Universidad de Valladolid

Facultad de Ciencias

Trabajo Fin de Grado

Grado en Estadística

Modelo de clasificación para la reducción de la tasa de abandono (Churn Rate) en una empresa del sector de distribución minorista

Autora: Cristina Bocho Moreno

Tutora: Teresa González Arteaga

Índice

Resumen

En cualquier estrategia de reducción de la de tasa de abandono es obligatorio saber por qué abandonan los clientes. Sólo conociendo esto, será posible evitar su marcha actuando con planes de retención adecuados. El objetivo de este trabajo es exponer el proceso para llegar a descubrir estas reglas que permiten anticipar la baja de un cliente. Para ello se implementa todo el proceso para un caso de estudio concreto en el sector minorista utilizando la librería rpart de R. El trabajo incluye desde la comprensión de los datos hasta la evaluación e implementación del modelo analítico, tal como se contempla en la metodología CRISP-DM (Cross Industry Standard Process for Data Mining) para el desarrollo de proyectos de minería de datos.

Abstract

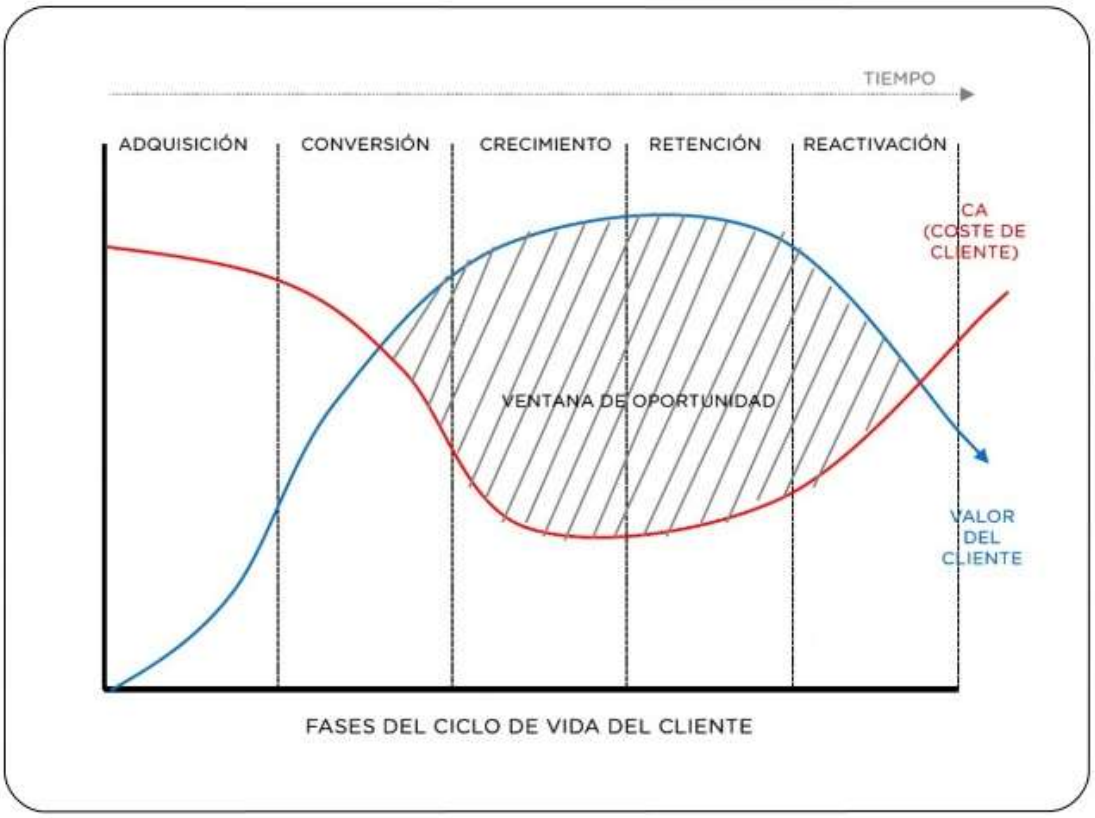
In any strategy to reduce the attrition rate, it’s mandatory to know why customers leave. Only by knowing this will it be possible to prevent them from leaving by acting with appropriate retention plans. The objective of this paper is to expose the process to discover these rules that allow us to anticipate a client's cancellation. To do this, the entire process is implemented for a specific study case in the retail sector using rpart library of R. The work includes all steps from understanding the data to evaluating and implementing the analytical model, as contemplated in the CRISP-DM (Cross Industry Standard Process for Data Mining) methodology for the development of data mining projects.

1 Introducción

El abandono de los clientes es con seguridad el mayor problema al que se enfrenta una empresa, con independencia de su sector, y más aún si nos encontramos en mercados altamente saturados. En estos mercados se requiere de un esfuerzo muy alto para conseguir la captación de nuevos clientes y por lo tanto se considera crucial para el éxito del negocio lograr la retención de los mismos, máxime cuando se ha invertido gran parte de los ingresos de la empresa en la captación de los mismos (ver Ilustración 1).

De hecho, podríamos afirmar, que tener una tasa de abandonos elevada, significa que una parte importante de los ingresos de una empresa se invertiran en la captación de nuevos clientes que reemplazaran a los que abandonan.

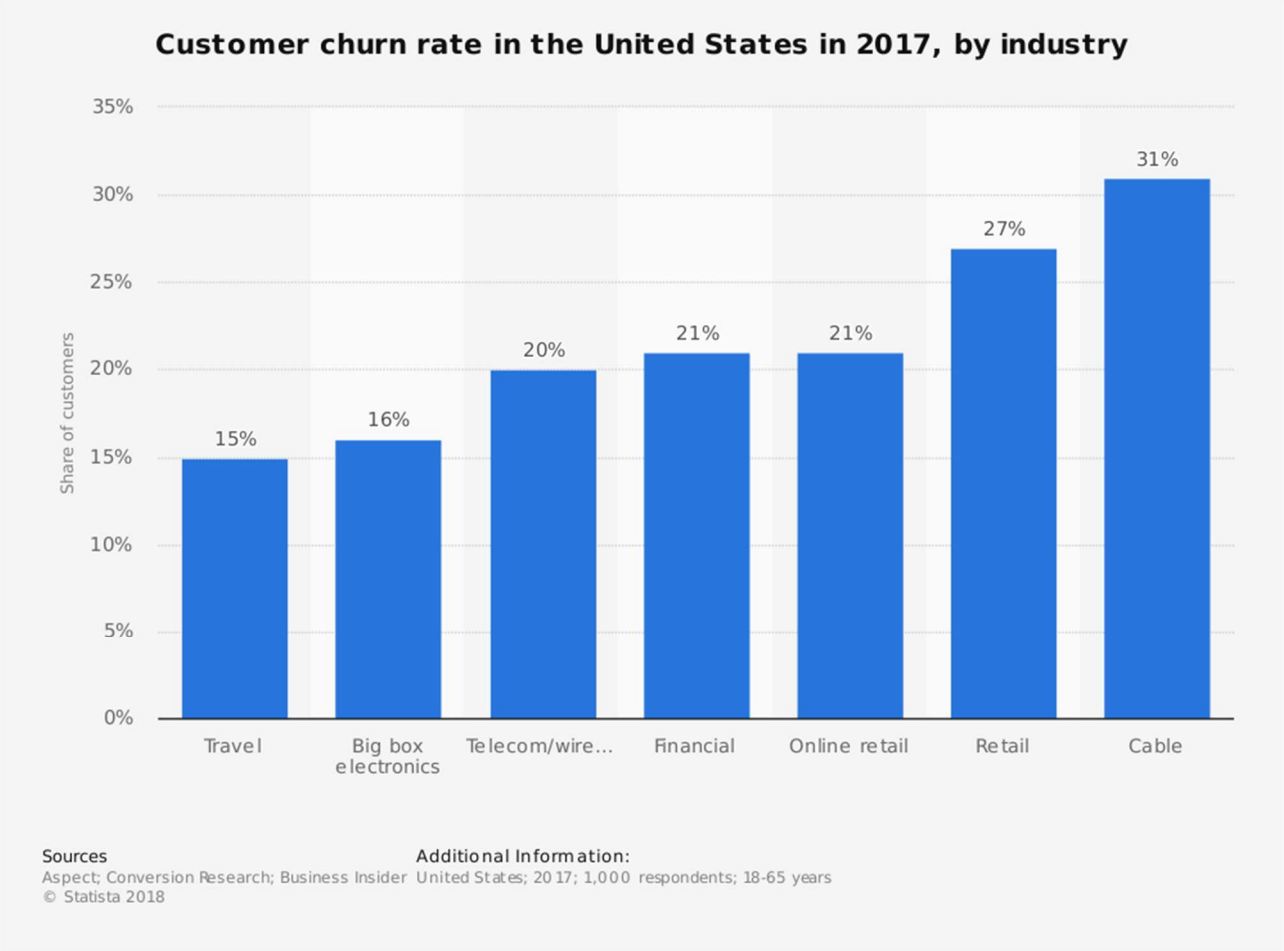
Ilustración 1. Gráfico estándar de las Etapas del Ciclo de Vida de los clientes



(Fuente: https://image.slidesharecdn.com/ciclodevidadelcliente)

Existen ciertas estadísticas al respecto que son estándar por industria, teniendo por ejemplo en el caso concreto del sector de distribución minorista, que esta tasa de abandono se encuentra en torno al 27% dentro del territorio americano (ver Ilustración 2). Atendiendo a esta cifra, el caso de estudio que tratamos en este TFG presenta una tasa de abandono cercana al 20%, por lo que no nos encontramos en una situación crítica. Sin embargo, conseguir una reducción en esta tasa para cualquier industria, implica un incremento considerable en ingresos, debido principalmente a la reducción en el esfuerzo de captación de nuevos clientes que reemplacen en igual o mayor medida a los abandonistas.

Ilustración 2. Distribución de la tasa de abandono por sector en Estados Unidos



En el caso de España no hay publicado un estudio similar donde se encuentre la tasa de abandono por sector industrial y que nos dé una comparativa más real de esta tasa en la empresa objeto del TFG (a partir de ahora ***EmpresaXX***) con las de su mismo sector.

Hay compañías que hoy redoblan sus esfuerzos en captar clientes más que en retenerlos, si analizamos las cifras estas hablan por sí solas.

Según una encuesta de la firma norteamericana CRMGURU, el 80% de los directores generales considera la lealtad de clientes como un asunto de "alta o extrema importancia". Sin embargo, algo debe fallar cuando el 40% asegura dar prioridad a la captación de clientes nuevos y sólo el 22% prioriza la retención de clientes en cartera.

La apuesta por la captación en detrimento de la retención puede ser calificada como error. No es que la retención sea más importante que la búsqueda de nuevos clientes, pero está claro que ambos procesos deben ir en paralelo y que la retención es más fácil, lógica y rentable en términos económicos.

Ahora la pregunta de fondo es ¿cómo saber cuáles son los clientes que tienen la más alta probabilidad de abandonarnos?

Existen técnicas tales como los *mapas de abandono* que sirven para obtener una descripción clara de la situación de fuga de clientes de una compañía y nos entrega un estado de situación descriptivo útil para un diagnóstico en profundidad. Un mapa de este tipo debe contener como mínimo:

- Perfil de clientes abandonistas.
- Qué establecimientos o sucursales son los que tienen el más alto porcentaje de abandono.
- Qué zonas geográficas son las más afectadas.
- Etc…

Uno de los pasos más críticos en la prevención del abandono es la obtención de las reglas que nos permitan identificar cuándo un cliente podría entrar en alto riesgo de

abandono y por lo tanto focalizar donde deben ponerse los esfuerzos de la compañía para su retención.

Una técnica en minería de datos que nos permitirá la obtención de estas reglas son los denominados árboles de decisión. Dentro de esta familia de algoritmos el que se considera para el presente trabajo es el árbol CART. Esta técnica, es idónea para la obtención de árboles bietápicos, y se ha optado por su utilización debido a su sencilla interpretación e implementación en un sistema de base de datos relacional, ya que podemos convertir las reglas obtenidas en sencillas sentencias IF-THEN-ELSE en lenguaje SQL. Además, esta técnica permite trabajar con variables independientes categóricas, como es el caso que interesa en este TFG y presenta además una alta tolerancia a variables con outliers.

Este TFG se desarrolla de la siguiente forma. En el apartado 2 describimos en que consiste el árbol CART. A continuación en el apartado 3 se presenta el caso de estudio. Describimos el problema en la empresa XX, cómo se recopilan los datos para el análisis y definimos el contenido de la base de datos. En el apartado 4 se procede a explicar cómo emplear R para la construcción del modelo. Vemos la etapa de entrenamiento, evaluación y despliegue. Finalmente en el apartado 5 se incluyen unas conclusiones al TFG.

2 Metodología

Para el desarrollo del TFG, hemos utilizado como estructura la contemplada en la metodología para el desarrollo de modelos estadísticos denominada CRISP-DM. Esta metodología considera seis etapas como necesarias para el correcto desarrollo de un modelo y pueden encontrarse detalladas en el apartado de Anexos.

El problema planteado, consistente en disminuir la tasa de abandono de los clientes miembros del programa de fidelidad de la empresa XX, se aborda a partir del desarrollo de un árbol de decisión CART (Classification and Regression Trees) (para más detalle consultar: Leo Breiman (Autor), Jerome Friedman (Autor), Charles J. Stone (Autor), R.A. Olshen (Autor) *Classification and Regression Trees (Wadsworth Statistics/Probability)*, 1984).

Como principales ventajas para la elección de la técnica de árbol CART podemos destacar las siguientes:

- No necesita hipótesis acerca de la distribución de las variables.
- Puede trabajar con datos de distintos tipos: categóricos y continuos.
- Sus resultados son robustos a los outliers.
- Permite combinaciones lineales entre las variables.
- Selecciona automáticamente las variables que más reducen los errores de clasificación.

El proceso seguido para el desarrollo del modelo puede esquematizarse en cuatro fases: (1) construcción del árbol, (2) parada del proceso de crecimiento del árbol (se constituye un árbol máximo que sobreajusta la información contenida en nuestra base de datos), (3) podado del árbol haciéndolo más sencillo y dejando sólo los nodos más importantes y, por último, (4) selección del árbol óptimo con capacidad de generalización.

La construcción del árbol comienza en el nodo raíz, que incluye todos los registros de la base de datos. A partir de este nodo el programa debe buscar la variable más adecuada para partirlo en dos nodos hijos. Para elegir la mejor variable debe utilizarse una medida

de pureza en la valoración de los dos nodos hijos posibles (la variable que consigue una mayor pureza se convierte en la utilizada en primer lugar, y así sucesivamente). Debe buscarse una función de partición que asegure que la pureza en los nodos hijos sea la máxima. La función utilizada en el caso del árbol CART es la denominada Gini (se alcanza un índice de pureza que se considera como máximo).

El índice de Gini en el nodo *t*, *g(t)*, se puede formular del modo siguiente:

$$g(t) = \sum_{j \neq i} p\,(j/t)\; p\,(i/t)$$

donde *i* y *j* son las categorías de la variable predictora y *p* es proporción.

La función de criterio Φ *(s,t)* para la división *s* en el nodo *t* se define como:

$$\Phi(s,t) = g(t) - p_L g(t_L) - p_R g(t_R)$$

donde p_L es la proporción de casos de t enviados al nodo hijo de la izquierda, y p_R al nodo hijo de la derecha.

Cuando se comienza en el nodo raíz $t = 1$ (y también en las particiones sucesivas), se busca la división s^*, de entre todas las posibles de S, que de un valor con mayor reducción de la impureza:

$$\Phi(s^*,1) = \max_{s \in S} \Phi(s,1)$$

Luego se divide el nodo 1 en 2 nodos hijos $(t = 2\ y\ t = 3)$ utilizando la división s^*. Este valor de la función de impureza, ponderado por la proporción de todos los casos del nodo *t*, es el valor del que se informa en el árbol como «mejora».

En el proceso sucesivo de construcción y crecimiento del árbol se debe asignar una clase (etiqueta) a cada nodo (desde la raíz hasta los nodos hoja). El procedimiento de asignación de clase debe hacerse por medio de una función de asignación, en la que se

tiene en cuenta la probabilidad a priori asignada a cada clase, la pureza de la partición y la proporción final de casos que aparecen en los nodos hojas. Al igual que puede determinarse la pureza para un nodo concreto, puede evaluarse de forma conjunta para todo el árbol.

El crecimiento de un árbol continúa hasta que se produce cualquiera de estas 3 posibilidades: sólo hay una observación (caso) en cada nodo hoja, todas las observaciones tienen la misma probabilidad asignada en los nodos hoja (es imposible determinar el criterio de máxima pureza), o se ha fijado un límite externo de la profundidad (número de niveles máximo) del crecimiento del árbol. El árbol que se ha generado de esta forma clasifica correctamente los registros utilizados en su proceso de aprendizaje (se dice que este «sobreaprendizaje» se obtiene porque el modelo ha «sobreajustado» los datos empleados en esta fase), pero cuando se enfrente a nuevos registros no se asegura su capacidad de generalización.

El árbol complejo que se ha creado debe simplificarse para que alcance esta capacidad de generalización. Se utiliza un método de podado del árbol. El procedimiento asegura que sólo se retiran los nodos que incrementan muy poco la precisión del árbol. Se utiliza una medida de coste-complejidad (que combina los criterios de precisión frente a complejidad en el número de nodos y velocidad de procesamiento), buscando el árbol que obtiene menor valor en este parámetro. Los árboles más sencillos (podados con este criterio) aseguran una mayor capacidad de generalización.

De todos los árboles podados posibles debe seleccionarse el mejor. El mejor árbol (árbol solución) será el que consigue menor error en el ajuste de los registros utilizados en su proceso de aprendizaje. Pero esta condición no es suficiente, debe ajustar bien la base de datos utilizada en su aprendizaje, pero también debe ajustar registros no empleados en esta fase.

Para comparar los distintos modelos se medirán sus propiedades de sensibilidad, especificidad, área bajo la curva ROC y porcentaje de correcta clasificación (todos estos parámetros se definen a lo largo del desarrollo del modelo).

Para la realización de los cálculos estadísticos durante el desarrollo haremos uso del software estadístico R. Las librerías utilizadas figuran descritas más adelante, concretamente en la parte del código donde se requieren en primer lugar.

