

Tipo de documento: Tesis de maestría

Master in Management + Analytics

Modelos de intervención preventiva de experiencias conversacionales negativas en servicios de atención al cliente

Autoría: González Giménez, Martín Cruz
Año académico: 2023

¿Cómo citar este trabajo?

González Giménez, M. (2023) "Modelos de intervención preventiva de experiencias conversacionales negativas en servicios de atención al cliente". [*Tesis de maestría. Universidad Torcuato Di Tella*].

Repositorio Digital Universidad Torcuato Di Tella

<https://repositorio.utdt.edu/handle/20.500.13098/12441>

El presente documento se encuentra alojado en el Repositorio Digital de la Universidad Torcuato Di Tella bajo una licencia Creative Commons Atribución-No Comercial-Compartir Igual 4.0 Argentina (CC BY-NC-SA 4.0 AR)
Dirección: <https://repositorio.utdt.edu>

UNIVERSIDAD TORCUATO DI TELLA, ESCUELA DE NEGOCIOS

MASTER IN MANAGEMENT + ANALYTICS

Modelos de intervención preventiva de experiencias conversacionales negativas en servicios de atención al cliente

Tesis de maestría

Alumno: González Giménez, Martín Cruz

Director: Gálvez, Ramiro H.

Fecha de entrega: 15 de octubre de 2023

Resumen

Los canales conversacionales se volvieron uno de los medios más elegidos por empresas y usuarios para relacionarse. Muchas empresas buscan ofrecer parte de sus servicios y su atención al cliente mediante estos canales. Esto se debe a que disminuyen la fricción en adopción para el usuario y son potencialmente más intuitivos para navegar que canales tradicionales como sitios web o aplicaciones. Sin embargo, lograr adaptar la experiencia ofrecida durante el transcurso de las conversaciones para maximizar la satisfacción de los usuarios, es un desafío que en la mayoría de estas empresas no se está tratando.

Bajo este contexto, Bleett es una startup que diseña, implementa y desarrolla soluciones en el canal conversacional (chatbots) para este tipo de empresas. El objetivo de este trabajo es intentar encontrar un modelo que permita clasificar experiencias conversacionales negativas, utilizando la información disponible dentro de los primeros intercambios de cualquier conversación, para rescatarlas mediante la participación de operadores humanos y así mejorar la experiencia de usuario que ofrece el canal. Se utilizarán los datos de las conversaciones de un cliente Bleett, que se mantendrá anónimo, para perseguir este objetivo.

Los resultados obtenidos sugieren que dicha estrategia de rescate de conversaciones es posible y justifican su implementación a modo de evaluación para validar el impacto real que podría tener en el nivel de satisfacción del usuario. En este trabajo se analizan datos de conversaciones de atención al cliente reales para validar la naturaleza de la propuesta. Se presenta en detalle las decisiones metodológicas detrás de los modelos desarrollados, se explican las transformaciones realizadas sobre la estructura de datos para generar los features utilizados por los modelos, se evalúa la performance de los mismos y finalmente se discuten como aplicar los modelos entrenados dentro de la solución y posibles mejoras para futuras iteraciones.

Abstract

Conversational channels have become one of the preferred means for companies and users to engage one another. Many businesses aim to deliver some of their services and customer support through these channels. This is because they allow them to reduce friction in user adoption and are potentially more intuitive to navigate than traditional channels such as websites or applications. However, achieving the adaptation of the user experience throughout conversations to maximize user satisfaction is a challenge that, in most of these companies, is not being addressed.

In this context, Bleett is a startup that designs, implements, and develops solutions in the conversational channel (chatbots) for these types of companies. The objective of this study is to attempt to find a model that enables the classification of negative conversational experiences, utilizing the information available within the 'initial' exchanges of any conversation, in order to 'rescue' them through the involvement of human operators and thus improve the user experience offered by the channel. Data from conversations from one of Bleett's clients, who will remain anonymous, will be used to pursue this goal.

The results obtained suggest that such a 'rescue' strategy for conversations is feasible and justify its implementation, at least as a trial, to evaluate the real impact it could have on user satisfaction levels. In this study, real customer service conversation data is analyzed to validate the proposed approach's nature. The methodological decisions behind the developed models are detailed, explaining the transformations made to the data structure to generate the features used by the models. The performance of the models is evaluated, and finally, considerations are made on how to apply the trained models within the solution, along with potential improvements for future iterations.

1 - Introducción.....	4
1.1 - Contexto.....	4
1.2 - Problema y Solución.....	5
2 - Datos.....	6
2.1 - Dominio.....	6
2.2 - Estructura.....	8
2.2.1 - Base principal.....	8
2.2.1.1 - Sesión.....	8
2.2.1.2 - Descripción de los campos:.....	8
2.2.2 - Base con calificaciones.....	10
2.2.3 - Formas de entender o leer los datos:.....	10
2.3 - Posible data leakage.....	14
3 - Metodología.....	16
3.1 - Modelos.....	16
3.1.1 - Algoritmo.....	16
3.1.1 - Modelos implementados.....	18
3.1.2 - Variable objetivo.....	19
3.2 - Estrategia de validación.....	20
3.2.1 - Métrica de evaluación de modelos.....	20
3.2.2 - Conjuntos de entrenamiento, validación y testeo.....	20
3.3 - Definición de datos de trabajo.....	22
3.3.1 - Selección de conversaciones.....	22
3.3.2 - Ingeniería de atributos.....	29
3.3.2.1 - Bolsa de intenciones.....	29
3.3.2.2 - Bolsa de palabras.....	31
3.3.2.3 - Información histórica de usuarios.....	31
3.4 - Optimización de hiperparámetros.....	32
4 - Análisis exploratorio de los datos.....	33
4.1 - Distribución de los datos.....	33
4.2 - Comparación entre chatbot y operadores humanos.....	36
4.3 - Relación entre largo de la sesión y calificación.....	38
4.4 - Relación entre intenciones y calificación.....	42
5 - Resultados.....	49
5.1 - Performance.....	49
5.5.1 - Modelos e hiperparámetros.....	49
5.5.2 - Performance en testeo.....	51
5.2 - Importancia de features.....	53
5.3 - Interpretación.....	61
6 - Conclusiones.....	64
6.1 - Limitaciones y posibles mejoras.....	65
6.2 - Aplicaciones prácticas.....	67
6.3 - Conclusión.....	68
7 - Bibliografía.....	70

1 - Introducción

1.1 - Contexto

En la actualidad, muchas empresas optan por utilizar canales conversacionales como una herramienta complementaria e incluso alternativa a las plataformas tradicionales para interactuar y ofrecer servicios a sus clientes, como sitios web, aplicaciones móviles, correos electrónicos, entre otros. Los canales conversacionales se destacan por su naturaleza intuitiva y facilidad de navegación en comparación con muchas páginas web y aplicaciones. Además, ofrecen una ventaja significativa en términos de adopción, ya que los usuarios generalmente no están limitados por una plataforma o software específico. Esto permite a las empresas establecer un contacto directo con sus clientes, superando la barrera de la instalación de una nueva aplicación en los dispositivos de estos últimos. En lugar de eso, aprovechan plataformas que ya están instaladas en los dispositivos de los usuarios, como *WhatsApp*, *Google Messages*, mensajes directos en *Instagram*, entre otras. De esta manera, las empresas pueden proporcionar una experiencia conversacional con los mismos objetivos que persiguen en otras plataformas, con un costo reducido en términos de rediseño e implementación.

Bajo este contexto, se han realizado considerables esfuerzos para automatizar y optimizar las conversaciones que las empresas pueden ofrecer a sus usuarios. Sin embargo, en general, la mayor parte del trabajo en inteligencia artificial y aprendizaje automático se ha centrado en interpretar y responder de manera más efectiva a mensajes de los usuarios, lo cual es aplicable en la implementación y el funcionamiento de los chatbots. En los últimos años se dieron grandes avances en esta área, como es el caso de chat GPT (Brown et al., 2020). Mientras tanto, el estudio de la relación entre el rendimiento de estos servicios y las métricas de satisfacción, como C-sat y NPS, no ha recibido el mismo nivel de atención y profundidad. Existen análisis de sentimiento sobre texto escrito por usuarios, inicialmente concentrados en reseñas o comentarios, ya aplicados a conversaciones orientadas a servicio al cliente (Feine et al., 2019). Además, en el mundo de la atención telefónica, que cuenta con mayor trayectoria, se pueden encontrar ejemplos de estudios que utilicen aprendizaje automático sobre transcritos

de conversaciones entre operadores y usuarios para predecir los niveles de satisfacción reportados por los usuarios (Bockhorst et al., 2017).

1.2 - Problema y Solución

Bleett es una startup que diseña, implementa y desarrolla soluciones en el canal conversacional (chatbots) para diferentes empresas. Este trabajo se concentra en el servicio que ofrece una empresa de seguros, que permanecerá anónima, mediante el canal conversacional que gestiona Bleett. Un desafío para Bleett es que, dada la variabilidad de los usuarios que utilizan el canal, es difícil diseñar una única experiencia (o abanico de opciones) que maximice la satisfacción para todos. Especialmente porque está en el usuario la carga de navegar las opciones que ofrece el chatbot y navegar la experiencia.

Una posible solución a este problema que explorará este trabajo es desarrollar la capacidad de identificar de forma temprana conversaciones que serían experiencias de usuario negativas, para intervenirlas y rescatarlas de forma preventiva. La naturaleza de la intervención aquí propuesta, es mediante la participación de operadores humanos que tienen una mayor capacidad de resolutivez que el chatbot¹. Para lograr discriminar las conversaciones que deben ser intervenidas se propone entrenar un modelo de aprendizaje automático supervisado que categorice conversaciones, únicamente utilizando información de las primeras interacciones² entre el usuario y el bot³. El alcance del trabajo es definir, entrenar y evaluar posibles modelos que demanda la solución propuesta, concluyendo con recomendaciones para la implementación de la solución y posibles próximos pasos a seguir.

A continuación se resume la estructura del resto de este documento. En la [Sección 2](#) se presentarán los datos desarrollando su dominio y estructura. En la [Sección 3](#) se

¹ Esta declaración está respaldada en un análisis desarrollado en la sección de [4 - Análisis exploratorio de los datos](#) de este trabajo.

² El proceso de definición del momento de decisión e intervención o punto de corte está desarrollado en el transcurso del trabajo y representa una parte considerable del ejercicio. Por lo tanto no puede ser resumido en esta introducción.

³ Es la primera vez que se realiza un ejercicio de este tipo con estos datos, por lo tanto gran parte del trabajo consiste en probar la existencia de una señal entre los datos disponibles. No se encontró bibliografía semejante (más allá de lo mencionado en la sección 1.1 Contexto) que busque intervenir conversaciones dentro del contexto de atención vía chatbot. De todas maneras considerando el objetivo propuesto, cualquier método que permita probar la existencia de dicha señal sería satisfactorio hasta cierto punto.

brindará una descripción y justificación de la metodología que se utilizará para generar y evaluar los modelos. En la [Sección 4](#) se realizará un análisis descriptivo de los datos para entender en mayor detalle el problema que enfrenta el modelo y también explorar la validez de la solución propuesta en relación a la experiencia de los usuarios. En la [Sección 5](#) se mostrarán los resultados de los modelos con sus performance e interpretación. Finalmente, en la [Sección 6](#) se presentarán las conclusiones del trabajo, sus limitaciones y posibles futuras mejoras. Además en la [Sección 6](#) se encontrarán recomendaciones para la posible implementación de la solución planteada.

2 - Datos

2.1 - Dominio

Los datos trabajados provienen de las interacciones gestionadas por el servicio de atención al público que ofrece cierta compañía de seguros. Dicha compañía brinda este servicio en un formato conversacional mediante su propio chatbot disponible tanto en whatsapp como en la misma página web de la compañía a través de una ventana de webchat presente en todo el dominio.

El registro de estas conversaciones es guardado por la plataforma con la que se implementa el chatbot y se disponibiliza mediante una API. Dicha API devuelve un csv con toda la información relevante para las conversaciones. Estos csv son subidos al data lake de la compañía en Google Cloud Platform (GCP) donde pueden ser trabajados como tablas SQL utilizando el servicio de *Bigquery* dentro de GCP.

Este chatbot actúa en representación de la compañía frente a los usuarios que se comunican por alguno de sus dos canales conversacionales, whatsapp o el webchat disponible en la página web de la compañía. La naturaleza de las conversaciones puede variar dependiendo de lo que esté buscando el usuario. Los usuarios que son clientes pueden realizar diferentes acciones en el canal relativas a los productos que tienen contratados como denunciar un siniestro, solicitar auxilio mecánico, consultar su próxima fecha de pago y solicitar una copia de su póliza o tarjeta de circulación entre otras. Los usuarios no clientes pueden encontrar en el canal información sobre los productos de la compañía e incluso cotizar una cobertura proporcionando sus datos en

el canal. Todas estas funcionalidades son atendidas automáticamente por el bot sin necesidad de intervención humana. Aún así, existe en el canal la posibilidad de contactar un operador humano si el usuario no pudiera resolver su consulta con las funcionalidades que ofrece el bot. En diferentes puntos de la conversación, o como respuesta a mensajes específicos del usuario, el chatbot deriva al usuario para ser atendido por un operador humano dentro del mismo canal. Durante la derivación el bot le informa al usuario que va a ser transferido con un representante de la compañía y lo acompaña en la espera hasta que un operador disponible atienda el chat presentándose. El servicio en el canal es transparente frente al usuario en términos de quién está hablando en el chat, distinguiendo cuando el que habla es el bot o un operador en cuyo caso este se presentará con su propio nombre. En términos generales, aproximadamente el 75% de las conversaciones son atendidas exclusivamente por el bot y el 25% restante tienen alguna intervención de operadores humanos.

Existen dos indicadores de calidad para las conversaciones que ocurren en el canal, ambos se obtienen preguntando al usuario que califique su experiencia sobre el final de las conversaciones. La diferencia entre ambos indicadores es en relación a qué parte del servicio se le pide al usuario que se evalúe, puede ser la experiencia relativa a los operadores humanos o la experiencia con el chatbot. Una conversación no suele tener ambos tipos de calificaciones ya que por regla de negocio si una conversación es derivada a un operador humano, la pregunta de satisfacción que se le hace al usuario es relativa al operador humano. Solo se pregunta la satisfacción con la atención brindada por el chatbot en conversaciones donde no interviene un humano. El caso de la pregunta de satisfacción sobre el chatbot ocurre en el final de las conversaciones cuando el usuario expresa que ya no tiene más consultas o pasado cierto tiempo de inactividad, automáticamente se pregunta al usuario que califique su atención del 1 al 10. Estas calificaciones son el principal indicador de la calidad de las atenciones con las que cuenta la compañía. En este trabajo se utilizan únicamente los puntajes otorgados al chatbot como label que se intenta predecir.

2.2 - Estructura

2.2.1 - Base principal

La estructura básica de la tabla principal en *Bigquery* tiene como registros/filas cada mensaje individual que pasa por el canal. Los features o columnas de la tabla contienen información relativa al mensaje. Los features relevantes para este proyecto son: *id*, *creation_time*, *msg_from*, *message*, *rule_name*, *session_id*, *customer_id*, *session_creation_time*, *message_number*.

2.2.1.1 - Sesión

Hasta este punto se habló de “conversaciones” que ocurren dentro del canal. Una conversación en un chat tiene un significado intuitivo y por eso se utilizó este término. Pero es necesario aclarar que a partir de aquí también se utilizará el término sesión para referirse al mismo concepto, y a continuación se brindará una explicación más específica de cómo se define exactamente este concepto de sesión o conversación dentro del contexto del proyecto y la base de datos.

2.2.1.2 - Descripción de los campos:

- *id*: Identificador del mensaje

Es una secuencia alfanumérica, única para cada mensaje..

- *creation_time*: Fecha y hora del mensaje.

Es el timestamp correspondiente al momento de envío de cada mensaje.

- *msg_from*: Autor del mensaje

Puede tomar 3 valores, bot, user u operator. Indica a cual de esas 3 categorías pertenece el emisor del mensaje.

- *message*: Mensaje

Es un campo de texto con el contenido de los mensajes. El campo incluye texto y emojis. En el caso de mensajes con contenido multimedia como imágenes, videos, archivos y notas de voz se los reemplaza con el texto *_file_*.

- *rule_name*: Intención del Bot

Es el título de la intención a la que responde el bot en los mensajes para los cuales aplica (solo se encuentra disponible para mensajes enviados por el bot). La intención se refiere a las respuestas predefinidas que puede ejecutar el bot. Se pueden disparar como respuesta a palabras específicas que escribe el usuario, seleccionadas desde la implementación; por ejemplo, cuando el usuario dice “hola” se ejecuta la intención “Saludo”. O por la lógica secuencial del bot, definida desde la implementación; por ejemplo, primero se ejecuta la intención “Mensaje Inicial” y se envía el mensaje correspondiente, luego se ejecuta la intención “Menú Principal” y se envía el mensaje con las opciones del menú en forma de botones y dependiendo del botón seleccionado por el usuario se ejecutará la intención siguiente para enviar el mensaje correspondiente.

- *customer_id*: Identificador de Usuario

Es una secuencia alfanumérica única para cada usuario que habla con el bot, sigue la misma lógica que el número de teléfono en chats de whatsapp con la ventaja de que es anónimo y de que, también, se puede usar en otros canales conversacionales que no utilizan número de teléfono, como el webchat. Todos los mensajes de las conversaciones de un mismo usuario tendrán el mismo valor en el campo *customer_id* sin importar si son enviados por el usuario mismo, el bot o el operador.

- *session_creation_time*: Fecha y hora de inicio de sesión

Es el timestamp correspondiente al momento de envío del primer mensaje de cada conversación. La definición de primer mensaje y de sesión depende del tiempo entre mensajes con el mismo *customer_id*. Los mensajes consecutivos del mismo *customer_id* con menos de 8 horas entre uno y otro forman parte de la misma sesión⁴ y tienen el mismo *session_creation_time*. Cuando pasan más de 8 horas entre dos mensajes con el mismo *customer_id*, esas sesiones se separan y el primer mensaje, pasadas las 8 horas, será el primer mensaje de la próxima sesión. El *session_creation_time* de esta nueva

⁴ Por regla de negocio de Bleett se determinaron ventanas de tiempo de al menos 8 horas como delimitadores entre sesiones. Esta decisión se basa en la inspección de conversaciones aleatorias buscando encontrar separaciones naturales entre conversaciones. Bleett encontró que en general conjuntos de mensajes con más de 8 horas de intervalo suelen estructurarse como conversaciones independientes. Entonces, dicha regla está definida arbitrariamente pero basada en evidencia observacional.

sesión es *timestamp del creation_time* de este mensaje. Todos los mensajes que corresponden a una misma sesión tienen el mismo valor en el campo *session_creation_time*.

- *session_id*: Identificador de Sesión

Es una cadena alfanumérica única para cada sesión, se genera concatenando el *customer_id* del usuario a quien le corresponde la sesión y el *session_creation_time* de la misma. Todos los mensajes que corresponden a una misma sesión tienen el mismo valor en el campo *session_id*.

- *message_number*: Número de mensaje

Es el número que indica el orden de los mensajes dentro de una misma sesión. Los mensajes con el mismo *session_id* se ordenan por *creation_time* y el *message_number* es el contador que hace explícito ese orden. El primer mensaje de cada sesión toma el valor 1 y los mensajes siguientes van sumando de a una unidad al contador.

2.2.2 - Base con calificaciones

Existe una tabla auxiliar que también es relevante en este proyecto, la misma contiene las respuestas de los usuarios frente a la pregunta de satisfacción como un campo específico asociado a cada sesión. Cada registro en esta tabla es una sesión y los features relevantes son dos: *session_id* y *score*. El score es la respuesta del usuario a la pregunta de satisfacción del tipo: “¿Del 1 al 10 cómo calificarías tu atención?”

2.2.3 - Formas de entender o leer los datos:

Con esta estructura se pueden recrear y leer las conversaciones en su totalidad filtrando o agrupando por *session_id* y ordenando los mensajes por *creation_time*. En la [Tabla 1](#) se puede ver un ejemplo de una porción de conversación recreada de esta manera.

Tabla 1: Ejemplo de lectura de conversación desde la base de datos

session_id	session_creation_time	message_number	creation_time	rule_name	msg_from	message
-------------------	------------------------------	-----------------------	----------------------	------------------	-----------------	----------------

		ber ⁵				
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	1	2020-02-03 10:50:37	Inicio Webchat	bot	¡Hola! Soy el asistente virtual
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	2	2020-02-03 10:50:37	Inicio Webchat	bot	Estos son algunos de los temas más frecuentes de atención al cliente:
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	3	2020-02-03 10:51:42		user	Hola
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	4	2020-02-03 10:51:42	Saludo bienvenida	bot	¡Hola ! ¿En qué puedo ayudarte?
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	5	2020-02-03 10:51:48		user	Botón -> Menú
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	6	2020-02-03 10:51:48	Menu	bot	Podes ver las opciones más abajo 🖱️ o preguntarme lo que quieras saber:
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	7	2020-02-03 10:51:48	Menu	bot	
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	8	2020-02-03 10:51:56		user	Botón -> Cotizar auto
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	9	2020-02-03 10:51:56	Cotizar seguro vehículo	bot	Buenísimo. Necesitamos algunos datos tuyos y de tu vehículo para cotizar el mejor valor de la cuota de tu seguro.
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	10	2020-02-03 10:51:56	Preguntar nombre y apellido	bot	1- ¿Cómo te llamas?
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	11	2020-02-03 10:52:04		user	Jesse
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	12	2020-02-03 10:52:04	Preguntar si tiene seguro	bot	2- ¡Gracias, Jesse! ¿Tenés una póliza actualmente con nosotros?

⁵ Los registros con *message_number* 7, 18, 24 y 27 tienen valores nulos en el campo *message*. Esto sucede cuando el bot envía botoneras o menús desplegables, en estos casos el contenido de esos mensajes no es correctamente reflejado en la información disponible mediante API causando que figuren como mensajes vacíos.

