima se muestra en la Tabla 5.7.

Hiperparámetro	**Valor**
num_iterations	500
learning_rate	0,1
min_gain_to_split	0
min_child_weigth	30
num_leaves	500
max_depth	59
lambda_l1	7
lambda_l2	7
min_data_in_leaf	20

Tabla 5.7: Configuración óptima de hiperparámetros para el *LightGBM*.

Una vez entrenado el modelo *LightGBM* con la configuración óptima de hiperparámetros, se muestran en la Figura 5.6 las 20 variables con mayor importancia tomadas por el modelo.

Se puede entonces observar que que las 3 variables con mayor importancia vuelven a ser las mismas que en el modelo *XGBoost* ajustado en la Sección 5.3.

A continuación, se muestra en la Tabla 5.8 la matriz de confusión dada por el modelo sobre el

CONFIDENCIAL

Figura 5.6: Variables con mayor importancia para el *LightGBM*.

conjunto de Validación, aquel que no se usó ni para la selección de hiperparámetros ni para el entrenamiento del modelo.

LightGBM	Observación						
Predicción	**Single**	**Sin hijos**	**Hijos 0-6**	**Hijos 6-18**	**Hijos >18**	**Nido vacío**	**Viudo**
Single	13832	233	1450	1471	685	88	89
Sin hijos	21	3168	610	720	33	1	7
Hijos 0-6	361	888	4062	1486	22	0	5
Hijos 6-18	697	1493	2934	17654	4081	17	111
Hijos >18	161	3	43	3297	7565	1498	386
Nido vacío	4	0	0	46	210	290	66
Viudo	44	1	9	128	309	235	2883

Tabla 5.8: Matriz de confusión obtenida mediante la aplicación del método de clasificación *LightGBM*.

Las métricas obtenidas a partir de esta matriz de confusión serán las usadas para evaluar el modelo. En concreto, se obtiene un AUC para el modelo global de 0,9125, lo que indica que estamos ante un modelo muy bueno. A continuación, se representa en la Figura 5.7 las curvas ROC para cada categoría definida por la variable objetivo *Ciclo_de_vida*.

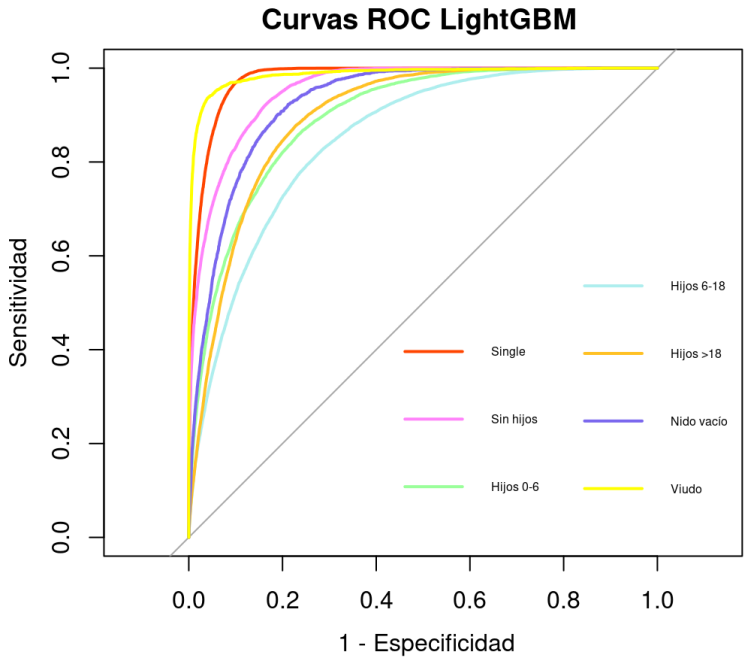


Figura 5.7: Curvas ROC por categoría para el *LightGBM*.