Partiendo de las conclusiones del TFG como base, sería muy interesante comparar los resultados obtenidos con otros algoritmos de clasificación, por ejemplo:

- **Árbol C4.5 ó C5**: Este algoritmo, diseñado por Quinlan, es uno de los más empleados en el entorno de los árboles de decisión, sustituyendo el Índice de Gini como medida de selección de variables que emplea el CART por una medida basada en entropía, el Gain Ratio, definido como $I(X_i, C) / H(X_i)$. De esta manera se consigue evitar que las variables con mayor numero de posibles valores salgan beneficiadas en la selección. Al igual que el CART presenta las principales características de los árboles de decisión, pero además hay que destacar que los cortes dentro de cada nivel no tienen que ser binarios, como ocurre en el CART, pudiendo ser múltiples. Esta cualidad facilita en muchos problemas una mejor adaptación a las variable de entrada y una posible mejora en el performance del modelo de clasificación.

- **Regresión logística**: Ampliamente utilizada, sobre todo para problemas con target binario ofrece muy buenos resultados y una ecuación resultante fácilmente implementable en cualquier sistema de BD. Lo que es una ventaja frente a los árboles de decisión que tienen una implementación algo más laboriosa. Para el uso de la regresión logística sería necesario un preprocesado de las variables en cuanto a:

    - Transformación de variables discretas en n-1 variables "dummy".
    - Imputación de nulos o perdidos.
    - Tratamiento de outliers.

- Se podría probar también con la evolución que Breiman dio a sus árboles de decisión (CART) que son los Random Forest, demostrando que una cantidad elevada de clasificadores débiles pueden dar lugar a muy buenos resultados en problemas de clasificación.
- Por último se propondría usar los modelos más de actualidad, excluyendo el Deep learning, como son el GBM o XGboost.

También sería interesante validar los resultados del modelo en t períodos, esto es entrenar el modelo en el período t y probar con períodos sucesivos que la performance del modelo se mantiene y que no se ha sobreajustado a un momento temporal concreto. Este tipo de pruebas son de gran interés cuando se va a hacer el paso de un modelo a un entorno de producción en la empresa, ya que se deberá ejecutar de forma periódica, por ejemplo mensual, y sería un proceso además del que se esperaría una gran estabilidad.

3 Presentación del caso de estudio

3.1 Descripción del problema

En el año 2011 comenzó un programa de fidelización en una empresaXX del sector de distribución minorista. A continuación, se presenta la evolución de los miembros de dicho programa al final de cada ejercicio, entendiendo por el término ejercicio lo siguiente:

- **Ejercicio**: Periodo de tiempo de 12 meses que da comienzo el 1 de marzo y finaliza el último día de febrero del año siguiente. Adicionalmente, podemos indicar que un ejercicio se divide a su vez en dos temporadas, una primera denominada Primavera-Verano (en adelante Pri/Ver) y una segunda denominada Otoño-Invierno (en adelante Oto/Inv).

En la tabla siguiente (ver Tabla 1), se puede ver la distribución de los miembros del programa en función de su segmento de actividad para los ejercicios 2013 a 2018. Los términos que aparecen en la tabla se definen como:

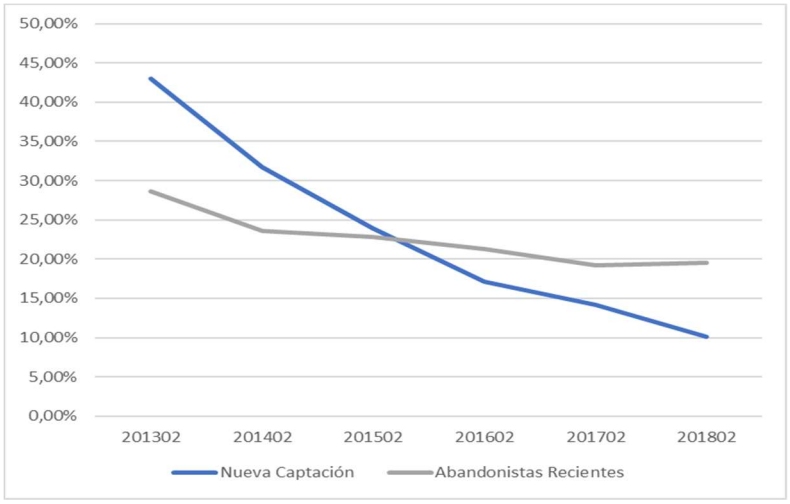
- **Segmento de actividad**: Clasificación que se realiza en función del tiempo que un cliente lleva sin comprar. Pudiéndose encontrar la siguiente clasificación para los clientes miembros del programa de fidelización:
  - **Nueva captación**: Se han dado de alta en el último ejercicio y han realizado al menos una compra durante el mismo.
  - **Activos**: Han realizado al menos una compra en el último ejercicio y tienen al menos un año de antigüedad en el programa.
  - **Abandonistas recientes**: Sin compras en el último ejercicio.
  - **Abandonistas**: Sin compras en más de dos ejercicios.

En la ilustración 3 se representan valores de la tabla. Se puede observar cómo el segmento de nueva captación va reduciendo su peso según se aproximan los ejercicios al momento actual y de igual forma, cómo también va creciendo el parque de clientes dentro de segmentos abandonistas.

Tabla 1. Distribución de los clientes por segmento y ejercicio

Peso de Miembros del Club al final de cada ejercicio	Ejercicio					
Segmento de Actividad	201302	201402	201502	201602	201702	201802
Nueva Captación	43%	32%	24%	17%	14%	10%
Activos	28%	29%	30%	30%	29%	27%
Abandonistas Recientes	29%	24%	23%	21%	19%	20%
Abandonistas	0%	15%	24%	31%	37%	43%
Total general	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**

Ilustración 3. Evolución de las nuevas captaciones frente a los abandonistas



En la ilustración anterior, se aprecia cómo la Nueva Captación va cayendo y cómo los abandonistas recientes se mantienen constantes a lo largo del tiempo, alcanzando un punto de equilibrio a principios del ejercicio del 2015.

Nos preguntamos: ¿Qué valor tienen las nuevas captaciones? ¿Cuál era el valor de los socios que nos abandonan un año antes de producirse su abandono? Antes de responder a esta pregunta, es necesario explicar el concepto "decil de actividad".

- **Decil de actividad**: Es el segmento de valor en el que se encuentra el miembro al final de cada ejercicio. Se ordenan los clientes en función del gasto y se hacen 10 divisiones iguales. Se asigna el valor 1 a los miembros con más valor. Si el cliente no ha tenido actividad se le asigna el valor 0.

Si nos centramos en la nueva captación vemos que más del 30% de los socios pertenecen a deciles de poco consumo (ver Tabla 2). En cambio, podemos observar que son los socios de los primeros deciles quiénes generan más ingresos (ver Tabla 3).

Los dos primeros deciles de socios, es decir un 14% de la nueva captación, son los que generan el 45% de las ventas de dicho segmento, independientemente del año de alta del miembro.

A la vista de la Tabla 3 podemos destacar también, el descenso de ingresos según va incrementando la antigüedad del programa de fidelidad. Los primeros años se captaba con más facilidad, en cambio ahora, atraer a nuevos clientes resulta más complicado.

Tabla 2. Distribución de clientes nuevos por ejercicio y valor

Nº de miembros	Decil de Actividad										
Clasificación	1	2	3	4	5	6	7	8	9	10	Total general
NUEVA CAPTACION 13	6,27%	8,19%	9,19%	10,00%	10,62%	10,98%	11,06%	11,52%	12,38%	9,79%	100,00%
NUEVA CAPTACION 14	6,04%	7,89%	9,04%	9,88%	10,60%	11,20%	11,08%	12,56%	12,18%	9,53%	100,00%
NUEVA CAPTACION 15	5,86%	7,95%	8,94%	9,66%	10,29%	11,08%	12,97%	12,39%	10,27%	10,59%	100,00%
NUEVA CAPTACION 16	6,02%	8,04%	8,85%	9,81%	10,56%	10,80%	12,02%	11,64%	11,73%	10,53%	100,00%
NUEVA CAPTACION 17	6,10%	8,02%	8,96%	9,99%	10,31%	11,26%	12,66%	12,19%	10,43%	10,08%	100,00%
Total general	**6,06%**	**8,02%**	**9,00%**	**9,87%**	**10,49%**	**11,06%**	**11,87%**	**12,06%**	**11,50%**	**10,07%**	**100,00%**

Tabla 3. Total de ventas por ejercicio y valor de nuevos clientes (Millones de euros)

Venta	Decil de Actividad										
Clasificación	1	2	3	4	5	6	7	8	9	10	Total general
NUEVA CAPTACION 13	8,2	5,4	4,1	3,3	2,7	2,1	1,6	1,3	0,9	0,3	30,0
NUEVA CAPTACION 14	7,5	5,0	3,9	3,2	2,6	2,1	1,6	1,3	0,8	0,2	28,2
NUEVA CAPTACION 15	6,2	4,3	3,3	2,6	2,1	1,7	1,5	1,0	0,6	0,2	23,5
NUEVA CAPTACION 16	5,9	4,0	3,0	2,5	2,0	1,6	1,3	0,9	0,6	0,2	21,9
NUEVA CAPTACION 17	5,2	3,4	2,6	2,1	1,7	1,4	1,1	0,8	0,5	0,2	19,0
Total general	**33,0**	**22,0**	**17,0**	**13,7**	**11,0**	**8,9**	**7,1**	**5,3**	**3,4**	**1,2**	**122,6**

Ahora hacemos el mismo ejercicio para los abandonistas recientes con la peculiaridad que la venta estudiada es de un año anterior, o lo que es lo mismo, cuando el cliente aún estaba activo.

La distribución de socios también es más alta en los deciles de actividad menores, alcanzando más de un 30% en los tres últimos grupos (ver Tabla 4).

Tabla 4. Distribución de clientes abandonistas por ejercicio y valor

Nº de miembros	Decil de Activid										
Clasificación	1	2	3	4	5	6	7	8	9	10	Total general
ABAND. RECIENTE 13	4,37%	6,69%	8,05%	9,23%	10,36%	11,16%	11,95%	12,29%	12,75%	13,14%	100,00%
ABAND. RECIENTE 14	4,22%	6,62%	8,08%	9,29%	10,43%	11,21%	11,81%	12,38%	12,98%	12,97%	100,00%
ABAND. RECIENTE 15	4,28%	6,60%	8,12%	9,27%	10,48%	11,38%	11,62%	12,48%	13,01%	12,75%	100,00%
ABAND. RECIENTE 16	4,26%	6,67%	8,05%	9,17%	10,17%	11,36%	12,14%	12,62%	12,62%	12,94%	100,00%
ABAND. RECIENTE 17	4,31%	6,67%	7,96%	9,20%	10,48%	11,06%	12,00%	12,43%	12,77%	13,10%	100,00%
Total general	**4,28%**	**6,65%**	**8,05%**	**9,23%**	**10,39%**	**11,23%**	**11,91%**	**12,45%**	**12,82%**	**12,98%**	**100,00%**

Respecto a la venta (ver Tabla 5) que generan estos clientes, parece ser similar a lo que vimos en los miembros de nueva captación. Los primeros deciles de actividad, que representan un 11% del segmento, son los que generan un 38% de la venta del mismo.

Estos clientes abandonistas han generado una venta inferior durante los primeros años del programa si lo comparamos con los de nueva captación, pero según ha ido incrementando la antigüedad del programa, estas ventas se han ido igualando.

Tabla 5. Total de ventas por ejercicio y valor de abandonistas recientes (Millones de euros)

Venta	Deci										
Clasificación	1	2	3	4	5	6	7	8	9	10	Total
ABAND. RECIENTE 13	4,3	3,5	2,9	2,5	2,1	1,7	1,4	1,1	0,8	0,3	20,6
ABAND. RECIENTE 14	5,0	4,1	3,5	2,9	2,5	2,1	1,7	1,3	0,9	0,4	24,3
ABAND. RECIENTE 15	5,5	4,5	3,8	3,2	2,7	2,3	1,8	1,4	1,0	0,4	26,6
ABAND. RECIENTE 16	5,7	4,7	3,9	3,2	2,7	2,3	1,8	1,4	1,0	0,4	27,1
ABAND. RECIENTE 17	6,2	5,1	4,2	3,5	3,0	2,4	2,0	1,5	1,0	0,4	29,4
Total general	**26,7**	**21,9**	**18,2**	**15,3**	**13,1**	**10,8**	**8,7**	**6,7**	**4,7**	**1,7**	**127,8**

A la vista de todo lo que hemos ido viendo, se podría considerar que efectivamente resulta relevante para la compañía identificar qué clientes podrían abandonar, ya que cada año que va pasando se va inclusive complicando más la labor de captación.

3.2 Recopilación de los datos

Actualmente la recogida y el procesamiento de los datos es de vital importancia en las compañías, con independencia del sector. En estos últimos años son numerosas las noticias que giran en torno a la importancia y el valor que tienen los datos al compararlos incluso con el petróleo del siglo XXI. De hecho se estima que las empresas invertirán más de 40.000 millones de euros al año a partir de 2019, según los cálculos de la consultora IDC. Por este motivo, las empresas despliegan complejas arquitecturas de sistemas para recoger cada vez más información de todo tipo y de una gran variedad de fuentes.

Para poder entender los datos de los que se dispone en el presente trabajo, es necesario describir cómo se recoge la misma en los sistemas de la compañía elegida. Se trata de una empresa cuya recolección de los datos asociados a las ventas y a los clientes se realiza principalmente en los terminales de puntos de venta (TPV) de su red de tiendas. Dentro de las acciones que pueden registrarse a través de estos TPV, podemos diferenciar las dos siguientes:

- Asociar la venta con los miembros del programa de fidelización siempre que se identifiquen como tal en el momento de hacer su compra.
- Una persona puede adherirse al programa de fidelización e incorporar sus datos personales a la base de datos de clientes objeto de nuestro TFG.

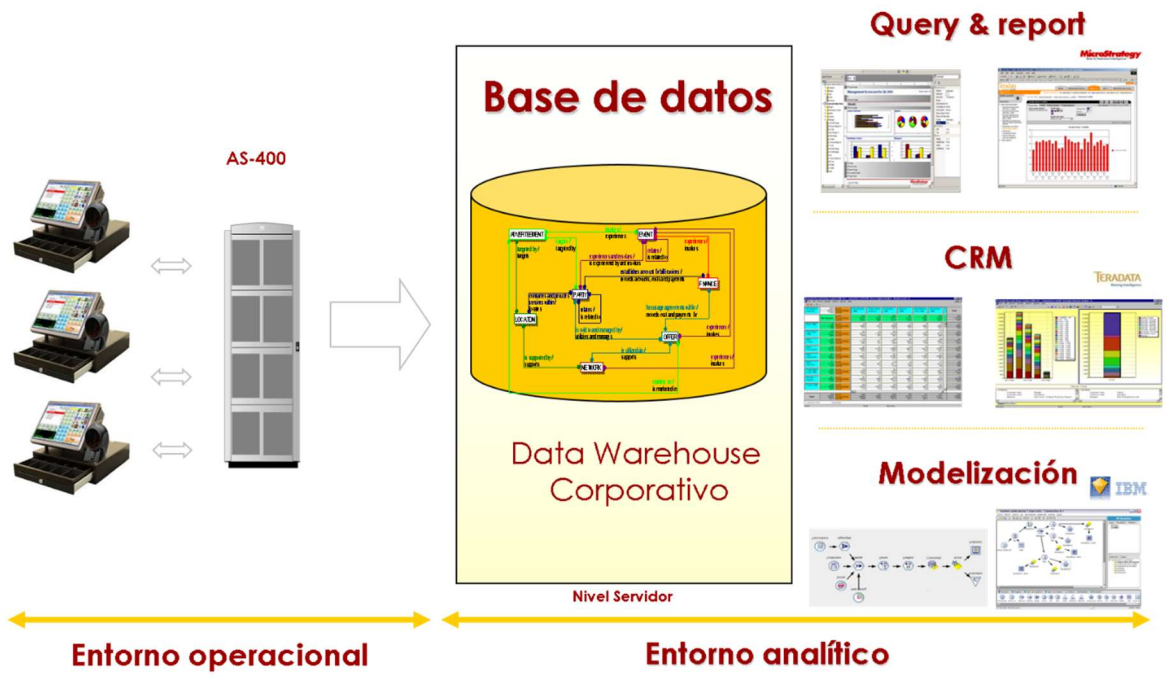
Una vez concluida cualquiera de las dos acciones anteriores, los datos navegan hasta los sistemas de la compañía para terminar almacenándose en tablas de un servidor IBM AS400. Este proceso se lleva a cabo en tiempo real al tratarse lógicamente de uno de los sistemas operacionales de la empresa.

Posteriormente y a través de procesos ETL (Extraction, Transformation and Load), estos datos se almancenan en el Data Warehouse, donde a continuación se modelan hacia una capa lógica con el propósito de realizar analítica sobre el negocio que se explota a través de herramientas para el tratamiento de información, como por ejemplo Microstrategy. Esto último formaría parte del entorno analítico de la empresa. También forman parte de este entorno los siguiente componentes:

- KPIs calculados a partir de los procesos citados anteriormente, que se encuentran almacenados en un datamart analítico. Estos KPIs se explotan habitualmente a través de sentencias SQL o directamente haciendo representaciones gráficas a través de una herramienta de reporting (p. ej. Microstrategy).
- Segmentaciones de negocio sobre colectivos de clientes que son utilizadas para acciones de marketing microsegmentadas a través del gestor de campañas de la empresa.
- Modelos predictivos.

Todo lo anterior puede verse representado en la ilustración siguiente.

Ilustración 4. Arquitectura de sistemas



(Fuente: Documento interno de la empresa XX)

3.3 Contenido de la base de datos

Los modelos basados en árboles de decisión, nos permiten clasificar el abandono de los clientes en función de información propia de los mismos, determinando qué variables son las más relevantes para ello. Por tanto, para llevar a cabo su desarrollo habrá que considerar al menos los siguientes tipos de variable:

- Variable objetivo o dependiente: Esta es la variable que determina si un cliente es baja o no en nuestro caso.
- Variables independientes o explicativas: Variables que explican la acción de darse de baja. Estas a su vez pueden clasificarse en:
    - Variables inherentes al cliente (sociodemográficas): edad, nivel cultural, ciudad, etc…
    - Variables comportamentales: productos comprados, mes de compra, gasto, etc…
- Variables exógenas (externas): no están controlados por el cliente, pudiendo encontrar cualquiera de los siguientes:
    - Descuento en precios e incentivos monetarios.
    - Ofertas, regalos, promociones...
- Variables de consumo: variables que nos van a determinar en qué medida ejerce la actividad de compra el cliente.
    - Volumen de compras.
    - Volumen de devoluciones.
    - Tipos de productos adquiridos.
    - Etc…

Con fines analíticos, se ha creado un data mart donde se encuentran agregados los datos por la dimensión tiempo (día, mes, año, temporada, ejercicio,…), de modo que puedan identificarse patrones de comportamiento de los miembros del programa de fidelización. Para el presente trabajo, se han considerado las variables agrupadas por temporada, disponiendo de información relativa a seis temporadas, o lo que es lo mismo, de tres ejercicios. Toda esta información se ha extraído a partir de sentencias SQL (ver Anexos). Paso a detallar a continuación las variables que conforman el fichero de datos original.