J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	13	2020-02-03 10:52:10		user	Botón -> No
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	14	2020-02-03 10:52:10	Preguntar patente	bot	3- Ok, ahora vamos con los datos de tu vehículo. Ingresá tu patente, por ejemplo AAA-123 o de las nuevas AB-123-CD
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	15	2020-02-03 10:52:45	Inactividad 35 segundos (Webchat)	bot	Continuemos (Y)...
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	16	2020-02-03 10:52:45	Preguntar patente	bot	3- Ok, ahora vamos con los datos de tu vehículo. Ingresá tu patente, por ejemplo AAA-123 o de las nuevas AB-123-CD
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	17	2020-02-03 10:53:20	Inactividad 01:10 segundos (Webchat)	bot	¿Preferís contactarte de otra manera?
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	18	2020-02-03 10:53:20	Inactividad 01:10 segundos (Webchat)	bot	
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	19	2020-02-03 10:53:29		user	*** **
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	20	2020-02-03 10:53:29	Mensaje cuando no entiende	bot	
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	21	2020-02-03 10:53:29	Mensaje cuando no entiende	bot	¿Podrías escribirlo de otra forma, por favor?
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	22	2020-02-03 10:53:39		user	Botón -> Agendar llamado
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	23	2020-02-03 10:53:39	Solicitar celular y agendar un llamado	bot	¡Excelente! Por favor, ingresá tu teléfono celular para asesorarte en caso de ser necesario. Separá el código de área con un

						guión y no incluyas el 15 (por ejemplo: 11-55555555)
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	24	2020-02-03 10:53:39	Solicitar celular y agendar un llamado	bot	
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	25	2020-02-03 10:53:55		user	*****
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	26	2020-02-03 10:53:55	Agendar un llamado	bot	Ok, por favor indicanos cuándo prefieres que te contactemos telefónicamente:
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	27	2020-02-03 10:53:55	Agendar un llamado	bot	
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	28	2020-02-03 10:54:01		user	Botón -> Ahora!
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	29	2020-02-03 10:54:01	Contacta por llamado hoy o más tarde	bot	Perfecto, ¡te contactaremos, Jesse!
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	30	2020-02-03 10:54:05	Cierre de proceso	bot	¡Muchas gracias por confiar en *****! ¿Te puedo ayudar en algo más?
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	31	2020-02-03 10:54:19		user	Botón -> Preguntas frecuentes
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	32	2020-02-03 10:54:19	Preguntas frecuentes	bot	Estos son algunos de los temas más frecuentes de atención al cliente:
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	33	2020-02-03 10:54:25		user	Botón -> Cubiertas
J7SLA0JGVIG3NC I6YT2C_2020-02-03 10:50:37.227	2020-02-03 10:50:37	34	2020-02-03 10:54:25	Preguntas frecuentes	bot	Estos son algunos de los temas más frecuentes de cubiertas:

También es posible identificar diferentes sesiones de un mismo usuario y visualizarlas en orden cronológico filtrando o agrupando por *customer_id* y ordenando por *session_creation_time*.

Algunas operaciones básicas posibles para resumir los datos son:

- Contar *customer_id* únicos para calcular el número de usuarios.
- Contar *session_id* únicos para calcular el número de sesiones.
- Contar *id* únicos para calcular el número de mensajes.

A su vez, dada la relación jerárquica entre estos tres campos se pueden utilizar para abrir los indicadores mencionados. Por ejemplo:

- Calcular el número de mensajes totales para cada sesión.
- Calcular el número de sesiones para cada usuario.

El campo *msg_from* puede ser utilizado para identificar el emisor de cada mensaje y así contar cuantos mensajes envía cada autor posible (chatbot, usuario u operador) contando los mensajes y agrupando por los valores que toma este campo. O para filtrar solo los mensajes de algún tipo de emisor.

Finalmente la tabla con scores puede ser pensada como un diccionario donde las claves son los *session_id* de las conversaciones y el valor es el score otorgado por el usuario a cada conversación. De esta forma es posible asociar cada score a cualquier elemento de la tabla principal mediante el *session_id*.

2.3 - Posible *data leakage*

Dado el origen de los datos y la estructura de las tablas que los contienen, existe un claro problema a evitar a la hora de entrenar cualquier tipo de modelo que busque predecir scores en base al contenido de las conversaciones. Los datos existen en dos tablas, una principal que contiene toda la información relativa a las conversaciones y una segunda tabla auxiliar, más simple, que es básicamente un diccionario con el id de cada sesión como clave y su respectiva puntuación como valor. Es natural buscar utilizar el contenido de la tabla con calificaciones como objetivo de predicción y entrenamiento

(*labels*) en un modelo de aprendizaje automático supervisado, y la totalidad del contenido de la tabla principal como *features* sobre los cuales el modelo debe aprender y subsecuentemente utilizar para predecir. Sin embargo esta simple división no considera un problema inherente en el tipo de datos, son conversaciones, lo que significa que la calificación que el usuario le da al canal no solo está alojada en la tabla auxiliar sino que se encuentra dentro del mismo contenido de la conversación. Es así como existe el riesgo de tener una brecha de información sobre el campo objetivo dentro de los datos utilizados para entrenar cualquier modelo de categorización. Específicamente se trataría de una brecha de información explícita donde campos utilizados por el modelo para entrenar representan una transformación o proxy sobre el label que se busca predecir⁶.

Considerando esta realidad es necesario limpiar la base principal de todo el contenido que puede indicar explícitamente la calificación asignada a cada sesión. El ejemplo más claro de este tipo de contenido son los mensajes de los usuarios donde explícitamente califica la atención enviando el número de la calificación. Aunque también existen otros mensajes igual de problemáticos como por ejemplo las respuestas del chatbot a dicha calificación, ya que por como está configurada la experiencia la respuesta varía si el usuario brinda una calificación positiva o una negativa. Frente a calificaciones positivas el bot responde con una intención particular donde el mensaje es de agradecimiento y alegre mientras que frente a calificaciones negativas el bot contesta con otra intención y el mensaje es más bien apologético y preguntando puntos de mejora.

Con esta particularidad en mente se identificaron todos estos mensajes que revelan explícitamente el label que se busca predecir y se optó por filtrarlos completamente. La estrategia que se utilizó para filtrar estos datos y minimizar la posibilidad de *data leakage* fue localizar el primero de estos mensajes para cada sesión y filtrar todos los mensajes posteriores a éste. Ya que la presencia de cualquiera de estos mensajes, ya sea la calificación del usuario o las respuestas del bot a ser calificado, indican que el score de la conversación ya es conocido a partir de ese momento. De modo que es prudente no considerar cualquier mensaje subsecuente tanto para entrenar como para predecir. Además de evitar *data leakage* esto asegura la integridad del ejercicio, el objetivo del

6

[https://c3.ai/glossary/data-science/information-leakage/#:~:text=Information%20\(or%20data\)%20leakage%20is,generalize%20well%20to%20unseen%20data](https://c3.ai/glossary/data-science/information-leakage/#:~:text=Information%20(or%20data)%20leakage%20is,generalize%20well%20to%20unseen%20data)

modelo es identificar de forma anticipada aquellas conversaciones donde la experiencia es negativa y probablemente fueran a ser evaluadas negativamente, por lo tanto la base de datos con la que debe trabajar el modelo debe estar compuesta exclusivamente de información disponible en las sesiones previo al momento de la calificación. Por estos dos motivos se tomó la decisión de filtrar toda la información de las sesiones que existiera post momento de evaluación.

El filtrado de mensajes post calificación del usuario tiene un límite al final de cada sesión. Esto significa que los mensajes de las conversaciones de un usuario que existan después de una instancia de calificación al bot y que no sean parte de la sesión en la que se da esa calificación no serán filtrados de la base de datos. Se utilizó este criterio ya que la pregunta de calificación hace referencia a la experiencia del usuario en el momento de la pregunta y no a sus conversaciones pasadas o futuras en el canal. Las sesiones junto a sus calificaciones asociadas son tratadas como registros independientes. Llevado a un ejemplo, pensar en una conversación donde el usuario califica al bot, se despide y luego pasan más de 8 horas hasta que se comunica de nuevo y vuelve a dar una calificación cuando el bot la pide. En este ejemplo quedan separadas ambas interacciones en dos sesiones distintas (ver definición de sesión) y los mensajes post calificación que son parte de la primera sesión son filtrados de la base de datos. Mientras que los mensajes de la segunda sesión, si bien técnicamente ocurren después de una calificación, son incluidos en la base con la nueva calificación. Finalmente los mensajes de la segunda sesión que ocurren después de la segunda calificación también son filtrados porque revelarían información de esa segunda calificación.

3 - Metodología

3.1 - Modelos

3.1.1 - Algoritmo

Como parte del proyecto se entrenaron múltiples modelos utilizando el mismo algoritmo, Extreme Gradient Boosting (*XGBoost*). Este tipo de modelo combina árboles de decisión con el proceso de potenciación de gradiente y aprendizaje aditivo (Friedman, 2001). Una de las ventajas de este tipo de modelos está en la selección

interna de features, otorgando cierta robustez frente a variables irrelevantes. Esta es una característica que se hereda de los modelos de árbol de decisión. A su vez son indiferentes a variaciones en la escala de los valores que puede tomar cada variable (Van Ryzin et al., 1986). La principal desventaja de los árboles de decisión es su falta de precisión dada la baja capacidad de especificidad, especialmente en árboles pequeños. Sin embargo, los procesos de potenciación permiten mitigar esta desventaja y generar modelos más sensibles y específicos.

Otra razón por la que se optó por utilizar este algoritmo es por su eficiente manejo de datos faltantes (Chen y Guestrin, 2016). Más adelante en el trabajo está explicado cómo se estructuraron las conversaciones dentro de la base de datos, pero aquí se puede adelantar que la estructura resultante presenta una gran cantidad de valores faltantes entre sus features. En este trabajo se aprovechó el hecho de que *XGBoost* cuenta con mecanismos para manejar valores faltantes, ya que utilizando la estructura planteada fue necesario trabajar con una base de datos mala.

También se aprovecharon las funcionalidades de regularización que ofrece *XGboost* (Chen y Guestrin, 2016) que se mencionan en la sección [3.4 - Optimización de hiperparámetros](#). Estas opciones buscan impedir el *overfitting* en los modelos entrenados para lograr que sean generalizables. El algoritmo facilita el uso de las mismas desde sus hiperparámetros, siendo esta otra de las razones por las que fue seleccionado.

La última razón por la que se eligió este algoritmo es porque ofrece un grado razonable de interpretabilidad sobre sus resultados. Gracias a que existen técnicas para ayudar con la interpretación y visualización de las decisiones que toman este tipo de modelos (Lundberg y Lee, 2017). Más allá de los resultados predictivos que alcancen los modelos, es importante poder entender, al menos de forma parcial, cómo toman decisiones y qué limitaciones existen para poder plantear próximos pasos y posibles mejoras. En la sección [5 - Resultados](#) se encuentra un análisis en detalle de la importancia de los features utilizados y la performance de los diferentes modelos, siendo una de las partes más valiosas de este ejercicio. Dicho análisis aprovecha la interpretabilidad del algoritmo utilizado.

En la sección [6.1 Limitaciones y posibles mejoras](#) se exploran otros posibles algoritmos a ser aplicados en relación a este problema mencionando técnicas como word2vec para representar palabras como vectores densos o el uso de BETO para trabajar con los datos como lenguaje natural. Si bien es posible que dichos algoritmos generen buenos resultados frente al problema propuesto, en este trabajo se eligió explorar XGBoost ya que es computacionalmente menos exigente y porque permite un mayor grado de interpretabilidad sobre la importancia de los features utilizados. En esta instancia se considera prioritario evaluar si existe una señal identificable dentro de los datos y lograr interpretarla con un mayor nivel de detalle, por sobre la búsqueda de maximizar los indicadores de performance.

3.1.1 - Modelos implementados

Todos los modelos comparten un único objetivo, que consiste en identificar conversaciones con puntuaciones negativas previo a ser calificadas. Parte del problema es definir en qué punto de la conversación es recomendable ejecutar el modelo para intentar predecir la satisfacción del usuario antes de recibir una calificación negativa. Para ello se generaron diferentes bases utilizando el mismo registro de conversaciones, empezando por el primer mensaje de cada conversación en la primera base y agregando de a un mensaje para las bases siguientes, repitiendo el proceso 17 veces. De esta forma se logró entrenar 17 modelos independientes, donde el primero solo trabaja con los datos disponibles hasta el primer mensaje de las conversaciones y cada modelo subsiguiente incrementa en uno la cantidad de mensajes que tiene disponible para evaluar, hasta el modelo número 17 que trabaja con la información disponible dentro de los 17 primeros mensajes de las conversaciones. Lo que se busca es poder evaluar cómo varía la performance del mismo algoritmo cuando varía el punto de corte sobre las conversaciones en el que se espera que se realiza la predicción. En la sección [3.3.2 - Conjuntos de entrenamiento, validación y testeo](#) está explicado cómo se trabajaron estas bases para minimizar efectos aleatorios al momento de comparar resultados.

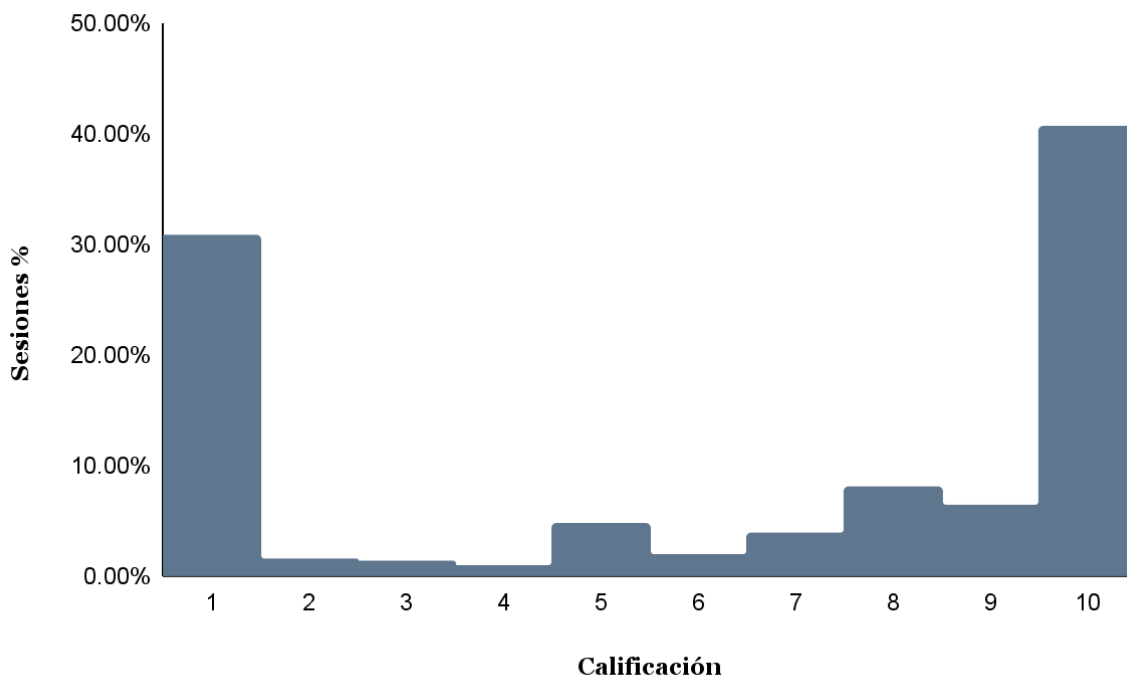
A partir de este punto, al largo expresado en el número de mensajes utilizados en las conversaciones también se lo llamará punto de corte. Y siguiendo esta lógica, cada modelo entrenado estará numerado de la misma manera dependiendo de qué punto de corte se utilice en las conversaciones con las que trabaja, desde el modelo 1 hasta el

modelo 17. De esta forma es sencillo interpretar a simple vista el largo de conversaciones con el que opera cada modelo basándose en su nomenclatura y viceversa.

3.1.2 - Variable objetivo

Como se mencionó en la sección de Datos [2.2.2 - Base con calificaciones](#), existe una tabla con las calificaciones del usuario frente a la pregunta de satisfacción con la atención del bot. Esta calificación es un valor entero entre 1 y 10. No se busca predecir la calificación exacta, sino categorizar las experiencias entre negativas y no negativas. Por lo tanto, para generar la variable objetivo se llevó a cabo el ejercicio de binarizar las calificaciones, siendo 1 todas aquellas menores o iguales a 4 y 0 aquellas mayores a 4. De esta forma, en la variable objetivo binaria, la clase positiva contiene las experiencias particularmente malas con calificaciones menores o iguales a 4. Ya que es este el grupo que se pretende identificar de forma prematura para poder ser rescatado durante las conversaciones.

Figura 0: Distribución de sesiones calificadas por usuarios en función de la calificación obtenida



En la [Figura 0](#) se puede observar que naturalmente las calificaciones se concentran en los valores extremos del rango posible. Esto es positivo ya que indica que no se estaría

perdiendo mucha información, particularmente de calificaciones moderadas, al resumirlas en la variable objetivo binaria descrita anteriormente. Originalmente el 30,79% de las sesiones calificadas se encuentran en el extremo inferior con una calificación de 1 y el 40,64% en el extremo superior con 10. Llevado a la variable objetivo binaria el 34,49% de las sesiones toma valor 1 (con calificaciones menores o iguales a 4) y el 65,5% quedan en el otro grupo, 0.

3.2 - Estrategia de validación

3.2.1 - Métrica de evaluación de modelos

La métrica de evaluación utilizada es el área bajo la curva ROC (AUC), tanto para evaluar la calidad predictiva final de los modelos como para guiar las iteraciones de optimización durante el entrenamiento de los modelos *XGBoost*. La curva de ROC (Receiver Operating Characteristic) mide la relación entre la tasa de verdaderos positivos (verdaderos positivos sobre positivos totales) y la tasa de falsos positivos (falsos positivos sobre negativos totales) de un clasificador binario para sus posibles umbrales de decisión (Müller y Guido, 2016). La curva ROC siempre tiene origen en (0,0) donde el umbral de cualquier clasificador impide que se clasifiquen casos como positivos por lo tanto ambas tasas son 0%, y su otro extremo está en (1,1) donde el umbral clasifica todas las instancias como positivas por lo tanto ambas tasas son 100%. Curvas ROC que pasan cerca del punto (0,1) indican mejores clasificadores ya que esto significa una alta tasa de verdaderos positivos acompañada de una baja tasa de falsos positivos. Calculando el área bajo la curva ROC se puede saber la calidad de un clasificador independientemente del umbral de decisión. Este indicador puede tomar valores entre 0 y 1, donde 1 indicaría un clasificador perfecto que pasa por el punto (0,1), 0,5 significa clasificaciones compatibles con la que obtendría un clasificador que asigna clases de manera totalmente azarosa y 0 significa que el modelo es perfectamente incorrecto (de modo que al invertir sus predicciones, se estaría prediciendo de manera perfecta).

3.3.2 - Conjuntos de entrenamiento, validación y testeo

Las unidades de análisis del proyecto son conversaciones o sesiones en el contexto de la base de datos trabajada. Este nivel de análisis se desprende del objetivo del modelo,

poder predecir conversaciones particulares que van a ser malas experiencias para intervenirlas de forma anticipada y evitar ese resultado. Otra posible unidad de análisis que se podría tomar es toda la historia de cada usuario en el canal y categorizar por usuario. Sin embargo, esto no sería tan útil debido a una serie de factores que hacen que la compañía también prefiera usar la sesión como unidad de análisis. Uno de los principales factores es que estas experiencias pueden estar separadas por grandes periodos de tiempo y naturalmente son consideradas como interacciones individuales por los usuarios. Otro factor significativo es que el diseño de la experiencia está segmentado en sesiones; el ejemplo más evidente de este comportamiento es que el usuario es saludado por el chatbot al comienzo de una sesión y despedido sobre el final. Ese es un simple ejemplo pero la presencia de esta lógica de segmentación en el diseño de las conversaciones abarca toda la experiencia. Por estas razones se optó por utilizar sesiones individuales como unidad de análisis para el modelo.

Al dividir la base de datos en conjuntos de entrenamiento, validación y testeo se debió incluir dos consideraciones adicionales a un ejercicio de entrenamiento supervisado clásico.

La primera se desprende de la posibilidad de que exista más de una sesión para un mismo usuario. Tomando en cuenta esta posibilidad se optó por separar los conjuntos primero por usuarios y alocar todas las sesiones correspondientes a cada usuario a un único conjunto. Del total de sesiones en la base de datos se extrajo la lista de usuarios únicos que las componían y se la dividió para generar los índices que indicarían a que grupo corresponden las sesiones de cada usuario. Primero se seleccionó un subconjunto aleatorio del 10% de los usuarios totales que se asignó al conjunto de testeo, luego del 90% de usuarios restantes se tomó un 20% para utilizar en el conjunto de validación y el resto se utilizó en el conjunto de entrenamiento. La división final de la base de usuarios es 10% en grupo de testeo, 18% en grupo de validación y 72% en grupo de entrenamiento. Como la división de usuarios es completamente aleatoria, el promedio de sesiones por usuario prácticamente no varía entre los grupos, por lo que el balance de sesiones mantiene proporciones similares: 10,08% en grupo de testeo, 18,04% en grupo de validación y 71,87% en grupo de entrenamiento. De esta manera se aseguró que no hayan sesiones de usuarios que son parte de los conjuntos de validación y testeo en el conjunto de entrenamiento.

La segunda consideración tiene relación con el ejercicio de comparar el desempeño del modelo para diferentes puntos de corte en el largo de la conversación. Para minimizar el impacto de la aleatoriedad en este ejercicio de evaluación de puntos de corte se optó por utilizar el mismo índice de división de conjuntos para las bases de datos de todos los puntos de corte a evaluar. En otras palabras, una vez elegidos los usuarios que pertenecen a cada grupo, esta división se respetó en el ejercicio de entrenamiento de todos los modelos y por lo tanto la selección de sesiones también se mantuvo entre modelos. Lo único que varió entre las bases de datos para cada punto de corte es la información disponible dentro de cada sesión, pero la cantidad de sesiones y su agrupación en entrenamiento, validación o testeo se sostuvo para el entrenamiento y evaluación de los 17 modelos.

De esta manera cada modelo se entrenó sobre el conjunto de entrenamiento y se evaluó contra el conjunto de validación. En la sección [3.4 - Optimización de hiperparámetros](#) se explica cómo se repitió este proceso en cada uno de los 17 modelos para encontrar estocásticamente buenos conjuntos de hiperparámetros para el algoritmo.

