uc3m | Universidad Carlos III de Madrid

Doble Grado en Ingeniería Informática
y Administración de Empresas

2021-2022

Trabajo Fin de Grado

"Uso de técnicas de aprendizaje automático para la detección de errores de transferencia dentro del *call center*"

David Hernández López

Tutores

Miguel Ángel Patricio Guisado

Antonio Berlanga de Jesús

Colmenarejo, julio 2022

Resumen

La presente memoria corresponde al estudio realizado sobre los datos de llamadas realizadas dentro del *call center* o centro de llamadas de la empresa de telecomunicaciones MasMovil.

Debido a que estos centros son cada día más usados por los clientes para hacer consultas, la cantidad de llamadas tanto entrantes como salientes se incrementan cada año en gran medida, por eso los sistemas son cada vez más complejos, lo que se traduce en errores y casos de pérdida de eficiencia. Para arreglar estos errores, las diferentes empresas que ofrecen estos servicios se están centrando en la optimización de los *call centers* mediante el uso de diversas técnicas basadas en aprendizaje automático.

En este trabajo se mostrarán diversos acercamientos tratados por otros autores para lograr estas optimizaciones, así como la explicación de las diversas técnicas usadas en este estudio cuyo objetivo son las de determinar la existencia de errores de transferencia en las llamadas dentro del *call center*.

También se explicará todo el proceso de preparación y limpieza de datos proseguido del uso de técnicas de agrupamiento cuyo objetivo es el de encontrar los segmentos en las diferentes llamadas donde un cliente está hablando con un agente y este le redirige con otro con el que el cliente no necesitaba hablar. Para poder comprender de qué se componen estas agrupaciones se han utilizado técnicas que permiten la explicabilidad de los modelos de agrupación, los cuales han sido los árboles de decisión y las reglas de asociación Apriori.

Tras este estudio se ha podido determinar la existencia de estos errores de transferencia, así como otros errores de redirección que causan problemas de eficiencia, y que, aunque estos no ocurren con mucha frecuencia, su eliminación podría llegar a ser de gran beneficio para la empresa ya que reducirían costes y aumentarían la satisfacción de los clientes.

Conceptos clave: call center, optimización, aprendizaje automático, errores de transferencia, agrupación, explicabilidad.

Índice general

Índice de figuras

Índice de tablas

1 Introducción y objetivos

Este trabajo busca mostrar los procedimientos y resultados obtenidos de la investigación sobre datos reales de llamadas dentro de un *call center*, y poder demostrar los conocimientos adquiridos en diversas áreas de la Ingeniería Informática. El estudio se ha podido llevar a cabo gracias a los datos aportados por la empresa de telecomunicaciones española MasMovil, que todos los años ofertan una serie de becas a los alumnos de la Universidad Carlos III de Madrid.

En este capítulo se explicarán diversos apartados introductorios para la mejor comprensión del trabajo, como los objetivos principales que se persiguen y su importancia, así como el entorno tanto regulador como socioeconómico y la metodología llevada a cabo, para terminar con la estructura que tendrá el documento.

Primeramente, se van a explicar los elementos más importantes mencionados en el título del trabajo, que son el *call center*, el aprendizaje automático y los errores de transferencia.

Actualmente son pocas las empresas que no cuenten con su propio *call center* o centro que permita realizar y recibir llamadas con el objetivo de contactar con los clientes y realizar diversos servicios para estos, pero esto no ha sido siempre así, o al menos no siempre han existido estos sistemas como los conocemos actualmente.

Es complicado establecer un origen preciso a la aparición de esta tecnología, ya que muchas empresas o pequeños negocios han usado la línea telefónica para contactar con sus clientes, pero se cree que fue un pequeño pastelero alemán al primero que se le ocurrió esta forma de marketing a principios del siglo XX, ya que decidió contactar con sus consumidores de forma telefónica, obteniendo un aumento considerable de las ventas y fidelización. [1]

No fue hasta la década de los 60 cuando Ford empezó a usar esta técnica con el objetivo de ahorrar trabajo a sus vendedores mediante el uso del teléfono, aumentando el número de entrevistas con clientes potenciales, así como consiguiendo una reducción considerable de los costes. [2]

Ya en la década de los 70 se incorporaron nuevas tecnologías que permiten una mayor automatización, como el que se puede considerar el primer *ACD* (distribuidor automático de llamadas) por la empresa estadounidense Rockwell [3] o la aparición del primer *IVR* (respuesta de voz interactiva) [2] la cual permite interactuar con una serie de grabaciones de voz y reconocimiento de respuestas simples mejorando las redirecciones y ahorrando

tiempo y dinero al mismo tiempo. Estas técnicas permitieron la mejora de la experiencia del cliente, así como una mayor agilización de las llamadas.

En estas últimas cuatro décadas se han desarrollado diversos sistemas que permiten la recolección de datos en las llamadas, lo cual, sumado al exponencial aumento en el uso de técnicas de inteligencia artificial, las cuales, según un estudio de McKinsey, se está incrementando su uso en un 25% por parte de las empresas [4], puede ayudar a las compañías a mejorar sus sistemas. Entre estas mejoras entra el objeto de este estudio, ya que puede ayudar a la minimización del número de transferencias erróneas, lo cual conllevaría a un decremento de insatisfacción y costes.

Las transferencias se definen como o bien el paso del cliente dentro de una llamada de un agente a otro, o bien, el paso desde una IVR a un agente. Por lo que un error en una transferencia se produce cuando a un cliente se le envía con un agente con el que no quería hablar ya que este es de un departamento, o servicio dentro de este, que no le interesa. Esto ocurre cuando un agente o IVR comete un error y no manda al cliente al destino adecuado.

Una vez se ha hablado previamente de la inteligencia artificial, se va a explicar en qué consiste esta y sus diferencias con respecto al aprendizaje automático o *machine learning*.

La IA es en computación una disciplina que pretende simular la inteligencia humana mediante programas a través de computadoras que sean capaces de sentir, razonar, actuar y adaptarse. Esta se puede aplicar de formas muy distintas, diferenciando entre todas estas técnicas una rama muy importante dentro de la inteligencia artificial llamada aprendizaje automático, la cual consiste de diversos algoritmos que son capaces de aprender por sí mismos gracias a la experiencia y al ser expuestos a diversos datos. Es por ello por lo que estas técnicas son muy útiles para poder encontrar patrones en los datos, permitiendo a los modelos aprender de estos y realizar predicciones en base al conocimiento adquirido.

Dentro del aprendizaje automático se pueden diferenciar tres grupos principales de técnicas las cuales dependen de los datos que se utilicen para el entrenamiento:

1. **Aprendizaje supervisado**: estos algoritmos basan su aprendizaje en un conjunto de datos previamente etiquetados con el objetivo de poder predecir la clase correcta de datos que no estén etiquetados según los patrones y características que compartan los datos entre ellos. Entre estas técnicas también cabe diferenciar ente las que sirven para resolver problemas de *clasificación*, cuya predicción será una

clase entre un conjunto limitado de estas, y las de *regresión* cuya predicción será un número dentro de un conjunto infinito de posibles resultados.

2. **Aprendizaje no supervisado**: este tipo de aprendizaje se usará con datos de entrada que no estén etiquetados de antemano, o lo que es lo mismo, que no se tiene conocimiento previo de a que clase o conjunto de clases pertenece, por lo que estas técnicas buscarán los patrones comunes que permitan la agrupación de las diferentes instancias de los datos de entradas según sus características y estructura.

3. **Aprendizaje por refuerzo**: es un área de aprendizaje automático que basa sus acciones en el objetivo de maximizar las recompensas o premios que puede obtener. Es por ello por lo que el aprendizaje de la máquina se basa en prueba y error gracias a un esquema de premios y castigos, siendo muy útil en problemas en los que se tenga que determinar cuál es el paso siguiente más adecuado para lograr un resultado deseado en el que se desconoce el camino óptimo para lograrlo.

En este trabajo se usarán las expresiones inteligencia artificial y aprendizaje automático de forma indiferenciada.

1.1 Motivación

En el mundo actual en el que nos encontramos, la velocidad y el ahorro de tiempo se han convertido en algo fundamental, y esto se extrapola a la eficiencia y rapidez con la que un cliente es capaz de contactar con un proveedor de servicios en caso de tener algún problema o duda. Una mayor facilidad para el contacto mejorará la relación cliente-empresa, con lo que se podría conseguir una ventaja competitiva que diferencie a la compañía con respecto a otras que no tomen acciones en este aspecto y cuya atención al cliente no sea lo suficientemente buena.

En el mercado actual y con el objetivo de aumentar el alcance y la flexibilidad, se ha sacrificado la interacción “cara a cara” sustituyéndola por un servicio de *call center*. Este sistema también permite una mayor reducción de los costes así como una mayor monitorización de los contactos cliente-operador, ya que además de poder grabar las

llamadas, también se pueden guardar una gran variedad de métricas de servicio como tiempos de llamada o satisfacción de los clientes, las cuales pueden ser usadas por parte de la compañía para aumentar la eficiencia, manteniendo a los operadores siempre activos y con consultas que les sean fáciles de resolver, así como para aumentar la satisfacción de los clientes por un buen trato y por resolver sus problemas rápidamente.

Para poder mejorar este sistema y poder conseguir los beneficios descritos previamente, se busca combinar esta tecnología con modelos de aprendizaje automático, gracias a los recientes avances en este campo, para dar soluciones y soporte a los *call centers* y poder prevenir futuros problemas.

Según los datos de los que se disponga y las necesidades de la empresa, se podrá optar por uno u otro enfoque, los cuales tendrán diferentes características, así como sus propias ventajas y limitaciones.

Entre los posibles enfoques caben destacar aquellos que son eficaces con los datos no supervisados, ya que estos son muy útiles, debido a que con la aparición del *Big data* y la gran saturación de datos que se tienen, es de gran complicación poder etiquetar los datos a mano para poder conseguir un conjunto suficientemente grande de entrenamiento y poder usar técnicas de aprendizaje supervisado, por lo que ser capaz de clasificar los datos de forma automática y rápida sin la necesidad de intervención humana tiene un gran valor para las empresas.

1.2 Objetivos

El principal objetivo de este trabajo será el de analizar los datos sobre llamadas ofrecidos por la empresa de telecomunicaciones MasMovil para poder detectar si existen errores de transferencia, y en caso de existir, determinar cuándo y por qué se han producido.

Existen departamentos enteros dentro de la compañía dedicados a que las redirecciones entre agentes y el resto de los actores implicados dentro de una llamada sean lo mejor posible, pero no se tiene constancia de si actualmente existen estos errores o que los puede estar produciendo y si son realmente evitables.

Cabe destacar que en el conjunto de datos proporcionado no se especifica cuando se ha producido este error en la transferencia, por lo que se tendrán que utilizar diversos enfoques de aprendizaje no supervisado para detectar patrones y poder llevar a cabo diversas agrupaciones.

Una vez definido el objetivo principal, se pueden definir los pasos propuestos para la consecución de dicho objetivo:

- Analizar el funcionamiento de un *call center,* así como las diversas partes de las que se compone (IVR, ACD, YDILO…).
- Identificar los objetivos de la empresa para este proyecto, así como el entorno en el que se encuentra.
- Llevar a cabo un análisis de los datos, así como una limpieza de estos con el objetivo de poder entenderlos y prepararlos para el posterior uso de técnicas de aprendizaje no supervisado.
- Generar diferentes modelos que puedan agrupar las llamadas, intercambiando las diferentes variables para determinar aquellas que sean más adecuadas para el problema.
- Caracterizar dichas agrupaciones para conseguir un mayor entendimiento de los grupos.
- Analizar los resultados obtenidos para determinar si existen errores de transferencia o si es necesario el uso de otros enfoques.
- Establecer limitaciones actuales y posibles líneas de investigación futuras.

1.3 Impacto socioeconómico

El impacto socioeconómico vendrá determinado por los efectos que tenga la eliminación de los errores de transferencia en las llamadas dentro del *call center* del Grupo MasMovil en los clientes y en la misma compañía.

Como se ha introducido previamente, la aparición de nuevos modelos basados en Inteligencia Artificial y la continua mejora y afinamiento de estos permite un mayor impacto socioeconómico, afectando a diversos aspectos como el marketing, segmentación de clientes, reducción de agentes, predicción y solución de comportamientos anómalos, mejora en la calidad…

La investigación de este documento es un ejemplo más de todos los beneficios que se podrían conseguir usando estos modelos. En cuanto a los beneficios sociales, como ya se ha comentado los clientes obtendrían una mejoría en los contactos con la empresa ya que tendrían que estar un menor tiempo en colas y hablando con agentes con los que no quieren hablar, consiguiendo resolver sus dudas o necesidades de forma más rápida, y en

cuanto al impacto económico de la empresa, se podrían reducir los costes de personal en agentes de atención telefónica debido a que al reducir las transferencias erróneas se reducirían las transferencias totales y la necesidad de un mayor número de agentes, así como la reducción de una pérdida potencial de clientes debido a un mal servicio telefónico.

Teniendo en cuenta que en el último año el Grupo MasMovil tuvo algo más de 47 millones de llamadas con clientes y bajo la estimación propuesta por la empresa de que en el 1% de las llamadas se producen errores de transferencia, en un año se habrían producido medio millón de situaciones en las que los clientes se sentirían insatisfechos. Con el objetivo realista de que el sistema sea capaz de detectar al menos un 70% de estas llamadas, ya que muchas se producirán por errores humanos y serán difíciles de prevenir, a un coste estimado de 1€ que le supone a la empresa tener a un agente atendiendo una llamada, las detecciones de este sistema podrían ayudar a decrementar los costes en más de 300.000€ anuales.

Por último, en el apéndice aparece detallado el presupuesto necesario para la elaboración del TFG, ya que este incurre en los costes de la empresa, y por lo tanto también genera un impacto económico en esta.

1.4 Marco regulador

Debido a que este trabajo usa datos reales de clientes de MasMovil, se ha llevado a cabo una investigación previa de las diferentes regulaciones que pueden afectar a la realización de este y se emprendieron procesos previos como la modificación de ciertos datos sensibles, entre los que se encuentra el número de teléfono de los clientes, mediante una función *hash*, con el objetivo de garantizar el derecho a la intimidad, así como garantizar el secreto de las comunicaciones, tal y como se dicta en el artículo 18 de la Constitución Española [5].

Esta medida también busca garantizar las regulaciones actuales en materia de tratamiento de datos, tanto a nivel nacional como europeo, las cuales vienen descritas en la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales [6] y en el Reglamento Europeo de Protección de Datos descrito en el Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016 [7].

Además, en cuanto a las licencias usadas del software para la realización de este trabajo, estas han sido provistas por el Grupo MasMovil o se han utilizado aquellas con licencia Open Source (libre de uso), como la BSD (Berkeley Software Distribution) o como la licencia PSFL (Python Software Foundation License).
Por último, y con el objetivo de garantizar el cumplimiento de las regulaciones en el área de la Inteligencia Artificial, se ha de cumplir lo dictado por la Comisión Europea, la cual el 21 de abril de 2021 creó un nuevo marco regulatorio [8] el cual aplica tanto a los sectores públicos como privados que hagan uso de esta tecnología. Este propone un enfoque basado en el riesgo, con cuatro niveles:

- **Riesgo inaceptable**: aquellos usos de IA que violan derechos fundamentales.
- **Alto riesgo**: aquellos sistemas que pueden crear un impacto adverso en la seguridad de las personas.
- **Riesgo limitado**: aquellos sistemas a los que se le imponen ciertos requisitos de transparencia porque existe un claro riesgo de manipulación.
- **Riesgo mínimo**: Todos los demás sistemas de IA se pueden desarrollar y utilizar sujetos a la legislación existente sin obligaciones legales adicionales

Este trabajo no entraría en ninguna de las categorías determinadas como de alto riesgo ni prohibidas por la Comisión Europea, sino que sería de riesgo mínimo, por lo que se podría continuar con el estudio.

1.5 Metodología y organización

Para la realización de un proyecto de forma eficiente, organizada y con el objetivo de conseguir unos buenos resultados, es fundamental el uso de la metodología adecuada para el problema en cuestión.
Hay una gran cantidad de metodologías para proyectos de análisis, minería y ciencia de datos como la “Metodología fundamental para la Ciencia de Datos” propuesto por el analista de IBM John Rollins [9] o el método SEMMA, pero en este caso se ha decidido usar CRISP-DM, ya que es la metodología más extendida [10] a la hora de trabajar con datos.

Como se puede observar en la *figura 1*, esta metodología consta de seis pasos fundamentales, los cuales aparecerán aplicados a esta investigación en los diferentes capítulos con los que cuenta este documento.

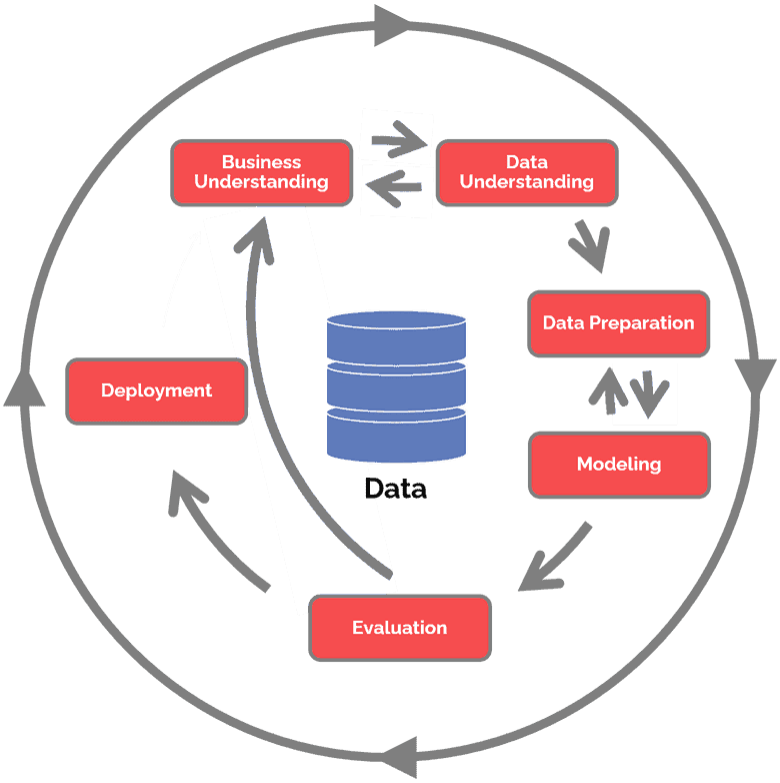

Figura 1. Metodología CRISP-DM [11]

El primer paso consistirá en entender el negocio, cuáles son los objetivos que se pretenden conseguir desde el punto de vista de la empresa y desde el punto de vista de la minería de datos, así como determinar cuáles son las tecnologías y recursos necesarios para la realización del proyecto. Todo esto viene definido entre el primer y parte del segundo capítulo del documento.

Tras este paso se llevará a cabo un proceso de entendimiento y preparación de los datos que está explicado en el tercer apartado. Este proceso consistirá en una descripción de los datos, así como una selección y limpieza de estos.

En el cuarto capítulo se muestran los modelos usados de aprendizaje automático y cuáles han sido los resultados finales obtenidos tras un continuo proceso de evaluación con la ayuda de expertos del Grupo MasMovil. Este proceso consistió en una serie de reuniones periódicas con el objetivo de mostrar los resultados obtenidos mediante el uso de

diferentes técnicas y atributos para poder determinar la validez de los modelos y encontrar los errores de transferencia.

1.6 Estructura

En este apartado, con el objetivo de una mayor comprensión de la estructura y facilidad de lectura del documento, se van a enumerar y explicar brevemente las diferentes partes de las que consta.

Este primer capítulo presenta aquellos apartados que dan una breve introducción y contexto al trabajo, mostrando las motivaciones y principales objetivos que se buscan conseguir mediante esta investigación.

El segundo capítulo, el estado del arte, buscará continuar con la introducción y dar contexto al trabajo exponiendo proyectos en los que diversos autores se enfrentasen a problemas similares, mostrando y explicando las diferentes técnicas y procesos usados.

En los capítulos tres y cuatro se expondrá de manera más específica la metodología usada, mostrando los datos utilizados, así como el preprocesado que ha sido necesario para poder usarlos, y los diferentes modelos y técnicas empleadas para lograr los objetivos del proyecto.

Por último, en el quinto capitulo, se mostrarán las conclusiones, finalizando con diferentes propuestas para investigaciones futuras por si se decidiese seguir indagando en este proyecto.

2 Estado del arte

Para la realización de este trabajo, se ha llevado a cabo un estudio previo sobre productos, desarrollos y trabajos que tienen relación con el campo que se pretende investigar, para que sirvan de referencia y justificación a las decisiones que se tomen a posteriori. Por lo que en este apartado se presentarán de forma resumida aquellos puntos más importantes y relevantes de estos trabajos académicos.

Debido a que esta investigación se centra en una parte muy específica como son los errores en las transferencias entre agentes y no hay estudios muy relevantes entorno a este campo, en este capítulo se realizará una revisión más general de diferentes trabajos que buscan una optimización del *call center,* lo cual ayudará a entender mejor el sistema, así como la explicación de diferentes técnicas de Inteligencia Artificial que puedan ser útiles para la realización de este proyecto.

Por último, se presentarán las tecnologías y el *framework* que se han utilizado para la investigación, mostrando las diferentes alternativas y la justificación de por qué se ha decidido por utilizar estos y no otros sistemas.

2.1 Arquitectura del *call center* y sus elementos

Actualmente, debido al creciente uso de los *call centers*, también han aumentado las diferencias entre estos, ya que las empresas buscan modificar sus los elementos para conseguir una mayor optimización que lleve a un decremento de los costes y un incremento en la calidad de servicio. En este aparado se van a explicar los elementos básicos que tienen casi todos los *call centers*, incluido el de MasMovil.

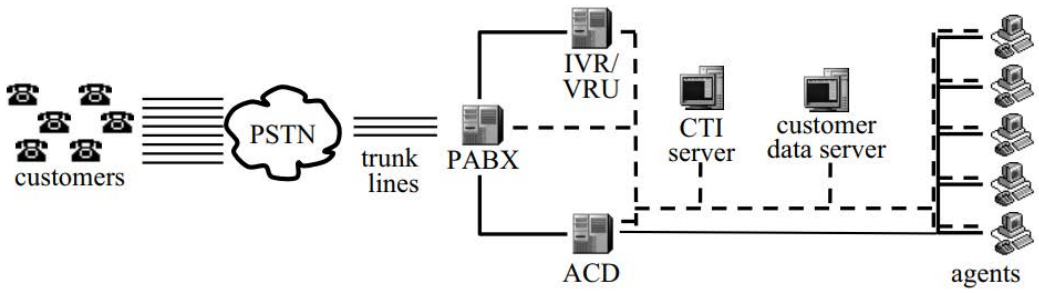

Figura 2. Diagrama esquemático de un Call Center. [12]

Como se puede observar en la *figura 2*, un *call center* se compone de una serie de agentes, ordenadores y equipos de telecomunicaciones que permiten a los clientes acceder a aquellos servicios que necesiten. Estos centros pueden estar físicamente alojados en sitios muy diferentes e incluso en zonas horarias distintas, lo que puede ayudar a estar más tiempo disponibles y a reducir costes.