- Variable objetivo (*ABANDONA*): Para realizar el cálculo de esta variable se han seguido los siguientes criterios.
    - Partimos del indicador que determina si el miembro del programa ha tenido actividad de compra durante las cuatro últimas temporadas, denominado *F_ACTIVIDAD_[temporada]*. Considerando que un cliente comienza a ser abandonista cuando lleva dos temporadas sin comprar se han marcado los clientes de la siguiente forma:

```
IF

(F_ACTIVIDAD = 0 and F_ACTIVIDAD_1 = 0 and F_ACTIVIDAD_2 = 1)

OR

(F_ACTIVIDAD = 0 and F_ACTIVIDAD_1 = 0 and F_ACTIVIDAD_3 = 1)

THEN 1

ELSE 0
```

Atendiendo a este hecho, se ha decidido eliminar del fichero aquellas variables que se correspondan con las temporadas 0 y 1, ya que como hemos indicado anteriormente, para que el cliente se le considere abandonista no ha debido tener ninguna actividad en el último ejercicio. Teniendo esto en cuenta, la información correspondiente a los clientes abandonistas para las dos últimas temporadas ha sido nula, y precisamente lo que queremos identificar es el comportamiento que haya tenido mientras aún presentaba actividad.

Descripción de las variables utilizadas.

- *ID*: Identificador único del cliente.
- *TRAMO_EDAD*: Intervalo de edad en el que se encuentra el cliente en el momento de extraer los datos. Los valores posibles son: "18-24", "25-29", "30-34", "35-39", "40-44", "45-49", "50-54", "55-59", "60-64", "65-69", "70 y más".
- *GASTO_[num_temporada]*: Suma de los importes netos de venta (sin descuentos ni impuestos) de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado. Si no existen operaciones en el periodo evaluado se asigna el valor cero.

- *GASTO_CAB_[num_temporada]*: Suma de los importes netos de venta (sin descuentos ni impuestos) de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado, de productos que pertenezcan a la sección de caballero. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *GASTO_SRA_[num_temporada]*: Suma de los importes netos de venta (sin descuentos ni impuestos) de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado, de productos que pertenezcan a la sección de señora. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *GASTO_VIP_CAB_[num_temporada]*: Suma de los importes netos de venta (sin descuentos ni impuestos) de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado, de productos que pertenezcan a la gama más alta de productos de la sección de caballero. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *GASTO_VIP_SRA_[num_temporada]*: Suma de los importes netos de venta (sin descuentos ni impuestos) de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado, de productos que pertenezcan a la gama más alta de productos de la sección de señora. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *GASTO_VSD_[num_temporada]*: Suma de los importes netos de venta (sin descuentos ni impuestos) de las operaciones (tanto de compra como de devolución) realizadas en viernes, sábado o domingo. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *GASTO_PROMO_[num_temporada]*: Suma de los importes netos de venta (sin descuentos ni impuestos) de las operaciones (tanto de compra como de devolución) realizadas con aplicación de alguna promoción. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *GASTO_DCTO_[num_temporada]*: Suma de los descuentos (sin impuestos) de las operaciones (tanto de compra como de devolución) realizados en el periodo evaluado. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.

- *NUM_DIST_GFAM_[num_temporada]*: Suma del número de distintas grandes familias adquiridas durante el período evaluado. Se deben tener en cuenta las operaciones de compra y devolución realizadas en el período evaluado. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_DIST_FAM_[num_temporada]*: Suma del número de distintas familias adquiridas durante el período evaluado. Se deben tener en cuenta las operaciones de compra y devolución realizadas en el período evaluado. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_ART_[num_temporada]*: Suma del número de artículos de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_ART_CAB_[num_temporada]*: Suma del número de artículos de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado, de aquellos patrones que pertenezcan a la sección de caballero. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_ART_SRA_[num_temporada]*: Suma del número de artículos de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado, de aquellos patrones que pertenezcan a la sección de señora. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_ART_PROMO_[num_temporada]*: Suma del número de artículos de las operaciones (tanto de compra como de devolución) realizadas con alguna promoción. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_ART_VSD_[num_temporada]*: Suma del número de artículos de las operaciones (tanto de compra como de devolución) realizadas en viernes, sábado o domingo. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.

- *NUM_ART_VIP_CAB_[num_temporada]*: Suma del número de artículos adquiridos en las operaciones (tanto de compra como de devolución) realizadas en el período evaluado, que pertenezcan a la gama más alta de productos de la sección de caballero. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_ART_VIP_SRA_[num_temporada]*: Suma del número de artículos de las operaciones (tanto de compra como de devolución) realizadas en el período evaluado, que pertenezcan a la gama más alta de productos de la sección de señora. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_OPER_[num_temporada]*: Número de operaciones (distintos tickets) de COMPRA realizadas en el periodo evaluado. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_OPER_PROMO_[num_temporada]*: Número de operaciones de COMPRA realizadas en el período evaluado con alguna promoción. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *NUM_OPER_VSD_[num_temporada]*: Número de operaciones de COMPRA realizadas en el período evaluado en viernes, sábado o domingo. Si no existen operaciones de dicha sección en el periodo evaluado se asigna el valor cero.
- *ARRIBA_[num_temporada]*: Indica si el socio ha adquirido en el periodo evaluado algún producto que pertenezca a la parte superior. Los posibles valores son:
    - -99: No ha comprado producto de esas grandes familias en el período evaluado (valores nulos).
    - -1: Ha devuelto producto de esas grandes familias en el período evaluado.
    - 0: Ha comprado y devuelto producto de esas grandes familias en el período evaluado.
    - 1: Ha comprado producto de esas grandes familias en el período evaluado.
- *ABAJO_[num_temporada]*: Indica si el socio ha adquirido en el periodo evaluado algún producto que pertenezca a la parte inferior. Los posibles valores son:
    - -99: No ha comprado producto de esas grandes familias en el período evaluado (valores nulos).
    - -1: Ha devuelto producto de esas grandes familias en el período evaluado.

  - 0: Ha comprado y devuelto producto de esas grandes familias en el período evaluado.
  - 1: Ha comprado producto de esas grandes familias en el período evaluado.
- *SASTRE_[num_temporada]*: Indica si el socio ha adquirido en el periodo evaluado algún producto que pertenezca a sastrería. Los posibles valores son:
  - -99: No ha comprado producto de esas familias en el período evaluado (valores nulos).
  - -1: Ha devuelto producto de esas familias en el período evaluado.
  - 0: Ha comprado y devuelto producto de esas familias en el período evaluado.
  - 1: Ha comprado producto de esas familias en el período evaluado.
- *BANO_[num_temporada]*: Indica si el socio ha adquirido en el periodo evaluado algún producto que pertenezca a baño. Los posibles valores son:
  - -99: No ha comprado producto de esas familias en el período evaluado (valores nulos).
  - -1: Ha devuelto producto de esas familias en el período evaluado.
  - 0: Ha comprado y devuelto producto de esas familias en el período evaluado.
  - 1: Ha comprado producto de esas familias en el período evaluado.
- *F_ACTIVIDAD_[num_temporada]*: Flag que indica si el socio ha operado en el periodo evaluado. Sus posibles valores son:
  - 0: El socio no ha operado en el período evaluado
  - 1: El socio ha operado en ese periodo.
- *TICKET_MEDIO_[num_temporada]*: Importe de venta (incluyendo operaciones de venta y devolución) asociados al miembro en el período evaluado dividido entre el número de operaciones de COMPRA realizadas en el período.
- *APO_[num_temporada]*: Número de artículos (incluyendo operaciones de venta y devolución) asociados al miembro en el período evaluado dividido entre el número de operaciones de COMPRA realizadas en el período.
- *PORC_CONTRI_[num_temporada]*: Porcentaje de margen de contribución del PVP. Se calcula como la suma del importe neto de ventas (con operaciones de venta y devolución) menos la suma de los PVI netos (con operaciones de venta y

devolución), todo ello dividido entre la suma del importe neto de ventas, para cada período evaluado. Si un socio no ha operado en ese período asignar -1.

- *DECILA_[num_temporada]*: Es el segmento de valor en el que se encuentra el miembro al final de cada periodo. Se ordenan los clientes en función del gasto y se hacen 10 divisiones iguales. Se asigna 1 a los miembros con más valor. Si el cliente no ha tenido actividad se le asigna el valor -1. Se asigna el valor 0 si el cliente ha efectuado una compra y una devolución por el valor total de la misma.

La asignación del número de temporada [*num_temporada*] sigue el siguiente criterio:

- Sin valor: Datos desde 01-09-2017 a 28-02-2018 (Temporada Oto/Inv)
- 1: Datos desde 01-03-2017 a 31-08-2018 (Temporada Pri/Ver)
- 2: Datos desde 01-09-2016 a 28-02-2017 (Temporada Oto/Inv)
- 3: Datos desde 01-03-2016 a 31-08-2016 (Temporada Pri/Ver)
- 4: Datos desde 01-09-2015 a 28-03-2016 (Temporada Oto/Inv)
- 5: Datos desde 01-03-2015 a 31-08-2015 (Temporada Pri/Ver)

4 Construcción de un modelo CART

Después de describir el problema que pretendemos resolver y los datos de los que disponemos, comenzaremos el trabajo de desarrollo del modelo, bien para lo cual utilizaremos el software estadístico R.

La elección de R para la realización de este trabajo, viene determinada por los siguientes aspectos:

- Una de las librerías que tiene, denominada *rpart*, está especialmente destinada para el desarrollo del modelo propuesto en el presente trabajo.
- Dispone de un extenso catálogo de librerías para la resolución de cualquier problema estadístico.
- Es un software libre.
- Existe una gran comunidad donde poder encontrar multitud de recursos, siendo difícil no encontrar solución a las dudas que puedan surgir en su utilización.

Lo primero que debemos hacer es cargar las librerías que serán utilizadas durante este proceso. Pasamos a introducir los siguientes comandos en la consola de R para disponer de todos los paquetes necesarios durante el análisis, para posteriormente iniciar la construcción del dataset inicial.

```
library(caret)       # Usada para crear matrices de confusión
library(corrplot)    # Visualizar en modo gráfico matrices de correlación
library(dplyr)       # Creación de datasets Training y Test
library(ROCR)        # Cálculo y representación gráfica de curvas ROC
library(rpart)       # Árboles de clasificación CART
library(rpart.plot) # Visualizaciones sobre los resultados de rpart.
```

Partimos de un fichero con la información de compras realizadas por aproximadamente 2,3 millones de clientes durante las temporadas Pri/Ver y Oto/Inv de los años 2016 y 2017. En el fichero se encuentran identificados 127 campos, perteneciendo aproximadamente 30 campos a cada temporada, además de los campos que son inherentes a cada cliente y se informan por tanto una única vez durante la

permanencia del mismo en la base de datos de la empresa (p. ej. Edad, Antigüedad, etc…).

A pesar de no tener previsto trabajar con toda la información del fichero, efectuamos una carga inicial de toda la información al completo para poder proceder a su posterior análisis. Para ello nos dirijimos nuevamente a la consola de R y ejecutaremos lo siguiente:

```
dataset <- read.csv("./data/dataset.csv", header=FALSE, sep=";", fileEncoding = "UTF-8")

names(dataset) <- c('CTA_ID', 'FX_NACIMIENTO', 'FX_ALTA', 'SEXO_ID', 'F_ACTIVIDAD_2',
                    'DECILA_2', 'GASTO_2', 'GASTO_CAB_2', 'GASTO_SRA_2', GASTO_VIP_CAB_2',
                    'GASTO_VIP_SRA_2', 'GASTO_VSD_2', 'GASTO_PROMO_2', 'GASTO_DCTO_2',
                    'NUM_ART_2', 'NUM_ART_CAB_2', 'NUM_ART_SRA_2', 'NUM_ART_PROMO_2',
                    'NUM_ART_VSD_2', 'NUM_ART_VIP_CAB_2', 'NUM_ART_VIP_SRA_2',
                    'NUM_DIST_GFAM_2', 'NUM_DIST_FAM_2', 'NUM_OPER_2', 'NUM_OPER_PROMO_2',
                    'NUM_OPER_VSD_2', 'NUM_OPER_DEV_2', 'ARRIBA_2', 'ABAJO_2', 'SASTRE_2',
                    'BANO_2', 'TICKET_MEDIO_2', 'APO_2', 'PORC_CONTRI_2', 'F_ACTIVIDAD_3',
                    'DECILA_3', 'GASTO_3', 'GASTO_CAB_3', 'GASTO_SRA_3', GASTO_VIP_CAB_3',
                    'GASTO_VIP_SRA_3', 'GASTO_VSD_3', 'GASTO_PROMO_3', 'GASTO_DCTO_3',
                    'NUM_ART_3', 'NUM_ART_CAB_3', 'NUM_ART_SRA_3', 'NUM_ART_PROMO_3',
                    'NUM_ART_VSD_3', 'NUM_ART_VIP_CAB_3', 'NUM_ART_VIP_SRA_3',
                    'NUM_DIST_GFAM_3', 'NUM_DIST_FAM_3', 'NUM_OPER_3', 'NUM_OPER_PROMO_3',
                    'NUM_OPER_VSD_3', 'NUM_OPER_DEV_3', 'ARRIBA_3', 'ABAJO_3', 'SASTRE_3',
                    'BANO_3', 'TICKET_MEDIO_3', 'APO_3', 'PORC_CONTRI_3', 'F_ACTIVIDAD_4',
                    'DECILA_4', 'GASTO_4', 'GASTO_CAB_4', 'GASTO_SRA_4', GASTO_VIP_CAB_4',
                    'GASTO_VIP_SRA_4', 'GASTO_VSD_4', 'GASTO_PROMO_4', 'GASTO_DCTO_4',
                    'NUM_ART_4', 'NUM_ART_CAB_4', 'NUM_ART_SRA_4', 'NUM_ART_PROMO_4',
                    'NUM_ART_VSD_4', 'NUM_ART_VIP_CAB_4', 'NUM_ART_VIP_SRA_4',
                    'NUM_DIST_GFAM_4', 'NUM_DIST_FAM_4', 'NUM_OPER_4', 'NUM_OPER_PROMO_4',
                    'NUM_OPER_VSD_4', 'NUM_OPER_DEV_4', 'ARRIBA_4', 'ABAJO_4', 'SASTRE_4',
                    'BANO_4', 'TICKET_MEDIO_4', 'APO_4', 'PORC_CONTRI_4', 'F_ACTIVIDAD_5',
                    'DECILA_5', 'GASTO_5', 'GASTO_CAB_5', 'GASTO_SRA_5', GASTO_VIP_CAB_5',
                    'GASTO_VIP_SRA_5', 'GASTO_VSD_5', 'GASTO_PROMO_5', 'GASTO_DCTO_5',
                    'NUM_ART_5', 'NUM_ART_CAB_5', 'NUM_ART_SRA_5', 'NUM_ART_PROMO_5',
                    'NUM_ART_VSD_5', 'NUM_ART_VIP_CAB_5', 'NUM_ART_VIP_SRA_5',
                    'NUM_DIST_GFAM_5', 'NUM_DIST_FAM_5', 'NUM_OPER_5', 'NUM_OPER_PROMO_5',
                    'NUM_OPER_VSD_5', 'NUM_OPER_DEV_5', 'ARRIBA_5', 'ABAJO_5', 'SASTRE_5',
                    'BANO_5', 'TICKET_MEDIO_5', 'APO_5', 'PORC_CONTRI_5', 'TRAMO_EDAD',
```

```
                    'ANYO_ANTIG', 'ABANDONA')

#---------------------------------------------------------------------------
# Definición de variable continua o discreta sobre el dataset inicial
# Creamos para ello un vector donde: 0(Factor) - 1(num)
#---------------------------------------------------------------------------
type <- c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
          1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0)

# Asignamos a cada variable su tipo correcto
for (i in 1:dim(dataset)[2]){
  if (type[i]==1){dataset[,i]<- as.numeric(dataset[,i])}
  else {dataset[,i]<- as.factor(dataset[,i])}
}
```

Tendríamos por lo tanto el fichero ya cargado en R con los nombres de campo y tipos correspondientes de forma correcta. Podemos por tanto analizar la distribución de nuestra variable objetivo, de modo que veamos en qué proporción se encuentran los clientes abandonistas en el dataset que hemos preparado.

```
> table(dataset$ABANDONA)/length(dataset$ABANDONA) # Tabla de proporciones
        0         1
0.8075468 0.1924532
```

Analicemos también algunas de las variables para entender si pueden presentar algún tipo de relación con nuestra variable objetivo ABANDONA. Para ello construiremos varias tablas de frecuencia.

```
# Proporción con respecto al total por columna

> prop.table(table(dataset$TRAMO_EDAD, dataset$ABANDONA), 2)*100
                    0         1
  18-24     1.818972  2.808784
```

```
  25-29     5.528474  6.679087
  30-34     8.808044  9.374371
  35-39    11.965463 12.089326
  40-44    14.388784 14.247181
  45-49    12.867945 12.870712
  50-54    11.782963 11.667206
  55-59    10.369781 10.034118
  60-64     8.138444  7.625296
  65-69     6.060673  5.542279
  70 y más  6.742990  5.954225
  Errores   1.527467  1.107416

> prop.table(table(dataset$SEXO_ID, dataset$ABANDONA), 2)*100
            0         1
  1 30.522075 30.937776
  2 65.063692 66.178828
  3  4.414233  2.883396

> prop.table(table(dataset$DECILA_2, dataset$ABANDONA), 2)*100
             0         1
  -1 12.966820  0.000000
  0  63.306092  0.000000
  1   3.731485  4.300562
  2   3.167333  6.666878
  3   2.860525  7.952004
  4   2.562177  9.203443
  5   2.258332 10.480882
  6   2.119853 11.059235
  7   1.893923 12.007931
  8   1.790900 12.440000
  9   1.710399 12.779595
  10  1.632161 13.109469

> prop.table(table(dataset$F_ACTIVIDAD_2, dataset$ABANDONA), 2)*100
           0        1
  0 82.62751 46.78910
  1 17.37249 53.21090
```

Viendo los resultados obtenidos, podríamos intuir que la variable DECILA_2, puede ser determinante cuando veamos las reglas obtenidas en el modelo. Existe una clara correlación con la variable objetivo, ya que según incrementa el valor de DECILA_2 el porcentaje de abandonistas va volviéndose muy superior con respecto al de no abandonistas. No parece existir esta relación por ejemplo en el caso de la variable SEXO, distribuyéndose de forma muy similar tanto en el caso de los abandonistas como los que permanecen en el programa de fidelización.