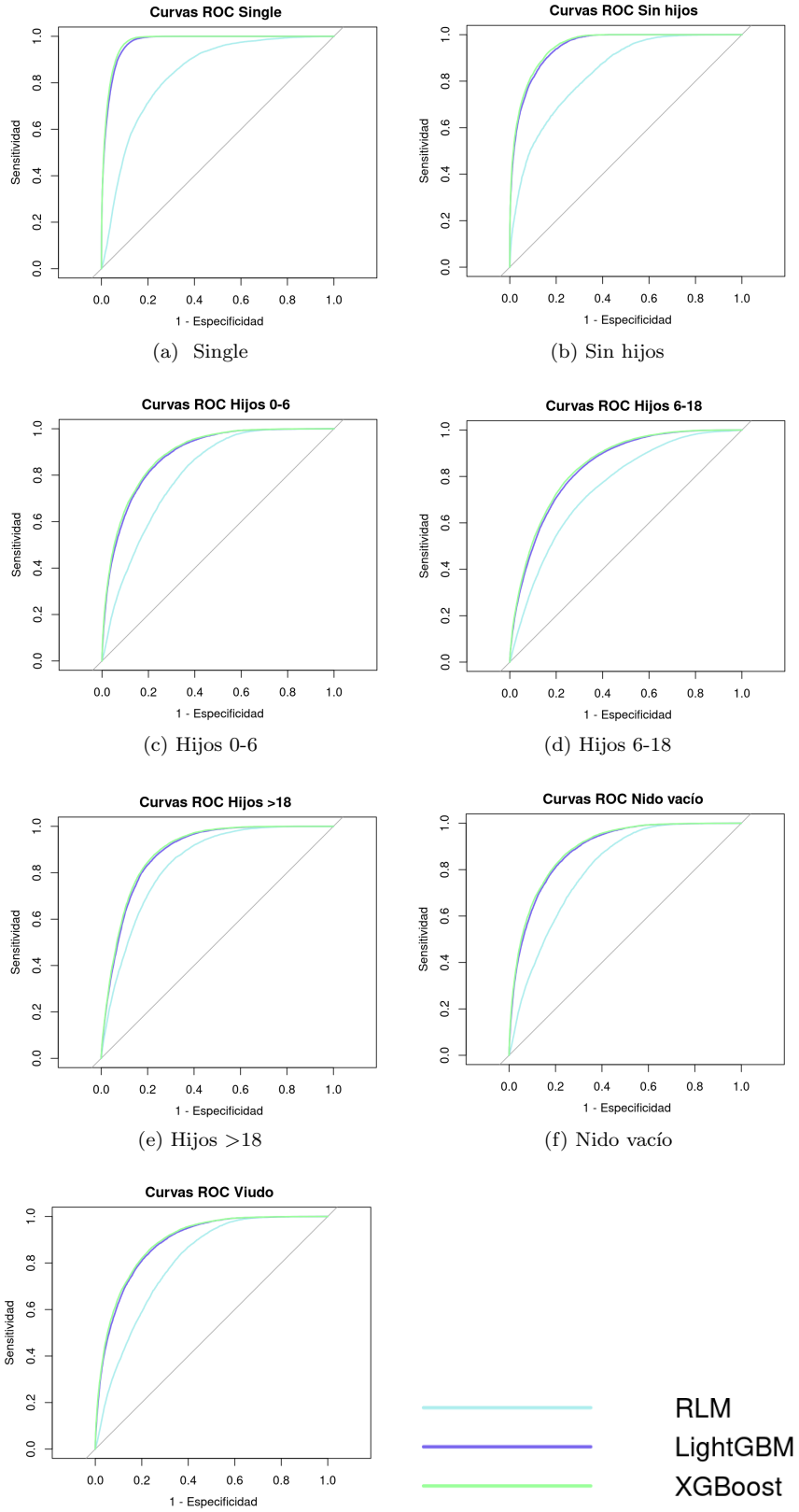
En ella podemos ver que las curvas ROC más próximas al vértice superior izquierdo son las pertenecientes a las categorías de Single y Viudo, por lo que será las categorías que mejor estima el modelo, con un AUC de 0,9777 y 0,9863, respectivamente. Esto era algo que ya se puede ver a simple vista en la matriz de confusión representada en la Tabla 5.6, donde se ve que las observaciones pertenecientes a estas dos categorías se califican incorrectamente pocas veces.

Asimismo, podemos ver que las curvas ROC pertenecientes al resto de categorías, aunque no tanto como las anteriormente mencionadas, están próximas al vértice superior izquierdo. En concreto se tienen los AUC de 0,9435 para las parejas sin hijos, de 0,8844 para las parejas con hijos de 0 a 6 años, de 0,8394 para las parejas con hijos de 6 a 18 años, de 0,8899 para las parejas con hijos mayores de edad y de 0,9204 para el nido vacío.