Es por esto y por la creciente complejidad en la gestión del tráfico de llamadas que es un gran desafío el decidir cuál es el número de agentes óptimo y cuantos tipos de agente debe haber, con sus respectivos conocimientos, para satisfacer una demanda de servicio incierta y variable en el tiempo [13] [14].

Volviendo a la *figura 2*, se van a explicar de forma abreviada las diferentes partes de las que se compone:

- La PSTN (Public switched telephone network) es la red tradicional de teléfono que permite realizar llamadas a larga distancia en tiempo real y de forma fluida. Estas redes han sido fundamentales desde el siglo XIX hasta la actualidad, pero debido a la digitalización están siendo sustituidos por sistemas más recientes como el sistema VoIP que añade los servicios como el *streaming* y el acceso a internet [15].
- La PABX (Private Automatic Branch Exchange) es cualquier central privada telefónica que permita la ramificación de la red primaria pública de teléfonos para la gestión de las llamadas internas, entrantes y salientes.
- La IVR (Interactive Voice Responses), como ya se ha explicado anteriormente en este documento, es un sistema que permite la interacción mediante una serie de grabaciones de voz y reconocimiento de respuestas simples. Esta puede ayudar a los clientes a resolver ciertas cuestiones simples sin tener que recurrir a agentes con la pérdida de tiempo que eso conlleva esperando en una cola. De hecho, se comprobó que hasta un 80% de las cuestiones de los clientes podían ser resueltas por IVRs en la industria banquera [12].

  En caso de que la llamada no pudiese ser resuelta por la IVR y se necesitase hablar con un agente, esta se pasaría al ACD.
- El ACD (Automatic Call Distributor), está diseñado para enrutar las llamadas a agentes individuales, pero como estos normalmente no estarán disponibles inmediatamente, la llamada se mantendrá en espera en el ACD y puesta en una cola.

- El CTI (Computer Telephony Integration) brinda una mayor información y control a los agentes, identificando a los clientes y buscando información en la base de datos acerca de su historial, llamadas previas, etc.

Todos estos pasos con sus respectivas variables se pueden ver resumidos en la *figura 3*. Este trabajo se centrará en identificar aquellas llamadas que tras hablar con un agente vuelven al sistema. Dentro de este grupo se tendrán que diferenciar tres tipos de llamadas: las llamadas que vuelven al ACD o IVR porque tienen otras consultas, las llamadas en las que ha habido un error de transferencia y se les está llevando con el agente con el que debería haber hablado desde el principio, y las llamadas que tienen que pasar por varios agentes porque es un comportamiento normal y especificado.

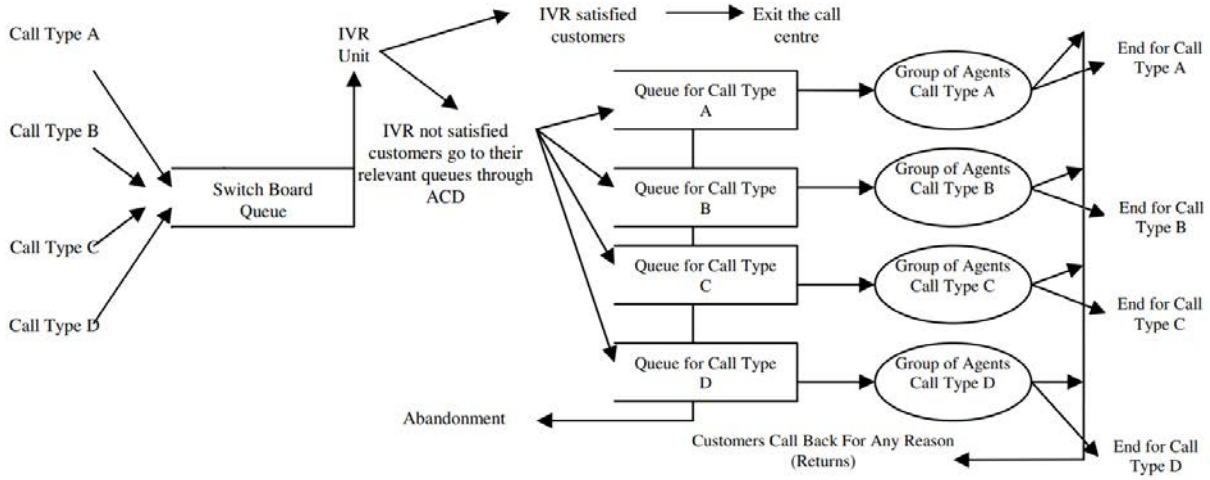

Figura 3. Diagrama del transcurso de una llamada por el call center. [16]

2.2 Trabajos previos enfocados en la optimización del call center

Una vez se ha entendido el funcionamiento básico que hay detrás de un *call center*, se mostrará la investigación previa que se ha realizado para ver como otros autores han enfocado trabajos relacionados con la optimización de diversos problemas en los *call centers*, así como las técnicas y algoritmos utilizados y como sus soluciones pueden ayudar a una mejoría en el sistema de llamadas.

En el artículo, *The Role of Artificial Intelligence on Enhancing Customer Experience* [17], se estudia si es verdad que la inteligencia artificial puede mejorar la experiencia de los usuarios, mediante cuestionarios y entrevistas, determinando que la IA explica un 26,4% de la varianza de la experiencia de los clientes, llegando a la conclusión de que recomiendan el uso de IA, ya que esta permite un acortamiento de los tiempos de espera y ofrece servicios más personalizados que afectan a la experiencia global de los clientes.
Existen muchos artículos relacionados con el uso de Inteligencia Artificial para la optimización de los *call centers*, por lo que tan solo se van a mencionar algunos de los que han parecido más interesantes.
Lo que se pretende conseguir con la IA es poder sustituir al máximo el número de agentes sin que los clientes pierdan satisfacción, buscando conseguir en el futuro que todo el sistema sea llevado por software [18] sin necesidad de acción humana, lo que conllevaría una reducción de costes en salarios muy significativa.
Esto se puede conseguir de muchas formas, como mediante la predicción de cuantos agentes son necesarios en un *call center* en un día y una hora determinada mediante Random Forest [19], mediante la predicción del estrés de los agentes, que puede afectar al desempeño de estos, usando técnicas de Deep Learning sobre pistas de audio [20], o la detección de sentimientos y temática de la llamada usando técnicas de procesamiento de lenguaje natural combinado con CNN (Convolutional Neural Networks) [21]. Estos son solo algunos ejemplos de todo lo que se puede conseguir aplicando las técnicas y modelos apropiados a la enorme cantidad de datos que los *call centers* ofrecen a sus respectivas compañías.
El artículo, *A two stage classification model for call center purchase prediction* [22], busca resolver un problema bastante similar al propuesto en este documento, ya que en sus datos no hay información acerca de los usuarios ni de detalles sobre los productos, y es un problema no supervisado. En este trabajo, mediante un algoritmo de clusterización, en este caso k-medias, se agruparon los productos y una vez se tenían estos clasificados se utilizó el algoritmo SVM (máquinas de vectores de soporte) para clasificar y predecir las compras de los clientes. Gracias a este proceso de agrupación de los productos, se consiguió una mayor información acerca de estos que permitió una mayor precisión a la hora de recomendar los productos a los clientes.
Como se ha podido ver en los diferentes artículos hay una gran multitud de técnicas que pueden servir para la optimización de los *call centers*, ya que actualmente estos son muy complejos y pueden tener una gran cantidad de problemas de eficiencia para poder atender

todas las peticiones que los clientes buscan satisfacer con las llamadas. La propuesta de este proyecto es una más de todos los elementos que pueden ser mejorados pero que no se le está prestando tanta atención entre los diversos artículos que hay publicados sobre optimizaciones. Es por ello que se puede afirmar la gran novedad de esta investigación y el importante impacto que esta puede llegar a tener, ya que como ya se ha explicado durante este documento, la detección y posterior eliminación de los errores de transferencia puede llegar a tener un gran valor e impacto en las diferentes compañías y opiniones de los clientes.

2.3 Algoritmos y modelos

Tras la lectura de este conjunto de artículos y teniendo en cuenta como son los datos prestados por la compañía MasMovil, con sus fortalezas y limitaciones que serán repasadas en el tercer capítulo, se va a explicar la parte técnica, o lo que es lo mismo, que algoritmos y herramientas se van a utilizar para la caracterización de las llamadas y para poder encontrar aquellos agentes en los que se han producido un error de trasferencia.
Al ser este un problema no supervisado, o lo que es lo mismo, que no se tiene conocimiento de cuando se han producido ni si ocurren errores de transferencia, se van a llevar a cabo agrupaciones de los agentes según sus características mediante técnicas de clusterización, y posteriormente, para entender mejor estos conjuntos y por qué el algoritmo ha decidido que un agente pertenezca a una clase o a otra se usarán otras técnicas de inteligencia artificial como los árboles de decisión y las reglas de asociación, que permitan un mayor entendimiento y explicabilidad de los clústeres. Los estudios que avalan el por qué se ha decidido usar estas técnicas se encuentran en el apartado 2.4, pero para poder entender estos mejor, se va a explicar primero en qué consisten estos modelos.

2.3.1 Clustering

Los algoritmos de clusterización son aquellos que agrupan las instancias de un conjunto de datos en diferentes clases según las similitudes de sus atributos de forma que las instancias de un grupo se parezcan más entre ellas a las de los otros clústeres. Es una de las técnicas más utilizadas en aprendizaje no supervisado, cuando los datos no vienen previamente etiquetados.

En caso de que las instancias solo tuviesen un par de atributos o incluso tres, estas diferencias se podrían llegar a ver directamente de forma visual con un gráfico de puntos sin tener que utilizar ningún algoritmo, como se puede observar en la *Figura 4*, pero en esta investigación al tener un gran número de atributos, es mucho más difícil poder ver las diferencias entre clústeres y qué los diferencia.

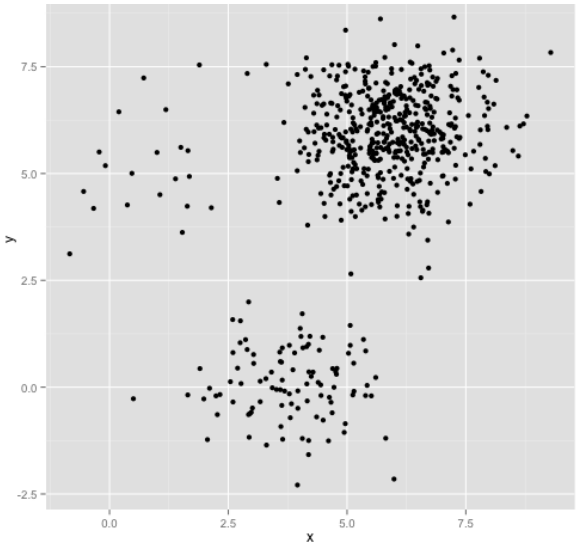

Figura 4. Tres clústeres esféricos en 2D. [23]

Existen un gran número de algoritmos de agrupamiento, cada uno con sus propias ventajas e inconvenientes. Se van a repasar brevemente algunos de los más representativos, como son los algoritmos de k-medias y aquellos basados en densidad.

k-medias

El algoritmo k-medias [24][25] es el algoritmo de clusterización más común y se basa en tres pasos fundamentales:

1. Se eligen el número de clústeres que se asignarán a los datos y se añaden tantos centroides como clústeres haya en el espacio de datos, eligiendo su posición de antemano o de forma aleatoria.
2. A cada instancia se le asigna una clase según el centroide que esté más cerca.
3. Se asigna una nueva posición para los centroides como el punto medio de todas las instancias de su clase (2.1).

$$\arg\min_{S} \sum_{i=1}^{k} \sum_{x_j \in S_i} \|x_j - \mu_i\| \qquad (2.1)$$

4. Se repiten los pasos 2 y 3 hasta que los centroides se mantengan en una posición constante.

Las ventajas del algoritmo son su gran eficiencia y facilidad para determinar los diferentes clústeres, pero tiene dos grandes inconvenientes:

- El primero es la sensibilidad a la posición inicial de los centroides, ya que dependiendo de esta los grupos finales podrán variar. Debido a que no existe un método teórico que determine cual es la posición inicial óptima de los centroides, la mejor práctica será la repetición del algoritmo con diferentes posiciones para conseguir una aproximación de las clases de cada instancia.
- El segundo problema es la dificultad para determinar cuál es el número de clústeres óptimo que tiene que usar el algoritmo para separar los datos. Cuantos más clústeres haya el error será menor, pero también lo será la cantidad de información que proporcione la agrupación. Como se puede apreciar en la *Figura 5*, es de gran dificultad determinar si esos datos deberían ser agrupados por dos o cuatro clústeres, ya que podría depender de los resultados que se esperasen obtener. Aunque no son una solución definitiva, hay ciertos enfoques que permiten tratar este problema, como son el *Elbow Method*, *Convex Clustering Techniques* [26] o *enfoques Bayesianos* [27].

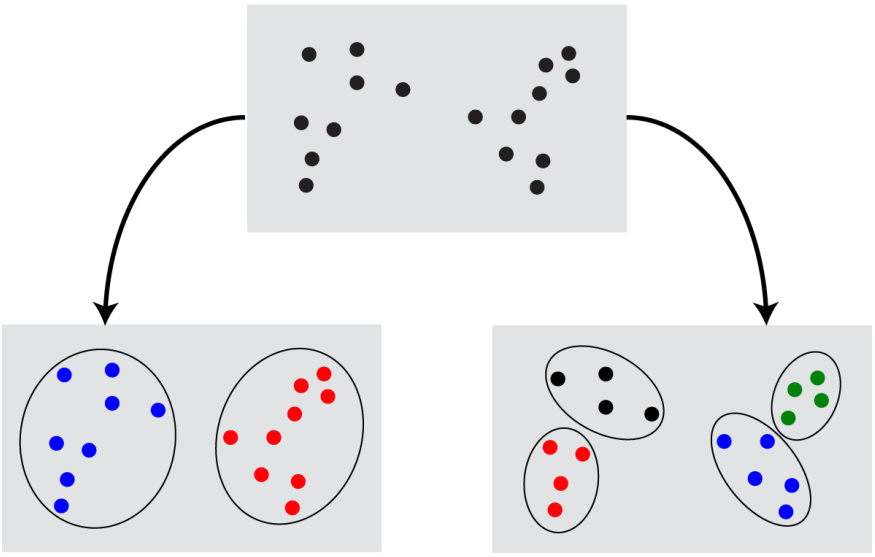

Figura 5. Diferencias en los resultados con dos y cuatro clústeres. [23]

Basados en densidad

Para poder corregir las desventajas del algoritmo k-medias podemos usar otros tipos de clusterización como aquellos basados en densidad, entre los que destacan DBSCAN y OPTICS, que es una ampliación del primero:

- **DBSCAN**: Fue el primer modelo basado en densidad. Mediante este algoritmo no hay que especificarle al modelo cuantas agrupaciones tiene que hacer, sino que este detectará las zonas con concentraciones de puntos y el resto los clasificará como ruido. Para ello se le tendrán que pasar dos datos, el mínimo de puntos que debe haber en la concentración para que se considere un clúster y la distancia máxima del radio del clúster [28].

  El principal problema de este método es que no se pude utilizar con conjuntos con densidades muy diferentes.
- **OPTICS**: Es un algoritmo muy parecido al DBSCAN, pero a diferencia de este, es capaz de detectar grupos significativos en datos de densidad variable [29].

Las principales desventajas de estos métodos son la cantidad de parámetros que se pueden modificar, como las entidades mínimas por clúster o la distancia máxima a considerar, lo cual puede llegar a ser una ventaja para conseguir mejores resultados, pero se tiene que dedicar un tiempo mayor a la modificación de estos parámetros y el análisis de los resultados. También tiene una complejidad computacional cuadrática (DBSCAN) por lo que si se tiene una gran cantidad de datos el tiempo que tardará en ejecutarse será mucho mayor que con el algoritmo k-medias.

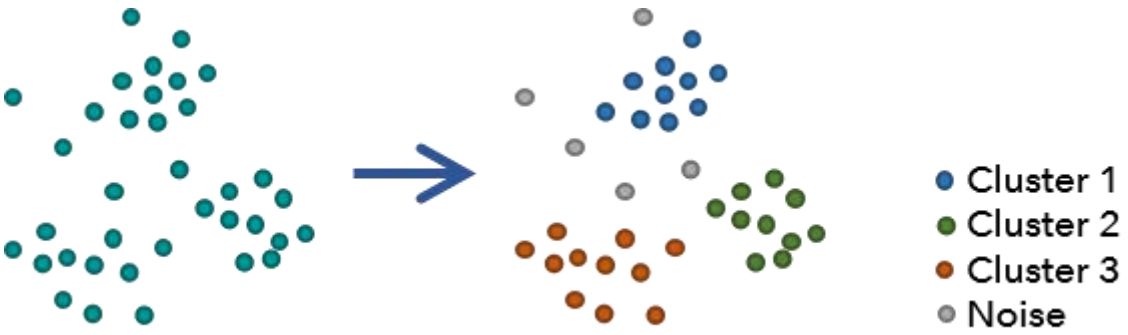

Figura 6. Clúster basado en densidad. [30]

2.3.2 Elbow method

El Elbow Method es una de las herramientas con las que se cuenta para determinar el número óptimo de clústeres para el algoritmo k-medias.

Este método consiste en la ejecución del mismo modelo de k-medias repetidamente cada vez con un número diferente de clústeres, guardando en cada caso los valores de distorsión e inercia.

- **Distorsión**: se calcula como la media de todas las distancias de las instancias con el centriolo de su clase.
- **Inercia**: se calcula como la suma de todas las distancias de las instancias con el centriolo de su clase.

El número óptimo de clústeres se encontrará en el "*elbow*" o codo, que es el momento en el que la distorsión o inercia pasan a decrecer de forma lineal.

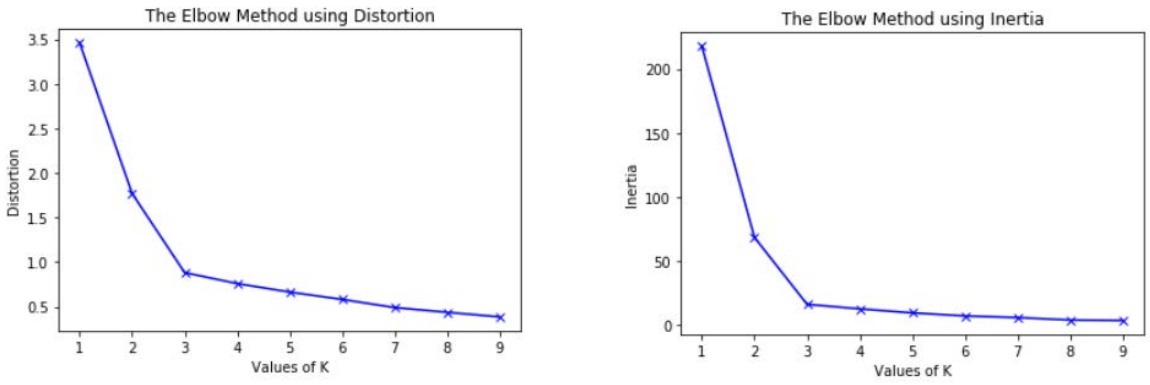

Figura 7. Ejemplo de Elbow Method para una clusterización óptima de tres clusters. [31]

2.3.3 Árboles de decisión (CART)

Los árboles CART (*classification and regression tree*) son uno de los modelos más sencillos del aprendizaje automático y uno de los más utilizados dentro de los árboles de decisión. Estos se basan en árboles de decisión binaria y se van expandiendo os hasta llegar a los nodos hoja, que son los que determinan si los datos de entrada pertenecen a una clase o a otra.

Hay dos funciones que pueden ser utilizadas que determinan la ganancia de información y como se expande el árbol hacia abajo, que son el índice de Gini (2.2) y la entropía (2.3),

los cuales eligen como siguiente nodo a aquel que tenga el menor de uno de estos dos valores [32].

$$Gini = 1 - \sum_{k=1}^{K} p_k^2 \qquad (2.2)$$

$$Entropy = -\sum_{k=1}^{K} p_k \log_2 p_k \qquad (2.3)$$

Debido a que este modelo, a diferencia del clustering, utiliza datos supervisados, será fundamental que no se produzca sobre entrenamiento o lo que es lo mismo, que el modelo no se ajuste demasiado a los valores de entrenamiento y que no sea capaz de clasificar correctamente nuevos datos nunca antes vistos. Para ello, se utiliza la técnica de *pruning* o poda, que consiste en la eliminación de partes del árbol que disminuyan el error al clasificar nuevas instancias del conjunto de validación, haciendo el árbol más pequeño y menos específico.

La gran ventaja de este algoritmo frente a otros modelos como las redes neuronales es la explicabilidad, ya que se puede ver gráficamente por qué se ha clasificado un elemento de una clase o de otra, mientras que la mayoría de los algoritmos de aprendizaje automático, aunque puedan ser más potentes, actúan como "cajas negras", siendo de gran dificultad entender el proceso que han llevado a cabo. Por lo tanto, para poder explicar los clústeres, este modelo es de mucha mayor utilidad.

2.3.4 Reglas de asociación Apriori

Como ya se verá con posterioridad, para poder entender mejor los clústeres y ver las relaciones dentro de estos, además de los árboles de decisión se probarán otras técnicas como las reglas de asociación, en este caso basadas en el algoritmo Apiori.

Apriori fue uno de los primeros algoritmos para la construcción de reglas de asociación, que tienen como objetivo encontrar conjuntos de elementos que aparezcan frecuentemente juntos dentro de un determinado límite y crear reglas.

Las reglas se definen como un conjunto de ítems X e Y disjuntos donde X precede a Y, y estará medida según una serie de valores:

- *Support* o Soporte: indica como de frecuente es la regla dentro del conjunto de todos los datos (2.4).

$$Soporte\ (X \rightarrow Y) = \frac{cont(X \cup Y)}{N} \qquad (2.4)$$

- *Confidence* o Confianza: indica como de frecuente aparece Y entre todas las instancias que contienen X (2.5).

$$Confianza\ (X \rightarrow Y) = \frac{soporte(X \cup Y)}{soporte(X)} \qquad (2.5)$$

- *Lift* o Empuje: indica si la regla existe verdaderamente, ya que compara la frecuencia de la regla con la frecuencia esperada por mero azar, por lo que valores superiores a uno indican que existe un patrón. Cuanto más alto sea el valor, mayor evidencia habrá (2.6).

$$Empuje\ (X \rightarrow Y) = \frac{soporte(X \cup Y)}{soporte(X) * soporte(Y)} \qquad (2.6)$$

Debido a que el número de reglas viene determinado por el número de atributos y que su complejidad es exponencial es exponencial, ya que el número de reglas totales será igual a $3^d - 2^{d+1} + 1$, siendo "d" el número de atributos, Apriori utiliza una heurística de *antimonotonicidad* por la que determina que, si una regla es infrecuente, un conjunto más grande que contenga esa regla también lo será por lo que se puede podar, como se puede observar en la *Figura 8*.

