uc3m | Universidad Carlos III de Madrid

Máster en Ciencias Actuariales y Financieras
Curso académico 2021-2022

Trabajo Fin de Máster

Modelo predictivo de venta cruzada en productos de Vida y Salud: Random Forest vs XGBoost

---

Marcos Sánchez Sardaña

Tutores

Dr. José Miguel Rodríguez-Pardo

Dr. Jesús R. Simón del Potro

Madrid, junio de 2022

DETECCIÓN DEL PLAGIO

La Universidad utiliza el programa **Turnitin Feedback Studio** para comparar la originalidad del trabajo entregado por cada estudiante con millones de recursos electrónicos y detecta aquellas partes del texto copiadas y pegadas. Copiar o plagiar en un TFM es considerado una **Falta Grave**, y puede conllevar la expulsión definitiva de la Universidad.

RESUMEN/ABSTRACT

Actualmente la tecnología avanza muy deprisa existiendo nuevos progresos como la Inteligencia Artificial (IA) y el *machine learning* que permiten a los algoritmos analizar el *Big Data* mucho más rápido y mejor que los humanos. Muchas empresas aseguradoras aún desconocen este tipo de tecnologías, o se encuentran aún en fase de implementación, sin saber muy bien cómo emplearlas para mejorar su nivel de ventas y suscripción de pólizas. Por ello, una gran oportunidad para optimizar los recursos de la empresa, y mejorar las ventas pasa por aplicar estas tecnologías a las campañas de venta cruzada.

Este trabajo presenta un estudio comparado de dos modelos predictivos de venta cruzada en el contexto del mundo asegurador entre productos de Vida y Salud. En concreto, el presente estudio comparado se ha realizado sobre dos sistemas de aprendizaje automático supervisados (Random Forest y XGBoost) aplicados a la cartera de la compañía aseguradora ASESUISA – Seguros SURA de El Salvador con los objetivos, tanto de desarrollar una metodología, como de comparar ambos modelos, y determinar que algoritmo predice mejor los potenciales clientes objetivos en el caso de una campaña de venta cruzada y así optimizar los recursos y esfuerzos de las compañías. Para ello se han empleado dos carteras durante el período de 2015 a 2020, una de Vida Individual y otra de Salud.

Los resultados empíricos proporcionan conclusiones interesantes para la venta cruzada como el perfil de los clientes. Aun así, independientemente de los resultados del sistema de aprendizaje automático la dirección de la compañía aseguradora es la encargada de iniciar una campaña de venta cruzada, y en última instancia son los esfuerzos de los empleados los que los que determinan el éxito o fracaso de la misma. Por lo tanto, este estudio adopta una perspectiva desde el punto de vista cuantitativo y de análisis de datos ya que pretende determinar los potenciales clientes objetivo y no establecer la estrategia de venta más adecuada. Adicionalmente se proporcionan posibles futuras líneas de investigación y metodologías a aplicar relacionados con la venta cruzada y el up-selling en el ámbito asegurador.

Palabras clave / Key words: Bosque aleatorio, XGBoost, venta cruzada, algoritmo, bagging, boosting, árbol de decisión, Big Data.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

MODELO PREDICTIVO DE VENTA CRUZADA EN PRODUCTOS DE VIDA Y SALUD: RANDOM FOREST vs XGBOOST

1. INTRODUCCIÓN

En diversos campos del conocimiento y la ciencia como la meteorología, la medicina o las finanzas, por citar algunos, los expertos intentan predecir un resultado basándose en observaciones o mediciones anteriores. Por ejemplo, los meteorólogos intentan predecir el tiempo de los próximos días a partir de las condiciones climáticas de los días anteriores. En medicina, se recogen mediciones e información de cada paciente como la presión arterial, la edad o el historial médico para diagnosticar el estado de los mismos, y así en infinidad de disciplinas. En todos estos casos, el objetivo es la predicción de una variable de respuesta basada en un conjunto de variables predictoras observadas.

Durante siglos, estos problemas siempre se han resuelto derivando marcos teóricos a partir de unos principios básicos y una gran acumulación de conocimientos para modelar, analizar y comprender el fenómeno estudiado. Siguiendo con el ejemplo de los médicos, saben por experiencia que los pacientes de edad avanzada que sufren infartos de miocardio y tienen la presión arterial baja suelen ser de alto riesgo. En este aspecto, el mundo empresarial no se queda atrás, y una gran mayoría de empresas realizan predicciones y pronósticos con expectativas de cifras de ventas que intentan batir año tras año para aumentar así sus beneficios y contentar a sus accionistas. Para poder alcanzar esos objetivos de ventas, las empresas llevan a cabo campañas de ventas agresivas, como las campañas de venta cruzada, e invierten muchos recursos para tratar de cumplir sus objetivos.

Sin embargo, cada vez las cuestiones que se intentan predecir son más y más complejas y requieren de más datos y patrones no fácilmente reconocibles por los humanos. Pero desde mediados del siglo XX, gracias al avance de las nuevas tecnologías y al desarrollo de la Inteligencia Artificial y el aprendizaje automático ("*machine learning*") los ordenadores son capaces de aprender de los datos y de descubrir por sí mismos la estructura predictiva de los problemas. Las técnicas y los algoritmos derivados del campo del aprendizaje automático se han convertido en una poderosa herramienta para el análisis de datos complejos. Ejemplos famosos que han supuesto hitos han sido el uso de árboles de decisión en el análisis estadístico de los datos que condujeron a la detección del Bosón de Higgs en el CERN (Chatrchyan et al., 2012) o la implementación de diversas técnicas de aprendizaje automático para la construcción del sistema de inteligencia artificial IBM Watson (Ferrucci et al., 2010) entre otros.

Retomando el contexto empresarial, las corporaciones aplican las técnicas previamente mencionadas de aprendizaje automático para tratar de predecir con gran exactitud el número de ventas que se van a obtener en un año, o qué clientes tiene una mayor probabilidad de comprar un determinado producto. Esto permitiría a las empresas destinar

de forma más eficiente sus recursos obteniendo así mejores resultados económicos con un menor coste de capital, tiempo y esfuerzo.

Centrándose concretamente en el ámbito de la venta cruzada, muchos de los estudios existentes se enfocan en la perspectiva del marketing de emplear un sistema informático de gestión de clientes para ayudar a identificar estas oportunidades de venta cruzada. Este caso se dio en el estudio titulado "C*ross-Selling in the Financial Sector: Customer Profitability is Key*" de Jarrar y Neely (2002) que fue un éxito en la conversión de ventas cruzadas, pero los autores concluyeron que los resultados se debían a la capacidad de los empleados para comprender las necesidades de los clientes, y no al sistema informático que debía apoyar la iniciativa. Posteriormente, Kamakura et al. (2005) propusieron un modelo que estimaba la propensión de los clientes a comprar una categoría de productos y el momento en que lo iban a hacer, basándose en los momentos de compra anterior. Sin embargo, existen pocas publicaciones que traten la venta cruzada desde una perspectiva de la ciencia de datos, y menos aún en el mundo asegurador.

En este contexto, el objetivo de esta publicación es proporcionar un análisis completo y comparativo de dos algoritmos predictivos basados en árboles de decisión aplicado a un caso empírico de una campaña de venta cruzada en el ámbito del Seguro. Por un lado, se tendría el algoritmo conocido como "*Random Forest*" o Bosque Aleatorio, creado por Leo Breiman (2001) basándose en el trabajo de Tin Kam Ho (1995), el cual según estudios como "*Do We Need Hundreds of Classifiers to Solve Real World Classification Problems?"* (Fernández-Delgado, 2014) donde se evaluaban diferentes clasificadores se llegó a la conclusión de que los bosques aleatorios alcanzaban la mejor precisión global, aparte de ser de los más rápidos. Por otro lado, se tendría el algoritmo recientemente descubierto por Chen and Guestrin, (2016) conocido como XGBoost que ha ganado mucha popularidad gracias a obtener muy buenos resultados en las competiciones organizadas por la plataforma de machine learning "*Kaggle*" (The Machine Learners, 2021). Estos dos sistemas de aprendizaje automático se aplicarán sobre una base de datos de clientes de la Aseguradora Suiza Salvadoreña (ASESUISA), de El Salvador, para contrastar empíricamente que algoritmo es mejor prediciendo los potenciales clientes de una campaña de venta cruzada, obtener el perfil del cliente y así optimizar la asignación empresarial de recursos.

2. VENTA CRUZADA

Hoy día los hábitos de consumo de los clientes son drásticamente diferentes a los de principios de la primera década del siglo XXI, destacando por encima de todo el consumo digital, ya sea de contenidos (periódicos, series o películas) como de bienes y servicios (comprar unas entradas de cine, un seguro o unos zapatos).

Tal como los hábitos de los consumidores cambian, los de los anunciantes también deberían cambiar para adaptarse a esta nueva modalidad de consumo y conseguir un mayor impacto en el cliente y atraer su atención. Además, aprovechar así todas las posibilidades que brinda esta nueva forma de consumo. Por ejemplo, se observa que la publicidad directa y los anuncios de prensa escrita desempeña cada vez un papel menor al igual que la publicidad masiva y el spam, y cada vez se ven más anuncios en redes sociales.

Los anuncios están cambiando no sólo hacia la digitalización, sino también hacia la personalización, impulsada por algoritmos cada vez más sofisticados y modelos predictivos que analizan los datos de las transacciones y las tendencias de los medios digitales (por ejemplo, qué temas están de moda en las redes sociales o qué búsquedas ha realizado una persona en internet). Realidades palpables de este cambio se ven en empresas archiconocidas como son Amazon.com, Inc o Netflix, Inc, entre otras, que llevan a cabo de forma exitosa recomendaciones a sus clientes empleando algoritmos.

Esto remarca la importancia del análisis de datos de ventas y de los clientes para poder implementar de forma más eficiente y eficaz campañas de marketing o técnicas de ventas que permitan maximizar el beneficio que obtiene la empresa por cada consumidor.

2.1 Concepto

Antes de desarrollar que es un modelo predictivo de venta cruzada es necesario definir el concepto de aquello que se quiere predecir, en este caso, la "venta cruzada". Por eso, a continuación, se estudiará el termino de venta cruzada, siendo el elemento principal el concepto de venta.

Una venta es un término muy cotidiano ya que se percibe diariamente en las sociedades occidentales, no así tanto en las orientales donde es más frecuente el trueque. Jurídicamente hablando este concepto viene definido en el art. 1445 del Código Civil del Ordenamiento Jurídico español donde se define el contrato de compraventa como: *"por el contrato de compra y venta uno de los contratantes se obliga a entregar una cosa determinada y el otro a pagar por ella un precio cierto, en dinero o signo que lo represente"* (Código Civil). Siendo la venta la acción llevada a cabo por la parte contratante encarga de entregar una cosa o servicio a cambio de un precio determinado. Mismamente, y en concordancia con la definición del Código Civil, la Real Academia Española (RAE) define "*venta*" en su primera acepción como "*acción y efecto de vender*", y en su tercera como "*contrato en virtud del cual se transfiere a dominio ajeno una cosa propia por el precio pactado*" (RAE, 2021).

Sobre esta definición jurídica y etimológica del propio concepto de venta se puede ahora construir el concepto de venta cruzada. Este término es definido académicamente por Jones et al. en "*Selling ASAP: Art, Science, Agility, Performance*" (Jones, Chonko, Jones, & Stevens, 2012) como la oferta a los clientes de productos o servicios adicionales que pueden aportarles un valor añadido.

Otros autores han definido la venta cruzada con otros nombres como *venta sugestiva* y *venta complementaria* (Polonsky, y otros, 2000). Polonsky, y otros, definían así la venta sugestiva como el método utilizado por los promotores de ventas para aumentar el volumen de ventas de la empresa y retener a los clientes o adquirir un nuevo cliente. Todo ello basándose en la comprensión del comportamiento del cliente y en cómo satisfacerlo para convertirlo de un cliente temporal a un cliente permanente y fidelizarlo.

Sin embargo, no todos los autores presentan la misma visión, y otros como Shepard (1999) y Kamakura et al. (2003) definen la venta cruzada no tanto como una estrategia para atraer clientes, sino más como una estrategia para maximizar el número de productos o servicios que un cliente utiliza en una empresa.

En base a todo lo comentando sobre venta cruzada, se podría definir de forma sencilla como una técnica de venta usada para maximizar el beneficio obtenido por cada consumidor vendiéndole un producto adicional relacionado con un producto previamente comprado con el fin de maximizar la utilidad del cliente y de la empresa.

Una de las empresas más grandes del mundo, que es de los mayores exponentes a la hora de aplicar la venta cruzada es Amazon.com, Inc. la cual atribuye un 35% de sus beneficios a la venta cruzada (Forbes, 2015). Las dos principales ideas básicas que subyacen en la filosofía de Amazon.com, Inc a la hora de realizar la venta cruzada son:

- Qué productos son comprados de forma conjunta con frecuencia.
- Los clientes que compraron el artículo "*A*" también compraron "*B*".

De esa forma Amazon.com, Inc publicita, en tiempo real, en su web productos relacionados con el artículo que está viendo el cliente. Pero para que Amazon.com, Inc pueda llevar a cabo toda esta estrategia de venta necesita de una gran cantidad de datos y algoritmos de aprendizaje automático.

Otros ejemplos de grandes empresas que llevan a cabo esta práctica son McDonald's cuando ofrece patatas con las hamburguesas o Expedia que una vez que el cliente compra el billete de avión le ofrece habitaciones de hotel y vehículos de alquiler en el destino del billete.

Ventajas

Cuando una campaña de venta cruzada es eficaz, es decir cuando los costes de la campaña son menores a los beneficios reportados por la misma, supone un aumento de las ventas, mayores niveles de satisfacción y fidelidad de los clientes, así como un mayor nivel de gasto global por cliente (Levine, 1996). Además, la venta cruzada reduce simultáneamente los costes de adquisición de clientes ya que la comercialización a los

clientes existentes suele ser más barata que la adquisición de nuevos clientes acorde con el estudio realizado por Kamakura et al (2003).

Al mismo tiempo, aumenta el coste de abandono del cliente a otra nueva empresa. Este efecto fue definido por Sippel y Gouthro (Cross-Selling: The Unfulfilled Promise, 1987) como el coste de oportunidad de comprar productos en una nueva compañía y dejar de comprar los productos en la compañía actual. Este aumento de los costes de abandono, unido a una mejor satisfacción de los clientes, conduce en última instancia a una mayor retención de clientes y unos ingresos promedio mayores y más estables. Las ventajas se podrían listar de la siguiente manera:

- Incremento de ventas.
- Aumento de estabilidad e ingreso promedio por cliente. Es decir, mayor rentabilidad por cliente.
- Reducción de costes logísticos y de adquisición.
- Aumento de fidelidad y satisfacción del cliente, y por ende de la retención de los mismos.
- Adquirir posibles nuevos clientes.
- Identificación de clientes a través de la segmentación.
- Proporciona información al cliente sobre otros productos o servicios que le puedan interesar.

Inconvenientes

Sin embargo, el éxito de las campañas de venta cruzada depende de muchas variables como el número de productos utilizados por cada cliente, la solidez de la relación existente entre el proveedor y el cliente, la competitividad de la oferta de productos del proveedor en comparación con la oferta de la competencia y la capacidad de generar buenas ofertas de venta cruzada a los clientes adecuados (Neslin, Knott, Aaron, & Hayes, 2002). El principal problema de las ventas es que el resultado final depende única y exclusivamente del potencial cliente que es quien decide si compra o no. Claramente existen técnicas para persuadir al cliente e incitarle a la compra, pero gran parte del esfuerzo depende de la motivación y conocimientos del vendedor (Andy Neely, 2002).

Y, de hecho, la venta cruzada no es siempre una buena idea. Según un estudio de Harvard Business Review publicado por Denish Shah y V. Kumar llamado "*The Dark Side of Cross-Selling*" (2012), algunos clientes pueden hacer que una campaña de venta cruzada pase de ser exitosa a arrojar pérdidas. Según los autores, los perfiles de clientes capaces de conseguir esto serian:

- Demandantes de servicios: son aquellos clientes que cuantos más productos compran más uso hacen del servicio de atención al cliente. Aumentando el coste de los servicios ofrecidos.
- Clientes que generan muchas devoluciones: los clientes de este segmento generan ingresos, pero luego los retiran mediante devoluciones. En las empresas que

venden servicios, las reversiones de ingresos suelen consistir en el impago o la rescisión anticipada de préstamos o contratos.

- Maximizador de promociones: este tipo de cliente solo compra artículos con grandes descuentos y evita los productos de precio regular.
- Clientes con gasto limitado: los clientes de este segmento gastan sólo una pequeña cantidad fija en una empresa determinada, ya sea por limitaciones financieras o porque reparten sus compras entre varias empresas.

Es por ello que para garantizar el mayor éxito posible de la campaña es necesario analizar las métricas y los datos del cliente objetivo con el fin de asegurar que los esfuerzos de la compañía son útiles para aumentar la rentabilidad global, así como determinar qué clientes deben quedar fuera de la campaña o qué clientes deben abordarse con diferentes métodos, tales como el "*up-selling*" (ofrecer una mejora o versión ampliada del producto que oferta) por ejemplo. En general, realizar una venta cruzada con demasiadas opciones a un número demasiado amplio de clientes puede ser contraproducente si no cuenta con una estrategia bien planificada.

Pasos campaña de venta cruzada

De esta forma, los pasos correctos para implementar exitosamente una campaña de venta cruzada de acuerdo con la compañía Salesforce (2022) serían los siguientes:

1. Determinar los productos y servicios relaciones que se quieren vender mediante la venta cruzada. Para este paso, se lleva a cabo una metodología como la descrita previamente para Amazon.com, Inc.
2. Identificar y analizar los datos de los clientes a los que va dirigida la campaña. En este apartado es muy importante la implementación de técnicas de análisis de datos, big data, aprendizaje automático e Inteligencia Artificial que permitan a la empresa detectar patrones de consumo (por ejemplo, la conversión de ventas a través de un determinado canal de distribución) o segmentación de clientes para optimizar los esfuerzos a la hora de llevar a cabo un proceso de venta y obtener una ratio de conversión mayor.
3. Desarrollar una estrategia para establecer contacto con el cliente y llevar a cabo la venta. En este punto es donde el equipo de marketing debe decidir como acercarse al cliente para que éste se sienta interesado en el producto, desarrollar un argumentario, los anuncios que se vaya a realizar, dónde se van a emitir (internet, televisión, radio), etc.
4. Análisis y evaluación de los resultados. Una vez terminada la campaña comprobar las métricas clave pare poder determinar si la campaña fue exitosa o no y ver en qué se puede mejorar para futuras ocasiones.

El desarrollo posterior de este trabajo se centrará especialmente en el segundo paso de la de la campaña de venta cruzada implementando análisis de datos y procesos de aprendizaje automático para poder determinar potenciales clientes.

2.2 Técnicas de Venta Cruzada

Bundling o Empaquetado

Algunas técnicas de venta cruzada son tan simples como ofertas o descuentos en una segunda unidad, o una simple recomendación del vendedor ofreciendo un complemento al producto original comprado. Pero en especial destaca la conocida como "*bundling*" o empaquetado. El bundling se refiere a la práctica de comercializar dos o más productos, o servicios, en un único paquete a un precio especial (Guiltinam, 1987), en otras palabras, crear packs de productos. Varios ejemplos de bundling de servicios se encuentran comúnmente en las agencias de viajes, como Expedia, que ofrecen paquetes de viajes aéreos más alojamiento. La razón original del bundling es de naturaleza económica, ya que el coste marginal de vender productos adicionales es relativamente bajo en comparación con los costes totales. Además, la mayoría de los servicios que ofrecen las empresas son normalmente interdependientes por naturaleza (Guiltinam, 1987).

Es una técnica muy sencilla ya que no cuesta mucho más esfuerzo ofrecer productos adicionales que son de naturaleza complementaria y que posiblemente el cliente necesite tarde o temprano. Por último, a pesar de que los beneficios son evidentes tanto para el vendedor (por ejemplo, aumento de los ingresos, retención de clientes, etc.) como para el cliente (satisfacción, compra única, etc.), el bundling es una técnica de venta cruzada bastante barata en lo que respecta a coste logísticos. Sin embargo, va dirigida a un público muy amplio, no obteniendo siempre una gran ratio de conversión de ventas. Por eso, a pesar de ser barato no es una gran estrategia de venta cruzada. Por eso en este trabajo se explorarán otras técnicas que predicen los futuros potenciales clientes para mejorar esa ratio de conversión de ventas.

Psicografía interactiva

Peltier et al. (2002) sugieren en su publicación "*Interactive Psychographics: Cross-Selling in the Banking Industry*" una estrategia de venta cruzada que acuñaron como "*psicografía interactiva*", según la cual la empresa identifica primero los clientes basándose en la psicografía de los clientes, es decir, valores, motivos, actitudes, creencias y estilos de vida de éstos, perfila dichos clientes, empareja a los individuos con el segmento apropiado y, por último, desarrolla estrategias y productos específicos para adaptarse mejor a las necesidades psicológicas y de compra de cada cliente. Entre sus sugerencias, basadas en los resultados empíricos, está la de dirigirse a los clientes con estudios superiores, a los varones y a los clientes con mayor nivel de ingresos para realizar esfuerzos de venta cruzada. Su razonamiento para estas sugerencias es que estos clientes son más probables que compren productos adicionales en un periodo de tiempo más corto.

Clústeres de productos

Este método de venta cruzado es muy básico, y consiste en distribuir estratégicamente el espacio físico de las tiendas para colocar cierto tipo de productos que suelen ser complementarios. Port ejemplo, en una tienda de ropa vender en una zona zapatillas y cerca posicionar los calcetines, ya que, si un cliente necesita comprar zapatillas, es más probable que también quiera nuevos calcetines (Parnell & Anderson, 2022).

Universos temáticos

Este modo de venta cruzada siempre suele ir dirigido a un público concreto al que le gusta una temática en particular, y por lo tanto el cliente objetivo es más probable que realice una venta cruzada no por el producto en sí, sino por la temática del producto, que puede ir desde una película a un equipo de futbol. La idea principal es vender una variedad de productos pero que tengan alguna característica o cualidad en común. Un ejemplo muy común tiene lugar con productos promocionales de eventos, películas, series o de un determinado equipo de futbol donde el cliente objetivo es un admirador/fan de esa temática y comprará el producto no por las características intrínsecas del producto sino porque pertenece a dicha temática concreta que le interesa al cliente.

Multi-espacios

Esta técnica de venta cruzada es muy sencilla de implementar y se basa en la diversificación de servicios o productos en establecimientos donde originalmente sólo se vende un producto. Un claro ejemplo de este tipo de venta cruzada es llevado a cabo por grandes almacenes, como "El Corte Inglés", donde en una misma planta se pueden encontrar productos similares y complementarios. Amazon.com, Inc lleva a cabo también este tipo de venta cruzada, pero a través de un formato web, donde en una misma página web puedes encontrar infinidad de productos similares.

2.3 Venta cruzada en el mundo del Seguro y servicios financieros

A pesar del uso generalizado de la venta cruzada en la práctica, existen pocas investigaciones empíricas sobre la materia en la literatura académica. Sin embargo, la mayoría de las que existen son sobre la aplicación de la venta cruzada en el ámbito de los servicios financieros y seguros.