Vemos conveniente entender si podríamos considerar un fichero con menor cantidad de variables y registros, de modo que pudiésemos continuar con un desarrollo menos exigente en capacidades de procesamiento, prescindiendo por ejemplo de variables que puedan encontrarse altamente correlacionadas con el resto. Es importante indicar que en el caso del algoritmo que utilizaremos (ver Anexo) no es un requisito imprescindible eliminar variables correlacionadas, tratándose por tanto más bien de un ejercicio para simplificar el número de variables de cara al desarrollo del modelo, basándonos en el criterio de correlación para ello.

Nos dirigimos nuevamente a la consola de R y procederemos a la ejecución del siguiente código:

```
# Preparamos un dataset con información de las temporadas 2 y 3

# Y otro con la información de las temporadas 2 y 4

var_selection_t2t3 <- c(FALSE, FALSE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, TRUE, TRUE, TRUE)


dataset_t2t3 <- dataset[,var_selection_t2t3]


var_selection_t2t4 <- c(FALSE, FALSE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, TRUE,
TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE,
```

```
FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
FALSE, FALSE, FALSE, TRUE, TRUE, TRUE)


dataset_t2t4 <- dataset[,var_selection_t2t4]
```

Puesto que disponemos de la población total, siguiendo la lógica de simplificación, podemos permitirnos reducir el tamaño de los dataset, realizando para ello un muestreo aleatorio sobre cada uno de los anteriores dataset por un valor del 25% sobre el total de los datos iniciales.

Para que el análisis sea además replicable, utilizaremos semillas fijas a la hora de realizar el muestreo. Utilizaremos para ello el siguiente código R.

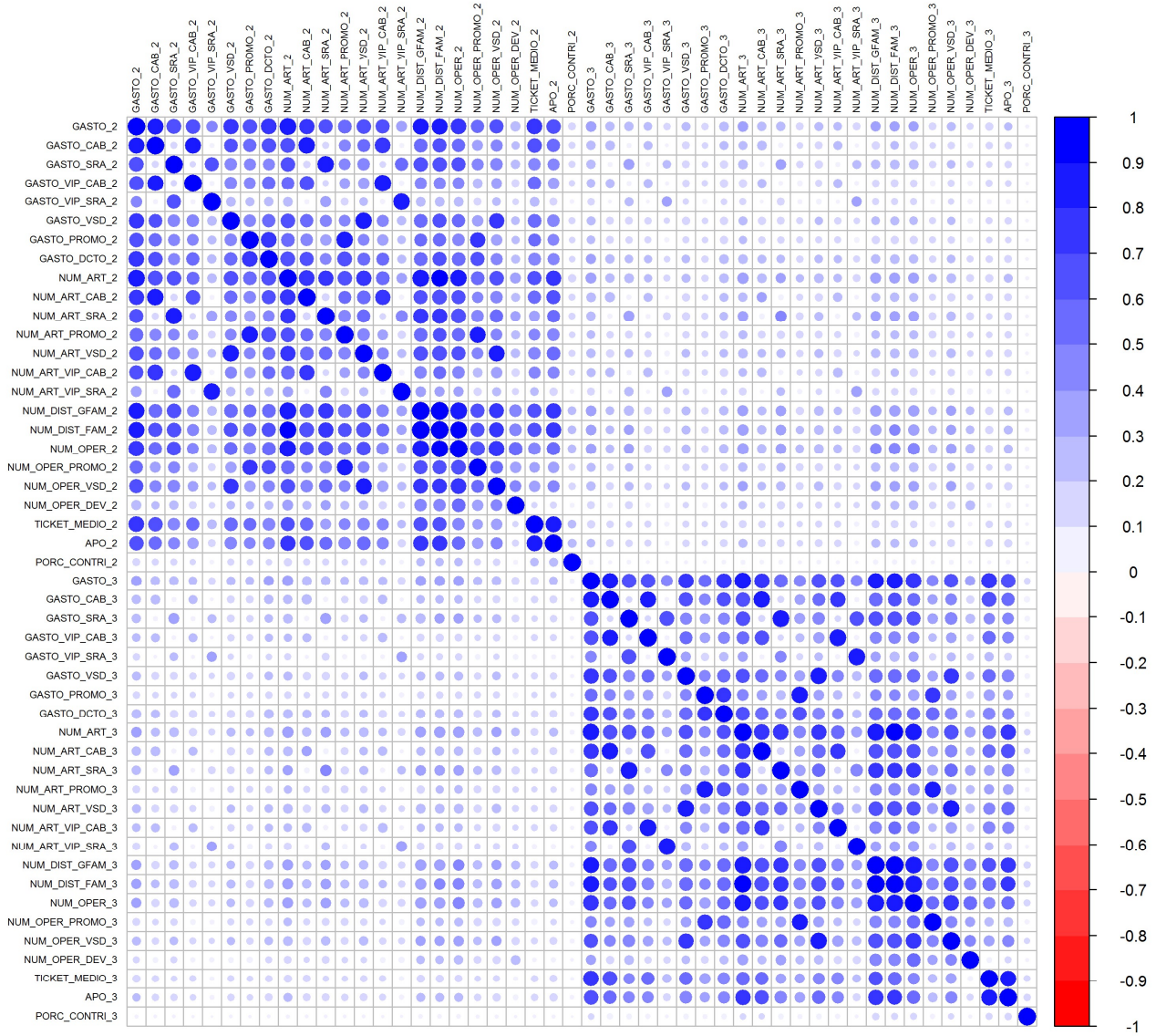
```
# Fijamos primero una semilla de modo que el análisis sea replicable
set.seed(1234)

# Procedemos primero a la creación de la muestra sobre las temporadas 2 y 3
dataset_t2t3_sample <- sample_frac(dataset_t2t3, .25)

# Procedemos por último a la creación de la muestra sobre las temporadas 2 y 4
dataset_t2t4_sample <- sample_frac(dataset_t2t4, .25)
```

En el siguiente paso, procederemos a la construcción de la matriz de correlación sobre el dataset correspondiente a las temporadas 2 y 3, de modo que podamos entender cómo parecen relacionarse las variables de ambas temporadas entre sí. Para ello deberemos introducir la siguiente secuencia de comandos.

```
vector_corr <- c(0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, 0, 0, 0, 1, 1, 1, 0, 0, 0)


vector_corr <- ifelse(vector_corr == 1, TRUE, FALSE)


col <- colorRampPalette(c("red", "white", "blue"))(20)


corrplot(cor(dataset_t2t3_sample[,vector_corr]), order="original", col=col, tl.cex=.5,
tl.col="black")
```

En la siguiente ilustración podemos ver lo que obtenemos como salida del paso anterior (ver Ilustración 4). Es interesante ver cómo parece que efectivamente hay altas correlaciones entre las variables dentro de cada temporada de forma independiente y sin embargo hay muy poca o ninguna si comparamos las mismas variables pero distinta temporada.

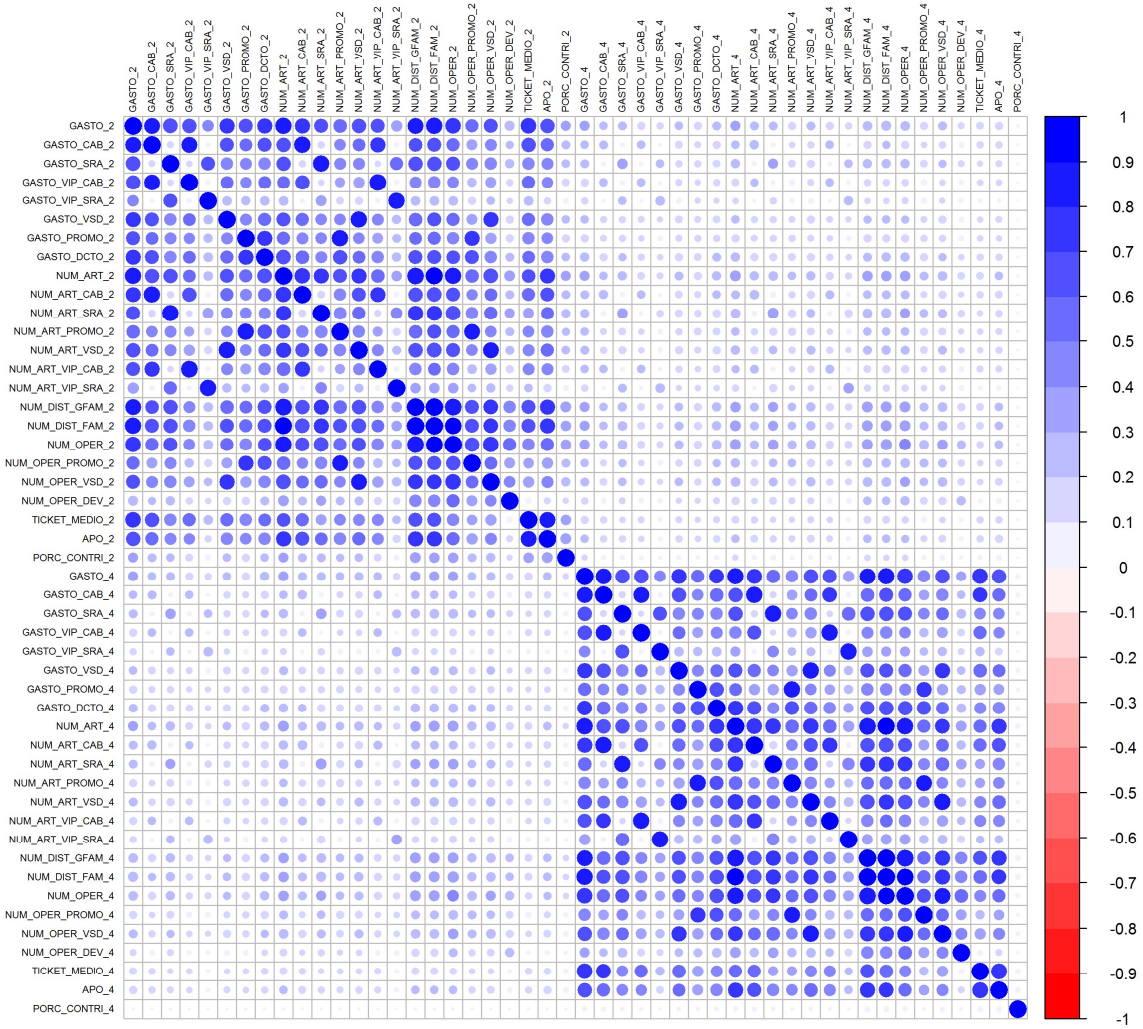
Ilustración 5. Matriz de correlación para las temporadas 2 y 3



Viendo los resultados obtenidos, podríamos concluir que el comportamiento seguido por los clientes no presenta ninguna relación entre temporadas, sin embargo, para corroborarlo, crearemos una nueva matriz de correlación, pero esta vez entre las temporadas 2 y 4 tal como sigue.

```
corrplot(cor(dataset_t2t4_sample[,vector_corr]), order="original", col=col, tl.cex=.5,
tl.col="black")
```

Ilustración 6. Matriz de correlación para las temporadas 2 y 4



A la vista de las correlaciones de los apartados anteriores podemos ratificar la afirmación que planteamos en el párrado anterior, por lo que el comportamiento de los clientes en diferentes temporadas efectivamente no parece presentar ningún tipo de correlación.

No obstante, dentro de la misma temporada sí existen correlaciones muy altas entre diferentes variables. Teniendo en cuenta cómo se construyen las variables, esta correlación dentro de temporada parece bastante lógica, ya que por ejemplo, el GASTO es una combinación lineal producida por la suma de los gastos de los clientes en las diferentes líneas de productos (GASTO_CAB, GASTO_SRA, GASTO_VIP_CAB, GASTO_VIP_SRA, etc…).

De igual forma, vemos que se presentan correlaciones muy elevadas entre las variables relacionadas con el gasto y el número de artículos adquiridos, debido lógicamente a que el GASTO se calcula en función del PxQ, o lo que es lo mismo, en función del COSTE x NUM_ARTICULOS.

Resumiendo, de cara al presente proyecto, adoptaremos como criterio lo siguiente:

1. En caso de correlación superior al 0.85 eliminamos una de las dos variables.
2. Eliminar las variables con la cantidad de artículos comprados, quedándonos en caso de correlación con la variable que represente el gasto.

Crearemos una nueva matriz de correlación, pero esta vez utilizando tan sólo las variables de la temporada 2, ya que a la vista de las matrices anteriores, será suficiente analizar el resultado para una sóla temporada y extrapolar al resto.

Vayamos por tanto a la consola R e introduzcamos el siguiente código.

```
# Seleccionamos únicamente las variables que son numéricas.
t2_numeric <- c("GASTO_2", "GASTO_CAB_2", "GASTO_SRA_2", "GASTO_VIP_CAB_2",
"GASTO_VIP_SRA_2", "GASTO_VSD_2", "GASTO_PROMO_2", "GASTO_DCTO_2", "NUM_ART_2",
"NUM_ART_CAB_2", "NUM_ART_SRA_2", "NUM_ART_PROMO_2","NUM_ART_VSD_2", "NUM_ART_VIP_CAB_2",
"NUM_ART_VIP_SRA_2", "NUM_DIST_GFAM_2", "NUM_DIST_FAM_2", "NUM_OPER_2",
"NUM_OPER_PROMO_2", "NUM_OPER_VSD_2", "NUM_OPER_DEV_2", "TICKET_MEDIO_2", "APO_2",
"PORC_CONTRI_2")


# Ejecutamos la matriz de correlación
matriz_corr <- cor(dataset_t2t3_sample[, t2_numeric], use="pairwise", method="pearson")


# Representamos finalmente la matriz en forma gráfica y lo exportamos a un .PNG
col <- colorRampPalette(c("red", "white", "blue"))(20)
png("./outputs/matriz_corr_t2.png", width=10, height=10, units='in', res=300)
```

```
corrplot(matriz_corr, order="original", col=col, tl.cex=.7, tl.col="black",
         method="number", cl.cex=.7, number.cex=.7, number.digits=2)

dev.off()
```

El detalle con el listado de variables y sus correspondientes correlaciones podemos encontrarlo dentro del apartado Anexos. En este punto, nos limitaremos a indicar cuáles son las correlaciones que cumplen con el criterio que nos habíamos fijado en el párrafo anterior ($R^2 > 0.85$) (ver Tabla 6).

Tabla 6. Resumen de variables con $R^2 > 0.85$

Variable 1	Variable 2	Correlación
GASTO_PDH_SRA_2	NUM_ART_PDH_SRA_2	0,861
GASTO_PROMO_2	NUM_ART_PROMO_2	0,866
GASTO_PDH_CAB_2	NUM_ART_PDH_CAB_2	0,887
GASTO_SRA_2	NUM_ART_SRA_2	0,871
GASTO_VSD_2	NUM_ART_VSD_2	0,862
NUM_DIST_GFAM_2	NUM_OPER_2	0,893
GASTO_CAB_2	NUM_ART_CAB_2	0,880
NUM_DIST_FAM_2	NUM_OPER_2	0,901
NUM_DIST_FAM_2	NUM_DIST_GFAM_2	0,976
NUM_ART_2	NUM_OPER_2	0,852
NUM_ART_2	NUM_DIST_GFAM_2	0,889
NUM_ART_2	NUM_DIST_FAM_2	0,924
GASTO_2	NUM_ART_2	0,884

4.1 Etapa de entrenamiento

Tras el paso anterior, tomamos la decisión de entrenar el árbol con un total de 44 variables, tomando obviamente como variable objetivo la variable que nos indica si el cliente ha sido o no abandonista. Este nuevo dataset contendrá además un total de 574.544 registros, o lo que es lo mismo, datos correspondientes a 574.544 clientes.

Lo primero que tendremos que hacer es determinar el nuevo dataset, para lo cual deberemos indicar en R lo siguiente.

```
# Indicamos las variables que finalmente entran al modelado
dataset_model_t2t3 <- c(TRUE, TRUE, TRUE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
          FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, TRUE, TRUE,
          TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, TRUE, TRUE,
          TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,
          FALSE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
          TRUE, TRUE, TRUE)


# El dataset definitivo será: dataset_model_t2t3
dataset_model_t2t3 <- dataset_t2t3_sample_1[,dataset_model_t2t3]
```

A continuación, deberemos separar el dataset creado en dos muestras, denominadas training y test, de modo que con la primera de ellas realicemos el entrenamiento del modelo y con la segunda podamos evaluar los resultados obtenidos, para lo cual calcularemos la matriz de confusión y la curva ROC. La muestra training por lo general se conforma con el 70% de los datos y la de test con el 30% restante. El objetivo de hacer esto es que el modelo se valide con datos sobre los cuáles no haya sido entrenado, ya que podría darse el caso en que el modelo sobreaprendiese con la muestra de training y no fuera eficaz con ningún otro set de datos, quedando por tanto invalidado el resultado.

Pasamos nuevamente a R para realizar los pasos descritos. En un primer paso utilizaremos la función **sample_frac()** de la librería *dplyr* para obtener un subconjunto de nuestros datos, que conformará la muestra de training, para después generar con la diferencia la muestra de test.

```
# Fijamos una semilla para que el dataset siempre sea el mismo.
set.seed(500)

# Utilizamos simple_frac de la librería dplyr para realizar el muestreo
training_t2t3 <- sample_frac(dataset_model_t2t3, .7)

# Generamos también la muestra test con la diferencia
test_t2t3 <- setdiff(dataset_model_t2t3, training_t2t3)
```

Una vez que hemos creado el dataset para el entrenamiento del modelo, pasamos a la creación del árbol, para lo cual utilizaremos la función *rpart()*, perteneciente a la librería con el mismo nombre. Esta función nos pide una fórmula para especificar la variable objetivo de la clasificación, en nuestro caso ABANDONA, la cual intentaremos predecir usando el resto de variables de nuestro dataset como predictoras.