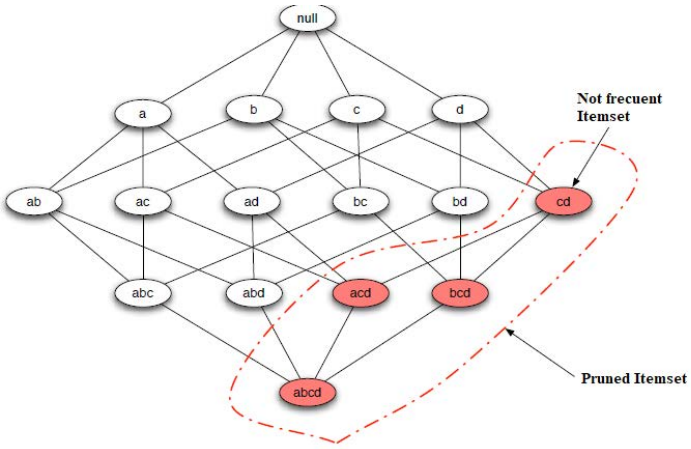

Figura 8. Algoritmo Apriori. [33]

2.4 Caracterización clústeres

Aunque todos los métodos previamente descritos en el apartado 2.2 busquen también una optimización del *call center* como en este proyecto, ninguno se centra en los errores de transferencia ni como caracterizarlos o evitarlos.

Debido a que se está ante un problema no supervisado, se buscará hacer una *clusterización* o agrupamiento de los diferentes segmentos de agente que haya en las diferentes llamadas para encontrar aquel clúster que pueda indicar un error en la transferencia.

En este subapartado se mostrarán diferentes artículos que utilicen técnicas de caracterización de clústeres que se puedan extrapolar a este problema y faciliten la detección de estos errores.

Primeramente, tenemos aquellos trabajos que utilizar un árbol de decisión para la caracterización de las agrupaciones previamente realizadas. Este árbol sirve para ver de forma esquematizada y visual cuales son los atributos que determinan que una instancia pertenezca a una clase u a otra. Esto puede ser muy útil para mostrarlo a personas que no tengan conocimientos en el ámbito de la Inteligencia Artificial, como podría ser el departamento de negocio, para que tomen una decisión u otra basada en los datos mostrados por el árbol.

Como se pude ver en el artículo, *Churn Management in Telecommunications: Hybrid Approach Using Cluster Analysis and Decision Trees* [34], se buscaba determinar por qué los clientes pasan de un servicio de telecomunicaciones a otro usando técnicas de clusterización y árboles de decisión. En la primera etapa se utilizó el algoritmo k-medias para identificar los segmentos donde se producía una mayor tasa de abandono y después

se utilizaron árboles de decisión para encontrar qué atributos son los que determinaban un nivel de abandono más alto o más bajo.

El mismo procedimiento y el uso de estas dos técnicas combinadas los podemos ver en otros muchos más artículos, como *Cluster and CART analyses identify large subgroups of adults with cystic fibrosis at low risk of 10-year death* [35], cuyo objetivo es el de identificar los atributos que conllevan un bajo riesgo de muerte en adultos con fibrosis quística, gracias a una clusterización previamente hecha y utilizando un árbol CART (árbol de regresión y clasificación), o en otros como *Central sensitivity is associated with poor recovery of pain: Prediction, cluster, and decision tree analyses* [36], que sigue el mismo procedimiento pero su objetivo es el de detectar los atributos que determinan la variación de la recuperación de pacientes con diversos tipos de dolores, clasificándolos en si se produjo una recuperación, mantenimiento o si empeoraron los síntomas.

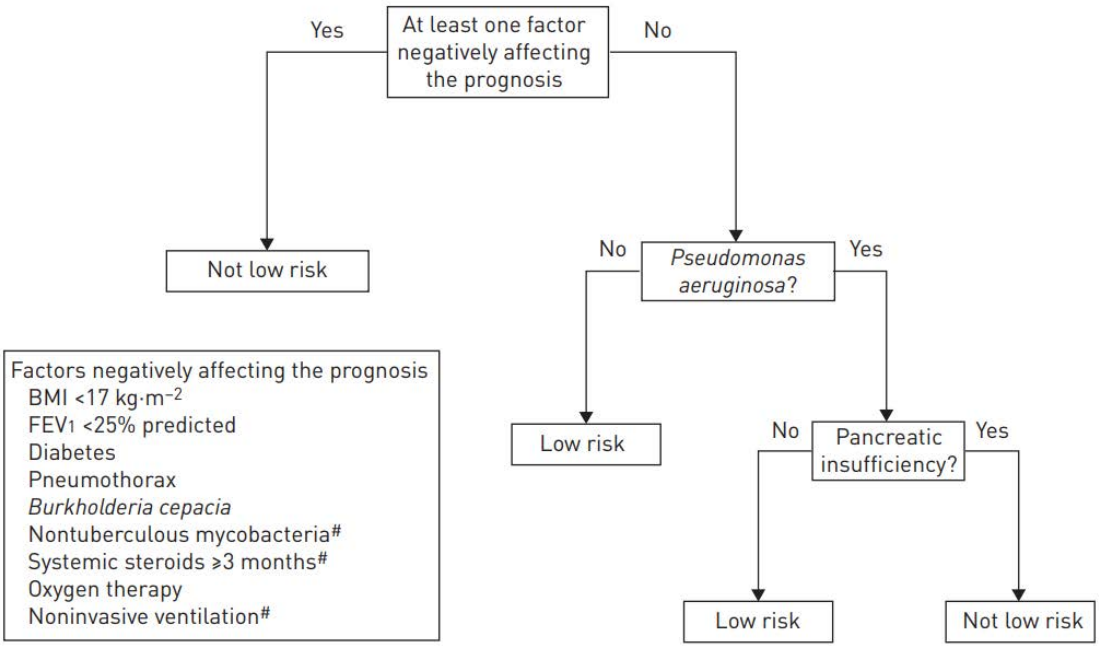

Figura 9. Resultados obtenidos del riesgo de muerte en adultos con fibrosis cística obtendios mediante árbol CART. [35]

También hay trabajos que muestran cómo se pueden caracterizar los clústeres mediante reglas de asociación gracias al algoritmo Apriori. Gracias a estas reglas, se puede ver como interaccionan los diferentes atributos que determinan la pertenencia a una u otra clase, así como ver cuantas veces aparecen con respecto al total o con respecto a esa clase, y si es más probable que aparezcan en esa clase que en otra.

En el artículo, *Application of a two-step cluster analysis and the Apriori algorithm to classify the deformation states of two typical colluvial landslides in the Three Gorges, China* [37], se buscaba la reducción del riesgo de deslizamientos de tierra en la región de Three Gorges, China. Para ello se utilizaron datos de 6 años y utilizando un algoritmo de clusterización se llegaron a tres agrupaciones de curvas de desplazamiento mensual: deformación inicial, deformación constante y deformación rápida. Tras esto, se utilizó el algoritmo Apriori para obtener reglas que describiesen las relaciones entre las deformaciones y otros parámetros influyentes como pueden ser la precipitación acumulada del mes anterior, la precipitación máxima diaria o el nivel medio del embalse. Como se puede ver en la *Figura 10*, mediante tres parámetros, Support, Confidence y Lift, se pueden ver las diferencias entre los clústeres, caracterizándolos de forma estadística. Este método sirvió como referencia para posteriores análisis de deformación por los deslizamientos de tierra en la región de Three Gorges.

Consequent	Antecedent	Support (%)	Confidence (%)	Lift
Rapid deformation	MDPM = high, ARLM = low, RLCM = slow and CPPM = high	4.00	100	2.03
Rapid deformation	MDPM = high, ARLM = low, RLCM = ordinary and CIRL = slow	4.00	100	2.03
Rapid deformation	MDPM = moderate, RLCM = fast and CPPM = high	4.00	100	2.03
Constant deformation	MDPM = moderate, ARLM = moderate and CIRL = slow	4.00	100	2.03
Constant deformation	CIRL = ordinary, ARLM = moderate and MDPM = low	4.00	100	2.03
Constant deformation	RLCM = ordinary, CPPM = moderate and MDPM = low	4.00	100	2.03
Constant deformation	MDPM = moderate, CPPM = moderate, ARLM = low and CIRL = slow	5.33	75	1.52
Initial deformation	MDPM = very low, ARLM = high and RLCM = slow	8.00	83	1.95
Initial deformation	MDPM = very low, ARLM = high, RLCM = slow and CIRL = slow	8.00	83	1.95
Initial deformation	MDPM = very low, ARLM = high, RLCM = slow, CPPM = low and CIRL = slow	8.00	83	1.95
Initial deformation	CPPM = low, RLCM = ordinary, ARLM = moderate and CIRL = slow	5.33	75	1.76

Figura 10. Reglas de asociación para la deformación por deslizamientos de tierra. [37]

Además, la eficacia del algoritmo Apriori aumenta al llevar a cabo una clusterización previa, esto nos lo muestra el artículo, *Analysis of Accuracy K-Means and Apriori Algorithms for Patient Data Clusters* [38]. En este trabajo se muestra como Apriori tiene una gran debilidad en el tiempo computacional, ya que tiene que escanear todos los datos repetidamente para cada conjunto de atributos. Es por ello que, como podemos ver en la *Figura 11*, se utilizó el algoritmo Apriori dos veces, uno sobre todos los datos y otro después de haber hecho una clusterización por K-medias, para ver si se observaban discrepancias entre los tiempos de ejecución.

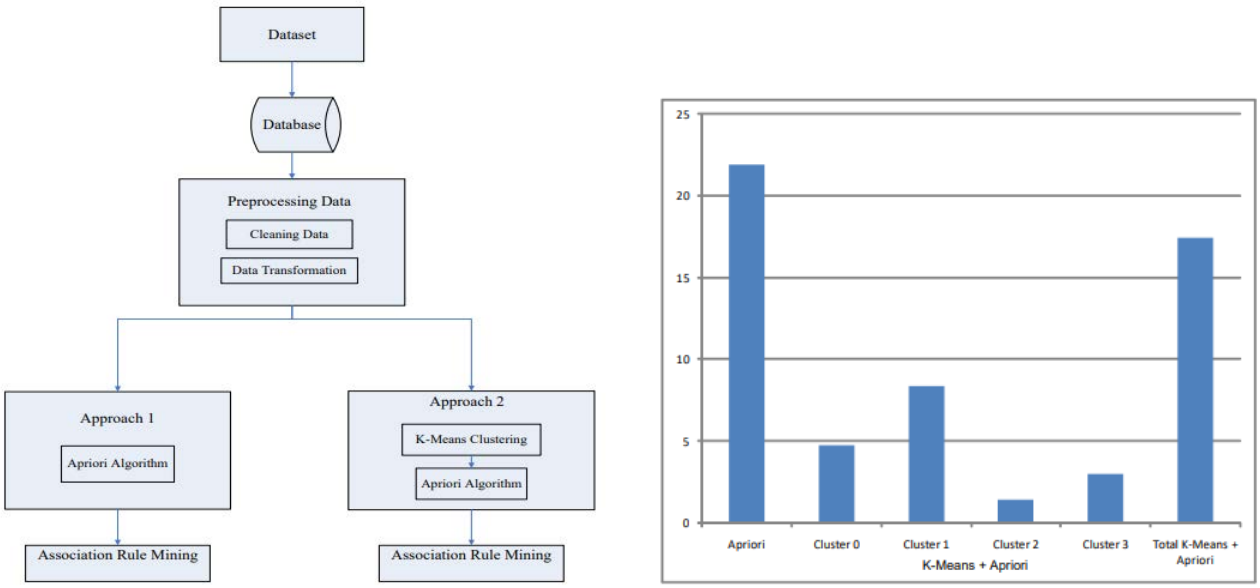

Figura 11. Disminución de tiempo al usar Apriori tras K-medias. [38]

Como se puede ver en la *Figura 11*, además de la obtención de una información más detallada y completa, al usar estos dos algoritmos se consiguió una reducción del tiempo en un 21%.

2.5 Tecnologías y Framework de investigación

Este subapartado tiene como objetivo la enumeración de las diferentes herramientas y programas que han sido usados durante la realización de la investigación, así como dar una justificación de por qué han sido seleccionadas estas y no otras.

En cuanto a la primera parte de preprocesado de datos que será explicada en mayor medida en el punto 4 de este trabajo, se ha usado la base de datos desarrollada por Google, *BigQuery*. En la actualidad existen numerosas bases de datos, siendo las más conocidas aquellas basadas en SQL como *MySQL* o *MariaDB* y las no basadas en SQL (o también llamadas "no solo SQL") como *MongoDB* o *Cassandra*.

En este caso se decidió usar *BigQuery*, ya que fue mediante esta herramienta como MasMovil nos dio acceso a los datos a estudiar. Esta base de datos además de soportar los comandos SQL que permiten hacer consultas para entender los datos y hacer limpieza en estos, también tiene incorporadas herramientas de Aprendizaje Automático, permitiendo hacer modelos de forma más rápida.

En cuanto a la elección del lenguaje a utilizar en el resto del trabajo se ha hecho un pequeño estudio debido al gran número de estos que hay, así como debido al incremento

del uso de Inteligencia Artificial, que ha conllevado a que aparezca un creciente énfasis en el desarrollo de software para llegar a resultados de forma más rápida, sencilla y con una mayor precisión. Es por ello por lo que la elección de uno u otro lenguaje de programación es fundamental según los beneficios que más interesan para un problema dado. Esto es así debido a que si con esta investigación se buscase una gran eficiencia, debido a que los datos de entrada fuesen muy pesados, se podrían optar por lenguajes como R o C/C++,debido a su eficiencia con datos, mientras que si esta no fuese una prioridad, como es el caso en el que nos encontramos, se podría optar por otros lenguajes como Python, que ha sido el lenguaje elegido, el cual tiene una serie de beneficios que se expresarán más adelante y es por ello que es el lenguaje más usado actualmente, según datos aportados por Anaconda [39], la popular herramienta de Machine Learning, así como el lenguaje más enseñado por los profesores para preparar a los alumnos en la entrada al Aprendizaje Automático, con un 88% de los encuestados.

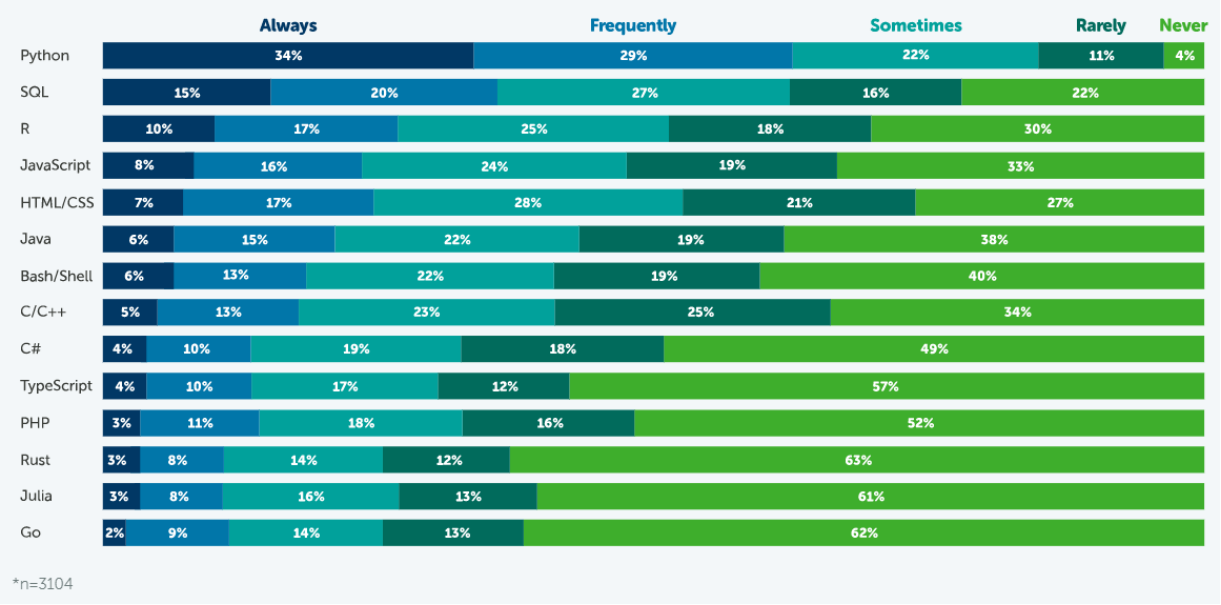

Figura 12. Encuesta sobre el uso de lenguajes de programación. [39]

Los principales beneficios de *Python* son:

- Es un lenguaje de programación sencillo con una baja curva de aprendizaje.
- Es compatible con todos los sistemas operativos, pudiendo ejecutar un script en cualquier entorno.
- Debido a su gran popularidad, tiene una comunidad fuerte, por lo que se pueden encontrar respuestas a errores con gran facilidad

- Muy fácil de leer e interpretar.
- Tiene librerías y frameworks muy completos como aquellos para el tratamiento de datos, como *Pandas* o *Numpy*, o para aprendizaje automático, como *Tensorflow* o Scikit-learn.
- Permite una muy buena visualización de datos gracias a librerías como Seaborn o Matplotlib

Además, es compatible con *Google Colaboratory*, el cual nos permite entrenar modelos en la nube, sin tener que usar los recursos de nuestros ordenadores, y pudiendo ejecutar varios scripts a la vez, asignando a cada uno su propia memoria RAM, agilizando en gran medida el proceso. Este sistema está basado en *Jupyter Notebook*, el cual permite ejecutar ciertas partes del código al hacer cambios en este sin tener que compilarlo todo de nuevo, lo cual es muy útil para entrenar los diferentes modelos modificando los parámetros de forma más ágil y eficaz.
Las grandes limitaciones de este servicio son la memoria RAM que se nos ofrece de forma gratuita 13GB y el tiempo máximo de ejecución, que es de 12 horas. Para ciertos modelos, estas prestaciones pueden llegar a ser insuficientes.

3 Preprocesado de datos

Una vez se ha entendido el objeto de la investigación y antes de proceder a la aplicación de técnicas para la detección de errores de transferencia se tendrá que llevar a cabo un preprocesado de los datos que consistirá en el análisis y entendimiento de estos para una posterior limpieza, transformación y preparación que se adecúe a los diferentes modelos que serán explicados en el siguiente apartado.

3.1 Base de datos

Este primer subapartado se centrará en la explicación de las tablas de datos aportadas por el Grupo MasMovil, así como sus diferentes atributos y los tipos de información de estos, y servirá como primer paso para entender los datos y como justificación a las decisiones tomadas a posteriori en lo referente a la limpieza y transformación de estos.
Como ya se ha comentado con anterioridad, los datos se cedieron a través de la base de datos *BigQuery*, dando acceso a dos tablas, siendo la primera referente a **llamadas** y la segunda a los **segmentos** de estas llamadas, que contienen datos recogidos entre el 1 de enero de 2019 y el 11 de agosto de 2021.

3.1.1 Llamadas

En esta tabla se almacenan los datos sobre las llamadas que se hacen en el *call center*, para las diversas operadoras que forman parte del Grupo MasMovil. Estas han sido un total de 133.959.865 de llamadas en los últimos 3 años, lo que supone una media de alrededor de unas 100.000 llamadas diarias.
En la *tabla 1* se muestran los diferentes campos que se recogen por cada una de las diferentes llamadas, con sus tipos, una pequeña descripción y un ejemplo para que al lector le sea de mayor facilidad su comprensión.
Se cuenta con un total de 16 atributos por cada llamada, o lo que es lo mismo, unos 21 GigaBytes que han de ser procesados ya que en este trabajo debido a las limitaciones técnicas en cuanto a memoria no se puede trabajar con tanta información.

Atributo	Tipo	Descripción	Ejemplo
day	*Date*	Día en el que se realizó la llamada.	2019-10-01
ucid	*String*	Universal call ID. Es único para cada llamada y no se puede repetir.	10201132401569914611
call_data_source	*String*	Es el software utilizado que proporciona las funcionalidades de una central telefónica. También es el sistema que se encarga de la recolección de estos datos.	ASTERISK
phone_number_hash	*Integer*	Número de teléfono *hasheado* desde el que se ha llamado.	30872878911622026
call_type	*String*	Puede tomar los valores *INCOMING* u *OUTGOING* dependiendo de si la llamada tiene origen en el cliente o en el *call center*.	INCOMING
start_time	*Timestamp*	Momento de comienzo de la llamada.	2019-10-01 07:23:31 UTC
end_time	*Timestamp*	Momento de finalización de la llamada.	2019-10-01 07:23:52 UTC
talk_time	*Integer*	Tiempo total que el cliente ha hablado con los agentes.	10
waiting_time	*Integer*	Tiempo que el cliente ha estado esperando en las colas.	5
hold_time	*Integer*	Tiempo que los agentes han mantenido al cliente en espera.	3
after_call_time	*Integer*	Tiempo utilizado por los agentes tras la llamada para redactar informes antes de atender otra llamada.	4
ringing_time	*Integer*	Tiempo que han tardado los agentes en contestar la llamada cuando se la han pasado	6
some_massive_result_ok	*Boolean*	Toma valor TRUE cuando ha habido una incidencia masiva como la caída de una antena que provoque una gran	TRUE

		cantidad de llamadas entrantes.	
lp_offer_speech	*Boolean*	Tomará valor TRUE cuando se permita la conversación mediante audios.	FALSE
lp_accept_whatsapp	*Boolean*	Tomará valor TRUE cuando se permita la conversación por whatsapp.	TRUE
lp_accept_sms	*Boolean*	Tomará valor TRUE cuando se permita la conversación por sms.	TRUE

Tabla 1. Atributos tabla llamadas.

Como se puede observar hay una gran cantidad de datos que se extraen por cada llamada, por lo que se ha llevado a cabo un proceso de comprensión para determinar los atributos que dan información relevante para el caso y cuáles pueden ser eliminados sin que afecte a los resultados finales.

Mientras se realizaba este estudio se observó la cantidad de información faltante que tenían algunas columnas como *some_massive_result_ok, lp_offer_speech, lp_accept_whatsapp* o *lp_accept_sms*, que llegaban a más del 95% de valores *null*.

La variable *call_data_source* da información sobre que software es el que se ha utilizado en la central telefónica para cada llamada, y, como se puede observar en la *figura 13,* ASTERISK es el que mayor número de llamadas tiene. Esto es así porque es el principal sistema que la compañía utiliza, pero también hay otras fuentes, ya que al incorporar nuevas marcas al Grupo MasMovil se mantiene el sistema de estas hasta que se consigue unificar todo para tener un sistema común, lo cual tarda un tiempo.