Algunos de los estudios existentes sobre la venta cruzada se enfocan desde una perspectiva más analítica del marketing como el caso de "C*ross-Selling in the Financial Sector: Customer Profitability is Key*" de Jarrar y Neely (2002) donde se relata cómo un software de datos y gestión de clientes puede utilizarse potencialmente para optimizar las oportunidades de venta cruzada y asignar mejor los recursos de la empresa. El estudio dio como resultado un éxito en las ventas cruzadas, pero los autores concluyeron que los resultados se debían a la capacidad de los empleados para comprender las necesidades de los clientes, y no al sistema informático que debía apoyar la iniciativa. Sin embargo, otros estudios posteriores proponen modelos que estima la propensión de los clientes a comprar en una categoría de productos y el momento en que lo hacen, basándose en los momentos de compra anteriores (Neslin, Knott, Aaron, & Hayes, 2002).

Otro ejemplo de una publicación sobre la venta cruzada en los servicios financieros es el estudio sobre la retención de clientes y la venta cruzada en el sector de los seguros de Harrison y Ansell (2002) que constato que los clientes casados, con mayor poder adquisitivo, de mayor edad y de sexo femenino eran más propensos a comprar un segundo producto.

En lo que respecta a la venta cruzada a través de los diferentes canales de distribución de productos de seguros, Lymberopoulos et al. (2004) concluyeron que la mayor oportunidad proviene de vender productos de seguros empleando a los bancos como proveedores de los mismo. Esta conclusión sugiere que los bancos están bien posicionados para capitalizar las oportunidades de venta cruzada en el mundo de los seguros. Bajo esta premisa, el resulto fue que el banco ayudó a generar beneficios mediante la venta cruzada, citando también que el mejor predictor del éxito de la venta cruzada es la propiedad actual del producto.

Normalmente, los agentes de seguros independientes, además de su oferta estándar de seguros de no-vida (automóvil, incendio, hogar, etc.) y vida, ofrecen también productos de seguro no clásicos como *unit-linked*, reaseguro financiero o fondos de inversión, y los clientes, a menudo, desconocen todas las opciones disponibles. Aun así, Ernest Mack (1997), llevo a cabo un estudio sobre los efectos de la venta cruzada en el mundo del seguro obteniendo como resultados unos datos interesantes sobre el éxito de la venta cruzada:

1) La retención de clientes que han formado parte de un proceso de venta cruzada es un 61% mayor que la de las que no han realizado una compra cruzada.
2) Como los clientes de venta cruzada tienen más coberturas y servicios, estas cuentas producen un 31% más de primas y comisiones.
3) Los agentes y corredores tienen una probabilidad del 66% de vender a clientes existentes, frente a un 20% de vender con éxito a potenciales clientes.

Basándose en estas estadísticas, la venta cruzada entre los agentes de seguros es un esfuerzo que merece la pena. Las aseguradoras ven en la venta cruzada una forma de captar la fidelidad de sus clientes satisfaciendo todas sus necesidades financieras (Mack, 1997).

Una vez explicado el concepto de venta cruzada en este capítulo, y de observar las bondades y debilidades que presenta, así como su impacto en el mundo de los servicios financieros y el seguro, se procederá en el siguiente capítulo a explicar el concepto de aprendizaje automático o "*machine learning*" y de los diferentes algoritmos para crear modelos predictivos que permiten disgregar a los clientes potenciales de una campaña de venta cruzada de los que no.

3. MACHINE LEARNING

3.1 Concepto de Machine Learning

El aprendizaje automático o "*machine learning*" (ML) es una rama de la inteligencia artificial (IA) y de la informática que se centra en el uso de datos y algoritmos para imitar la forma en que los humanos piensan y aprenden, de tal forma que a medida que se entrenan esos algoritmos mejora su precisión (IBM Cloud Education, 2020).

El creador del término de aprendizaje automático fue asignado a Arthur L. Samuel, un antiguo trabajador de IBM quien público "*Some studies in machine learning using the game of checkers*" (1959) en el cual se definía el aprendizaje automático como "*la programación de los ordenadores para que aprendan de la experiencia y así acabar con la necesidad y el detalle del gran esfuerzo de la programación*". En dicha publicación Arthur planteaba un algoritmo de aprendizaje automático que aprendiese a jugar a las damas y en 1962 el campeón de las damas Robert Nealey fue vencido por dicho algoritmo.

Comparado con lo que se puede hacer hoy en día, como la construcción del sistema de inteligencia artificial IBM Watson (Ferruci, Brown, Chu-Carroll, & Fan, 2010), esta hazaña es trivial, pero se considera un hito importante dentro del campo de la inteligencia artificial. Desde entonces, y en especial en los últimos años, los desarrollos tecnológicos en torno al aprendizaje automático y la capacidad de procesamiento han permitido la creación de complejos algoritmos que son capaces de recomendar de manera acertada productos que pueden interesar al consumidor (Netflix o Amazon mencionados previamente) o incluso de conducir coches de forma autónoma (Tesla).

El aprendizaje automático es un componente importante del creciente campo de la inteligencia artificial, y gracias al uso de métodos estadísticos, los algoritmos se entrenan para hacer clasificaciones, predicciones, descubriendo patrones o relaciones existentes complejas dentro de los proyectos de big data que costarían mucho de identificar al ser humano en tiempo y esfuerzo. Estos patrones o relaciones entre los datos existentes impulsan posteriormente la toma de decisiones de las empresas lo que repercute en su desempeño.

Como funciona el Ml

La Universidad de Berkeley (Berkeley School of Information, 2020) desglosa las bases de todos los sistemas de aprendizaje automático en tres partes principales:

1. Un proceso de decisión. En general, los algoritmos de machine learning se utilizan para hacer una predicción o clasificación. Basándose en unos datos de entrada – que pueden estar etiquetados o no – y el algoritmo producirá una estimación sobre un patrón o una relación en los datos.
2. Una función de error. La función de error sirve para evaluar la predicción del modelo. Si hay ejemplos previos conocidos, la función de error puede hacer

una comparación para evaluar la precisión del modelo. Normalmente para esto se disgregan los datos en dos conjuntos, unos de entrenamiento y otro de validación.

3. Un proceso de optimización del modelo. Si el modelo puede ajustarse mejor al conjunto de datos de entrenamiento, se ajustan los pesos de cada variable para reducir la discrepancia entre el ejemplo conocido y las estimaciones del modelo. El algoritmo repetirá este proceso de evaluación y optimización, actualizando los pesos de forma autónoma hasta alcanzar un umbral de precisión deseado o posible.

Por ejemplo, volviendo al ejemplo de Netflix y el sistema de recomendación de películas. El algoritmo puede alimentarse de información sobre el consumidor y su historial de visionado como entrada. Entonces el algoritmo tomará esa información y aprenderá a dar un resultado preciso de películas que le gustarán al consumidor. Algunos datos de entrada pueden ser las películas que ha visto y valorado, el porcentaje de películas que ha visto de un determinado género cinematográfico o cuántas películas tienen un actor en particular. El trabajo del algoritmo consiste en encontrar estos parámetros y asignarles pesos. Si el algoritmo acierta, las ponderaciones que ha utilizado no varían. Por el contrario, si se equivoca, las ponderaciones que le llevaron a tomar la decisión equivocada se reducen para que no vuelva a cometer ese tipo de error.

Dado que un algoritmo de aprendizaje automático se actualiza de forma autónoma. La precisión del análisis mejora con cada ejecución, ya que se enseña a sí mismo a partir de los datos que analiza. Esta naturaleza iterativa del aprendizaje es única y valiosa porque se produce sin intervención humana, lo que permite al algoritmo descubrir ideas ocultas sin estar programado específicamente para ello.

3.2 Clases de aprendizaje automático

Acorde con el libro "*The Elements of Statistical Learning: Data Mining, Inference and Predictions*" (Hastie, Tibshirani, & Friedman, 2008), existen cuatro categorías principales de aprendizaje automático:

3.2.1 Aprendizaje automático supervisado

El aprendizaje supervisado, es el más simple, y se caracteriza por el uso de conjuntos de datos previamente etiquetados para entrenar algoritmos que clasifiquen datos o predigan resultados con precisión basando en los pares de datos de entrada y salida (Hastie, Tibshirani, & Friedman, 2008). A medida que los datos de entrada se introducen en el modelo, éste ajusta sus ponderaciones hasta que el modelo se ajusta adecuadamente. Para comprobar que el modelo está bien calibrado, y no se ajuste en exceso o en defecto, se lleva a cabo un proceso de validación cruzada mediante la separación de los datos en un conjunto de datos de entrenamiento y otro conjunto de datos de validación que el algoritmo nunca ha visto para después contrastar el resultado observado con el esperado y medir el grado de error. Ejemplos de algoritmos de aprendizaje automático supervisado pueden ser clasificar el correo spam dentro del correo electrónico, o un algoritmo que

prediga la altura de una persona en base a su edad, siempre y cuando dispongamos de un conjunto de datos de entradas con el que entrenar el algoritmo. A su vez, el aprendizaje supervisado se divide en dos subcategorías: regresión y clasificación.

Aprendizaje Automático Supervisado de regresión

En los modelos de regresión, el resultado es una variable continua. A continuación, se presentan algunos de los tipos más comunes de modelos de regresión acorde con los autores Hastie, Tibshirani y Friedman:

Regresión lineal

La idea básica de la regresión lineal es simplemente encontrar la línea que mejor se ajusta a los datos o una nube de puntos. Entre los tipos de regresiones lineales se incluyen regresiones lineales múltiples y la regresión polinómica.

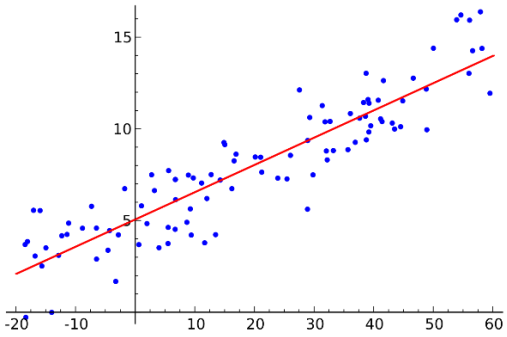

Figura 3.1. Regresión lineal. Fuente: Wikipedia, 2008.

Árboles de decisión

Los árboles de decisión (Szufel, Kamiński, & Jakubczyk, 2017) son un modelo popular, utilizado tanto en la planificación estratégica como el aprendizaje automático. Más adelante será desarrollado en mayor profundidad, ya que es la base del bosque aleatorio, pero los árboles de decisión son sencillos de construir y fáciles de interpretar, aunque a veces carecen de precisión. La idea principal es que en cada nodo se evalúa una cualidad de los datos y se produce una diferenciación. Se encuentran formados por nodos de los que salen ramas que conectan con más nodos hasta llegar a las hojas. Cuantos más nodos tenga un árbol, más preciso será su resultado generalmente. Los últimos nodos del árbol de decisión, donde se toma una decisión, se denominan hojas del árbol.

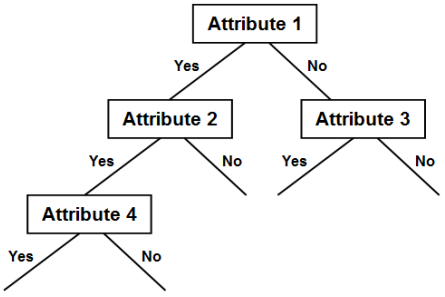

Figura 3.2. Árbol de decisión. Fuente: NVIDIA, 2017.

Bosque aleatorio

Posteriormente será desarrollado al completo, pero a modo de breve descripción, los bosques aleatorios son una técnica de aprendizaje desarrollada por Breiman (2001) que se basa en los árboles de decisión y usa como técnica de ensamble el "*bagging*", también conocido como agregación por "*bootstrap*" (muestreo con remplazo). Con esta técnica de bagging, se crean varios bosques aleatorios que son entrenados por muestras "boostrapeadas" después se realizan las predicciones de cada árbol para una observación en concreto y posteriormente se promedian esos resultados para obtener la predicción final del modelo. Adicionalmente, en cada nodo de cada árbol se observan sólo un subconjunto aleatorio de variables que hacen que los árboles estén descorrelacionados, lo que permite reducir el error.

Redes Neuronales

Una red neuronal fue primeramente descrita por Paul Werbos (Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences, 1974) y se define esencialmente como una red de ecuaciones matemáticas que toma una o más variables de entrada, y al pasar por una red de ecuaciones, da como resultado una o más variables de salida. Otra forma de decirlo es que una red neuronal toma un vector de entradas y devuelve un vector de salidas. Las redes neuronales tienen tres capas: entrada, salida y oculta. Las capas de entrada y salida son visibles y como su nombre dice son los datos de entrada y de salida del modelo. Como se ve en la imagen inferior, la capa de entrada es azul y la de salida violeta. Pero la capa oculta (en color verde) no se puede ver y está formado por nodos que contienen una función lineal o una función de activación por la que pasan los nodos de la capa anterior y que, en última instancia, conduce a los nodos de salida.

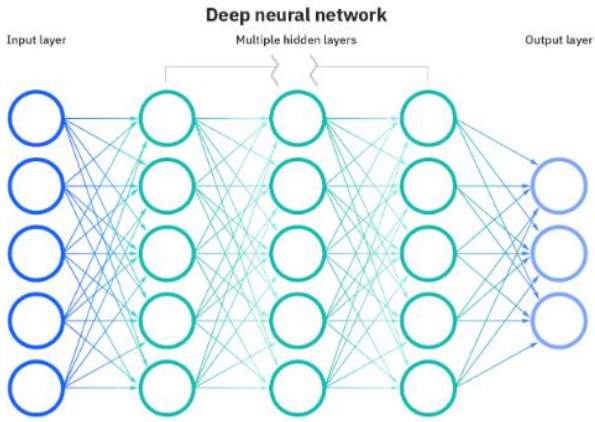

Figura 3.3. Esquema de red neuronal. Fuente: IBM, 2020.

Aprendizaje Automático Supervisado de regresión

Prosiguiendo con la clasificación realizada en "*The Elements of Statistical Learning Data Mining, Inference and Prediction*" en los modelos de clasificación, a diferencia de los de regresión, la salida es discreta. A continuación, se presentan los tipos más comunes de modelos de clasificación:

Regresión logística

La regresión logística es similar a la regresión lineal, pero se utiliza para modelar la probabilidad de un número finito de resultados, aunque normalmente se establece un umbral de probabilidades para definir cada estado deseado. Como el resultado son probabilidades, una función logística (con forma de "*S*") se crea de tal manera que los valores de salida sólo pueden estar comprendidos entre 0 y 1. Una situación muy típica en la que se aplican estas regresiones es por los bancos para decidir si otorgar o no una hipoteca.

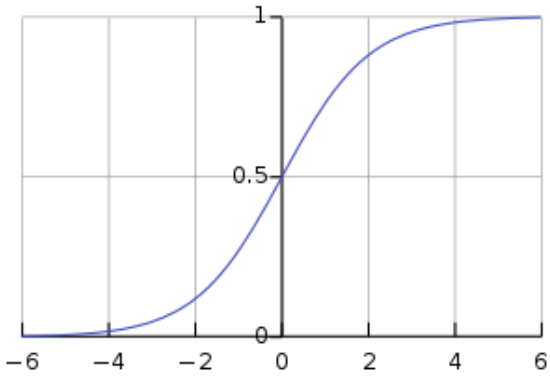

Figura 3.4. Regresión logística. Fuente: Wikipedia, 2022.

Máquina de vectores de soporte

Una máquina de vectores de soporte (SVM) es una técnica de clasificación supervisada que puede llegar a ser bastante complicada en conjunto de datos con muchas dimensiones, pero es muy intuitiva de entender. Una SVM es un clasificador discriminativo definido formalmente por un hiperplano de separación que maximice el margen, la distancia, entre las dos clases (Matlab, 2022). Hay muchos planos que pueden separar dos conjuntos de datos, pero sólo un plano puede maximizar el margen o la distancia entre las clases. De esta forma, partiendo de unos datos de entrenamiento etiquetados, el algoritmo produce un hiperplano óptimo que categoriza los nuevos ejemplos. En una gráfica de dos dimensiones, este hiperplano es una línea que divide la gráfica en dos partes separando los datos en dos clases, una a cada lado de la recta como se muestra en la imagen de abajo.

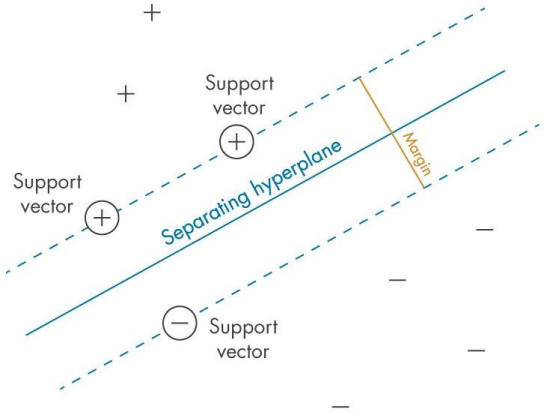

Figura 3.5. Esquema de SVM. Fuente: Matlab, 2022.

Árboles de decisión, Bosque aleatorio y redes neuronales

Estos modelos siguen la misma lógica explicada anteriormente. La única diferencia es que esa salida es discreta en lugar de continua.

3.2.2 Aprendizaje automático no supervisado

El siguiente tipo de aprendizaje que reseñan Hastie, Tibshirani y Friedman es el aprendizaje automático no supervisado que es también relativamente sencillo. Utiliza algoritmos de aprendizaje automáticos para hacer inferencias, encontrar patrones, analizar y agrupar conjuntos de datos no etiquetados en diferentes categorías. Estos algoritmos descubren patrones ocultos o agrupaciones de datos sin necesidad de intervención humana por lo que a veces es difícil medir su desempeño. Su capacidad para descubrir similitudes y diferencias en la información lo convierten en la solución ideal para el análisis exploratorio de datos, las estrategias de venta cruzada, la segmentación de clientes y el reconocimiento de imágenes y patrones. Los métodos que más destacan son:

Clustering

El clustering es una técnica no supervisada descubierta por J.MacQueen (1967) que implica la agrupación, es decir, la formación de clústeres, de puntos de datos con características en común. Se utiliza con frecuencia para la segmentación de clientes, la detección de fraudes y la clasificación de documentos. Las técnicas de clustering más comunes son el uso de k-medias, el clustering jerárquico, el clustering de desplazamiento de medias y el clustering basado en la densidad. Aunque cada técnica tiene un método diferente para encontrar clústeres, todas pretenden conseguir el mismo objetivo.

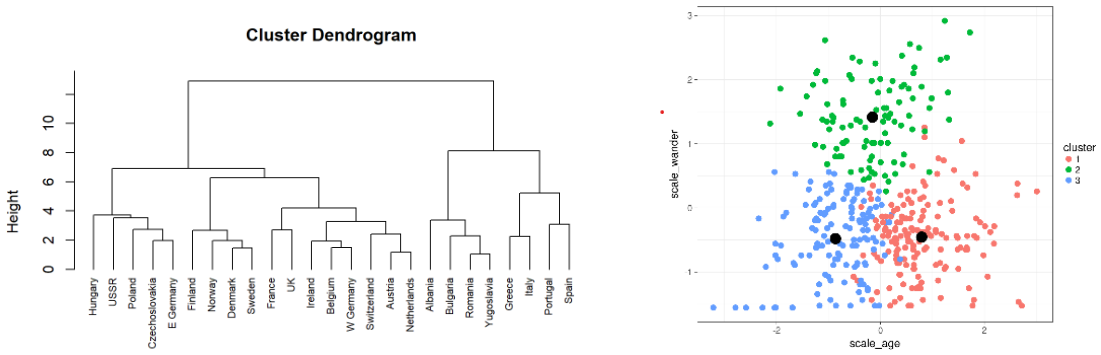

Figura 3.6. Clúster jerárquico (izquierda) y clúster de k-medias(derecha). Fuente: Rpubs, 2018.

Reducción de la dimensionalidad

Entre los métodos más populares de reducción de la dimensionalidad destacan la descomposición en valores singulares (SVD) y, en especial el análisis de componentes principales (PCA). El PCA es una técnica que permite reducir la dimensionalidad (o número de características) de los conjuntos de datos, aumentando la capacidad de interpretación y minimizando al mismo tiempo la pérdida de información. Para ello, se crean nuevas variables no correlacionadas que maximizan sucesivamente la varianza.

Encontrar esas nuevas variables, los componentes principales, se reduce a la resolución de un problema de valores y vectores propios, y las nuevas variables se definen en función del conjunto de datos. Un ejemplo sencillo de entender es proyectar datos de mayor dimensión (por ejemplo, 3 dimensiones) a un espacio más pequeño (por ejemplo, 2 dimensiones). El resultado es una dimensión de datos más baja (2 dimensiones en lugar de 3), manteniendo todas las variables originales en el modelo.

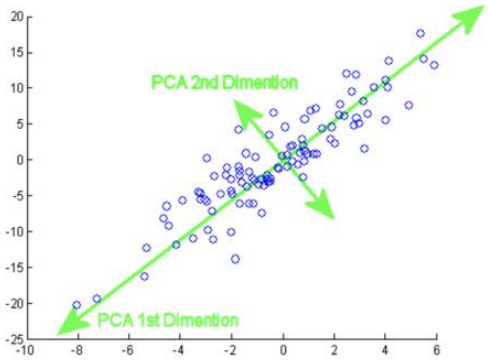

Figura 3.7. Gráfico de PCA. Fuente: Programmathically, 2022.

3.2.3 Aprendizaje automático semisupervisado

El tercer tipo de aprendizaje mencionado en "*The Elements of Statistical Learning Data Mining, Inference and Prediction*" es el aprendizaje semisupervisado que ofrece un término medio entre el aprendizaje supervisado y el no supervisado de tal forma que los datos pueden estar etiquetados o no. Durante el entrenamiento del algoritmo se emplea un conjunto de datos etiquetados más pequeño para guiar la clasificación y la extracción de características de un conjunto de datos más grande y sin etiquetar. El aprendizaje semisupervisado puede resolver el problema de no tener suficientes datos etiquetados (o no poder etiquetar suficientes datos) para entrenar un algoritmo de aprendizaje supervisado.

3.2.4 Aprendizaje automático reforzado

El último tipo de aprendizaje destaco por Friedman y otros es el aprendizaje automático por refuerzo por primera vez explicado en "*Simple statistical gradient-following algorithms for connectionist reinforcement learning*" por Ronald J. Williams (1992). Es un modelo de aprendizaje interesante, basado en la psicología conductual, donde el algoritmo es entrenado para tomar una secuencia de decisiones, como si de una cadena de Markov se tratase, con el fin de alcanzar un objetivo en un entorno incierto y potencialmente complejo. Este sistema de aprendizaje también destaca porque los datos no están etiquetados, por lo que no se conocen (a priori) los pares de estado-acción óptimos y correctos, a excepción del objetivo final. Además, como existen múltiples posibles acciones para cada estado implica que existen diferentes soluciones y el proceso varía en funciones de cada estado posible. Durante el entrenamiento, el algoritmo emplea el método de ensayo y error para encontrar la solución al problema y explora

aleatoriamente los pares estado-acción para construir una tabla de pares con todos los posibles resultados de cada par. A continuación, en la validación de la información obtenida, emplea los diferentes pares estado-acción para elegir la mejor acción para un estado determinado que conduzca a algún estado objetivo.