3.3 - Definición de datos de trabajo

3.3.1 - Selección de conversaciones

No todas las conversaciones que ocurren en el canal son relevantes para el negocio y por lo tanto tampoco lo son para el ejercicio de predicción. Una manera simple y efectiva para identificar y poder descartar conversaciones no relevantes es mediante su largo. Como se observó en la sección de datos el campo *message_number* puede ser usado para conocer el número del último mensaje de cada sesión, es decir, el largo de estas.

Sesiones demasiado cortas no llegan a proporcionar información de lo que buscaba el usuario ni son suficientes para evaluar si representan una experiencia buena o mala. El caso más común de este tipo de conversaciones se da cuando un usuario envía un primer mensaje al canal que dispara la primera respuesta del bot y luego no envía ningún otro mensaje, esto es considerado una conversación abandonada por el usuario. Es prácticamente imposible determinar qué causas generan estos casos de uso particulares. Los posibles motivos pueden ir desde factores completamente ajenos al negocio y a la conversación que no son relevantes. Hasta motivos relevantes como

querer cancelar la comunicación por no haber encontrado la respuesta del bot satisfactoria. El ejercicio de intentar entender estas razones es, *a priori*, completamente especulativo, por lo que no se justifica incluir estas sesiones en el análisis. En otras palabras, este tipo de conversaciones no serán consideradas como parte del problema a atacar.

Primero se analizó la distribución de cantidad de sesiones relativa al largo de las mismas. Este análisis contempla la totalidad de las sesiones, no solo las que cuentan con una calificación que serán utilizadas como *labels* para entrenar y evaluar los modelos. Se analizó la totalidad de las sesiones para entender la realidad del canal sobre el cual eventualmente trabajará el modelo, ya que en la práctica el modelo intentará predecir en todas las sesiones que lleguen al punto de corte determinado. La [Figura 1](#) muestra el porcentaje de sesiones sobre las sesiones totales para cada posible largo de conversación⁷. Es claro que existe una proporción no despreciable de sesiones con una baja cantidad de mensajes, por ejemplo menos que 10. Al ver la misma distribución pero de forma acumulada ([Figura 2](#)) se observa que el 27% de las sesiones tienen 10 mensajes totales o menos. Esta proporción es aún más clara en la [Figura 3](#) que muestra el porcentaje de sesiones sobre el total que tiene al menos X cantidad de mensajes, donde el 100% de las sesiones contienen 1 mensaje y ese porcentaje disminuye a medida que aumenta el número de mensajes.

⁷ Se excluyeron algunas sesiones outliers con cantidades de mensajes muy altas para mantener la escala del eje horizontal.

Figura 1: Distribución de sesiones totales en función de su cantidad de mensajes

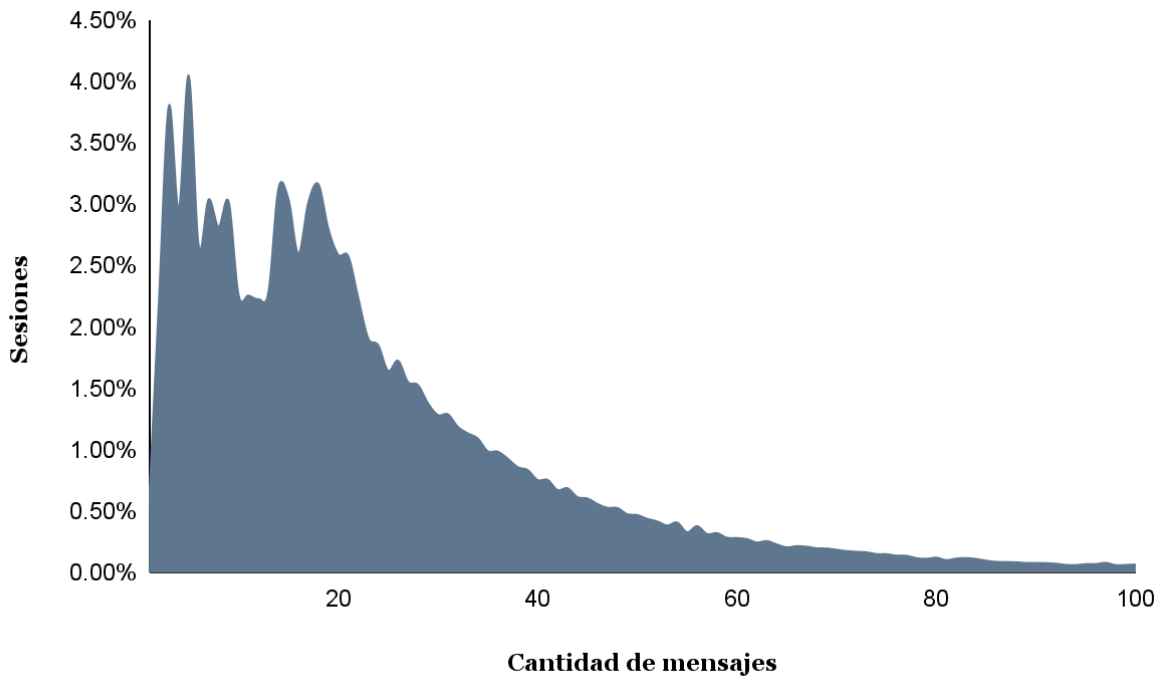


Figura 2: Distribución de sesiones totales acumuladas en función de su cantidad de mensajes

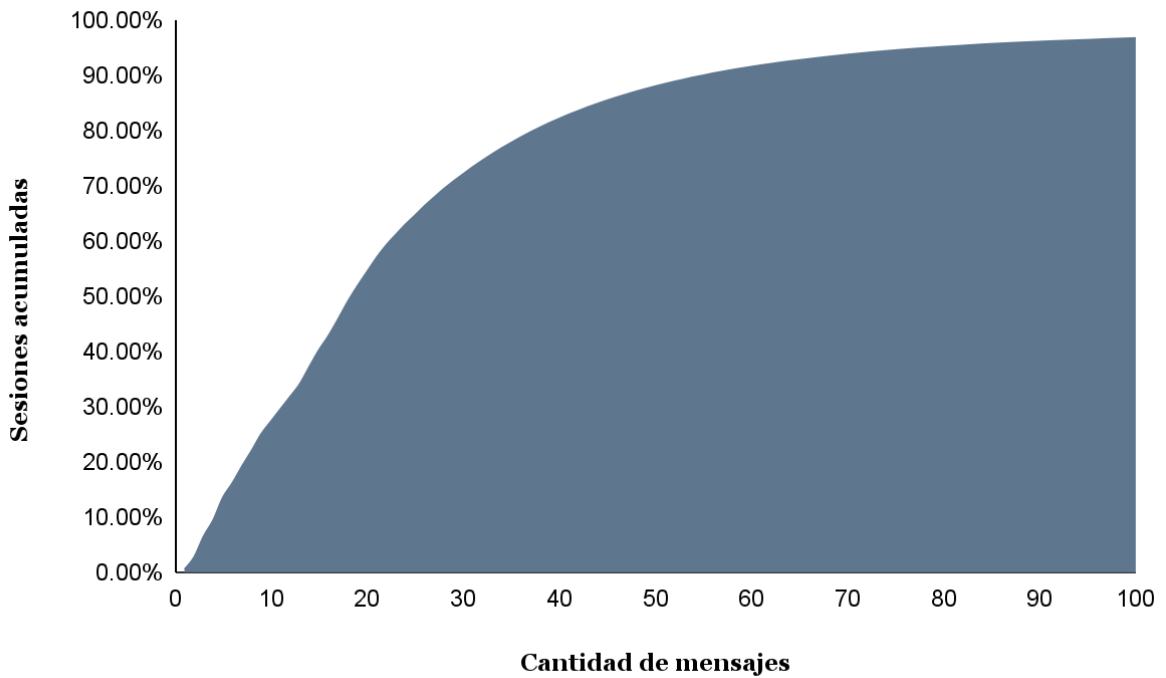
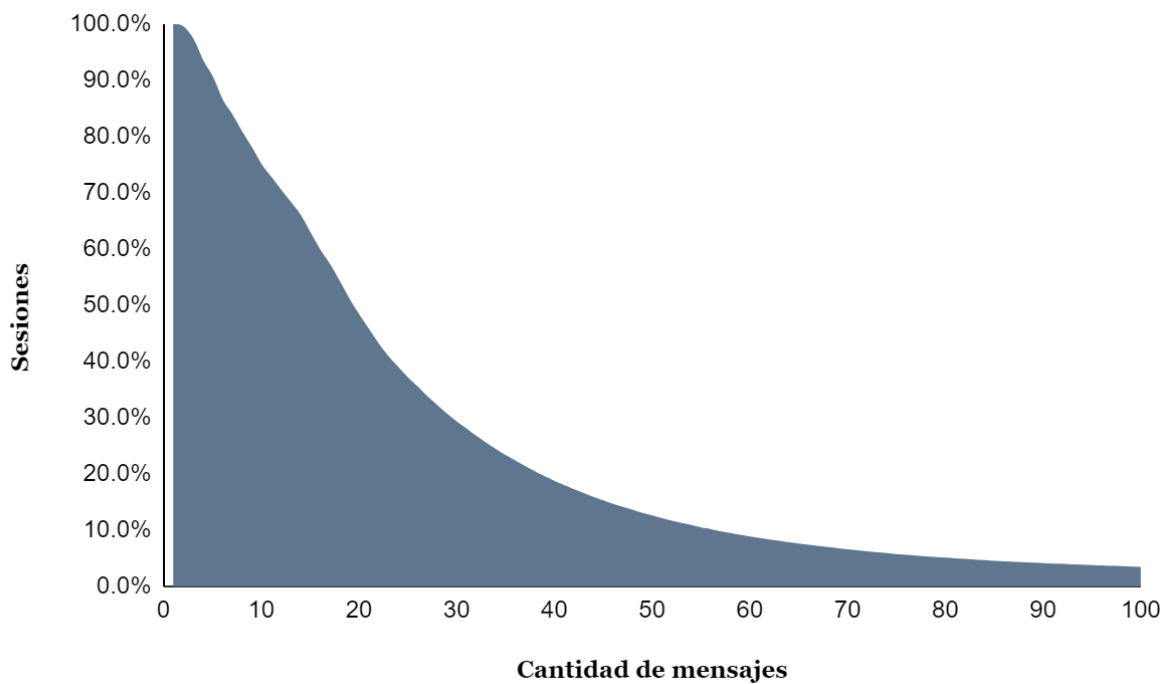


Figura 3: Distribución de sesiones totales en función de la cantidad de mensajes que contienen



Estas sesiones calificadas como cortas no entran todas exactamente en el ejemplo descrito anteriormente de un único mensaje del usuario y la primera respuesta del chatbot, pero presentan básicamente el mismo problema. Es por esto que en este trabajo se buscó un criterio de selección de sesiones para asegurar que las que fueran incluidas en el ejercicio de predicción sean relevantes.

Se continúa este análisis, ahora únicamente con las sesiones que se utilizarán en este trabajo, es decir aquellas que cuentan con una calificación del usuario. Este subconjunto de las conversaciones totales se distribuye según la [Figura 4](#) y la [Figura 5](#). Aquí la proporción de sesiones cortas es mucho menor. Usando el mismo ejemplo de 10 mensajes de largo se ve que solo el 1,32% de las conversaciones con *score* entran en esta ventana. A su vez en la [Figura 6](#), que muestra qué porcentaje de sesiones calificadas sobre el total de sesiones calificadas tienen al menos X cantidad de mensajes, es claro que la mayoría de las conversaciones con *score* no son cortas y por lo tanto relevantes *a priori*.

Figura 4: Distribución de sesiones calificadas en función de su cantidad de mensajes

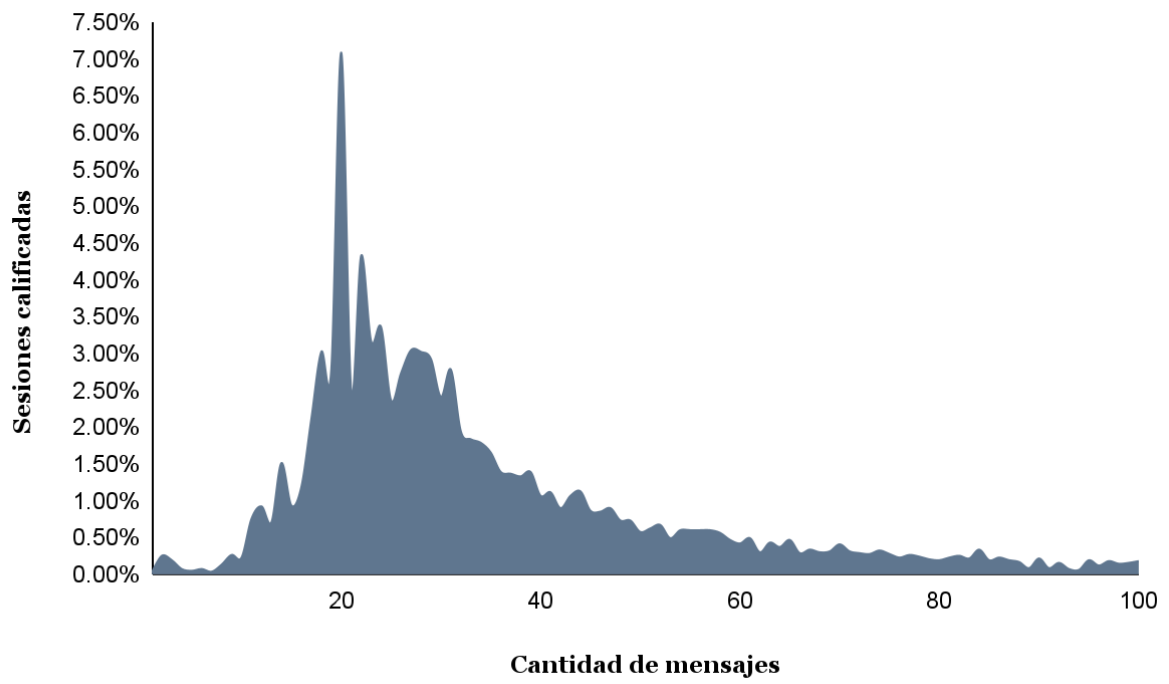


Figura 5: Distribución de sesiones calificadas acumuladas en función de su cantidad de mensajes

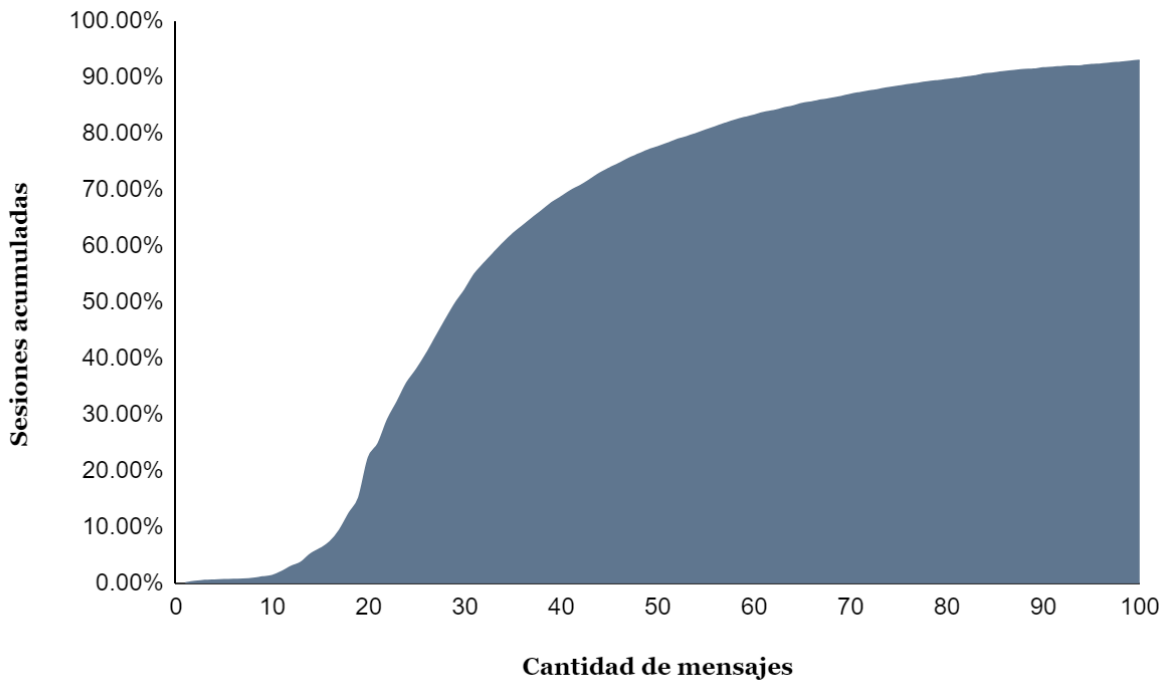
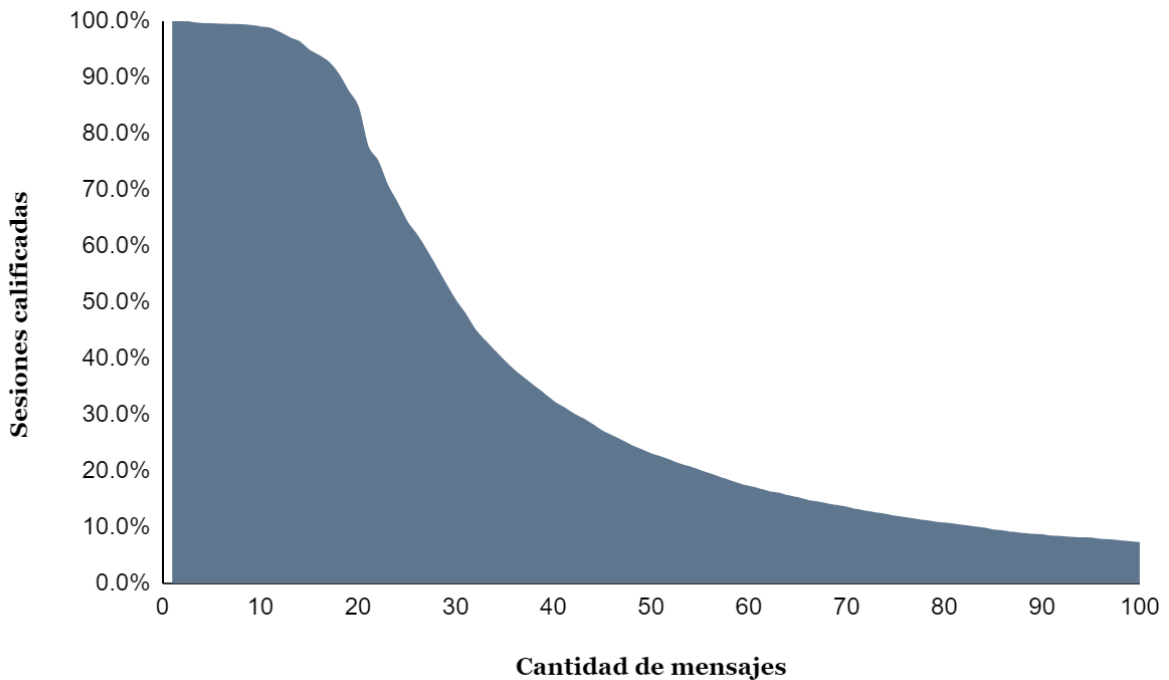


Figura 6: Distribución de sesiones calificadas en función de la cantidad de mensajes que contienen



Entonces, una de las limitaciones de entrenar un modelo de aprendizaje automático supervisado donde se necesitan *labels* para utilizar como objetivo de predicción (Müller

y Guido, 2016), en este caso tiene el beneficio de descartar estas conversaciones cortas que no son relevantes para el negocio. Aún así, la condición de haber sido puntuadas por el usuario no parece ser una solución lo suficientemente robusta como para asegurar no incluir conversaciones no relevantes en el ejercicio. Existen conversaciones atípicas donde, incluso después de pocas interacciones entre el bot y el usuario, el primero pide una calificación y el segundo la provee, aunque la experiencia no haya sido realmente significativa. Esto sucede en parte por la naturaleza de las conversaciones, donde un usuario habla en un chat con un bot de atención al cliente. Es un contexto casual, y hay que entender que es posible que el usuario envíe mensajes sin mucha discreción o premeditación y estos no reflejen la realidad de la experiencia.

La respuesta del usuario a la pregunta de satisfacción del bot debe ser entendida como una medición de un indicador propio de las ciencias sociales. En este contexto una variable es aquel aspecto abstracto de la realidad que se busca conocer mientras que un indicador es el instrumento utilizado para obtener información práctica y medible sobre el concepto⁸. La calidad real de la experiencia es la variable latente que se intenta estimar y las calificaciones brindadas son un posible indicador con el que se infiere esta variable. Con esta perspectiva, se busca que el modelo pueda predecir la variable, no el indicador. Y la existencia de conversaciones muy cortas en las que no parece haber ningún intercambio relevante de información pero con calificación de la experiencia por parte del usuario, deben ser entendidas como un error de medición⁹.

Es por este motivo que se buscó un criterio adicional para minimizar la inclusión de este tipo de conversaciones en el proceso de entrenamiento y evaluación del modelo. Mediante la inspección de conversaciones y la distribución de sesiones observada en las figuras [1](#), [2](#), [3](#), [4](#), [5](#), [6](#) se definió utilizar el número 17 como la cantidad mínima de mensajes requerida en una sesión para ser incluida en el análisis. Conversaciones de al menos 17 mensajes parecen indicar interacciones relevantes entre el usuario y el bot. De esta manera se buscó disminuir el posible error de medición en la calificación provista. Este criterio permite utilizar el 92,7% de las sesiones calificadas, como

⁸ https://www.argentina.gob.ar/sites/default/files/ii_informacion_social_indicadores.pdf

⁹ "En todo proceso de medición existen limitaciones dadas por los instrumentos usados, el método de medición, el observador (u observadores) que realizan la medición. Asimismo, el mismo proceso de medición introduce errores o incertezas." (Gil y Rodríguez, 2001)

muestra la [Figura 6](#), y aprovechar la mayor parte de los datos disponibles en este ejercicio de predicción.