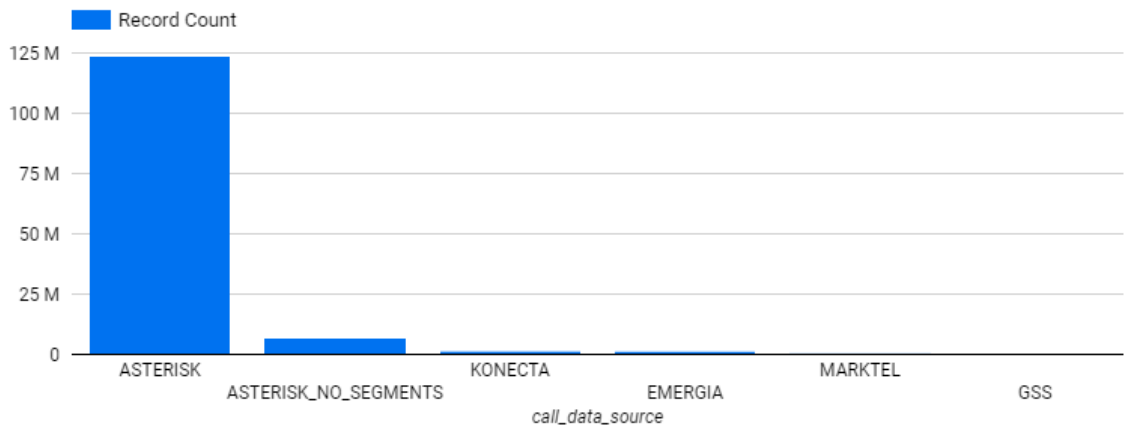

Figura 13. Cantidad ocurrencias variable call_data_source

Mientas se produce este proceso de unificación a ASTERISK y estas operadoras continúan usando los sistemas anteriores, los datos se guardan de forma diferente y es por eso que algunas variables como *talk_time, waiting_time, hold_time, after_call_time* y *ringing_time* tienen todos los valores nulos con los sistemas KONECTA, MARKTEL, EMERGIA, GSS y ASTERISK_NO_SEGMENTS.
Además, mediante se produce este proceso de migración se pueden llegar a usar varios sistemas a la vez dependiendo del departamento en el que se encuentre el cliente, por lo que las llamadas pueden aparecer duplicadas o divididas con diferente *call_data_source*.

3.1.2 Segmentos

Cada una de las llamadas que se registran en la tabla anterior se dividen y se almacenan en la tabla segmentos. Cada segmento es uno de los actores que actúan dentro de la llamada. Algunos de estos actores como la IVR, ACD o lo agentes ya fueron explicados en el apartado 2.1 de este documento, pero hay más que serán explicados más adelante.
En la *tabla 2,* se mostrará que datos son los almacenados dentro de la tabla de segmentos dando una pequeña explicación de cada uno de ellos, así como un ejemplo de muestra.
Este conjunto de datos cuenta con un total de 33 atributos, o lo que es lo mismo, unos 73 GigaBytes en las más de 350 millones de filas.

Atributo	**Tipo**	**Descripción**	**Ejemplo**
day	*Date*	Día en el que se realizó la llamada.	2019-03-03
id	*Integer*	Identificador único del segmento.	486639025
ucid	*String*	Identificador de la llamada a la que pertenece el segmento.	100211824515515943
segment_order	*Integer*	Orden del segmento dentro de la llamada.	2
phone_number_hash	*Integer*	Número de teléfono *hasheado* desde el que se ha llamado.	794964198597292852
current_vdn	*String*	Identificador que mapea por donde tiene de ir la llamada.	91850
output_vdn	*String*	VDN de salida.	90657
segment_type	*String*	Tipo de segmento (Agente, IVR, ACD…)	AGENT
call_type	*String*	Origen de la llamada.	INCOMING

acd	*String*	Identificador de la cola que tomará valor siempre que sea un segmento ACD.	65508
start_time	*Timestamp*	Momento de comienzo del segmento.	2019-03-03 06:26:02 UTC
end_time	*Timestamp*	Momento de finalización del segmento.	2019-03-03 06:26:57 UTC
length_time	*Integer*	Duración total del segmento.	55
talk_time	*Integer*	Tiempo que el cliente ha hablado con un agente.	50
waiting_time	*Integer*	Tiempo que el cliente ha estado esperando en una cola.	5
hold_time	*Integer*	Tiempo que el agente mantiene al cliente en espera.	6
after_call_time	*Integer*	Tiempo utilizado por el agente tras la llamada para redactar informes antes de atender otra llamada.	20
ringing_time	*Integer*	Tiempo que ha tardado el agente en contestar la llamada cuando se la han pasado	5
acd_overflowed	*Boolean*	Toma el valor TRUE cuando se ha llenado una cola (ACD) y mandan al cliente a otra.	FALSE
categories_open_question	*String*	Categorías que extrae el segmento de YDILO a lo expresado por el cliente con lenguaje natural.	[INTERNET, AVERIAS]
subcategories_open_question	*String*	Subcategorías extraídas por el YDILO.	[CONSUMO, INTERNET_FIJO]
categories_subcategories_open_question	*String*	Unión de las categorías y su subcategoría de lo extraído por el YDILO.	[INTERNET__CONSUMO,AVERIAS__INTERNET_FIJO]
some_massive_result_ok	*Boolean*	Toma valor TRUE cuando ha habido una incidencia masiva.	TRUE
agent_id	*Integer*	ID del agente.	3971
payment_ok	*Boolean*	Toma el valor TRUE cuando se está en el departamento de cobros y se produce un pago.	FALSE
brand	*String*	Operadora del Grupo MasMovil que se hace cargo de la llamada.	YOIGO
department	*String*	Departamento del agente.	ATC
service	*String*	Servicio que se lleva a cabo en el departamento.	ATENCION NO TECNICA

acd_agency	*String*	Agencia que se hace cargo del ACD.	GLOBALIA
acd_workplace	*String*	Lugar en el que se encuentra la agencia de ACD.	MALLORCA
lp_offer_speech	*Boolean*	Tomará valor TRUE cuando se permita la conversación por audios.	TRUE
lp_accept_whatsapp	*Boolean*	Tomará valor TRUE cuando se permita la conversación por whatsapp.	FALSE
lp_accept_sms	*Boolean*	Tomará valor TRUE cuando se permita la conversación por sms.	TRUE

Tabla 2. Atributos tabla segmentos.

Como se puede observar en la *figura 14*, hay siete tipos diferentes de segmentos en los que se pueden dividir las llamadas:

- **IVR**: Como ya se ha explicado con anterioridad la IVR es el sistema con el que empiezan casi todas las llamadas que puede resolver las consultas o redireccionar a los clientes a un ACD. Es el más numeroso ya que la mayoría de las llamadas cuentan con al menos uno, pudiendo ser varios en caso de que el cliente tuviese más de una consulta.
- **YDILO**: Es un IVR especial que es capaz de comprender el lenguaje natural. Este segmento le hace una pregunta abierta al cliente del tipo *"¿Qué es lo que quiere?"* y es capaz de extraer una serie de atributos de la respuesta que servirán para redireccionar al cliente a un agente u otro. Estos atributos se guardan en forma de *array* de *Strings* en las variables *categories_open_question, subcategories_open_question* y *categories_subcategories_open_question*.

  A diferencia que con el segmento IVR, que no se tienen datos sobre lo que han dicho los clientes, en el YDILO sí que se tienen estos atributos, haciendo que este segmento sea mucho más rico en información y pudiendo conseguir entender por qué se ha mandado a un cliente a un agente y no a otro y poder arreglarlo en caso de que se produzca un error de transferencia.
- **AGENT**: Son las personas físicas que atienden a los clientes.
- **ACD**: Como ya se ha explicado con anterioridad son las colas donde esperan los clientes a ser atendidos por los agentes.

- **NO_INFO** y **OUTSIDE_CALL_CENTER**: es el valor guardado por el sistema cuando no puede determinar donde se encuentran los clientes. La mayoría de las veces ocurren cuando durante una llamada se cambia de *call_data_source*, por ejemplo, de ASTERISK a KONECTA.
- **PCI**: Es una IVR especial destinada a cobros.

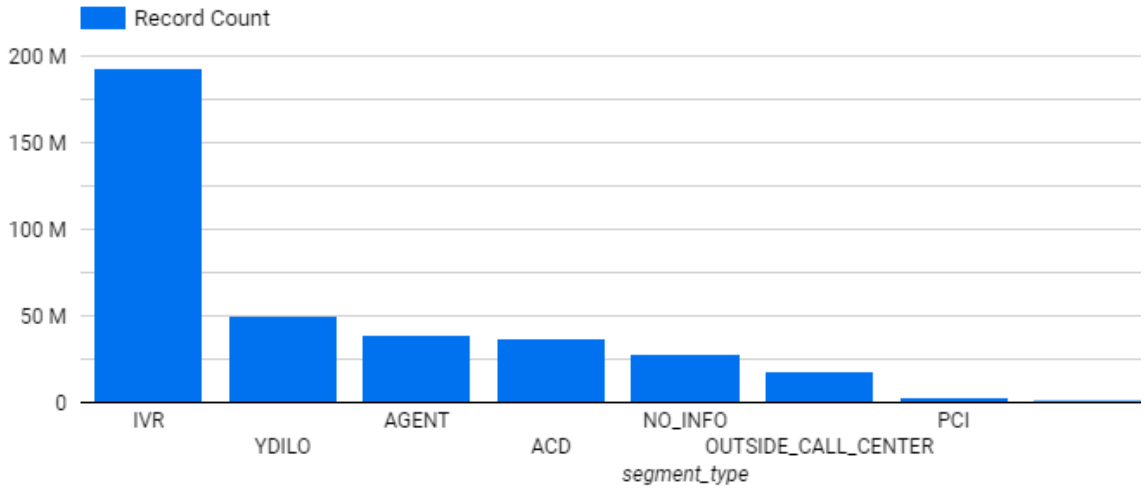

Figura 14. Cantidad ocurrencias variable segment_type

Para ver la relación entre las tablas de llamadas y segmentos, se ha realizado un breve análisis para determinar cuántas llamadas hay y cuantos segmentos tienen estas, buscando determinar también si existen llamadas que tienen más de dos agentes para poder estudiar si existen errores de transferencia entre estos.

Métricas	**Valores**
Número de llamadas	133.959.865
Número de segmentos	369.862.103
Media de segmentos por llamada	2.76
Media de agentes por llamada	0.28
Llamadas con al menos un agente	34.529.327
Llamadas con al menos dos agentes	2.919.086

Tabla 3. Resumen tablas llamadas y segmentos.

Se puede observar como la mayoría de las llamadas no son pasadas con ningún agente, resolviéndola gracias al uso de la IVR, y tan solo un 2% cuentan con más de dos agentes, entre los que habrá que diferenciar entre errores de transferencia y aquellos clientes que cuenten con más de una consulta y tengan que pasar por diferentes agentes para resolverlas.

Todos estos datos provienen de cuatro operadoras, que son las principales fuentes de beneficios del Grupo MasMovil, que son: Yoigo, MasMovil, PepePhone y LlamaYa, siendo Yoigo la que más segmentos tiene ya que es también cuenta con mayor número de clientes.

Los resultados de esta investigación podrían variar si se usasen los datos de todas las operadoras, ya que la estrategia de PepePhone y LlamaYa va más orientada a bajos precios y tarjetas con saldo, por lo que sus llamadas son muy diferentes a las que se hacen a MasMovil y Yoigo.

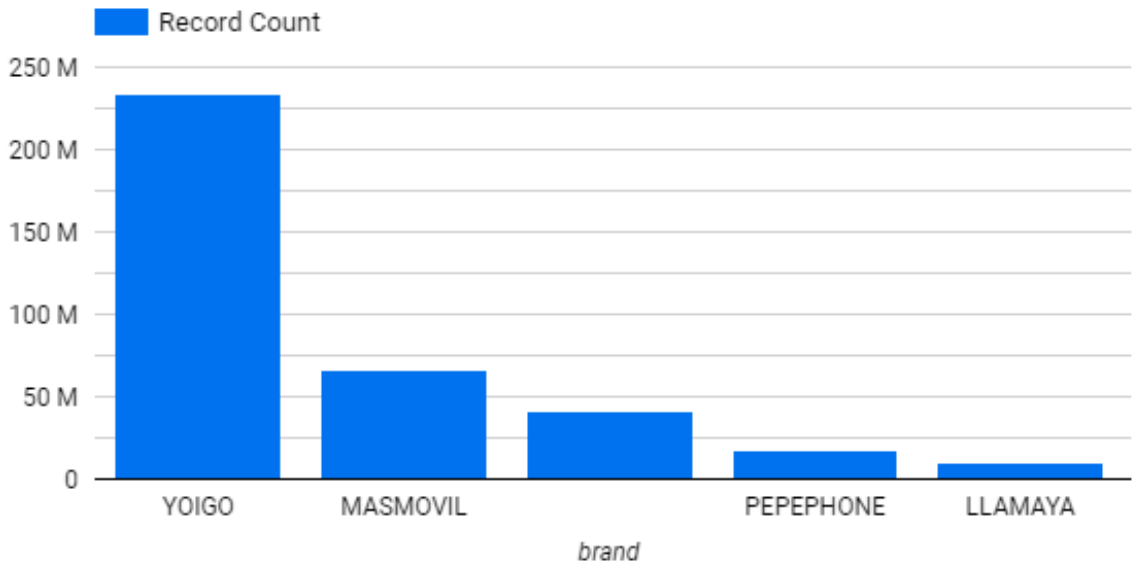

Figura 15.Cantidad ocurrencias variable brand.

En cuanto a los departamentos de los agentes, hay nueve diferentes, siendo el de ATC (atención al cliente) el que mayor número de clientes absorbe como se puede observar en la *figura 16*.

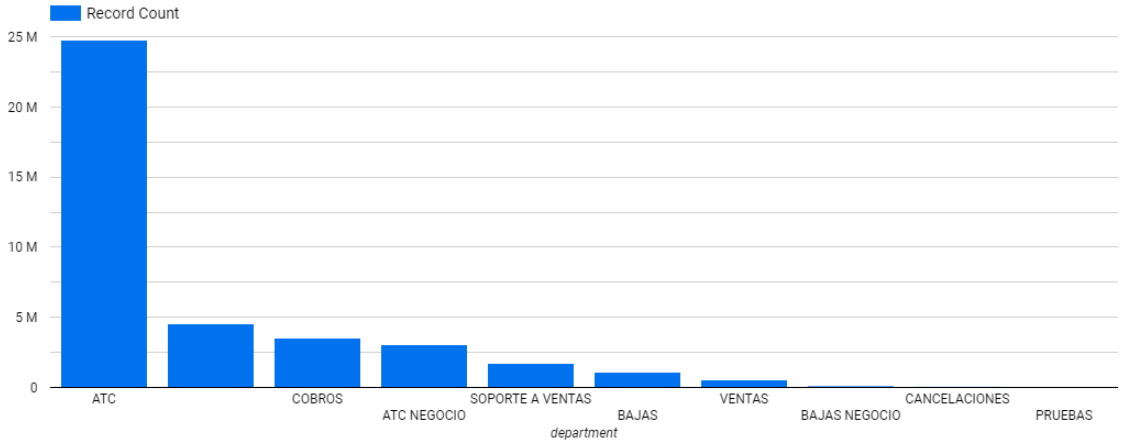

Figura 16.Cantidad ocurrencias variable department cuando el segmento es de tipo Agente.

Estos departamentos a su vez se subdividen en servicios, que serán las tareas que puedan realizar los agentes dentro de estos. Dependiendo de la granularidad que sea preferible para el problema en cuestión, será más útil utilizar algo más general como son los departamentos o algo más específico, como los servicios.

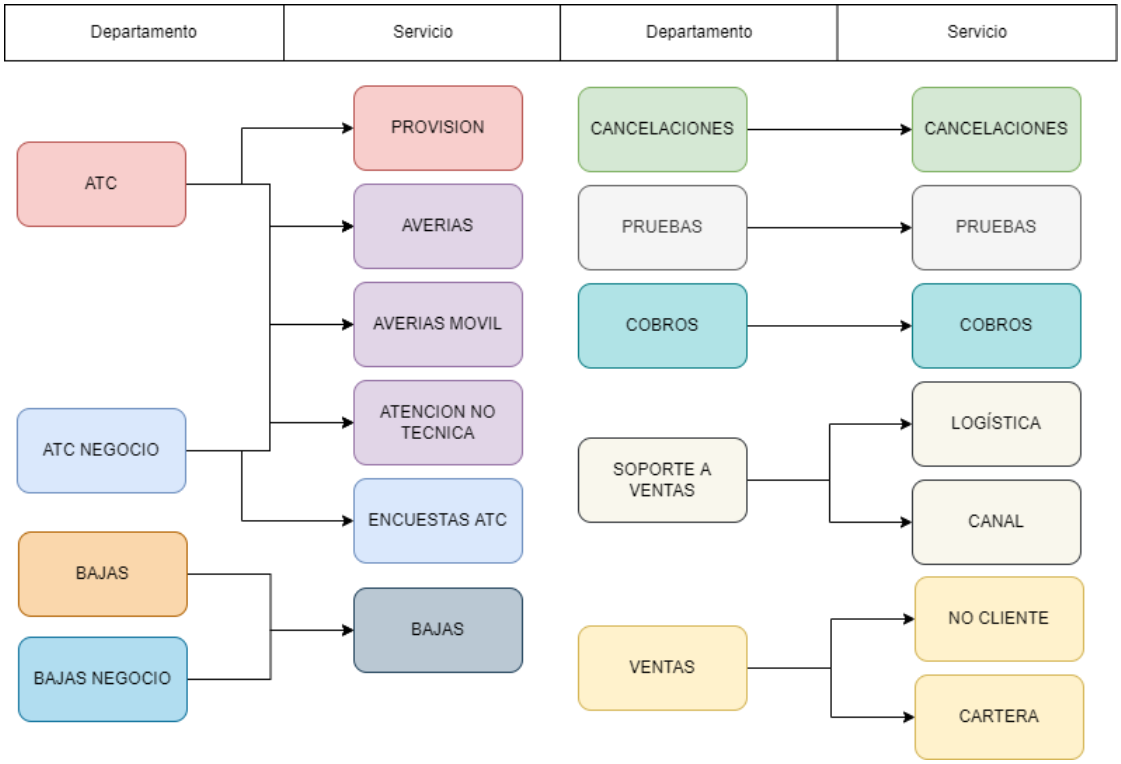

Figura 17. Servicios por departamento.

Como último punto en este apartado, cabe mencionar el análisis de otros atributos como *current_vnd, output_vnd* y *acd*, ya que se ha detectado que son redundantes, debido a que

estos son los atributos que el sistema guarda y mediante un proceso de postprocesado llevado por la compañía, se determinan los valores de los atributos *department*, *service*, *acd_agency* y *acd_workspace* a partir de estos otros.
También existen otros atributos irrelevantes como *call_type* cuyo valor es *INCOMING* en un 99% de los casos por lo que también será un atributo a descartar.

3.2 Limpieza y corrección de errores

Una vez que se han estudiado los datos ofrecidos y se tiene un mayor entendimiento de las tablas, así como de las distribuciones de sus atributos, se lleva a cabo el proceso de limpieza que constará de varios pasos, empezando con una eliminación de aquellos atributos que se consideren irrelevantes o que no contengan suficientes valores, seguido de una corrección de errores presentes en los datos, finalizando con la selección del conjunto que va a ser utilizado para entrenar los modelos.
Como se ha visto en el subapartado anterior hay grandes similitudes entre las tablas de llamadas y de segmentos, siendo esta segunda más descriptiva y teniendo casi todos los atributos de la primera y añadiendo otros muchos, por lo que a partir de este apartado tan solo se van a usar los datos de la tabla segmentos desechando la tabla de llamadas.
Debido a que nos encontramos ante un problema con datos no supervisados, determinar si unos atributos son relevantes o no es de gran complejidad, ya que no se pueden hacer pruebas del tipo chi^2 ni con ningún otro estimador que refleje estadísticamente si una columna produce un efecto significativo en una clase, por lo que se tendrán que hacer diversos experimentos variando el número de atributos en la clusterización para determinar cuáles son los más importantes y los que determinan la pertenencia a una clase u otra.
Por lo tanto, en este paso, solo se eliminarán aquellos atributos que cuenten con una gran cantidad de valores nulos o el mismo valor repetidamente y que por lo tanto no aporten mucha información, aquellos que sean redundantes y aquellos que sean categóricos y cuenten con muchos valores diferentes y no se haya detectado que sean muy importantes para el caso. Estos atributos son: *day, id, phone_number_hash, current_vdn, output_vdn, call_type, acd, start_time, end_time, acd_overflowed, some_massive_result_ok, agent_id, payment_ok, acd_agency, acd_workplace, lp_offer_speech, lp_accept_whatsapp* y *lp_accept_sms*.

En cuanto a la corrección de errores, se han detectado diversos problemas, de los cuales algunos han podido ser subsanados, siendo estos, errores en los valores de los tiempos y datos faltantes sobre los departamentos y servicios en algunos segmentos.

Duración de segmento nula

En algunos casos la variable *length_time* aparecía con valor nulo, esto se podía dar por varias razones, la primera que hubiese habido algún tipo de error en la toma de las variables *start_time* y *end_time* siendo alguna de ellas nula, ya que *length_time* se calcula como la resta de las dos anteriores. Esto se arregló de dos formas diferentes, la primera fue cogiendo los valores de *start_time* del segmento posterior y el de *end_time* del segmento anterior, en caso de que hubiese, para sustituirlos por los del segmento con valores nulos. En caso de que no existiesen estos segmentos posteriores o anteriores se hizo la suma de los tiempos de *talk_time, waiting_time, hold_time* y *ringing_time*, en caso de que no fuesen nulos también, ya que la suma de estos es ser el total de tiempo del segmento.

ucid	segment_order	segment_type	start_time	end_time	length_time
10025330101549654110	1	IVR	2019-02-08 19:28:30 UTC	2019-02-08 19:29:01 UTC	31
10025330101549654110	2	ACD	2019-02-08 19:29:01 UTC	2019-02-08 19:29:13 UTC	12
10025330101549654110	3	AGENT	2019-02-08 19:29:13 UTC	*null*	*null*
10025330101549654110	4	IVR	2019-02-08 19:29:30 UTC	2019-02-08 19:30:54 UTC	24

Figura 18. Error en la toma del end_time corregido con el siguiente start_time

También se producían algunos nulos en caso de que alguna llamada transcurriese entre dos días, pero estas fueron fácilmente arregladas con una simple resta de los valores *start_time* y *end_time*.

Duración 0 o muy pocos segundos en segmento de agente

En otros casos aparecían segmentos de agente con una duración de entre 0 y 3 segundos, que como se puede ver en la *figura 19*, siempre tienen un agente después con un valor mayor de tiempo. No hay una razón clara de por qué sucede este error, pero se solucionó fácilmente eliminando todos estos segmentos, debido a que realmente no han ocurrido, ya que no se ha llegado a hablar con ningún agente.

ucid	segment_order	segment_type	start_time	end_time	length_time
10021007771585045701	7	AGENT	2020-03-24 10:37:00 UTC	2020-03-24 10:37:00 UTC	0
10021007771585045701	8	AGENT	2020-03-24 10:37:01 UTC	2020-03-24 10:37:01 UTC	0
10021007771585045701	9	AGENT	2020-03-24 10:37:02 UTC	2020-03-24 10:37:03 UTC	1
10021007771585045701	10	AGENT	2020-03-24 10:37:03 UTC	2020-03-24 10:37:04 UTC	1
10021007771585045701	11	AGENT	2020-03-24 10:37:05 UTC	2020-03-24 10:37:05 UTC	0
10021007771585045701	12	AGENT	2020-03-24 10:37:06 UTC	2020-03-24 10:38:42 UTC	96

Figura 19. Segmentos de agente con duración 0.

