UNIVERSIDAD
TORCUATO DI TELLA

Master in Management + Analytics Tesis
Escuela de Negocios

Clasificación de Envíos mediante Procesamiento de Texto

Realizado por: ***Damián Izarnotegui***

Tutor: ***Pablo Roccatagliata***

Año: ***2021***

Resumen

El proceso logístico es uno de los componentes más importantes en la propuesta de valor que una determinada empresa de comercio electrónico puede ofrecerle al cliente. Al mismo tiempo, los costos operativos se pueden disparar si no es gestionada de manera eficiente. Entre sus desafíos se encuentra la identificación de ciertas características de los productos que pueden modificar la forma de gestionar los paquetes a través del proceso de envío. Si el paquete se gestiona de manera incorrecta, se genera la posibilidad de incurrir en costos adicionales, además de afectar la experiencia del usuario. Todo lo mencionado adquiere mayor criticidad si el contexto se da de manera electrónica y de forma masiva.
El presente trabajo argumenta que, haciendo uso de técnicas de aprendizaje automático, puede sacarse provecho del enorme volumen de datos disponible en los sistemas de información de una de las empresas de comercio electrónico más importante de Latinoamérica para construir soluciones que permitan mejorar el proceso de la toma de decisiones en un determinado punto del proceso adquisición y envío de un producto.
Con el objetivo de poner a prueba esta hipótesis, se tuvo a disposición una gran cantidad de datos de productos que atravesaron el proceso logístico antes mencionado. El foco del trabajo estuvo puesto en la construcción de un modelo de aprendizaje automático, específicamente utilizando técnicas de modelado del lenguaje, que pudiera predecir la probabilidad de que producto pudiese ser identificado con una etiqueta que está asociada a ciertas características físicas del producto y a partir de los diferentes descriptores textuales que conforman la identificación de los productos publicados por los vendedores.

Como resultado, se obtuvo una serie de modelos con rendimientos prometedores que presentan un ahorro mayor al 95% respecto al mayor ahorro que se podría alcanzar sobre un conjunto de datos de testeo. Además, presentan una muy buena calidad predictiva logrando separar las clases de los productos maquinables y no maquinables, prediciendo correctamente más del 95% de las etiquetas de los datos no vistos y utilizados para la evaluación.
.

Abstract

The logistics process is one of the most important components in the value proposition that a certain e-commerce company can offer to the customer. At the same time, operating costs can skyrocket if not managed efficiently. Its challenges include identifying certain product characteristics that can change the way packages are managed through the shipping process. If the package is managed incorrectly, it creates the possibility of incurring additional costs, in addition to affecting the user experience. All the aforementioned becomes more critical if the context is given electronically and in a massive way.

The present work argues that, by making use of machine learning techniques, it is possible to take advantage of the enormous volume of data available in the information systems of one of the most important electronic commerce companies in Latin America to build solutions that allow improving the process of the decision making at a certain point in the process of purchasing and shipping product.

In order to test this hypothesis, a large amount of data was available on products that went through the logistics process. The focus of the work was on the construction of a machine learning model, specifically using language modeling techniques, that could predict the probability that the product could be identified with a label that is associated with certain physical characteristics of the product and based on the different textual descriptors that make up the identification of the products published by the sellers.

As a result, a series of models with promising results were obtained that present savings of more than 95% compared to the highest savings that could be achieved on a test data set. In addition, these present a very good predictive quality, managing to separate the classes of processable and non-processable products, correctly predicting more than 95% of the labels of the data not seen and used for the evaluation.

Agradecimientos

En primer lugar, me gustaría agradecer a mi tutor, Mg. Pablo Roccatagliata, por su continuo apoyo durante mi investigación y desarrollo. Por su presencia constante para ayudarme a atravesar este desafío tan gratificante.
Mi profundo agradecimiento hacia la Universidad Torcuato Di Tella que me brindó las herramientas para seguir evolucionando mi carrera profesional. Mis compañeros y profesores del Master in Management + Analytics por compartir su conocimiento y experiencia que me ayudaron a crecer en el aspecto profesional, académico y por último y no menor, en lo personal.
No puedo terminar sin decir lo agradecido que estoy con familia y amigos por haberme brindado tanto apoyo y ánimos durante este trayecto de mi vida.
Gracias!!

Índice

1. Introducción
   - 1.1. Motivación
   - 1.2. Descripción del Contexto
     - 1.2.1. Introducción
     - 1.2.2. Ecosistema
     - 1.2.3. Modalidades de Envíos
     - 1.2.4. Tipos de Logística
     - 1.2.5. Servicio Tercerizado
     - 1.2.6. Dimensionamiento Inicial
   - 1.3. Descripción de la Problemática
     - 1.3.1. Proceso de Publicación
     - 1.3.2. Proceso de Compra
     - 1.3.3. Proceso de Envío
     - 1.3.4. Correios
     - 1.3.5. Problemática
     - 1.3.6. Dimensionamiento
   - 1.4. Descripción de la Oportunidad
     - 1.4.1. Objetivo
     - 1.4.2. Aplicación
     - 1.4.3. Justificación
     - 1.4.4. Hipótesis
   - 1.5. Aprendizaje Automático
     - 1.5.1. Introducción
     - 1.5.2. Procesamiento del Lenguaje Natural
   - 1.6. Metodología
   - 1.7. Alcance

2. Datos
   - 2.1. Dataset
   - 2.2. Análisis Exploratorio
     - 2.2.1. Calidad de los Datos
     - 2.2.2. Exploración de Variables
   - 2.3. Limpieza
   - 2.4. Preprocesamiento
     - 2.4.1. Principales Generadores de Ruido
     - 2.4.2. Expresiones Regulares
     - 2.4.3. Separación del Conjunto de Datos
   - 2.5. Separación del Conjunto de Datos

3. Métodos y Procedimientos
   - 3.1. Aprendizaje Supervisado
   - 3.2. Clasificación

3.2.1. Tipos de Clasificación
3.2.2. Algoritmos de Clasificación
3.2.3. Fast Text
3.3. Pipeline
3.3.1. Introducción
3.3.2. Ingeniería de Features
3.4. Evaluación del Modelo Predictivo
3.4.1. Métodos de Evaluación
3.4.2. Métricas
3.5. Software

4. Experimentos
4.1. Motivación
4.2. Descripción
4.3. Resultados
4.4. Interacción de las Variables

5. Conclusiones

6. Recomendaciones
6.1. Limitaciones
6.2. Próximos pasos

7. Anexos

8. Bibliografía

1. Introducción

Como introducción, se dará a conocer la información relevante y con el nivel de detalle necesario para comprender los diferentes componentes que construyen el contexto de la oportunidad que se plantea como desarrollo central del presente trabajo.

1.1. Motivación

La oportunidad tiene su origen en una empresa de la industria de internet, de comercio electrónico, conformada por diferentes unidades de negocio (*Marketplace, Publicidad, Envíos, Shops y Pagos*).