3.3.2 - Ingeniería de atributos

3.3.2.1 - Bolsa de intenciones

En la sección [2.2 - Estructura](#) se mencionó un campo dentro de la base principal que será uno de los más importantes para generar la estructura con la que trabajará el modelo, este campo es *rule_name*. Las intenciones o *rule_names* que ejecuta el bot durante sus conversaciones son el principal indicador del flujo que sigue cada conversación. Esto se da porque la lógica con la que se configuran las respuestas del bot es cerrada.

Se llamó cerrada a este tipo de lógica por 3 motivos:

- Ante cualquier mensaje del usuario el bot solo puede contestar con la intención que esté predefinida para responder dicho mensaje particular. La única excepción son los mensajes de no entendimiento que se disparan justamente cuando el bot no tiene una respuesta preprogramada para algún mensaje específico del usuario. En estos casos, contesta con intenciones genéricas donde se le pide al usuario que repita su consulta con otras palabras o que elija dentro de una serie de opciones la que mejor se ajuste a su consulta.
- La mayor parte de las intenciones sólo pueden ser ejecutadas en un orden predeterminado. Por ejemplo, en un flujo de cotización de seguro para auto, primero se pregunta el año del vehículo, luego la marca y el modelo y finalmente se ofrece un costo estimado de la cobertura en base a esos datos. En este ejemplo la intención que ofrece el costo estimado de la cobertura nunca puede ser ejecutada sin antes haber pasado por las preguntas de año, marca y modelo.
- El bot en general promueve ser navegado mediante botones y no mensajes libres. La combinación de los dos puntos anteriores hace que la manera más simple de implementar los flujos del bot y a su vez más clara para el usuario a la hora de entender qué información o servicios están disponibles en el canal, sea mediante botoneras que le permiten al usuario navegar la conversación sin ni siquiera escribir. No está consensuado que tipo de navegación genera menos fricción para

los usuarios, si lenguaje natural libre o interfaz de menús, pero existen estudios que parecen justificar el uso de menús (Nguyen et al., 2022). Aunque siempre exista la posibilidad de escribir un mensaje en lenguaje natural el usuario está incentivado dentro del flujo de la conversación a ir avanzando según las opciones presentadas por el chatbot.

Dada esta realidad en las conversaciones del canal, un humano que posea conocimiento conceptual sobre las intenciones del chatbot y tenga acceso a una lista secuencial de las intenciones ejecutadas en una conversación podría obtener una visión bastante completa de dicha conversación en su totalidad. Esto se debe a que es posible inferir los mensajes del usuario a partir de las respuestas del chatbot, salvo aquellas que se clasifiquen como no entendidas. Y siguiendo este ejercicio, este humano podría reconstruir la mayoría de las conversaciones solo con el listado de intenciones ejecutadas sin ni siquiera tener la información secuencial del orden en que se ejecutaron ya que esta información está embebida dentro de la descripción de cada intención. Por ejemplo, para el flujo de cotización si se sabe que se ejecutaron las 4 intenciones mencionadas: pregunta año; pregunta marca; pregunta modelo; estimación de costo, también es conocido el orden en el que se ejecutaron porque es el único posible.

Se tomó inspiración en una estructura de bolsa de palabras donde documentos de texto son representados como un conjunto no ordenado de palabras manteniendo su multiplicidad (Hapke et al., 2019). De esta manera se generó una estructura de bolsa de intenciones siguiendo la lógica de una bolsa de palabras pero utilizando intenciones en lugar de palabras. Esta estructura permite representar la información de cada conversación en un formato compatible con el algoritmo *XGBoost*. Virtualmente no se pierde la mayoría de la información de las conversaciones ya que se aprovecha la lógica cerrada explicada en esta sección. En esta estructura de bolsa de intenciones, cada posible intención que ejecuta el bot es una columna y el valor que toma es la cantidad de veces que la intención correspondiente envió un mensaje para la sesión descrita. Así cada sesión tiene la información de todas las intenciones que se ejecutaron y la cantidad de mensajes que se enviaron por cada intención.

3.3.2.2 - Bolsa de palabras

Partiendo de la estructura de bolsa de intenciones mencionada previamente y considerando la llamada lógica cerrada con la que el bot responde a los mensajes de los usuarios solo falta información sobre un tipo de mensajes posibles. Estos son los mensajes abiertos enviados por el usuario que disparan un “no entendido” en el bot. Como ya se explicó, los mensajes del usuario que se envían al oprimir un botón y los mensajes del usuario que disparan alguna intención específica del bot pueden ser resumidos conociendo la intención con la que el bot responde y esto queda capturado en la estructura de bolsa de intenciones. Sin embargo la intención de “no entendido” es la única a la que no aplica esta lógica ya que puede ser disparada por cualquier mensaje posible que no tenga una respuesta preconfigurada. Por lo tanto es necesario agregar a esta estructura de datos una forma de capturar la información de estos mensajes. Lo que se hizo es tomar todos los mensajes del usuario que no son botones y agregarlos a la base de datos como bolsa de palabras (Hapke et al., 2019).

Para generar esta bolsa de palabras se lematizaron las palabras utilizando la función WordNetLemmatizer de la librería NLTK (Bird & Loper, 2004). Por otra parte, ya que las conversaciones pueden incluir emojis en los mensajes enviados por los usuarios estos fueron incluidos como features ya que pueden contener información relevante. Entonces el proceso de tokenización y construcción de la bolsa de palabras consistió de la lematización de las palabras de lenguaje natural e inclusión de los posibles emojis enviados.

3.3.2.3 - Información histórica de usuarios

Finalmente el único tipo de información que resta incluir a la estructura generada es información sobre la historia del usuario con el canal. La unidad de análisis siempre es una sesión particular, sin embargo existe la posibilidad de que cualquier sesión no sea la primera vez que el usuario se comunica mediante el canal. Para capturar esta información se generaron dos nuevos campos: *usuario_retornante* y *segundos_desde_ultima*.

- *usuario_retornante*: es una variable binaria que toma el valor 1 si el usuario al que corresponde la sesión tiene al menos una sesión previa y 0 si no.

- *segundos_desde_ultima*: muestra la cantidad de segundos que pasaron desde el inicio de la última sesión del usuario previa a la actual. Cuando no existe una sesión previa (el campo *usuario_retornante* tendría el valor 0) este campo es nulo.

Estos dos campos le permiten al modelo tener cierta información contextual sobre la sesión que está intentando evaluar.

3.4 - Optimización de hiperparámetros

Para buscar una buena combinación de hiperparámetros se utilizó la estrategia de “random search” (Bergstra y Bengio, 2012) donde se definió un rango para cada hiperparámetro a optimizar y se encontraron 20 combinaciones aleatorias de estos dentro de los rangos definidos. Luego se entrenaron los 20 posibles modelos resultantes de las combinaciones aleatorias generadas y el mismo conjunto de datos y se eligió el modelo que tuviera un mejor AUC ROC en los datos de validación. Como el proyecto originalmente implicaba entrenar 17 modelos sobre el mismo conjunto de conversaciones pero moviendo el punto de corte desde el mensaje número 1 hasta el número 17, se aplicó este proceso a cada uno de esos modelos independientemente.

Los rangos utilizados para cada hiperparámetro son los siguientes:

'learning_rate': (0.01, 0.3)

'max_depth': (3, 10)

'subsample': (0.5, 1)

'colsample_bytree': (0.5, 1)

'gamma': (0, 5)

'reg_alpha': (0, 1e-5)

'reg_lambda': (0, 1e-5)

Para resumir el proceso, lo que se hizo para cada uno de los 17 puntos de cortes fue generar 20 nuevas combinaciones aleatorias de hiperparámetros; entrenar 20 modelos con esas combinaciones para el punto de corte dado; y seleccionar la combinación que

tuviera una mejor performance en el set de validación. Por lo tanto se entrenaron 340 (17x20) modelos¹⁰ y se seleccionó el mejor conjunto de hiperparámetros (en el set de validación) para cada uno de los 17 puntos de corte utilizados.

4 - Análisis exploratorio de los datos

4.1 - Distribución de los datos

Entre el 29-03-2021 y el 21-05-2023 existieron 183.688 conversaciones en el canal conversacional. Estas sesiones pertenecen a 107.108 usuarios únicos implicando que el promedio de sesiones por usuario es de 1,71. Las sesiones consisten de 5.729.453 mensajes en total, dividiendo estos mensajes entre los emisores posibles se obtiene la siguiente distribución: 2.242.467 (39,14%) mensajes de usuarios; 2.901.428 (50,64%) mensajes del chatbot y 585.558 (10,22%) mensajes de operadores (agentes humanos). Por lo tanto una sesión promedio tiene el largo de 31,2 mensajes aproximadamente, donde 12,2 son enviados por el usuario, 15,8 por el chatbot y 3,2 por un operador humano. En la Tabla 2 se encuentran resumidos todos estos valores. Esta es la descripción de la totalidad de las conversaciones en el canal, sin embargo, no todas las conversaciones son de interés para este proyecto.

Tabla 2: Descripción general de los datos totales

Indicador	Totales
Usuarios	107.108
Sesiones	183.688
Sesiones promedio por usuario	1,71
Mensajes	5.729.453
Mensajes de usuario	2.242.467
Mensajes de chatbot	2.901.428
Mensajes de operador	585.558
Porcentaje de mensajes de usuario	39,14%
Porcentaje de mensajes de chatbot	50,64%
Porcentaje de mensajes de operador	10,22%
Mensajes promedio por sesión (Largo promedio)	31,2

¹⁰ El tiempo de cómputo dedicado para el entrenamiento de los 340 modelos fue de 2 horas y 30 minutos aproximadamente, trabajando localmente con un procesador "Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz" y 16 GB de memoria RAM.

Mensajes de usuario promedio por sesión	12,2
Mensajes de chatbot bot promedio por sesión	15,8
Mensajes de operador promedio por sesión	3,2

La base de datos utilizada en este proyecto consiste solo de 8.587 conversaciones (sesiones) en las cuales el usuario califica su experiencia en relación con el bot en este mismo período, esto representa solo un 4,67% de las conversaciones totales. Este porcentaje es pequeño en relación a la totalidad de las conversaciones y se debe en parte a las reglas implementadas en el comportamiento del chatbot y a la baja tasa de respuesta a la pregunta. El chatbot no está programado para siempre preguntar satisfacción al usuario, básicamente existen dos condiciones principales que excluyen conversaciones de recibir la encuesta.

La primera condición es que todas las sesiones en donde participa un operador de atención al cliente humano reciben una pregunta de satisfacción diferente donde se pide evaluar la atención del agente humano y no al chatbot en sí. En el período analizado la tasa de conversaciones derivadas a un operador humano es del 35,48% por lo tanto esta porción del total de conversaciones no es elegible para tener una calificación de satisfacción con el chatbot. Existe una cantidad marginal de 582 sesiones con una calificación para el chatbot que también son derivadas (6,78% sobre el total de sesiones calificadas), pero estas sólo ocurren cuando se derivan después de haber recibido una calificación, y debido al filtro de mensajes de sesiones definido en la sección [2.3 - Posible data leakage](#) estos segmentos de las conversaciones calificadas donde aparecen agentes humanos son excluidos de los datos trabajados.

La segunda condición por la que otra parte de las sesiones no derivadas no reciben la pregunta de satisfacción con el bot es porque son demasiado cortas, de pocas interacciones entre el usuario y el chatbot. En estos casos está determinado por regla de implementación que no se haga la pregunta de satisfacción con el bot ya que no son indicativas de ningún tipo de experiencia relevante. Este último punto está relacionado con la distribución expuesta en la sección [3.3.1 Selección de conversaciones](#) en la [Figura 3](#) donde se muestra el volumen de conversaciones cortas no relevantes. Tomando el punto de corte utilizado para la selección de conversaciones a incluir en el modelo de al menos 17 mensajes se encuentra que el 42,9% de las sesiones totales no alcanzan este

umbral. Si bien desde la implementación del chatbot las reglas que expresan esta decisión de diseño son más complejas y detalladas que el punto de corte general definido en este trabajo, el resultado y las sesiones excluidas son muy similares.

Tomando en cuenta estas condiciones se puede calcular la proporción de sesiones calificadas utilizadas en este proyecto sobre el total de sesiones ajustando este denominador, quitando todas aquellas sesiones inelegibles para ser calificadas. La cantidad real de sesiones donde se envía al usuario la pregunta de calificación es 37.713. Con este denominador, las 8.587 sesiones calificadas representan el 22,76% del total de sesiones que recibieron la pregunta, que son el universo de sesiones relevantes para el trabajo.

Continuando con esta distinción entre sesiones calificadas y no calificadas se puede volver a describir las conversaciones como se hizo en la [Tabla 2](#) discriminando entre los dos grupos. En la [Tabla 3](#) se pueden observar ambas descripciones. El foco de este proyecto está en el grupo de sesiones calificadas que está compuesto por las ya mencionadas 8.587 sesiones de 7.959 usuarios únicos, donde cada usuario tiene un promedio de 1.08 sesiones. Consisten de 356,307 mensajes en total: 139.071 (39,03%) mensajes de usuarios; 210.836 (59,17%) mensajes del chatbot y 6.400 (1,80%) mensajes de operadores (agentes humanos). Una sesión calificada promedio tiene el largo de 41,5 mensajes totales donde 16,2 son enviados por el usuario, 24,6 por el chatbot y 0.7 por un operador humano.

Tabla 3: Descripción general de los datos totales

Indicador	Sin Calificación	Con Calificación
Usuarios	99.149	7.959
Sesiones	175.101	8.587
Sesiones promedio por usuario	1,77	1,08
Mensajes	5.373.146	356.307
Mensajes de usuario	2.103.396	139.071
Mensajes de chatbot	2.690.592	210.836
Mensajes de operador	579.158	6.400
porcentaje de mensajes de usuario	39,15%	39,03%
porcentaje de mensajes de chatbot	50,07%	59,17%
porcentaje de mensajes de operador	10,78%	1,80%

Mensajes promedio por sesión (Largo promedio)	30,7	41,5
Mensajes de usuario promedio por sesión	12,0	16,2
Mensajes de chatbot bot promedio por sesión	15,4	24,6
Mensajes de operador promedio por sesión	3,3	0,7

4.2 - Comparación entre chatbot y operadores humanos

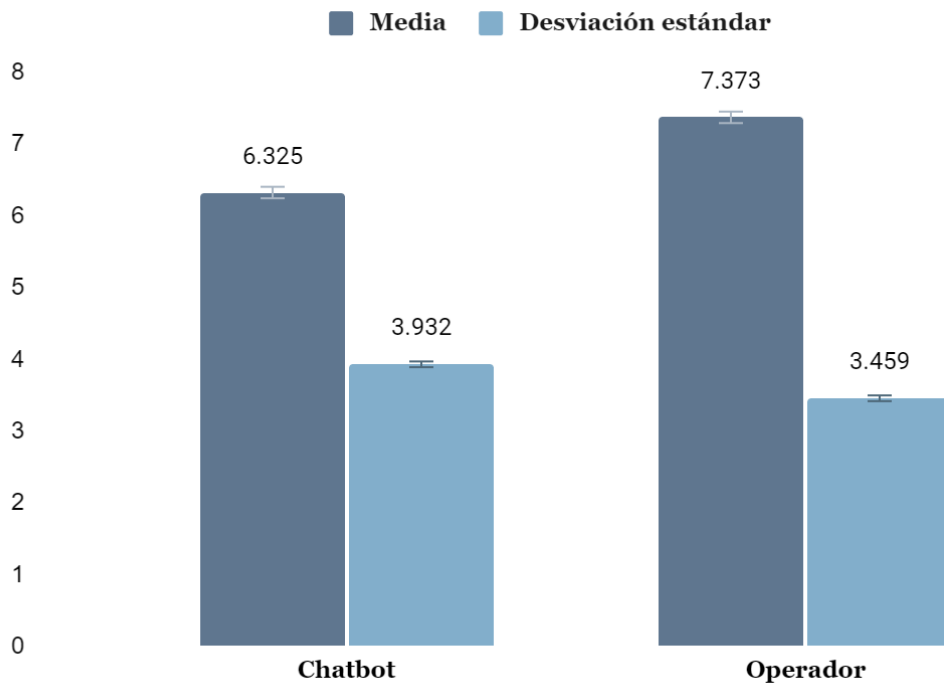
Aunque el enfoque principal del proyecto se centra en la detección temprana de experiencias de usuario negativas para permitir intervenciones y, por ende, mejorar la experiencia en esas conversaciones, otra parte fundamental de esta solución reside en la naturaleza de la intervención propuesta. Si bien excede el alcance de este trabajo definir en detalle tanto la intervención como la manera en la que se implementaría, la propuesta general es que aquellas sesiones que se identifiquen como negativas sean derivadas prontamente a operadores humanos. La razón por la cual se plantea que un operador humano podría ser más eficaz que el chatbot y, por ende, ofrecer una mejor experiencia en casos que resultan problemáticos para el chatbot fue, en un principio, una intuición basada en la experiencia derivada de la inspección de un número significativo de sesiones. No obstante, afortunadamente, los datos proporcionan información que respalda esta intuición.

Como ya fue mencionado existen dos posibles calificaciones para una sesión. La primera es sobre la que se estuvo haciendo foco, la calificación de satisfacción del usuario con la atención ofrecida por el bot. La segunda, resulta útil en esta sección, y es la calificación de satisfacción del usuario frente a la atención ofrecida por los operadores humanos. En principio ambas calificaciones son mutuamente excluyentes, aunque existen excepciones marginales como se mencionó en la sección [4.1 - Distribución de los datos](#). La existencia de ambos indicadores permite comparar la calidad de ambas experiencias.

La [Figura 7](#) muestra la calificación promedio recibida por el chatbot y los operadores respectivamente junto con sus respectivas desviaciones estándar y márgenes de error para un nivel de confianza del 95%. Se observa que existe una diferencia entre ambos promedios muy por encima de los márgenes de error calculados, siendo estadísticamente significativa. Particularmente la calificación promedio que reciben los operadores es 1,048 puntos mayor que la del chatbot o 0,266 desviaciones estándar (del

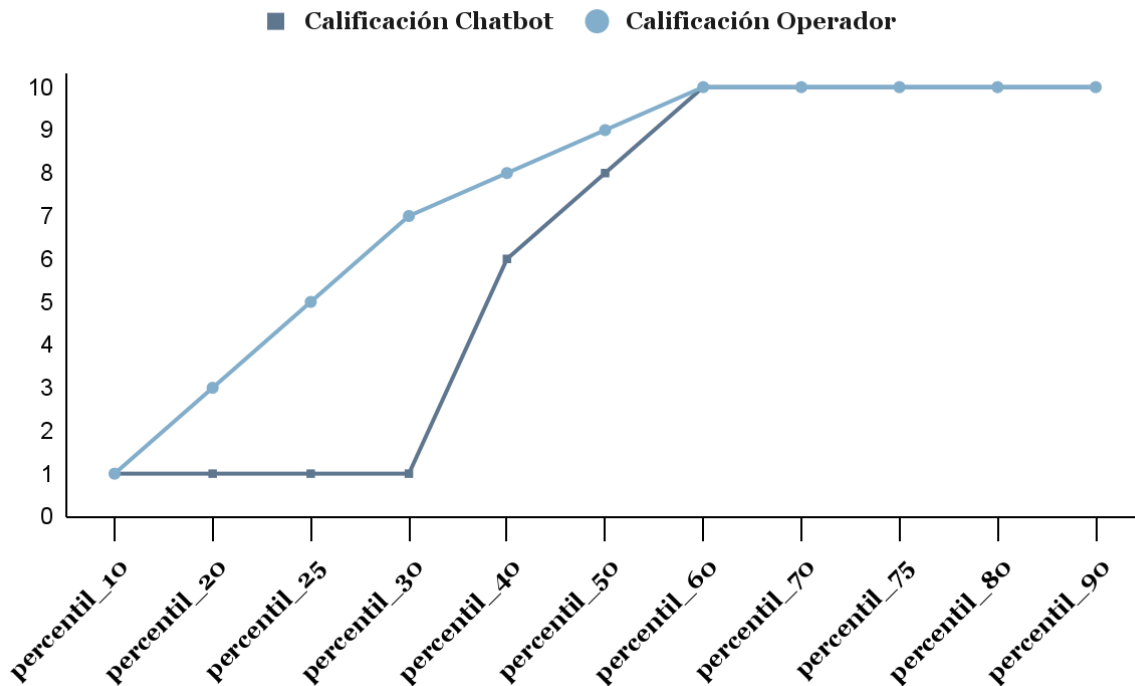
promedio del chatbot) más alto. Esta diferencia parece indicar una mejor calidad de atención en conversaciones derivadas en relación a las no derivadas.

Figura 7: Calificaciones chatbot y operadores



La [Figura 8](#) muestra cómo se distribuyen las calificaciones para cada tipo en todos sus deciles y cuartiles. Aquí también se puede apreciar la diferencia en la calidad de ambas experiencias más allá de los promedios ya mencionados. La figura indica que ambas experiencias se unifican a partir del sexto decil en adelante con calificaciones de 10 sobre 10. Las diferencias se concentran en los primeros 5 deciles, para el chatbot los primeros 3 deciles mantienen una calificación de 1 y suben abruptamente hasta 10 en el 30% de sesiones que existe entre el percentil 30 y el percentil 60, mientras que en los operadores solo el primer decil tiene una calificación de uno y aumenta gradualmente en los siguientes hasta llegar a 10 en el percentil 60.

Figura 8: Distribución de calificaciones chatbot y operadores



Esta diferencia en la proporción de calificaciones extremadamente malas (calificación de 1) sumado a la diferencia en las calificaciones promedio parecen justificar la intuición de que la calidad de la atención de los operadores es superior a la del chatbot. Por lo tanto se podría esperar que la intervención planteada de derivar preventivamente sesiones problemáticas a operadores humanos debería mejorar la calidad de esas experiencias y como consecuencia impactar positivamente en la calidad de la experiencia del canal en general.