Por ejemplo, consideremos a un jugador de blackjack. Los estados representan la suma de las cartas del jugador y las acciones representan lo que el jugador puede hacer, en este caso, pedir o plantarse. Entrenar a un jugador de blackjack implicaría muchas manos donde el resultado de una acción – plantarse o pedir otra carta – sea pasar a un estado de ganar (la suma de las cartas es veintiuno) o perder (la suma de las cartas es mayor que veintiuno o inferior al del resto de jugadores). Por ejemplo, el valor para probabilístico de un estado donde la suma de las cartas es 10 sería 1 para pedir otra carta y 0 para plantarse (lo que indica que pedir otra carta es la opción óptima). Para el estado 20, la recompensa aprendida probablemente sea 0 para pedir otra carta y 1 para plantarse. Para una mano alta pero no tanto, como un estado 18 los valores de pedir pueden ser de 0,1 y de plantarse de 0,9 para ganar el juego. Este jugador debería plantarse el 90% de las veces que este en un estado 18 y pedir otra carta el 10% de las ocasiones para ganar. Estos resultados se estudian a lo largo de muchas manos para poder indicar la mejor elección posible (pedir o plantarse) para un estado (o mano) determinado.

Adicionalmente, este sistema de aprendizaje también posee un componente colaborativo en línea por el que cualquier individuo que use el algoritmo puede entrenarlo permitiendo un equilibrio entre la explotación del algoritmo, es decir el uso del algoritmo en su estado actual, y la exploración de este, aumentar la capacidad del mismo. Un ejemplo muy popular y real que aplica este tipo de aprendizaje es la inteligencia artificial GPT-3 (OpenAI, 2021) que permite generar texto, incluido código de programación, de la forma más natural y humana posible.

4. BOSQUE ALEATORIO y XGBOOST

4.1 Bosque aleatorio

Tal y como se comentaba en el primer capítulo de esta publicación, el algoritmo de Bosque aleatorio o "*Random forest*" fue seleccionado para ser contrastado con el XGBoost porque ambos se construyen sobre la base de los árboles de decisión, y porque en "*Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?"* de Fernández-Delgado et al. (2014) se evaluaron 179 clasificadores en 121 conjuntos de datos y se descubrió que los bosques aleatorios alcanzaban la mejor precisión global. Estas conclusiones se ven reforzadas por los descubrimientos de otros investigadores como Caruana et al. (2008) también llegaron a la misma conclusión cuando evaluaron 10 familias de clasificadores en 11 problemas de clasificación binaria con alta dimensionalidad. Los bosques aleatorios obtuvieron el mayor rendimiento de media en todos los conjuntos de datos considerados, superando a las SVM, las redes neuronales, la regresión logística, etc. Los bosques aleatorios son rápidos y funcionan bien en la práctica, lo que lo convierte en un gran método de predicción. Por ello a continuación se explicarán en profundidad los conceptos fundamentales del bosque aleatorio.

El bosque aleatorio es un algoritmo de aprendizaje automático supervisado de ensamble comúnmente utilizado, que fue creado por Leo Breiman y Adele Cutler en 2001 basándose en el trabajo previo de Tim Kam Ho (1995), que combina la salida de múltiples árboles de decisión para llegar a un único resultado. Leo y Adele registraron bajo la marca comercial "*Random Forest"* en 2006 el algoritmo, y en 2019 la empresa Minitab, Inc adquirió los derechos de propiedad. Su facilidad de uso y su flexibilidad han impulsado su adopción, ya que maneja tanto problemas de clasificación como de regresión, aunque en este documento sólo trataremos la parte de árboles de clasificación. De hecho, actualmente el bosque aleatorio se emplea en campos muy diversos como las finanzas, para detectar fraudes en cuentas bancarias o evaluar clientes con altos niveles de riesgo, en la industria farmacéutica con el fin de desarrollar terapias genéticas, descubrir biomarcadores etc. (Qi, 2012) y obviamente también en los comercios para predecir número de ventas o venta cruzada como es el caso de esta publicación.

El algoritmo del bosque aleatorio es una agrupación de árboles de decisión para generar estimadores con los que predecir – de ahí el nombre, ya que varios árboles forman un bosque – y usa el "*bagging*" como método de ensamblado de esos estimadores. Previamente a la explicación del algoritmo se deben explicar estos elementos.

Árboles de decisión

Como se ha mencionado anteriormente entre las diferentes técnicas de aprendizaje automático supervisado, los árboles de decisión tratan de encontrar la mejor división para subdividir los datos y pueden utilizarse tanto para problemas de clasificación como de regresión. La principal diferencia entre ambos usos depende del tipo de respuesta deseada. Un árbol de clasificación genera una respuesta cualitativa que trata de predecir a qué

categoría pertenece cada observación asignándola a la categoría con la cual tiene más propiedades en común en función de las observaciones de entrenamiento. Los árboles de clasificación se componen de tres partes:

- Raíz: Es el punto de origen o nodo principal del que nace el árbol de decisión. En este primer nodo se evalúa si la observación cumple o no una característica deseada del conjunto de datos, y en función de la respuesta este nodo principal se separar a su vez en dos subnodos o ramas.
- Nodos o ramas: Son los puntos intermedios entre la raíz y las hojas (nodo final). Cada nodo evalúa una característica o variable diferente del conjunto de datos y en función de la respuesta se vuelve a dividir en dos ramas, hasta llegar a un nodo final u hoja del árbol de decisión.
- Hojas: Las hojas son nodos finales de los que no sale ningún otro nodo y donde se produce la evaluación de la variable que se quiere predecir. La calificación depende del camino realizado por la observación.

Los árboles de decisión siempre crecen mediante una división binaria recursiva en cada nodo para las diferentes características del conjunto de datos como se observa en la figura 3.2 más arriba. Es decir, en cada nodo se evalúa si la observación en cuestión cumple o no una determinada característica del conjunto de datos hasta llegar a una división de los datos deseada para poder predecir que observaciones son las de interés.

Al realizar las divisiones de forma binarias, los árboles de clasificación presentan una tasa de error en cada hoja que es igual a la fracción de las observaciones en el nodo que no pertenecen a la clase más común. En caso de que una hoja sólo tenga predicciones correctas se considera una hoja pura. Para calcular el error de clasificación del árbol se lleva a cabo la suma ponderada de los errores de clasificación de cada hoja en función del número de observaciones de las mismas, sin embargo, a veces el error de clasificación no es lo suficientemente sensible y por ello se emplean otras métricas más precisas como la impureza de Gini. La impureza de Gini mide la pureza de los nodos previamente comentada (Karabiber, 2022). Por ejemplo, si la impureza de Gini (G) es pequeña, significa que una hoja contiene predominantemente observaciones de una sola clase, y por lo tanto se considerará un buen árbol de decisión porque permite disgregar correctamente el conjunto de datos. Otras métricas menos usadas para medir la pureza de los nodos es la entropía cruzada (Brownlee, 2019) pero al fin y al cabo todas tratan de medir la calidad en la separación de los datos, que es lo que marca la importancia de los nodos normalmente.

Los árboles de decisión estándar tienen la desventaja de que son propensos a sobreajustarse al conjunto de datos entrenamiento y no clasifican bien los datos no vistos. Pero gracias al diseño del bosque aleatorio se puede compensar esta falla, hacer que los árboles de decisión sean más robustos y predigan resultados de forma más precisa, especialmente cuando los árboles individuales no están correlacionados entre sí.

Bagging

El problema de los árboles de decisión, como se menciona en el párrafo anterior es que sufren una gran varianza por culpa de sobreajuste a los datos de entrada del modelo, lo que significa que, si dividimos los datos de entrenamiento en dos partes al azar y ajustamos un árbol de decisión a ambas mitades, los resultados que se obtendrán serán muy dispares. Por ello, con el objetico de reducir esta varianza se emplea un método de agregación o ensamble, cuya función es la de agregar un conjunto de estimadores, como son múltiples árboles de decisión, para identificar el resultado promedio o el más popular/mayoritario. El random forest emplea el método de ensamble conocido como "*agregación por bootstrap*" o también llamado "*bagging*" (Breiman, 2001).

El "*bagging*" selecciona múltiples muestras aleatorias con reemplazo de el set de datos de entrenamiento, lo que significa que las observaciones de datos individuales pueden ser elegidos más de una vez. Una vez generadas varias muestras de datos, se crea un árbol de decisión para cada muestra, y estos árboles son entrenados de forma independiente en paralelo para posteriormente predecir una observación en cada árbol y, dependiendo del tipo de tarea – regresión o clasificación –, el promedio o la mayoría de esas predicciones arrojan una estimación más precisa. De esta forma es como se reduce la varianza en un conjunto de datos ruidosos.

Los árboles de decisión clasificatorios se basan en un umbral determinado para clasificar que por defecto suele ser 0,5. Es decir, un umbral de decisión de 0,5 significa que el modelo clasifica una observación en la clase más comúnmente predicha entre todos los predictores, aquella que sea predicha más de un 50% de las veces, conocido como votación por mayoría. Sin embargo, este umbral puede modificarse para ajustar la clasificación, de forma que, si se baja o sube se pierde o gana precisión, pero se modifica también el ajuste por defecto o exceso al conjunto de datos respectivamente.

Con las observaciones que no han sido seleccionadas por el bootstrap, que según diferentes estudios se demuestra que son un tercio de las observaciones originales (Ahlin & Ranby, 2019) se conocen como observaciones fuera de bolsa ("*Out of the Bag*" u OOB por sus siglas en ingles). Con el OOB se puede predecir la respuesta de la *i-esima* observación mirando cada uno de los árboles en los que la observación se encuentra en el OOB, es decir no ha sido seleccionada por el bootstrap, y se emplea para validar las predicciones. La idea del OOB es que bajo la premisa de que una tercerea parte de las observaciones son no seleccionadas por el bootstrap, se produce un porcentaje de acierto en la predicción de alrededor de una tercera parte del número de árboles de decisión creados para la observación *i-esima*. Después se puede promediar la predicción mediante un sistema de mayoría que da lugar a una única predicción OOB para la observación *i-esima*. De este modo se puede obtener una predicción OOB para cada una de las *n* observaciones, a partir de la cual se puede calcular el error de clasificación. El error OOB resultante es una estimación válida del error del modelo, ya que la respuesta para cada observación se predice empleando sólo los árboles que no se utilizaron para esa observación (Ahlin & Ranby, 2019).

Algoritmo del Bosque aleatorio

Como se ha mencionado, el bosque aleatorio se basa en los árboles de decisión y en el bagging. La diferencia entre el bosque aleatorio y los árboles de decisión es que el bosque aleatorio son un conjunto de árboles de decisión que están descorrelacionados, lo cual estabiliza los resultados de la predicción. Esta descorrelación ocurre porque cuando el algoritmo de bosque aleatorio genera los árboles de decisión, en cada nodo del árbol sólo se observan un número limitado de variables. Es decir, en cada nodo sólo se tienen en cuenta un número de variables seleccionadas aleatoriamente que es menor que todas las variables del conjunto de datos. Esta parte del algoritmo es conocida como *método subespacial aleatorio* (Breiman, 2001) y es una diferencia fundamental entre los árboles de decisión y el algoritmo del bosque aleatorio, ya que mientras los árboles de decisión observan todas las variables o predictores en cada nodo, el random forest sólo observa unas pocas variables predictoras elegidas al azar. Por defecto y en general, el número seleccionado de variables predictoras observadas en cada nodo es la raíz cuadrada del número máximo de variables que contiene el conjunto de datos original. Así el algoritmo al no poder utilizar todas las variables descorrelaciona los árboles de decisión, y al tener en cuenta toda la variabilidad potencial de los datos, tanto en las variables observadas, como en las observaciones, se puede reducir el riesgo de sobreajuste, el sesgo y la varianza general, lo que da lugar a predicciones más precisas. A continuación, se mostrará de forma ordenada y esquemática los pasos del algoritmo del bosque aleatorio:

1. Bagging. Sobre el set de datos de original se realiza el proceso previamente explicado de bagging por el cual se crean *N* conjuntos de datos bootstrapeados.

2. Método subespacial aleatorio. Por cada subconjunto de datos bootstrapeado se crea un árbol de decisión en el que en cada nodo se observa sólo un número limitado de variables seleccionadas al azar. Así se tiene en cuenta toda la variabilidad potencial de los datos, y da lugar a predicciones más precisas y robustas.

3. Entrenamiento de los estimadores. Cada árbol es entrenado con la muestra bootstrapeada. Cada árbol intenta predecir la misma variable, pero el resultado y la capacidad de cada árbol será dispar al tener tanto observaciones como variables predictoras completamente diferentes.

4. Realizar la inferencia agregando las predicciones de los estimadores. Para cada observación del conjunto de datos original se estima el resultado en cada árbol de decisión creado obteniendo así para una observación tantas predicciones como árboles haya. Sin embargo, la predicción final del bosque aleatorio para esa observación será el resultado de la votación por mayorías (si el umbral es 0,5 o superior) de todos los árboles de decisión, o el promedio de los resultados en caso de regresión.

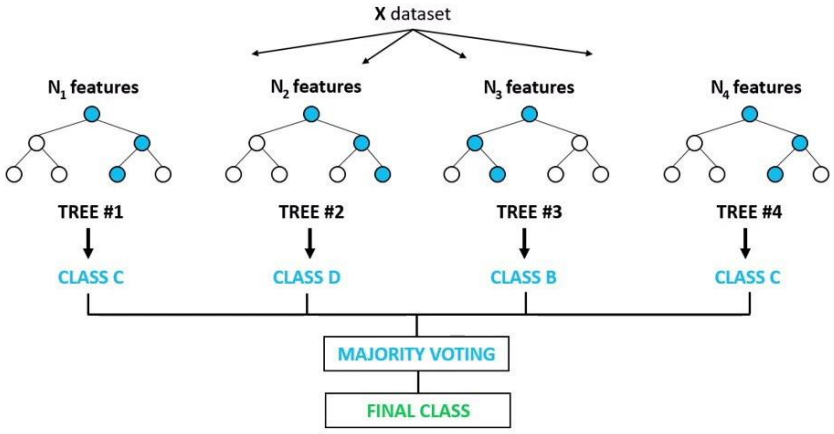

Figura 4.1. Esquema de Random Forest. Fuente: Grupo Quality, 2021.

Este uso de muchos estimadores es la razón por la que el algoritmo del bosque aleatorio se denomina método de ensamble. Ya que cada estimador/árbol individual es un estimador débil, pero cuando se combinan muchos estimadores débiles pueden producir un estimador más robusto y preciso.

Parámetro del RF

Ahora que ya se entiende el funcionamiento del bosque aleatorio, también hay que mencionar los parámetros que se pueden modificar dentro de este modelo para mejorar el rendimiento del mismo, así como su poder predictivo. Los parámetros o factores son:

- Número de estimadores. Este parámetro afecta al número de árboles de decisión que se crean. A mayor número de árboles más poder de predicción – dentro de un límite –, pero también afecta a la velocidad del algoritmo haciéndolo más lento.
- Máximo número de variables observadas por nodo. Este parámetro modifica el método subespacial aleatorio cambiando el número máximo de variables a observar en cada nodo. Al igual que el parámetro anterior, a mayor número de variables más poder predictivo pero el algoritmo es más lento.
- Profundidad de los árboles. Número máximo de nodos en cada árbol. A mayor número de nodos los árboles que se crean son más precisos, pero también el algoritmo es más lento.
- Número de procesadores en uso. Este parámetro afecta al hardware donde se esté ejecutando el algoritmo. A mayor cantidad de procesadores más actividades en paralelo podrá realizar el ordenador y más rápido irá.
- Aleatoriedad de la muestra. Con este parámetro se modifica la aleatoriedad de la muestra. Por ejemplo, si se fija una semilla en la simulación y los parámetros son se mantienen siempre se obtendrá el mismo resultado, pero sino el resultado puede variar de una ejecución a otra.

Beneficios y desventajas del Bosque aleatorio

Tal y como hemos visto el bosque aleatorio es un algoritmo ampliamente empleado en múltiples disciplinas y que es robusto gracias la técnica del bagging que permite emplear varios estimadores débiles para crear uno fuerte. Pero no son todo ventajas, por eso a continuación se muestra una tabla con las ventajas y desventajas del bosque aleatorio.

VENTAJAS	***DESVENTAJAS***
Es flexible ya que puede emplearse para clasificación y regresión.	El bosque aleatorio es un modelo complejo de interpretar si se compara con los árboles de decisión, en los que las decisiones pueden tomarse siguiendo la trayectoria del árbol.
La predicción es estable, reduce la varianza, el sesgo y el sobreajuste de los árboles de decisión al aleatorizarlos y descorrelacionar los mismos.	El tiempo de entrenamiento es mayor en comparación con otros modelos debido a su complejidad. Cada vez que tiene que hacer una predicción, cada árbol de decisión tiene que generar una salida.
Es inmune a los problemas de dimensionalidad al no tener en cuenta todas las variables en cada nodo del árbol.	Debido a su complejidad requiere de más tiempo de procesamiento y de un hardware potente, así como de más espacio para almacenar los elementos.
Permite observar con facilidad la importancia de cada variable gracias a métricas como la impureza de Gini o la entropía cruzada.	
Cuando se ejecuta también se crea el OOB que sirve para realizar la validación del modelo.	

Tabla 4.1. Ventajas y desventajas del Bosque aleatorio. Fuente: Elaboración propia, 2022.

En definitiva, se puede decir que el bosque aleatorio es un modelo relativamente complejo, pero robusto y preciso que soluciona una gran cantidad de inconvenientes de los árboles de decisión simples. Lo cual lo convierte, como decía Fernández-Delgado et al. (2014) y Caruana et al. (2008), en un clasificador con un gran rendimiento y de gran utilidad para predecir.

4.2 XGBoost

Frente al bosque aleatorio, el otro algoritmo, también de aprendizaje automático supervisado, que se ha decido contrastar es el algoritmo de Potenciación de Gradiente Extremo, o mejor conocido como "*XGBoost*" – que es más breve y sencillo –, el cual ha adquirido recientemente mucha popularidad por los buenos resultados obtenidos en las competiciones de la plataforma Kaggle (The Machine Learners, 2021). El XGBoost es un algoritmo de código abierto relativamente novedoso que fue descubierto por Tianqi Chen y Carlos Guestrin en 2016 en la publicación "*XGBoost: A Scalable Tree Boosting System*" (Chen & Guestrin, 2016). Principalmente se emplea para problemas de ordenación, clasificación y regresión, al igual que el bosque aleatorio, y además se basa en principios similares como los árboles de decisión y un método de ensamble para a partir de estimadores débiles crear un estimador fuerte.

El XGBoost es fácil de obtener, ya que se encuentra implementado en diferentes bibliotecas de software de código abierto disponible en múltiples plataformas como Python, R o C++ entre otras (XGBoost Developers, 2021), e implementa la optimización de los algoritmos de potenciación de gradiente (*GBoost*) aplicado a los árboles de decisión mediante la paralelización del "*boosting*", que es el modelo de ensamble que emplea este algoritmo.

Para entender el funcionamiento del XGBoost es fundamental comprender primero los conceptos en los que se basa, algunos previamente explicados, como son el aprendizaje automático supervisado y los árboles de decisión, y otros como el boosting como método de ensamble y la potenciación de gradiente. Por eso a continuación se procederá a explicar estos conceptos.

Boosting

El Boosting es un método de ensamble, alternativo al bagging, que como todo método de ensamble su función es la de tomar varios estimadores débiles y unificarlos para obtener un estimador fuerte. Al igual que el bagging, el boosting se emplea principalmente para reducir el sesgo y la varianza en una técnica de aprendizaje automático supervisado.

El funcionamiento del boosting es contrario al del bagging. Mientras en el bagging el entrenamiento de los estimadores débiles era en paralelo y simultaneo, y posteriormente se tomaba el promedio o la mayoría como estimador fuerte, en el boosting el entramiento es secuencial. Es decir, en el boosting se parte de un conjunto de datos original del cual se toman una muestra aleatoria y se entrena el primer estimador débil que suele ser un árbol de decisión. Una vez entrenado el primer estimador se realiza la predicción de todo el conjunto de datos original con ese estimador débil y se observa que predicciones son correctas y cuáles no, en otras palabras, se observa el error de predicción o función de perdida. Todas aquellas predicciones incorrectas son tomadas y se emplean para entrenar un segundo estimador débil con el que se vuelven a realizar predicciones del conjunto original y se vuelven a tomar las predicciones incorrectas para entre un tercer estimador. Así sucesivamente hasta entrenar el número deseado de estimadores débiles para después combinarlos y obtener un estimador fuerte. Es decir, la idea del boosting es la potenciar el mismo modelo con cada iteración.

GBoost

En el apartado superior se ha explicado el boosting, por eso en este apartado se explicará el siguiente paso que es la potenciación de gradiente conocida como GBoost propuesto por Firedman (2019). La idea del GBoost nace del boosting de mejorar un único modelo débil combinándolo con otros modelos débiles para generar un estimador. Sin embargo, el GBoost plantea un proceso de generación aditiva de estimadores débiles con el objetivo de minimizar en cada estimador una función objetivo del modelo siempre menor que el anterior. Es decir, la intención es hacer que la pendiente/gradiente – de ahí el nombre – de una función objetivo del modelo sea descendente. Esa función objetivo suele ser el error de predicción llamado función de pérdida. De esta manera el GBoost establece los resultados objetivo para el siguiente modelo en un esfuerzo por minimizar los errores. Los resultados deseados para cada estimador débil se basan en el gradiente del error con respecto a la predicción. Es decir, el GBoost se basa en los siguientes tres componentes:

1. Función de pérdida. El papel de la función de pérdida es estimar lo bueno que es el modelo para hacer predicciones con los datos dados.
2. Estimador débil. Son modelos de predicción simples que tienen bajo poder predictivo, o lo que es lo mismo, una alta función de pérdida. Los estimadores débiles normalmente son árboles de decisión, al igual que en el bosque aleatorio, pero en el GBoost se pueden utilizar otros modelos.
3. Modelo aditivo. Es un proceso secuencial por el cual en cada iteración se adiciona un árbol de decisión más que se entrena con los errores del árbol previo. Cada iteración debe reducir el valor de la función de pérdida. El entrenamiento se detiene una vez que la pérdida alcanza un nivel aceptable, ya no mejora en un conjunto de datos de validación externo o se alcanza el límite máximo de árboles establecido.

En resumen, el GBoost entrena de forma iterativa un conjunto de árboles de decisión a modo de estimadores débiles, y cada iteración utiliza los errores residuales del modelo anterior para ajustar el siguiente modelo y hacerlo así más robusto. Así finalmente, los modelos se agregan mediante una suma ponderada para obtener una predicción que minimiza el sesgo y ajusta mejor la predicción a los datos de entrada del modelo.

Algoritmo de XGBoost

El XGBoost es una implementación escalable y precisa del GBoost aplicado a árboles de decisión que maximiza el poder computacional de construcción de éstos para dinamizar y agilizar el rendimiento de los modelos de aprendizaje automático y velocidad de cálculo. Para ello el XGBoost construye los árboles de decisión en paralelo y aplica una estrategia por niveles, explorando los valores del gradiente y utilizando sumas parciales para evaluar la calidad de las divisiones en cada una de las posibles divisiones del conjunto de entrenamiento (XGBoost Developers, 2021). De hecho, el algoritmo parte de una sola hoja y añade iterativamente ramas al árbol evaluando las cualidades de cada división y eligiendo repetidamente las divisiones para cada nodo que minimizan la función de pérdida o de error de predicción con el objetivo de que el error de predicción

se lo más cercano a cero en cada iteración. Pero la gran diferencia, y mejora, entre el GBoost y el XGBoost es que mientras en el GBoost se emplean los errores del modelo anterior para robustecer el siguiente modelo, en el XGBoost se emplean árboles de decisión para robustecer el siguiente estimador.