Si bien en los últimos años se ha enfocado en desarrollar y acelerar la curva de maduración de los negocios correspondientes a Fintech (*Pagos, Créditos*, etc) y a *Envíos*, la unidad de *Marketplace* sigue siendo una parte muy importante del ecosistema y específicamente, en el mercado brasileño, el cual representa aproximadamente el 70 % de dicha unidad y es el más importante.

En Brasil, la operación genera más de *1 millón de envíos diarios*. Muchos de estos envíos son gestionados directamente entre compradores y vendedores que utilizan la plataforma, otra parte de los envíos son realizados directamente a través de la logística propia donde los artículos vendidos están almacenados en los propios centros distribución y, por otro lado, más del 18% de los envíos son gestionados a través de proveedores tercerizados. De estos últimos, alrededor de un 5% presentan dimensiones que permiten clasificar a los paquetes como "grandes".

Cabe destacar que, más del 85% de los envíos gestionados por la plataforma corresponden a un solo ítem. De hecho, debido a los beneficios de la gestión, rastreo y protección que provee este servicio, muchos vendedores se ven motivados y optan por fraccionar la orden de compra (en caso de que la compra sea de más de un ítem) con el objetivo de utilizar la opción de envío gestionado.

A finales de 2019, todos los paquetes, sin importar sus dimensiones, tenían el mismo costo de envío por kilómetro. Sin embargo, durante el año 2020, los proveedores decidieron aumentar hasta un 200% los precios de los envíos de paquetes de gran tamaño (si alguno de los lados es > 140 cm por lado). Por otra parte, cada paquete enviado que supere las dimensiones indicadas por los carriers (cuando alguno de los lados es > 70 cm por lado), denominados *No Maquinables*, genera una multa de $R 80 (reales).

A raíz de las nuevas condiciones impuestas por el proveedor, la ineficiencia que puede introducir la empresa e-commerce dentro del proceso logístico, es el hecho de enviar paquetería *No Maquinable* (cuando alguno de los lados es > 70 cm por lado).
Es importante mencionar que la mensura y el registro de los paquetes enviados se realiza in situ, es decir, una vez que estos llegaron al operador logístico. En este punto, el proceso de logística inversa si bien es factible, no es la mejor opción debido a los costos asociados y la afectación de la experiencia del usuario. No obstante, el paquete *No Maquinable* continúa su camino para llegar a destino y la empresa e-commerce percibe la multa antes mencionada.

Este costo adicional se debe a la interrupción momentánea en el proceso de recepción de mercadería. Si el paquete no puede ser mensurado, etiquetado, ni registrado mediante el proceso automático, este es separado de la línea para realizar la operación de forma manual y luego continuar con el proceso de envío.

Por otro lado, las publicaciones activas no cuentan con una descripción obligatoria de las dimensiones del producto ofrecido. Por lo tanto, no es factible pausar estas publicaciones ni calcular un nuevo costo de envío para transferirlo al comprador o al vendedor de forma previa a la finalización del proceso de compra.
Por lo antes mencionado, resulta dificultoso determinar de antemano si el paquete a enviar podría ser motivo de la percepción de la multa o no.

A continuación, se realizará una descripción del contexto que gira en torno a la problemática para que se pueda comprender de manera holística.

1.2. Descripción del Contexto

1.2.1. Introducción

Envíos es el servicio que permite a los vendedores ofrecer envíos gratis, destacar sus publicaciones y entregar sus productos de forma rápida y segura. El vendedor se apoya en la gestión de la plataforma y servicios de logística que brinda la empresa. De esta manera, los productos adquiridos llegan a sus compradores de acuerdo con las condiciones pactadas y brindando la mejor experiencia de compra posible.

El objetivo es mejorar la experiencia de los usuarios haciendo envíos cada vez más rápidos, con una promesa de entrega precisa y al mejor precio del mercado. El compromiso es simplificar la tarea de los vendedores, resolviendo su logística para que ellos puedan enfocarse en vender más.

¿Cómo funciona?

1. El vendedor activa la opción de *Envíos*.
2. El comprador paga el producto junto con el envío a través de la plataforma de *Pago*.
3. El vendedor imprime la etiqueta, la pega en el paquete y en función del servicio que tenga activo lo lleva a una sucursal de correo o es recolectado por su domicilio.
4. ¡Listo! El comprador en función de la opción que haya elegido recibe el producto en su domicilio o lo retira por una sucursal de correo o punto de pick up.

1.2.2. Ecosistema

La empresa está conformada por una plataforma de comercio electrónico con operaciones en 13 países de América latina donde millones de usuarios compran y venden productos a través de Internet. Se venden productos de pequeñas y medianas empresas, productores, fabricantes, importadores, emprendedores, minoristas, mayoristas, individuos particulares, concesionarios, etc. La plataforma de *Pago* es una de las más grandes de la región. Desde 2003 es la solución de pagos para aquellas empresas, emprendimientos o personas físicas que quieren vender en su propio sitio web, redes sociales e incluso e-mails.

Permite pagar, cobrar por Internet de manera segura, simple y cómoda con una gran variedad de medios de pagos en Argentina, Brasil, Chile, Colombia, México y Venezuela.
Shops pone a disposición de las empresas, pymes y emprendedores su tecnología para que puedan llevar sus negocios al mundo digital y vender por comercio electrónico con sencillez y seguridad. Esto permite tener medios de pagos online con la solución de *Pago*, llevar tráfico al sitio con *Publicidad* y gestionar el stock de sus productos en un solo lugar.

Publicidad le permite a las marcas y empresas crear anuncios de texto con links para ampliar su visibilidad en las diferentes páginas dentro del *Marketplace*. El sitio cuenta con millones de productos publicados al mismo tiempo, lo que lo convierte en una de las mayores audiencias de internet en sitios retail, además de ser uno de los lugares de decisión de compra por excelencia.

Figura 1: esquema del ecosistema de la empresa.

Habiendo dado un repaso por el ecosistema conformado por la empresa e-commerce. Se explicarán resumidamente las *modalidades de envíos y los tipos de logística* dentro de Envíos.

1.2.3. Modalidades de Envíos

- No Especificado

El vendedor no especifica ningún precio de envío para sus artículos y el comprador tiene que ponerse en contacto con el vendedor para acordar una opción de envío.

- Custom

El vendedor carga una tabla con los precios de envío a cada región y se hace cargo de toda la logística.

- E1

La gestión y seguimiento del envío corre por cuenta de la plataforma pero la ejecución operativa de este la realiza el vendedor. Este caso se da cuando el vendedor tiene una red logística ya desarrollada y la utiliza.

- E2

La gestión, seguimiento y realización del envío está a cargo de la plataforma. Para este caso, se utilizan diversos servicios logísticos tercerizados llamados *carriers* (correos). Los precálculos que se realizan, como el costo del envío, fecha estimada de entrega, entre otros; son realizados por la plataforma.

Cabe destacar que, la modalidad E2 será el foco de este proyecto debido a que la problemática se genera específicamente en esta modalidad de envío.
A continuación, se mencionarán los diferentes tipos de logística existentes, es decir, los diferentes caminos por el cual un vendedor decide realizar su envío.