Es importante indicar que el valor establecido inicialmente para CP (parámetro de complejidad) es muy bajo. Esto lo hacemos para que el algoritmo llegue al mayor número de nodos posible, aun a riesgo de producir overfitting, ya que en un siguiente paso buscaremos cuál es el óptimo de este valor para volver a recalcular el árbol.

```
arbol_t2t3_1 <- rpart(formula=ABANDONA ~ ., data=training_t2t3, method="class", cp=0.0001)
```

4.2 Evaluación del modelo

Pasemos a revisar los resultados obtenidos en este primer entrenamiento del modelo. Para ello tendremos que prestar especial atención al valor de CP que parece ser el óptimo, de modo que podamos realizar un segundo entrenamiento con este nuevo valor, sin que suframos pérdida de precisión (accuracy), este último parámetro será objeto de revisión también. No podemos pensar en conseguir un modelo óptimo con un solo entrenamiento, de hecho esta parte del proceso es la más costosa ya que nos exigirá realizar varias iteraciones, en las cuales iremos realizando ajustes y pruebas sobre los distintos parámetros del algoritmo que estemos utilizando. No obstante, en el caso que aplica, nos conformaremos con realizar el proceso paso por paso de modo que podamos comprender la metodología seguida, no siendo tan crítica la obtención de un buen resultado como el aprendizaje del proceso.

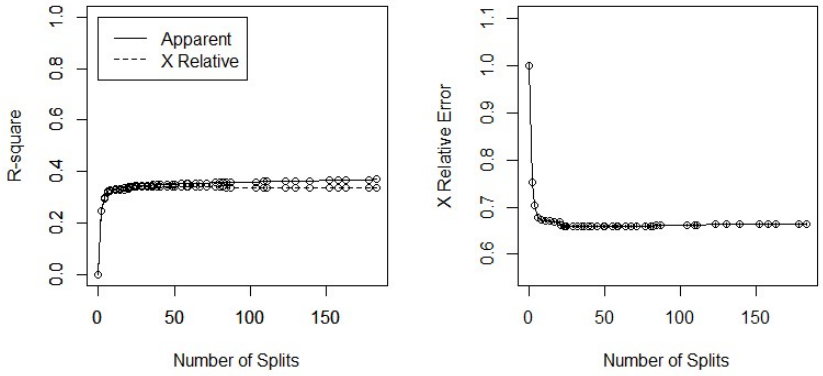
```
# Pasamos a revisar un resumen del árbol creado
> printcp(arbol_t2t3_1)
Classification tree:
rpart(formula = ABANDONA ~ ., data = training_t2t3, method = "class", cp = 0.0001)


Variables actually used in tree construction:
 [1] ABAJO_2          ABAJO_3          ANYO_ANTIG       APO_2            APO_3
 [6] ARRIBA_3         DECILA_2         DECILA_3         F_ACTIVIDAD_2    F_ACTIVIDAD_3
[11] GASTO_CAB_2      GASTO_CAB_3      GASTO_DCTO_2     GASTO_DCTO_3     GASTO_PROMO_2
[16] GASTO_PROMO_3    GASTO_SRA_2      GASTO_SRA_3      GASTO_VIP_CAB_2  GASTO_VIP_CAB_3
[21] GASTO_VIP_SRA_2  GASTO_VSD_2      GASTO_VSD_3      NUM_OPER_2       NUM_OPER_3
[26] NUM_OPER_DEV_2   PORC_CONTRI_2    PORC_CONTRI_3    SASTRE_2         SEXO_ID
[31] TICKET_MEDIO_2   TICKET_MEDIO_3   TRAMO_EDAD

Root node error: 77438/402181 = 0.19255

n= 402181

           CP nsplit rel error  xerror      xstd
1  0.12332447      0   1.00000 1.00000 0.0032291
2  0.02533640      2   0.75335 0.75335 0.0028840
3  0.01300395      4   0.70268 0.70327 0.0028021
4  0.00191766      6   0.67667 0.67879 0.0027604
```

```
5  0.00139897      8   0.67284 0.67379 0.0027518
6  0.00114070     11   0.66864 0.67187 0.0027484
7  0.00110196     14   0.66522 0.67048 0.0027460
8  0.00108904     17   0.66191 0.66883 0.0027431
9  0.00102017     20   0.65864 0.66762 0.0027410
10 0.00054883     21   0.65762 0.66161 0.0027305
11 0.00052946     23   0.65653 0.65995 0.0027275
12 0.00040032     24   0.65600 0.65960 0.0027269
13 0.00039386     25   0.65560 0.65959 0.0027269
14 0.00037449     29   0.65402 0.65952 0.0027268
15 0.00030993     32   0.65290 0.65908 0.0027260
16 0.00029701     35   0.65194 0.65898 0.0027258
17 0.00027979     36   0.65164 0.65895 0.0027258
18 0.00027118     39   0.65080 0.65899 0.0027258
19 0.00026688     41   0.65026 0.65952 0.0027268
20 0.00025827     45   0.64911 0.65934 0.0027264
21 0.00021953     49   0.64808 0.65935 0.0027265
22 0.00021307     50   0.64786 0.65981 0.0027273
23 0.00019370     55   0.64676 0.65973 0.0027271
24 0.00018079     58   0.64618 0.65964 0.0027270
25 0.00016357     59   0.64600 0.65919 0.0027262
26 0.00015927     64   0.64512 0.65961 0.0027269
27 0.00015819     67   0.64464 0.65974 0.0027272
28 0.00015496     71   0.64401 0.66008 0.0027277
29 0.00015066     77   0.64295 0.66009 0.0027278
30 0.00014851     80   0.64250 0.66005 0.0027277
31 0.00014205     82   0.64220 0.66018 0.0027279
32 0.00013774     84   0.64192 0.66048 0.0027285
33 0.00012914     87   0.64151 0.66074 0.0027289
34 0.00012483    104   0.63929 0.66089 0.0027292
35 0.00012268    109   0.63860 0.66176 0.0027307
36 0.00011622    111   0.63836 0.66177 0.0027307
37 0.00011299    123   0.63695 0.66305 0.0027330
38 0.00011106    130   0.63595 0.66305 0.0027330
39 0.00010977    139   0.63487 0.66314 0.0027331
40 0.00010761    152   0.63319 0.66315 0.0027332
41 0.00010589    158   0.63254 0.66301 0.0027329
42 0.00010331    163   0.63202 0.66299 0.0027329
43 0.00010008    178   0.63047 0.66292 0.0027328
44 0.00010000    183   0.62995 0.66333 0.0027335
```

A la vista del resumen obtenido, podemos indicar lo siguiente:

- El árbol ha empleado finalmente 33 de las 44 variables iniciales.
- La frecuencia de la variable que queremos predecir (ABANDONA=1) está en torno a un 19% en el dataset utilizado (training).
- El valor de CP ha alcanzado 44 valores distintos, llegando a 183 nodos diferentes en el último de ellos, que si observamos es el que establecimos por defecto en nuestra sentencia (0.0001)
- El árbol obtenido, presenta un claro caso de sobreaprendizaje. La disminución del XError se estanca alrededor del nsplit=8 (ver Ilustración 6).

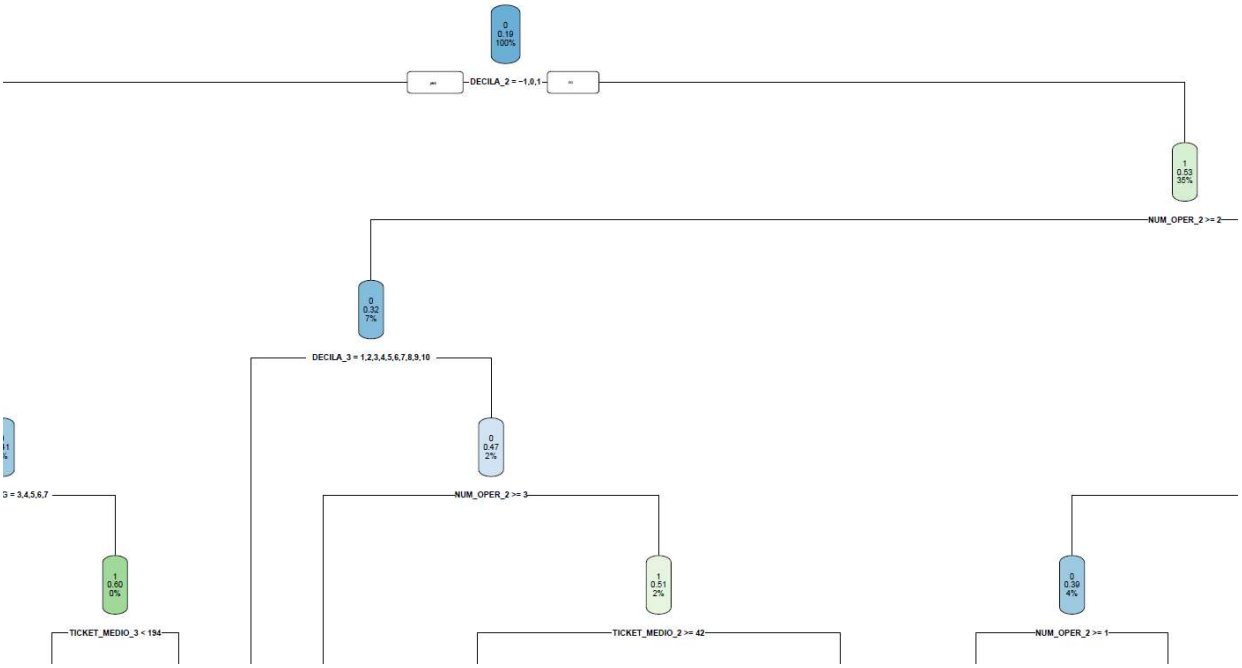
Ilustración 7. Gráficas RSquare y XError (*arbol_t2t3_1*)



Para poder tener una representación gráfica del árbol obtenido, ejecutaremos lo siguiente. Cabe advertir que resultará difícilmente legible debido a la cantidad de nodos resultantes, podemos ver el resultado aun así en la Ilustración 8. No obstante, si hiciésemos zoom sobre ello veríamos algo similar a la Ilustración 7.

```
# Representación gráfica del árbol obtenido
rpart.plot(arbol_t2t3_1)
```

Ilustración 8. Ejemplo de nodos obtenidos con la función *rpart()*

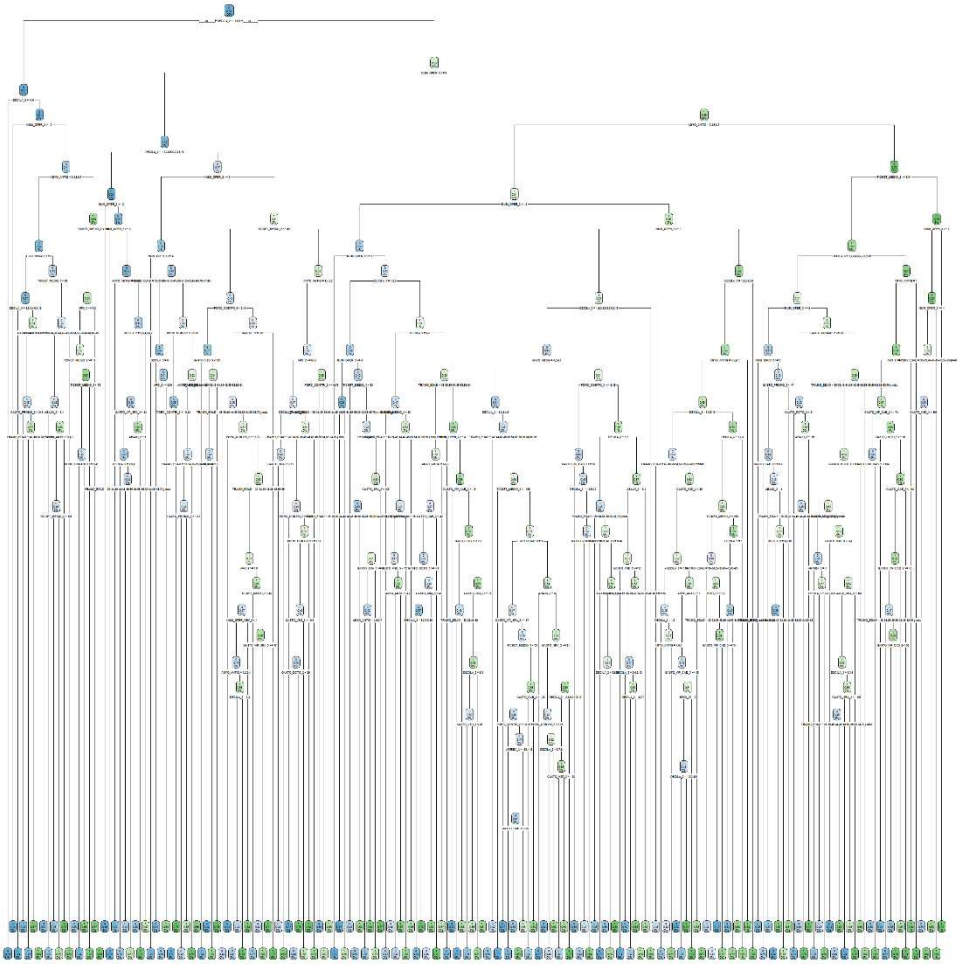


En estos gráficos, cada uno de los rectángulos representa un nodo de nuestro árbol, con su regla de clasificación.

Cada nodo está coloreado de acuerdo a la categoría mayoritaria entre los datos que agrupa. Siendo esa la categoría que ha predicho el modelo para ese grupo. En nuestro caso, podemos ver cómo se diferencia en color verde cuando la categoría agrupada es ABANDONA=1 y en azul cuando se trata de ABANDONA=0.

Dentro del rectángulo de cada nodo se nos muestra qué proporción de casos pertenecen a cada categoría y la proporción del total de datos que han sido agrupados allí. Estas proporciones nos dan una idea de la precisión de nuestro modelo al hacer clasificaciones.

Ilustración 9. Representación gráfica de *arbol_t2t3_1*



Aún queda una pregunta por contestar: ¿qué precisión (accuracy) hemos alcanzado con este primer entrenamiento? Pues bien, si recordamos, a la hora de construir el dataset de entrenamiento dejamos también apartada una muestra que denominamos test, que dijimos además que nos serviría para la validación del resultado obtenido. Es el momento por tanto de dirigirnos a R y ejecutar la evaluación del modelo a partir de dos métodos: matriz de confusión y curva ROC.

```
# Calculamos la capacidad de clasificación sobre la muestra test
> arbol_t2t3_1_predict <- predict(arbol_t2t3_1, test_t2t3, type = "class")

# Representamos la matriz de confusión
> confusionMatrix(data = arbol_t2t3_1_predict, test_t2t3$ABANDONA)

Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 20400  9100
         1 12245 22808

               Accuracy : 0.6693
                 95% CI : (0.6657, 0.673)
    No Information Rate : 0.5057
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3393
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6249
            Specificity : 0.7148
         Pos Pred Value : 0.6915
         Neg Pred Value : 0.6507
             Prevalence : 0.5057
         Detection Rate : 0.3160
   Detection Prevalence : 0.4570
      Balanced Accuracy : 0.6699

       'Positive' Class : 0
>
```

La precisión que obtenemos es de prácticamente un 67%. Si tenemos en cuenta que la proporción de ABANDONA=1 que tenemos en nuestros datos es del 19%, podríamos concluir que estamos mejorando la correcta clasificación del abandono en 3.5 veces, lo cual evidentemente es un dato bastante aceptable. Si atendemos además al índice Kappa (ver Ilustración 9), veremos también que el grado de acierto lo podríamos considerar

"Mediano", lo cual es un dato que también nos hace pensar que el resultado obtenido mejora bastante lo que seríamos capaces de clasificar de forma aleatoria.

Ilustración 10. Grado de acierto según el índice Kappa

Kappa (κ)	Grado de acuerdo
< 0,00	Sin acuerdo
0,00-0,20	Insignificante
0,21-0,40	Mediano
0,41-0,60	Moderado
0,61-0,80	Sustancial
0,81-1,00	Casi perfecto

Otros dos parámetros en los que debemos prestar especial atención son: sensibilidad y especificidad. Pero antes veamos qué significa y qué nos permite medir cada uno de ellos en nuestro caso. Para ello, nos deberemos fijar en la Ilustración 10, donde se representa una matriz de confusión tipo.

Ilustración 11. Matriz de confusión

	REAL.1	REAL.0	Total
PREDIC.1	Verdadero positivo **(a)**	Falso positivo **(b)**	a+b
PREDIC.0	Falso negativo **(c)**	Verdadero negativo **(d)**	c+d
Total	a+c	b+d	a+b+c+d

- **Sensibilidad**: Es el parámetro de validación que define la probabilidad de que un cliente que abandona realmente haya sido marcado en nuestro modelo como que también abandona. Se calcularía de la siguiente manera:

$$S = \frac{a}{a+c}$$

- **Especificidad**: Es el parámetro de validación que define la probabilidad de que un cliente que no abandona realmente, efectivamente haya sido marcado en nuestro modelo como que no abandona. Se calcularía como sigue:

$$E = \frac{d}{b+d}$$

Estos dos parámetros sólo se pueden calcular cuando la variable objetivo de nuestro modelo sea binaria, tal como es el caso que nos aplica. Si vemos los valores de estos parámetros sobre el árbol que hemos generado, podemos ver cómo parece que estamos clasificando algo mejor los casos en los que el cliente finalmente no abandona que los que sí (especificidad>sensibilidad).

Tal como comentamos anteriormente, otra medida que nos permitirá conocer el rendimiento global de nuestro modelo, y que precisamente consiste en la representación gráfica de los parámetros de sensibilidad y especificidad, es la curva ROC. En el caso que nos aplica, podremos calcular esta curva a partir del siguiente código R.

```
# Calculamos la predicción del modelo
predict.rpart <- predict(arbol_t2t3_1,test_t2t3,type = "prob")[,2] #prob. clase=yes
predict.rocr  <- prediction (predict.rpart,test_t2t3$ABANDONA)
perf.rocr     <- performance(predict.rocr,"tpr","fpr") #True y False postivie.rate

# Representación gráfica de la curva ROC
auc <- as.numeric(performance(predict.rocr ,"auc")@y.values)
plot(perf.rocr,type='o', main = paste('Area Bajo la Curva =',round(auc,2)))
abline(a=0, b= 1)
```

Obtenemos como resultado lo que podemos ver en la Ilustración 11. En los modelos con un alto rendimiento, esta curva tendría un crecimiento muy rápido sobre los verdaderos positivos (sensibilidad) y progresivamente se suavizaría a medida que se desplazase por el eje x (especificidad). Para entenderlo, podemos fijarnos en la Ilustración 12, donde se presentan tres tipos de curva en función del rendimiento que conseguiríamos con el modelo en cada caso: bueno, regular y malo.