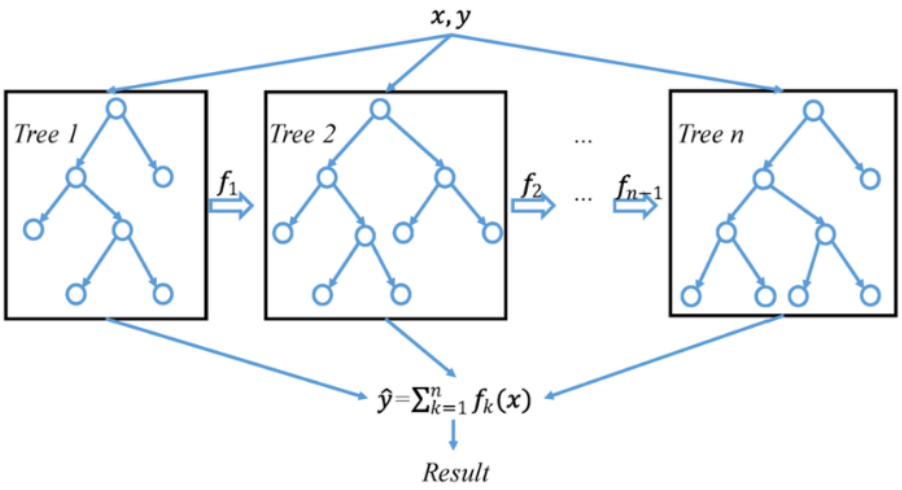

Figura 4.2. Esquema de XGBoost. Fuente: Researchgate, 2019.

La secuencia procedimental del XGBoost sería la siguiente:

1. Se toma una muestra aleatoria del conjunto de datos originales y se crea un estimador débil.
2. Se calculan los residuos del estimador con las observaciones reales, y se crea un segundo estimador que trata de predecir los errores del estimador anterior.
3. En la siguiente iteración se vuelve a crear un estimador débil que trate de predecir los errores conjuntos de los estimadores previos.
4. Repetir este proceso tantas veces como estimadores se desee crear.
5. Para evitar el overfitting del modelo emplear un valor de regularización (*learning rate*) que ajuste los pesos de cada estimador en el conjunto.

Parámetros

Ahora que ya se entiende el funcionamiento del XGBoost, también hay que mencionar los parámetros que se pueden modificar dentro de este modelo para mejorar el rendimiento del mismo, así como su poder predictivo (Gupta, 2021). Los parámetros o factores son:

- Parámetros similares a los del bosque aleatorio, previamente explicados. Como por ejemplo la profundidad de los árboles, el número de predictores observables en cada nodo, la aleatoriedad de la muestra, etc.
- Learning rate. También se conoce como valor de regularización, y se emplea para ponderar el peso de cada estimador débil, y afecta al ajuste del modelo a los datos. A menor learning rate mayor número de árboles se necesitan y el modelo se ajusta

mejor a los datos y viceversa. El valor de la tasa de aprendizaje varía de 0 a 1, siendo el valor más comúnmente utilizado el de 0,01.

Beneficios y desventajas del XGBoost

El XGBoost es un algoritmo cuya principal desventaja es su complejidad, pero tiene ciertas ventajas como son:

- Fácil implementación gracias a las librerías de código abierto donde diferentes desarrolladores pueden implementar mejoras continuas en el modelo. Además, es compatible con múltiples lenguajes e infinidad de hardware.
- Paralelización del proceso de boosting ya que aborda el proceso de construcción secuencial de árboles utilizando una implementación paralela.
- Árboles adaptativos y poda automática de los árboles. Al empezar a crear los árboles seleccionando una hoja y después añadir iterativamente ramas al árbol evaluando las cualidades de cada división y eligiendo repetidamente las divisiones para cada nodo que minimizan la función de pérdida, se mejora el rendimiento computacional de forma significativa ya que la extensión de los árboles varia.
- Mejor distribución de pesos en cada hoja. Debido a la forma de construcción de los árboles de decisión, el XGBoost da más peso a las hojas más puras lo que le permite obtener mejores predicciones.
- Optimización del hardware. Este algoritmo ha sido diseñado para hacer un uso eficiente de los recursos de hardware.

4.3 Comparativa Teórica

En este capítulo se han estudiado los dos algoritmos que posteriormente se van a aplicar al caso empírico de predicción de venta cruzada que son el bosque aleatorio y el XGBoost. Por este motivo a continuación se realizará una comparativa teórica entre ellos para resaltar sus parecidos y diferencias sobre el papel.

En un principio ambos son modelos de aprendizaje automático supervisado que emplean técnicas de ensamble que a partir de estimadores débiles crean un estimador fuerte para predecir. Además, ambos modelos usan árboles de decisiones como estimadores débiles, pero se diferencian principalmente en la forma que tienen de crear estos árboles y de posteriormente unirlos para crear el estimado fuerte.

Por un lado, el XGBoost se basa en el boosting y la idea de un aprendizaje iterativo secuencial. Esto significa que el modelo genera unas predicciones iniciales y posteriormente da más peso a los errores cometidos para mejorar la predicción en la siguiente iteración. Este proceso continúa como un ciclo aditivo. Por lo tanto, técnicamente, si se ha hecho una predicción, hay cierta seguridad en que no ha ocurrido de forma casual, sino con una comprensión profunda de los patrones que presenta el conjunto de datos.

Por otro lado, el bosque aleatorio se basa en el bagging o agregación por bootstrap lo cual le dota de un componente de mayor aleatoriedad donde cada árbol es entrenado de manera independiente y al final se promedian los resultados de cada árbol, en vez de ser aditivas. Esto no quiere decir que uno sea mejor que el otro, simplemente diferentes.

Respeto a la formación de los árboles de decisión el XGBoost poda directamente los árboles al construirlos empezando por una hoja y añadiendo iterativamente ramas al árbol evaluando las cualidades de cada división y eligiendo repetidamente aquellas para cada nodo que minimizan la función de pérdida hasta que sea mínima en cada árbol. Por el contrario, el bosque aleatorio construye los árboles de forma totalmente aleatoria con el método subespacial aleatorio y las muestras con reemplazo.

En definitiva, ambos modelos tienen cosas buenas y cosas malas, pero en general se puede decir que, dado que los bosques aleatorios plantean el entrenamiento de cada árbol de forma independiente, son bastante robustos y es menos probable que se ajusten en exceso a los datos de entrenamiento, lo que en teoría facilita que las predicciones sobre conjuntos de datos no vistos sean mejores. Mientras que el XGBoost es más difícil de ajustar los parámetros al ser más complejos, pero pueden ofrecer modelos más potentes si se utilizan correctamente.

5. CASO EMPÍRICO. ANÁLISIS DE LA CARTERA

Como se ha mencionado previamente en esta publicación, el objetivo final es un estudio comparado entre dos sistemas de aprendizaje automático supervisados muy comunes, como son el Bosque Aleatorio o "*Random Forest*" (Breiman, 2001) y el XGBoost (Chen & Guestrin, 2016) con el fin de predecir clientes potenciales en una campaña de venta cruzada y optimizar los recursos y esfuerzos de las compañías.

Como caso empírico de estudio se han aplicado estos dos algoritmos sobre la cartera de productos de Vida Individual de la compañía ASESUISA – Seguros SURA, de El Salvador. La Aseguradora Suiza Salvadoreña (ASESUISA) es una compañía fundada en 1969 con sede en San Salvador, El Salvador, proveedora de seguros y soluciones financieras que opera tanto en el sector minorista como banca. ASESUISA cuenta con más de 50 años de experiencia y una plantilla superior a 1.000 profesionales entre empleados, colaboradores y asesores. La compañía ofrece seguros de autos, vida, salud y propiedad para familias, y varias coberturas para empresas, como robo, seguros de colectivos, fianzas y responsabilidad civil, entre otros. Desde 2012 ASESUISA fue adquirida por Seguros Suramericana S.A (SURA) con sede en Colombia, con más de 70 años de experiencia (ASESUISA - Seguros SURA, 2020).

Para el estudio empírico se empleó la cartera de ASESUISA Vida Individual entre los períodos 2015 a 2020 compuesta por un total 11.738 asegurados. Esta cartera está compuesta de seis productos diferentes del ramo de Vida llamados:

- ASESUISA 2000
- Seguro Dotal
- Seguro Temporal
- Vida Entera
- Vida Múltiple
- Vida Personal

Aparte de una cobertura principal de mortalidad cada producto lleva asociados otras coberturas adicionales como dobles o triples indemnizaciones, desmembramiento, muerte accidental o gastos funerarios entre otras. A continuación de proceder a mostrar un perfil de la cartera indicando la distribución por género, por edad, etc.

Distribución por género

La distribución por genero de la cartera de ASESUISA se compone de un 39,89% de mujeres frente a un 60,11% de hombres. Sin embargo, en 2020, acorde con los datos del Banco Mundial (2022) la población salvadoreña estaba compuesta en 2020 de un total de 6.486.201 personas, de las cuales 3.036.424 eran hombre y 3.449.777 mujeres. Es decir, la distribución por sexo de la población de El Salvador era de un 47% de hombres y un 53% de mujeres.

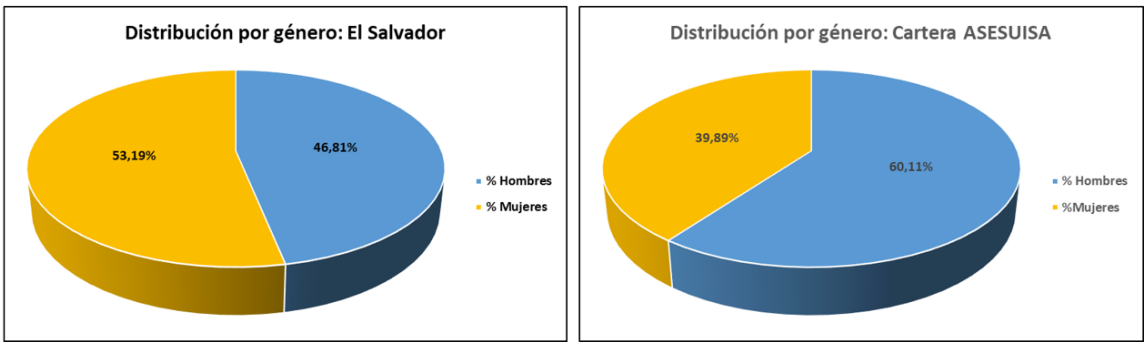

Figura 5.1. Porcentaje de hombres y mujeres en la cartera de ASESUISA (derecha) y la población de El Salvador (Izquierda). Fuente: Elaboración propia.

Esto es reseñable porque se observa cómo la población asegurada es dispar a la población total de El Salvador, siendo más presente el sexo masculino dentro de la población asegurada. Esto probablemente sea debido a motivos como una menor participación del mercado laboral por parte de las mujeres, y por consecuente un menor nivel de ingresos que les permita adquirir un seguro como indica el artículo de LA Network (Carrero, 2022).

Distribución de edades

En lo que respecta a la distribución por edades se observa que la mayoría de los individuos que componen la cartera se encuentran comprendidos entre los 30 y los 50 años siendo la edad mínima de 18 años y la máxima de 90. Se puede apreciar que apenas existe diferencia entre la forma de la distribución de hombres y mujeres teniendo incluso cuartiles similares. La única diferencia apreciable es la mencionada previamente sobre la mayor cuantía de hombres que de mujeres.

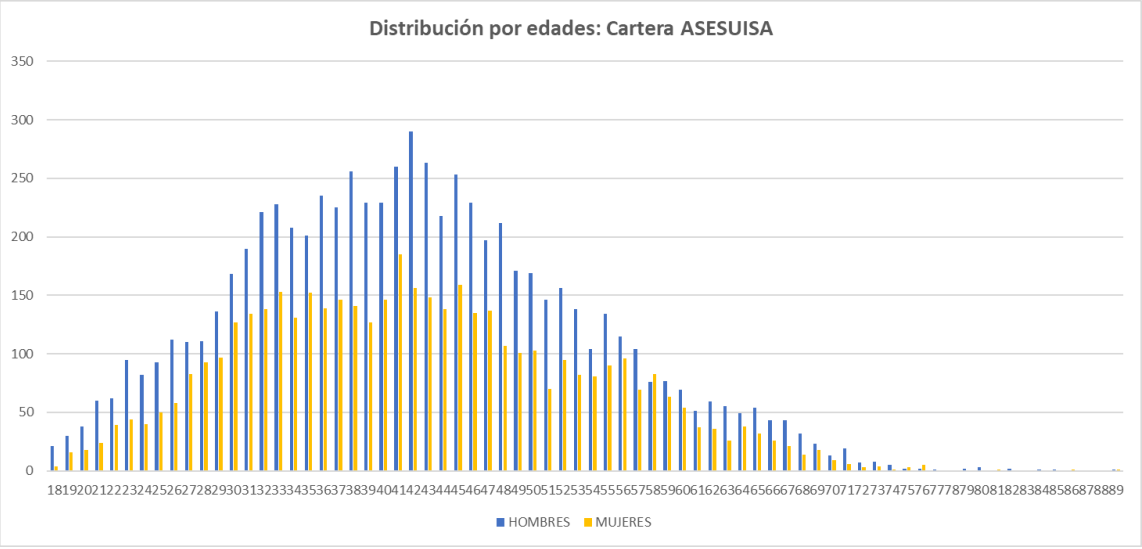

Figura 5.2. Distribución por edades de hombres y mujeres en la cartera de ASESUISA. Fuente: Elaboración propia

	Min	*25% Cuartil*	*Mediana*	*Media*	*75% Cuartil*	*Máximo*	*Moda*
Hombres	18	35	43	43,02	52	90	43
Mujer	18	34	42	43,24	50	90	42

Tabla 5.1. Datos de la distribución por edades. Fuente: Elaboración propia

En comparación con la pirámide poblacional de El Salvador a fecha de diciembre de 2020 (Expansión, 2022), se observa que más o menos la estructura es similar. Aun así, sí es verdad que, en edades jóvenes, por debajo de los 20 años, la población del salvador es superior a la que conforma la cartera. Pero ello se explica debido a que la población asegurada suele tener en promedio ingresos superiores a la no asegurada (Instituto Nacional de Estadística e Informática, 2018).

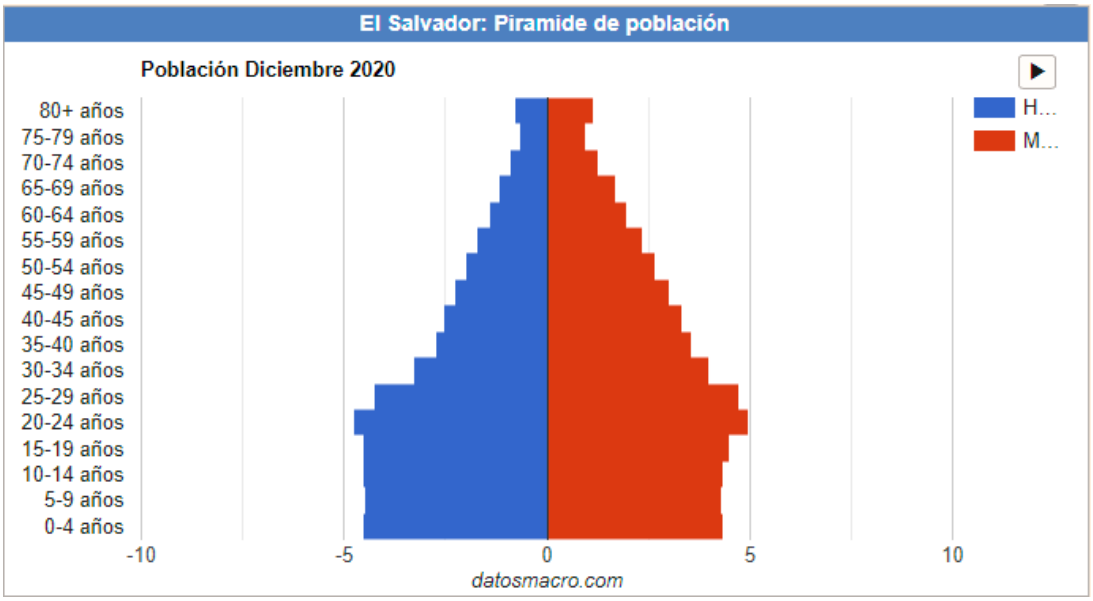

Figura 5.3. Pirámide poblacional de El Salvador en diciembre 2020. Fuente: Expansión

También sorprende que en la estructura de la cartera haya bastante asegurados con una esperanza de vida superior al promedio de la población salvadoreña que es de 74 años, tanto para hombres (69 años) como para mujeres (78 años) de acuerdo con los datos del Banco Mundial (2022).

Distribución por Suma Asegurada

Realizando un estudio de la distribución de la suma asegurada se obtiene que el total de la suma asegurada es de $1.682.861.497,95, con un promedio de $143.380,89 por asegurado. Sin embargo, si se disgrega por sexo se observa que el 64,16% corresponde a los hombres, haciendo un total de $1.079.752.324,37 para este grupo, y el 35,84% restante corresponde a mujeres con una cantidad de $603.109.173,58, en concordancia con la distribución por género.

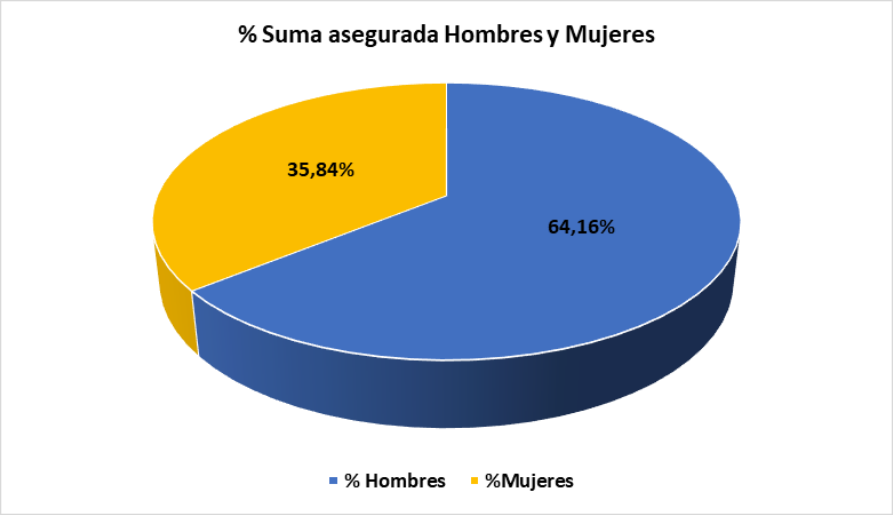

Figura 5.4. Porcentaje de suma asegura por género en la cartera de ASESUISA. Fuente: Elaboración propia

Como es natural, la distribución de la suma asegurada por edades se corresponde y es muy similar a la distribución por edades del porfolio, siendo la mayor cuantía entre las edades 40 y 46. Esto indica que no existe ningún capital especialmente grande, a excepción de la edad 63-64 donde sí se puede observar un incremento en la suma asegurada para esa franja por encima de lo normal.

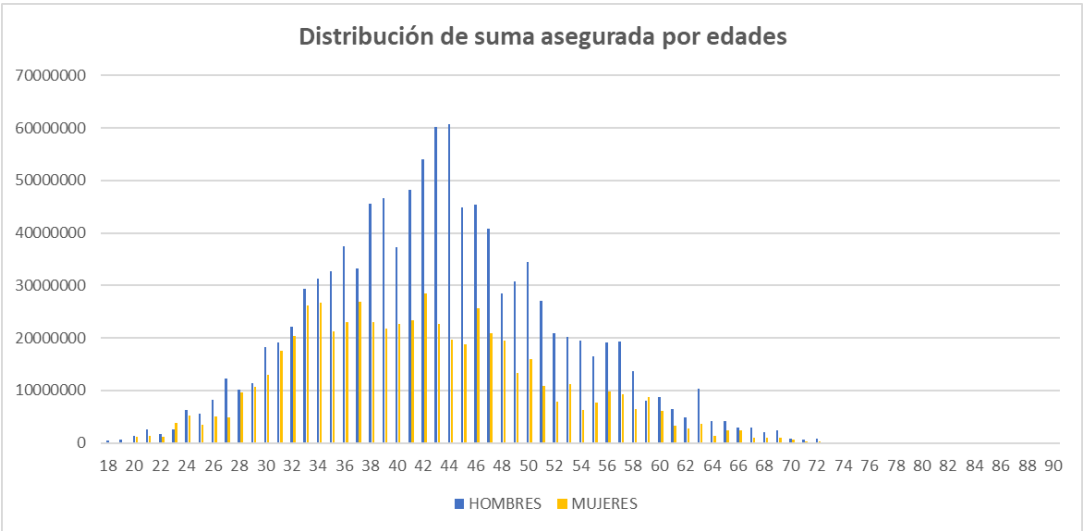

Figura 5.5. Distribución de la suma asegurada por edades. Fuente: Elaboración propia

A continuación, se muestra un gráfico de la distribución de sumas aseguradas. En él se puede ver los niveles de suma asegurada más frecuentes. Siendo el más frecuente una suma asegurada $151.800 con más de 700 clientes (400 hombres y 300 mujeres) con esa cantidad asegurada, seguido por $102.300 como segunda cuantía asegurada más frecuente.

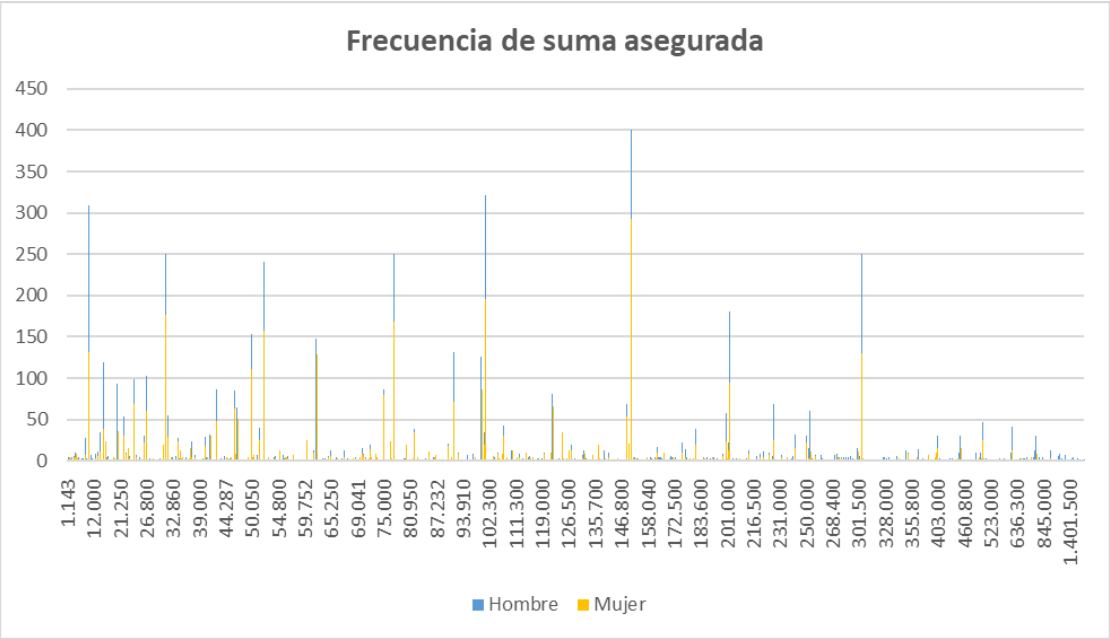

Figura 5.6. Frecuencia de suma asegurada. Fuente: Elaboración propia

	Min	*25% Cuartil*	*Mediana*	*Media*	*75% Cuartil*	*Máximo*	*Moda*
Hombres	*1.143*	*41.800*	*91.800*	*153.174*	*186.660*	*3.451.800*	*151.800*
Mujer	*1.715*	*46.500*	*81.800*	*127.806*	*151.800*	*3.000.000*	*151.800*

Tabla 5.2. Porcentaje de hombres y mujeres en la cartera de ASESUISA. Fuente: Elaboración propia

Distribución por Productos (Hombre/mujer)

En lo que respecta a la distribución por el tipo de producto es prácticamente idéntica para ambos sexos, siendo los productos más comprados el vida temporal, vida personal y vida múltiple.