1.2.4. Tipos de logística E2

- Fulfillment (F)

Los vendedores envían sus productos especialmente seleccionados al centro de distribución de la empresa e-commerce, donde son almacenados hasta el momento de concretar una venta. Luego se los distribuye con algunos de los carriers contratados.

- Self Service (SS)

El vendedor se integra al proceso logístico, aportando recursos propios y la plataforma presta su know - how y tecnología para gestionar todo el desarrollo.

- Cross Docking (CD)

Para los vendedores con mayor volumen de ventas, un carrier se encarga de recolectar los productos por el punto de origen. Luego, todos los ítems son entregados en un hub donde se elige el carrier más conveniente para continuar con el recorrido.

- Drop Shipping (DS)

El vendedor despacha los productos en un punto de correo designado. Los paquetes siguen el flujo propio del correo hasta que este le llega al comprador. La plataforma no tiene injerencia en el proceso logístico.

Cabe destacar que, este último tipo de logística dentro de la modalidad de envío E2, es donde se ha identificado la oportunidad para el uso de aprendizaje automático.
Un actor muy importante que entra en juego dentro del proceso logístico es el proveedor tercerizado. La empresa actual no tiene la infraestructura, ni la flota de vehículos para proveer el servicio de punta a punta.

1.2.5. Servicio Tercerizado

Para cualquiera de las modalidades y de los tipos de logística descritos anteriormente, los carriers utilizados tienen condiciones de servicio que son acordadas y documentadas bajo contrato.

La condición de interés que generó la necesidad de innovar sobre el proceso logístico y de la cual es partícipe de este proyecto, corresponde a las dimensiones de los paquetes enviados.

Todos los productos enviados a través de carriers deben cumplir con dimensiones máximas de envío. Ninguno de los lados de un paquete puede superar los 70 cm de longitud, estos son denominados paquetes *Maquinables*; por ende, los paquetes que superen dichas medidas en al menos uno de sus lados son denominados *No Maquinables*.

A continuación, se brinda un detalle inicial de los números principales que dimensionan la unidad de negocios de *Envíos* de la empresa e-commerce para los países más importantes donde tiene su operación.

1.2.6. Dimensionamiento Inicial

Como se mencionó, la operación de *Envíos* en Brasil es la más importante para el *Marketplace.* Se envían más de 1 millón de envíos por día entre las diferentes modalidades de envío y tipos de tipos de logística. Se presentan los indicadores principales del dimensionamiento.

Envíos Totales (mes promedio)		
Brasil	México	Argentina
31.000.000	12.000.000	9.000.000

Brasil (mes promedio)		
Cross Docking	Fulfillment	**Drop Shipping**
16.000.000	9.000.000	**6.000.000**

Los datos presentados están calculados en base a datos históricos de los últimos 6 meses. Son valores promedio.

Respecto al mercado de Brasil, se espera un crecimiento proyectado YoY del 30% el cual se basa principalmente en el incremento de ventas. Por otro lado, se estima que la penetración de Drop Shipping proyectada sea del 1%.

Habiendo descrito el tamaño de *Envíos*, se dará introducción a la problemática que se quiere resolver. Para ello, se comenzará a explicar los tres procesos principales que tiene relación con la oportunidad de aprendizaje automático que será detallada más adelante.

1.3. Descripción de la Problemática

1.3.1. Proceso de Publicación

En esta plataforma e-commerce, más del 90% de los vendedores están conformados por personas físicas y personas jurídicas con bajo volumen de ventas respecto a otros grandes jugadores del mercado que también tienen marcas establecidas y productos determinados. Cuando se menciona bajo volumen, implica que la publicación suele ser solo de una unidad. Sin embargo, existe la posibilidad de que la publicación tenga stock de algunas unidades. No suele superar el orden de magnitud de diez.
Esto influye en la cantidad de artículos que pueden ser estandarizados y publicados de una misma manera dentro del sitio web, bajo el mismo título, descripción, atributos, fotografías, etc.

Dentro del ecosistema de la plataforma, resulta muy dificultoso homogeneizar los productos publicados y que todos los vendedores se pongan de acuerdo en la generación de la misma publicación para un mismo producto. De hecho, todos se quieren destacar de alguna manera para lograr mayores ventas. Pequeñas variaciones en los descriptores mencionados generan nuevos productos dentro del sistema, por ende, la generación de nuevos productos a nivel sistema es muy alta y frecuente.

Cuando un vendedor quiere generar una nueva publicación debe realizar los siguientes pasos:

1. Se debe elegir la opción de dominio de productos más grande, a más alto nivel (productos, servicios, vehículos, otros).

2. Eligiendo el dominio de productos. Luego, se debe ingresar el título del producto (que se concibe como una breve descripción) para verificar si corresponde a un artículo ya publicado y/o estandarizado.
   Como resultado, se despliegan las principales categorías a las cuales puede pertenecer el artículo a publicar.

3. Se debe elegir la categoría a la cual pertenece el producto. En caso de que no se visualice una coincidencia, se puede definir la categoría para dicho ítem.

4. El siguiente paso es indicar si la condición del producto es nuevo o usado.

5. Se debe confirmar el título de la publicación o se puede seguir completando (tiene un límite de cantidad de caracteres) con el objetivo que el producto sea fácil de identificar para el usuario comprador.

6. Luego, se solicita que se agreguen fotografías que permitan visualizar el artículo y que mejoren la exposición de la publicación.

7. Se deben establecer las condiciones de ventas y envío.

8. Se debe elegir el tipo de publicación y luego se procede a generar la publicación.

1.3.2. Proceso de Compra

1. El usuario busca algún producto que se encuentra publicado en el Marketplace con la intención de compra. Dicha publicación y dicho ítem son identificados de manera unívoca.

2. Al comprar un producto, se genera la orden de compra. En caso de que la publicación contenga la opción de *Envío* y se elija esta opción, se generará un identificador del envío.

   Las opciones de envío que un comprador puede visualizar son: Envío a Domicilio, Retiro en Correo y Otros Puntos ó Retiro en Domicilio del Vendedor

3. Dentro de la orden de compra, el comprador puede haber comprado una o más unidades de este u otro ítem distinto.

4. En base a las características físicas del artículo y ruta (origen, destino) se determina el precio teórico que tiene cada envío.

   El *costo real* del envío está conformado por el *costo teórico - descuentos - multas + seguro de envío + redondeo.*

5. Una vez que la compra haya sido concretada. El vendedor imprime la etiqueta de envío, la cual adhiere al mismo paquete y continua con el proceso logístico hasta que el producto llega al comprador.

1.3.3. Proceso de Envío

1. Una vez concretada la compra por parte del usuario comprador, el vendedor debe realizar el embalaje del producto de acuerdo con los requerimientos indicados por la plataforma, con el objetivo de preservar y poder manipular el paquete de la manera correcta.

2. Luego de embalar el producto, el vendedor debe imprimir la etiqueta (la cual descarga de la plataforma) y adherirla al paquete. La etiqueta debe tener impreso algunos descriptores del artículo vendido más los datos de origen y destino.