5.5. Selección del mejor modelo

Una vez vistos los modelos a lo largo de este capítulo, compararemos las curvas ROC pertenecientes a cada categoría para cada uno de los tres modelo: regresión logística multinomial (Sección 5.1), *XGBoost* (Sección 5.3) y *LightGBM* (Sección 5.4). Se puede ver esta comparación en la Figura 5.8. El modelo *Random Forest* no será incluido en esta comparación por no adecuarse a los requisitos y no ser capaz de predeecir una de las categorías, como fue explicado en la Sección 4.3.2.

Como se puede observar, la curva ROC más distante al eje superior izquierdo es la correspondiente a la regresión logística multinomial, por lo que será el modelo menos indicado. Por otro lado, se ve que las curvas ROC correspondientes al *XGBoost* y al *LightGBM* son las más próximas al vértice superior izquierdo. Pese a estar las dos casi igualadas, hay una que es superior en todas las categorías, la correspondiente al *XGBoost*. Por tanto, se establece que el modelo *XGBoost* es uniformemente mejor que los otros dos modelo considerados.



(a) Single (b) Sin hijos

(c) Hijos 0-6 (d) Hijos 6-18

(e) Hijos >18 (f) Nido vacío

(g) Viudo (h) Leyenda

Figura 5.8: Curvas ROC de cada modelo para cada categoría.

Sin embargo, observando la Figura 5.4, donde se nos muestra la importancia de las variables del modelo, se puede apreciar la ausencia de las variables correspondientes a la tipología de gastos del cliente. Teniendo en cuenta que ABANCA es una entidad bancaria, por lo que su principal interés son este tipo de variables, y que una parte de los clientes no poseen gastos observados, surge la pregunta: ¿El modelo funciona igual de bien en los clientes con gastos observados y en los clientes sin ellos?

Entoces procederemos a evaluar el mismo modelo, por un lado exclusivamente con los clientes que tienen observaciones de gastos y por otro lado con aquellos clientes que carecen de ellas.

Como se vio en la Sección 5.3, el modelo tiene un AUC global de 0,9199. Cuando lo evaluamos sobre los clientes con gastos se experimenta una disminución del AUC, obteniendo un AUC de 0,9044, mientras que lo evaluamos sobre los clientes sin gastos se experimenta una subida de este, quedando un AUC de 0,9234. Por tanto, se justifica la necesidad de dos modelos independientes, uno destinado a los clientes con observaciones de gastos y otro destinado a los clientes con ausencia de ellas.

Estos modelos seguirán la metodología *XGBoost*, ya que es la que mejores resultados dio globalmente y con la que la entidad más cómoda se siente trabajando. Lo resultados se mostrarán en las Secciones 5.5.1 y 5.5.2.

5.5.1. Modelo para clientes con gastos

En este modelo solo se considerarán los clientes con observaciones sobre sus gastos. Se hará de nuevo la partición del conjunto en entrenamiento, test y validación, resultando en lo mostrado en la Tabla 5.9.

Conjunto de datos:	**Entrenamiento**	**Test**	**Validación**
Clientes	-	-	-
Singles (%)	-%	-%	-%
Sin hijos (%)	-%	-%	-%
Hijos 0-6 (%)	-%	-%	-%
Hijos 6-18 (%)	-%	-%	-%
Hijos >18 (%)	-%	-%	-%
Nido vacío (%)	-%	-%	-%
Viudos (%)	-%	-%	-%

Tabla 5.9: Número de clientes y proporción de cada etapa del ciclo de vida familiar en cada conjunto de la partición de los datos correspondiente a los clientes con observaciones de gastos.

Para este modelo se han considerado los mismos hiperparámetros con los mismos rangos que los propuestos en la Sección 5.3. Se realizaron pruebas con un total de 150 combinaciones posibles entre

los rangos dados para los hiperparámetros, con el fin de encontrar aquella que diera mejores resultados, evaluados sobre la muestra de Test. Esta configuración óptima se encuentra en la Tabla 5.10.