Estos segmentos de agente inexistentes también ocurrían cuando se intentaba pasar con un agente y este no cogía el teléfono, y tras un tiempo de estar sonando este se redirigía la llamada a otro agente, como se puede ver en la *figura 20*. La solución a este error fue la misma que la anterior, eliminar todos estos segmentos en los que no se ha llegado a hablar con ningún empleado.

ucid	segment_order	segment_type	start_time	end_time	length_time	talk_time	ringing_time
10021371421576142256	1	YDILO	2019-12-12 09:17:36 UTC	2019-12-12 09:17:59 UTC	23	*null*	*null*
10021371421576142256	2	ACD	2019-12-12 09:18:08 UTC	2019-12-12 09:22:21 UTC	253	*null*	*null*
10021371421576142256	3	AGENT	2019-12-12 09:20:56 UTC	2019-12-12 09:21:17 UTC	21	0	21
10021371421576142256	4	AGENT	2019-12-12 09:22:19 UTC	2019-12-12 09:27:36 UTC	317	316	1
10021371421576142256	5	IVR	2019-12-12 09:27:36 UTC	2019-12-12 09:28:08 UTC	32	*null*	*null*

Figura 20. Redirección llamada por ausencia del agente.

Duración de segmento muy larga

También se detectaron segmentos cuyo valor de *length_time* era muy elevado llagando a duraciones de más de 7 millones de segundos, o lo que es lo mismo, unos 3 meses hablando con un mismo agente. En estos casos no se pudo subsanar el error, pero se eliminaron todas estas llamadas para que estos errores no afectasen al resto de la base de datos.

ucid	segment_order	segment_type	start_time	end_time	length_time
10027487841546646172	1	IVR	2019-01-04 23:56:13 UTC	2019-01-04 23:56:53 UTC	40
10027487841546646172	2	AGENT	2019-01-04 23:56:53 UTC	2019-04-06 00:31:01 UTC	7864448

Figura 21. Segmentos con duraciones superiores a 3 meses.

Tiempo negativo hablando con un agente

El último error encontrado con respecto a los tiempos de los segmentos está en la aparición de tiempos con valor negativo en la variable de *talk_time*.

ucid	segment_order	segment_type	start_time	end_time	length_time	talk_time	hold_time	after_call_time	ringing_time
10036034211623950906	1	AGENT	2021-06-17 17:28:51 UTC	2021-06-17 17:28:57 UTC	6	-19	0	30	25
10036162001623936422	1	AGENT	2021-06-17 13:27:30 UTC	2021-06-17 13:27:51 UTC	21	-7	0	0	28
10036221111623938074	1	AGENT	2021-06-17 13:55:00 UTC	2021-06-17 13:55:19 UTC	19	-7	0	0	26
10036435651623944255	1	AGENT	2021-06-17 15:38:03 UTC	2021-06-17 15:38:27 UTC	24	-4	0	0	28
10042353111623930286	7	AGENT	2021-06-17 11:48:50 UTC	2021-06-17 11:49:21 UTC	31	-1	0	0	32
10043440821623954719	5	AGENT	2021-06-17 18:36:41 UTC	2021-06-17 18:36:47 UTC	6	-61	0	0	67
10026503781592409207	2	AGENT	2020-06-17 15:54:04 UTC	2020-06-17 15:54:05 UTC	1	-28	0	0	29
10043411751630337751	7	AGENT	2021-08-30 15:43:11 UTC	2021-08-30 15:43:11 UTC	0	-46	0	0	46

Figura 22. Segmentos de agente con talk_time negativo.

Tampoco está claro de por qué ocurre este fenómeno, pero es un caso muy parecido al de los segmentos donde no se hablaba con un agente, sino que todo el tiempo era esperando a que este cogiese la llamada y como eso no ocurría se le redireccionaba al cliente con otro agente. Por lo que la solución también fue la misma, eliminar todos los segmentos en los que ocurriese este fenómeno (*ringing_time* >= *length_time*).

Transformación de departamentos y servicios nulos

Otro error que se detectó fue la aparición de valores nulos para las variables de departamento y servicios en algunos segmentos de agente. Estos se producen cuando un cliente es redirigido a otro agente sin pasar por el ACD, esto provoca que el sistema no sea capaz de determinar estos campos, ya que es el campo de ACD el que el sistema utiliza para rellenar estos atributos.

day	ucid	segment_order	segment_type	length_time	department	service	agent_account_hash
2019-03-30	10025484771553931523	1	IVR	20	ATC	ATENCION NO TECNICA	*null*
2019-03-30	10025484771553931523	2	IVR	12	ATC	ATENCION NO TECNICA	*null*
2019-03-30	10025484771553931523	3	IVR	23	ATC	ATENCION NO TECNICA	*null*
2019-03-30	10025484771553931523	4	ACD	156	ATC	ATENCION NO TECNICA	*null*
2019-03-30	10025484771553931523	5	AGENT	1019	ATC	ATENCION NO TECNICA	6452779334151820360
2019-03-30	10025484771553931523	6	AGENT	777	*null*	*null*	8875988688757287565

Figura 23. Segmento con valores nulos en los atributos de departamento y servicio.

Esta falta de datos se pudo arreglar debido a que todos los agentes se encuentran en un mismo departamento realizando un mismo servicio durante un cierto periodo de tiempo antes de cambiar a otro, por lo que se creó una nueva tabla que contenía la información de en qué departamentos y servicios habían estado todos los agentes y su periodo de tiempo. Una vez se tenía la tabla fue muy fácil sustituir los valores nulos por los correspondientes gracias al *agent_id* y el día de la llamada.

day_start	day_end	department	service	agent_account_hash
2019-01-11	2019-04-26	ATC	ATENCION NO TECNICA	8875988688757287565
2019-02-28	2019-03-18	ATC	ATENCION NO TECNICA	8881807260356478194
2019-05-18	2019-05-18	ATC	AVERIAS	8882740961851975467
2019-02-11	2019-03-03	ATC	PROVISION	8882740961851975467

Figura 24. Tabla con los datos de todos los agentes.

Selección del conjunto de datos a usar

A la hora de seleccionar el conjunto de datos para entrenar los modelos se decidió coger los segmentos entre las fechas del 1 de enero del 2019 (primer día desde el que se tienen datos) hasta el 1 de junio del 2020, ya que a partir de esa fecha es cuando se empezaron a usar otros softwares de central telefónica como EMERGIA o MARKTEL que producían llamadas duplicadas y la aparición de los segmentos NO_INFO y OUTSIDE_CALL_CENTER cuando en una misma llamada se utilizaban diversos sistemas.

Además, con el objetivo de poder descubrir llamadas con errores de transferencia entre agentes, se descartaron todas aquellas llamadas que no tuviesen al menos dos segmentos de agente y que no hubiese ninguna IVR o YDILO entre medias ya que esto significa que son consultas diferentes.

Por último, se descartaron todos aquellos segmentos que no fuesen de agente, ya que el objetivo es el de hacer agrupaciones de estos para poder encontrar donde se han producido los errores.

3.3 Aumento de atributos

Debido a la eliminación de todos los segmentos que no fuesen de agentes y a que no se quería que se produjese una pérdida de información por este motivo, se decidió añadir una serie de atributos que pudiesen ser de ayuda para la *clusterización* que diesen información acerca de todo lo que ha pasado antes y después de que el cliente llegase al segmento de agente.

Estos atributos son:

- Diferente departamento que anterior/siguiente agente: son dos variables que tomarán valor 0 o 1. Toman valor 1 en caso de que el departamento anterior/posterior sea de un departamento diferente. Esta variable pude ser interesante ya que cuando se produce un error de transferencia siempre se va a redirigir a los clientes con otro agente, el cual será del departamento que quería el usuario en un primer momento.
- Diferente servicio que anterior/siguiente agente: son dos variables iguales a las anteriores, pero en este caso guardarán valor 1 en caso de que el servicio sea diferente, ya que en cada departamento puede haber varios servicios como se vio en la *figura 17*, por lo que es posible que ante un error de transferencia no se produjese un cambio de departamento, sino que se podría pasar la llamada a un agente del mismo, pero que lleve a cabo un servicio diferente.
- Departamento antes/después: son dos variables categóricas que guardan el valor del departamento que va antes/después en caso de que lo haya y el valor "*ninguno"* en caso de que no. Da información de si los departamentos que van antes o después determinan una mayor afluencia de errores.
- Servicio antes/después: son iguales que las dos variables anteriores y tendrán el mismo objetivo, pero usando el servicio en vez del departamento.
- Número de agentes antes/después: son dos variables numéricas que guardan el número de agentes que hay antes/después de este segmento.
- Después de YDILO: toma valor 1 en caso de que el segmento que precede al agente sea del tipo YDILO. Esta variable da información de si los errores de transferencia provienen de agentes o de una mala redirección del YDILO.

- Posición después YDILO: es una variable numérica que determina en que posición se encuentra el segmento después del YDILO, o lo que es lo mismo, cuantos agentes hay entre el YDILO y este segmento.
- Tiempo antes/después: son dos variables numéricas que guardan el valor en segundos del tiempo que ha pasado hablando con otros agentes antes de llegar a este segmento y el tiempo que le queda al cliente hablando con agentes para que finalice la llamada.

3.4 Trasformación y normalización de los datos

Antes de poder introducir los datos en los modelos se tiene que llevar a cabo una serie de transformaciones con el objetivo de maximizar la eficacia de estos, ya que normalmente los conjuntos de datos contienen muchos atributos diferentes, que se encuentran en diferentes escalas y son de diferentes tipos.

Es importante arreglar las diferencias de escala en las variables numéricas ya que no es lo mismo hablar del tiempo del segmento que puede tomar valores desde los 10 segundos hasta los 30.000, que el número de agentes después que pueden ir de 0 a 13 como máximo. Tener variables con valores máximos tan dispares puede producir que el modelo de más importancia a unas variables que a otras, alterando los resultados. Es por ello por lo que se ha realizado un proceso de normalización, en este caso se han reescalado todas las variables entre los valores 0 y 1, siendo el 0 el valor más bajo de ese atributo y el 1 el más alto.

Además, antes de poder usar los datos en los modelos, se ha de tener en cuenta los tipos de cada columna de datos, ya que deben ser todos numéricos. Como se puede observar en la *tabla 2*, hay una serie de atributos de tipo *booleano* (que solo aceptan valores de verdadero o falso) y de tipo *String* (que solo aceptan valores en forma de cadena de caracteres).

La transformación de los *booleanos* es muy sencilla, ya que solo se han de remplazar los valores verdaderos por un 1 y los falsos por un 0, haciendo estas variables numéricas.

En cuanto a las de tipo *String*, son variables categóricas, las cuales pueden aceptar una serie de valores, los cuales son limitados, por lo que se utilizó la técnica de *one hot encoding*.

Esta técnica consiste en crear una columna nueva por cada valor distinto que exista en cada atributo. Por lo tanto, para pasar la variable *brand* de categórica a numérica se crean 4 nuevas columnas (*brand_yoigo, brand_masmovil, brand_llamaya, brand_pepephone*) y se desecha este atributo. Cada una de las nuevas columnas toman el valor 1 en caso de que el segmento pertenezca a esa marca y 0 en caso negativo.

Se ha podido usar esta técnica en casi todos los atributos categóricos ya que ninguno de estos tenía más de 10 valores diferentes, por lo que el número de columnas todavía era tratable, excepto para las *variables categories open question, subcategories open question* y *categories subcategories open question*.

El número de valores diferentes de estas columnas era mucho mayor que el del resto, ya que *categories open question* tiene 54 valores diferentes, *subcategories open question* 256 *y categories subcategories open question* 396. Haciendo un breve estudio también se determinó que *categories open question* es una generalización de las otras dos columnas, por lo que se podían descartar las otras dos, ya que estaban altamente correlacionadas.

Esta variable como se vio en la *tabla 2*, está compuesta de un *array* de *Strings,* ya que el cliente pude especificar más o menos lo que quiere. Como se puede ver en la siguiente tabla, en más de un 13% de los casos la longitud de estas características que el YDILO es capaz de detectar es de más de 1, siendo la máxima longitud de 13, por lo que se tuvo que llevar a cabo un proceso para la separación de este array en valores sueltos.

Longitud array categories open question	**Cantidad**	**Porcentaje**
1	800951	86,75%
2	92115	9,98%
3	23888	2,59%
4	4908	0,53%
5	1120	0,12%
Más de 6	256	0,08%

Tabla 4. Longitud de la variable categories open question.

Una vez se tenían los valores sueltos, debido a la gran cantidad de categorías diferentes de atributos que el YDILO es capaz de extraer de lo que dicen los clientes, que como ya se ha dicho es de 54, se buscó agrupar estos para minimizar el número de categorías perdiendo el mínimo posible de información al hacerlo.

Para ello y tras a la aprobación de los expertos de MasMovil se juntaron las categorías en 7 nuevas como se puede observar en la *tabla 5*.

Nuevas categorías	**Antiguas categorías de la variable *categories open question***
Agente	AGENTE
Vacío	empty, SIN_INFORMACION, OTROS, GENERAL, SILENCIO, NO_SERVICIO_ADICIONAL, AMBIGUO, CLIENTES, REPETIR, OTHER, SALUDO, NO_INT_HUNGUP, INVALID, MAX_UNKNOWN_HUNGUP
Problemas físicos	AVERIAS, INSTALACION, COBERTURA
VB	VB_WELCOME_TRANSFER, VB_WELCOME_HUNGUP, VB_WELCOME_IDENTIFICATION_TRANSFER, VB_WELCOME_IDENTIFICATION_HUNGUP, VB_OQ_HUNGUP, VB_NPA500MB, VB_WELCOME_END, VB_WELCOME_INTERNAL_ERROR
Contrato	TARIFAS, PORTABILIDAD, SUSCRIPCION, ALTA, BAJA, CONTRATO, PROMOCIONES, RENUEVO, OTRA_LINEA, CONSUMO, RECARGA
Servicios	INTERNET, PEDIDO, SIM, TV, ROAMING, PREMIUM, CONTESTADOR, SERVICIOS, SMS_MMS, TIENDA, PIN_PUK, TERMINAL
Dinero	DEUDA, FACTURA, RECLAMACION

Tabla 5. Nuevas categorías de la variable categories open question.

Como se puede ver en la *figura 25*, para las nuevas distribuciones de ocurrencias, la nueva categoría problemas físicos es la que más valores tiene, esto es debido a que la variable averías, como se puede ver en la *figura 26*, tiene muchas más ocurrencias que todas las demás variables.

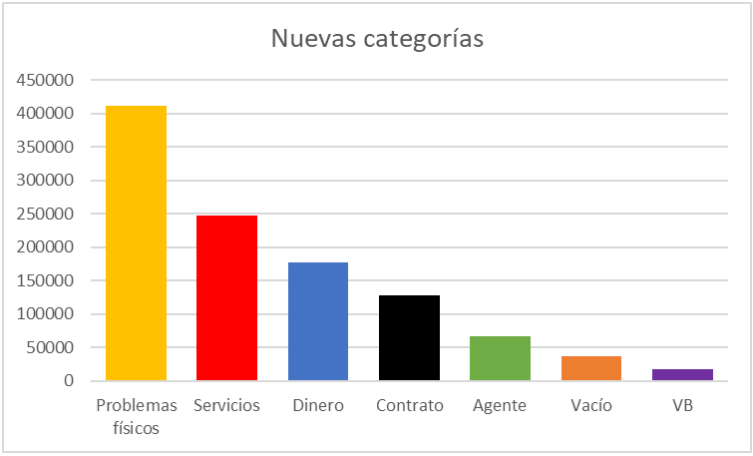

Figura 25. Cantidad de ocurrencias nuevas variables categories open question.

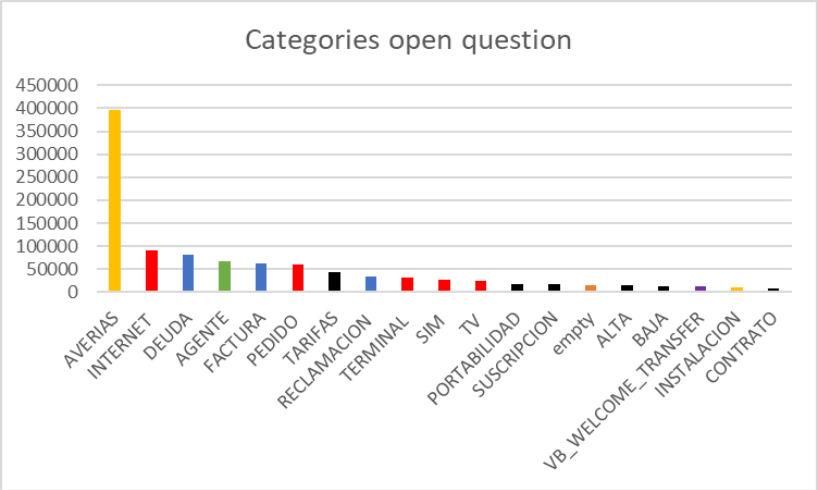

Figura 26.. Cantidad de ocurrencias antiguas variables categories open question

Por lo tanto, tras todo el proceso de limpieza de datos, selección de atributos, ampliación de columnas y transformación de los datos, estos fueron los atributos finales que fueron usados en los diferentes modelos, los cuales se fueron intercambiando, añadiendo o quitando, con el objetivo de conseguir los mejores resultados posibles:

- length_time
- talk_time
- waiting_time
- hold_time
- after_call_time
- ringing_time
- categories_open_question (categórica, 7 valores diferentes)
- brand (categórica, 4 valores diferentes)
- department (categórica, 9 valores diferentes)
- service (categórica, 13 valores diferentes)
- diferente_dep_anterior
- diferente_dep_siguiente
- numero_agentes_antes
- numero_agentes_despues
- despues_YDILO
- posicion_despues_YDILO
- tiempo_antes

- tiempo_despues
- departamento_antes (categórica, 9 valores diferentes)
- departamento_despues (categórica, 9 valores diferentes)
- servicio_antes (categórica, 13 valores diferentes)
- servicio_despues (categórica, 13 valores diferentes)
- diferente_serv_anterior
- diferente_serv_siguiente

Cabe destacar que como ya se ha dicho con anterioridad, cada una las variables categóricas tendrán una columna por cada valor, por lo que las columnas finales con las que los modelos trabajarán serán mayores que las aquí enumeradas.

4 Modelos y resultados

Una vez se ha llevado a cabo la investigación previa y se ha preprocesado el conjunto de datos, el siguiente paso es el de la experimentación mediante modelos y la evaluación de estos resultados. Como ya se ha dicho con anterioridad, el proceso más complicado fue el de la determinación de cuáles eran los atributos más importantes que podían caracterizar los errores de transferencia, por lo que se llevaron a cabo diferentes agrupaciones modificando los atributos de entrada y caracterizando estos para una posterior evaluación con el conjunto de expertos en el Grupo MasMovil.

En este apartado tan solo se mostrarán las técnicas y resultados obtenidos con el conjunto de atributos que se ha considerado más óptimo, ya que el proceso de modelado y caracterización ha sido el mismo en todas las pruebas realizadas.

Durante la realización de los diferentes experimentos se determinó que era más fácil encontrar al agente posterior a cuando se ha producido un error de transferencia que el segmento que se ha equivocado en la redirección, ya que después de este agente posterior siempre debería haber otro segmento de agente de un servicio distinto. Esto se puede ver en la *figura 27*.

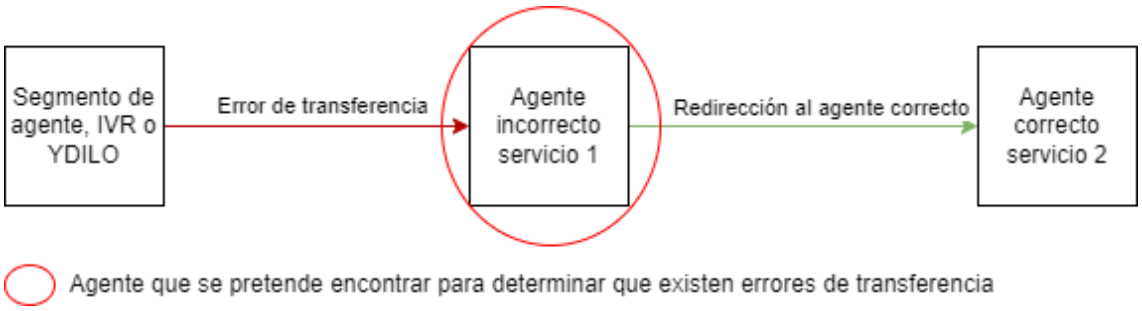

Figura 27. Agente que se pretende encontrar para determinar que existen errores de transferencia.

También se detectó que la información acerca del departamento era mucho menos útil que la de servicio, ya que muchas veces los agentes pasaban a los clientes a otro agente del mismo departamento lo cual en principio no se debería tomar como un error de transferencia ya que el cliente se encuentra en un mismo lugar, pero si esos mismos casos se analizan con la información de servicio, sí que se producía un cambio en este aunque no se cambiase el departamento, lo cual sí que pude indicar que se ha producido un error en la transferencia.

Por ello, para conseguir detectar el segmento de agente incorrecto, que es el posterior cuando se produce un error de transferencia, como se ve en la *figura 27*, los atributos finalmente elegidos son estos 9:

- length_time
- categories_open_question
- service
- numero_agentes_despues
- despues_YDILO
- tiempo_despues
- servicio_antes
- servicio_despues
- diferente_serv_siguiente

Para poder trabajar con estos datos y poder crear modelos con *Python*, primeramente, se pasaron estos desde *Google BigQuery* a *Google Colab*, gracias a la biblioteca de *BigQuery* para *Python*, la cual es capaz de ejecutar una consulta de SQL y devolver los resultados como un *DataFrame* de *Pandas*, que es una herramienta que permite trabajar con tablas de forma sencilla, pudiendo modificarlos valores y formas de estas, así como su formato.

Una vez se tenían los datos cargados en *Colab*, se llevó a cabo todo el proceso de creación de columnas y de transformación del atributo *categories_open_question* descritos en los apartados 3.3 y 3.4 respectivamente de este documento y se guardó toda esta nueva tabla en un archivo CSV (comma separated values) gracias a la función de *Pandas* "to_csv" ya que el proceso de bajar los datos de *BigQuery* es muy lento y consume una gran cantidad de recursos, mientras que los archivos CSV son mucho más livianos y mucho más rápidos para cargar en *Colab*.

También se llevó a cabo todo el resto de los procesos de transformación descritos en el apartado 3.4, aplicando la técnica de *one hot encoding* gracias a la función de *Pandas* "get_dummies" que lleva a cabo todo este proceso de forma rápida, y aplicando el reescalado gracias a la función "MinMaxScaler" de la biblioteca *Sklearn.* Por lo que finalmente el *dataset*, previo al uso de modelos de Inteligencia Artificial, consta de un total de 52 columnas con valores numéricos entre el 0 y el 1.

4.1 Cluistering basado en densidad

Una vez se tiene los datos procesados y preparados se va a llevar a cabo la agrupación de los diferentes segmentos de agente. Para ello y con el objetivo de conseguir los mejores resultados posibles, se utilizaron diferentes técnicas.

Como ya se comentó en el apartado del estado del arte, las técnicas de clusterización basadas en densidad son muy potentes, ya que permiten hacer agrupaciones sin saber el número óptimo de estas y etiquetan como ruido todo lo que no esté cerca de una agrupación.

En este estudio se han usado dos de estos métodos, el DBSCAN y el OPTICS, gracias a las funciones de la librería *Sklearn* de su mismo nombre, las cuales permiten modificar una gran cantidad de parámetros, como el mínimo número de muestras que debe tener un clúster o la métrica usada para calcular las distancias entre las instancias.

Debido a la gran cantidad de instancias y a la limitada memoria RAM de la que se disponía para este proyecto sumado a la gran cantidad de recursos que necesitan estos algoritmos, finalmente no se consiguieron resultados usando estas técnicas.