3. Teniendo el artículo listo para enviar, el vendedor debe dejar el paquete en la sucursal de correo más cercana.

4. El proveedor logístico, realiza el retiro de paquetería de manera periódica y lleva todas las encomiendas al centro de distribución correspondiente.

   Cabe destacar que, el vendedor solo embala e identifica el paquete. No realiza ninguna tarea de mensura o pesaje.

Figura 2: flujograma simplificado del proceso logístico. El color verde indica los pasos desarrollados por la plataforma y el color azul representa los pasos realizados por el vendedor.

Habiendo repasado los procesos de publicación, compra y envío. Se presentará el proveedor principal del servicio tercerizado con el cual desencadenó la problemática.

1.3.4. Correios

Correios es el proveedor más grande de servicios logísticos en Brasil. Correios es el nombre comercial de la Empresa Brasileira de Correios e Telégrafos, también llamada ECT.

Se trata de la empresa pública federal responsable del sistema de envío y entrega de correspondencias en Brasil. La legislación brasileña prevé el monopolio de la ECT en los servicios de carta, tarjeta postal, correspondencia agrupada y telegrafía. Tiene su origen en Brasil (año 1663), con la creación del Correio-Mor en Río de Janeiro. En 1931, con el decreto 20.859 se fundó la Dirección General de Correios con la Subdirección General de Telégrafos y crea el Departamento de Correios y Telégrafos.

La ECT fue creada en 1969, como empresa pública vinculada al Ministerio de Comunicaciones mediante la transformación de la autarquía federal que era, entonces, Departamento de Correios y Telégrafos.

Siendo una empresa estatal tiene una gran participación en el mercado y tiene un poder de negociación muy importante. Además, la actividad que realiza es representada por uno de los gremios laborales más importantes en todo el país. No es una opción no utilizar el servicio de dicha empresa a menos que el proceso logístico de punta a punta sea desarrollado de manera propia.

Este proveedor es un eslabón muy importante de la cadena de distribución que corresponde al proceso de *Drop Shipping*, caso de uso en cuestión.

Habiendo brindado el contexto suficiente para poder interpretar la relación entre los diferentes puntos del proceso de envíos de los paquetes, se dará un mayor detalle sobre la problemática que se introdujo al inicio del presente trabajo.

1.3.5. Problemática

Sobre finales de 2019 y para los contratos correspondientes al año 2020, los proveedores de transporte presentaron nuevos requisitos respecto a la manipulación y procesamiento de la paquetería que llegase a los centros de distribución de estos.

Esto se debió a la incorporación de nueva tecnología que permite automatizar los procesos internos de manejo de materiales, identificación, etiquetado y ordenamiento para distribución final. Ejemplo de una máquina de procesamiento, *Fig 3.*

Figura 3: ejemplo de una de las máquinas utilizadas dentro del proceso de logística interna para la manipulación y mensura de paquetería. Fuente: Falcon Autotech

Si bien la incorporación de esta tecnología produce una economía de escala[1] para la empresa que lo incorpore, esto conlleva algunas restricciones para sus clientes. La más importante y la cual se convierte en la esencia de la problemática para la empresa de e-commerce, es que las máquinas con las cuales se automatizan[3] los procesos tienen limitaciones en cuanto a las dimensiones[2] de paquetería que pueden manejar. Ninguno de los lados de un paquete puede superar los 70 cm. Como se mencionó, este ítem se denomina *Maquinable*. Caso contrario, el paquete se denomina *No Maquinable.*

Cuando un paquete No Maquinable arriba al centro de distribución del proveedor debe ser manipulado de manera semi manual (con ayuda de elementos o dispositivos que facilitan el manejo de materiales) y debe atravesar un proceso paralelo para que pueda ser enviado a destino.

Independientemente de que el paquete sea Maquinable o No Maquinable, todo artículo debe ser pesado, medido, identificado, registrado y ordenado para su posterior distribución hacia su destino final.

El procesamiento manual acarrea un costo operativo mayor, por ende, esta fue la principal motivación por la cual el proveedor quiso cambiar las condiciones del contrato de servicio.
En este, se expresó que *en el caso de recibir paquetería No Maquinable, se impondría una multa de 80 reales* (este monto se actualiza en el tiempo), *15 dólares* para redondear.
Cabe destacar que esta problemática no hace foco en el proveedor logístico. Los prestadores, en especial Correios, mejoraron sus procesos para manipular y procesar mejor los paquetes maquinables. Esto impacta aguas hacia arriba en el proceso de logística, es decir, le genera un problema en la gestión de su cliente, en este caso, la plataforma e-commerce.

1.3.6. Dimensionamiento

Trayendo a colación el dimensionamiento inicial, quedaría por dimensionar la oportunidad potencial dentro alcance establecido (Modalidad de Envío E2 - Drop Shipping).

Envíos Totales (mes promedio)		
Brasil	México	Argentina
31.000.000	12.000.000	9.000.000

Brasil (mes promedio)		
Cross Docking	Fulfillment	**Drop Shipping**
16.000.000	9.000.000	**6.000.000**

Envíos de un solo ítem (mes promedio)		
Drop Shipping	Proporción	**Resultado**
6.000.000	85%	**5.100.000**

Envíos No Maquinables (mes promedio)		
Drop Shipping	Proporción	**Resultado**
5.100.000	4%	**204.000**

Nuevos ítems (mes promedio)		
Drop Shipping	Proporción	**Resultado**
204.000	62%	**126.480**

Si bien, más adelante se definirá una función de ahorro específica para evaluar la presente propuesta; de manera aproximada, se podría decir que la oportunidad de ahorro potencial promedio de la problemática es de USD 1.897.200 al mes. Este número se basa en el simple cálculo de multiplicar 126.480 ítems No Maquinables por la suma de USD 15. Desde ya, esto sólo representaría un caso ideal ya que se debe contemplar los costos y ahorros que se pueden dar en cada escenario al realizar una predicción. Esto será desarrollado más adelante.

A continuación, se desarrollarán las aristas principales que conforman la oportunidad descubierta.

1.4. Descripción de la Oportunidad

1.4.1. Objetivo

El costo del procesamiento y envíos de paquetes depende principalmente de las características físicas de estos y del nivel de servicio requerido[4]. Las dimensiones de los paquetes juegan un rol importante, sobre todo con la implementación de procesos automatizados. Aquí es cuando los paquetes No Maquinables no son aptos para el procesamiento automático y deben ser manejados y procesados de manera tradicional a través de diversos recursos de manipulación y movimiento de materiales. Por esta razón, el procesamiento tradicional es relativamente más costoso que el procesamiento automático.
Por lo antes mencionado, se desea desarrollar un modelo de aprendizaje automático que permita clasificar los paquetes, de un solo ítem, en *Maquinable o No Maquinable*.