Ilustración 12. Curva ROC de *arbol_t2t3_1*

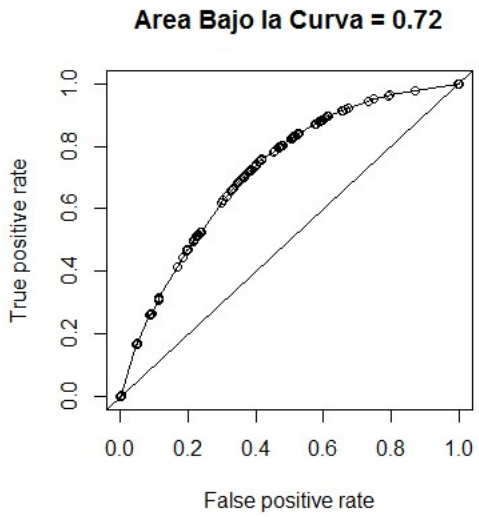
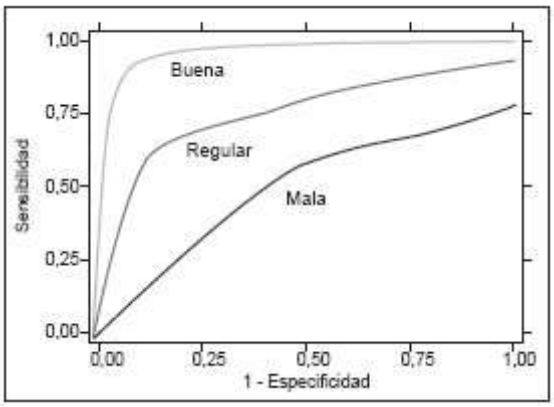


Ilustración 13. Gráfico explicativo para diferentes tipos de curvas ROC



Atendiendo a los resultados presentados hasta el momento, podríamos concluir que el primer entrenamiento del modelo nos ha permitido conseguir unos resultados bastante aceptables en cuanto a la capacidad de clasificación. Ahora bien, es el momento de recuperar el concepto de poda (CP), que como comentamos anteriormente, es en base al parámetro de complejidad, donde podremos reducir el nivel de nodos resultantes sin pérdida de precisión en nuestro modelo. Haciendo uso de R, podremos calcular cuál sería en nuestro caso el CP óptimo gracias al cual estaremos simplificando en gran medida el árbol resultante y por lo tanto nos permitirá una implementación de las reglas que obtengamos bastante más sencilla.

Nos dirigimos por lo tanto nuevamente a la consola de R y ejecutaremos lo siguiente.

```
> cp.choice_ad <- arbol_t2t3_1$cptable[which.min(arbol_t2t3_1$cptable[,"xerror"]),"CP"]
> cp.choice_ad
[1] 0.0002797937
```

El resultado que obtenemos como CP óptimo es 0.0002797937. Si nos fijamos nuevamente en la salida que obtuvimos inicialmente, donde mostrábamos la salida con todos los valores calculados hasta el mínimo fijado para el parámetro CP, estaríamos en una poda que reduciría el número de nodos a 37, una cifra que como ya adelantábamos, dista mucho de los 183 nodos obtenidos con el primer entrenamiento.

Pasemos por tanto a R para entrenar de nuevo el árbol haciendo uso del valor óptimo del CP calculado, utilizaremos la función *prune()* de la librería *rpart* para realizar el podado.

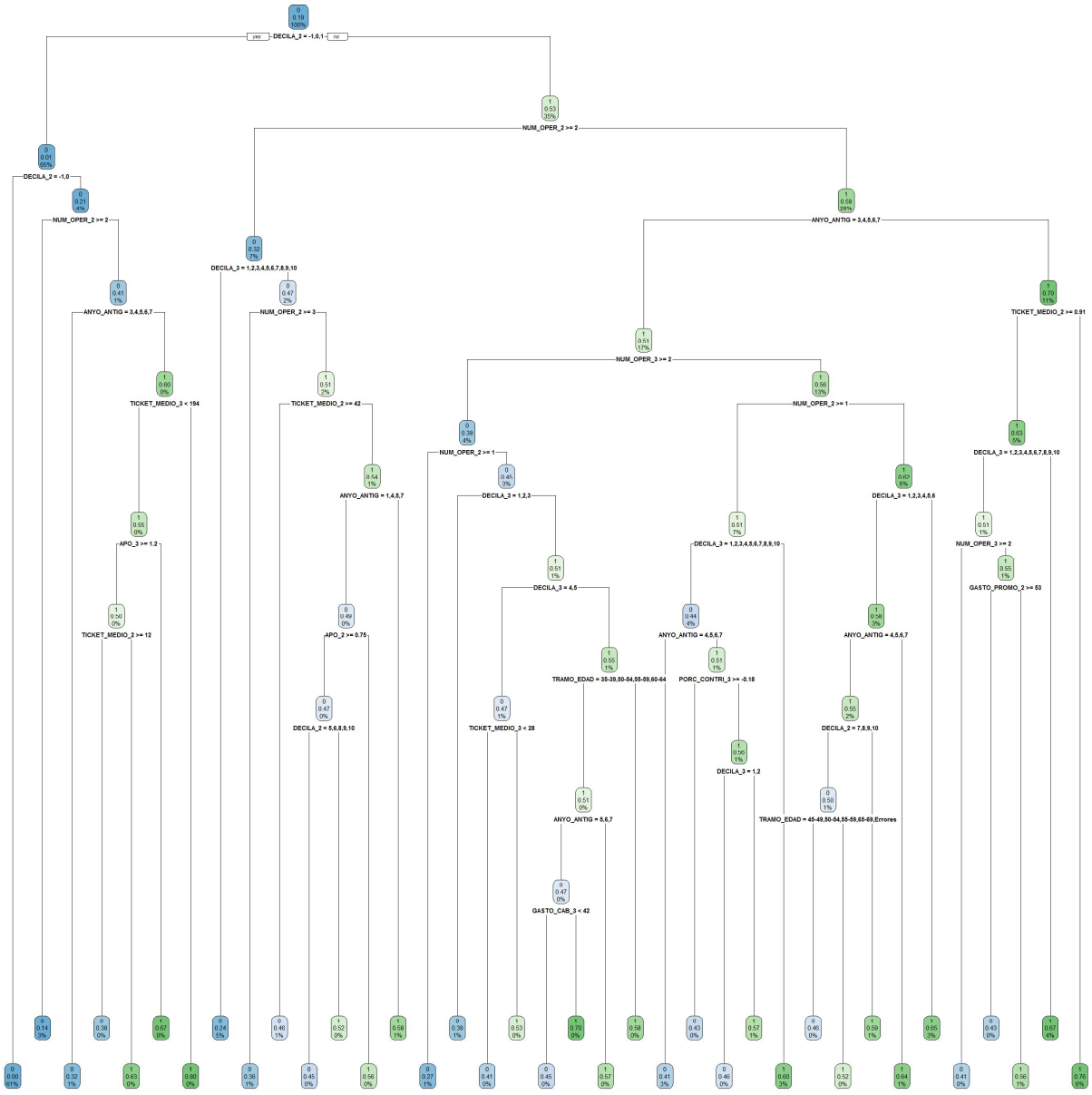
```
# Podamos por el valor óptimo de CP

> arbol_t2t3_2 <- prune(arbol_t2t3_1, cp = cp.choice_ad)
```

Tras la reducción de nodos, podremos pasar a representar gráficamente el árbol obtenido (ver Ilustración 13), y revisar a continuación en detalle cuáles son las reglas que se han generado. Otro punto que deberemos comprobar será que no haya habido pérdida de precisión ya que no debería verse reducida tal como explicamos anteriormente.

Ejecutamos la siguiente sentencia en R, solicitando en este caso además que la salida se lleve a un archivo .PNG con una alta resolución, de modo que podamos ver fácilmente toda la estructura del árbol.

```
> png("./outputs/arbol_t2t3_2.png", width=10, height=10, units='in', res=300)
> rpart.plot(arbol_t2t3_2)
> dev.off()
```

Ilustración 14. Representación gráfica de *arbol_t2t3_2*



Antes de pasar a comentar en detalle las reglas que se han obtenido, comprobemos que efectivamente no ha bajado el rendimiento y precisión del modelo. Volveremos a calcular la matriz de confusión y curva ROC para revisar los principales parámetros.

Nuevamente nos dirigimos a la consola de R e introducimos las siguientes sentencias, que son lógicamente las mismas que utilizamos en la primera iteración del proceso de evaluación del modelo.

```
# Calculamos la predicción sobre la muestra test
> arbol_t2t3_2_predict <- predict(arbol_t2t3_2, test_t2t3, type = "class")

# Representamos la matriz de confusión
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 20473  9256
         1 12172 22652

               Accuracy : 0.6681
                 95% CI : (0.6644, 0.6717)
    No Information Rate : 0.5057
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3367
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6271
            Specificity : 0.7099
         Pos Pred Value : 0.6887
         Neg Pred Value : 0.6505
             Prevalence : 0.5057
         Detection Rate : 0.3172
   Detection Prevalence : 0.4605
      Balanced Accuracy : 0.6685

       'Positive' Class : 0
>
```

Como cabría esperar, la precisión del modelo sigue manteniéndose próxima al 67%, al igual que se mantienen los parámetros sensibilidad, especificidad y kappa. Hemos sido capaces por tanto de reducir notablemente la complejidad del árbol inicial sin penalizar la precisión y rendimiento del modelo, tal como nos habíamos propuesto. Ahora sí pasaremos a revisar las principales reglas obtenidas, que lógicamente podremos visualizar a través de una función en R, dentro de la librería *rpart*. Para ver cuáles son las variables que se han considerado más importantes en la creación del árbol, podemos utilizar la función *summary()*, gracias a la cual obtendremos los siguientes resultados.

```
Call:
rpart(formula = ABANDONA ~ ., data = training_t2t3, method = "class",
    cp = 0.0001)
  n= 402181
            CP nsplit rel error    xerror        xstd
1  0.1233244660      0 1.0000000 1.0000000 0.003229104
2  0.0253363982      2 0.7533511 0.7533511 0.002883971
3  0.0130039515      4 0.7026783 0.7032723 0.002802137
4  0.0019176632      6 0.6766704 0.6787882 0.002760423
5  0.0013989686      8 0.6728350 0.6737906 0.002751764
6  0.0011406975     11 0.6686381 0.6718665 0.002748417
7  0.0011019568     14 0.6652160 0.6704848 0.002746009
8  0.0010890433     17 0.6619102 0.6688318 0.002743123
9  0.0010201710     20 0.6586430 0.6676180 0.002741000
10 0.0005488262     21 0.6576229 0.6616132 0.002730455
11 0.0005294558     23 0.6565252 0.6599473 0.002727516
12 0.0004003203     24 0.6559958 0.6595986 0.002726901
13 0.0003938635     25 0.6555954 0.6595857 0.002726878
14 0.0003744931     29 0.6540200 0.6595212 0.002726764
15 0.0003099254     32 0.6528965 0.6590821 0.002725988
16 0.0002970118     35 0.6519409 0.6589788 0.002725805
17 0.0002797937     36 0.6516439 0.6589530 0.002725760
Variable importance
      DECILA_2     NUM_OPER_3  F_ACTIVIDAD_3 TICKET_MEDIO_3          APO_3
            25             13             13             13             13
 PORC_CONTRI_3     NUM_OPER_2 TICKET_MEDIO_2  F_ACTIVIDAD_2          APO_2
            12              3              1              1              1
 PORC_CONTRI_2       DECILA_3     ANYO_ANTIG NUM_OPER_VSD_2
             1              1              1              1
```

Revisando los resultados obtenidos, podemos comprobar cómo efectivamente se llegó a un total de 36 nodos, siguiendo para ello las reglas determinadas por las principales variables, también listadas en el resultado anterior.

A continuación, tal como adelantábamos en el párrafo anterior, pasamos a mostrar las reglas obtenidas, para ello ejecutaremos la siguiente función en la consola de R.

```
# Creamos la función list.rules.rpart() cuyo autor es Graham J. Williams

list.rules.rpart <- function(model)
{
  if (!inherits(model, "rpart")) stop("Not a legitimate rpart tree")
  #
  # Get some information.
  #
  frm     <- model$frame
  names   <- row.names(frm)
  ylevels <- attr(model, "ylevels")
  ds.size <- model$frame[1,]$n
  #
  # Print each leaf node as a rule.
  #
  for (i in 1:nrow(frm))
  {
    if (frm[i,1] == "<leaf>")
    {
      # The following [,5] is hardwired - needs work!
      cat("\n")
      cat(sprintf(" Rule number: %s ", names[i]))
      cat(sprintf("[yval=%s cover=%d (%.0f%%) prob=%0.2f]\n",
                  ylevels[frm[i,]$yval], frm[i,]$n,
                  round(100*frm[i,]$n/ds.size), frm[i,]$yval2[,5]))
      pth <- path.rpart(model, nodes=as.numeric(names[i]), print.it=FALSE)
      cat(sprintf("   %s\n", unlist(pth)[-1]), sep="")
    }
  }
}

# Ejecutamos finalmente la función sobre el arbol_t2t3_2
```

```
> list.rules.rpart(arbol_t2t3_2)

 Rule number: 4 [yval=0 cover=247130 (61%) prob=0.00]
   DECILA_2=-1,0,1
   DECILA_2=-1,0

 Rule number: 10 [yval=0 cover=11414 (3%) prob=0.14]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2>=1.5
 Rule number: 22 [yval=0 cover=2794 (1%) prob=0.32]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7

 Rule number: 184 [yval=0 cover=369 (0%) prob=0.38]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_3< 194.2
   APO_3>=1.185
   TICKET_MEDIO_2>=12.39

 Rule number: 185 [yval=1 cover=336 (0%) prob=0.63]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_3< 194.2
   APO_3>=1.185
   TICKET_MEDIO_2< 12.39

 Rule number: 93 [yval=1 cover=306 (0%) prob=0.67]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_3< 194.2
```

```
   APO_3< 1.185

 Rule number: 47 [yval=1 cover=250 (0%) prob=0.80]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_3>=194.2

…
```

Debido al elevado número de reglas obtenidas, estas pueden verse en mayor detalle dentro del apartado Anexos.

Lo que nos van indicando cada una de estas reglas, fijándonos en lo más relevante y por lo tanto lo que nos interesa para el presente ejercicio, es lo siguiente:

- yval: Valor de la variable objetivo (0:No abandona ó 1:Abandona).
- (*valor*%): Cantidad de la muestra que entra en la regla.
- prob=*valor*: Probabilidad de ocurrencia del valor objetivo.
- [nombre_variable]=*valor*: Regla que se debe cumplir para llegar al nodo. En el caso de existir más de una se deben cumplir todas ellas.

4.3 Despliegue del modelo

Tras la evaluación del modelo, considerando que hemos conseguido el modelo con mayor rendimiento a la hora de clasificar a los clientes abandonistas, debemos pasar a la etapa de implantación del mismo en los sistemas de la compañía.

El despliegue del modelo dentro de la organización puede abordarse desde distintos enfoques. Generalmente, se lleva a cabo considerando inicialmente los siguientes puntos:

- Comunicación de resultados: tras realizar la evaluación del modelo y selección del "champion" o modelo ganador, se comparte el resultado de la misma con las distintas áreas de negocio siguiendo criterios de eficacia operativa y sin perder de vista que será de aplicación a acciones y decisiones. Es muy importante que el área usuaria comprenda los beneficios del modelo y cómo va a ayudarles en su toma de decisiones, así como en la realización de acciones de marketing preventivas sobre los clientes que se clasifican como posibles abandonistas.
- Reporting: el modelo ha de integrarse con esta función y el nexo de unión son las herramientas de inteligencia de negocio existentes. Resulta interesante comprobar que, frecuentemente, se emplea como un punto de referencia para la colaboración y la consulta. De hecho, es fundamental que de cara a la detección temprana de una posible descalibración del modelo, se integre en las herramientas de reporting empresariales un informe donde poder analizar si el modelo está sufriendo una posible pérdida de eficacia y por lo tanto se requiriese su reentreno.
- Integración con aplicaciones: esta fase del despliegue es fundamental para garantizar realmente que todos los usuarios de negocio pueden beneficiarse del conocimiento que aporta el modelo. En el momento que el modelo pasa a integrarse con una aplicación de la organización, puede empezar a notarse su ventaja a nivel operativo.

Al haber elegido un árbol de decisión, la implementación sobre los sistemas de la empresa pasa por la conversión a lenguaje SQL de todas aquellas reglas que han sido identificadas por el modelo. Una vez realizada esa traducción, el proceso pasaría a formar parte de la cadena existente de procesos en los sistemas de la compañía para que con la frecuencia establecida, en este caso por temporada, se proceda a la clasificación de los clientes. De este modo, las áreas de negocio podrán disponer de la información necesaria para establecer colectivos a los que dirigirse y enfocar los esfuerzos de modo que se evite el abandono de los clientes potencialmente identificados gracias al modelo.

5 Conclusiones

En este TFG revisamos la importancia que en la actualidad presenta la identificación de clientes abandonistas para cualquier compañía. Se concluye que efectivamente su detección debe ser un claro objetivo. Apoyándose para ello en la elaboración de modelos que permitan la identificación y/o clasificación de estos clientes, permitiendo incrementar en mayor o menor medida los ingresos.

La metodología utilizada para la etapa de desarrollo del modelo, denominada CRISP-DM, es el estándar que mejor recoge la elaboración de proyectos de data mining. En primer lugar plantea el entendimiento que se debe tener del problema de negocio que se pretende resolver. Este punto es esencial y su ausencia puede hacer fracasar muchos de los proyectos que se ponen en marcha actualmente en las compañías.

En el desarrollo del modelo, para el caso de estudio presentado en este TFG, ha resultado de gran ayuda para la consecución del modelo óptimo, destacar la facilidad de implementación que la librería *rpart* de R aporta en la construcción de los árboles de decisión CART. En particular por la facilidad del cálculo de coeficiente de poda óptimo.

Esta técnica nos permite obtener buenos resultados a pesar de introducir variables explicativas con altos niveles de correlación y outliers. De hecho, durante el desarrollo del modelo, se efectuó un primer entrenamiento sin prescindir de las variables altamente correladas, tras el cual se pudo constatar que tanto las reglas de salida como las variables más relevantes para el modelo, fueron las mismas que en el desarrollo expuesto en el TFG donde sí se eliminan las variables con alta correlación. Sin embargo, como sí se comenta en el trabajo, conseguimos una disminución notable en los tiempos de procesamiento durante los análisis al reducir notablemente el volumen del dataset, por lo que no debemos descartar seguir esta misma estrategia en el desarrollo de modelos similares. Incluso revisando la metodología CRISP-DM, hay un apartado dedicado exclusivamente, dentro de la preparación de los datos, sobre cómo precisamente abordar este tipo de análisis de correlación además de con el mismo objetivo que el nuestro, buscando también los problemas de multicolinealidad que sí son relevantes en otro tipo de algoritmos que pudiésemos usar (p. ej. Regresión Logística).