4.3 - Relación entre largo de la sesión y calificación

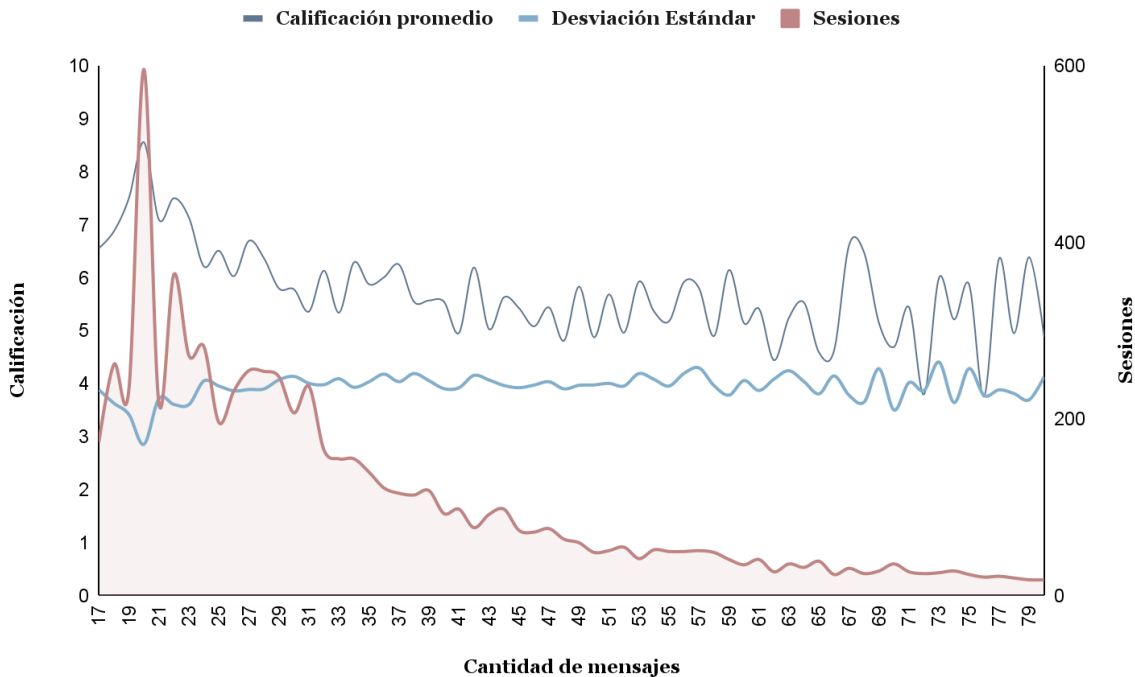
El largo de las sesiones no es utilizado como un predictor en sí mismo, pero entender si existe una correlación con la calificación de las mismas puede ser útil para evaluar la posible eficacia de las intervenciones que plantea este proyecto. Recordar, el objetivo del modelo al buscar identificar experiencias negativas en los primeros mensajes de cada conversación es proporcionar al chatbot la capacidad de derivar dichas conversaciones hacia un operador humano, quien tiene una mayor probabilidad de satisfacer las necesidades del cliente. Para eso es útil entender cómo se distribuyen las calificaciones

de las sesiones a medida que aumenta la cantidad de mensajes totales que la componen, o en otras palabras su largo.

El 89.8% de todas las sesiones calificadas tienen entre 17 y 80 mensajes totales, el 10.2% restante de las sesiones se distribuyen entre 81 y 201 mensajes totales con pocos o individuales casos para cada largo. Por lo tanto, el análisis se concentró en el primer segmento que incluye la mayoría de las conversaciones y permite observar correlaciones más claras, ya que los datos agrupados como la calificación promedio tienen menor variabilidad y son más representativos al estar contruidos con múltiples observaciones.

En la [Figura 9](#) se puede observar en el eje horizontal todos los posibles largos entre 17 y 80 mensajes, en el eje vertical izquierdo las posibles calificaciones que pueden obtener las sesiones entre 1 y 10, y finalmente en el eje vertical derecho la cantidad de sesiones. Se puede observar la calificación promedio de las sesiones para cada largo, así como la desviación estándar de cada uno de estos promedios leyendo sobre el eje vertical izquierdo. La línea "Sesiones" se lee sobre el eje vertical derecho y mapea la cantidad de sesiones para cada largo posible.

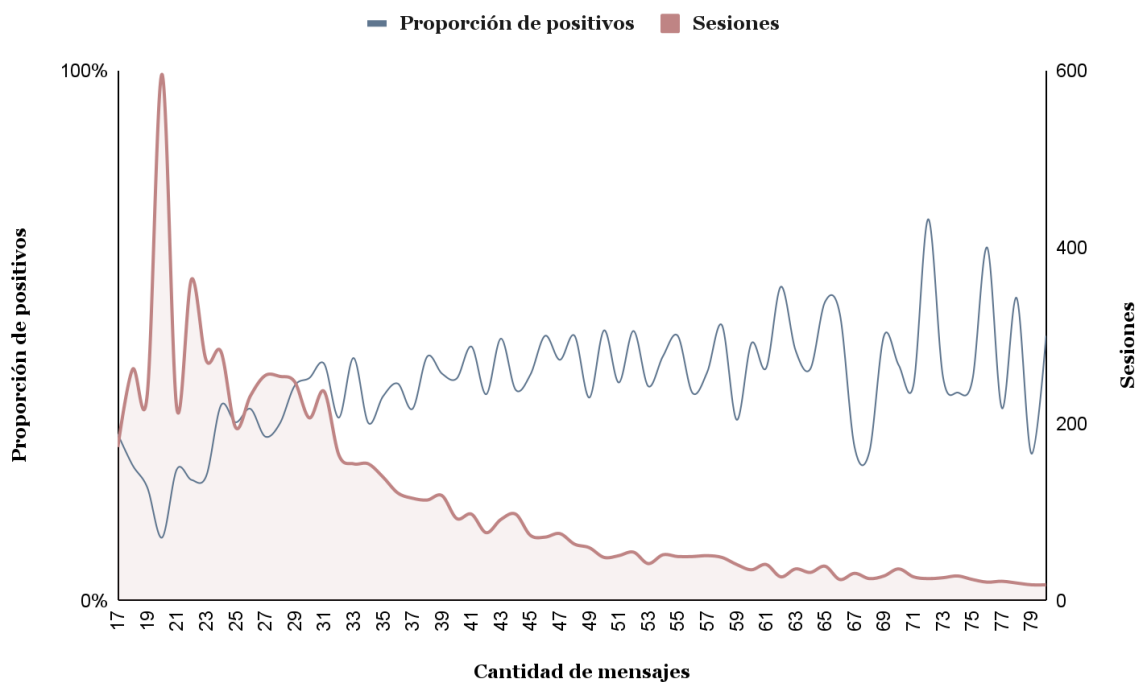
Figura 9: Calificaciones en función del largo de sesiones



En la figura se puede ver que sesiones más cortas parecen obtener calificaciones promedio más altas. Particularmente la calificación promedio para el 25% de las sesiones más cortas es de 7,55 mientras que el promedio para el 75% de las sesiones restantes es de 5,83 observando una disminución del 22,72% entre ambos grupos. Esta tendencia sirve para justificar el espíritu de la intervención planteada ya que implica que sesiones más largas entre el usuario y el chatbot suelen tener peores calificaciones. Intuitivamente esto se puede entender considerando que la información y los servicios que disponibiliza el bot suelen ser fáciles de acceder desde los primeros menús que se ofrecen en la conversación. De esta forma tiene sentido pensar que sesiones donde la oferta del chatbot satisface las necesidades del usuario tenderán a ser más cortas. Mientras que sesiones más largas pueden ser casos en los que el usuario no encuentra rápidamente lo que busca dentro de las primeras interacciones con el bot. Aquí el usuario navega la conversación por más tiempo hasta: encontrar una opción que se adecúe a sus necesidades; conformarse con alguna opción que se ajuste parcialmente a lo que estaba buscando; o abandonar la conversación por completo sin haber satisfecho su necesidad.

La [Figura 10](#) muestra la misma información que la [Figura 9](#) con la diferencia de que en lugar de las calificaciones promedio, se muestra la proporción de casos positivos en la variable binaria objetivo del modelo. Aquí 1 son sesiones con una calificación menor o igual a 4 y 0 sesiones con calificaciones mayores a 4. Otra diferencia es que no se incluye la desviación estándar ya que no resulta informativa dada la naturaleza binaria de la variable objetivo. Replicando los mismos cortes mencionados anteriormente, el primer cuartil de las sesiones más cortas cuentan con una proporción de valores positivos en la variable objetivo del 20,92% mientras que las sesiones restantes presentan un 39,71%, observando un aumento del 89,75% entre ambos grupos. De esta forma en la [Figura 10](#) se puede observar una relación positiva entre el largo de sesión y la variable objetivo transmitiendo el mismo mensaje que la [Figura 9](#). Sesiones más cortas son positivas en una mayor proporción mientras que sesiones más largas tienen una parte más grande de sesiones con calificaciones negativas, iguales o menores a 4.

Figura 10: Proporción de casos positivos en la variable objetivo en función del largo de sesiones



Este análisis parece avalar la idea de que sería beneficioso interrumpir sesiones potencialmente negativas con una participación humana. Se busca evitar este fenómeno donde el usuario se frustra o baja sus niveles de satisfacción mientras continua

hablando con el bot exclusivamente. El hecho de que la desviación estándar sea relativamente amplia comparado al rango dentro del cual pueden existir las calificaciones, en promedio 3,66 en las calificaciones sobre 10, indica que para cada largo existe gran variedad en cuanto a la calidad de las experiencias. Por lo tanto un modelo que pudiera discriminar las experiencias negativas de las positivas permitiendo dejar las potenciales experiencias positivas intactas e intervenir las negativas podría tener un impacto muy significativo en la calificación de satisfacción general del canal. Esta observación justifica el esfuerzo de generar una solución compleja, como la propuesta en este trabajo, por sobre alguna regla dura y fácil de implementar como derivar automáticamente pasados los X mensajes entre el bot y el usuario.

4.4 - Relación entre intenciones y calificación

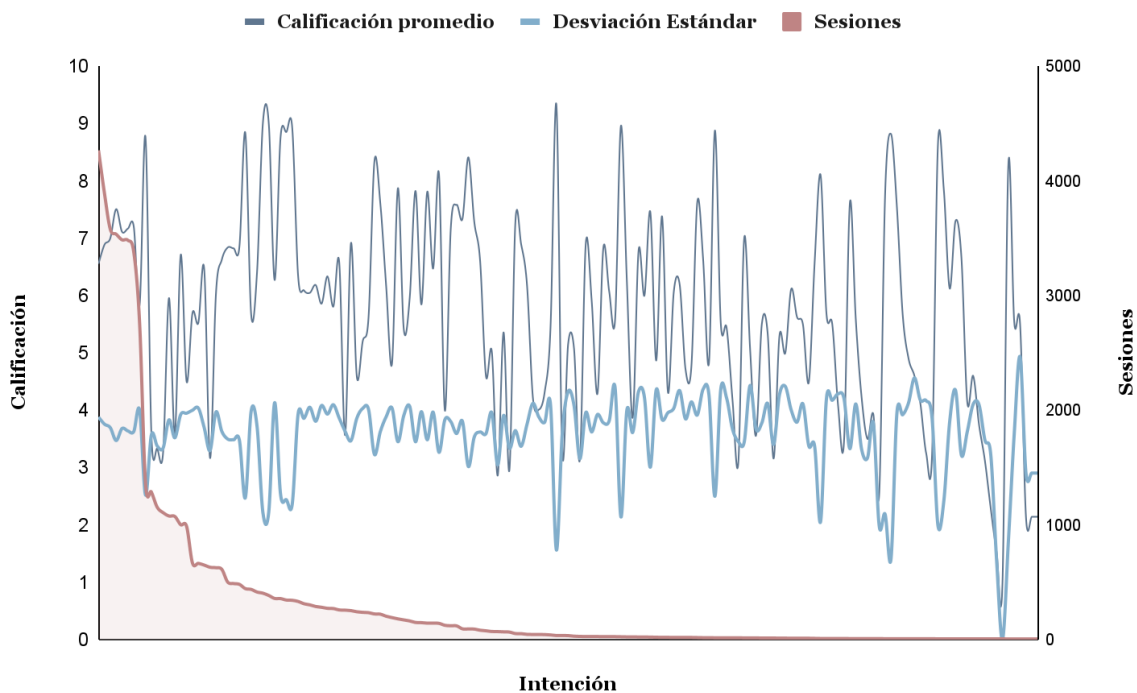
En la sección [3 - Metodología](#) se explicó que la mayor parte de la información contenida en cada conversación puede ser resumida con una estructura de bolsa de intenciones. Para evaluar el potencial de éxito de dicha metodología para predecir la variable objetivo se analizó si variaciones en qué intenciones componen cada sesión tienen algún tipo de correlación con variaciones en las calificaciones de las mismas.

La tarea de mapear diversas composiciones de conversaciones sería un desafío considerable, dado que las combinaciones posibles son prácticamente infinitas. Este ejercicio implicaría una notable similitud con el objetivo que persigue el modelo entrenado. Por lo tanto, como parte del análisis exploratorio de los datos, se exploró esta covariación utilizando una estructura más simple. Lo que se hizo fue agrupar las calificaciones de las sesiones por intenciones únicas de forma tal que si una intención se ejecutó al menos una vez dentro de una sesión la calificación de esta se considera en la agrupación de dicha intención.

Existen 442 intenciones distintas que se ejecutaron al menos una vez en conversaciones durante el periodo estudiado. Como los modelos planteados trabajan durante el comienzo de las conversaciones, las intenciones que solo aparecen después del mensaje número 17 fueron descartadas del análisis, ya que este es el umbral de corte más alto entre los modelos entrenados. Con esta salvedad quedan 327 intenciones, aún así, dentro de este total, algunas intenciones solo aparecieron en una cantidad marginal de

conversaciones. Por lo tanto para este análisis solo se tomaron intenciones que intervinieron en más de 5 conversaciones, descartando otras 166 posibles intenciones. La [Figura 11](#) muestra: sobre el eje vertical izquierdo la calificación promedio y su desviación estándar de las sesiones que incluyen cada una de las 161 intenciones distintas restantes; y sobre el eje vertical derecho la cantidad de sesiones donde figura cada una de las intenciones o en otras palabras una referencia de qué tan común o popular es cada intención. En el eje horizontal se listan todas las intenciones analizadas, no es importante discriminar los valores para cada una individualmente sino entender la distribución de los mismos en el espacio que representan estas intenciones.

Figura 11: Distribución de calificaciones promedio en función de intenciones



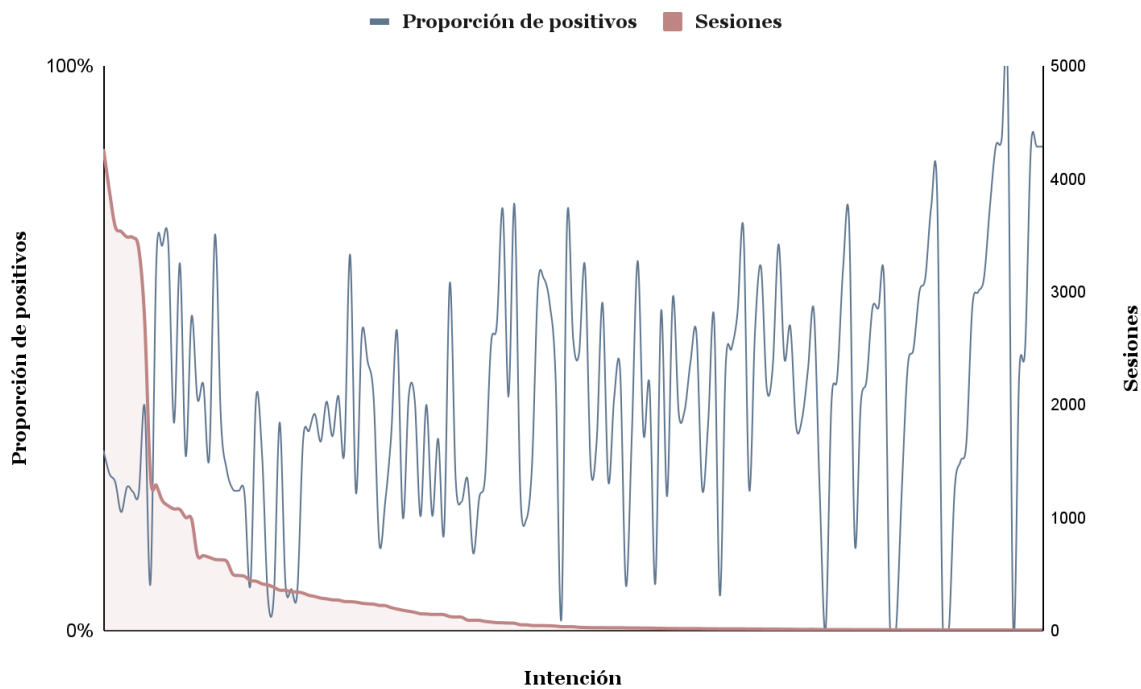
Como se puede observar en la [Figura 11](#) existe un gran rango de variabilidad en las calificaciones promedio para cada posible intención. Esto parece sostener la intuición de que un modelo que utilice la presencia y recurrencia de diferentes intenciones como features de entrenamiento podría capturar estas diferencias en las calificaciones que se observan en la figura, lo cual es alentador.

De todas formas esta figura muestra también dos posibles limitaciones para el modelo planteado. La primera y menos significativa es el desbalance en la presencia de ciertas intenciones en las sesiones. Como se puede observar existe una gran diferencia entre la

minoría de intenciones con la mayor cantidad de sesiones (intenciones muy populares) y la mayoría de intenciones restantes con cantidades de sesiones muy bajas, incluso después de haber filtrado las 166 intenciones con 5 o menos sesiones. Esta posible limitación no es muy significativa ya que debería poder ser saldada por la robustez del modelo *XGBoost* para lidiar con features no relevantes (ver sección [3.1.1 - Algoritmo](#)) como parecen serlo estas intenciones con pocas sesiones. La segunda limitación se puede observar en los valores que toma la desviación estándar para cada uno de estos promedios, que suele ser alta considerando el rango total en el que habitan las calificaciones. El promedio ponderado por la cantidad de sesiones que incluye cada intención, de los valores que hacen a la serie de desviaciones estándar se encuentra en 3,62. Esto indica que las calificaciones de diferentes sesiones que pasan por una misma intención tienen una alta variabilidad. Si bien existen diferencias entre las calificaciones promedio para diferentes intenciones que podrían ser aprovechadas por el modelo planteado, la variabilidad en las calificaciones para sesiones que comparten features podría confundir al modelo y limitar su capacidad predictiva.

La [Figura 12](#) incluye las mismas sesiones e intenciones que la [Figura 11](#) pero utilizando el objetivo de predicción binario en lugar de las calificaciones originales, donde calificaciones menores o iguales a 4 son 1 o positivas y el resto 0 o negativas. Se muestra el promedio de variable objetivo por intención, en el eje vertical izquierdo y la cantidad de sesiones en el eje vertical derecho al igual que la [Figura 11](#). Al igual que en el caso de la [Figura 10](#) esta figura no incluye la desviación estándar sobre los promedios ya que no resulta informativa dada la naturaleza binaria de la variable objetivo. El promedio es equivalente a la proporción de observaciones positivas y es lo único que se necesita para resumir la totalidad de la distribución en la variable objetivo. La homogeneidad en la variable objetivo para una intención dada es explicada en qué tan lejos de 0,5 se encuentra su promedio. Intenciones con promedios cerca de 1 o 0 contienen sesiones mayoritariamente con ese valor respectivo, mientras que intenciones con promedios cerca de 0,5 contienen una distribución más balanceada entre los dos posibles valores.

Figura 12: Proporción de casos positivos en la variable objetivo en función de intenciones

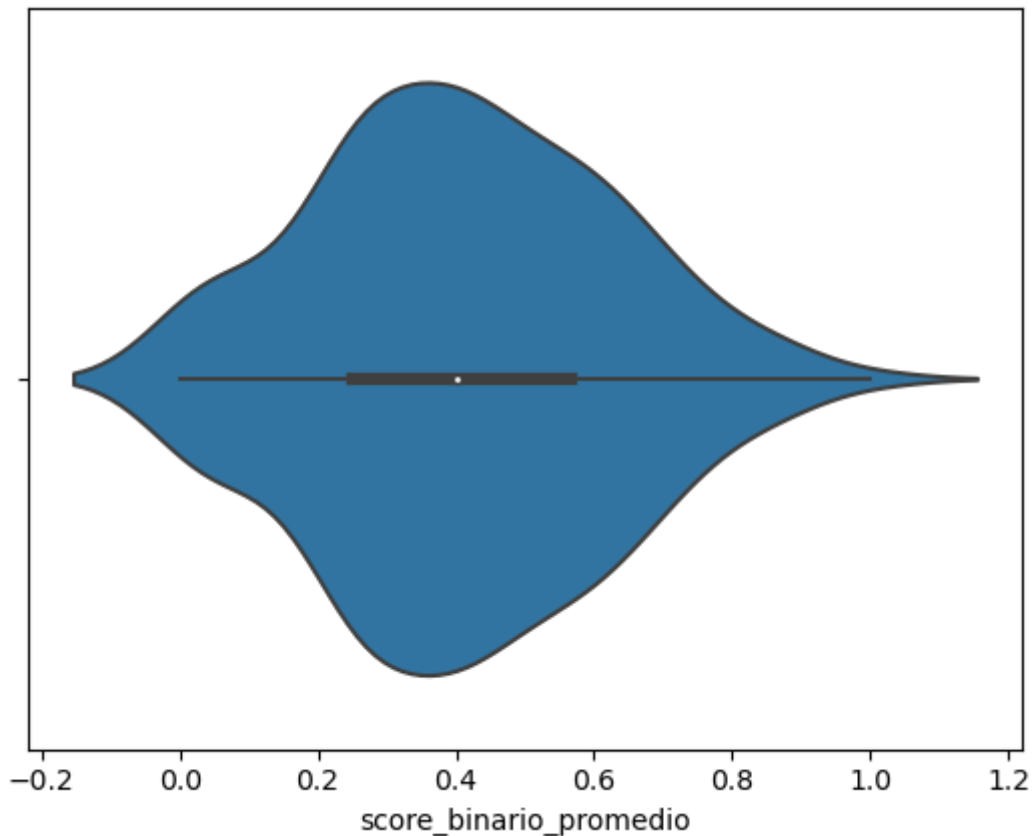


En la [Figura 12](#) se puede observar una gran rango y variabilidad en la distribución de promedios sobre la variable objetivo para las diferentes intenciones. Esto significa que el poder explicativo de las intenciones que parecían mostrar las calificaciones promedio en la [Figura 11](#) se mantiene incluso después de traducir las calificaciones de los usuarios a la variable objetivo binaria planteada.