1.4.2. Aplicación

Dado que la implementación de la solución sería realizada al final del proceso de publicación de un artículo, tanto el comprador como el vendedor, podrían visualizar las opciones de envío habilitadas, filtradas previamente gracias al modelo de clasificación.
En caso de que la predicción del modelo arroje como resultado que el ítem es *no* maquinable, en una primera instancia, el vendedor no podría ofrecer el envío por cuenta y orden de la plataforma e-commerce; sólo tendrá las opciones de gestionar el despacho del producto por sus propios medios o acordar la entrega con el comprador.
Alternativamente, se podría evaluar si es factible modificar la experiencia del usuario vendedor, agregando la opción de asumir todos los costos adicionales y utilizar el circuito gestionado a pesar de que el artículo sea no maquinable.
Dado que esta oportunidad se plantea como un problema de clasificación binario, se consideran los siguientes casos de uso:

- *Verdadero - Positivo*: paquete *maquinable* predicho como *maquinable*.
- *Verdadero - Negativo*: paquete *no maquinable* predicho como *no maquinable*.
- *Falso - Positivo*: paquete *no maquinable* predicho como *maquinable*.
    - Podría verse afectada la venta de este ítem en particular pero el vendedor tendría la posibilidad de rectificar la información predicha y mitigar el FP para otra venta futura.
- *Falso - Negativo*: paquete *maquinable* predicho como *no maquinable*.
    - El proceso de envío no se interrumpe. Se incurre en el costo adicional (multa)

Para la evaluación de los experimentos a realizar, se desarrollará una función de costo que contemple cada uno de los escenarios planteados. Cada escenario tendrá las variables que incrementan y disminuyen el costo asociado.

1.4.3. Justificación

La cadena de suministro es uno de los puntos más importantes y sensibles dentro de la cadena de valor para los usuarios, compradores y vendedores. Los grandes proveedores que funcionan como una tercera parte dentro del proceso logístico, requieren mensurar las características físicas de los paquetes que son enviados a través de su red.

Si bien existe un proceso de logística inversa, este presenta sobre costos en la operación física del e - commerce, además de ofrecer una mala experiencia para los usuarios del sitio lo cual puede afectar significativamente la confianza, la practicidad y la satisfacción de realizar transacciones a través de este.
La presente propuesta intenta mitigar la contingencia que un producto no maquinable puede generar sobre el proceso logístico. Dada la tecnología e infraestructura existente es factible desarrollar un paso dentro del proceso online, programático y sistemático que puede reducir la tasa de error, disminuir el costo adicional operativo sobre la cadena de suministro y evitar que la experiencia de los usuarios se vea afectada por tal motivo.

Por otro lado, el impacto potencial de la oportunidad se presentó en las secciones 1.2.6 y 1.3.6.

1.4.4. Hipótesis

Debido a que la mayoría de los atributos pertenecientes a un artículo y persistentes en el sistema de información, son campos con texto. Se plantea la hipótesis de que, en base a datos históricos de los miles de artículos creados en el sitio, es factible representar y modelar la oportunidad mediante técnicas de Aprendizaje Automático[5] (Machine Learning en inglés). Específicamente, el Procesamiento del Lenguaje Natural.
Como paso final, se desea obtener una clasificación binaria que permita clasificar los artículos maquinables y no maquinables y así, evitar que el producto sea enviado a través del proceso de envío incorrecto.

1.5. Aprendizaje Automático

1.5.1. Introducción

Si se tuviera que describir qué es el aprendizaje automático en una línea de tweet, según Cassie Kozyrkov[6] (Chief Decision Scientist, Google); es un “etiquetador de cosas” que toma la descripción de algo y te dice la etiqueta que le debería corresponder. Es una nueva forma de programación, una nueva manera de comunicar tus deseos a una computadora a través de ejemplos, en vez de instrucciones impuestas. Dicho de otra manera, es la ciencia que persigue que las computadoras actúen sin estar programadas explícitamente.

Yendo un poco más profundo y para entender de dónde proviene, el aprendizaje automático es una rama de las ciencias de la computación[7][8]. Hace referencia a un conjunto de algoritmos que utilizan la ciencia estadística para encontrar patrones y relaciones en los

datos de entrada. La revolución tecnológica y el crecimiento exponencial de la capacidad de cómputo han impulsado la aplicación de estas técnicas a diferentes industrias como fintech, comercio, seguros, salud, agro, entre otras. Por mencionar algunos ejemplos de aplicaciones, sistemas de recomendación, prevención de fraude, personalización de contenido en redes sociales, predicción del tiempo de entrega para la logística, etc.

Además, los algoritmos de aprendizaje automático han demostrado ser extremadamente útiles para realizar clasificaciones de clases o etiquetas.
En la actualidad, producto del comercio electrónico, millones de datos de usuarios son registrados a través de diferentes sistemas de compras y pagos alrededor del mundo. Todas las transacciones, detalles de estas, datos financieros y demográficos conforman una gran variedad y un gran volumen de datos que deben ser explorados y explotados a gran velocidad (big data) con el objetivo de obtener información que permitan identificar las necesidades, las costumbres, los gustos de los usuarios; para acercarles el producto, el precio conveniente, en el momento adecuado.
Los avances del aprendizaje automático están brindándoles a los comercios información detallada y sin precedentes sobre el comportamiento del cliente, lo que les permite mejorar la experiencia de este de diversas formas: recomendaciones de productos, búsqueda personalizada, atención al cliente, precios dinámicos, entre otros.[9]

1.5.2. Procesamiento del Lenguaje Natural

Por “procesamiento”, nos referimos a que las computadoras ven el texto de manera diferente a los humanos. Las computadoras no pueden ver nada más que números[10]. Cada carácter (letra) que vemos en una computadora es en realidad una representación numérica para una computadora. Como ejemplo, se puede realizar la asignación entre números y caracteres determinada por una "tabla de codificación". El más simple pero más común en el análisis de texto, es la codificación ASCII. Se puede visualizar una pequeña tabla ASCII de muestra.

Símbolo	DEC	OCT	HEX	BIN
a	97	141	61	01100001
b	98	142	62	01100010
c	99	143	63	01100011
d	100	144	64	01100100
e	101	145	65	01100101

En la tabla que aparece a continuación, se puede ver que hay seis formas diferentes de codificar la palabra “CAFÉ” en código ASCII [11]. La palabra de la izquierda es lo que ve el humano y su representación ASCII (lo que ve la computadora) está a la derecha.

Palabra	Representación ASCII

cafe	99-97-102-101
Cafe	67-97-102-101
CAFE	66-65-70-69
café	99-97-102-233
Café	67-97-102-233
CAFÉ	66-65-70-201

Cualquier ser humano podría darse cuenta de que se trata de la misma palabra solo que está escrita de seis maneras distintas. Para una computadora, son seis palabras diferentes y por esto, algo que se explicará más adelante, es muy importante la forma con la cual se transforman los datos de entrada para que puedan ser consumidos por un algoritmo de aprendizaje automático.
Por "lenguaje natural", nos referimos a un lenguaje que los seres humanos utilizamos para la comunicación diaria; a través de idiomas como español o portugués. A diferencia de los lenguajes artificiales como los lenguajes de programación y las notaciones matemáticas, los lenguajes naturales han evolucionado a medida que pasan de generación en generación, y son difíciles de precisar con reglas explícitas. Consideraremos al procesamiento del lenguaje natural, o NPL para abreviar (siglas en inglés), en un sentido amplio para cubrir cualquier tipo de manipulación informática del lenguaje natural. En un extremo, podría ser tan simple como contar la frecuencia de palabras para comparar diferentes estilos de escritura. En el otro extremo, podría implicar el hecho de "comprender" las expresiones humanas completas, al menos en la medida de ser capaz de dar respuestas útiles.