En la etapa de evaluación del modelo hemos conseguido clasificar al colectivo abandonista con un 67% de precisión. Si tenemos en cuenta que la proporción en la población de partida es de un 19% para este colectivo, podemos concluir que hemos mejorado en 3.5 veces la capacidad de clasificar a los clientes abandonistas con respecto a la realización de una clasificación totalmente aleatoria.

Analizando las correlaciones de las variables utilizadas, hemos podido comprobar que los clientes no presentan un comportamiento similar entre diferentes temporadas, o lo que es lo mismo, no presentan un hábito continuado de compra en el tiempo. Esta conclusión es importante trasladársela a las áreas de marketing de la empresa XX para que implementen planes de acción que inciten a aumentar la frecuencia de compra.

Hemos podido comprobar también que los resultados de este tipo de técnicas, pueden interpretarse sin excesiva dificultad, lo cual evidentemente podemos considerar de gran ayuda a la hora de hacer extensible los resultados obtenidos a equipos de una organización que no dispongan de los conocimientos estadísticos necesarios.

Bibliografía

Breiman L, Friedman J, Stone J, and Olshen R A. *Classification and regressions trees. Taylor & Francis, 1984*.

Ross Quinlan. *Discovery and Use of Decision Trees, 1988*.

Graham J. Williams. *The Essentials of Data Science: Knowledge Discovery Using R (Chapman & Hall/CRC The R Series) 1st Edition. 2017.*

Terry Therneau and Beth Atkinson (2018). *rpart: Recursive Partitioning and Regression Trees*. R package version 4.1-13. https://CRAN.R-project.org/package=rpart

Max Kuhn. Contributions from Jed Wing, Steve Weston, Andre Williams, Chris Keefer, Allan Engelhardt, Tony Cooper, Zachary Mayer, Brenton Kenkel, the R Core Team, Michael Benesty, Reynald Lescarbeau, Andrew Ziem, Luca Scrucca, Yuan Tang, Can Candan and Tyler Hunt. (2018). *caret: Classification and Regression Training*. R package version 6.0-80. https://CRAN.R-project.org/package=caret

Hadley Wickham, Romain François, Lionel Henry and Kirill Müller (2018). *dplyr: A Grammar of Data Manipulation*. R package version 0.7.6. https://CRAN.R-project.org/package=dplyr

Tobias Sing, Oliver Sander, Niko Beerenwinkel, Thomas Lengauer. Package ROCR. *https://cran.r-project.org/web/packages/ROCR/ROCR.pdf*

Taiyun Wei and Viliam Simko (2017). R package *"corrplot": Visualization of a Correlation Matrix* (Version 0.84). Available from https://github.com/taiyun/corrplot

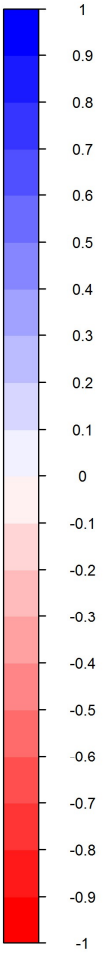
CRISP-DM methodology. *http://crisp-dm.eu/http://crisp-dm.eu/*

The R Project for Statistical Computing. *https://www.r-project.org/*

Anexos

Matriz de correlaciones para Temporada 2

	GASTO_2	GASTO_CAB_2	GASTO_SRA_2	GASTO_VIP_CAB_2	GASTO_VIP_SRA_2	GASTO_VSD_2	GASTO_PROMO_2	GASTO_DCTO_2	NUM_ART_2	NUM_ART_CAB_2	NUM_ART_SRA_2	NUM_ART_PROMO_2	NUM_ART_VSD_2	NUM_ART_VIP_CAB_2	NUM_ART_VIP_SRA_2	NUM_DIST_GFAM_2	NUM_DIST_FAM_2	NUM_OPER_2	NUM_OPER_PROMO_2	NUM_OPER_VSD_2	NUM_OPER_DEV_2	TICKET_MEDIO_2	APO_2	PORC_CONTRI_2
GASTO_2	1	0.82	0.68	0.68	0.41	0.73	0.65	0.76	0.88	0.74	0.61	0.59	0.66	0.63	0.37	0.82	0.83	0.77	0.54	0.61	0.3	0.76	0.66	0.18
GASTO_CAB_2	0.82	1	0.13	0.83	0.06	0.6	0.54	0.67	0.68	0.88	0.14	0.45	0.52	0.76	0.07	0.58	0.61	0.55	0.37	0.45	0.2	0.66	0.54	0.14
GASTO_SRA_2	0.68	0.13	1	0.11	0.62	0.49	0.44	0.46	0.65	0.15	0.87	0.44	0.48	0.12	0.56	0.66	0.66	0.62	0.45	0.48	0.26	0.47	0.44	0.14
GASTO_VIP_CAB_2	0.68	0.83	0.11	1	0.07	0.5	0.43	0.54	0.52	0.67	0.11	0.33	0.4	0.89	0.08	0.44	0.46	0.42	0.28	0.35	0.17	0.54	0.4	0.1
GASTO_VIP_SRA_2	0.41	0.06	0.62	0.07	1	0.29	0.24	0.25	0.28	0.06	0.38	0.17	0.21	0.07	0.86	0.3	0.28	0.28	0.18	0.22	0.12	0.28	0.17	0.06
GASTO_VSD_2	0.73	0.6	0.49	0.5	0.29	1	0.47	0.55	0.64	0.54	0.44	0.43	0.86	0.46	0.27	0.6	0.61	0.55	0.38	0.75	0.2	0.57	0.49	0.13
GASTO_PROMO_2	0.65	0.54	0.44	0.43	0.24	0.47	1	0.76	0.57	0.48	0.39	0.87	0.43	0.38	0.21	0.52	0.54	0.48	0.73	0.38	0.2	0.5	0.43	0.12
GASTO_DCTO_2	0.76	0.67	0.46	0.54	0.25	0.55	0.76	1	0.69	0.62	0.43	0.68	0.51	0.5	0.24	0.59	0.61	0.58	0.6	0.47	0.24	0.5	0.46	0.12
NUM_ART_2	0.88	0.68	0.65	0.52	0.28	0.64	0.57	0.69	1	0.78	0.75	0.65	0.75	0.58	0.33	0.89	0.92	0.85	0.58	0.68	0.35	0.61	0.74	0.19
NUM_ART_CAB_2	0.74	0.88	0.15	0.67	0.06	0.54	0.48	0.62	0.78	1	0.17	0.5	0.59	0.75	0.07	0.64	0.68	0.61	0.41	0.5	0.24	0.55	0.63	0.15
NUM_ART_SRA_2	0.61	0.14	0.87	0.11	0.38	0.44	0.39	0.43	0.75	0.17	1	0.49	0.55	0.12	0.44	0.72	0.74	0.69	0.48	0.54	0.3	0.38	0.51	0.14
NUM_ART_PROMO_2	0.59	0.45	0.44	0.33	0.17	0.43	0.87	0.68	0.65	0.5	0.49	1	0.48	0.34	0.18	0.58	0.6	0.54	0.81	0.43	0.25	0.42	0.49	0.12
NUM_ART_VSD_2	0.66	0.52	0.48	0.4	0.21	0.86	0.43	0.51	0.75	0.59	0.55	0.48	1	0.45	0.24	0.67	0.7	0.63	0.43	0.83	0.25	0.47	0.57	0.14
NUM_ART_VIP_CAB_2	0.63	0.76	0.12	0.89	0.07	0.46	0.38	0.5	0.58	0.75	0.12	0.34	0.45	1	0.09	0.48	0.51	0.47	0.29	0.38	0.2	0.46	0.44	0.11
NUM_ART_VIP_SRA_2	0.37	0.07	0.56	0.08	0.86	0.27	0.21	0.24	0.33	0.07	0.44	0.18	0.24	0.09	1	0.33	0.32	0.32	0.2	0.25	0.14	0.22	0.19	0.06
NUM_DIST_GFAM_2	0.82	0.58	0.66	0.44	0.3	0.6	0.52	0.59	0.89	0.64	0.72	0.58	0.67	0.48	0.33	1	0.98	0.89	0.61	0.7	0.47	0.64	0.73	0.22
NUM_DIST_FAM_2	0.83	0.61	0.66	0.46	0.28	0.61	0.54	0.61	0.92	0.68	0.74	0.6	0.7	0.51	0.32	0.98	1	0.9	0.62	0.71	0.49	0.61	0.72	0.21
NUM_OPER_2	0.77	0.55	0.62	0.42	0.28	0.55	0.48	0.58	0.85	0.61	0.69	0.54	0.63	0.47	0.32	0.89	0.9	1	0.68	0.78	0.57	0.47	0.55	0.21
NUM_OPER_PROMO_2	0.54	0.37	0.45	0.28	0.18	0.38	0.73	0.6	0.58	0.41	0.48	0.81	0.43	0.29	0.2	0.61	0.62	0.68	1	0.52	0.38	0.35	0.38	0.14
NUM_OPER_VSD_2	0.61	0.45	0.48	0.35	0.22	0.75	0.38	0.47	0.68	0.5	0.54	0.43	0.83	0.38	0.25	0.7	0.71	0.78	0.52	1	0.42	0.39	0.45	0.16
NUM_OPER_DEV_2	0.3	0.2	0.26	0.17	0.12	0.2	0.2	0.24	0.35	0.24	0.3	0.25	0.25	0.2	0.14	0.47	0.49	0.57	0.38	0.42	1	0.1	0.13	0.07
TICKET_MEDIO_2	0.76	0.66	0.47	0.54	0.28	0.57	0.5	0.5	0.61	0.55	0.38	0.42	0.47	0.46	0.22	0.64	0.61	0.47	0.35	0.39	0.1	1	0.82	0.21
APO_2	0.66	0.54	0.44	0.4	0.17	0.49	0.43	0.46	0.74	0.63	0.51	0.49	0.57	0.44	0.19	0.73	0.72	0.55	0.38	0.45	0.13	0.82	1	0.23
PORC_CONTRI_2	0.18	0.14	0.14	0.1	0.06	0.13	0.12	0.12	0.19	0.15	0.14	0.12	0.14	0.11	0.06	0.22	0.21	0.21	0.14	0.16	0.07	0.21	0.23	1

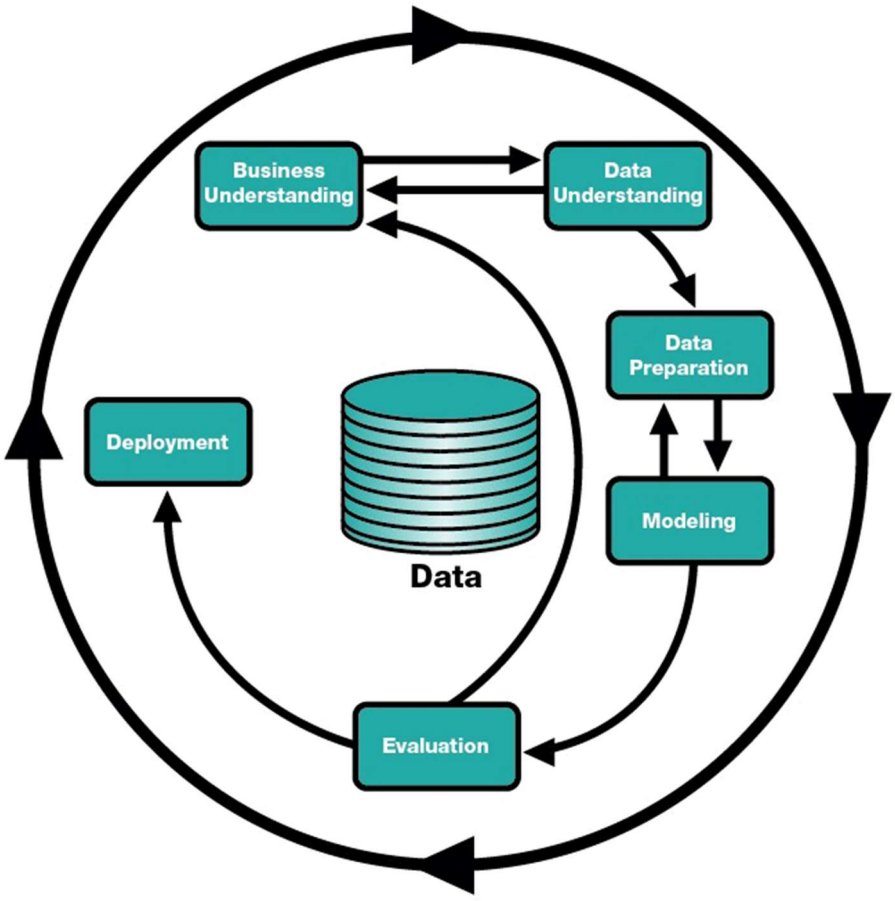


Metodología CRISP-DM

La metodología proporciona una descripción del ciclo de vida del proyecto de minería de datos. Esta contiene las fases de un proyecto, sus tareas respectivas, y las relaciones entre estas tareas. En este nivel de descripción, no es posible identificar todas las relaciones. Las relaciones podrían existir entre cualquier tarea de minería de datos según los objetivos, el contexto, y –lo más importante- el interés del usuario sobre los datos.

El ciclo de vida del proyecto de minería de datos consiste en seis fases, mostrado en la Ilustración 15. La secuencia de las fases no es rígida, pudiéndose avanzar o retroceder de fase durante el desarrollo del proyecto.

Ilustración 15. Fases de la metodología CRISP-DM



El círculo externo en la Ilustración 15 simboliza la naturaleza cíclica de la minería de datos. La minería de datos no se termina una vez que la solución es desplegada. Los insights obtenidos durante el proceso y el modelo finalmente desplegado pueden provocar nuevas, a menudo más - preguntas enfocadas en el negocio. Los procesos de minería subsecuentes se beneficiarán de las experiencias previas.

A continuación hacemos un breve resumen de cada fase:

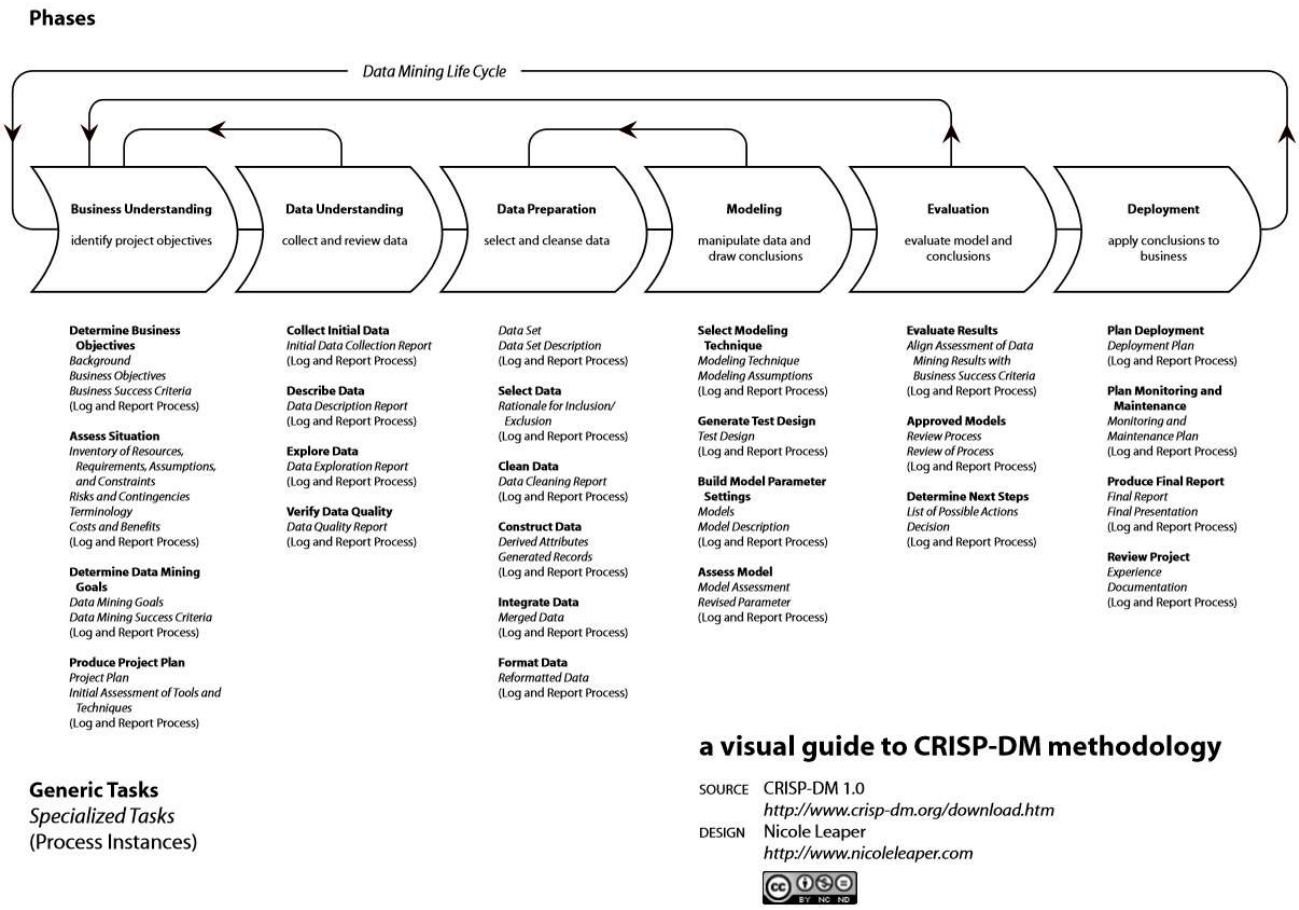
1. **Comprensión del negocio**: Esta fase inicial se enfoca en la comprensión de los objetivos de proyecto y exigencias desde una perspectiva de negocio, para finalmente convertir este conocimiento de los datos en la definición de un problema de minería de datos y en un plan preliminar diseñado para alcanzar los objetivos.
2. **Comprensión de los datos**: La fase de entendimiento de datos comienza con la recolección de datos inicial y continúa con las actividades que nos permiten familiarizarnos primero con los datos, identificar los problemas de calidad de estos, descubrir los primeros insights subyacentes en los datos, y/o descubrir subconjuntos interesantes para formar hipótesis en cuanto a los insights planteados.
3. **Preparación de datos**: Esta fase cubre todas las actividades necesarias para construir el conjunto de datos final a partir de los recolectados en la fase anterior. Las tareas de preparación de datos probablemente van a ser realizadas muchas veces y no en cualquier orden. Las tareas incluyen la selección de tablas, registros, y atributos, así como la transformación y la limpieza de datos.
4. **Modelado**: En esta fase, son seleccionadas varias técnicas de modelado y aplicadas, calibrando sus correspondientes parámetros a valores óptimos, obteniendo diferentes modelos. Típicamente hay varias técnicas para el mismo tipo de problema de minería de datos. Algunas técnicas tienen requerimientos específicos sobre la forma de los datos. Por lo tanto, es habitual volver nuevamente a la fase de preparación de datos.
5. **Evaluación**: En esta etapa del proyecto, procederemos a evaluar los modelos resultantes durante la fase anterior. Antes de proceder al despliegue final del modelo elegido como el mejor, es importante realizar una evaluación exhaustiva,

así como la revisión de los pasos ejecutados para crearlo, para comparar el modelo obtenido con los objetivos marcados por negocio. Un objetivo clave es determinar si hay alguna cuestión importante de negocio que no ha sido suficientemente considerada.