Hiperparámetro	**Valor**
nrounds	200
eta	0,1
gamma	1
min_child_weigth	300
max_delta_step	7
max_depth	9
subsample	0,66
colsample_bytree	1
colsample_bylevel	0,66

Tabla 5.10: Configuración óptima de hiperparámetros para el *XGBoost* para clientes con gastos.

Ahora, con el modelo *XGBoost* entrenado con los hiperparámetros óptimos, se muestran en la Figura 5.9 las 20 variables con mayor importancia para el modelo. En esta se puede apreciar que las 3 variables con mayor importancia son las mismas que en el modelo *XGBoost* ajustado en la Sección 5.3. Se observa entonces que en este modelo sí se toman como importantes ciertas variables referentes a la tipología de gastos.

A continuación, se muestra en la Tabla 5.11 la matriz de confusión dada por el modelo sobre el conjunto de Validación, que no fue usado ni para la selección de hiperparámetros ni para el entrenamiento del modelo.

Las métricas obtenidas a partir de esta matriz de confusión serán las usadas para evaluar y validar el modelo. En concreto, se obtiene un AUC global de 0,9164, lo que nos indica que estamos ante un modelo muy bueno. Se presentan entonces en la Figura 5.10 las curvas ROC para cada categoría de la variable objetivo *Ciclo_de_vida*.

Se observa que, de nuevo, las curvas ROC más cercanas al vértice superior izquierdo son las pertenecientes a las categorías Single y Viudo, con un AUC de 0,9651 y 0,9854, respectivamente.

Para el resto de categorías se tiene un AUC de 0,9293 para las parejas sin hijos, de 0,8838 para las parejas con hijos de 0 a 6 años,de 0,8483 para las parejas con hijos de 6 a 18 años, de 0,9012 para las parejas con hijos mayores de edad y de 0,9315 para el nido vacío.

Entonces, la cuestión final sobre este modelo es: ¿funciona mejor que el *XGboost* general definido en la Sección 5.3?

CONFIDENCIAL

Figura 5.9: Variables con mayor importancia para el *XGBoost* para clientes con gastos.

XGBoost	Observación						
Predicción	**Single**	**Sin hijos**	**Hijos 0-6**	**Hijos 6-18**	**Hijos >18**	**Nido vacío**	**Viudo**
Single	9485	184	1088	1103	498	83	66
Sin hijos	17	955	285	222	14	0	0
Hijos 0-6	355	614	3220	1174	23	0	5
Hijos 6-18	722	852	2094	11342	2289	89	89
Hijos >18	202	2	28	1708	4074	788	182
Nido vacío	4	0	0	25	68	121	17
Viudo	40	0	3	91	196	160	1099

Tabla 5.11: Matriz de confusión *XGBoost* para clientes con gastos.

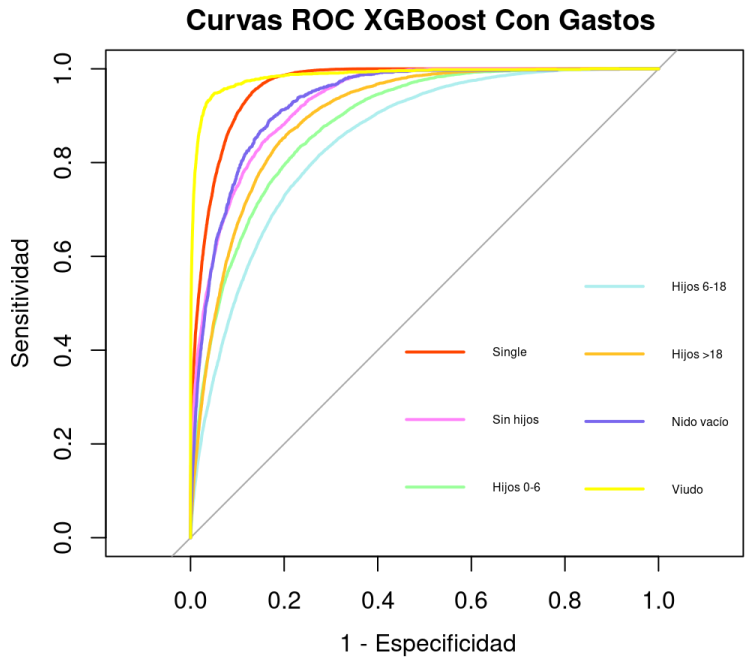


Figura 5.10: Curvas ROC por categoría para el *XGBoost* para clientes con gastos.