Al realizar las mismas pruebas disminuyendo el número de instancias sí que se obtuvieron resultados, pero debido a la gran cantidad de columnas que tiene cada instancia, ambos algoritmos clasificaban como ruido a la gran mayoría de las muestras, por lo que, aunque puede ser un método muy útil para otros trabajos con datos no supervisados, se acabó desechando en este estudio.

4.2 Clustering K-medias

El segundo método usado en esta investigación para la agrupación de los diferentes segmentos de agentes fue el clustering mediante K-medias. Como ya se ha explicado en este trabajo, es un método mucho más rápido que los basados en densidad, pero tiene una gran desventaja que es tener que especificar el número de clústeres en los que se van a separar las instancias.

Para el algoritmo de K-medias también se usó una función de la librería *Sklearn* llamada "KMeans" la cual se fue ejecutado de forma reiterada con diferentes valores de clústeres y calculando en cada paso los valores de inercia y distorsión para determinar el número de agrupaciones óptimas gracias al *elbow method*.

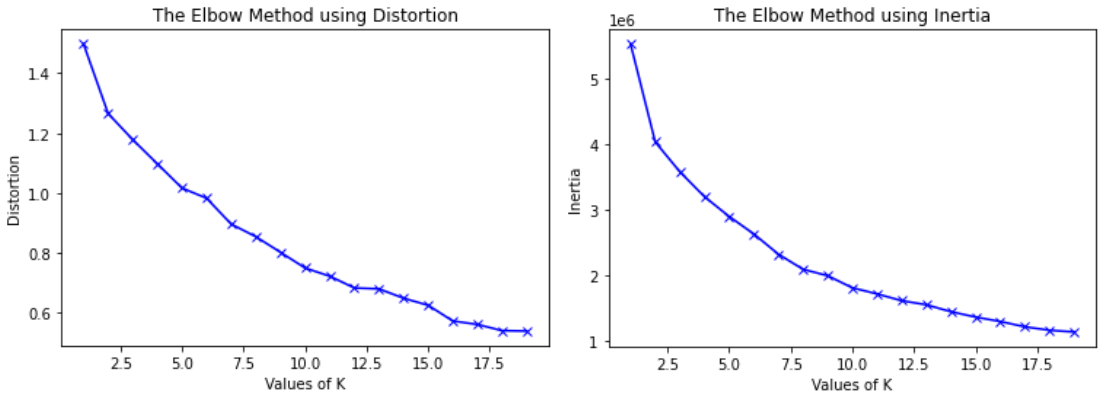

Figura 28. Valores de distorsión e inercia usando el algoritmo k-medias con 2-19 clústeres.

Debido a la dificultad para determinar donde se encuentra el codo o "elbow" en los valores de distorsión e inercia que se pueden visualizar en la *figura 28*, se acabó optando por usar una herramienta de la librería *yellowbrick* llamada "kelbow_visualizer", que como se puede observar en la *figura 29*, es capaz de encontrar con mayor precisión el número óptimo de clústeres usando los valores de distorsión y el índice de Calinski y Harabasz [40].

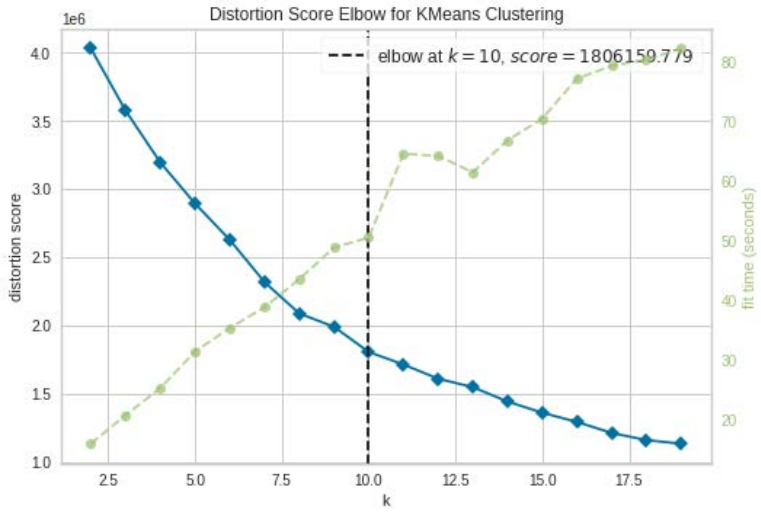

Figura 29. Número óptimo de clústeres para el algoritmo k-medias.

Por lo tanto, una vez determinado el número de clústeres que se van a usar para el algoritmo k-medias se ejecutó este en repetidas ocasiones variando el estado inicial de los centroides con el objetivo de encontrar las distribuciones que menos cambiaban al variar estos, descubriendo que estas apenas diferían al cambiar el estado inicial, lo que indica

que la agrupación es robusta y no está hecha de forma aleatoria. En la *figura 30* se pueden observar el número de instancias que el algoritmo ha clasificado en cada clase.

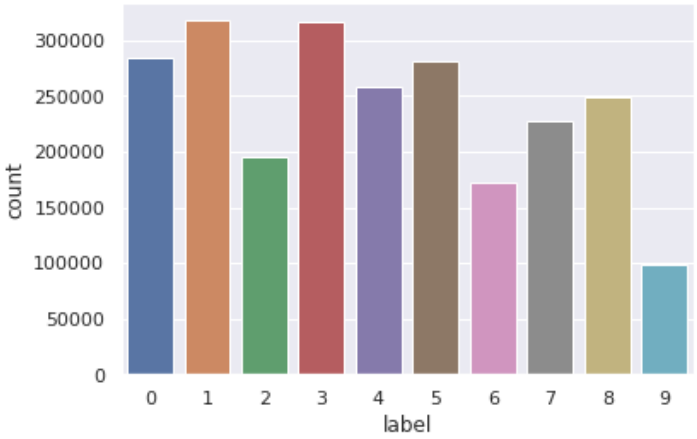

Figura 30. Distribución de instancias por clase.

4.3 Caracterización

Una vez se tienen todos los datos clasificados se tiene que llevar a cabo un estudio que determine la razón de que una instancia pertenezca a una clase u otra. Si el problema en cuestión tan solo tuviese un par de columnas sería de gran facilidad ver de forma gráfica los diferentes clústeres, pero al tener una gran cantidad de variables y que estas sean en la gran mayoría datos no ordinales, dificultan en gran medida la tarea. Es por eso que para caracterizar las agrupaciones se han utilizado árboles de decisión y reglas de asociación usando el algoritmo Apriori.

4.3.1 Árboles de decisión

Como ya se explicó en el apartado de estado del arte los árboles de decisión pueden ser utilizados para la caracterización debido a su gran explicabilidad, ya que no se comportan como cajas negras, si no que se puede ver de forma visual por qué el algoritmo adjudica una clase u otra a cada instancia.

En este caso se ha utilizado un árbol CART usando una función de la librería *Sklearn* llamada "DecisionTreeClassifier". Si se decidiese meter todo el conjunto de datos en este modelo se produciría un sobre entrenamiento ya que el árbol se ajusta a todas las

instancias buscando conseguir un 100% de precisión entre las clases reales y las que el árbol predice, obteniendo una estructura con más de 160 nodos y una profundidad de más de 15 niveles, con lo que este perdería esta cualidad de explicabilidad ya que un árbol tan grande es muy difícil de interpretar y comprender.

Para reducir este sobre entrenamiento y conseguir un árbol más general se ha llevado a cabo un proceso de poda, en este caso usando el método de poda por coste y complejidad mínimo, el cual se basa en los siguientes pasos:

1. Calcular alfa por cada nodo no terminal.
2. Podar los nodos con el valor más pequeño de alfa.
3. Repetir los pasos hasta que solo queda el primer nodo o nodo raíz.

El valor de Alfa se calcula de la siguiente forma:

$$\propto = \frac{R(t) - R(Tt)}{NT - 1} \qquad (4.1)$$

Siendo R(t) la tasa de error si el nodo se recorta, R(Tt) la tasa de error si no se recorta y NT el número de hojas.

Como se puede observar en las *figuras 31* y *32*, al aumentar Alfa, también aumenta la impureza de las hojas ya que se ha producido una mayor poda en el árbol, haciendo que este tenga un menor número de nodos, así como una profundidad también menor.

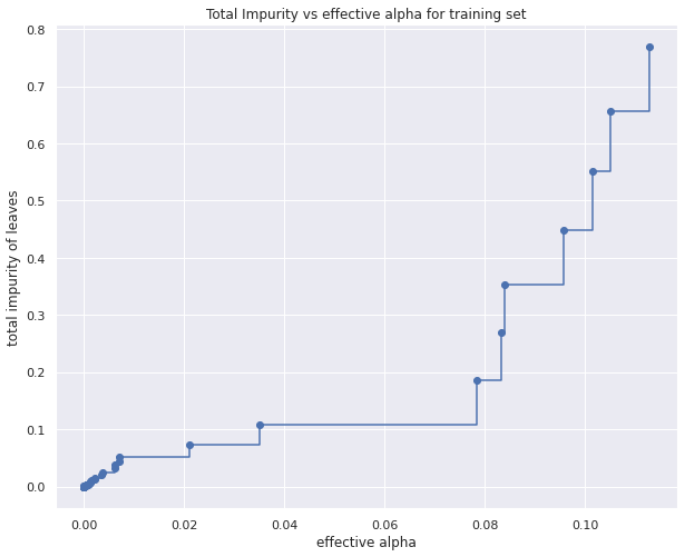

Figura 31. Impureza de las hojas con respecto al valor de alfa.

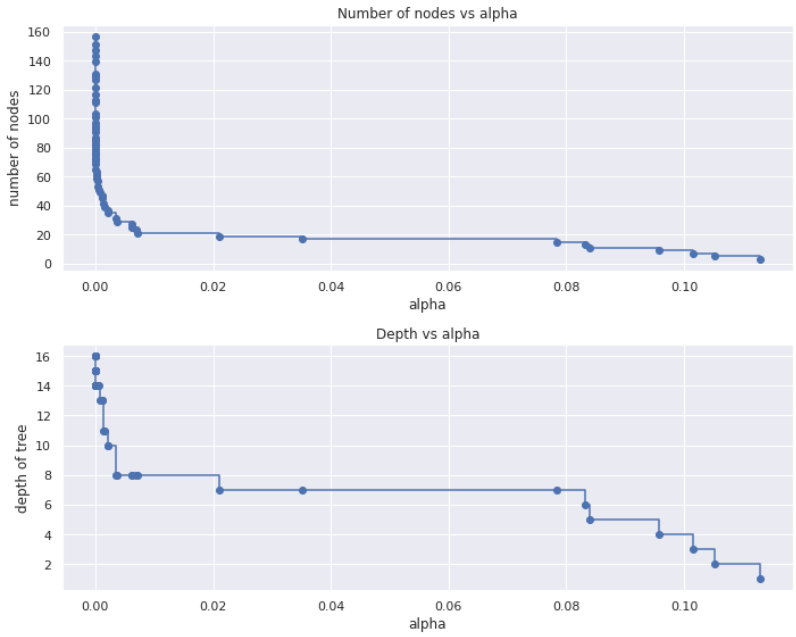

Figura 32. Número de nodos y profundidad del árbol con respeto al valor de alfa.

Para decidir en qué momento se debe parar de podar, antes de empezar el proceso de entrenamiento se separó el conjunto de datos de forma aleatoria en los conjuntos de entrenamiento y testeo, de forma de que el 75% de los datos fueron asignados el primer grupo y el 25% restante al segundo conjunto. El conjunto de entrenamiento sirvió para entrenar el árbol, mientras que el segundo, al ser datos que el modelo no ha visto previamente, sirve para determinar hasta donde se debe podar para evitar el sobre entrenamiento.

Figura 33. Precisión de los conjuntos de entrenamiento y testeo con respecto al valor de alfa.

En este caso, para conseguir la mayor precisión en el conjunto de testeo se usará un Alfa de 0,02, que como se puede ver en la *figura 33*, con este valor se consigue una precisión del 97%.

Así mismo, también se hizo una matriz de confusión que permitiese observar cuales son las clases que el modelo consigue clasificar correctamente y cuales le cuesta más. Como se puede observar en la *figura 34*, todas las clases tienen una precisión superior al 95% excepto el clúster 9, el cual solo alcanza hasta un 78%.

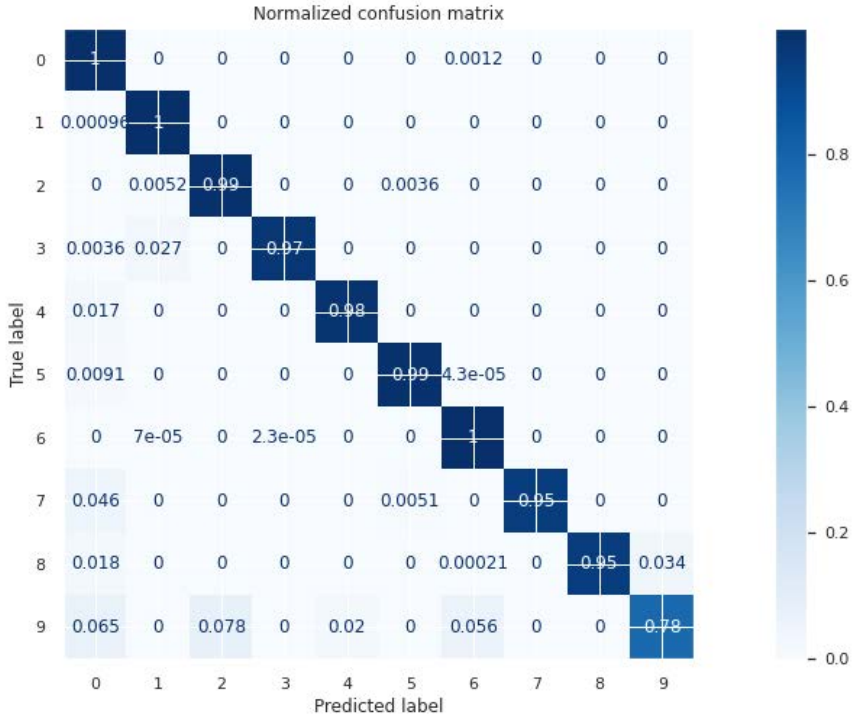

Figura 34. Matriz de confusión normalizada para un valor de alfa de 0,02.

Por último, en este apartado, gracias a la función "export_graphviz" de *Sklearn* se puede ver gráficamente en la *figura 35* el árbol de decisión generado, dando a conocer por qué se le ha asignado una clase u otra a cada segmento de agente. También da la información del número de instancias por clase, así como los valores reales de la clase de esas instancias, por ejemplo, en la clase 1 de las 245.621 muestras que ha asignado a esa clase, 238.211 está bien clasificadas mientras que hay 805 que realmente pertenecen a la clase 2, 6. 604 que pertenecen a la clase 3 y una que pertenece a la clase 6.

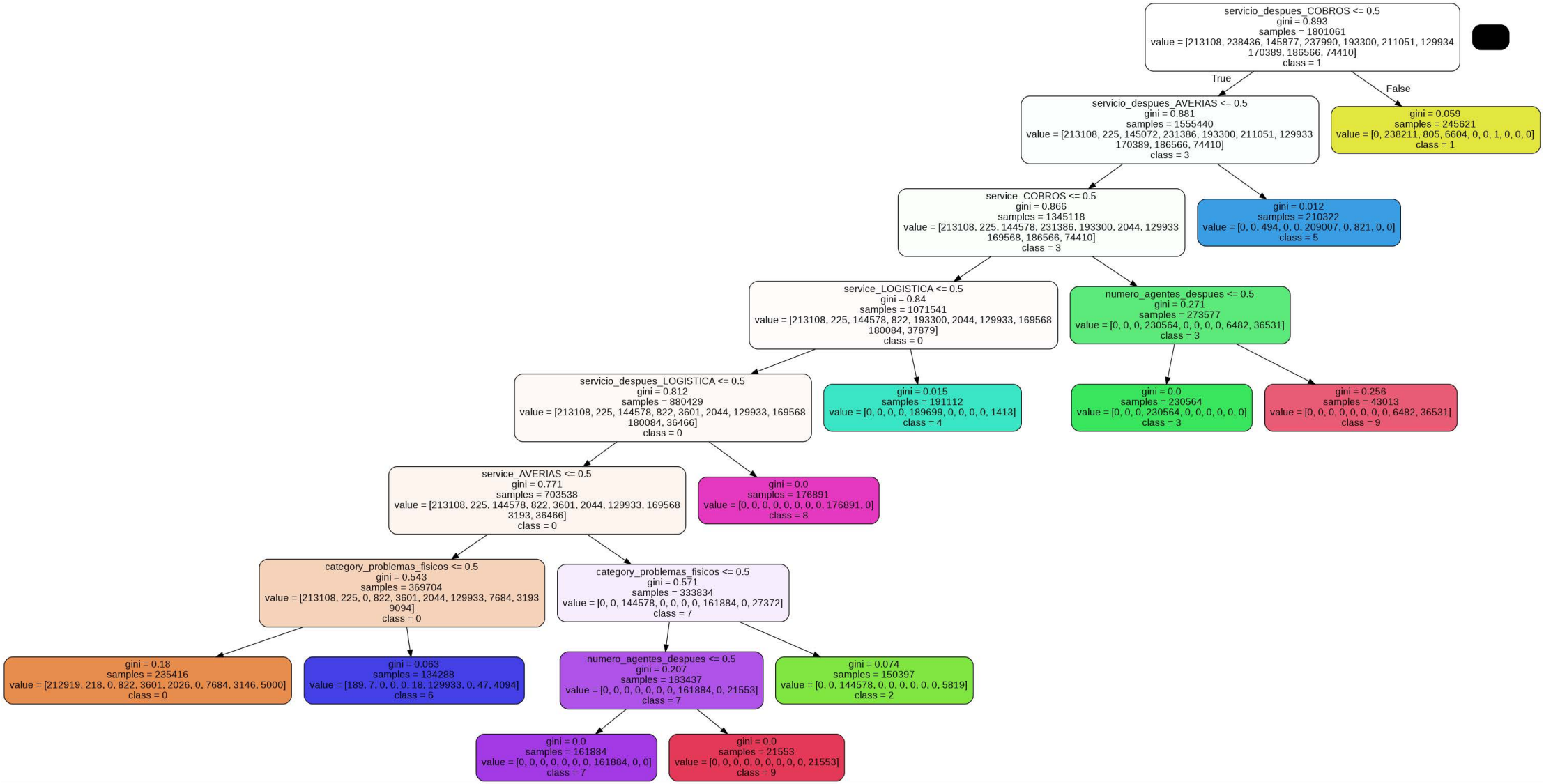

Figura 35. Árbol de decisión generado tras la poda.

Esta es la información que ofrece el árbol de decisión visto en la *figura 35*:

- **Clúster 0 y 6**: Se tiene que estudiar más este clúster, para saber cómo son sus elementos. Se sabe que su servicio no es de cobros ni logística y su servicio después no es ni logística ni averías ni cobros. Se diferencian entre ellos por la categoría de problemas físicos.
- **Clúster 1**: Servicio después cobros.
- **Clúster 2 y 7**: Servicio averías. Se diferencian en la categoría problemas físicos.
- **Clúster 3**: Servicio cobros.
- **Clúster 4**: Servicio logística.
- **Clúster 5**: Servicio después averías.
- **Clúster 8**: Servicio después logística.
- **Clúster 9**: El árbol no tiene mucha precisión con este clúster, ya que hay muchos valores de este que el modelo de árbol de decisión ha estimado incorrectamente y se verá por qué en los siguientes apartados.

4.3.2 Reglas de asociación Apriori

Esta información aportada por el árbol, aunque ofrece un primer acercamiento para entender los diferentes clústeres, no ofrece toda la información necesaria para determinar la existencia de errores de transferencia, por lo que en este apartado se estudiarán estos clústeres de forma más intensiva mediante el uso de reglas de asociación para entender de que están formados cada una de las agrupaciones y qué las diferencia entre ellas.
Para ello se ha usado la función "apyori" de la librería de Python *apriori*, la cual permite modificar los valores de soporte, confianza y empujes mínimos para que esta no devuelva todas las combinaciones posibles entre los atributos.
Para poder usar este algoritmo se tuvieron que hacer también una serie de transformaciones de los datos, ya que cada instancia tiene que aparecer como una lista de atributos como si fuese una lista de la compra en la que se ponen todos los productos/atributos que se van a comprar/que tienen los agentes. Al ser una lista de valores, estos no pueden ser numéricos, ya que las reglas perderían su significado, por lo que la variable número de agentes después se cambió a *agentes_despues* en caso de tener agentes y la variable *length_time* y *tiempo_después* se dividió en tres rangos iguales,

siendo estos bajo, medio y alto, quedando los datos antes del uso del algoritmo de la forma que se puede observar en la *figura 36*.

```
['category_contrato',
 'agentes_despues',
 'diferente_serv_siguiente',
 'service_AVERIAS',
 'servicio_antes_ATENCION NO TECNICA',
 'servicio_despues_ATENCION NO TECNICA',
 'length_time_alto',
 'tiempo_despues_medio'],
['category_problemas_fisicos',
 'agentes_despues',
 'despues_YDILO',
 'diferente_serv_siguiente',
 'service_PROVISION',
 'servicio_antes_ninguno',
 'servicio_despues_ATENCION NO TECNICA',
 'length_time_medio',
 'tiempo_despues_alto'],
```

Figura 36. Dos instancias de datos procesados y listos para ser usados por el algoritmo Apriori.

Una vez se obtuvieron las reglas se produjo un proceso de análisis de estas, concluyendo con los resultados que van a ser mostrados a continuación con respecto a cada clúster.

Clúster 0 y 6

Se van a analizar ambos clústeres juntos ya que son muy parecidos entre ellos, como ya se vio reflejado en el árbol de decisión. Estos conjuntos agrupan a agentes de los servicios de atención no técnica y provisión, los cuales pueden tener un agente después, pero este siempre es del mismo servicio, por lo que no se cumpliría lo estipulado en la *figura 27*, y por lo tanto no se considerarían errores de transferencia.

En cuanto al clúster 0, el 90% de las instancias son del servicio de atención no técnica, de los cuales un 79% no tienen ningún agente después, mientras que el 21% restante sí que tienen agentes después, pero siempre son del mismo servicio, atención no técnica. Estas transferencias ocurren cuando el agente es precedido por la IVR u otro agente, pero nunca cuando van después del YDILO.

El clúster 6 es muy parecido al 0, pero en este siempre viene precedido por el YDILO, y los clientes hablan en este acerca de la categoría de problemas físicos. Tan solo un 6% de los datos tienen un agente después y este siempre es del mismo servicio, de provisión o atención no técnica.

Clúster 2 y 7

Los clústeres 2 y 7 son muy parecidos a los 0 y 6, pero en este caso el servicio de los agentes es averías y no tienen agentes después. Además, también se puede observar en ambos casos un tiempo de duración de segmento más elevado que en los otros clústeres, ya que el 60% de los casos tienen una duración de más de 10 minutos.

Clúster 3

Este clúster agrupa a todos aquellos agentes del servicio de cobros. En este caso tampoco se encuentran agentes después y son precedidos en un 84% de los casos por el servicio de atención no técnica. Además, en el 68% de los casos el tiempo que el cliente está con este agente no supera los 4 minutos, por lo que son segmentos muy cortos.