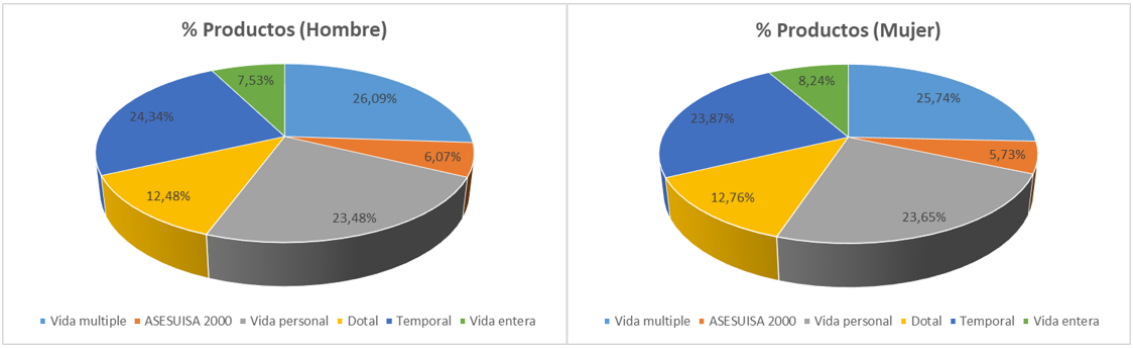

Figura 5.7. Distribución de productos entre hombres y mujeres en la cartera de ASESUISA. Fuente: Elaboración propia

Una vez presentado el perfil de la cartera y ser contrastada con la población salvadoreña, la cartera de Vida Individual de ASESUISA fue cruzada con la cartera de Salud de la misma compañía en el mismo periodo (2015 - 2020) compuesta por 55.175 asegurados. El solapamiento, o cruce, existente entre la cartera de Vida Individual con la cartera Salud obtenido fue de un total de 2.428. Es decir, existen 2.428 clientes del total de 11.738 de la cartera de Vida que también tienen productos de Salud, lo que corresponde con un 20,68% de la cartera de Vida Individual ha adquirido productos de Salud. Esto es un muy buen margen para poder plantear un estudio de venta cruzada sobre esta cartera siguiendo el planteamiento previamente de Amazon.com, Inc de los clientes que compraron "A" también compraron "B", y ver que productos se suelen comprar conjuntamente.

6. ESTUDIO COMPARADO EMPÍRICO

6.1 Tratamiento de datos

Antes de llevar a cabo el experimento en R-Studio fue necesario instalar los siguientes paquetes o bibliotecas de funciones:

- randomforest
- xgboost

La primera biblioteca, llamada "*randomforest*", es una biblioteca de R que contiene el algoritmo del bosque aleatorio de Breiman (basado en el código Fortran original de Breiman y Cutler) para la clasificación y la regresión. También puede usarse para el aprendizaje automático no supervisado que permite evaluar la distancia entre los puntos de datos (RDocumentation, 2022).

La biblioteca número dos con el nombre de "*xgboost*" permite implementar de forma eficientemente las funciones necesarias que subyacen al algoritmo de XGBoost. Esta librería, además, sirve de interfaz en R para el usuario. El paquete incluye un solucionador de modelos lineales y algoritmos de aprendizaje en árbol (XGBoost Developers, 2021). La librería permite realizar automáticamente cálculos paralelos en el ordenador, lo que hace que sea mejor que otras librerías de XGBoost existentes. También admite varias funciones objetivo, como la regresión, la clasificación y el ranking, o incluso aquellas funciones que otros usuarios desarrollen e incluyan en la librería.

Una vez instalados los paquetes de R-studio el siguiente paso fue el tratamiento de los datos. En un principio existían cinco ficheros donde cada uno contenía las pólizas existentes en la cartera de Vida Individual de cada año, desde el 2015 al 2020. Esos ficheros fueron unidos y se eliminaron duplicados. De esa forma se obtuvo una base de datos con un total de 11.738 asegurados para la cartera de Vida Individual de ASESUISA entre los años 2015 a 2020. Este mismo proceso fue repitió para los ficheros de la cartera de Salud, donde se obtuvo un registro con un total de 55.157 asegurados para el periodo 2015 a 2020. Finalmente, estos registros se cruzaron para observar que asegurados estaban presentes en ambas bases de datos y así obtener la variable que se desea predecir, que es la variable que indica si ha existido o existe venta cruzada entre Vida Individual y Salud para cada asegurado.

Esto es en lo que respecta a la variable que se desea predecir, sin embargo, la siguiente lista enumera la cantidad de predictores o variables observadas para predecir la existencia de venta cruzada y una breve descripción de cada una:

NOMBRE	DESCRIPCIÓN	CÓDIGO
ID POLIZA	Nº aleatorio de identificación de póliza (no se corresponde con el nº real de póliza de ASESUISA)	Numérico
SEXO	Variable que indica el sexo del tomador	1 = Hombre 2 = Mujer
FUMADOR	Variable que indica si la persona es fumadora o no	0 = No fumador 1 = Sí es fumador
MATRIMONIO	Variable que indica el estado civil de la persona	0 = No casado 1 = casado
CANAL_DISTRIBUCION	Variable que indica cual fue el canal de venta de la póliza.	1= Agente 2 = Ofina 3 = Digital
EDAD_ACTUAL (2020)	Variable numérica que indica la edad de la persona.	Numérico
CAP_PAGO	Variable que indica el percentil de la ratio de la prima total pagada en los últimos 5 años y la suma asegurada ponderado por la duración máxima.	1 = percentil 0 a 33. 2 = percentil 34 a 66. 3 = percentil 67 a 100.
NET_VALUE	Variable que indica el percentil en el que se encuentra el asegurado en función de la suma asegurada.	1 = percentil 0 a 20. 2 = percentil 21 a 40. 3 = percentil 41 a 60. 4 = percentil 61 a 80. 5 = percentil entre 80 y 100.
REGION	Variable que indica la región en la que reside el tomador.	1 = Occidental 2 = Central 3= Paracentral 4 = Oriental
OCC_SECTOR	Variable que indica el sector ocupacional en el que se encuentra el tomador.	1 = Primario 2 = Secundario 3 = Terciario
PREMIUM_5_Y	Variable numérica que indica la prima total pagada durante los últimos 5 años del asegurado.	Numérico

AVG_PREM_5_YEAR	Variable numérica que indica la prima promedio de los últimos 5 años del asegurado.	Numérico
LAST_PREMIUM	Variable numérica que toma el valor de la última prima pagada por el asegurado	Numérico
PRODUCTO	Variable que indica qué producto de Vida ha adquirido el tomador.	1 = Vida múltiple 2=ASESUISA 2000 3= Vida Personal 4 = Dotal 5 = Vida Temporal 6= Vida Entera
EDAD_ENTRADA	Variable numérica que indica a qué edad el asegurado entró a formar parte de la cartera.	Numérico
EDAD_SALIDA	Variable numérica que indica a qué edad el asegurado dejó de formar parte de la cartera. Si el asegurado mantiene su póliza vigente la edad tenida en cuenta es la actual.	Numérico
NB	Variable que indica si la póliza forma parte del nuevo negocio o no. Es decir, el periodo de vigencia que lleva trascurrido desde su inicio es menor a 1 año.	0 = No es nuevo negocio 1 = Nuevo negocio
RAMO	Valor numérico que indica de qué ramo forma parte la póliza.	18, 19 o 43
PLAZO_PAGO	Variable que indica el plazo de pago máximo en años.	Numérico
FORMA_PAGO	Variable que indica la periodicidad de pago (Anual, semestral, trimestral o mensual).	1 = Anual 2 = Semestral 3 = Trimestral 4 = Mensual.
SA_TOT	Variable numérica que indica la suma asegurada total de la póliza teniendo en cuanta las coberturas adicionales.	Numérico
SA_YEAR	Variable numérica que indica la cuantía anual de suma asegurada teniendo en cuenta la duración máxima de la póliza.	Numérico
SA_BAS	Variable numérica que indica el valor de la suma asegurada de la cobertura principal sólo (muerte).	Numérico
COB_BAS	Variable que índica si la póliza tiene cobertura principal o no.	1 = sí existe cobertura principal 0 = No existe cobertura principal.

SA_AD	Variable numérica que indica el valor de la suma asegurada sólo de las coberturas adicionales en caso de que haya como pueden ser doble indemnización, gastos funerarios, discapacidad, etc.	Numérico
NCOB_AD	Variable numérica que índica cuantas coberturas adicionales tiene la póliza	Numérico
RATIO_SAAD/SABAS	Variable numérica que indica la ratio existente entre la suma asegurada adicional y la suma asegurada de la cobertura principal.	Numérico
PROD_VI_AD	Variable que indica si el asegurado ha comprado otro producto de Vida	0 = no ha comprado otro producto de Vida 1 = sí ha comprado otro producto de Vida
YEAR_ISSUE	Variable que indica el año de emisión de la póliza	Numérico
YEAR_END	Variable que indica el año de caída/cancelación de la póliza	Numérico
DUR_REAL	Variable que indica la duración trascurrida desde la fecha de emisión	Numérico
STATUS	Variable que indica si la póliza está vigente, ha tenido algún siniestro y ha terminado o si ha sido cancelada	1= En vigor 2= Lapse 3= Siniestro
SINIESTRO	Variable que indica si los siniestros que ha tenido la póliza son parciales o no.	0 = No hay siniestro 1 = siniestro no parcial 2= Parcial
MAX_DUR	Variable que indica la vigencia máxima en años.	Numérico
SALUD	Variable que se desea predecir. Es una variable binaria que indica si el asegurado ha comprado o no un producto de Salud de ASESUISA.	0 = No ha realizado venta cruzada. 1 = Sí ha realizado venta cruzada

Tabla 6.1. Explicación de cada variable. Fuente: elaboración propia

Se disponía de un total de 34 predictores (la variable de identificación no se tiene en cuenta) de los cuales algunos se obtuvieron de la propia información facilitada por ASESUISA como pueden ser la suma asegurada, el sexo, la edad, el producto, el inicio de vigencia, fin de vigencia, si es o no nuevo negocio, etc. Otras variables tuvieron que ser derivadas de estas métricas originales como:

- Capacidad de pago. Este indicador se ha derivado como el percentil en el que se encuentra el porcentaje de suma asegurada pagado en los últimos cinco años respecto a la duración máxima posible de cada asegurado. Es decir, si un asegurado ha pagado el 80% de la suma asegurada en los últimos cinco años estará en un percentil superior a otro que solo ha pagado el 5% de la suma asegurada.
- Valor neto. Se ha calculado como el percentil del individuo en función de la suma asegurada total.
- Prima total de los últimos 5 años. Se ha obtenido mediante la suma de la prima en los últimos cinco años.
- Prima promedio de los últimos 5 años. Se ha calculado como el promedio de las primas pagadas en los últimos cinco años.
- Suma asegurada por año de duración de la póliza. se ha obtenido dividiendo la suma asegurada total entre la máxima duración de la póliza.
- Ratio de suma asegurada adicional sobre suma asegurada de la cobertura básica. se ha calculado dividiendo la suma asegurada de las coberturas adicionales entre la suma asegurada de la cobertura principal.

Sin embargo, con el fin de robustecer aún más la base de datos, en especial con variables de carácter socioeconómico a los que quizás ASESUISA no tiene acceso o no puede facilitarlas, se simularon ciertas variables como variables aleatorias siguiendo la distribución poblacional de El Salvador. Entre estas variables simuladas nos encontramos con:

- Matrimonio. Indica si la persona está casada o no. Está información no fue proporcionada por ASESUISA, por eso en base a los datos demográficos de El Salvador donde se establece que el 58,7% de la población es soltera se ha simulado esta variable (Banco Mundial, 2022).
- Región. siguiendo con los datos demográficos se ha simulado una variable que indica la zona en la que vive cada individuo siguiendo la misma proporción que en El Salvador que es la siguiente:

ZONA	***POBLACIÓN***	***PORCENTAJE %***
OCCIDENTAL	1.282.118	24,13%
CENTRAL	2.222.904	41,82%
PARACENTRAL	619.058	11,65%
ORIENTAL	1.190.881	22,40%

Tabla 6.2. Distribución geográfica poblacional de El Salvador. Fuente: Banco mundial, 2022

- Sector ocupacional. Esta información la aseguradora no la proporcionó, por ello en base al estudio de la consultora Deloitte sobre El Salvador (2020) se ha simulado esta variable donde un 16% se decida al sector primario, un 21% se dedica al sector secundario y el restante 63% al terciario.

- Fumador. La aseguradora tampoco facilito estos datos, así que en base a los datos del Banco mundial (2022) donde se estima que en El Salvador un 11,7% de la población es adicta al tabaco, se ha simulado esta variable.

Una vez que se obtuvieron todas las variables, para facilitar el trabajo de lectura de datos y cálculo a los algoritmos de bosque aleatorio y XGBoost los datos fueron codificados numéricamente de tal forma que por ejemplo la variable "*Fumador*" se convierte en una variable dummy donde toma el valor 0 en caso negativo y 1 en caso positivo, o la variable "*Sexo*" que toma el valor 1 en caso de ser hombre y 2 en caso de ser mujer. Y así con todas las variables excepto las ya numéricas como suma asegurada que mantienen su valor original.

6.2 Análisis previo

Con la base de datos final completa se decidió crear dos subconjuntos de datos. Uno que se empleara para el entrenamiento de los algoritmos y otro para la validación de los resultados y así evitar el sobreajuste en los datos. Para hacer esta partición se decidió optar por el método 80/20 como dicen los autores Afshin Gholamy, Vladik Kreinovich y Olga Kosheleva en su estudio "*Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation*" (2018) que determinan que es el mejor. Bajo este método el 80% de los datos originales se emplea para entrenar al algoritmo y el 20% restante se usan para validar.

Ahora, una vez han sido creados los dos subconjuntos de datos, para que funcione la validación se debe comprobar que la estructura de los datos sea la misma, o similar al menos, y para ello se comprobó que el porcentaje de individuos que habían realizado venta cruzada en ambos subconjuntos de datos era similar al original. Efectivamente el resultado tanto en el conjunto original como en los dos subconjuntos era de un 20% aproximado.

% DE INDIVIDUOS QUE HAN REALIZADO VENTA CRUZADA		
Conjunto de datos original	***Subconjunto de datos de entrenamiento***	***Subconjunto de datos de validación***
20,68 %	20,23 %	20,80 %

Tabla 6.3. Tasas de solapamiento. Fuente: Elaboración propia

Posteriormente se procedió a un análisis de la matriz de correlación. La matriz de correlación mide tanto la relación existente entre las variables como su fuerza, y puede tomar valores entre menos uno y uno. Cuanto mayor sea la relación entre dos variables, es decir, si cuando una aumenta la otra también aumenta en una magnitud similar, presentarán un coeficiente de correlación cercano a uno, e indicará que ambas variables miden características similares. Por el contrario, si la relación es cercana cero o negativa implica que las variables miden características diferentes o contrarias.

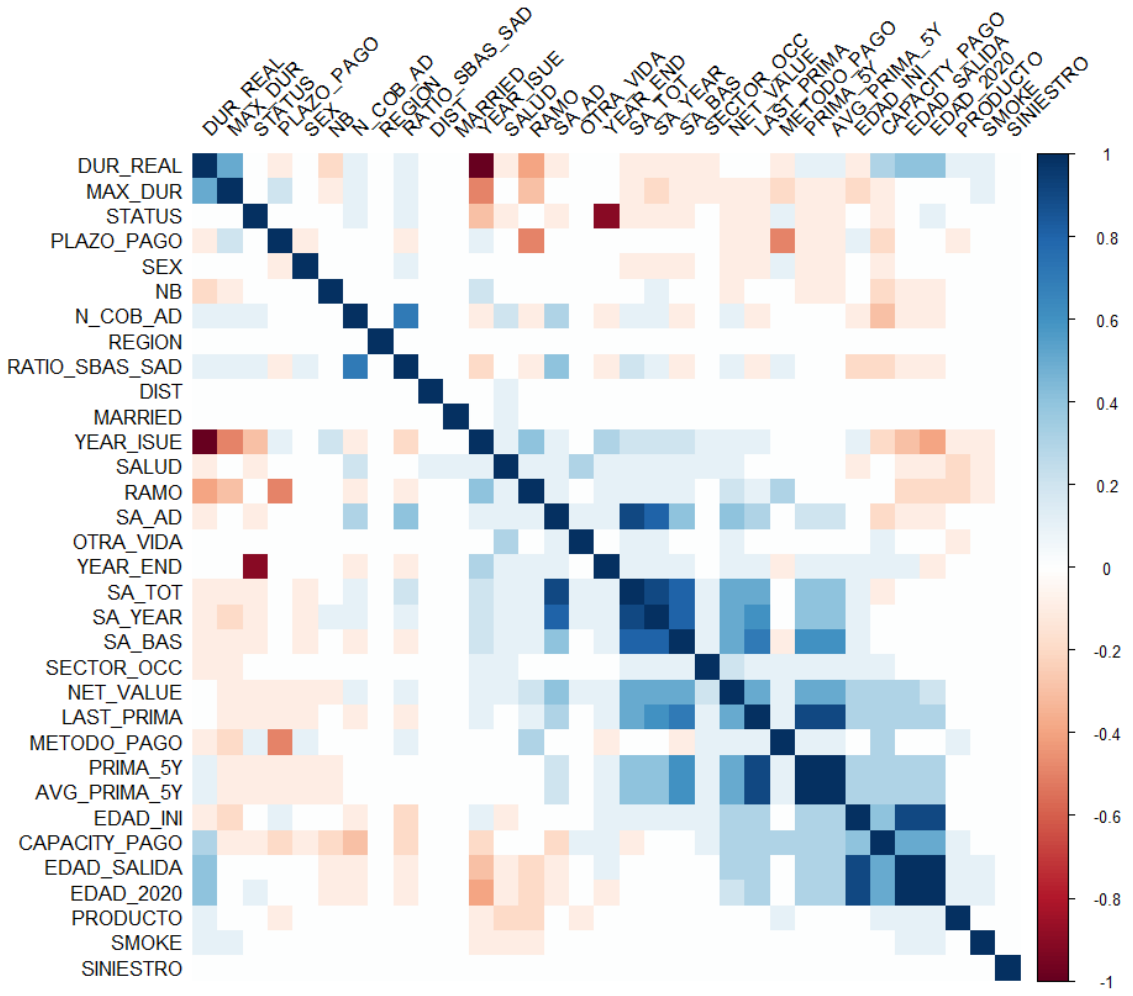

Figura 6.1. Matriz de correlación. Fuente: elaboración propia

En la imagen superior se puede apreciar como casi todas las variables presentan una correlación cercana a cero, a excepción de un pequeño grupo de variables que si presentan una relación algo superior a cero como son la suma asegurada total y la suma asegurada por año, por ejemplo, o la edad de salida y la edad del individuo en 2020. Esto se debe, efectivamente a que miden características similares que están relacionadas, pero la relación que presentan es inferior a 0,5 por lo que no es una relación excesivamente fuerte.

6.3 Fase de entrenamiento

Bosque aleatorio

Una vez ya decididas las variables que se van a emplear en el modelo, el siguiente paso es seleccionar los parámetros óptimos y después entrenar los modelos para poder comparar cuál de los dos es mejor.

El análisis de los parámetros puede hacerse de forma aislada y ver parámetro a parámetro qué valor es el mejor. Sin embargo, este enfoque no es el correcto ya que los parámetros interactúan entre sí y se afectan unos a otros. Por ello el mejor método es llevar a cabo un "*Grid search*" o búsqueda de rejilla (The Machine Learners, 2022). Es decir, establecer varios valores para cada parámetro y crear múltiples combinaciones aleatorias entre éstos

para después explorar que combinación aleatoria es la que minimiza el OOB, el error estándar o maximiza el AUC. Los parámetros que se pueden ajustar son los siguientes:

- Ntree. Este parámetro indica el número de árboles de decisión que crea el algoritmo. Como se explicó previamente en el apartado sobre el bosque aleatorio a mayor número de árboles mayor precisión, pero mayor coste computacional también. Por eso hay que encontrar el punto óptimo en el cual más número de árboles ya no mejore la predicción. El valor por defecto es de 500 árboles de decisión.
- Mtry. Este parámetro es el que permite decidir cuantas variables son observadas a la hora de crear un nodo en un árbol de decisión. Es decir, afecta a la característica denominada subespacio muestral aleatorio comentada en los capítulos anteriores. Al igual que con "*ntree*", hay que encontrar ese valor óptimo que maximice el poder de predicción minimizando el coste computacional. Por defecto, el valor tomado por el bosque aleatorio es el redondeo a la baja de la raíz cuadrada del número de predictores. Es decir, en este caso sería el valor anterior superior de la raíz de 34 que serían 5 variables.
- Max.depth. Este parámetro indica la profundidad de los árboles de decisión creados. A mayor profundidad se dan árboles más precisos, pero más lentos y sobreajustados.

A la hora de realizar el grid, se tuvieron en cuenta los siguientes parámetros:

HIPERPARÁMETROS PARA EL GRID SEARCH						
Nº trees	100	300	500	1.000	3.000	
Mtry	3	5	7	10	15	33
Max.depth	3	5	7	10	20	

Tabla 6.4. Tabla con los parámetros de grid seacrh. Fuente: Elaboración propia

Realizando el Grid aleatorio el resultado de los parámetros que maximizan la precisión y optimizan el resultado para el algoritmo del bosque aleatorio es el siguiente:

mtry	trees	max.depth	.metric	.estimator	mean	n	std_err
<dbl>	<dbl>	<dbl>	<chr>	<chr>	<dbl>	<int>	<dbl>
15	500	20	accuracy	binary	0.917	5	0.00153

Figura 6.2. Parámetros óptimos del modelo Random Forest. Fuente; Elaboración propia

De esta forma el modelo del bosque aleatorio tendrá un total de 500 árboles de decisión, con una profundidad de 20 nodos y en cada nodo se observarán aleatoriamente 15 variables. Estos parámetros minimizan el error del modelo a 0,00153 y maximizan la precisión en un total de 91,7%. Una vez encontrados los parámetros óptimos, hay que entrenar el modelo final y posteriormente realizar las predicciones. A continuación, se muestra el proceso de entrenamiento del XGBoost.