Para responder esta pregunta evaluaremos el modelo general sobre el conjunto de Validación de los clientes con observaciones sobre gastos disponibles. Se obtiene un AUC de 0,9088, inferior al obtenido con el modelo específico. Por tanto, este será el modelo usado para aquellos clientes que acumulan más de 150€ gastados con sus tarjetas en los últimos 12 meses.

5.5.2. Modelo para clientes sin gastos

En este modelo solo se considerarán los clientes sin observaciones sobre sus gastos. Se hará de nuevo la partición del conjunto en entrenamiento, test y validación, resultando en lo mostrado en la Tabla 5.12.

Para este modelo se han considerado los mismos hiperparámetros con los mismos rangos que los propuestos en la Sección 5.3. Se realizaron pruebas con un total de 150 combinaciones posibles entre los rangos dados para los hiperparámetros, con el fin de encontrar aquella que diera mejores resultados, evaluados sobre la muestra de Test. Esta configuración óptima se encuentra en la Tabla 5.13.

Ahora, con el modelo *XGBoost* entrenado con los hiperparámetros óptimos, se muestran en la Figura 5.11 las 20 variables con mayor importancia para el modelo.

Se observa que las 2 variables con mayor importancia son las mismas que en el modelo *XGBoost* ajustado en la Sección 5.3. Las demás variables también serán mayoritariamente las mismas, pero con distintas importancias y orden. También se puede ver la ausencia de variables relacionadas con la tipología de gastos, como sucedía en el modelo mencionado.

A continuación, se muestra en la Tabla 5.14 la matriz de confusión dada por el modelo sobre el conjunto de Validación, que no fue usado ni para la selección de hiperparámetros ni para el entrenamiento del modelo.

Conjunto de datos:	**Entrenamiento**	**Test**	**Validación**
Clientes	-	-	-
Singles (%)	-%	-%	-%
Sin hijos (%)	-%	-%	-%
Hijos 0-6 (%)	-%	-%	-%
Hijos 6-18 (%)	-%	-%	-%
Hijos >18 (%)	-%	-%	-%
Nido vacío (%)	-%	-%	-%
Viudos (%)	-%	-%	-%

Tabla 5.12: Número de clientes y proporción de cada etapa del ciclo de vida familiar en cada conjunto de la partición de los datos correspondiente a los clientes sin observaciones de gastos.

Hiperparámetro	**Valor**
nrounds	100
eta	0,1
gamma	0,5
min_child_weigth	30
max_delta_step	3
max_depth	9
subsample	0,3
colsample_bytree	0,75
colsample_bylevel	0,8

Tabla 5.13: Configuración óptima de hiperparámetros para el *XGBoost* para clientes sin gastos.

CONFIDENCIAL

Figura 5.11: Variables con mayor importancia para el *XGBoost* para clientes sin gastos.

XGBoost	Observación						
Predicción	**Single**	**Sin hijos**	**Hijos 0-6**	**Hijos 6-18**	**Hijos >18**	**Nido vacío**	**Viudo**
Single	4136	59	384	404	207	42	25
Sin hijos	9	2050	322	458	25	0	1
Hijos 0-6	37	255	826	348	8	0	1
Hijos 6-18	126	756	919	6639	2062	102	62
Hijos >18	67	3	12	1448	3291	711	225
Nido vacío	3	0	0	15	54	105	12
Viudo	12	0	0	72	130	115	1341

Tabla 5.14: Matriz de confusión *XGBoost* para clientes sin gastos.