Clúster 4

En este clúster como también se vio anteriormente en el árbol de decisión, se encuentran los agentes del servicio de logística, los cuales en el 97% de los casos no cuentan con ningún agente después. Además, en el 78% de los casos preceden del servicio de atención no técnica y nunca tienen como predecesor al YDILO ni a la IVR, sino que siempre tiene que haber un agente antes.

Todos estos clústeres anteriores al no tener agentes después de diferente servicio, no se llevó a cabo un mayor análisis de ellos, ya que no son tan importantes para el estudio como sí que lo son los siguientes que se van a mencionar. Por lo que en este documento solo se han mencionado brevemente algunas de sus características más importantes.

Los siguientes clústeres se diferencian del resto porque son agentes que siempre tienen otro agente después el cual siempre es de un servicio diferente, lo que podría significar que se ha producido un error de transferencia previo.

Clúster 1

El clúster 1 se compone de aquellos agentes que siempre tienen un agente después de diferente servicio, siendo en este caso este servicio después el de cobros. Además, el 87% no tiene ningún servicio antes, por lo que el error de transferencia en caso de haberlo sería

de la IVR o del YDILO. Un 44% va después del YDILO de los cuales más del 90% de estos hablan en la pregunta abierta cosas relacionadas con el dinero. También son segmentos con reducido *length time*, así como con reducido tiempo después hablando con otros agentes.

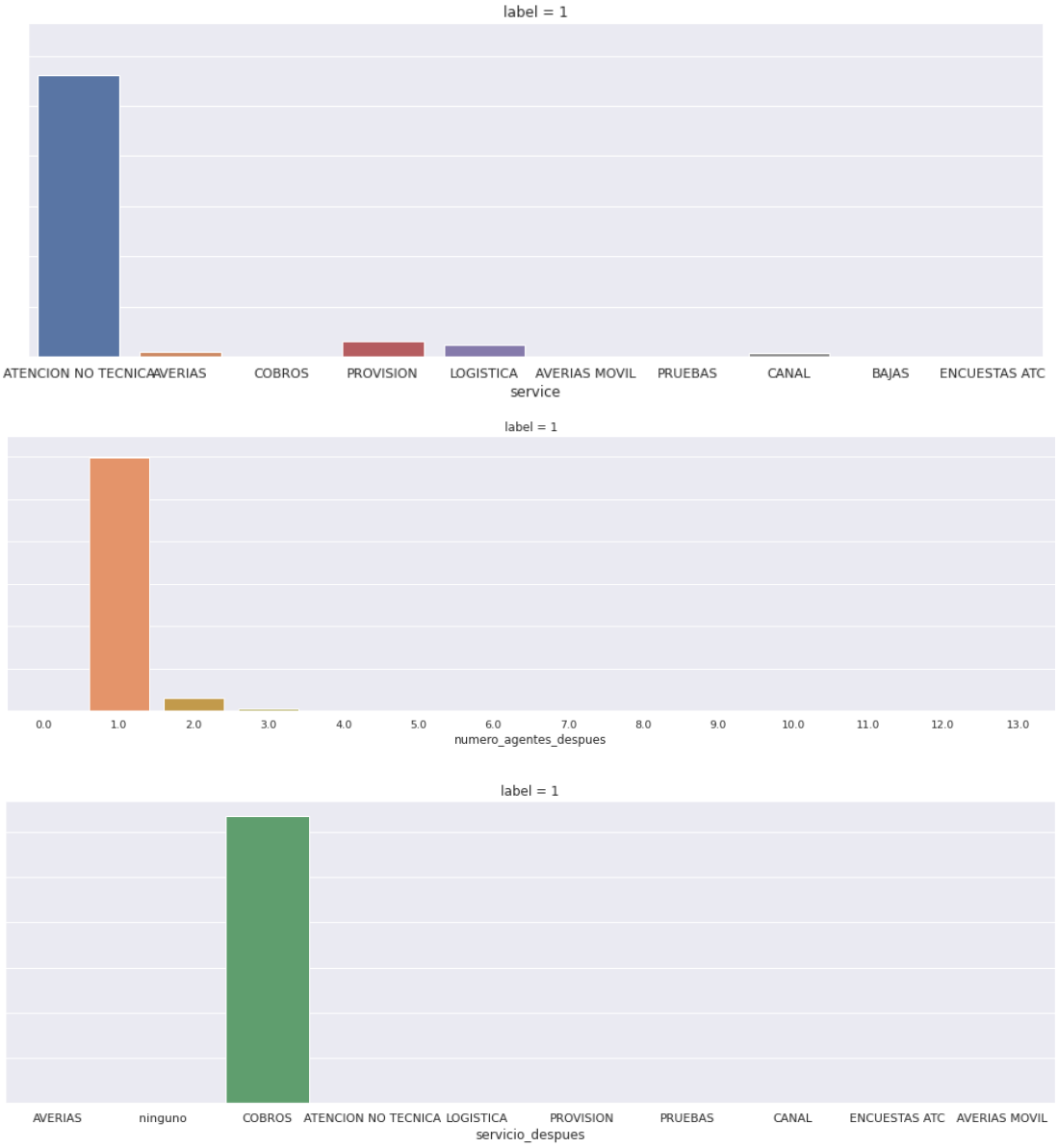

Figura 37. Distribuciones de los atributos de servicio, número de agentes después y servicio después para el clúster 1.

Este caso no parece que sea un error de transferencia, sino más bien un problema de eficiencia del servicio de *call center*, ya que siempre que los clientes tienen que dirigirse al servicio de cobros el sistema les manda primero al servicio de atención no técnica. Es posible que la reducción total de estos casos sea de gran dificultado para la empresa, pero

al menos el sistema debería ser capaz de dirigir correctamente al 40% de los clientes ,que son los que hablaron sobre cosas relacionadas con el dinero en el YDILO, ya que si nos fijamos en la *figura 38*, la mayoría de los clientes que hablan de dinero se les pasa primero con los agentes del clúster 1 y después con los del 3, pudiendo eliminar este primer paso, y ahorrando a los clientes tiempo de espera en colas y hablando con agentes.

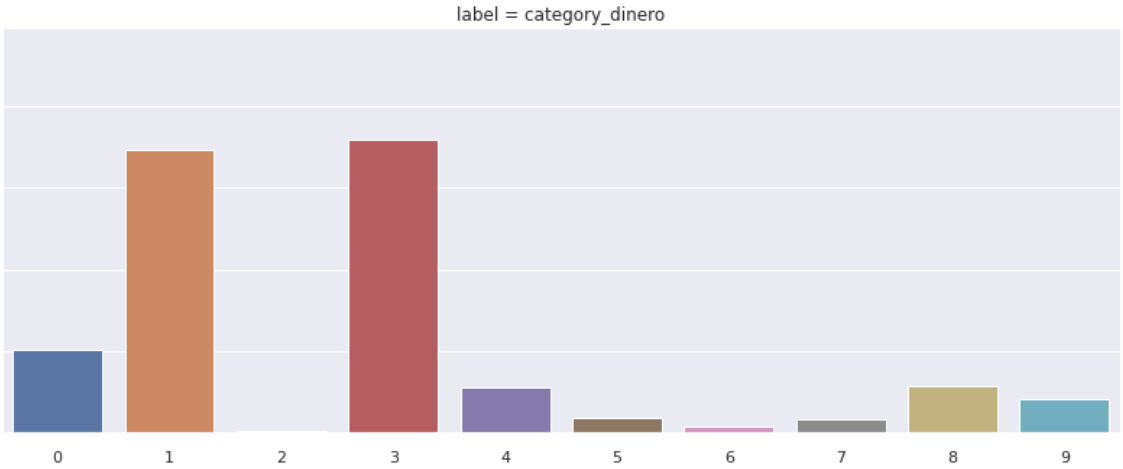

Figura 38. Distribución de la variable category_dinero entre los diferentes clústeres.

Clúster 5

Como se puede observar en la *figura 39*, el clúster 5 es muy parecido al 1, ya que son agentes mayoritariamente del servicio de atención no técnica, que siempre tienen un agente después, pero en este caso ese agente es del servicio de averías y no del de cobros. Este al igual que en el caso anterior parece un problema de eficiencia, ya que siempre que se pretende ir al servicio de averías, primero hay que pasar por atención no técnica.
En el 95% de los casos no hay ningún servicio antes y el 31% va después del YDILO.
Además, como también se puede ver en la *figura 39*, un 19% de los casos son del servicio de provisión, los cuales siempre tienen como predecesor al YDILO y en este los clientes siempre hablan sobre cosas relacionadas con problemas físicos.
Este clúster también tiene muy poco *length time*, pero mucho tiempo después, ya que como se ha visto antes el servicio de averías es el que más tiempo requiere de conversación entre cliente y agente.

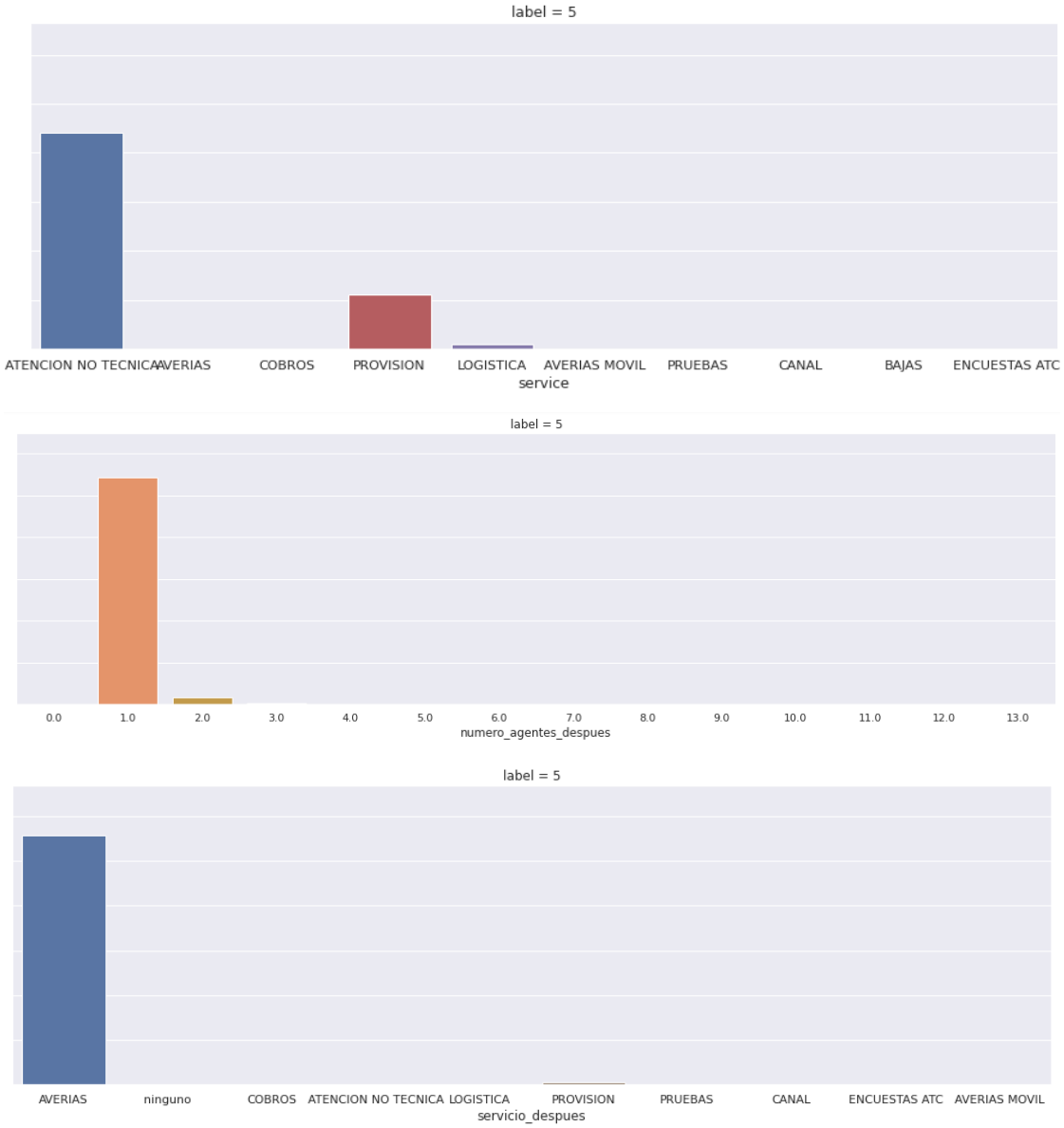

Figura 39. Distribuciones de los atributos de servicio, número de agentes después y servicio después para el clúster 5.

Clúster 8: Servicio después logística.

El clúster 8 sigue la misma tendencia de problema de eficiencia que los otros dos anteriores, ya que este también pertenece a atención no técnica, tiene un agente después, pero este pertenece al servicio de logística.

Como en los casos anteriores siempre que se pretende llegar a este servicio se tiene que pasar primero por el servicio de atención no técnica. En este caso también cuenta con un 95% de las instancias que no tienen agentes antes, un 49% de YDILO y un 25% de categoría de servicios en el atributo categories open question.

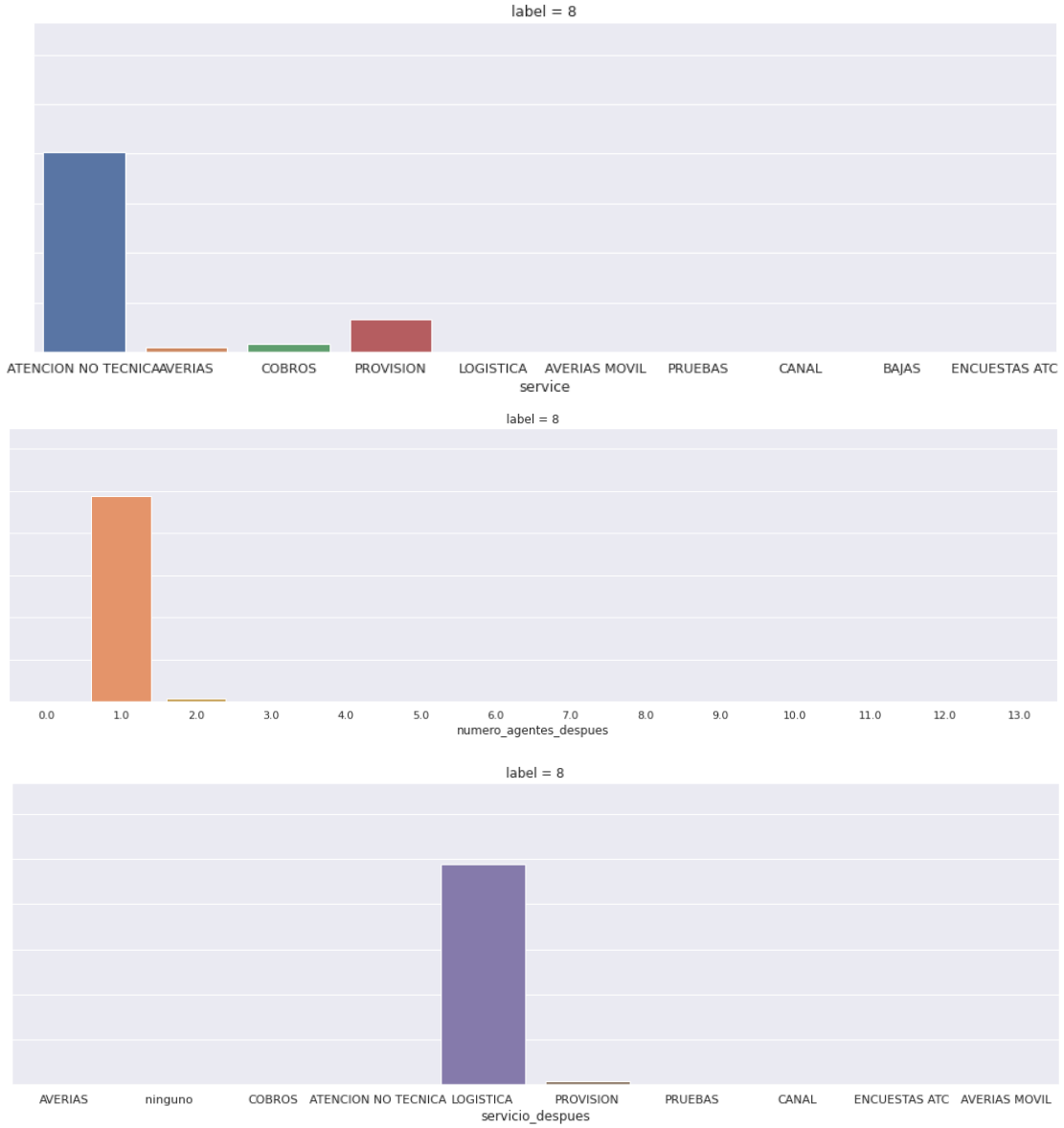

Figura 40. Distribuciones de los atributos de servicio, número de agentes después y servicio después para el clúster 8.

Clúster 9

Como se puede observar en la *figura 41*, este clúster no es igual a los mencionados anteriormente, ya que, aunque siempre tiene un agente después y este siempre es de un servicio diferente, no son agentes del servicio de atención no técnica que siempre pasen la llamada a otro servicio.

En el 50% de los casos la llamada es transferida a atención no técnica, ya que es el servicio más común, el cual como ya se ha visto antes sirve como nexo antes de poder llegar a otros servicios.

Este clúster, el cual como se pude observar en la *figura 27*, es el que menos ocurrencias tiene y en el que menos se parecen las instancias entre sí, ya que se agrupan aquellos agentes que siempre tienen un servicio diferente después pero que no es de atención no técnica a cobros, averías o logística.

Por lo que se podría considerar y etiquetar a estos como errores de transferencia, los cuales son ocasionados por la IVR en un 44% de los casos, en un 26% por el YDILO y en un 30% por un agente anterior.

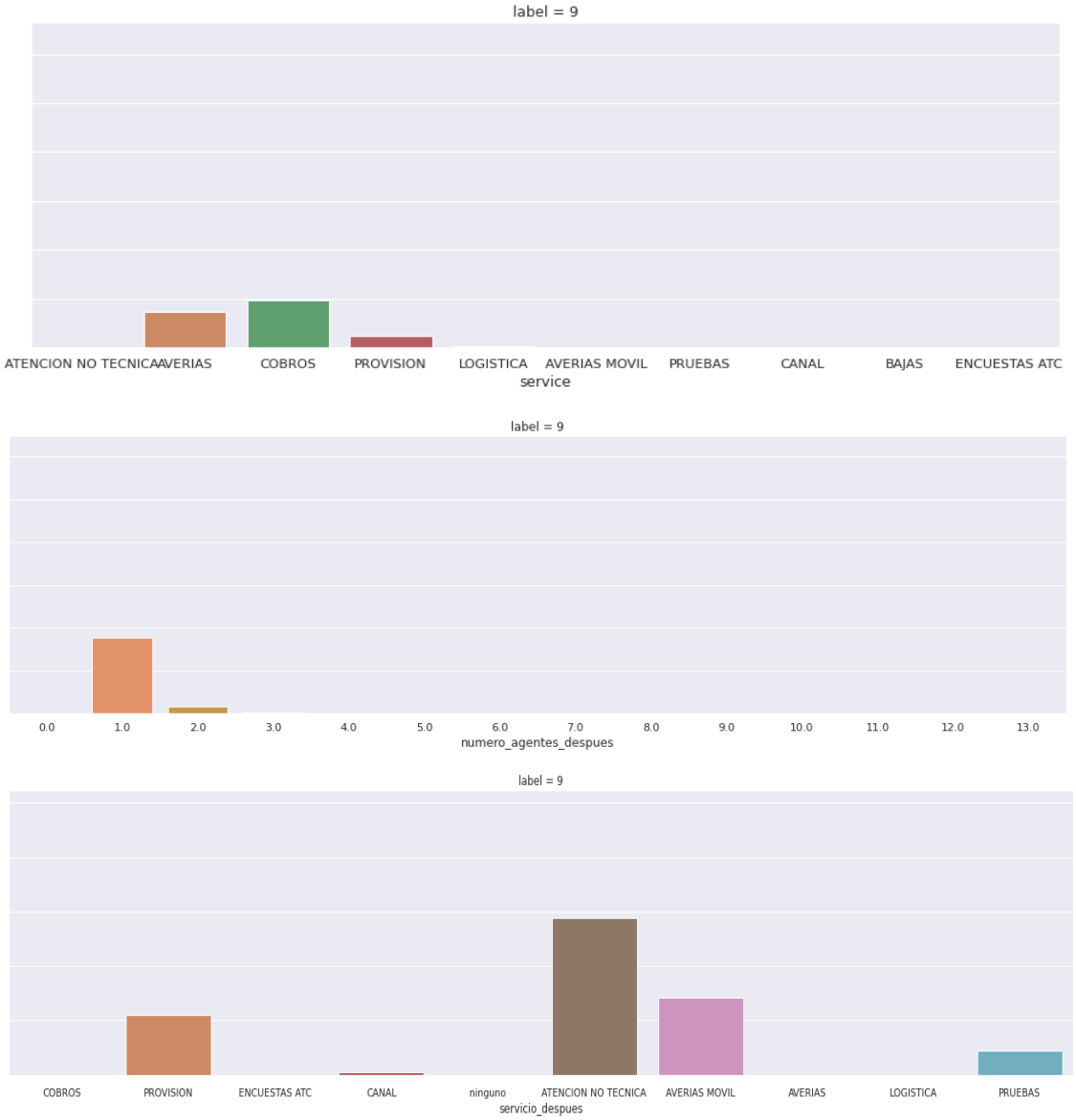

Figura 41. Distribuciones de los atributos de servicio, número de agentes después y servicio después para el clúster 9.

Además, como se puede ver en la *figura 42*, se pueden apreciar las diferencias entre los clústeres 1, 5, 8 y 9 con respecto al resto en cuanto a tiempos de duración del segmento,

lo que indicaría que son segmentos no necesarios en los que el agente redirige al cliente con otro agente, ya que este no debería estar allí. También se puede observar que el clúster 3 también tiene tiempos muy reducidos, pero esto se debe a que el servicio de cobros es el más eficiente y no a que sea un segmento inservible.

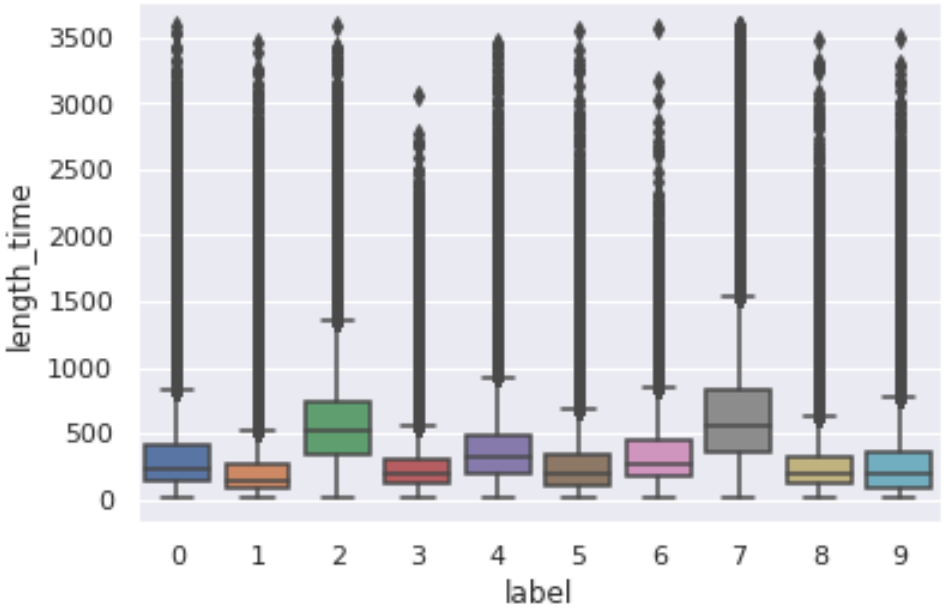

Figura 42. Comparación mediante Boxplot de los tiempos de segmento entre los diferentes clústeres.