XGBoost

El proceso de entrenamiento del XGBoost fue similar aplicando también un g*rid search* para encontrar los parámetros óptimos y posteriormente entrenar el modelo con esos parámetros para maximizar el área bajo la curva (AUC por sus siglas en inglés) y realizar las predicciones. Los parámetros empleados para el grid search fueron los mismos que en el bosque aleatorio, más el parámetro del "*learning rate*" que indica los pesos de cada estimador débil. Entre esos posibles valores de learning rate se encontraban 0,01;0,1 y 0,3. Finalmente los parámetros óptimos para el modelo XGBoost fueron:

mtry	trees	tree_depth	learn_rate	sample_size	.metric	.estimator	mean	n	std_err
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<chr>	<dbl>	<int>	<dbl>
33	500	5	0.1	0.75	accuracy	binary	0.950	5	0.00354

Figura 6.3. Parámetros óptimos del modelo XGBoost. Fuente: Elaboración propia

De esta forma, el modelo de XGBoost tendrá 500 iteraciones, es decir 500 estimadores débiles, cada uno con un learning rate de 0,1, los cuales tendrán una profundidad máxima de 5 nodos con un subespacio muestral aleatorio de 33 predictores. Esta configuración parece otorgar una precisión de un 95% al modelo que es superior a la del bosque aleatorio. Terminado los procesos de entrenamiento se realizaron las predicciones y observaron ciertas métricas como la matriz de confusión, la curva ROC, el AUC y el tiempo empleado en cada proceso como se exponen a continuación.

6.4 Resultados y comparativa

6.4.1 Predicción y Matriz de confusión

La matriz de confusión es una manera de representar los resultados de las predicciones y además una métrica muy buena y ampliamente utilizada para observar la eficacia y el rendimiento del modelo de aprendizaje automático (Narkhede, 2018).

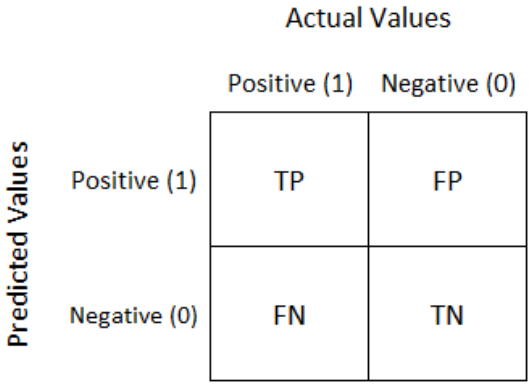

Figura 6.4. Esquema de matriz de confusión. Fuente: Towards Data Science, 2018

Su funcionamiento es muy sencillo y fácil de comprender. Se puede aplicar para problemas de clasificación de dos clases en adelante, pero para explicar el concepto es mejor reducirlo al mínimo que es como mejor se entiende. Como se observa en el esquema de arriba se dispone una matriz con cuatro casillas con los valores reales arriba

y los valores predichos a la izquierda y en cada casilla se almacenan las predicciones en función de si la predicción es correcta o no, y del tipo de error cometido (error tipo 1 o tipo 2). A continuación, se explica cada casilla:

- Primera casilla. Esta casilla se encuentra arriba a la izquierda y se llama casilla de verdaderos positivos (TP). Se llama así porque en ella se almacena aquellas predicciones que han sido predichas de forma correcta y cuya clase es positiva. Para el experimento realizado de venta cruzada en esta casilla estaría el número de observaciones que efectivamente han realizado venta cruzada con Salud y han sido predichos de forma correcta.
- Segunda casilla. Se encuentra arriba a la derecha y se denomina casilla de falsos positivos (FP). En esta casilla se almacenan las predicciones cuyo valor real es negativo, es decir, no han realizado venta cruzada, pero el modelo las ha clasificado como que sí han realizado venta cruzada. Este error es conocido como error de tipo 1.
- Tercera casilla. Se encuentra abajo a la izquierda y es conocida como casilla de falsos negativos (FN). Este caso es contrario al caso anterior y en ella se encuentra aquellas observaciones que han dicho predichas como que no han realizado venta cruzada, pero en realidad sí lo han realizado. Es conocido como error de tipo 2.
- Cuarta casilla. Se encuentra abajo a la derecha y es la casilla donde se almacenan aquellas observaciones que no ha realizado venta cruzada y han sido predichas de forma correcta, también llamados verdaderos negativos (TN).

La matriz de confusión es muy importante porque de ella se pueden obtener múltiples indicadores sobre la eficacia de un modelo como son:

- Exactitud. Mide la proporción total de resultados correctos. Tanto verdaderos positivos como verdaderos negativos sobre el total de valores.

$$Exactitud = \frac{TP + TN}{Total}$$

- Precisión. Mide la proporción de acierto dentro de sólo los casos positivos. A efectos más gráficos mide la dispersión de las observaciones.

$$Precisión = \frac{TP}{TP + FP}$$

- Sensibilidad. Es la capacidad del modelo de discriminar los casos positivos de los negativos. Se calcula como la proporción de los verdaderos positivos predichos sobre todos los casos positivos reales.

$$Sensibilidad = \frac{TP}{TP + FN}$$

- F-Score. Es una mezcla de la precisión y la sensibilidad en una sola métrica y sirve para casos donde la distribución de las clases no es pareja de tal forma que el F-Score pondera los valores. Se calcula siguiendo la siguiente formula.

$$F1 = \frac{2 * Precisión * Sensibilidad}{Precisión + Sensibilidad}$$

De todos estos indicadores se pueden extraer conclusiones interesantes como que un modelo de aprendizaje automático que presente una alta precisión y sensibilidad en sus predicciones es un buen modelo para ese conjunto de datos, o que, si presenta una alta sensibilidad, pero baja precisión es un buen modelo diferenciando las clases, pero no es bueno prediciendo, y viceversa. Sin embargo, la matriz de confusión se ve afectada por el umbral establecido en el modelo para determinar cuándo una predicción es considerada positiva. Por defecto suele ser 0,5, pero en caso de establecer el umbral en 0,8 el caso de positivos predichos será menor ya que se exige una probabilidad el 80% para ser elegido como positivos.

Una vez comprendido el concepto de matriz de confusión y todas las métricas que se pueden obtener de ella a continuación se muestran las matrices de confusión del bosque aleatorio y del XGBoost.

Bosque aleatorio

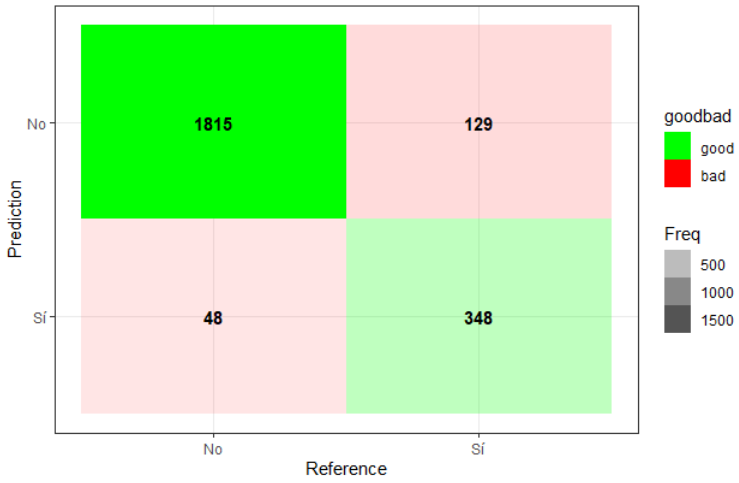

Figura 6.5. Matriz de confusión del Bosque Aleatorio. Fuente: Elaboración propia

XGBoost

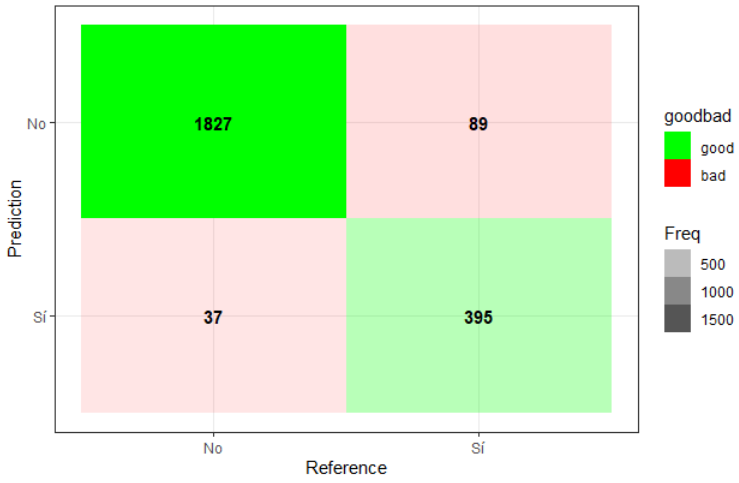

Figura 6.6. Matriz de confusión del XGBoost. Fuente: Elaboración propia

Bosque Aleatorio	
Exactitud	92,14 %
Precisión	87 %
Sensibilidad	72,65 %
F-Score	79,18 %

XGBoost	
Exactitud	94,63 %
Precisión	91,43 %
Sensibilidad	81,61 %
F-Score	86,24 %

Tabla 6.5. Tabla comparativa del XGBoost (dcha.) y Bosque aleatorio (izqda..). Fuente: Elaboración propia

Como se puede observar entre los dos modelos el mejor es el XGBoost ya que su porcentaje de acierto es ligeramente superior al del bosque aleatorio, y por ello las métricas que obtiene de exactitud, predicción, sensibilidad y F-Score son ligeramente mejores. A continuación, se observará la curva ROC y el AUC, pero observando la matriz de confusión el XGBoost será superior.

6.4.2 ROC y AUC

La curva ROC ("Receiver Operating Characteristic" que significa "Características operativas del receptor") y el área bajo la curva (AUC) son dos métricas empleadas para la evaluación de un modelo de aprendizaje automático que realiza clasificaciones binarias y mide la sensibilidad y la precisión (Bhandari, 2020).

La curva ROC es una curva de probabilidad que traza la tasa de verdaderos positivos contra la tasa de falsos positivos en varios valores de umbral para la matriz de confusión. El AUC es la medida de la capacidad de un clasificador para distinguir entre clases (sensibilidad) y se utiliza como resumen de la curva ROC.

En una curva ROC, a medida que se avanza en el eje X indica un mayor número de falsos positivos. Mientras que un valor más alto del eje Y indica un mayor número de verdaderos positivos. Así pues, la elección del umbral depende de la capacidad de equilibrio entre los falsos positivos y los falsos negativos.

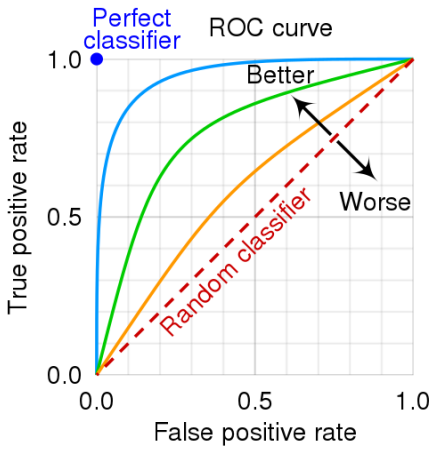

Figura 6.7. Esquema de curva ROC y AUC. Fuente: Wikipedia, 2018

Como se observa en la imagen superior, si el AUC es igual a uno, entonces el modelo predice perfectamente los datos, y si fuese cero entonces cambiaria los valores positivos por negativos y viceversa. Si el AUC es 0,5 entonces no es capaz de diferenciar las clases y la predicción es totalmente aleatoria. Por ello cuanto más cercano a 1 mejor sensibilidad tiene el modelo para clasificar correctamente.

Ya explicados estos conceptos a continuación se muestran la curva ROC y AUC de los modelos de XGBoost y bosque aleatorio para el caso de venta cruzada de ASESUISA.

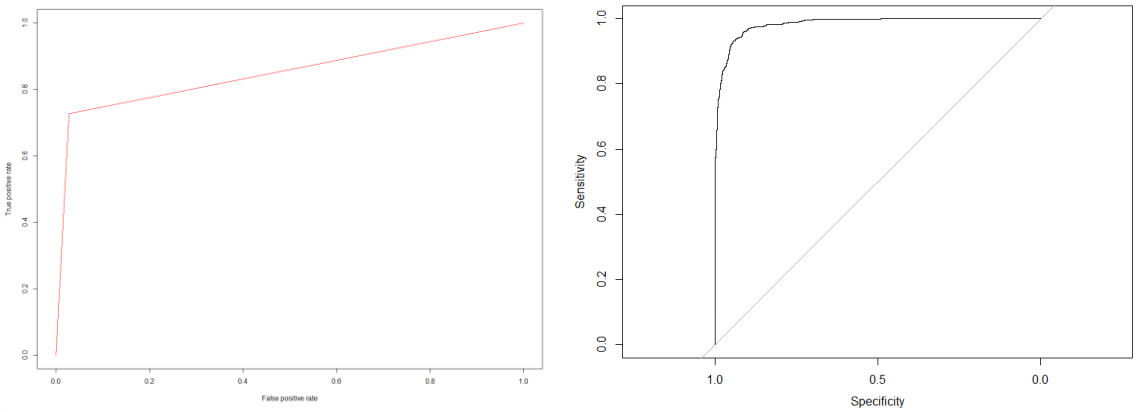

Figura 6.8. Curva ROC del Random Fortest (izqa.) y XGBoost (dcha.). Fuente: Elaboración propia

Bosque aleatorio AUC	XGBoost AUC
84,9%	97,9%

Tabla 6.6. Resultado ROC/AUC de Randon forest (izqa.) y XGBoost (dcha.). Fuente: Elaboración propia

Se observa, en concordancia con la matriz de confusión, que la forma de la curva ROC y el AUC indican que el modelo de XGBoost es superior, y por tanto, predice mejor que el bosque aleatorio.

6.4.3 Tiempo

Como ultima métrica se ha decidió medir el tiempo empleado por cada uno de los algoritmos cronometrando el tiempo que ha tardado cada uno en ser entrenado en segundos y predecir los resultados. Y el resultado ha sido el siguiente:

Tiempo del bosque aleatorio	Tiempo del XGBoost
1010,53 s.	753,84 s.

Tabla 6.7. Tiempo empleado por el Randon forest (izqa.) y XGBoost (dcha.). Fuente: Elaboración propia

Se observa que el algoritmo de XGBoost ha sido mucho más rápido, pero eso es debido en parte a que los parámetros óptimos del bosque aleatorio implicaban un mayor número de árboles de decisión, pero al fin y al cabo ha sido más rápido computacionalmente hablando.

Antes de observar el perfil del cliente que lleva a cabo venta cruzada entre Vida y Salud en la compañía ASESUISA es importante resaltar que los motivos por los que el XGBoost ha sido superior al bosque aleatorio.

Tal y como se vio en capítulos anteriores donde se explicaba la teoría que sustentaba ambos algoritmos, y se hizo una comparación teórica, el resultado empírico queda demostrado en línea con el explicado entonces. Se puede decir que ha quedado demostrado que el XGBoost es mejor debido a:

- Paralelización. Más rápido debido a su capacidad de paralelizar el proceso de boosting.
- Modelo aditivo. Al realizar un modelo aditivo donde un árbol se construye sobre el anterior permite ajustarse mejor a los datos y optimizar el resultado si se realizan las suficientes iteraciones. Además, es más rápido al realizar menos iteraciones.
- Método de construcción de árboles. Al empezar a crear los árboles seleccionando una hoja y después añadir iterativamente ramas al árbol evaluando las cualidades de cada división y eligiendo repetidamente las divisiones para cada nodo que minimizan la función de pérdida, se mejora el rendimiento computacional de forma significativa ya que la extensión de los árboles varia.
- Mejor distribución de pesos en cada hoja. Debido a la forma de construcción de los árboles de decisión, el XGBoost da más peso a las hojas más puras lo que le permite obtener mejores predicciones.

6.5 Perfil de cliente

Una vez realizado el análisis comparativo y observar que el XGBoost es superior en todas las métricas frente al bosque aleatorio a la hora de realizar la predicción debemos tomar los resultados obtenidos por este modelo para saber cuál es el perfil del potencial cliente que realice venta cruzada entre Vida individual y Salud de la compañía ASESUISA.

Para ello se debe obtener primero el grafico que nos muestra la importancia de las variables predictivas, y así poder establecer que características son las más determinantes a la hora de crear el perfil de este tipo de cliente. El gráfico obtenido es el siguiente:

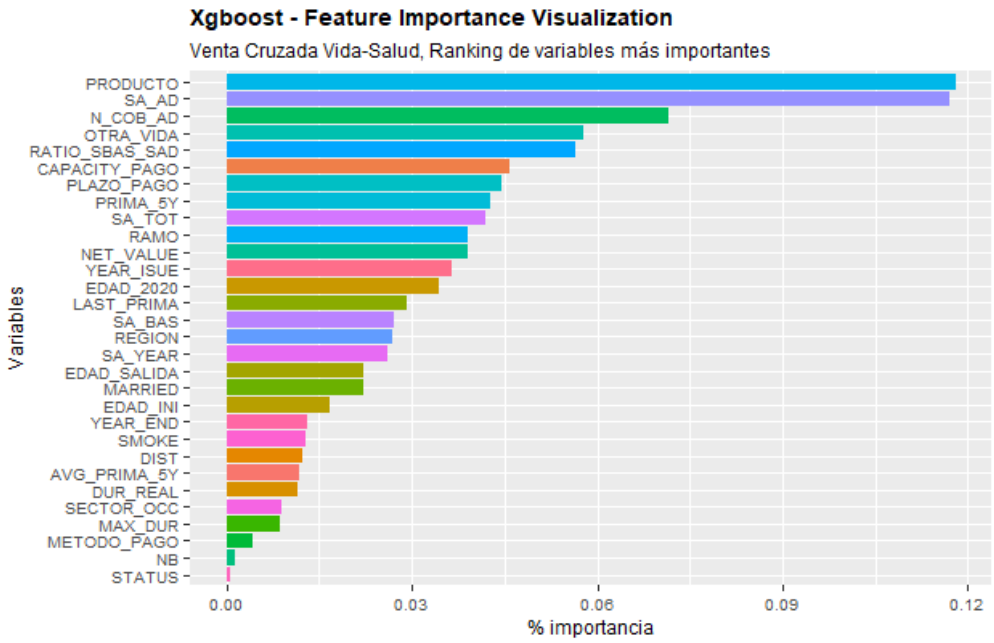

Figura 6.9. Distribución de las variables del modelo XGBoost. Fuente: Elaboración propia

En el grafico se puede observar, por orden de importancia, las variables más significativas. A continuación, se muestra un gráfico con la distribución de esas variables para el grupo de clientes que sí han realizado una venta cruzada. De las diferentes características se observa:

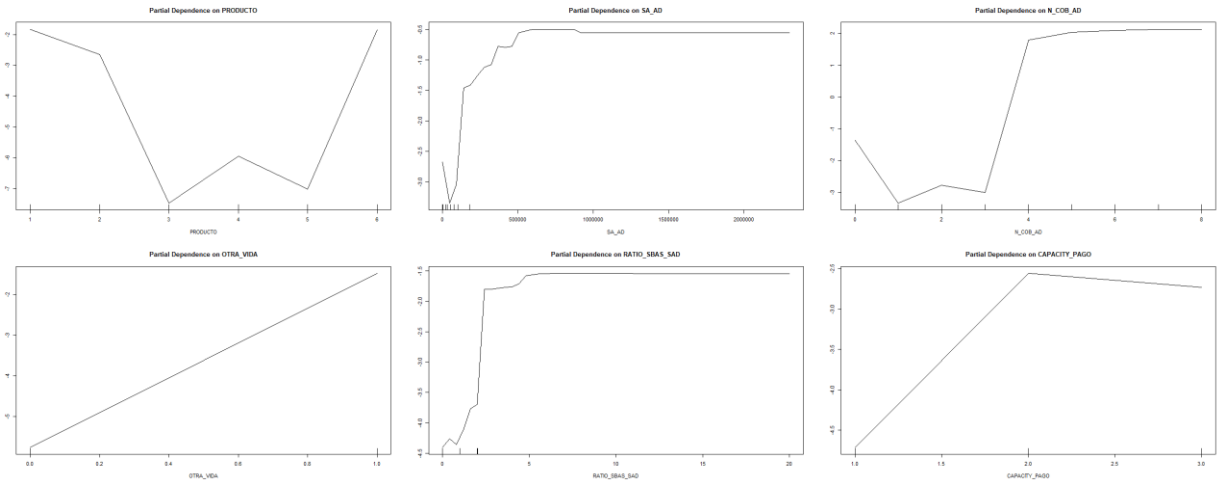

Figura 6.10. Importancia de las 6 variables más importantes del modelo XGBoost. Fuente: Elaboración propia

- Producto. Indica que tipo de producto de Vida ha comprado el cliente. Si se observa sólo el grupo que ha realizado venta cruzada se observa que los clientes que han realizado venta cruzada han comprado mayoritariamente los productos de Vida Múltiple (1), ASESUISA 2000 (2) y Vida Entera (6).
- Suma asegurada adicional y número de coberturas adicionales. Sorprendentemente esta conclusión se encuentra en línea con el resultado obtenido por Ernest Mack (1997) de que los clientes de venta cruzada tienen más coberturas y servicios, y es que al observar la distribución de las sumas aseguradas y el número de servicios para los individuos que han realizado venta cruzada es superior a 4 coberturas adicionales mayoritariamente y una suma asegurada adicional bastante alta.
- Producto adicional de Vida. Esta variable indicada si los clientes tienen más de un producto de Vida y va en línea con la conclusión anterior y la de Mack (1997). Se puede apreciar que los clientes que han realizado venta cruzada entre Vida y salud también tienen otro producto de Vida Individual
- Ratio de suma asegurada de cobertura principal frente a las adicionales. Esta variable fue una métrica desarrollada a partir de los datos que otorgó ASESUISA y es coherente con las métricas previas. En ella claramente se observa que la distribución para los individuos que sí han realizado venta cruzada es alta.
- Capacidad de pago. Esta variable también fue derivada a raíz de los datos que otorgó la compañía de seguros, e indica el percentil del porcentaje de pago realizado por el cliente respecto a la suma asegurada ponderado por el plazo máximo de pago. Se puede ver que la mayoría de los individuos dentro del grupo de personas que han realizado venta cruzada se encuentran en los grupos 2 y 3, que indica una alta capacidad de pago. Esta interpretación es coherente con la información otorgada por las variables previas ya que indica que indica que los clientes que han llevado a cabo una venta cruzada entre Vida y Salud tienen un alto poder adquisitivo.

De todas estas variables se obtiene una información muy valiosa para crear el perfil del cliente. De momento, se puede decir que los clientes que realizan una venta cruzada entre Vida individual y Salud son aquellos que tienen alguno de los productos de Vida múltiple, ASESUISA 2000 o Vida entera, con más de cuatro coberturas o algún otro producto de Vida, y con una alta capacidad de pago.

Otras variables que, aunque no sean tan determinantes son interesantes de estudiar a la hora de crear el perfil podrían ser:

- Edad actual. Dentro del conjunto de individuos que han realizado venta cruzada se puede observar como la gran mayoría se concentran en las edades de entre 30 y 40 años.