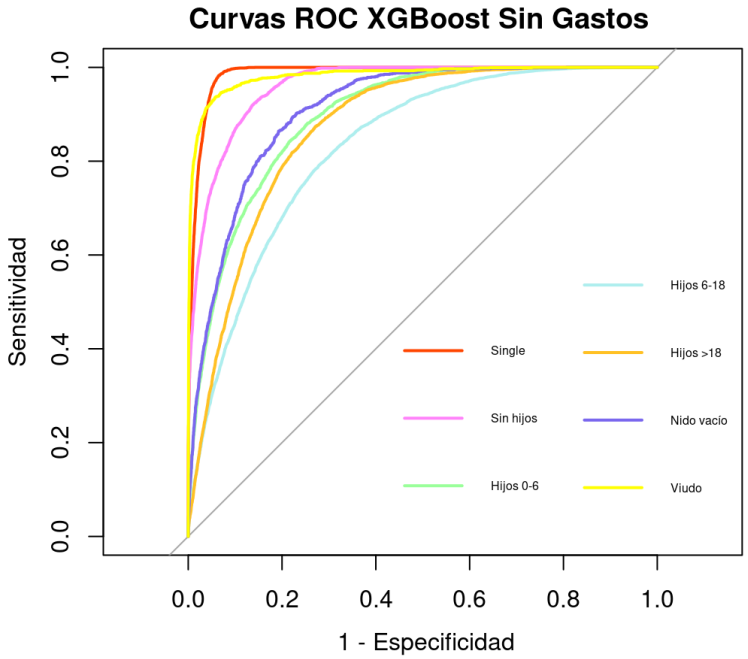


Figura 5.12: Curvas ROC por categoría para el *XGBoost* para clientes sin gastos.

Las métricas obtenidas a partir de esta matriz de confusión serán las usadas para evaluar y validar el modelo. En concreto, se obtiene un AUC global de 0,9225, lo que nos indica que estamos ante un modelo muy bueno. Se presentan entonces en la Figura 5.12 las curvas ROC para cada categoría de la variable objetivo *Ciclo_de_vida*.

Se observa que, de nuevo, las curvas ROC más cercanas al vértice superior izquierdo son las pertenecientes a las categorías Single y Viudo, con un AUC de 0,9866 y 0,9827, respectivamente.

Para el resto de categorías se tiene un AUC de 0,9607 para las parejas sin hijos, de 0,8958 para las parejas con hijos de 0 a 6 años,de 0,831 para las parejas con hijos de 6 a 18 años, de 0,8715 para las parejas con hijos mayores de edad y de 0,9101 para el nido vacío.

Entonces, estamos otra vez ante la cuestión final sobre este modelo: ¿funciona mejor que el *XGboost* general definido en la Sección 5.3?

Para responder esta pregunta evaluaremos el modelo general sobre el conjunto de Validación de los clientes sin observaciones sobre gastos disponibles. Se obtiene un AUC de 0,9204, que, aunque con diferencia mínima, es inferior al obtenido con el modelo ajustado en esta sección. Por tanto, nos quedaremos con este modelo para predecir en aquellos clientes sin observaciones de gastos.

Capítulo 6

Conclusiones y líneas futuras

Para finalizar esta memoria, se resumen brevemente de todo lo realizado a lo largo de este proyecto y se describen las líneas futuras de estudio que sería recomendable revisar o que ya se están realizando en la actualidad.

6.1. Conclusiones

El objetivo de este trabajo ha sido el desarrollo de un modelo predictivo que permita a la entidad predecir a que etapa del ciclo de vida familiar pertenece cada cliente. Esta es una característica muy importante del cliente, ya que en función a ella la entidad podrá ofrecerle los productos financieros más relevantes para él. Para ello, se han construido diversos indicadores utilizando el lenguaje SQL, lo que ha permitido determinar los clientes de los que se posee la suficiente información como para saber en que etapa del ciclo se encuentra. A partir de estos indicadores se ha formado la base de datos que nos ha permitido llevar a cabo técnicas de aprendizaje estadístico, tanto supervisado como no, observando el mejor funcionamiento de la primera opción. Tras aplicar diversos métodos de aprendizaje estadístico supervisado, se seleccionó el algoritmo basado en *Gradient Boosting* para árboles de decisión aplicado mediante el *software XGBoost*, una de las implementaciones más populares y eficientes en la actualidad (con un AUC de $0,9199$). Finalmente, tras obtener un modelo con muy buenos resultados, se concluyó que estos mejorarán aún más si se divide el conjunto de datos según la presencia de gastos en tarjetas del cliente, obteniendo así dos modelos *XGBoost* (con AUCs de 0,9164 y 0,9225).