Se realizó un análisis sobre la distribución de la proporción de observaciones positivas según la presencia de cada intención que puede ser resumido en la [Figura 13](#). Es necesario aclarar que en este análisis todas las intenciones (sin contar las 166 excluidas previamente por no figurar en más de 5 sesiones) tienen el mismo peso en la distribución, es decir se ordenaron sus valores promedio sobre la variable objetivo para obtener los puntos relevantes para un violín plot sin considerar el número de sesiones en las que aparecen. La figura muestra en el primer cuartil de la distribución, intenciones con valores de la variable objetivo mayoritariamente negativos, menor o igual al 0,25. La mediana se encuentra en 0,4 y el tercer cuartil en 0,566 mientras que los límites (0 y 1) están dados por la naturaleza binaria de la variable objetivo. Otro dato útil es el promedio de las proporciones de valores positivos que se sitúa en 0,408 cerca de la mediana. Profundizando sobre este análisis se calculó el promedio ponderado por

la cantidad de sesiones contabilizadas para cada intención, el cual se sitúa en 0,329 siendo aún más bajo que la mediana expuesta en la [Figura 13](#).

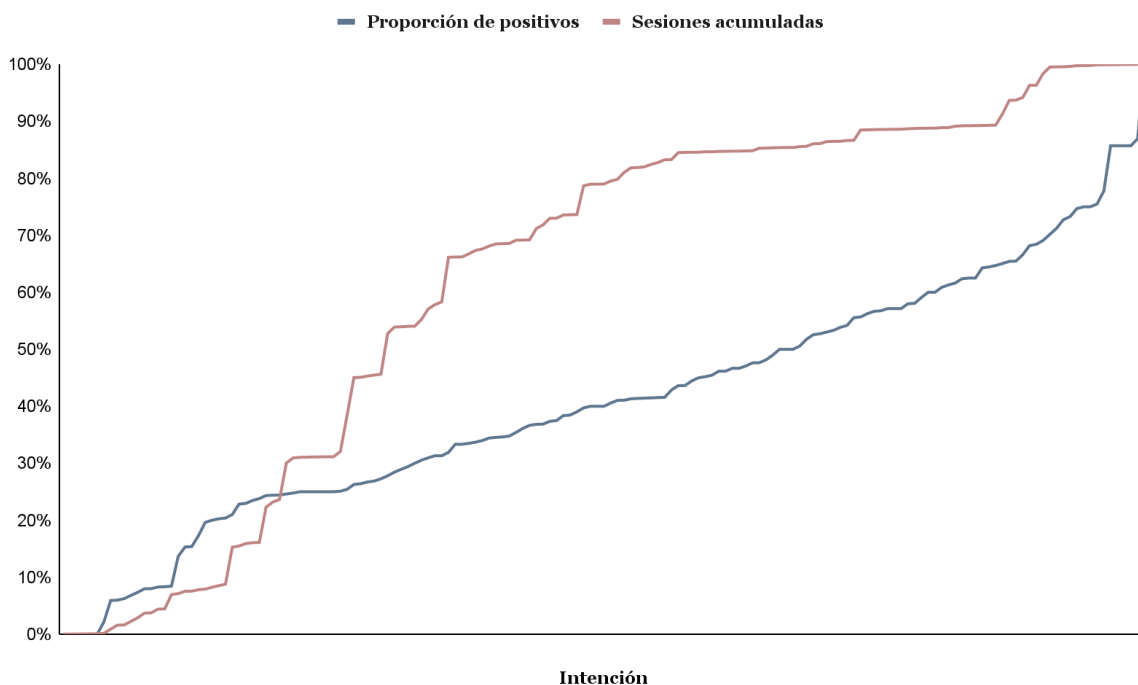
Figura 13: Distribución (violín plot) de proporción de casos positivos en variable objetivo en función de intenciones



¿Qué información se puede extraer de estos datos? Para empezar es claro que la mayoría de las intenciones (casi el 75%) aparecen en sesiones con valores de la variable objetivo mayoritariamente negativas (indicando experiencias de usuario positivas). Además se puede observar la limitación en los datos mencionada cuando se analizó la distribución de las desviaciones standard de las calificaciones promedio observables en la [Figura 11](#) esta vez reflejada en el hecho de que para el 50% de las intenciones la proporción de valores positivos en la variable objetivo está entre 0,25 y 0,566 con el 50% de estas (25% del total) entre 0,4 y 0,566. Con la interpretación ya mencionada donde proporciones cercanas a 0,5 indican balance entre valores positivos y negativos en la variable objetivo, se puede observar que existe una considerable proporción de intenciones que no le aportarán (individualmente) mucha información a los modelos para identificar experiencias negativas.

La [Figura 14](#) ofrece una forma más clara de representar esta distribución gráficamente y permite ajustarla según la cantidad de sesiones representadas en cada intención. Así se puede analizar en mayor detalle la limitación que se empezó a observar en la [Figura 13](#). Aquí en el eje horizontal otra vez están las intenciones seleccionadas (como en las figuras [11](#) y [12](#)) pero ordenadas según sus proporciones de valores positivos en la variable objetivo. Al igual que en la [Figura 12](#) se puede ver la proporción de valores positivos en la variable objetivo para las sesiones que incluyen cada intención. La ventaja de este gráfico es que muestra el porcentaje de sesiones acumulado (ordenado por la proporción de positivos) a medida que se incluyen nuevas intenciones, esto permite determinar cualquier percentil de la distribución corregido por la popularidad de cada intención. Particularmente se busca determinar cuánto pesan las intenciones con proporciones de valores positivos cercanas a 0,5 que intuitivamente parecen no ser útiles para discriminar tipos de experiencia.

Figura 14: Proporción de casos positivos en la variable objetivo y acumulativo de sesiones en función de intenciones



Se definió un rango de 0,4 a 0,6 en las proporciones de positivos en la variable objetivo para encontrar intenciones no útiles. Antes de utilizar los pesos de la serie “Sesiones Acumuladas” de la [Figura 14](#) se puede obtener que el 31,20% de las intenciones

analizadas caen en esta ventana terminando de detallar la distribución problemática que se comenzó a describir en la [Figura 13](#). Esto definitivamente es una limitación para el modelo ya que significa que aún más features no le serán útiles para aprender a discriminar experiencias. Sin embargo, el hecho de que no todas las intenciones sean igual de populares permite repetir este análisis utilizando la serie “Sesiones Acumuladas” en la [Figura 14](#) para ajustar por la popularidad de cada intención. Es necesario aclarar que el total de “Sesiones Acumuladas” no representa la totalidad de las sesiones real, ya que contabiliza múltiples veces la misma sesión por cada intención que la compone. Aún así permite tener una noción de que tan prevalente es cada intención sobre la totalidad de la base de datos. Utilizando este criterio, el 31,20% de intenciones identificadas no útiles solo tienen una presencia del 9,85% en la base de datos y por lo tanto no son tan relevantes como aparentan en un primer análisis. La otra cara de este análisis es que el 68,8% de las intenciones restantes con una presencia del 90,15% en la base de datos tienen proporciones de valores positivos más desbalanceadas lo que indicaría que deberían poder ser explotadas efectivamente por el algoritmo propuesto.

Sin embargo la [Figura 14](#) revela otra posible limitación al modelo, aún más relevante, no existe una gran cantidad de intenciones con proporciones mayormente positivas indicando experiencias negativas. El porcentaje de intenciones con una proporción de positivos igual o mayor a 0,7 es solamente del 8,7% y tienen una presencia menor a 1%. Estos datos parecen indicar que es más fácil identificar experiencias como positivas pero no existen claros indicadores para identificar experiencias negativas, al menos entre estos factores analizados individualmente.

La [Tabla 4](#) resume los puntos en la distribución mencionados en el análisis previo.

Tabla 4: Resumen de intenciones y sesiones acumuladas en función de proporción de casos positivos en la variable objetivo

Proporción de casos positivos	Percentil en función de intenciones	Percentil en Sesiones Acumuladas
0,25	25%	31,15%
0,4	48,8% (50%)	78,96%
0,5666666667	75%	88,54%
0,6	80%	88,81%
0,7	91,30%	99,52%

Para concluir esta sección, queda claro que existen relaciones entre las intenciones y la variable objetivo. También se observa que existe una proporción alta de intenciones con con bajo potencial predictivo que deberían poder ser efectivamente ignoradas por el algoritmo *XGBoost*. Y finalmente se advierte una posible limitación al usar intenciones como features que es que, al menos individualmente, no parecen indicar claramente experiencias negativas. Esta posible limitación podría quedar saldada en el modelo utilizando el resto de los features (la bolsa de palabras de mensajes del usuario) o relaciones secuenciales entre las intenciones aquí analizadas, dentro de los árboles de decisión que se creen en el algoritmo.

5 - Resultados

En esta sección se compartirán los resultados de los modelos entrenados para cada uno de los 17 puntos de corte con el detalle de las mejores combinaciones de hiperparámetros encontradas en cada caso. El foco estará en evaluar la variación en performance en relación a los diferentes puntos de corte. Además estarán expuestas medidas de importancia de features y se realizará un breve análisis de las diferencias observables en este aspecto entre los 17 modelos.

5.1 - Performance

5.5.1 - Modelos e hiperparámetros

Primero un repaso de la metodología, se construyeron 17 modelos *XGBoost* trabajando con un punto de corte diferente entre 1 y 17 para cada uno. El punto de corte determina la cantidad de mensajes desde el inicio de las sesiones dicho punto que usa cada modelo. Los 17 modelos entrenaron y predijeron sobre la misma selección de sesiones en los conjuntos de entrenamiento, validación y testeo. La diferencia en los datos utilizados por cada modelo estuvo en los features y valores que hacen a esas sesiones dependiendo del punto de corte correspondiente.

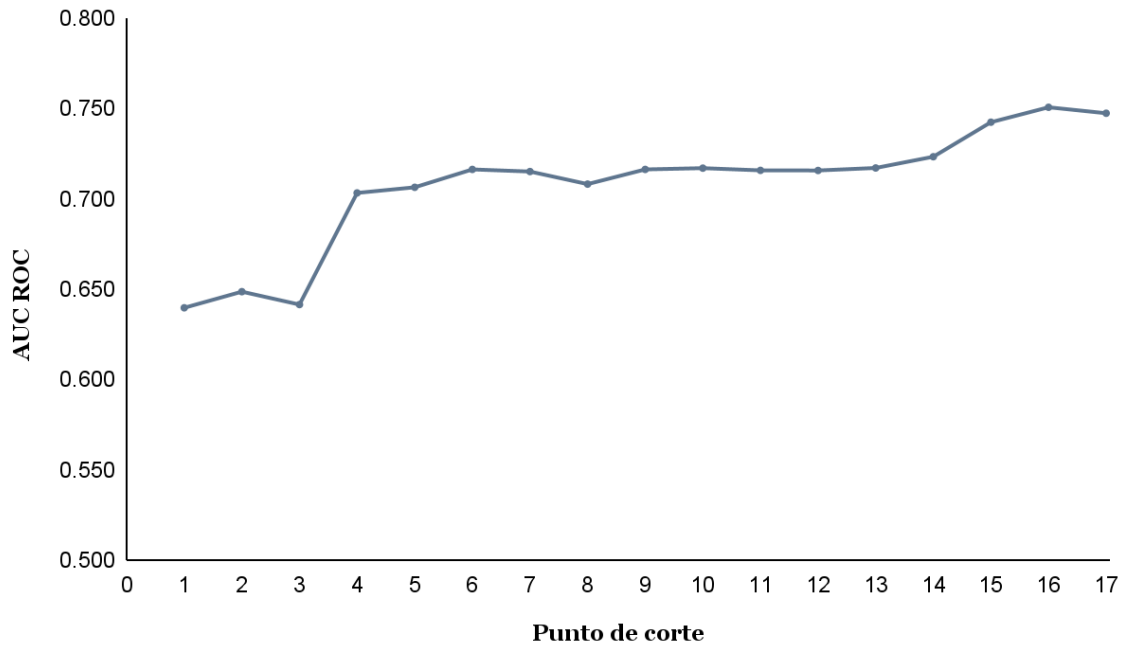
Para cada uno de los 17 puntos de corte se entrenaron 20 modelos con diferentes combinaciones aleatorias de hiper parámetros y se seleccionó la combinación con una mejor performance en AUC ROC sobre el conjunto de validación. La [Tabla 5](#) contiene, para cada modelo, el punto de corte, la combinación de hiperparámetros utilizada y el AUC en validación del mejor modelo encontrado.

Tabla 5: Hiperparámetros por modelo

Modelo	Punto de corte	learning_rate	max_depth	subsampling	colsampling_bytree	gamma	reg_alpha	reg_lambda	AUC validación
1	1	0,275	9	0,955	0,737	2,201	5,73E-06	5,74E-06	0,640
2	2	0,293	8	0,638	0,965	2,385	2,47E-06	5,20E-06	0,649
3	3	0,056	8	0,653	0,630	4,993	7,43E-06	5,64E-06	0,641
4	4	0,069	5	0,974	0,658	4,492	4,91E-06	3,99E-06	0,703
5	5	0,169	4	0,928	0,736	2,955	2,69E-08	6,16E-06	0,706
6	6	0,182	4	0,783	0,662	2,766	8,64E-06	2,37E-06	0,716
7	7	0,041	9	0,514	0,581	1,452	1,29E-06	9,23E-06	0,715
8	8	0,076	6	0,871	0,557	0,939	7,12E-06	9,00E-07	0,708
9	9	0,086	6	0,551	0,779	1,835	6,83E-06	8,75E-06	0,716
10	10	0,053	5	0,568	0,662	2,004	6,93E-06	1,65E-07	0,717
11	11	0,089	9	0,806	0,742	4,299	2,02E-07	5,09E-07	0,716
12	12	0,057	9	0,657	0,926	0,557	9,90E-06	6,42E-06	0,716
13	13	0,201	4	0,725	0,615	1,005	7,19E-06	2,35E-06	0,717
14	14	0,294	4	0,860	0,633	0,006	5,59E-06	4,88E-06	0,723
15	15	0,121	10	0,628	0,672	3,933	5,32E-06	5,06E-06	0,742
16	16	0,053	7	0,544	0,994	2,244	3,83E-06	8,24E-06	0,751
17	17	0,091	7	0,599	0,690	1,067	3,20E-06	9,91E-06	0,747

La [Figura 15](#) muestra la evolución de la métrica AUC ROC en los datos de validación en los modelos seleccionados para cada punto de corte o largo de la sesión utilizado.

Figura 15: AUC ROC sobre datos de validación por punto de corte



5.5.2 - Performance en testeo

Se evaluó la performance de cada modelo seleccionado sobre el conjunto de datos de testeo. La [Tabla 6](#) resume los resultados.

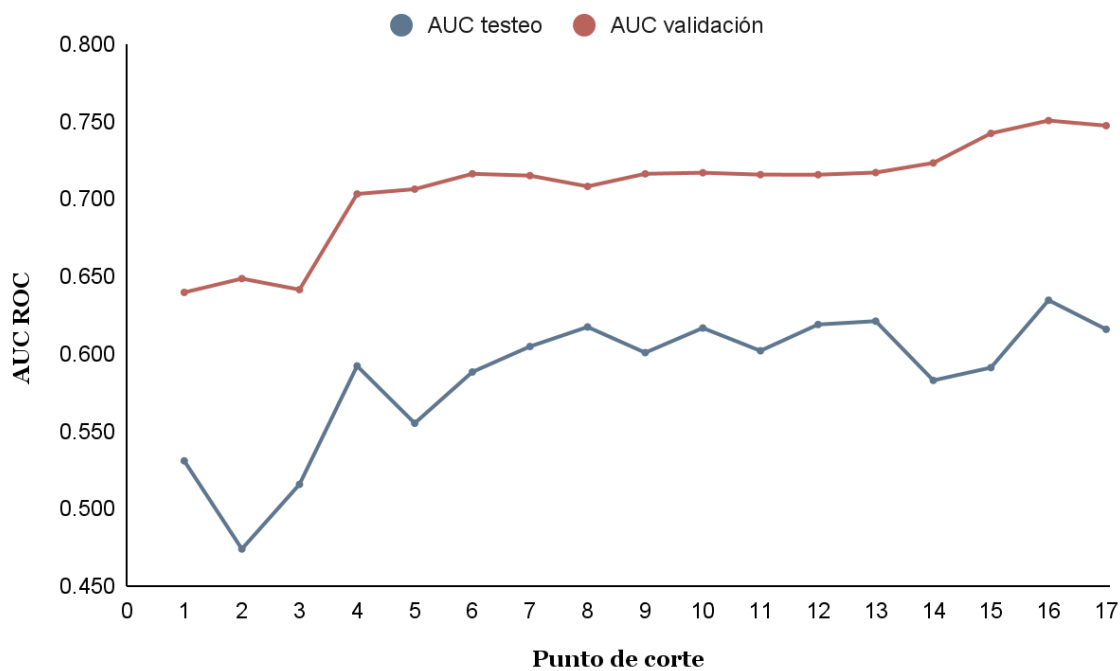
Tabla 6: Resumen de resultados por modelo

Modelo	Punto de corte	AUC validación	AUC testeo
1	1	0,640	0,531
2	2	0,649	0,474
3	3	0,641	0,516
4	4	0,703	0,592
5	5	0,706	0,555
6	6	0,716	0,588
7	7	0,715	0,605
8	8	0,708	0,617
9	9	0,716	0,601
10	10	0,717	0,617
11	11	0,716	0,602
12	12	0,716	0,619
13	13	0,717	0,621
14	14	0,723	0,583

15	15	0,742	0,591
16	16	0,751	0,635
17	17	0,747	0,616

Al igual que con los datos de validación, la [Figura 16](#) muestra la evolución de la métrica AUC ROC en los datos de testeo en relación al punto de corte utilizado.

Figura 16: AUC ROC sobre datos de testeo por punto de corte



Se puede observar una considerable diferencia entre la performance en validación y testeo para todos los modelos. Esto parece indicar que los modelos están realizando algo de *overfitting*, lo cual será explorado con mayor profundidad en las secciones de [5.3 Interpretación](#) y [6.1 Limitaciones y posibles mejoras](#)¹¹. Es rescatable que la performance en ambos conjuntos de datos tiende a mejorar a medida que aumenta el punto de corte, aunque la tendencia tiene mayor variabilidad en la performance en testeo.

¹¹ Las métricas de desempeño presentadas dentro del contexto del problema son suficientes para plantear la implementación de la solución propuesta siguiendo las recomendaciones que se brindarán en la sección 6.2 - Aplicaciones prácticas. Es importante rescatar que dichos resultados demuestran la existencia de cierta señal entre los datos y justifican futuros esfuerzos para optimizar la performance de este o futuros modelos posibles como se menciona en la sección 6.1 Limitaciones y posibles mejoras.

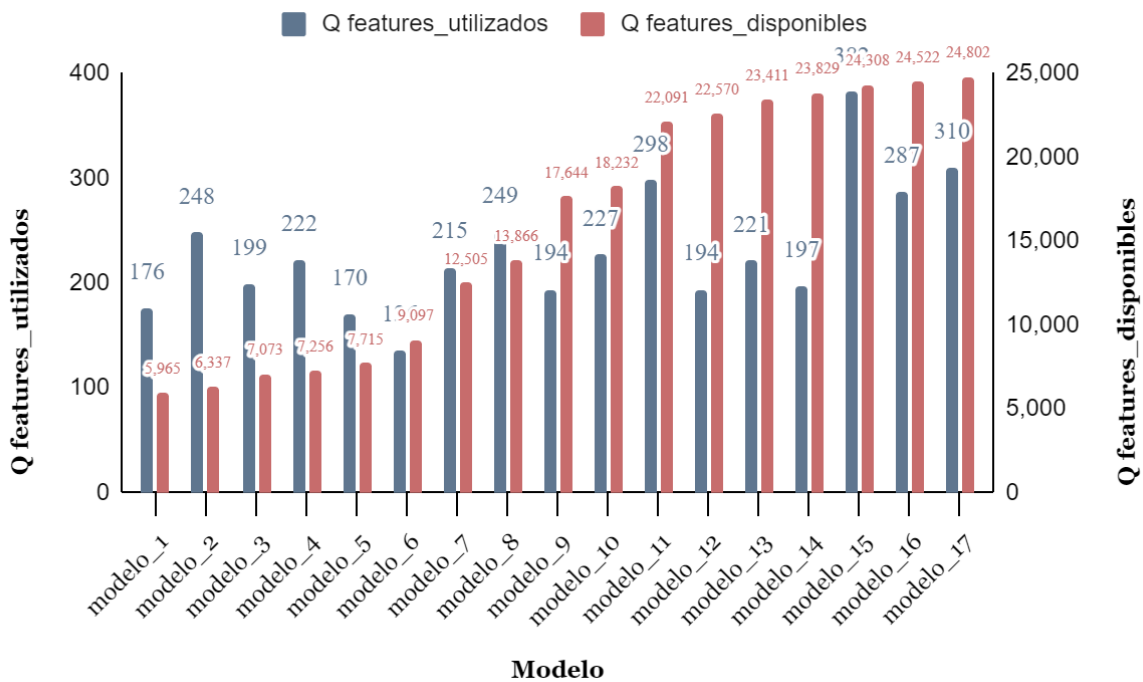
5.2 - Importancia de features

Debido a la metodología con la que se estructuraron las conversaciones en la base de datos (ver sección [3.3.2 - Ingeniería de atributos](#)), existe una amplia cantidad de features disponibles para ser utilizados por los modelos. La cantidad de features aumenta con el punto de corte, ya que conversaciones más cortas pueden ser resumidas con menos posibles intenciones dentro de la bolsa de intenciones y palabras dentro de la bolsa de palabras. Específicamente la base de datos utilizada por el modelo con punto de corte 1 cuenta con 5.962 features o columnas, esta cantidad crece junto al punto de corte hasta el largo de 17 donde las sesiones son representadas por 24.802 features.

Como se mencionó en la sección [3 - Metodología](#) el algoritmo utilizado (*XGBoost*) fue seleccionado en parte por su capacidad de ignorar features irrelevantes. Dada la cantidad de features disponibles, esta ventaja que ofrece el algoritmo resultó indispensable. En este sentido es interesante revisar cuántos y cuáles features terminaron siendo utilizados por el modelo e intentar entender cómo varía la importancia de cada feature entre los 17 modelos.