Las técnicas de NLP hacen posible que las computadoras puedan procesar masivamente cantidades de datos no estructurados, por ejemplo, texto de libros, mensajes de redes sociales, comentarios de clientes o cualquier registro de audio. En términos generales, NLP intenta dividir el input en partes elementales más cortas, minar y obtener las relaciones entre estas partes y explorar cómo funcionan las piezas juntas para crear una respuesta y un significado.

Las aplicaciones de NLP son diversas. Se puede utilizar para clasificar contenido, detección de plagios, análisis de sentimiento, traducción, conversión de texto a voz o viceversa, entre otros.

1.6. Metodología

Se plantea desarrollar el problema en base al siguiente proceso de trabajo.

Figura 4: flujograma del proceso simplificado utilizado para abordar el desarrollo de la problemática

El primer paso del proceso hace referencia al *Entendimiento del Negocio*, el cual se enfoca en entender y estructurar los objetivos y requisitos del proyecto desde una perspectiva del negocio. En este paso, se busca capturar la problemática del negocio, convertirla en una oportunidad de manera que pueda ser resuelta con la utilización del Aprendizaje Automático y desarrollar un plan de trabajo. Por otro lado, se debe obtener un claro criterio de éxito y que impacte realmente en la métrica de negocio deseada.

El siguiente paso se define como *Entendimiento de los Datos*, el cual comprende la recolección y exploración de los datos existentes. El objetivo de esta etapa es obtener un entendimiento profundo de los datos, identificar problemas de calidad y descubrir algunos de los primeros hechos sobre los datos y la problemática correspondiente. Durante esta exploración profunda, es posible que surjan nuevas hipótesis y hasta puede que afecte la definición inicial que se realizó en el primer paso.

La siguiente etapa se denomina, *Preparación de los Datos*. Comprende todas las tareas necesarias para construir el conjunto de datos final que será utilizado para realizar el entrenamiento del modelo y experimentos. Si bien esta etapa depende de la naturaleza de la problemática y de la forma que desee representar y modelar, las tareas que se suele realizar son: limpieza de datos, estandarización, transformación, ingeniería de la predicción, ingeniería de features y selección de variables.

Cabe destacar que, las etapas descritas anteriormente son cruciales para lograr una solución robusta. Solo los conjuntos de datos que representen la problemática de manera correcta serán los que permitan obtener resultados deseables y cercanos a la realidad.

Una vez finalizadas las etapas previas, se procede a realizar el *Modelado*. En esta fase se exploran las diferentes técnicas que se pueden utilizar para capturar las relaciones existentes entre los datos. Cada técnica tiene parámetros específicos que deben ser configurados para generar inferencias con valor. Esta etapa puede generar la necesidad de realizar modificaciones en etapas anteriores y generar un flujo de retroalimentación.
.
Después de haber desarrollado uno o más modelos, se requiere la etapa de *Evaluación* que tiene el objetivo de determinar la calidad predictiva de la solución desarrollada y más importante, si pueden impactar en la métrica de negocio.

Todas las etapas descritas pertenecen a un proceso estandarizado llamado CRISP-DM[12] (Proceso Estándar Entre Industrias para la Minería de Datos) que proporciona un enfoque estructurado para planificar un proyecto de minería de datos. Cabe destacar que, en la *Fig 4* se presenta como un proceso lineal para facilitar su compresión, en la práctica se convierte en un ciclo donde cada paso puede retroalimentar pasos anteriores. Los lineamientos de este marco de trabajo siguen siendo ampliamente utilizados en la mayoría de los proyectos que involucran aprendizaje automático y se utilizaron para guiar la mayor parte de este proyecto.

1.7. Alcance

El objetivo de la propuesta no contempla el análisis y/o solución sobre la manipulación, fraccionamiento o manejo de materiales una vez que los artículos llegan al operador logístico. El motivo principal es que la operación no depende de la empresa e-commerce, es un servicio totalmente tercerizado sobre el cual no se tiene injerencia. Los derechos y obligaciones del servicio prestado están sujetos mediante el contrato firmado por las partes.
Por otro lado, no se contempla la etapa de implementación de la solución, ni el desarrollo de software necesario para introducir el modelo en un proceso productivo.

2. Datos

De acuerdo con la metodología propuesta, la exploración de datos[13] es una parte clave de las etapas de entendimiento de la oportunidad y de la preparación de datos.
Los principales objetivos son:

- Detectar problemas de calidad en los datos, que podrían afectar negativamente a los modelos de aprendizaje automático: valores nulos, valores atípicos, falta de integridad, cardinalidad irregular, entre otros.
- Entendimiento de los datos: comprender las características principales tales como los tipos de valores que puede tomar una variable, los rangos en los que caen los valores de esta y cómo los valores de un conjunto de datos para una variable se distribuyen en el rango que pueden tomar, parámetros estadísticos principales, etc.

2.1. Dataset

El conjunto de datos está conformado por más de 2 millones de registros. Cada registro corresponde a un envío realizado en Brasil bajo la modalidad y tipo de logística Drop Shipping.
La muestra de los datos recolectada corresponde a un conjunto de envíos realizados entre el 01/03/2020 y el 30/04/2020 en el mercado de Brasil.

2.2. Análisis Exploratorio

Luego de comprender la oportunidad o problemática, es necesario analizar los datos que se encuentran disponibles; para eso, se debe realizar un análisis exploratorio[14] del conjunto de datos.
Cada registro del dataset representa el envío realizado de un ítem adquirido en la plataforma y cada uno de estos contiene los 43 atributos. Menos de la mitad del total de atributos (15) corresponden a variables numéricas; de estas, diez (10) corresponden a variables numéricas continuas referidas a las características físicas del envío / ítem, como ser, peso, altura, ancho, largo, volumen, etc. Los 5 atributos numéricos restantes son variables enteras que identifican al ítem en sí, por ejemplo, número de dominio al cual pertenecen, id del artículo, id del envío, etc. Por otra parte, el resto de las variables (24) son consideradas como variables categóricas. Se encuentra el título del ítem enviado, el

dominio, diferentes tipificaciones de respecto a las categorías, marca, entre otras. El detalle de esto se encuentra en el Anexo A: Datos.

Cabe destacar que, uno de los mayores desafíos será descubrir la concatenación de atributos textuales que le permita la mejor descripción de un ítem y que pueda ser aprendido por el modelo predictivo. Por otro lado, se deberá trabajar fuertemente en el preprocesamiento y transformación de los datos para que el modelo pueda minar la información lo mejor posible.