Cabe destacar que la elaboración de este Trabajo Fin de Máster ha sido un proceso de continuo aprendizaje. Concretamente, se han combinado y adquirido conocimientos de distintas áreas; sobre el ámbito económico se han adquirido conocimientos sobre el funcionamiento de los modelos de propensión utilizados en la entidad, sobre la caracterización de clientes desde un punto de vista financiero y sobre los indicadores (*leads*) utilizados por la entidad a la hora de construir un modelo de propensión o predictivo del cliente. Sobre el área estadística, se ha afrontado la resolución de un problema de clasificación multiclase, con las dificultades añadidas que esto conlleva, mediante diferentes algoritmos de aprendizaje estadístico, como lo pueden ser CLARA, RLM, *Random Forest*, *XGBoost* y *LightGBM*, a la vez que se ha familiarizado con la librerías de R que permiten su implementación.

6.2. Líneas futuras

En este Trabajo Fin de Máster se han obtenido unos muy buenos resultados desde el punto de vista empresarial, lo que ha llevado a diversas líneas de trabajo que sería interesante tratar.

- **Implementación del modelo a los modelos de propensión de la entidad**

  Debido al buen funcionamiento del modelo, el siguiente paso lógico será convertir la variable respuesta dada por el modelo desarrollado en este trabajo como una variable explicativa más en los modelos de propensión ya existentes en la entidad y ver si se consigue una mejora en ellos.

  Un claro ejemplo en el que se consigue una mejoría será en el caso de un modelo de propensión referente a la contratación de un seguro de vida. Cuando una persona tiene hijos, surge la necesidad de asegurar un capital para ellos en caso de fallecimiento, incapacidad del cabeza de familia o algún inconveniente imprevisible. Por tanto, la variable referente al ciclo de vida familiar permite estimar qué clientes tienen una alta probabilidad de tener hijos, y consecuentemente necesitar este tipo de seguro.

- **Aplicación de nuevas métricas de validación de modelos**

  En este trabajo la principal métrica de validación usada el AUC dado por las curvas ROC, debido al buen funcionamiento de esta en todos los modelos de la entidad y su preferencia por ella. Sin embargo, sería interesante aplicar nuevas medidas para modelos de clasificación multiclase que nos permitan tener una visión y una interpretación más amplia del problema, como lo puede ser la *F1Score*.

- **Nuevas metodologías**

  Dado el buen funcionamiento observado en los métodos de clasificación a partir de árboles de decisión a través de la informmación disponible sobre los clientes, se ha propuesto aplicar estas técnicas para otros modelos de la entidad, incorporando otras nuevas como pueden ser las Redes Neuronales o el SVM (*Support Vector Machines*).

- **Nuevas variables**

  El desarrollo de estos modelos se ha realizado a partir de variables ya presentes en la entidad, excluyendo las variables creadas en el Capítulo 3, viendo que la variable principal es la edad del cliente, con mucha diferencia. Surge entonces una nueva línea de investigación, crear nuevas variables, a partir de los datos presentes en la entidad, que pueda servir de utilidad, como lo podría ser una que indica si el cliente está suscrito a cursos de maternidad/paternidad.

Bibliografía

[1] Glick, Paul C. (1947). The family cycle. *American sociological review*, 12(2), 164-174.

[2] Wells, William D. and Gubar, George (1966). Life Cycle Concept in Marketing Research. *Journal of Marketing Research, 3(4)*, 355-363. https://doi.org/10.1177/002224376600300403

[3] Duvall, Emily M. (1988). Family Developments First Forty Years. *Family Relations, 37(2)*, 127-134. https://doi.org/10.2307/584309

[4] Moratto Vásquez, Nadia S. and Zapata Posada, Johanna J. and Messager, Tatiana. (2015). CONCEPTUALIZACIÓN DE CICLO VITAL FAMILIAR: UNA MIRADA A LA PRODUCCIÓN DURANTE EL PERIODO COMPRENDIDO ENTRE LOS AÑOS 2002 A 2015 (Conceptualization of family life cycle: a view of the production during the period between 2002 and 2015). *CES Psicología*, 8(2), 103-121.