4.4 Predicción y evitar errores de transferencia

Una vez se han determinado que existen errores de transferencia, para terminar este trabajo se va a intentar determinar por qué se producen y si se podrían evitar pasando del agente previo al agente después saltándose los agentes del clúster 9 como se puede ver en la *figura 43*.

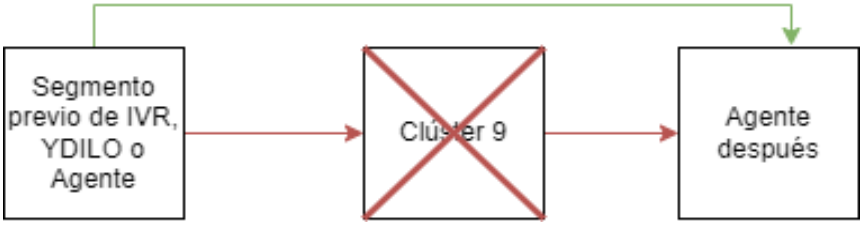

Figura 43. Flujo correcto para evitar errores de transferencia.

Debido a los pocos datos acerca de la IVR y el YDILO con los que se cuenta, ya que de la IVR no se han ofrecido datos acerca de las decisiones que toma el cliente y por qué se

le manda a un agente o a otro y del YDILO tampoco se tiene las conversaciones reales que ha mantenido el cliente, no ha sido posible determinar por qué se producen estos 70% de errores de transferencia que se han agrupado en el clúster 9.

Para el 30% restante que sí que tiene un agente antes se ha intentado descubrir si estos agentes anteriores tienen algo en común que los diferencie frente al resto y que permita identificarlos para prevenir estos errores de transferencia.

Debido a los pocos casos en los que se producen estos errores, los cuales tan solo son 23.462 agentes de los más de 2 millones que se obtuvieron tras el preprocesado, nos encontramos ante un gran desbalanceo entre los agentes que se equivocan en las transferencias y los que no, por lo que para que el modelo de clasificación no clasifique siempre las muestras como no error, ya que obtendría una mayor precisión, se llevó a cabo un proceso de "undersampling" o submuestreo reduciendo la clase que no cometen errores a 23 mil instancias para que sea igual en cantidad a la que cometen errores.

Tras esto, se entrenó un modelo de árbol de decisión con poda de igual forma que la explicada en el apartado 4.3, obteniendo los resultados que se pueden observar en la *figura 44*.

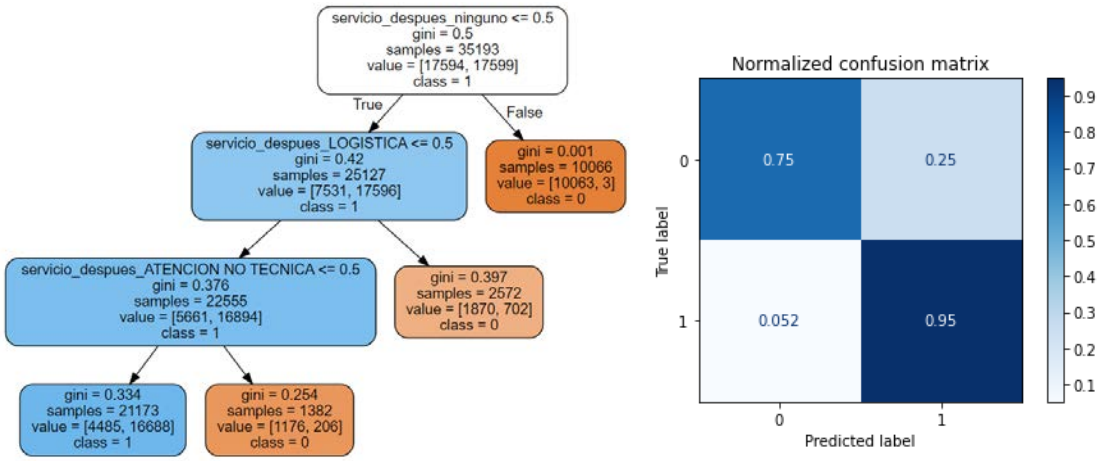

Figura 44. Árbol y matriz de confusión para determinar que tiene en común los agentes que producen errores de transferencia (class = 1)

Al igual que con la IVR y el YDILO, con los agentes anteriores es muy difícil determinar qué es lo que ha pasado y ha desencadenado una transferencia errónea debido a los datos tan limitados con los que se cuenta, ya que no son los mejores para esta parte del estudio. Este árbol tan solo es capaz de explicar que si el agente transfiere la llamada a los servicios de provisión o atención no técnica no se producen nunca errores de transferencia, mientras que si es a otro servicio sí que se pueden llegar a dar estos errores.

5 Conclusiones y trabajo futuro

Como ya se ha explicado en este documento, el objetivo de esta investigación era determinar si en la red del *call center* del Grupo MasMovil existían errores de transferencia, o lo que es lo mismo, poder determinar si hay llamadas en las que a los clientes se les manda con agentes de cuyo departamento o servicio no son los adecuados para resolver sus consultas.

Para ello se han usado técnicas de agrupamiento de datos no supervisados usando tanto algoritmos basados en densidad como k-medias, dando como resultado diez grupos diferentes de agentes. Debido a la complejidad de evaluación y comprensión de dichas agrupaciones, se han utilizado otras técnicas de aprendizaje automático entre las que se encuentran el uso de árboles de decisión con poda y las reglas de asociación usando el algoritmo *Apriori*. Estos modelos pretendían no solo determinar si realmente existen características en lo datos que puedan afirmar la existencia de estos errores de transferencia, sino que también poder determinar cuáles y si realmente son una gran amenaza para la compañía.

Tras el estudio de los diferentes modelos y los diversos resultados obtenidos se llegó a la conclusión de que estos errores sí que existen, pero aparecen en menos del 1% de todas las llamadas entre las fechas del 1 de enero del 2019 y el 1 de junio del 2020, ya que son menos de 100.000 con respecto al total de más de 16 millones que tienen al menos un agente. Por lo que será decisión de la empresa determinar si los gastos necesarios para poder arreglar dichos errores y por ende conseguir un mejor servicio y satisfacción del cliente son rentables o si por el contrario son errores asumibles para sus estándares de calidad.

Pero además de conseguir el objetivo de encontrar estos errores también se detectaron ciertas ineficacias en ciertas llamadas en las que los clientes necesitaban hablar con los departamentos de cobros, logística y averías, ya que, para llegar a dichos servicios, primeramente, los clientes tienen que pasar por el agente de atención no técnica, con su respectiva pérdida de tiempo esperando en el ACD a ser atendido y hablando con un agente con el que no se necesita hablar. Estos se corresponden con más de 850.000 casos los cuales podrían tener una solución más sencilla mejorando las redirecciones de la IVR y el YDILO.

Esta información se ha aportado a los responsables de la empresa, la cual ayudará a aumentar el conocimiento acerca del funcionamiento de su *call center* y, en base a esta información, determinar las medidas a tomar para su optimización.

Trabajo futuro

Gracias a que este estudio ha permitido identificar los errores de transferencia, en caso de que la compañía determinase que es beneficioso la corrección de estos, el siguiente paso sería determinar las causas que hacen que ocurran. Para esta nueva investigación sería necesaria la recolección de nuevos datos que permitiesen una mayor comprensión de todo lo que ha pasado en los segmentos de IVR, YDILO y Agentes y como ha afectado esto en la aparición de dichos errores.

Estos datos podrían estar compuestos por las conversaciones en lenguaje natural entre los clientes y los agentes para determinar si el cliente se ha expresado mal o si ha sido un error humano del agente, las respuestas dadas al YDILO y a la IVR que permitan determinar cómo estas afectan a las redirecciones, entre otros muchos parámetros que los *call centers* son capaces de recoger y almacenar actualmente. Para esto sería necesario el uso de técnicas de PLN (procesadores de lenguaje natural) que generasen nuevas variables que usar en diferentes modelos que permitiesen predecir cuándo y por qué se producen estos errores. Además, debido al escaso número de segmentos en los que ocurren estos errores también sería necesario un mayor estudio en técnicas de *undersampling*, *oversampling* u otros modelos que fuesen eficaces con conjuntos de datos con un claro desbalanceo.

Dichos datos también podrían ser útiles para poder determinar cómo se podría modificar la IVR y el YDILO, así como su direccionamiento con el objetivo de eliminar las ineficiencias para poder llegar a los servicios de cobros, logística y averías de forma más rápida.

También sería interesante poder obtener los datos acerca de la satisfacción del cliente en la llamada que permitiesen determinar si estos errores e ineficiencias realmente causan una mala experiencia e inconformidad, por lo que se tendrían que dedicar recursos a eliminarlos, o si bien arreglar estos no va a afectar a la forma de pensar que tienen los clientes acerca de la empresa.

Este trabajo tan solo es un primer paso que abre diferentes ramas de trabajos futuros, pero tendrá que ser la empresa la responsable de determinar qué es lo que más le conviene a la hora de actuar una vez tengan este informe con los diferentes resultados expuestos.

Referencias

[1] Cosmos call center (2019). “La empresa de Call Center; historia y evolución”. [En línea]. Disponible en: https://cosmoscallcenter.com/customer-care/la-empresa-de-call-center-historia-y-evolucion. Último acceso: 29/03/2022.

[2] Zendesk (2021). “Historia del call center: tecnología de ayer y de hoy”. [En línea]. Disponible en: https://www.zendesk.com.mx/blog/call-center-historia/. Último acceso: 29/03/2022.

[3] Call Centre Helper (2022). “The History of the Call Centre”. [En línea]. Disponible en: https://www.callcentrehelper.com/the-history-of-the-call-centre-15085.htm.Último acceso: 29/03/2022.

[4] McKinsey & Company (2020). “The state of AI in 2020”. [En línea]. Disponible en: https://www.mckinsey.com/business-functions/mckinsey-analytics/our-insights/global-survey-the-state-of-ai-in-2020.Último acceso: 29/03/2022.

[5] Boletín Oficial del Estado (1978). “Constitución Española”. [En línea]. Disponible en: https://www.boe.es/eli/es/c/1978/12/27/(1)/con. Último acceso: 30/03/2022.

[6] Boletín Oficial del Estado (2018). “Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales”. [En línea]. Disponible en: https://www.boe.es/eli/es/lo/2018/12/05/3/con. Último acceso: 30/03/2022.

[7] Parlamento Europeo y Consejo de la Unión Europea (2016). “Reglamento Europeo de Protección de Datos descrito en el Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016”. [En línea]. Disponible en: https://www.boe.es/doue/2016/119/L00001-00088.pdf. Último acceso: 30/03/2022.

[8] Comisión Europea (2021). “New rules for Artificial Intelligence”. [En línea]. Disponible en: https://ec.europa.eu/commission/presscorner/detail/en/QANDA_21_1683#1. Último acceso: 04/04/2022.

[9] Rollins, J (2015). “Foundational Methodology for Data Science”.

[10] Piatetsky, G (2014). “CRISP-DM, still the top methodology for analytics, data mining, or data science projects”. [En línea]. Disponible en: https://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html. Último acceso: 28/04/2022.

[11] Datascience-pm (2021). “What is CRISP DM?”. [En línea]. Disponible en: https://www.datascience-pm.com/crisp-dm-2/. Último acceso: 04/06/2022.

[12] N. Gans, G. Koole y A. Mandelbaum (2003), “Telephone Call Centres: Tutorial, Review, and Research Prospects”. Manufacturing and Service Operations Management Journal, Vol. 5, pp 79-141.

[13] O. Z. Aksin, M. Armony y V. Mehrotra (2007), “The Modern Call Centre: A Multi-Disciplinary Perspective on Operations Management Research”. Production and Operations Management Journal, 16(6), pp 665-688.

[14] V. Hlupic, and G. J. D. Vreede (2005), “Business Process Modelling Using Discrete-Event Simulation: Current Opportunities and Future Challenges”. International Journal of Simulation and Process Modelling, 1(1-2), pp 72 – 81.

[15] Sinaeepourfard, A. y Mohamed Hussain, H. (2011). “Comparison of VoIP and PSTN services by statistical analysis”. 2011 IEEE Student Conference on Research and Development, pp. 459–461.

[16] Akhtar, S., & Latif, M. (2010). Exploiting simulation for call centre optimization. WCE 2010 - World Congress on Engineering 2010, Vol. 3, pp. 2112–2117.

[17] Daqar, M. A. M. A., y Smoudy, A. K. A. (2019). “The role of artificial intelligence on enhancing customer experience”. International Review of Management and Marketing, 9(4), 22–31.

[18] Marshall, P. (2011). “Artificial Intelligence Can “smart” machines replace humans?”. CQ Press, 21(16), 361–384.

[19] Keon, Y., Kim, H., Choi, J. Y., Kim, D., Kim, S. Y., y Kim, S. (2018). “Call Center Call Count Prediction Model by Machine Learning”. Journal of advanced information technology and convergence, 8(1), 31–42.

[20] Bromuri, S., Henkel, A. P., Iren, D., y Urovi, V. (2020). “Using AI to predict service agent stress from emotion patterns in service interactions”. Journal of Service Management, 32(4), 581–611.

[21] M. Bussing, K. Nichols, T. Doll y S. Johnson. (2018) “A Machine Learning Approach to Topic and Sentiment”. Colorado School of Mines, Department of Computer Science, p. 9, 2018.

[22] Shuang, K., Ding, K. Z., Liu, X. H., & Wen, X. L. (2017). “A two stage classification model for call center purchase prediction”. Telkomnika (Telecommunication Computing Electronics and Control), 15(1), 351–356.

[23] Williams, A (2015). “Why is clustering hard?”. [En línea]. Disponible en: http://alexhwilliams.info/itsneuronalblog/2015/09/11/clustering1/. Último acceso: 12/04/2022.

[24] Sinaga, K. P., y Yang, M. S. (2020). “Unsupervised K-means clustering algorithm”. IEEE Access, 8, 80716–80727.

[25] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., y Wu, A. Y. (2002). “An efficient k-means clustering algorithms: Analysis and implementation”. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(7), 881–892.

[26] Chi, E y Lange, K (2014). “Splitting Methods for Convex Clustering”. Journal of Computational and Graphical Statistics, 24:4, 994-1013.

[27] Chen, E (2015). “Infinite Mixture Models with Nonparametric Bayes and the Dirichlet Process”. [En línea]. Disponible en: http://blog.echen.me/2012/03/20/infinite-mixture-models-with-nonparametric-bayes-and-the-dirichlet-process/. Último acceso: 12/04/2022.

[28] Ester, M., Kriegel, H.-P., Sander, J., y Xu, X. (1996). “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (pp. 226–231).

[29] Ankerst, M., Breunig, M. M., Kriegel, H. P., y Sander, J. (1999). “OPTICS: Ordering Points to Identify the Clustering Structure”. SIGMOD Record (ACM Special Interest Group on Management of Data), 28(2), 49–60.

[30] ArcGIS (2021). “Density-based Clustering”. [En línea]. Disponible en: https://pro.arcgis.com/en/pro-app/2.8/tool-reference/spatial-statistics/densitybasedclustering.htm. Último acceso: 12/04/2022.

[31] GeeksforGeeks (2021). “Elbow Method for optimal value of k in KMeans”. [En línea]. Disponible en: https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/. Último acceso: 13/04/2022.

[32] Aznar, P (2020). “Decision Trees: Gini vs Entropy”. [En línea]. Disponible en: https://quantdare.com/decision-trees-gini-vs-entropy/. Último acceso: 13/04/2022.

[33] Gil Martínez, C (2020). “Reglas de Asociación”. [En línea]. Disponible en: https://rpubs.com/Cristina_Gil/Reglas_Asociacionhttps://quantdare.com/decision-trees-gini-vs-entropy/. Último acceso: 13/04/2022.

[34] Pejić Bach, M., Pivar, J., y Jaković, B. (2021). "Churn Management in Telecommunications: Hybrid Approach Using Cluster Analysis and Decision Trees". Journal of Risk and Financial Management, 14(11), 544.

[35] Burgel, P. R., Lemonnier, L., Dehillotte, C., Sykes, J., Stanojevic, S., Stephenson, A. L., y Paillasseur, J. L. (2019). "Cluster and CART analyses identify large subgroups of adults with cystic fibrosis at low risk of 10-year death". European Respiratory Journal, 53(3).

[36] Shigetoh, H., Koga, M., Tanaka, Y., y Morioka, S. (2020). "Central sensitivity is associated with poor recovery of pain: Prediction, cluster, and decision tree analyses". Pain Research and Management, 2020.

[37] Wu, X., Benjamin Zhan, F., Zhang, K. y Deng, Q. (2016). "Application of a two-step cluster analysis and the Apriori algorithm to classify the deformation states of two typical colluvial landslides in the Three Gorges, China". Environmental Earth Sciences, 75(2), 1–16.

[38] Dharshinni, N. P., Azmi, F., Fawwaz, I., Husein, A. M. y Siregar, S. D. (2019). "Analysis of Accuracy K-Means and Apriori Algorithms for Patient Data Clusters". Journal of Physics: Conference Series (Vol. 1230). Institute of Physics Publishing.

[39] Anaconda (2021). "State of Data Science 2021". [En línea]. Disponible en: https://www.anaconda.com/state-of-data-science-2021. Último acceso: 05/04/2022.

[40] T. Caliński y J Harabasz (1974). "A dendrite method for cluster analysis". Communications in Statistics, 3:1, 1-27.

Apéndice

Planificación

En esta sección se describirá como ha sido la planificación que se ha llevado a cabo para la realización de esta investigación. La duración del proyecto se estipuló bajo un contrato remunerado con la empresa MasMovil de 8 meses, pactando una jornada laboral de 12 horas semanales.

Como ya se ha explicado en el capítulo de 1.4 de metodología y organización, se ha utilizado el método Crisp-DM como base para determinar cuáles son los pasos necesarios para conseguir buenos resultados en un proyecto de ciencia de datos y en este apartado se van a especificar las tareas y tiempos previstos para el cumplimiento de cada una de estas etapas:

1. **Comprensión del negocio**: en esta primera tarea la empresa expone la situación actual, así como los problemas y objetivos que se pretenden cumplir. En esta también se llevará a cabo un proceso de investigación de la empresa, así como del funcionamiento de un *call center* y las regulaciones que aplican a esta materia, que permitan determinar cuáles con los beneficios e impacto que este proyecto puede tener en el futuro. También se planificará cuáles son los pasos y tiempo que se asignará a cada tarea.
2. **Comprensión de los datos**: el siguiente mes y medio, y tras obtener los datos de parte del departamento de datos del Grupo MasMovil, se llevará a cabo un proceso de entendimiento de las tablas aportadas.
3. **Preparación de los datos**: los siguientes dos meses se llevará a cabo un proceso de limpieza de datos, así como una investigación de trabajos anteriores relacionados con el objetivo de realizar las transformaciones en los datos necesarias para poder trabajar en el posterior modelado. Tras los procesos de modelado y evaluación se volverá a realizar una preparación de datos cuyo objetivo sea el de la obtención de unos mejores resultados finales.
4. **Modelado y evaluación**: tras el proceso anterior, se podrá plantear la solución determinando los modelos más adecuados para el caso en cuestión. Será la tarea más larga, ya que contará con un proceso continuo de modelado y posterior

evaluación con expertos en este conjunto de datos para la obtención de los mejores y más útiles resultados.

5. **Despliegue**: este último paso cosiste en la extracción de conclusiones con el objetivo de presentar al departamento de datos del Grupo MasMovil todo el trabajo que se ha realizado y como puede aportar este para la mejora en los servicios que ofrecen. El último mes estará destinado en exclusiva a la realización de esta tarea.

Tarea	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5	Mes 6	Mes 7	Mes 8
Comprensión del negocio								
Comprensión de los datos								
Preparación de los datos								
Modelado								
Evaluación								
Despliegue								

Tabla 6. Diagrama de Gantt de la planificación del proyecto.

Presupuesto

Teniendo en cuenta la planificación de 8 meses que es lo que dura el contrato de la investigación, el presupuesto requerido estará determinado por los salarios de los diferentes trabajadores que han formado parte del mismo y los gastos en hardware y software necesarios para la realización de este proyecto.

En cuanto a los salarios, consumen gran cantidad del presupuesto total necesario, ya que se necesita de varios profesionales. Para la realización de la mayor parte del proyecto se necesita de un científico de datos, el cual es el encargado del procesado de los datos y la realización de los diferentes modelos. Este cuenta con una jornada laboral de 12 horas semanales y 8 meses de contrato acordados con el Grupo MasMovil. Además de este, también será necesaria la actuación de otros trabajadores cuyos cometidos son los de la obtención y transformación de los datos en tablas, los de validación de los resultados y la supervisión de trabajo realizado, siendo estos, el jefe de investigación y el analista de datos. La cantidad de horas estimadas de trabajo de estos será inferior, ya que el trabajo ha sido asignado al científico de datos, mientras que los otros dos actúan como profesionales de apoyo.

Puesto	**Salario bruto por hora**	**Días estimados de trabajo**	**Horas estimadas**	**Coste**
Jefe de investigación	35€/hora	15 días	15 días * 8 horas por día laborable = 120 horas	**4.200€**
Analista de datos	25€/hora	20 días	20 días * 8 horas por día laborable = 160 horas	**4.000€**
Científico de datos	15 €/hora	8 meses * 22 días laborables al mes = 176 días	176 días * 2.5 horas laborables al día = 440 horas	**6.600€**
Coste total salarios				**14.800€**

Tabla 7. Coste total salarios.

A estos costes hay que sumarle el equipo necesario. En cuanto al hardware, la compañía tiene una serie de portátiles HP EliteBook 840 g5, valorados actualmente en unos 575€.

Estimando como vida útil del producto unos 5 años, la amortización sería de unos 10€ mensuales, que será el coste asociado a la utilización de este hardware. Por lo tanto, habría que multiplicar dicho coste por los 8 meses más los 25 días de los otros dos profesionales asignados a esta investigación

Por último, se encontrarían los costes asociados a la compra de las licencias para el uso de los sistemas de *Google Cloud*. Dentro de estos están el uso de *BigQuery*, el almacenamiento en la nube de *Google* y el almacenamiento ilimitado en los servicios de *drive*.

Elementos	**Coste mensual**	**Coste 8 meses y 25 días**
Salarios	-	14.800€
HP EliteBook 840 g5	10€	90€
Servicios de Google	150€	1.350€
Coste total		**16.240€**

Tabla 8. Coste total proyecto.