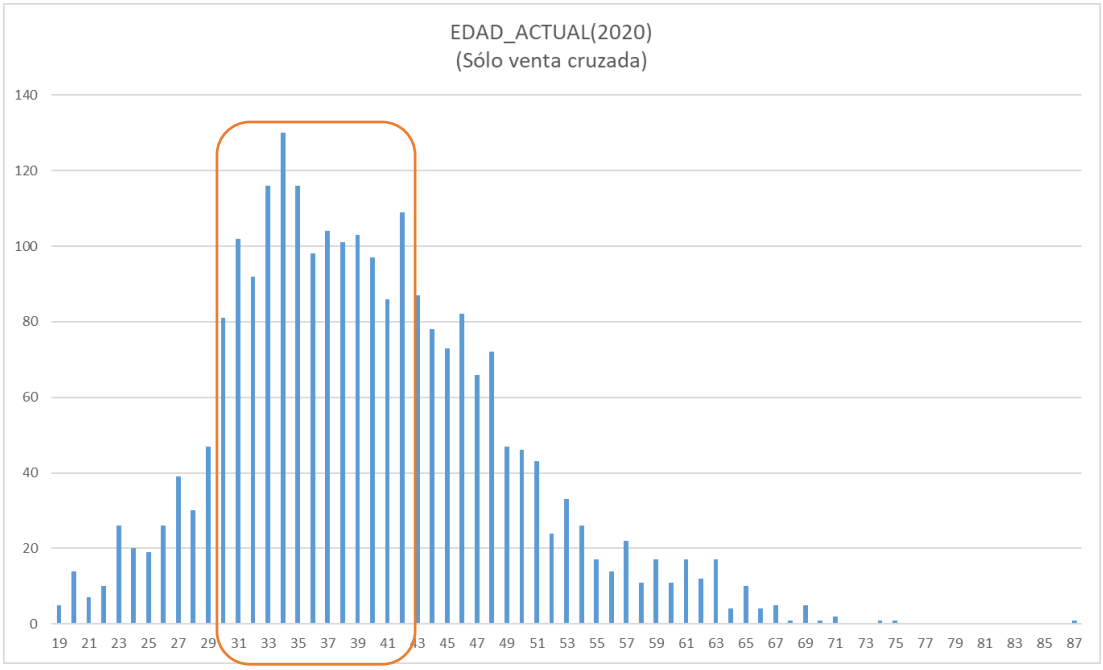

Figura 6.11. Distribución por edades de población que ha realizado venta cruzada. Fuente: Elaboración propia

- Sexo. Lógicamente como en el apartado del análisis del porfolio se observó, la probabilidad de que sea varón es mucho mayor.

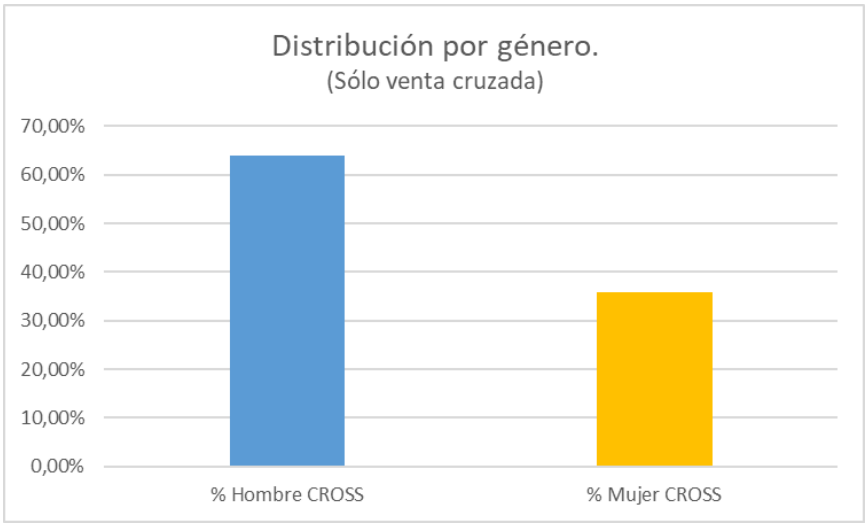

Figura 6.12. Distribución por género de población que ha realizado venta cruzada. Fuente: Elaboración propia

- Estado civil. Indica el estado civil del individuo. Se corresponde con la estadística real de El Salvador y el estudio de Harrison y Ansell (2002) que constataba que los clientes casados eran más propensos a adquirir un segundo producto.

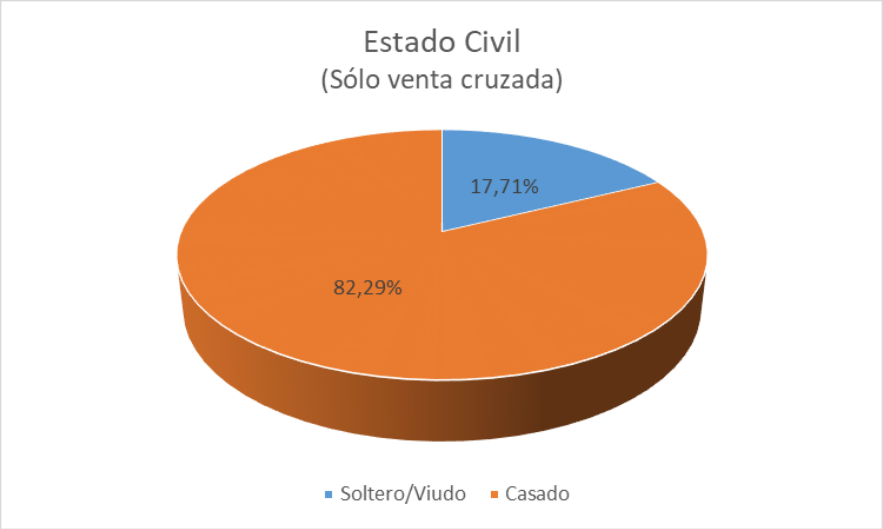

Figura 6.13. Porcentaje de casados de población que ha realizado venta cruzada. Fuente: Elaboración propia

- Región. Indica en qué zona vive el cliente y es útil para saber en qué lugares implementar más campañas de venta cruzada. Se observa que el resultado se corresponde con lo estudiado previamente y la distribución poblacional real de El Salvador.

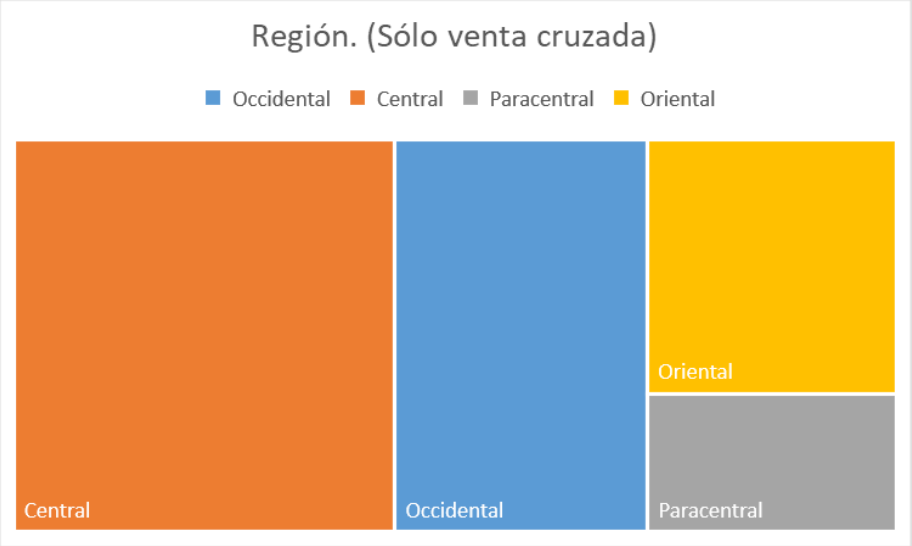

Figura 6.14. Distribución geográfica de población que ha realizado venta cruzada. Fuente: Elaboración propia

Una vez estudiadas las variables más importantes acorde a los datos, se puede determinar que el perfil del cliente potencial de venta cruzada entre Vida y Salud para la cartera de ASESUISA cumple con las siguientes características principales:

- Sexo: Varón.
- Edad: 30 – 40.
- Región: Central.
- Estado civil: Casado.
- Producto: Vida Múltiple, Vida Entera o ASESUISA 2000.
- Múltiples coberturas adicionales.
- Nivel socioeconómico: Alto.

El potencial cliente objetivo de una venta cruzada de ASESUISA cumple las características previas, y las conclusiones obtenidas se corresponde con las conclusiones de estudios previos realizados por otros autores como Harrison y Ansell (2002) y Peltier et al. (2002) que constataron que los clientes casados, con mayor poder adquisitivo y de mayor edad eran más propensos a comprar un segundo producto, y Ernest Mack (1997) que decía que los clientes de venta cruzada tienen más coberturas y servicios, aparte de tener una mayor retención.

7. CONCLUSIONES Y FUTURAS LÍNEAS DE INVESTIGACIÓN

En este estudio se ha analizado el concepto de venta cruzada y de los algoritmos predictivos aplicados a la misma, con el fin de realizar una comparativa empírica entre dos modelos, como son el bosque aleatorio y el XGBoost, para determinar cuál es mejor y obtener el perfil de un potencial cliente para el caso de venta cruzada entre productos de Vida Individual y Salud de la compañía ASESUIA-Seguros SURA de El Salvador y así optimizar la gestión de los recursos de la compañía.

En una primera instancia se ha determinado el concepto de venta cruzada como una técnica de venta empleada para maximizar el beneficio obtenido por cada consumidor vendiéndole un producto adicional relacionado con un producto previamente comprado con el fin de maximizar la utilidad del cliente y de la empresa. Y cómo éste es beneficial tanto para el cliente como para la empresa, ya que aumenta el ingreso promedio por cliente, maximiza la fidelización de éstos y disminuye, si se implementa correctamente, los costes logísticos. Adicionalmente se ha revisado la literatura académica de la venta cruzada aplicada al sector financiero y del seguro en la cual estudios diversos como los de Andy (2002), Harrison y Ansell (2002), Peltier et al. (2002) y Ernest Mack (1997) determinaban que los clientes más propensos eran aquellos de alto poder adquisitivo, pero también que el éxito o no de una campaña de venta cruzada dependía en gran medida de la habilidad de los empleados y de la organización de la propia empresa.

Posteriormente se estudió el concepto del aprendizaje automático y sus diferentes tipos, enfocándose especialmente en el aprendizaje supervisado, y más concretamente en los algoritmos del bosque aleatorio y del XGBoost. Después de revisar la literatura y ver cómo tanto el bosque aleatorio es respaldado académicamente por múltiples artículos (Caruana et al. (2008) y Fernández-Delgado et al (2014)), como el XGBoost es alabado por sus resultados empíricos en competiciones de la plataforma Kaggle (The Machine Learners, 2021) cabe destacar que ambos tienen cosas buenas y cosas malas, debido a sus diferentes planteamientos a la hora de predecir una variable (bagging y boosting), pero el XGBoost es superior a la hora de predecir, y así ha quedado demostrado en el caso de venta cruzada de ASESUISA tras evaluar ambos modelos con la matriz de confusión, la curva ROC y AUC y medir el tiempo que ha empleado cada uno.

Ello es debido a las características de este algoritmo que paraleliza el proceso de creación de árboles de decisión e implementación del boosting, que con menos iteraciones puede conseguir un mejor modelo. Además, el proceso de creación de los árboles seleccionando una hoja y después añadiendo iterativamente ramas al árbol evaluando las cualidades de cada división y eligiendo repetidamente las divisiones para cada nodo que minimizan la función de pérdida, se mejora el rendimiento computacional de forma significativa ya que la extensión de los árboles no es fija.

Una vez realizadas las predicciones y determinar qué modelo era mejor, se obtuvo el perfil del potencial cliente de venta cruzada para la cartera de ASESUISA entre Vida Individual y Salud que es lo que permitirá a la empresa identificar de forma más efectiva

a los clientes objetivos aumentando la ratio de conversión de ventas. Este potencial cliente es un varón, casado, de entre 30 y 40 años, que vive en la región Central de El Salvador, tiene un nivel socioeconómico alto y ha comprado uno o más de estos productos: Vida múltiple, Vida entera o ASESUISA 2000, con múltiples coberturas. Este resultado es acorde con la literatura académica sobre los estudios de venta cruzada previos realizados por otros autores. Con este perfil la empresa ASESUIA – Seguros SURA ahora es capaz de determinar que clientes de la cartera de Vida Individual son más propensos a comprar productos de Salud adicionalmente.

Como futuras líneas de investigación sería interesante poder aplicar esta misma técnica a la inversa. Es decir, sobre la cartera de Salud determinar que clientes compran productos de Vida individual. Así como también poder emplear otras técnicas de aprendizaje automático como los *Clúster* para poder segmentar diferentes tipos de clientes dentro de una misma cartera, y así poder crear productos específicos que se ajusten más a las necesidades de los mismos, o determinar sobre cuales se deberían realizar un *up-selling* (mejora del producto actual), o incluso implementar algún algoritmo de *Next-Product-to Buy* como el de Kamakura et al. (2005) para determinar cuándo se va a realizar la siguiente compra por parte de un cliente.

8. BIBLIOGRAFÍA

Agencia Estatal Boletín Oficial del Estado. (16 de 08 de 1889). Código Civil. *Código Civil*. Madrid, Madrid, España.

Ahlin, M., & Ranby, F. (2019). *Predicting Marketing Churn Using Machine Learning Models.* Suecia.

al, C. e. (2012). *Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC.* Suiza.

Andy Neely, Y. F. (2002). Cross-selling in the financial sector: Customer profitability is key. *Journal of Targeting, Measurement and Analysis for Marketing volume*, 282-296.

ASESUISA - Seguros SURA. (2020). *ASESUISA - seguros SURA*. Obtenido de Empresas SURA: https://www.asesuisa.com/empresas-sura

Banco Mundial. (2022). *Banco Mundial*. Obtenido de Datos - El Salvador: https://datos.bancomundial.org/pais/el-salvador

Berkeley School of Information. (26 de Junio de 2020). *Berkeley School of Information*. Obtenido de What Is Machine Learning?: https://ischoolonline.berkeley.edu/blog/what-is-machine-learning/

Bhandari, A. (16 de Junio de 2020). *Analytics Vidhya*. Obtenido de AUC-ROC Curve in Machine Learning Clearly Explained: https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/

Breiman, L. (2001). *Random Forest.* California.

Brownlee, J. (22 de Diciembre de 2019). *Machine Learning Mastery*. Obtenido de A Gentle Introduction to Cross-Entropy for Machine Learning: https://machinelearningmastery.com/cross-entropy-for-machine-learning/

Carrero, D. e. (7 de Marzo de 2022). *LA Network*. Obtenido de La mujer salvadoreña en el mercado laboral: más acceso y mejor calidad es posible: https://la.network/la-mujer-salvadorena-en-el-mercado-laboral-mas-acceso-y-mejor-calidad-es-posible/

Caruana, R. (2008). *An Empirical Evaluation of Supervised Learning in High Dimensions.* Helsinki. Obtenido de https://www.researchgate.net/publication/221346357_An_empirical_evaluation_of_supervised_learning_in_high_dimensions

Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System.*

Cohn, C. (Mayo de 2015). *Forbes*. Obtenido de A Beginner's Guide To Upselling And Cross-Selling: https://www.forbes.com/sites/chuckcohn/2015/05/15/a-beginners-guide-to-upselling-and-cross-selling/

Deloitte. (2020). *Doing Business El Salvador.*

Erich, S. W., & James, G. W. (1987). Cross-Selling: The Unfulfilled Promise. *Best's Review. Nº 88.*, 14 - 107.

Expansión. (23 de Junio de 2022). *Expasión*. Obtenido de Datosmacro.com: https://datosmacro.expansion.com/demografia/estructura-poblacion/el-salvador

Fernández-Delgado, M. (2014). *Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?* Santiago de Compostela.

Ferruci, D. A., Brown, E. W., Chu-Carroll, J., & Fan, J. (2010). *Building Watson: An Overview of the DeepQA Project.*

Gholamy, A., Kreinovich, V., & Kosheleva , O. (2018). *Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation.* Texas.

Guiltinam, J. P. (Abril de 1987). *The Price Bundling of Services: A Normative Framework.* Obtenido de Journal of Marketing: https://journals.sagepub.com/doi/10.1177/002224298705100206

Gupta, A. (2021). *Geek Culture*. Obtenido de XGBoost versus Random Forest: https://medium.com/geekculture/xGBoost-versus-random-forest-898e42870f30#:~:text=One%20of%20the%20most%20important%20differences%20between%20XG,more%20preferences%20to%20hyperparameters%20to%20optimize%20the%20model

Harrison, Tina, & Ansell, J. (2002). Customer Retention in the Insurance Industry: Using Survival Analysis to Predict Cross-Selling Opportunities. *Journal of Financial Servicies Marketing. Nº 6.*, 229 - 239.

Hastie, T., Tibshirani, R., & Friedman, J. (2008). *The Elements of Sattistical Learning: Data Mining, Inference and Prediction.* California: Springer.

Ho, T. K. (1995). *Random Decision Forest.* New Jersey.

IBM Cloud Education. (15 de Julio de 2020). *IBM*. Obtenido de Machine Learning: https://www.ibm.com/cloud/learn/machine-learning

Instituto Nacional de Estadística e Informática. (2018). *Población afiliada a algún seguro de salud.* Lima: INEI.

Jones, E., Chonko, L., Jones, F., & Stevens, C. (2012). *Selling ASAP: Art, Science, Agility, Performance.* Louisiana: Louisiana State University Press.

Kamakura, W. A. (2005). Choice Models and Customer Relationship Management. *Marketing Letters*, 279-291.

Kamakura, W. A., Wedel, M., de Rosa, F., & Jose, M. A. (2003). Cross-selling through database marketing: a mixed data factor analyzer for data augmentation and prediction. *International Journal of Research in Marketing*, 45-65.

Karabiber, F. (2022). *Learn Data Sci*. Obtenido de Gini Impurity: https://www.learndatasci.com/glossary/gini-impurity/#:~:text=More%20precisely%2C%20the%20Gini%20Impurity,class%20distribution%20in%20the%20dataset.

Kumar, V., & Shah, D. (Diciembre de 2012). *The Dark Side of Cross-Selling*. Obtenido de Harvard Business Review: https://hbr.org/2012/12/the-dark-side-of-cross-selling

Levine, L. (1996). Why Cross-Selling and Upselling Seem So Difficult to Implement. *Telemarketing and Call Center Solutions*, 122 - 125.

Lymberopoulos, Konstantinos, Chaniotakis, I. E., & Soureli, M. (2004). Opportunities for Banks to Cross-Sell Insurance Products in Greece. *Journal of Financial Services Marketig*, 34 - 48.

Mack, E. (1997). Cross-Selling by Any Name Makes Good Business Sense. *Best's Review*.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Berkeley Symposium on Mathematical Statistics and Probability*, 281 - 297.

Matlab. (2022). *Matlab*. Obtenido de Support Vector Machine (SVM): https://es.mathworks.com/discovery/support-vector-machine.html

Narkhede, S. (9 de Mayo de 2018). *Towards Data Science*. Obtenido de Understanding Confusion Matrix: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

Neslin, S. A., Knott, Aaron, & Hayes, A. (2002). Marketplaxe: Next-Product-To-Buy Models for Cross-Selling. *Journal of Interactive Marketing*, 59 - 75.

OpenAI. (25 de Marzo de 2021). *OpenAI*. Obtenido de GPT-3 Powers the Next Generation of Apps: https://openai.com/blog/gpt-3-apps/

Parnell, C., & Anderson, M. (2022). *The Parker Avery Group*. Obtenido de Retail Clustering Methods: Achieving Success with Assortment Planning: https://parkeravery.com/industry-experience/retail-clustering-methods/

Peltier, J. W., Schibrowsky, J. A., Schultz, D. E., & Davis, J. (2002). Interactive Psychographics: Cross-Selling in the Banking Industry. *Journal of Adverstising Research*, 7 - 22.

Polonsky, M. J., Cameron, H., Halstead, S., Ratcliffe, A., Stilo, P., & al, e. (2000). Exploring companion selling: does the situation affect customers' perceptions? *International Journal of Retail & Distribution Management. Nº 28.*

Qi, Y. (2012). *Random Forest for Bioinformatics.*

RAE. (2021). *Diccionario Real Academia Española*. Obtenido de Diccionario Real Academia Española: https://dle.rae.es/venta

RDocumentation. (2022). *RDocumentation*. Obtenido de randomForest: Classification and Regression with Random Forest: https://www.rdocumentation.org/packages/randomForest/versions/4.7-1.1/topics/randomForest

Salesforce. (2022). *¿Qué es un Cross Selling?* Obtenido de Salesforce.com: https://www.salesforce.com/es/learning-centre/sales/cross-selling/

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers.

Sheppard, D. (1999). *The New Direct Marketing: How to Implement A Profit-Driven Database Marketing Strategy.* McGraw-Hill.

Szufel, P., Kamiński, B., & Jakubczyk, M. (2017). A framework for sensitivity analysis of decision trees. *Central European Journal of Operations Research*.

The Machine Learners. (12 de 2021). *The Machine Learners*. Obtenido de XGBoost, el algoritmo que lo peta en las competiciones de Kaggle: https://www.themachinelearners.com/xGBoost-python/

The Machine Learners. (2022). *The Machine Learners*. Obtenido de Encuentra tu modelo perfecto: https://www.themachinelearners.com/busqueda-hiperparametros/

Werbos, P. (1974). Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. *National Science Foundation*.

Williams, R. J. (1992). *Simple statistical gradient-following algorithms for connectionist reinforcement learning.* Springer.