[5] Kaufman, Leonard and Rousseeuw, Peter J. (1990). *Finding Groups in Data-An Introduction to Cluster Analysis.* New York: John Wiley & Sons Inc. https://doi.org/10.1002/9780470316801

[6] Instituto Nacional de Estadística (2021). *Movimiento Natural de la Población.* https://ine.es

[7] Fernández-Casal, Rubén and Costa, Julián and Oviedo, Manuel (2020). *Aprendizaje estadístico.* Github. https://rubenfcasal.github.io/aprendizaje_estadistico/

[8] Wood, Simon N. (2017). *Generalized additive models an introduction with R.* CRC Press, Taylor & Francis Group.

[9] Engel, J. (1988). Polytomous logistic regression. *Statistica Neerlandica.* 42(4), 233-252. https://doi.org/10.1111/j.1467-9574.1988.tb01238.x

[10] Menard, Scott (2002). *Applied Logistic Regression Analysis*, 2. SAGE. 2 edition.

[11] Breiman, Leo and Friedman, Jerome and Olshen, Richard A. and Stone, Charles J. (1984). *Classification and regression trees.* The Wadsworth statistics / probability series. CRC. https://doi.org/10.1201/9781315139470

[12] James, Gareth and Witten, Daniela and Hastie, Trevor and Tibshirani, Robert (2021). *An introduction to statistical learning*, 112. Springer, 2 edition.

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

[14] Dietterich, Thomas G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2), 139-157.

[15] Breiman, Leo (2001b). Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3), 199-231. https://doi.org/10.1214/ss/1009213726

[16] Hastie, Trevor and Tibshirani, Robert and Friedman, Jerome (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, 2 edition. https://doi.org/10.1007/978-0-387-84858-7

[17] Valiant, Leslie G. (1984). A Theory of the Learnable. *Communications of the ACM*, 27(11), 1134-1142. https://doi.org/10.1145/800057.808710

[18] Kearns, Michael and Valiant, Leslie G. (1994). Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM*, 41(1), 67-95. https://doi.org/10.1145/174644.174647

[19] Freund, Yoav and Schapire, Robert E. (1996) Experiments with a New Boosting Algorithm. *Machine Learning: Proceedings of the Thirteenth International Conference*. 148-156.

[20] Chen, Tianqi and Guestrin, Carlos. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 16*, 785-794.

[21] Ke, Guolin and Meng, Qi and Finley, Thomas and Wang, Taifeng and Chen, Wei and Ma, Weidong and Ye, Qiwei and Liu, Tie-Yan (2016). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Advances in Neural Information Processing Systems*. 30.

[22] Jurafsky, Daniel and Martin, James H. (2008). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall. 3 edition.

[23] Alonzo, Todd A. and Pepe Margaret S. (2002). *Distribution-free ROC analysis using binary regression techniques*.

[24] Fawcett, Tom (2006). An introduction to ROC analysis. *Pattern recognition letters*, 27(8): 861-874.

[25] Erin LeDell and Navdeep Gill and Spencer Aiello and Anqi Fu and Arno Candel and Cli Click and Tom Kraljevic and Tomas Nykodym and Patrick Aboyoun and Michal Kurka and Michal Malohlava (2022). *h2o: R Interface for 'H2O'*. R package version 3.38.0.1.

[26] Tianqi Chen and Tong He and Michael Benesty and Vadim Khotilovich and Yuan Tang and Hyunsu Cho and Kailong Chen and Rory Mitchell and Ignacio Cano and Tianyi Zhou and Mu Li and Junyuan Xie and Min Lin and Yifeng Geng and Yutian Li (2023). xgboost: Extreme Gradient Boosting. R package version 1.7.5.1.

[27] Yu Shi, Guolin Ke, Damien Soukhavong, James Lamb, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu, Nikita Titov and David Cortes. (2023). *LightGBM*. R package version 3.3.5.

[28] Haslbeck J (2023). *mnet: Modeling Group Differences and Moderation Effects in Statistical Network Models*. R package version 0.1.1.

[29] Beaulieu, Alan (2009). *Learning SQL*. O'Reilly Media, 2 edition.

[30] Breiman, Leo (1996). Bagging predictors. *Machine learning*, 24(2), 123-140.

[31] Stahlbock, Robert and Weiss, Gary M. and Abou-Nasr, Mahmoud and Yang, Cheng-Ying and Arabnia, Hamid R. and Deligiannidis, Leonidas. (2021). *Advances in Data Science and Information Engineering*. Springer.