2.2.1. Calidad de los Datos

La revisión de la calidad de los datos es una de las instancias más importantes en el análisis exploratorio para detectar problemas en la integridad del conjunto de datos. Se puede realizar tanto para variables continuas, ordinales y categóricas.

Para las variables continuas, las características que deben documentarse principalmente son: primer cuartil, media, mediana, tercer cuartil, máxima y desviación estándar. También, se debe tener en cuenta el número total de registros en el conjunto de datos, el porcentaje de registros los cuales presentan valores nulos y la cardinalidad. Cabe destacar que de acuerdo con la naturaleza del problema, puede haber otras características que sean relevantes.
Para las variables categóricas, se debe explorar y registrar los niveles más frecuentes, los valores únicos; también se debe incluir el porcentaje de instancias en el conjunto de datos a las que les falta un valor y la cardinalidad

La revisión de la calidad de los datos también debe incluir gráficos que permitan visualizar mejor los descubrimientos obtenidos. Por ejemplo, realizar un histograma o boxplot para cada variable continua en el conjunto de datos. Para variables ordinales o categóricas con cardinalidad menor a 10, se sugiere usar gráficos de barras en lugar de histogramas, ya que esto generalmente mejora la interpretación.

Después de realizar la revisión de calidad de los datos, se identificaron algunos problemas. Para obtener un mayor detalle del análisis de calidad de datos, se puede consultar el Anexo B: Calidad de los Datos.

Resumimos aquí algunos de los principales problemas de calidad de los datos encontrados.

Variables Sin Información

Existe una cantidad significativa de variables que, si bien son parte del contexto del envío, no aportan gran información para poder identificar si un producto pudiese ser maquinable o no. Algunas de las variables son: “Unnamed”, “shp_service_id”, “shp_quantity”, “shp_shipment_id”, “site_id”, entre otras. Esto fue corroborado con el área de negocio.

Valores Nulos

En el conjunto de datos hay diferentes variables que presentan valores nulos. Algunas de estas variables presentan valores nulos debido a la estructura o conformación de los datos de los productos. Por ejemplo, muchos productos no pueden ser descritos con categorías de

bajo nivel, por ende, las variables “L4” a “L7” presentan una gran proporción de valores nulos. Algo similar sucede con la variable “brand_name” dado que no todos los productos poseen una marca registrada.
Por otro lado, otras variables presentan valores nulos debido a un error en la obtención de los datos, por ejemplo, las variables “domain_id” y “L3”.
Si bien estos casos no representan un problema en términos de calidad de la información, es un inconveniente para los modelos de aprendizaje automático dado que la mayoría de los algoritmos no manejan bien los valores nulos. Se evaluará cada variable para determinar el tratamiento o transformación que se deba realizar para asegurar la calidad de los datos.

Discrepancia de valores

Realizando un análisis de las variables numéricas, podemos encontrar diferencias notables entre las mediciones realizadas por la empresa logística tercerizada versus las mediciones realizadas por la empresa e-commerce cuando el producto en cuestión pasó por el centro de distribución de la propia empresa (depende del tipo logística) en algún punto del proceso. También se puede dar la posibilidad de que sean mediciones predichas por la empresa e-commerce.
Debido a que el foco de la problemática se centra en el proceso operativo del proveedor tercerizado, se considera saludable tomar como dimensiones verdaderas aquellas que fueron tomadas por la tercera parte. Esto fue consultado con el equipo de Envíos de dicha empresa.

Alta Cardinalidad

Ciertas variables categóricas presentan alta cardinalidad, es decir, presentan una proporción significativa de categorías encontradas respecto a la cantidad de registros en el conjunto de datos. Por ejemplo, “item_id” o “attributes”. Por lo general, no hay mucho que se pueda hacer con este tipo de variables salvo eliminarlas del conjunto de datos debido a que si el modelo pudiese aprender algo podría generar un sobreajuste (overfitting) sobre el conjunto de datos de entrenamiento o generar algún método para que el modelo pueda procesar matrices ralas (matrices de gran tamaño con gran dispersión y cantidad de ceros).

Tipo de datos

Haciendo un análisis del tipo de variables encontradas en el conjunto de datos, algunas de estas tienen un tipo de dato erróneo. Por ejemplo, las variables “shp_service_id”, “shp_shipment_id” o “seller_id”. Si bien esto no presenta un riesgo para el modelado de la problemática, será corregido con el objetivo de manejar los datos de manera correcta.

Para más información sobre las acciones realizadas para asegurar la calidad de los datos y el armado final del conjunto de datos de entrenamiento, se puede consultar el Anexo B: Calidad de los Datos.

2.2.2. Exploración de Variables

Variables numéricas

- La mayoría de las variables numéricas que presenta el conjunto de datos explorado corresponden a las diferentes dimensiones que son tomadas en diferentes puntos del proceso. *Fig. 5*.

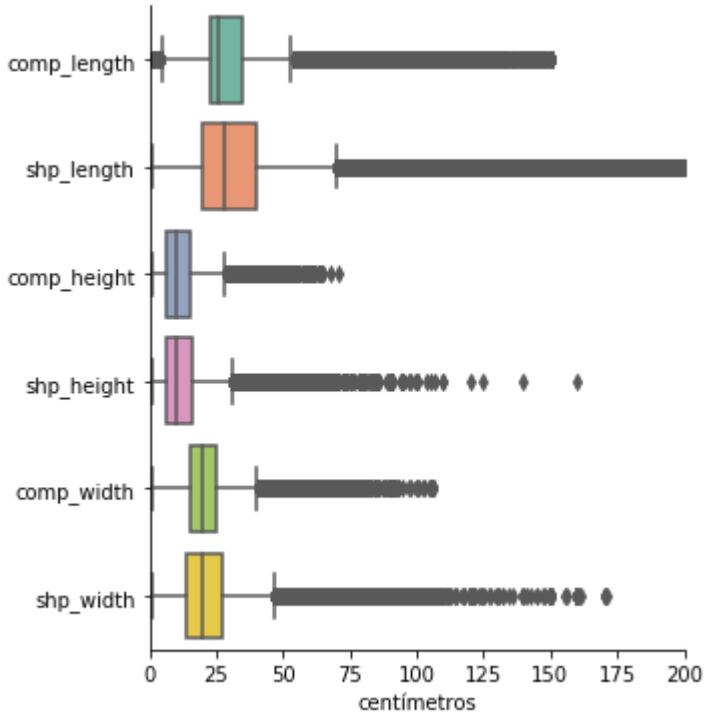


Figura 5: gráfico boxplot de variables numéricas. Estas corresponden a las dimensiones de los paquetes encontrados en el conjunto de datos.