6. **Despliegue**: La creación del modelo no es generalmente el final del proyecto. El resultado obtenido tendrá que ser organizado y presentado en el modo en el que el área de negocio pueda utilizarlo. Dependiendo de los requerimientos, la fase de despliegue puede ser tan simple como la generación de un informe o tan compleja como la ejecución periódica de un modelo en los sistemas de la empresa.

La Ilustración 16, presenta cada una de las principales fases de la metodología acompañadas por sus correspondientes tareas genéricas y sus resultados esperados.

Ilustración 16. Fases y tareas de la metodología CRISP-DM



Reglas obtenidas para el *arbol_t2t3_2*

```
Rule number: 4 [yval=0 cover=247130 (61%) prob=0.00]
   DECILA_2=-1,0,1
   DECILA_2=-1,0

 Rule number: 10 [yval=0 cover=11414 (3%) prob=0.14]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2>=1.5

 Rule number: 22 [yval=0 cover=2794 (1%) prob=0.32]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7

 Rule number: 184 [yval=0 cover=369 (0%) prob=0.38]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_3< 194.2
   APO_3>=1.185
   TICKET_MEDIO_2>=12.39

 Rule number: 185 [yval=1 cover=336 (0%) prob=0.63]
   DECILA_2=-1,0,1
   DECILA_2=1
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_3< 194.2
   APO_3>=1.185
   TICKET_MEDIO_2< 12.39

 Rule number: 93 [yval=1 cover=306 (0%) prob=0.67]
   DECILA_2=-1,0,1
   DECILA_2=1
```

```
  NUM_OPER_2< 1.5
  ANYO_ANTIG=1,2
  TICKET_MEDIO_3< 194.2
  APO_3< 1.185

Rule number: 47 [yval=1 cover=250 (0%) prob=0.80]
  DECILA_2=-1,0,1
  DECILA_2=1
  NUM_OPER_2< 1.5
  ANYO_ANTIG=1,2
  TICKET_MEDIO_3>=194.2

Rule number: 12 [yval=0 cover=18361 (5%) prob=0.24]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2>=1.5
  DECILA_3=1,2,3,4,5,6,7,8,9,10

Rule number: 26 [yval=0 cover=2944 (1%) prob=0.36]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2>=1.5
  DECILA_3=-1,0
  NUM_OPER_2>=2.5

Rule number: 54 [yval=0 cover=2642 (1%) prob=0.46]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2>=1.5
  DECILA_3=-1,0
  NUM_OPER_2< 2.5
  TICKET_MEDIO_2>=42.02

Rule number: 440 [yval=0 cover=1042 (0%) prob=0.45]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2>=1.5
  DECILA_3=-1,0
  NUM_OPER_2< 2.5
  TICKET_MEDIO_2< 42.02
  ANYO_ANTIG=1,4,5,7
  APO_2>=0.75
  DECILA_2=5,6,8,9,10
```

```
Rule number: 441 [yval=1 cover=487 (0%) prob=0.52]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2>=1.5
  DECILA_3=-1,0
  NUM_OPER_2< 2.5
  TICKET_MEDIO_2< 42.02
  ANYO_ANTIG=1,4,5,7
  APO_2>=0.75
  DECILA_2=4,7

Rule number: 221 [yval=1 cover=381 (0%) prob=0.56]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2>=1.5
  DECILA_3=-1,0
  NUM_OPER_2< 2.5
  TICKET_MEDIO_2< 42.02
  ANYO_ANTIG=1,4,5,7
  APO_2< 0.75

Rule number: 111 [yval=1 cover=2405 (1%) prob=0.58]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2>=1.5
  DECILA_3=-1,0
  NUM_OPER_2< 2.5
  TICKET_MEDIO_2< 42.02
  ANYO_ANTIG=2,3,6

Rule number: 56 [yval=0 cover=6013 (1%) prob=0.27]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2< 1.5
  ANYO_ANTIG=3,4,5,6,7
  NUM_OPER_3>=1.5
  NUM_OPER_2>=0.5

Rule number: 114 [yval=0 cover=5354 (1%) prob=0.39]
  DECILA_2=2,3,4,5,6,7,8,9,10
  NUM_OPER_2< 1.5
  ANYO_ANTIG=3,4,5,6,7
  NUM_OPER_3>=1.5
  NUM_OPER_2< 0.5
```

```
   DECILA_3=1,2,3

 Rule number: 460 [yval=0 cover=1479 (0%) prob=0.41]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3>=1.5
   NUM_OPER_2< 0.5
   DECILA_3=4,5,6,7,8,9,10
   DECILA_3=4,5
   TICKET_MEDIO_3< 28.25

 Rule number: 461 [yval=1 cover=1471 (0%) prob=0.53]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3>=1.5
   NUM_OPER_2< 0.5
   DECILA_3=4,5,6,7,8,9,10
   DECILA_3=4,5
   TICKET_MEDIO_3>=28.25

 Rule number: 1848 [yval=0 cover=750 (0%) prob=0.45]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3>=1.5
   NUM_OPER_2< 0.5
   DECILA_3=4,5,6,7,8,9,10
   DECILA_3=6,7,8,9,10
   TRAMO_EDAD=35-39,50-54,55-59,60-64
   ANYO_ANTIG=5,6,7
   GASTO_CAB_3< 41.71

 Rule number: 1849 [yval=1 cover=66 (0%) prob=0.70]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3>=1.5
   NUM_OPER_2< 0.5
```

```
   DECILA_3=4,5,6,7,8,9,10
   DECILA_3=6,7,8,9,10
   TRAMO_EDAD=35-39,50-54,55-59,60-64
   ANYO_ANTIG=5,6,7
   GASTO_CAB_3>=41.71

 Rule number: 925 [yval=1 cover=506 (0%) prob=0.57]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3>=1.5
   NUM_OPER_2< 0.5
   DECILA_3=4,5,6,7,8,9,10
   DECILA_3=6,7,8,9,10
   TRAMO_EDAD=35-39,50-54,55-59,60-64
   ANYO_ANTIG=3,4

 Rule number: 463 [yval=1 cover=1615 (0%) prob=0.58]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3>=1.5
   NUM_OPER_2< 0.5
   DECILA_3=4,5,6,7,8,9,10
   DECILA_3=6,7,8,9,10
   TRAMO_EDAD=18-24,25-29,30-34,40-44,45-49,65-69,70 y más,Errores

 Rule number: 232 [yval=0 cover=11656 (3%) prob=0.41]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3< 1.5
   NUM_OPER_2>=0.5
   DECILA_3=1,2,3,4,5,6,7,8,9,10
   ANYO_ANTIG=4,5,6,7

 Rule number: 466 [yval=0 cover=1407 (0%) prob=0.43]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
```

```
   NUM_OPER_3< 1.5
   NUM_OPER_2>=0.5
   DECILA_3=1,2,3,4,5,6,7,8,9,10
   ANYO_ANTIG=3
   PORC_CONTRI_3>=-0.185

 Rule number: 934 [yval=0 cover=357 (0%) prob=0.46]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3< 1.5
   NUM_OPER_2>=0.5
   DECILA_3=1,2,3,4,5,6,7,8,9,10
   ANYO_ANTIG=3
   PORC_CONTRI_3< -0.185
   DECILA_3=1,2

 Rule number: 935 [yval=1 cover=2206 (1%) prob=0.57]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3< 1.5
   NUM_OPER_2>=0.5
   DECILA_3=1,2,3,4,5,6,7,8,9,10
   ANYO_ANTIG=3
   PORC_CONTRI_3< -0.185
   DECILA_3=3,4,5,6,7,8,9,10

 Rule number: 117 [yval=1 cover=11876 (3%) prob=0.60]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3< 1.5
   NUM_OPER_2>=0.5
   DECILA_3=0

 Rule number: 944 [yval=0 cover=1534 (0%) prob=0.46]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
```

```
   NUM_OPER_3< 1.5
   NUM_OPER_2< 0.5
   DECILA_3=1,2,3,4,5,6
   ANYO_ANTIG=4,5,6,7
   DECILA_2=7,8,9,10
   TRAMO_EDAD=45-49,50-54,55-59,65-69,Errores

 Rule number: 945 [yval=1 cover=1800 (0%) prob=0.52]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3< 1.5
   NUM_OPER_2< 0.5
   DECILA_3=1,2,3,4,5,6
   ANYO_ANTIG=4,5,6,7
   DECILA_2=7,8,9,10
   TRAMO_EDAD=18-24,25-29,30-34,35-39,40-44,60-64,70 y más

 Rule number: 473 [yval=1 cover=5054 (1%) prob=0.59]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3< 1.5
   NUM_OPER_2< 0.5
   DECILA_3=1,2,3,4,5,6
   ANYO_ANTIG=4,5,6,7
   DECILA_2=2,3,4,5,6

 Rule number: 237 [yval=1 cover=3043 (1%) prob=0.64]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3< 1.5
   NUM_OPER_2< 0.5
   DECILA_3=1,2,3,4,5,6
   ANYO_ANTIG=3

 Rule number: 119 [yval=1 cover=12044 (3%) prob=0.65]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
```

```
   ANYO_ANTIG=3,4,5,6,7
   NUM_OPER_3< 1.5
   NUM_OPER_2< 0.5
   DECILA_3=0,7,8,9,10

 Rule number: 120 [yval=0 cover=1432 (0%) prob=0.41]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_2>=0.905
   DECILA_3=1,2,3,4,5,6,7,8,9,10
   NUM_OPER_3>=1.5

 Rule number: 242 [yval=0 cover=281 (0%) prob=0.43]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_2>=0.905
   DECILA_3=1,2,3,4,5,6,7,8,9,10
   NUM_OPER_3< 1.5
   GASTO_PROMO_2>=53.09

 Rule number: 243 [yval=1 cover=3186 (1%) prob=0.56]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_2>=0.905
   DECILA_3=1,2,3,4,5,6,7,8,9,10
   NUM_OPER_3< 1.5
   GASTO_PROMO_2< 53.09

 Rule number: 61 [yval=1 cover=15875 (4%) prob=0.67]
   DECILA_2=2,3,4,5,6,7,8,9,10
   NUM_OPER_2< 1.5
   ANYO_ANTIG=1,2
   TICKET_MEDIO_2>=0.905
   DECILA_3=-1,0

 Rule number: 31 [yval=1 cover=22315 (6%) prob=0.76]
   DECILA_2=2,3,4,5,6,7,8,9,10
```

```
NUM_OPER_2< 1.5
ANYO_ANTIG=1,2
TICKET_MEDIO_2< 0.905
```

Procesos SQL para extracción del dataset

```
--SENTENCIA SQL 1
select mes_id, segmento_desc, count(distinct c.cta_id)
from
	cuenta c,
	cta_mes_agre ca,
	segmento se
where
	c.cta_id = ca.cta_id
	and ca.segmento_id = se.segmento_id
	and mes_id in (201302, 201402, 201502, 201602, 201702, 201802)
	and club_id = '6'
	and cta_pais_id = 724
group by 1, 2

--SENTENCIA SQL 2
/*'AB_RE13 y ACT12*/
select
	seg_dec_activo, clasi,
	count(distinct cta_id),
	sum(_venta),
	sum(_contri),
	sum(_uds),
	sum(_dto),
	sum(_op)
from
(
select
	cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op
from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,
sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op
from  f_ventas v, tarjeta t, l_opcion o
where
```

```
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2012-03-01' and '2013-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (select cta_id from cta_mes_agre where mes_id = 201402 and segmento_id
between 15 and 19)
group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'AB_RE13 y ACT12' clasi
from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201302 and club_id = '6'
) seg
on
cuenta.cta_id = seg.cta_id
) a
group by 1,2

UNION ALL

/*'AB_RE14 y ACT13*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
sum(_uds),
sum(_dto),
sum(_op)
from
(
select
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op
from
(
```

```
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri, sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2013-03-01' and '2014-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                          where mes_id = 201502 and segmento_id between 15 and 19 )
group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'AB_RE14 y ACT13' clasi
from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201402 and club_id = '6'
)      seg
on
cuenta.cta_id = seg.cta_id
) a
group by 1,2

UNION ALL

/*'AB_RE15 y ACT14*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
sum(_uds),
```

```
sum(_dto),
sum(_op)
from
(
select
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op

from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,

sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2014-03-01' and '2015-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                          where mes_id = 201602 and segmento_id between 15 and 19 )
group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'AB_RE15 y ACT14' clasi
from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201502 and club_id = '6'
)      seg
on
cuenta.cta_id = seg.cta_id
) a
```

```
group by 1,2

UNION ALL

/*'AB_RE16 y ACT15*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
sum(_uds),
sum(_dto),
sum(_op)
from
(
select
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op

from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,

sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2015-03-01' and '2016-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                           where mes_id = 201702 and segmento_id between 15 and 19 )
```

```
group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'AB_RE16 y ACT15' clasi
from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201602 and club_id = '6'
)      seg
on
cuenta.cta_id = seg.cta_id
) a
group by 1,2

UNION ALL

/*'AB_RE17 y ACT16*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
sum(_uds),
sum(_dto),
sum(_op)
from
(
select
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op

from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,

sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
```

```
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2016-03-01' and '2017-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                          where mes_id = 201802 and segmento_id between 15 and 19 )
group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'AB_RE17 y ACT16' clasi
from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201702 and club_id = '6'
)      seg
on
cuenta.cta_id = seg.cta_id
) a
group by 1,2



SENTENCIA SQL 3

/*'NUEVA CAPTACION13*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
sum(_uds),
sum(_dto),
sum(_op)
from
(
select
```

```
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op

from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,

sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2013-03-01' and '2014-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                          where mes_id = 201402 and segmento_id between 1 and 2 )
Group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'NUEVA CAPTACION 13' clasi

from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201402 and club_id = '6'
)      seg
on
cuenta.cta_id = seg.cta_id
) a
group by 1,2

UNION ALL
```

```
/*'NUEVA CAPTACION14*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
sum(_uds),
sum(_dto),
sum(_op)
from
(
select
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op

from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,

sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2014-03-01' and '2015-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                          where mes_id = 201502 and segmento_id between 1 and 2 )
group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'NUEVA CAPTACION14' clasi
```

```
from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201502 and club_id = '6'
)      seg
on
cuenta.cta_id = seg.cta_id
) a
group by 1,2

UNION ALL

/*'NUEVA CAPTACION 15*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
sum(_uds),
sum(_dto),
sum(_op)
from
(
select
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op

from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,

sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2015-03-01' and '2016-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
```

```
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                          where mes_id = 201602 and segmento_id between 1 and 2 )
Group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'NUEVA CAPTACION 15' clasi

from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201602 and club_id = '6'
)      seg
on
cuenta.cta_id = seg.cta_id
) a
group by 1,2

UNION ALL

/*'NUEVA CAPTACION 16*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
sum(_uds),
sum(_dto),
sum(_op)
from
(
select
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op

from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
```

```
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,

sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2016-03-01' and '2017-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                          where mes_id = 201702 and segmento_id between 1 and 2 )
group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'NUEVA CAPTACION 16' clasi

from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201702 and club_id = '6'
)      seg
on
cuenta.cta_id = seg.cta_id
) a
group by 1,2

UNION ALL

/*'NUEVA CAPTACION 17*/
select
seg_dec_activo, clasi,
count(distinct cta_id),
sum(_venta),
sum(_contri),
```

```
sum(_uds),
sum(_dto),
sum(_op)
from
(
select
cuenta.cta_id, seg_dec_activo, clasi, _venta, _contri, _uds,_dto,_op

from
(
select t.cta_id,
sum(v.f_pvp_sin_imp)  _venta,
(sum(v.f_pvp_sin_imp) - sum(v.f_precio_pvi)) _contri,

sum(v.f_unidades_vendidas)  _uds,
sum(v.f_descu_siniva2 + v.f_descu_siniva1)  _dto,
count(distinct (case when v.cod_ven_dev in ('V') then v.ticket_id else null end)) _op

from  f_ventas v, tarjeta t, l_opcion o
where
v.tarjeta_id = t.tarj_id /*join venta con tarjeta*/
and v.dia_id between '2017-03-01' and '2018-02-28'
and v.flag_patron_venta=  'S'
and cadena_pais_id  = 'A724'
and v.club_id = '6'
and v.tarjeta_id <> '0'
and v.opcion_id = o.opcion_id /*join venta con producto*/
and o.gfam_id not in ('ÑA', 'WA', 'WB', 'WC', 'WD', 'WE', 'WF', 'WG', 'WH', 'WI', 'WJ',
'WK', 'WL', 'WM')
and cta_id in (     select cta_id       from cta_mes_agre

                          where mes_id = 201802 and segmento_id between 1 and 2 )
group by 1
) cuenta left outer join
(
select c.cta_id, seg_dec_activo, 'NUEVA CAPTACION 17' clasi

from cuenta c, cta_mes_agre ca
where
c.cta_id = ca.cta_id and mes_id = 201802 and club_id = '6'
)seg
on
```

```
cuenta.cta_id = seg.cta_id
) a
group by 1,2



Query para extraer los datos socio demográficos de los miembros del club. Se almacena en
un fichero que será procesado por SPSS Modeler
select
cuenta.cta_id,
club_id,
fx_nacimiento,
fx_alta,
sexo_id
from

(
select cta_id,
count(distinct soc_id) num_socios
from socio
where club_id in ('6')
and cadena_pais_id = 'A724'
group by 1
) cuenta

left outer join

(
select
s.cta_id,
club_id,
min (case when sexo_id = 'D' then 3
when sexo_id = 'M' then 2
when sexo_id = 'H' then 1
else 0 end ) sexo_id,
min(fx_nacimiento) fx_nacimiento,
min(fx_alta) fx_alta
from
socio s
where
club_id in ('6')
and cadena_pais_id = 'A724'
```

```
group by 1,2
) socio

on cuenta.cta_id = socio.cta_id



Query para extraer todos los datos relativos al comportamiento de compra:

select *
from dm_cta_temp_0_5
where club_id ='6'
```

Lista de ilustraciones

Lista de tablas