XGBoost Developers. (2021). *dmlc XGBoost*. Obtenido de XGBoost Documentation: https://xGBoost.readthedocs.io/en/latest/

9. ANEXOS

Anexo – 1: Código de obtención y análisis previo de datos.

```
1.  ########################
2.  ### 0. CARGAR BIBLIOTECAS
3.  ########################
4.  library(ISLR)
5.  library(MASS)
6.  library(dplyr)
7.  library(tidyr)
8.  library(skimr)
9.  library(ggplot2)
10. library(ggpubr)
11. library(tidymodels)
12. library(pROC)
13. library(tidyverse)
14. library(ranger)
15. library(doParallel)
16. library(corrplot)
17. library(randomForest)
18. library(xgboost)
19. library(tidymodels)
20. library(gbm)
21. library(themis)
22. library(caret)
23. library(randomForest)
24. library(ROCR)
25. library(pROC)
26. library(pdp)
27.
28.
29. ##############################################################################
30.
31. ########################
32. ### 1. CARGAR DATA-SET
33. ########################
34.
35. DATA<-read.csv2(file.choose()) # Seleccionar BD_ASESUISA
36. DATA<-data.table(DATA)
37.
38. head(DATA)
39.
40. NAMES <-
    c("ID","SEX","SMOKE","MARRIED","DIST","EDAD_2020","CAPACITY_PAGO","NET_V
    ALUE","REGION","SECTOR_OCC",
41.           "PRIMA_5Y","AVG_PRIMA_5Y","LAST_PRIMA","PRODUCTO","EDAD_INI","
    EDAD_SALIDA","NB","RAMO","PLAZO_PAGO","METODO_PAGO",
42.           "SA_TOT","SA_YEAR","SA_BAS","COB_PRINCIPAL","SA_AD","N_COB_AD"
    ,"RATIO_SBAS_SAD","OTRA_VIDA","YEAR_ISUE","YEAR_END",
43.           "DUR_REAL","STATUS","SINIESTRO","MAX_DUR","SALUD")
44.
45. # transformacion de variables
46. colnames(DATA) <-NAMES
47. head(DATA)[1]
48. str(DATA)
49. DATA$SA_YEAR <-as.numeric(DATA$SA_YEAR)
50. DATA$SALUD <-as.factor(DATA$SALUD)
51. DATA <- na.omit(DATA)
52. DATA$ID = NULL
53. summary(DATA)
54.
55. ########################
56. ### 2. ANALIIS PREVIO
57. ########################
```

```
58.
59. # % de solapamiento
60. plot(DATA$SALUD)
61. table(DATA$SALUD)/nrow(DATA) # 20,5% de cobertura
62.
63. # división del dataset en 2 partes de training y Validación - 80/20
64. sample.ind <- sample(2,nrow(DATA), replace = T, prob = c(0.8, 0.2))
65. DATA_TRAIN <- DATA[sample.ind==1,]
66. DATA_VAL <- DATA[sample.ind==2,]
67. table(DATA_TRAIN$SALUD)/nrow(DATA_TRAIN) # 20% aprox de cobertura en
    ambos casos
68. table(DATA_VAL$SALUD)/nrow(DATA_VAL) # 20% aprox de cobertura en ambos
    casos
69.
70. skim(DATA)
71.
72. # COEF CORRELACION
73. DATA2 <- DATA
74. DATA2$SALUD<-as.numeric(DATA2$SALUD)
75. DATA2$ID <- NULL # elimino porque no es numerica
76. DATA2$COB_PRINCIPAL <- NULL # elimino porque es cte
77. str(DATA2)
78.
79. CM<-round(cor(DATA2), 1)
80.       corrplot(CM,method="shade",shade.col=NA,tl.col="black",
    tl.srt=45,adCoef.col="black", order = "AOE")
```

Anexo – 2: Código del modelo Random Forest.

```
#########################
### 3. RANDOM FOREST
#########################
set.seed(2021)

t<-proc.time() #### CRONOMETRO EMPIEZA

## 3.1 DEFINICIÓN DEL MODELO Y DE LOS HIPERPARÁMETROS A OPTIMIZAR
modelo <- rand_forest(
  mode  = "classification",
  mtry  = tune(),
  trees = tune()
) %>%
  set_engine(
    engine     = "ranger",
    max.depth  = tune(),
    importance = "none",
    seed       = 2021
  )

transformer <- recipe(
  formula = SALUD ~ .,
  data    =  DATA_TRAIN
)

## 3.2 CREACIÓN DE PARTICIONES
cv_folds <- vfold_cv(
  data    = DATA_TRAIN,
  v       = 5,
  strata  = SALUD
)

## 3.3 WORKFLOW y GRID DE HIPERPARÁMETROS y ejecución
workflow_modelado <- workflow() %>%
  add_recipe(transformer) %>%
  add_model(modelo)

grid_param <- expand_grid(
  'trees'     = c(100, 300,500,1000,3000),
  'mtry'      = c(3, 5, 7, 10, 15, ncol(DATA_TRAIN)-1),
  'max.depth' = c(3,5,7,10,20)
)

cl <- makePSOCKcluster(parallel::detectCores() - 1)
registerDoParallel(cl)

grid_ajuste <- tune_grid(
  object    = workflow_modelado,
  resamples = cv_folds,
  metrics   = metric_set(accuracy),
  grid      = grid_param
)
stopCluster(cl)

# Mejores parámetros por validación cruzada - optimizar accuracy
show_best(grid_ajuste, metric = "accuracy", n = 1)

## 3.4 ENTRENAMIENTO FINAL - optimizar precisión
mejores_hiperpar <- select_best(grid_ajuste, metric = "accuracy")

modelo_final_fit <- finalize_workflow(
  x = workflow_modelado,
  parameters = mejores_hiperpar
) %>%
  fit(
```

```
    data = DATA_TRAIN
  ) %>%
  pull_workflow_fit()

# Predicción Final  - Cross validation final
PRED <- modelo_final_fit %>%
  predict(new_data = DATA_VAL)

PRED <- PRED %>%
  bind_cols(DATA_VAL %>% dplyr::select(SALUD))

accuracy_test  <- accuracy(
  data     = PRED,
  truth    = SALUD,
  estimate = .pred_class,
  na_rm    = TRUE
)
accuracy_test
m_confusion <- PRED %>%
  conf_mat(
    truth     = SALUD,
    estimate  = .pred_class
  )

m_confusion  ## MATRIZ DE CONFUSIÓN - RF

proc.time() - t #### CRONOMETRO TEERMINA

# GRAFICO MATRIZ CONFUSIÓN
lvs <- c("No", "Sí")
truth <- factor(rep(lvs, times = c(1863, 477)),
                levels = rev(lvs))
predi <- factor(
  c(
          rep(lvs, times = c(1815,48)),
          rep(lvs, times = c(129, 348))),
        levels = rev(lvs))

      confusionMatrix(predi, truth)

      table <- data.frame(confusionMatrix(predi, truth)$table)

      plotTable <- table %>%
        mutate(goodbad = ifelse(table$Prediction == table$Reference, "g
   ood", "bad")) %>%
        group_by(Reference) %>%
        mutate(prop = Freq/sum(Freq))

      ggplot(data = plotTable, mapping = aes(x = Reference,
   y = Prediction, fill = goodbad, alpha = Freq)) +
        geom_tile() +
        geom_text(aes(label = Freq), vjust = .5, fontface  = "bold",
   alpha = 1) +
        scale_fill_manual(values = c(good = "green", bad = "red")) +
        theme_bw() +
        xlim(rev(levels(table$Reference)))

      ## 3.5 AUC & ROC
      PRED2<- as.numeric(PRED$.pred_class, type ="SALUD")
      PRED3<-prediction(PRED2,DATA_VAL$SALUD)
      PERF<- performance(PRED3, measure = "tpr",x.measure = "fpr")
      plot(PERF,col=rainbow((14)))
      # AUC
      AUC<- performance(PRED3,"auc")
      str(AUC)
```

Anexo – 3: Código del modelo XGBoost.

```
#########################
### 4. XGBOOSTING  ###
#########################

set.seed(2021)

proc.time() #### CRONOMETRO EMPIEZA

#Creo matriz/lista con los datos
XDATA <- map_df(DATA, function(columna) {
  columna %>% as.factor() %>% as.numeric %>% {.-1}})
head(XDATA)
LISTA <- list()
LISTA$XDATATRAIN <- sample_frac(XDATA,size = 0.8)
LISTA$XDATAVAL <- sample_frac(XDATA,size = 1)
LISTA$XDATAVAL <- setdiff(LISTA$XDATAVAL,LISTA$XDATATRAIN)

LISTA$XTRAIN <-LISTA$XDATATRAIN %>% dplyr::select(-
SALUD) %>% as.matrix() %>%
  xgb.DMatrix(data=., label = LISTA$XDATATRAIN$SALUD)
LISTA$XTRAIN

LISTA$XVAL <-LISTA$XDATAVAL %>% dplyr::select(-
SALUD) %>% as.matrix() %>%
  xgb.DMatrix(data=., label = LISTA$XDATAVAL$SALUD)
LISTA$XVAL

# MODELO
modelo <- xgb.train(
  data    = LISTA$XTRAIN,
  params  = list(max_depth = 2),
  nrounds = 10,
)
modelo

# Error de test del modelo previo
predicciones <- predict(
  modelo,
  newdata = LISTA$XVAL
)

test_rmse <- sqrt(mean((predicciones - getinfo(LISTA$XVAL, "label"))^2))
paste("Error de test (rmse) del modelo:", round(test_rmse,2))

# oprimizacion de parametros

# Validación empleando k-cross-validation (root mean squared error)

resultados_cv <- xgb.cv(
  data      = LISTA$XTRAIN,
  params    = list(eta = 0.3, max_depth = 6, subsample = 1),
  nrounds   = 500,
  nfold     = 5,
  metrics   = "rmse",
  verbose   = 0
)
resultados_cv <- resultados_cv$evaluation_log
# Gráfico con la evolución de los errores
ggplot(data = resultados_cv) +
  geom_line(aes(x = iter, y = train_rmse_mean, color = "train rmse")) +
  geom_line(aes(x = iter, y = test_rmse_mean, color = "cv rmse")) +
  geom_point(
    data = slice_min(resultados_cv, order_by = test_rmse_mean, n = 1),
```

```
    aes(x = iter, y = test_rmse_mean),
    color = "firebrick"
  ) +
  labs(
    title = "Evolución del cv-error vs número árboles",
    x     = "número de árboles",
    y     = "cv-error (rmse)",
    color = ""
  ) +
  theme_bw() +
  theme(legend.position = "bottom")
paste("Valor óptimo de nrounds:", slice_min(resultados_cv,
order_by = test_rmse_mean, n = 1)$iter)

# learning rate ##
eta_range          <- c(0.001, 0.01, 0.1, 0.3)
df_resultados_cv   <- data.frame()

for(i in seq_along(eta_range)){
   resultados_cv <- xgb.cv(
    data    = LISTA$XTRAIN,
    params  = list(eta = eta_range[i], max_depth = 6, subsample = 1),
    nrounds = 4000,
    nfold   = 5,
    metrics = "rmse",
    verbose = 0
  )
    resultados_cv <- resultados_cv$evaluation_log
  resultados_cv <- resultados_cv %>%
    dplyr::select(iter, test_rmse_mean) %>%
    mutate(eta = as.character(eta_range[i]))

  df_resultados_cv <- df_resultados_cv %>% bind_rows(resultados_cv)
}

# Gráfico con la evolución de los errores
ggplot(data = df_resultados_cv) +
  geom_line(aes(x = iter, y = test_rmse_mean, color = eta)) +
  labs(
          title = "Evolución del cv-error vs learning rate (eta)",
          x     = "eta",
          y     = "cv-error (rmse)",
          color = ""
        ) +
        theme_bw() +
        theme(legend.position = "bottom")

      ## GRID SEARCH ##
      # DEFINICIÓN DEL MODELO Y DE LOS HIPERPARÁMETROS A OPTIMIZAR
      modelo <- boost_tree(
        mode         = "classification",
        mtry         = tune(),
        trees        = tune(),
        tree_depth   = tune(),
        learn_rate   = tune(),
        sample_size = tune(),
        stop_iter    = NULL
      ) %>%
        set_engine(
          engine     = "xgboost")

      # DEFINICIÓN DEL PREPROCESADO
      # En este caso no hay preprocesado, por lo que el transformer
solo contine
      # la definición de la formula y los datos de entrenamiento.
      transformer <- recipe(
```

```
126.        formula =  SALUD ~ .,
127.        data    =  DATA_TRAIN
128.      )
129.
130.      # DEFINICIÓN DE LA ESTRATEGIA DE VALIDACIÓN Y CREACIÓN DE
    PARTICIONES
131.      cv_folds <- vfold_cv(
132.        data    = DATA_TRAIN,
133.        v       = 5,
134.        strata  = SALUD
135.      )
136.
137.      # WORKFLOW
138.      workflow_modelado <- workflow() %>%
139.        add_recipe(transformer) %>%
140.        add_model(modelo)
141.
142.      # GRID DE HIPERPARÁMETROS
143.      grid_param <- expand_grid(
144.        "trees"      = c(100, 250, 500),
145.        "mtry"       = c(3,7,10,15,ceiling(sqrt(ncol(DATA_TRAIN))), nco
    l(DATA_TRAIN)-1),
146.        "tree_depth" = c(3, 5,7,10, 13),
147.        "learn_rate" = c(0.01, 0.1, 0.3),
148.        "sample_size" = c(0.5, 0.75)
149.      )
150.
151.      # EJECUCIÓN DE LA OPTIMIZACIÓN DE HIPERPARÁMETROS
152.      cl <- makePSOCKcluster(parallel::detectCores() - 1)
153.      registerDoParallel(cl)
154.
155.      grid_ajuste <- tune_grid(
156.        object    = workflow_modelado,
157.        resamples = cv_folds,
158.        metrics   = metric_set(accuracy),
159.        grid      = grid_param
160.      )
161.      stopCluster(cl)
162.
163.      # Mejores hiperparámetros por validación cruzada
164.      show_best(grid_ajuste, metric = "accuracy", n = 1)
165.
166.
167.      # MEJOR ITERACION del modelo
168.      param_list <- list(eta = 0.1,
169.                         max_depth = 5,
170.                         min_child_weight = 1,
171.                         subsample = 0.8,
172.                         colsample_bytree = 0.8,
173.                         objective = "binary:logistic")
174.      metric_list <- list("auc")
175.
176.      # Aplicar cross validation para encontrar mejor iteración
177.      xgb_cv <- xgb.cv(params = param_list,
178.                       data = LISTA$XTRAIN,
179.                       nrounds = 500,
180.                       nfold = 10,
181.                       label = as.matrix(LISTA$XDATATRAIN$SALUD),
182.                       verbose = 1,
183.                       print_every_n = 50,
184.                       metrics = metric_list)
185.
186.      #MEJOR ITERACIÓN PLOT.
187.      best_iter_train <- min(xgb_cv$evaluation_log[(xgb_cv$evaluation_l
    og)$test_auc_mean == max((xgb_cv$evaluation_log)$test_auc_mean), ]$iter)
188.      paste0("Iteration with highest Test AUC = ", best_iter_train)
```

```
189.
190.      # ROC y AUC de la mejor iteracion
191.      auc_log_train <- rbind(cbind("type" = rep("AUC
   Train", nrow(xgb_cv$evaluation_log)),
192.                                   xgb_cv$evaluation_log[, c("iter", "t
   rain_auc_mean")]),
193.                             cbind("type" = rep("test
   auc", nrow(xgb_cv$evaluation_log)),
194.                                   xgb_cv$evaluation_log[, c("iter", "t
   est_auc_mean")]),
195.                             use.names = FALSE) %>%
196.        setNames(c("auc_type", "iter", "auc_value"))
197.
198.      p_plot_auc_train <- ggplot(data = auc_log_train) +
199.        geom_point(mapping = aes(x = iter,
200.                                 y = auc_value,
201.                                 color = auc_type),
202.                   show.legend = TRUE, size = 1) +
203.        scale_color_manual(values = c("red", "blue")) +
204.        theme(text = element_text(size = 10),
205.              plot.title = element_text(face = "bold")) +
206.        labs(x = "Iterations",
207.             y = "Mean Test AUC",
208.             title = "Xgboost - Test AUC Variations (Training Data)",
209.             subtitle = paste0("Test AUC is highest at Iteration ",
210.                               best_iter_train, "\nTraining AUC keeps
   on increasing"))
211.
212.      p_plot_auc_train
213.
214.      LISTA$MODELO_1 <-xgboost(data =LISTA$XTRAIN,
   label = as.matrix(LISTA$XDATATRAIN$SALUD), params = param_list,
215.                               nrounds=best_iter_train,print_every_n =
   15 , eval_metric = "auc")
216.      LISTA$MODELO_1
217.
218.      proc.time() - t ## Fin del cronómetro
219.
220.      # importancia de cada variable - XGBoost
221.      xgb_model_imp_train <- xgb.importance(model = LISTA$MODELO_1)
222.
223.      p_feature_importances_train <- ggplot(data = xgb_model_imp_train)
    +
224.        geom_col(mapping = aes(x = Gain,
225.                               y = reorder(Feature, Gain),
226.                               fill = Feature),
227.                 show.legend = FALSE) +
228.        theme(text = element_text(size = 9),
229.              plot.title = element_text(face = "bold")) +
230.        labs(x = "% importancia", y = "Variables",
231.             title = "Xgboost - Feature Importance Visualization
   (Training Data)",
232.             subtitle = paste0("Venta Cruzada Vida-Salud, Top 10
   variables más importantes en el modelo"))
233.
234.      p_feature_importances_train
235.
236.      # EVALUACIN DEL MODELO
237.      # generacion de predicciones
238.      LISTA$PREDICT_01 <- predict (LISTA$MODELO_1, LISTA$XVAL)
239.      head(LISTA$PREDICT_01)
240.      head(LISTA$PREDICT_01>0.5)
241.      paste0("Success Rate =
   ", round(mean(ifelse(LISTA$PREDICT_01 > 0.5, 1, 0) == LISTA$XDATAVAL$SAL
   UD), digits = 4) * 100, "%")
242.
```

```
243.      # matriz de confusion
244.      cbind(LISTA$PREDICT_01>0.5,
   LISTA$XDATAVAL$SALUD) %>% data.frame() %>% table() %>% caret::confusionM
   atrix()
245.
246.      lvs <- c("No", "Sí")
247.      truth <- factor(rep(lvs, times = c(1864, 484)),
248.                      levels = rev(lvs))
249.      predi <- factor(
250.        c(
251.          rep(lvs, times = c(1827,37)),
252.          rep(lvs, times = c(89, 395))),
253.        levels = rev(lvs))
254.
255.      confusionMatrix(predi, truth)
256.
257.      table <- data.frame(confusionMatrix(predi, truth)$table)
258.
259.      plotTable <- table %>%
260.        mutate(goodbad = ifelse(table$Prediction == table$Reference, "g
   ood", "bad")) %>%
261.        group_by(Reference) %>%
262.        mutate(prop = Freq/sum(Freq))
263.
264.      ggplot(data = plotTable, mapping = aes(x = Reference,
   y = Prediction, fill = goodbad, alpha = Freq)) +
265.        geom_tile() +
266.        geom_text(aes(label = Freq), vjust = .5, fontface  = "bold",
   alpha = 1) +
267.        scale_fill_manual(values = c(good = "green", bad = "red")) +
268.        theme_bw() +
269.        xlim(rev(levels(table$Reference)))
270.
271.      table("preds" = ifelse(LISTA$PREDICT_01 > 0.5, 1, 0),
272.            "actuals" = LISTA$XDATAVAL$SALUD)
273.      cbind("preds" = ifelse(LISTA$PREDICT_01 > 0.5, 1, 0),
274.            "actuals" = LISTA$XDATAVAL$SALUD
275.      ) %>% as_tibble() %>%
276.        group_by(preds, actuals) %>%
277.        summarise(cnt = n(), .groups = "drop_last")
278.
279.      # EL ROC
280.      plot(pROC::roc(response = LISTA$XDATAVAL$SALUD,
281.                     predictor = LISTA$PREDICT_01,
282.                     levels=c(0, 1)), lwd=1.5)
283.
284.      training_data <- model.matrix(object = SALUD ~ .,
285.                                    data = DATA_TRAIN[, -1])[, -1]
286.
287.      param_list <- list(eta = 0.1,
288.                         max_depth = 5,
289.                         min_child_weight = 1,
290.                         subsample = 0.8,
291.                         colsample_bytree = 0.8,
292.                         objective = "binary:logistic")
293.      metric_list <- list("auc")
294.
295.      # Aplicar cross validation para encontrar mejor iteración
296.      xgb_cv_1 <- xgb.cv(params = param_list,
297.                         data = training_data,
298.                         nrounds = 500,
299.                         nfold = 10,
300.                         label = as.matrix(DATA_TRAIN$SALUD),
301.                         verbose = 1,
302.                         print_every_n = 50,
303.                         metrics = metric_list)
```

```
304.
305.      # Mejor iteración
306.      best_iter <- min(xgb_cv_1$evaluation_log[(xgb_cv_1$evaluation_log
   )$test_auc_mean == max((xgb_cv_1$evaluation_log)$test_auc_mean), ]$iter)
307.      paste0("Iteration with highest Test AUC = ", best_iter)
308.
309.      auc_log <- rbind(cbind("type" = rep("train
   auc", nrow(xgb_cv_1$evaluation_log)),
310.                             xgb_cv_1$evaluation_log[, c("iter", "train
   _auc_mean")]),
311.                       cbind("type" = rep("test
   auc", nrow(xgb_cv_1$evaluation_log)),
312.                             xgb_cv_1$evaluation_log[, c("iter", "test_
   auc_mean")]),
313.                       use.names = FALSE) %>%
314.        setNames(c("auc_type", "iter", "auc_value"))
315.
316.      p_plot_auc <- ggplot(data = auc_log) +
317.        geom_point(mapping = aes(x = iter,
318.                                 y = auc_value,
319.                                 color = auc_type),
320.                   show.legend = TRUE, size = 1) +
321.        scale_color_manual(values = c("red", "blue")) +
322.        theme(text = element_text(size = 10),
323.              plot.title = element_text(face = "bold")) +
324.        labs(x = "Iterations",
325.             y = "Mean Test AUC",
326.             title = "Xgboost - AUC Variations",
327.             subtitle = paste0("Test AUC is highest at Iteration ",
328.                               best_iter, "\nTraining AUC keeps on
   increasing"))
329.
330.      xgboost_model_final <- xgboost(data = training_data,
331.                                     label = as.matrix(DATA_TRAIN$SALUD
   ),
332.                                     params = param_list,
333.                                     nrounds = best_iter,
334.                                     print_every_n = 25,
335.                                     eval_metric = "auc")
336.
337.      # IMportancia de cad variable
338.      xgb_model_imp <- xgb.importance(model = xgboost_model_final)
339.
340.      p_feature_importances <- ggplot(data = xgb_model_imp) +
341.        geom_col(mapping = aes(x = Gain,
342.                               y = reorder(Feature, Gain),
343.                               fill = Feature),
344.                 show.legend = FALSE) +
345.        theme(text = element_text(size = 9),
346.              plot.title = element_text(face = "bold")) +
347.        labs(x = "% importancia", y = "Variables",
348.             title = "Xgboost - Feature Importance Visualization",
349.             subtitle = paste0("Venta Cruzada Vida-Salud, Ranking de
   variables más importantes"))
350.
351.      p_feature_importances
352.
353.      test_data_final <- model.matrix(object =  SALUD ~ .,
354.                                      data = DATA_VAL[, -1])[, -1]
355.
356.      # Predcciones
357.      pred_xgboost_final <- predict(object = xgboost_model_final,
358.                                    newdata = test_data_final)
359.
360.      # Predcciones del modelo
```

```
361.      pred_df <- as.data.frame(ifelse(pred_xgboost_final > 0.50, 1, 0))
    %>%
362.        setNames("SALUD")
363.
364.      pred_df %>% count(SALUD, sort = TRUE)
365.
366.      test_df_wresponse <- cbind(DATA_VAL[, 1], pred_df)
367.
368.      head(test_df_wresponse)
369.
370.      test_df_wresponse %>%
371.        count(SALUD, sort = TRUE)
372.
373.      # distribucion parcial de var importantes
374.      par(mfrow = c(2,3))
375.      partialPlot(xgboost_model_final,DATA,PRODUCTO,1)
376.      partialPlot(xgboost_model_final,DATA,SA_AD,1)
377.      partialPlot(xgboost_model_final,DATA,N_COB_AD,1)
378.      partialPlot(xgboost_model_final,DATA,OTRA_VIDA,1)
379.      partialPlot(xgboost_model_final,DATA,RATIO_SBAS_SAD,1)
380.      partialPlot(xgboost_model_final,DATA,CAPACITY_PAGO,1)
```