La [Figura 17](#) muestra la cantidad de features utilizados por cada modelo donde la numeración de cada modelo representa su punto de corte. A su vez se muestra la cantidad de features disponibles para cada uno de estos puntos de corte. La mínima cantidad de features utilizados fue de 136 en el modelo con punto de corte 6, mientras que el máximo fueron 382 en el modelo con punto de corte 15. El número de features promedio entre todos los modelos es de 231 y la mediana es 221. También parece existir una tendencia positiva en la relación entre punto de corte y cantidad de features utilizados, aunque es menos pronunciada que la relación entre punto de corte y cantidad de features disponibles en la base de datos previamente mencionada. Entre los 17 modelos solo se utilizan 546 features únicos entre los 24.802 disponibles.

Figura 17: Cantidad de features utilizados por modelo



La [Figura 18](#) indica la cantidad de modelos en los que se utiliza cada feature. Debido a la cantidad de features no es posible identificarlos individualmente en el eje horizontal. Por lo tanto esta información queda más claramente resumida en la [Figura 19](#) que indica el máximo número de modelos en los que se utiliza un porcentaje de features (sobre los 546 utilizados), empezando por 17 lo que significa que aparecen en todos los modelos disminuyendo hasta aparecer en solo 1. La figura permite ver, por ejemplo, que sólo el 6,04% de los features son utilizados por todos los modelos y el 16,48% de los 546 features sólo son utilizados exclusivamente por un modelo. Finalmente la [Figura 20](#) muestra la misma distribución que la [Figura 19](#) pero acumulando los porcentajes de features por cantidad de modelos en los que son utilizados para obtener el porcentaje de features que aparecen en al menos cada cantidad de modelos. De aquí se puede interpretar por ejemplo, que el 43,77% de los features son utilizados en al menos la mitad (redondeando a 8) de los modelos.

Figura 18: Distribución de presencia de features en modelos

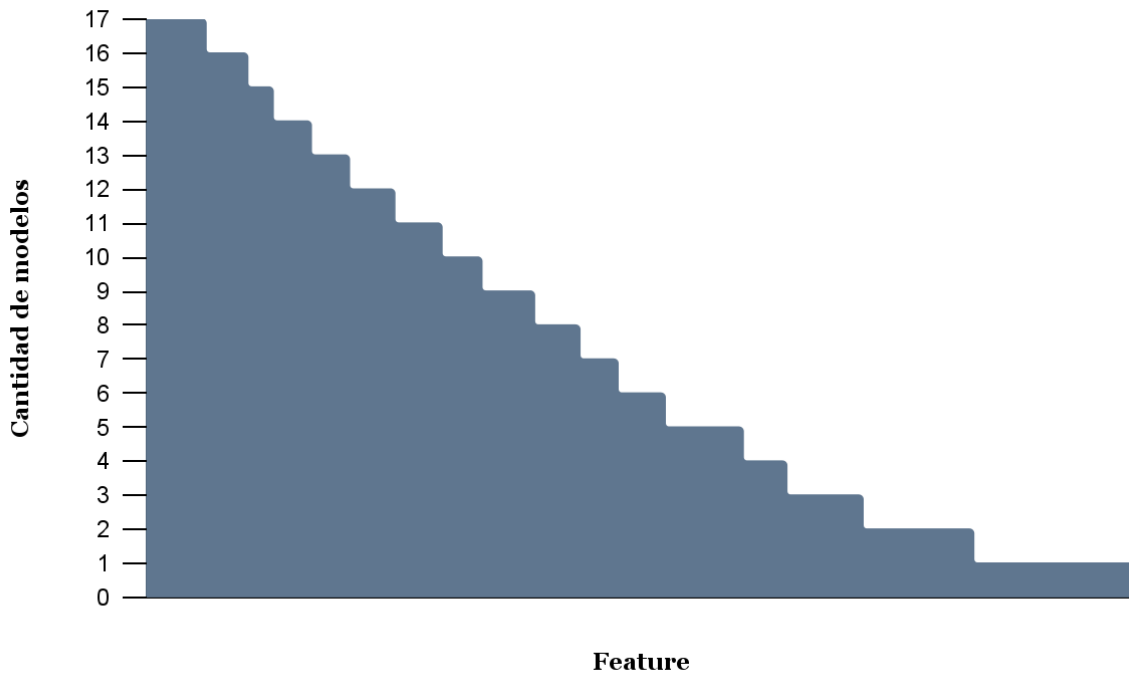
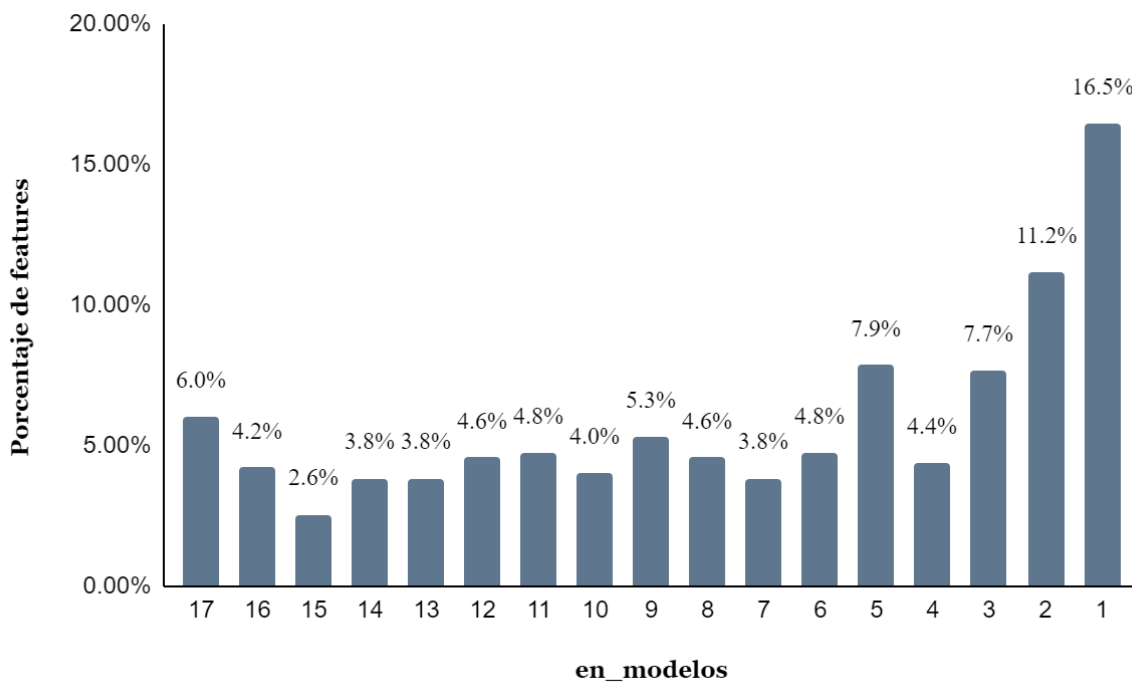
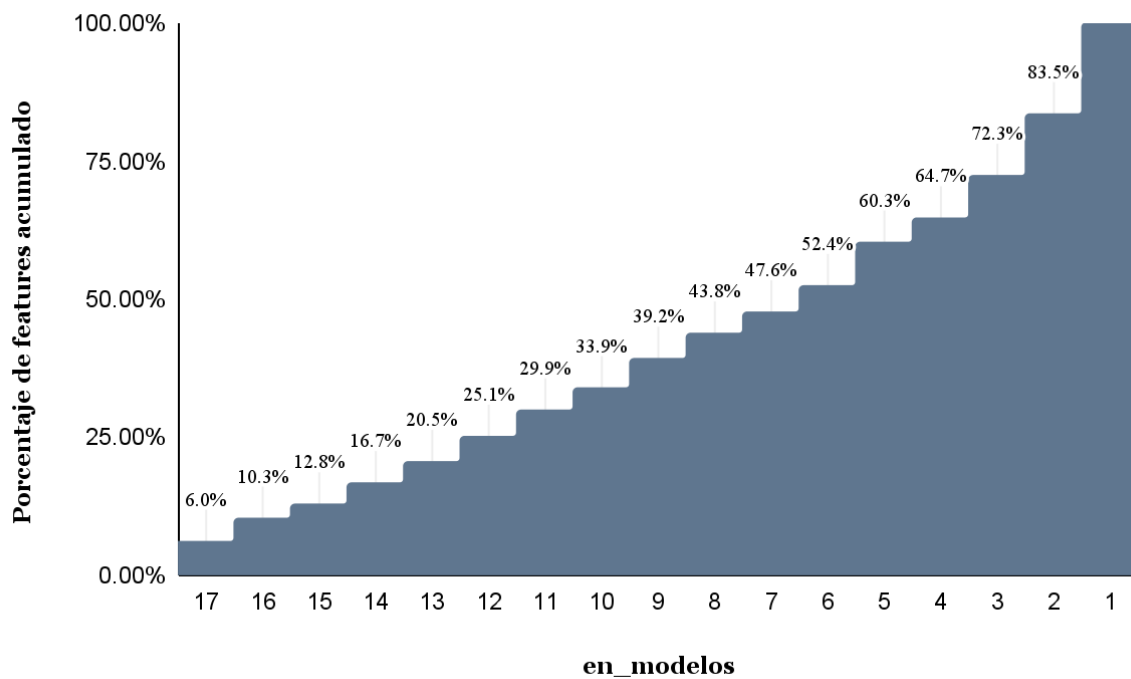


Figura 19: Porcentaje del total de features utilizados en función de la cantidad de modelos en los que aparecen¹²



¹² La distribución de la Figura 19 solo contempla los 546 features utilizados por al menos un modelo. No se graficaron los 24.256 features disponibles que no son utilizados por ninguno de los modelos para mantener la escala relativa de las participaciones.

Figura 20: Porcentaje acumulado del total de features utilizados en función de la cantidad de modelos en los que aparecen



A continuación se presenta la importancia de cada feature utilizando su peso en cada modelo. El peso de cada feature básicamente refleja la cantidad de veces que se utiliza para separar los datos dentro de los árboles de decisión de un modelo (Hastie et al., 2009). Dado que se entrenaron múltiples modelos que no utilizan los mismos features ni tampoco asignan necesariamente la misma importancia en el caso de los features compartidos, se optó por calcular la importancia promedio entre los modelos. Para incorporar el número de modelos en los que aparece un feature dentro de esta medición se calculó el promedio como la sumatoria de las importancias l , en cada modelo i , sobre la cantidad total de modelos n . Y se asignó importancia 0 a features en modelos en los que no aparecen.

Fórmula para calcular la importancia promedio entre modelos:

$$\frac{\sum_{i=1}^n l_i}{n}$$

La [Figura 21](#) muestra el valor de la importancia promedio para cada feature utilizado. Debido a la cantidad de features no es posible identificar en la figura las importancias individuales, pero es útil para entender la distribución de importancias por feature. Como complemento en la [Tabla 7](#) se pueden observar los 20 features con mayor importancia y la cantidad de modelos en los que aparece cada uno de ellos además la [Figura 21.1](#) muestra la distribución completa en escala logarítmica para visualizar diferencias más pequeñas entre las importancias promedio.

Figura 21: Importancia de features promedio para todos los modelos

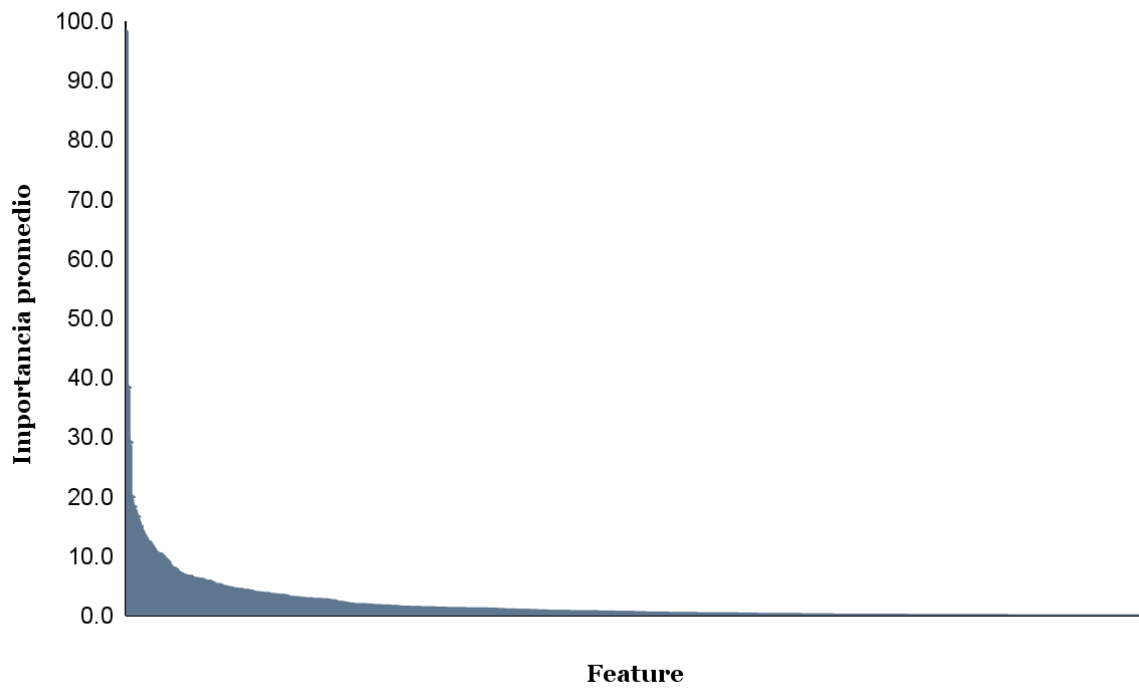


Figura 21.1: Importancia de features promedio para todos los modelos en escala logarítmica

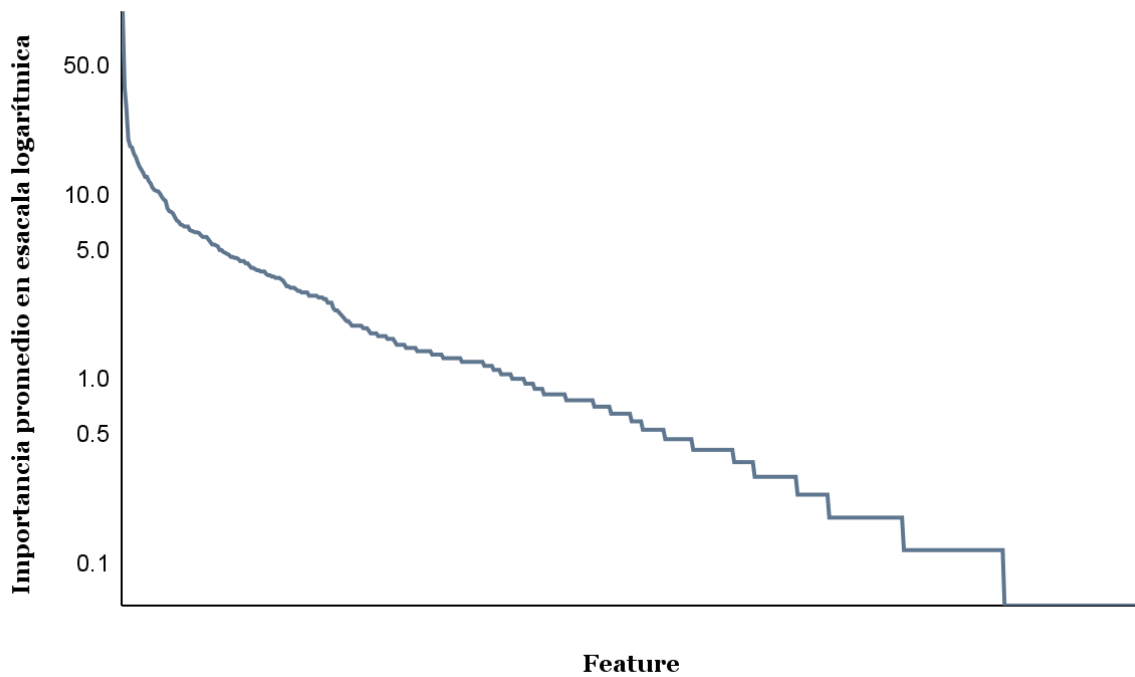


Tabla 7: Top 20 features entre todos los modelos

Feature	en_modelos	importancia_promedio
segundos_desde_ultima	17	99
usuario_retornante	17	38
Copia de Póliza menú *(Process Flow)	17	29
Derivación Solicitud Mail Interna (Process Flow)	17	20
CUANDO NO ENTIENDE - FLUJO INDEPENDIENTE	16	18
Pregunta DNI reimpresión de póliza	17	18
ESPERA ASIGNACIÓN 2º INTENTO	16	17
seguro	17	16
auto	14	15
Solicita reimprimir póliza	17	14
gracias	16	14
CP Apertura	16	13
saber	16	12
enviar	17	12
Derivación Solicitud DNI Interna (Process Flow)	17	12
tengo	17	11
Preguntar patente automotor para reimpresión de póliza	13	11
Servicio de reimpresión de póliza OK	15	11
ESPERA ASIGNACIÓN 3º INTENTO	12	10
Siniestro Denuncia Asegurado SI/NO (Process Flow)	17	10

Por último se presentan en la [Figura 22](#), [Figura 22.1](#) y la [Tabla 8](#) los mismos resultados que los expuestos en la [Figura 21](#), [Figura 21.1](#) y la [Tabla 7](#) pero en lugar de utilizar la importancia promedio, se presenta la importancia en el modelo con mayor punto de corte. Estos valores pueden ser más relevantes que los promedios calculados para todos los modelos ya que el modelo 17 cuenta con la mayor cantidad de información para entrenar y predecir. Es lógico pensar que los pesos de los features en el modelo más grande son los pesos definitivos aún cuando el modelo 17 no sea el que tenga la mejor performance. En la siguiente sección [5.3 - Interpretación](#) se explora esta idea en mayor profundidad.

Figura 22: Importancia de features en el modelo 17

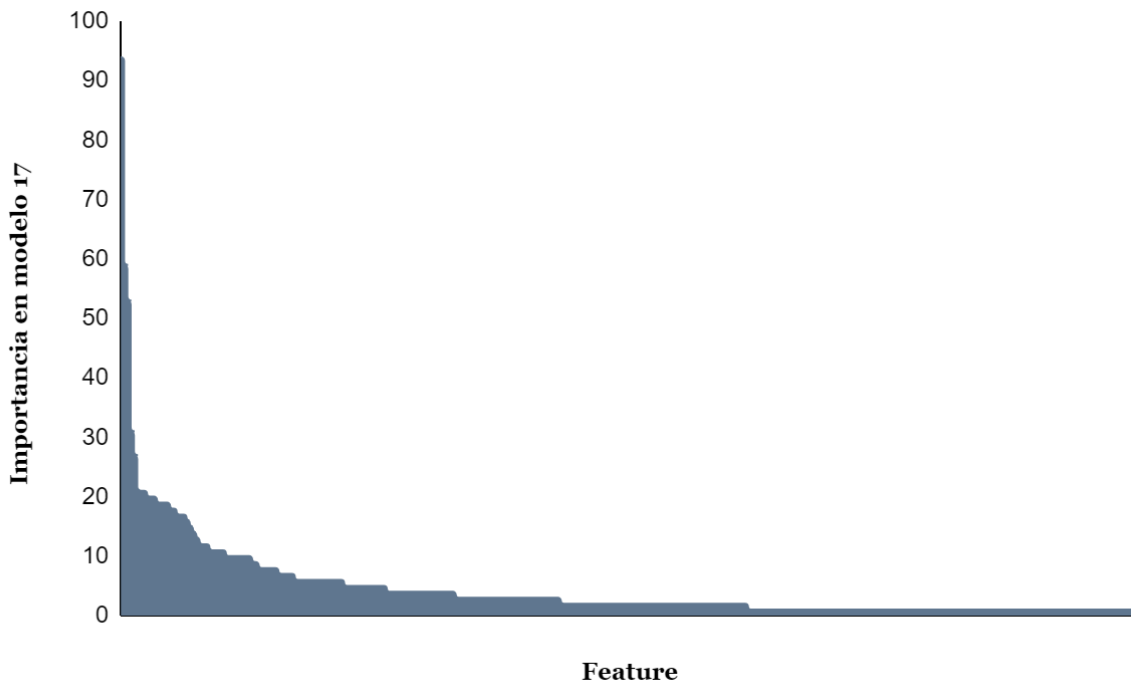


Figura 22.1: Importancia de features en el modelo 17 en escala logarítmica

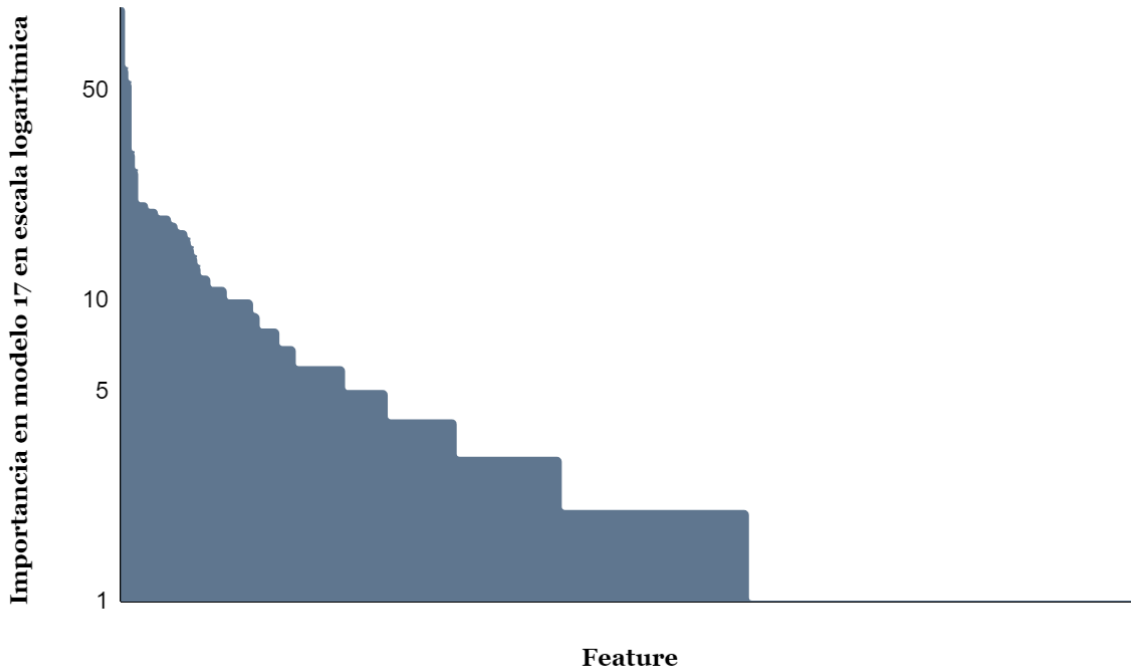


Tabla 8: Top 20 features en el modelo 17

Feature	Importancia
---------	-------------

segundos_desde_ultima	94
usuario_retornante	59
Servicio de reimpresión de póliza OK	53
Invocar servicio de reimpresión de póliza	31
Saludo bienvenida	27
CUANDO NO ENTIENDE - FLUJO INDEPENDIENTE	21
ESPERA ASIGNACIÓN 3º INTENTO	21
ESPERA ASIGNACIÓN 2º INTENTO	21
Copia Póliza Reimpresión *(Process Flow)	20
Solicitar datos para reimprimir póliza	20
Preguntar patente automotor para reimpresión de póliza	20
Servicio de reimpresión de póliza con ERROR	19
día	19
Solicita reimprimir póliza	19
CP Apertura	19
TEMAS MÁS CONSULTADOS	18
Copia de Póliza menú *(Process Flow)	18
Derivación Solicitud Mail Interna (Process Flow)	17
seguro	17
Siniestro Denuncia Asegurado Número Póliza (Process Flow)	17

5.3 - Interpretación

Los resultados presentados indican que los modelos logran predecir parcialmente la variable objetivo. El AUC ROC en el conjunto de testeo sugiere que, para la mayoría de los puntos de corte, las predicciones son al menos superiores a las de un clasificador que asigna clases de manera aleatoria. El valor más alto de AUC ROC alcanzado por cualquiera de los modelos es de 0,635. De los 17 modelos, 9 logran alcanzar valores iguales o superiores a 0,6. Aunque podrían esperarse mejores resultados para implementar la solución propuesta en un entorno productivo, los valores obtenidos demuestran que la metodología utilizada permite realizar predicciones que superan al azar. En la Sección [6.2 - Aplicaciones prácticas](#), se proporcionarán indicaciones sobre cómo se podría implementar y evaluar la solución propuesta. Con los resultados presentados, se justifica la implementación del modelo para su evaluación, siguiendo las recomendaciones detalladas en dicha sección.

Una posible interpretación sobre los datos de performance es que todos los modelos están realizando un grado de *overfitting* sobre los datos de validación. Si bien esta no es la interpretación tradicional del concepto de *overfitting*, es un fenómeno estudiado en la literatura sobre aprendizaje supervisado y un riesgo presente al entrenar este tipo de modelos (Ng, 1997). Entre los resultados expuestos se puede ver una caída en performance entre el AUC de los modelos sobre los datos de validación y el AUC sobre los datos de testeo. Aún cuando se utilizaron hiper parámetros de regulación para buscar que los modelos generalicen y no se sobreajuste a ningún conjunto de datos. En la [Tabla 5](#) se puede observar específicamente qué valores se utilizaron para cada modelo en los campos: *max_depth*, *subsample*, *colsample_bytree*, *gamma*, *reg_alpha*, *reg_lambda*. Más allá de los valores específicos, lo importante es que entre los 17 modelos se utiliza una variedad de valores diferentes para cada hiper parámetro, cuando estos están diseñados para limitar el *overfitting*¹³. Aún así la diferencia promedio entre la performance en validación y testeo para todos los modelos es de 0,121 con una diferencia mínima de 0,091 (modelo 8) y máxima de 0,175 (modelo 2). Estas diferencias no son deseables al evaluar la performance de cualquier modelo.

Dada la variedad de valores en hiper parámetros utilizados para evitar el *overfitting* es posible que las causas de esta diferencia entre validación y testeo sean más profundas que simplemente *overfitting*. Es posible que el modelo esté teniendo dificultades para generalizar a partir de la estructura de datos provista.

Un problema es que dado el uso de la técnica de bolsa de palabras sumado a la bolsa de intenciones, la base de datos tiene demasiadas dimensiones, sobre todo en relación a las 8.587 sesiones disponibles para utilizar en el proyecto. En el caso del punto de corte 17 la base cuenta con 24.802 features distintos de los cuales solo 327 representan intenciones del bot, 2 hacen referencia a la recurrencia del usuario y el resto son palabras en bolsa de palabras. Como se ve en las tablas [7](#) y [8](#) la mayoría de los features relevantes para los modelos son intenciones o variables de recurrencia y no palabras individuales. Parece que la variabilidad entre las posibles palabras utilizadas en cada sesión le dificultan al modelo encontrar algunas que sean lo suficientemente comunes

¹³ Es cierto que el tipo de *overfitting* que trata estos parámetros es la interpretación clásica de sobreajuste a los datos de entrenamiento y no a los de validación como se está planteando aquí. Sin embargo, a lo que se está refiriendo es al efecto positivo en la capacidad de generalización en modelos que se suele buscar con el uso de estos hiperparámetros, el cual aplicaría también para predicciones sobre datos “nuevos” como los de testeo.

como para utilizarlas en divisiones de árboles relevantes. Esto se da en parte por el proceso de tokenización utilizado y por la estructura de bolsa de palabras en sí misma.

En relación a las intenciones es posible que la advertencia levantada en la sección [4.4 - Relación entre intenciones y calificación](#) haya afectado la capacidad de generalización del modelo. Se observó que las intenciones individualmente se prestaban para identificar experiencias positivas pero no tanto así para determinar si una conversación es particularmente negativa. Se planteó que esta limitación podría ser superada por los modelos si lograban encontrar relaciones en base a la secuencialidad implícita de las intenciones o si pudiera aprovechar los mensajes de texto libre de los usuarios. En cuanto a los mensajes de los usuarios como complemento ya se observó que dada la variedad de posibles features en la bolsa de palabras, estos no llegaron a ser explotados efectivamente por el algoritmo. Y en cuanto a la posibilidad de encontrar relaciones secuenciales, si bien dentro de los árboles el algoritmo es capaz de tomar decisiones distintas usando el mismo feature dependiendo de en dónde esté en el árbol (en realidad dentro de los múltiples árboles) la estructura utilizada no revela la secuencialidad de las intenciones *a priori*. Es posible que sea demasiado exigente sobre el modelo pedir que recree estas relaciones secuenciales entre intenciones con los datos en formato de bolsa de intenciones, una limitación compartida con la técnica de bolsa de palabras o vectores raros cuando se utiliza para representar lenguaje natural (Jurafsky & Martin, 2000).

El último indicio de la dificultad para generalizar de los modelos está en la variedad de features utilizados por los diferentes modelos. Teniendo en cuenta que todos los modelos se entrenaron y evaluaron con las mismas sesiones en cada conjunto de datos, se podría esperar que si existen features relevantes, la mayoría de los modelos deberían utilizarlos. Sin embargo como se puede extraer de las figuras [19](#) y [20](#) solo el 20,51% de los features utilizados por al menos un modelo aparecen en tres cuartos de los modelos (13 de 17) mientras que el 39,7% son utilizados por 5 modelos o menos. Es esperable que parte de los features no sean compartidos por todos los modelos dadas las diferencias en los puntos de cortes. Pero incluso cuando se comparan sólo los modelos 15, 16 y 17 que tienen acceso a un rango de features similar, se encuentra que solo el 51,15% de los features utilizados por estos son compartidos entre los 3, con 23,27% compartido por 2 modelos y 25,58% apareciendo únicamente en uno de los 3. Lo que se

puede interpretar de estos datos es que cada modelo parece estar encontrando diferentes maneras de ajustarse a los datos de entrenamiento. El hecho de que sus soluciones parecen no tener tanto en común puede indicar que son soluciones específicas y no generales. Esto se ve reflejado en la caída en performance frente a los datos de testeo.

Para concluir esta sección de interpretación de resultados, se encontró que los modelos logran capturar parte del problema y predecir mejor que el azar. Sin embargo, se observó una caída en performance entre validación y testeo común a los 17 modelos entrenados. Esta caída parece estar explicada por las limitaciones del modelo y de la estructura de datos utilizada para representar conversaciones. Los modelos lograron reducir las dimensiones utilizadas y ajustarse moderadamente bien a los datos de entrenamiento y validación. Sin embargo, las limitaciones de la base de datos disponible y estructura utilizada parecen haber impedido al modelo generalizar y alcanzar resultados en testeo similares a los alcanzados en validación.

6 - Conclusiones

Para resumir, en este trabajo se buscó encontrar un modelo que permita clasificar experiencias conversacionales negativas, utilizando la información disponible dentro de los primeros intercambios de cualquier conversación. Dicho modelo serviría para intervenir estas conversaciones categorizadas como negativas con operadores humanos y mejorar la experiencia de usuario que ofrece el canal.

Para llegar a este objetivo primero se analizaron los datos disponibles y su estructura. En base a los datos se determinó, tanto el algoritmo a utilizar (*XGBoost*), como las transformaciones necesarias para adaptar la estructura original de los mismos y generar los features que utilizaría el algoritmo. En base al análisis de datos se evaluó la posible capacidad predictiva de los principales features que se generaron. Entre las transformaciones se utilizó una estructura personalizada aplicando una lógica similar a la de bolsa de palabras sobre las intenciones que utiliza el chatbot para funcionar. La mayoría de los features más importantes en los modelos entrenados resultaron de esta metodología. Además se buscó complementar esta estructura con técnicas clásicas,

como una bolsa de palabras para los mensajes del usuario de texto libre e información sobre la recurrencia de los usuarios.

Del análisis de la base de datos también surgieron observaciones valiosas para justificar la naturaleza de la intervención en las conversaciones propuestas. Primero, que los operadores humanos parecen brindar una calidad de atención superior, por lo tanto podrían ser usados como herramienta de corrección de experiencias negativas. Segundo, que el nivel de satisfacción con el chatbot tiende a disminuir en conversaciones más largas, indicando que una intervención temprana sería beneficiosa también en este sentido.

La definición del momento de intervención es una parte fundamental de la solución propuesta. Para esto se analizó la distribución de conversaciones por largo y se definió un punto máximo en 17 mensajes para incluir la mayor proporción de conversaciones posibles en el ejercicio. Además, se entrenaron 17 diferentes modelos para poder evaluar la evolución en el desempeño de estos en función de su punto de corte.

Finalmente se analizaron en profundidad los resultados de performance para los modelos entrenados. El AUC ROC de los modelos indica que la mayoría de estos captan parte de la complejidad del problema, generando predicciones que son mejores que el azar. Los resultados alcanzados justifican implementar una versión de estos modelos en la práctica si se lo hiciera bajo las recomendaciones que se brindarán en la sección [6.2 - Aplicaciones prácticas](#). Sin embargo, los resultados también indican ciertas limitaciones del algoritmo y las metodologías utilizadas que fueron explorados en la sección [5.3 - Interpretación](#) y serán revisadas en la siguiente sección [6.1 - Limitaciones y posibles mejoras](#).

6.1 - Limitaciones y posibles mejoras

Como se observó en la sección [5 - Resultados](#) es posible que la estructura planteada para resumir las conversaciones no sea óptima para lograr identificar experiencias de usuario negativas. Las técnicas utilizadas y la estructura de datos generada no son incorrectas *a priori*. Aún así parece que la combinación de la baja cantidad de sesiones en relación a las dimensiones totales de base de datos y la distribución de la variable

objetivo relativa a los features utilizados resultaron ser un desafío que no se ajustó al algoritmo utilizado.

El ejercicio realizado parece indicar que los features comunes en la mayoría de las sesiones, las intenciones, no son suficientes para identificar con precisión la calidad de la experiencia. También parece mostrar que la información sobre los mensajes del usuario en lenguaje natural no termina siendo aprovechada correctamente por el algoritmo, debido a su dispersión en múltiples dimensiones en la estructura de bolsa de palabras utilizada. Este modelo se beneficiaría de tener acceso a la información disponible en estos mensajes en un formato más reducido, con menos dimensiones, buscando que sean más comunes entre los registros para poder ser utilizados para tomar decisiones por los árboles del algoritmo. De este modo sería mejor complementada la información que se extrae de las intenciones. En este sentido, como futura mejora se podría pensar en utilizar herramientas más modernas de procesamiento de lenguaje natural para sumar las descripciones de los mensajes del usuario como features al modelo. El uso de vectores densos para representar palabras como en el caso de word2vec (Jurafsky & Martin, 2000) combinado con técnicas de captación de frases (Mikolov et al., 2013), deberían ayudar a capturar la información de los mensajes de texto libre.

Siguiendo esta línea, se podría intentar un enfoque distinto al planteado y directamente utilizar las conversaciones en un formato más cercano a su estado original y utilizar un algoritmo dedicado al procesamiento de lenguaje natural como BETO, la versión en español de BERT, Bidirectional Encoder Representations from Transformers (Devlin et al., 2018). Este tipo de algoritmo no utilizaría la estructura de intenciones planteada, en su lugar tomaría las conversaciones textuales para clasificarlas. Una ventaja de este modelo es que está pre entrenado con grandes cantidades de texto lo que le permite capturar información contextual bidireccional, es decir, entender una palabra en función de las palabras que la rodean tanto antes como después. De esta forma sería más sensible a las diferencias que presentan las conversaciones, ya que podría capturar diferencias semánticas complejas y aprovecharlas para categorizar entre experiencias negativas y no negativas¹⁴.

¹⁴ También se podría justificar la implementación de otros algoritmos como redes neuronales LSTM (Long Short-Term Memory). Sin embargo dados los esfuerzos ya realizados y el considerable trabajo en estructurar los datos para probar el algoritmo XGBoost, se recomendaría que futuras pruebas se

En un principio, no se emplearon estos métodos debido a que, dada la estructura de las conversaciones, se consideró que una solución computacionalmente menos compleja podría ser suficiente para abordar la necesidad expuesta, tal como se describe en la Sección [3 - Metodología](#). Sin embargo, a la luz de los resultados observados, parece necesario que el proyecto asuma el costo de la complejidad y utilice modelos pre-entrenados basados en transformers, como BETO, ajustándolos a la tarea deseada.

6.2 - Aplicaciones prácticas

A continuación está explicado por qué se cree que la intervención planteada mejoraría el nivel de satisfacción de los usuarios en el canal y se brindan recomendaciones para la aplicación práctica de los modelos discutidos. Los argumentos y recomendaciones aquí expuestos aplican tanto para los modelos entrenados en este ejercicio como para cualquier modelo futuro (como los que se plantean en la sección anterior).

En la sección [4 - Análisis exploratorio de los datos](#) se encontraron indicios de que la intervención planteada en este trabajo puede llevar a una mejor calificación de satisfacción del servicio en general. Se observó que las calificaciones al bot empeoran en conversaciones más largas y también que en general los operadores reciben mejores calificaciones, se asume que esto se da, al menos en parte por la calidad superior de su atención. Dado este contexto, la propuesta de derivar prematuramente sesiones que se espera que califiquen negativamente al bot, lógicamente debería llevar a mejores resultados en los indicadores de calidad que utiliza el servicio.

La solución propuesta sería un complemento al flujo de derivación que ya existe en el canal, que permitiría un rescate de experiencias negativas. Así se evitaría depender de que el usuario navegue hasta la derivación, aumentando potencialmente su grado de insatisfacción en el proceso. Además se disminuye el riesgo de que el usuario abandone la conversación sin haber encontrado lo que buscaba.

En la sección de [5 - Resultados](#) también se observó que la performance de cada modelo tiende a mejorar con puntos de corte mayores. Es razonable asumir que este también

concentren en algoritmos más modernos ya dedicados al procesamiento de lenguaje natural, como es el caso de BETO. Si el próximo objetivo es optimizar métricas de performance no se recomienda continuar profundizando esfuerzos en adaptar algoritmos dedicados a trabajar con estructuras clásicas como lo son las series temporales para las LSTM o la bolsa de palabras e intenciones utilizadas junto a XGBoost en este trabajo.

sería el caso entre modelos como los mencionados en la sección anterior. La mejora se debe a que puntos de corte mayores ofrecen más información sobre las sesiones independientemente del algoritmo que las procese. En este sentido, se puede recomendar implementar más de un modelo funcionando en diferentes puntos de corte. Un ejemplo arbitrario podría ser en los mensajes número 6, 11 y 17 utilizando un umbral de decisión más exigente (que disminuya los falsos positivos) en el primero y otros progresivamente más balanceados en los siguientes. De esta manera las sesiones tendrían más de una oportunidad de ser rescatadas por la derivación y se lograría abreviar aún más las experiencias negativas. La variación en el umbral de decisión se usaría para complementar el hecho de que los modelos con puntos de corte menores tienen peores resultados en AUC y así intentar minimizar los falsos positivos.

Para implementar cualquiera de los mecanismos propuestos, se recomienda realizar un test AB en su lanzamiento para validar la hipótesis de que mejoraría la calidad de la experiencia. Sería un experimento simple donde los usuarios aleatoriamente serían presentados con la experiencia original (grupo de control) o con la nueva experiencia y el mecanismo de derivación inteligente planteado (grupo de tratamiento). Una vez recolectados los datos de una cantidad estadísticamente significativa de conversaciones en ambos grupos, será posible comparar los indicadores de calidad entre ambos y validar la hipótesis planteada.

6.3 - Conclusión

En este trabajo se llegó a entrenar modelos que capturan parte de la complejidad del problema, prediciendo mejor que el azar. Estos podrían ser implementados bajo el esquema de aplicación y validación presentado en la sección anterior. Aun así, lo más valioso de este ejercicio es el *pipeline* de trabajo y el análisis exploratorio de los datos que sirve para justificar futuros esfuerzos en el desarrollo de la solución propuesta en este trabajo.

Gran parte de la metodología utilizada puede ser aprovechada para entrenar futuros modelos con otros algoritmos como los planteados en la sección [6.1 - Limitaciones y posibles mejoras](#). Principalmente, la lógica utilizada para la selección de conversaciones y el ejercicio de entrenamiento con diferentes puntos de corte. Además, los resultados

alcanzados en los modelos entrenados presentan un panorama optimista para futuros modelos posibles y pueden ser utilizados como *benchmark* para los próximos esfuerzos.

En conclusión, este trabajo ejemplifica cómo, utilizando los datos disponibles de las conversaciones en un canal de atención al cliente, es posible y valioso un ejercicio como el expuesto. Siguiendo el mismo *pipeline* de: análisis exploratorios para obtener indicios sobre posibles mejoras en el servicio; utilizar modelos de *machine learning* dentro de las posibles soluciones a los puntos de mejora identificados; y generar estrategias de implementación que permitan validar la efectividad de las intervenciones. De esta forma es posible avanzar en la prestación de un servicio de mayor calidad y en el aumento del nivel de satisfacción del cliente.

7 - Bibliografía

- Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*.
<https://doi.org/10.5555/2188385.2188395>
- Bird, S., & Loper, E. (2004). NLTK: The Natural Language Toolkit. En *Proceedings of the ACL Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics. <https://aclanthology.org/P04-3031/>
- Bockhorst, J., Yu, S., Polanía, L. F., & Fung, G. (2017). Predicting self-reported customer satisfaction of interactions with a corporate call center. En *Lecture Notes in Computer Science* (pp. 179-190).
https://doi.org/10.1007/978-3-319-71273-4_15
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E.J., Litwin, M., Gray, S., Chess, B., Clark, J., Brener, C., McCandlish, S., Radford, A., Sutsveker, I., Amodei, D. (2020). Language models are Few-Shot learners. *arXiv (Cornell University)*. <https://arxiv.org/pdf/2005.14165.pdf>
- Chen, T., & Guestrin, C. (2016). XGBoost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
<https://doi.org/10.1145/2939672.2939785>
- Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv (Cornell University)*. <https://arxiv.org/pdf/1810.04805v2>
- Feine, J., Morana, S., & Gnewuch, U. (2019). Measuring service encounter satisfaction with customer service chatbots using sentiment analysis. *Wirtschaftsinformatik und Angewandte Informatik*, 1115-1129.
<https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1248&context=wi2019>
- Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5). <https://doi.org/10.1214/aos/1013203451>
- Gil, S., & Rodríguez, E. (2001). *Física re-Creativa*. Pearson Educación.
https://books.google.com.cu/books/about/F%C3%ADsica_re_creativa_i_e_recreativa.html?id=2CRsPgAACAAJ&hl=es-419
- Hapke, H., Lane, H., & Howard, C. (2019). *Natural language processing in action*. Manning Publications.
<https://www.oreilly.com/library/view/natural-language-processing/9781617294631/>
- Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). The elements of statistical learning. En *Springer series in statistics*.
<https://doi.org/10.1007/978-0-387-84858-7>

- Jurafsky, D., & Martin, J. H. (2000). Speech and Language processing: An introduction to natural language processing, computational linguistics, and speech recognition. En *Prentice Hall eBooks*.
<https://nats-www.informatik.uni-hamburg.de/pub/CDG/JurafskyMartin00Comments/JurafskyMartin00-Review.pdf>
- Lundberg, S., & Lee, S. I. (2017, 22 mayo). *A unified approach to interpreting model predictions*. arXiv.org. <https://arxiv.org/abs/1705.07874>
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Neural Information Processing Systems*, 26, 3111-3119.
<http://ptrckprry.com/course/ssd/reading/Miko13.pdf>
- Müller, A., & Guido, S. (2016). *Introduction to Machine Learning with Python: a guide for data scientists*. <http://cds.cern.ch/record/2229831>
- Ng, A. Y. (1997). Preventing overfitting of Cross-Validation data. *International Conference on Machine Learning*.
<https://ai.stanford.edu/~ang/papers/cv-final.pdf>
- Nguyen, Q. N., Sidorova, A., & Torres, R. (2022). User interactions with Chatbot interfaces vs. menu-based interfaces: An empirical study. *Computers in Human Behavior*, 128, 107093. <https://doi.org/10.1016/j.chb.2021.107093>
- Van Ryzin, J., Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1986). Classification and regression trees. *Journal of the American Statistical Association*, 81(393), 253. <https://doi.org/10.2307/2288003>