En la visualización, queda en evidencia la discrepancia significativa que se produce entre las **dimensiones** del mismo tipo. Por ejemplo, el largo de los paquetes medidos en diferentes momentos del proceso logístico. Las variables que presentan el sufijo “comp”, son mediciones realizadas o predichas en puntos iniciales del procesamiento de la paquetería. Las variables que presentan el sufijo “shp”, son mediciones realizadas por el operador logístico tercerizado y de mayor confianza. Debido al entendimiento de la operación, este punto fue tratado dentro del plan de calidad de los datos. Independientemente del origen de la medición, todas las variables tienen una presencia alta de datos que podrían considerarse como outliers o extremos. De cualquier manera, no es factible descartar los registros correspondientes dado que estos valores podrían representar productos no maquinables. Recordemos que un producto que exceda, al menos un lado, las dimensiones permitidas, será considerado como no maquinable. Cabe destacar que, estas variables no podrían obtener durante el proceso predictivo real dado que son obtenidas luego de haber gestionado el envío.

Otra de las variables numéricas que podría ser relevante es el **precio** del producto. Intuitivamente, uno podría pensar que un producto de mayor tamaño, haciendo referencia a los productos no maquinables, podrían tender a poseer precios mayores respecto a artículos maquinables. Seguramente, esa regla podría cumplirse en algunos casos pero analizando el conjunto de datos vemos que para esta muestra, no lo es. Si vemos la tabla de los estadísticos principales, los valores medios de los precios de las clases son muy

cercanos. Por otro lado, vemos que ambas clases presentan valores máximos muy altos, como así también una desviación elevada.

Estadístico	Maquinable	No maquinable
conteo	6.59E+04	2.23E+06
media	6.06E+03	6.30E+03
desviación std	1.04E+06	2.56E+06
min	1.00E+00	1.00E-01
25%	1.02E+02	5.50E+01
50%	2.00E+02	1.20E+02
75%	3.99E+02	2.15E+02
max	2.23E+08	2.86E+09

Para poder entenderlo de mejor manera, se realizó un gráfico conocido como boxplot (*Fig. 6*) que hace más fácil la interpretación de los valores antes mencionados. Por practicidad, se impuso el límite del precio en el valor 5000 con el objetivo de tener una mejor visualización de las cajas.

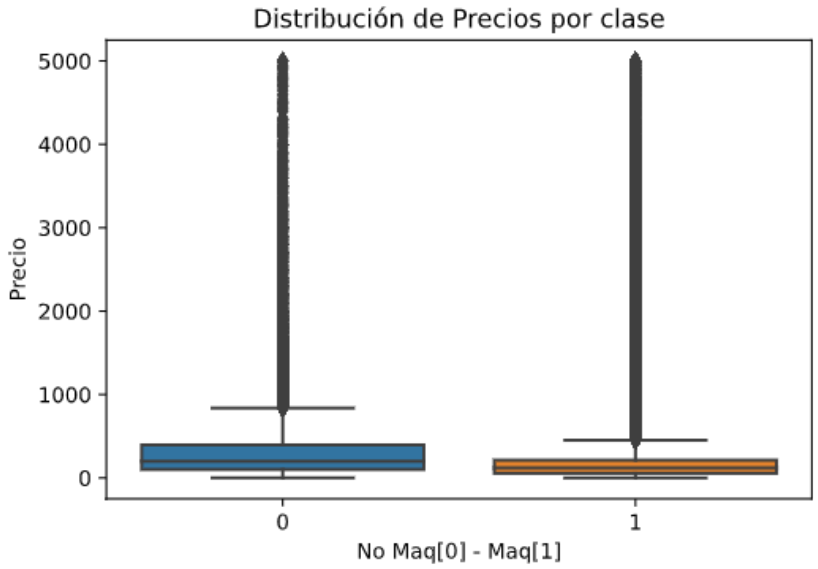


Figura 6: gráfico boxplot de la variable precio. A su vez, se muestra la segmentación por clases. No Maquinables y Maquinables

Tal como se puede ver, no se ve una diferencia tan grande entre los precios de las clases y, por otro lado, las colas de valores extremos son muy importantes. Esto refleja la diversidad de productos y precios para este dominio y muestra de datos. De manera complementaria, esto se puede ver a través de visualizaciones de

histogramas que reflejan la concentración de precios para cada una de las clases. *Fig. 7*.

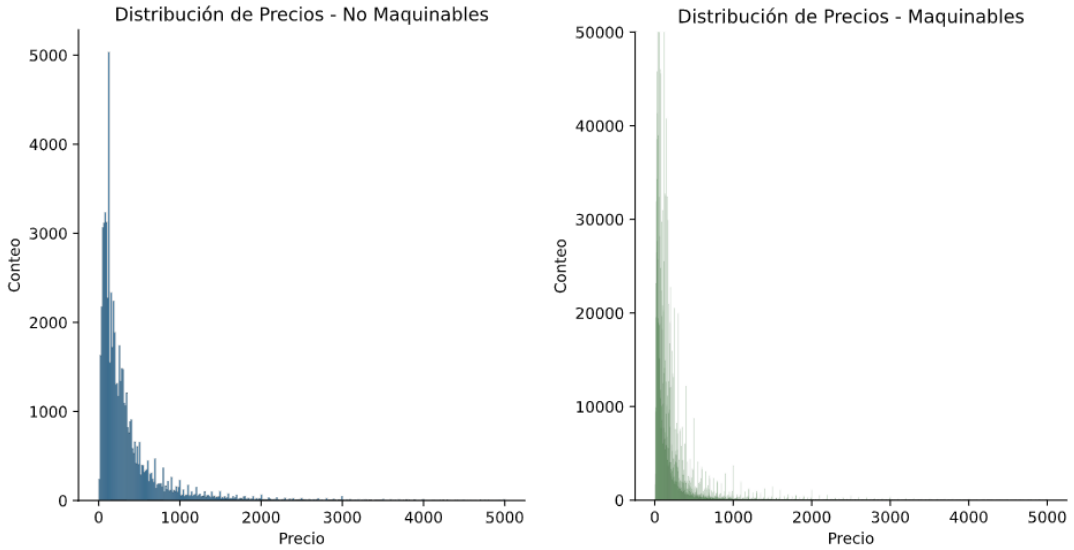


Figura 7: gráfico de distribución de la variable precio. A su vez, se muestra la segmentación por clases. No Maquinables y Maquinables.

En vista de esto, se podría inferir que el precio podría no ser una buena variable de la cual el modelo pudiese aprender.

Variables Categóricas

Debido a la naturaleza del problema y dado que la mayoría de los descriptores encontrados para un ítem están conformados por variables categóricas textuales, podría darse el caso que ninguna variable por sí sola represente y generalice correctamente el dominio de la problemática. Con el objetivo de continuar con la exploración se decide crear una nueva columna dentro del dataset que permita identificar los productos *Maquinables (todos los lados menores o iguales a 70 cm) y No Maquinables (alguno de los lados que supere los 70 cm)*. Recordemos que la definición antes mencionada se debe a las dimensiones de los paquetes que son necesarias para que puedan ser procesados a través del proceso manipulación, mensura, registro y ordenamiento automatizado. Respecto a los datos históricos, gracias a que se cuenta con las dimensiones de cada paquete es factible y simple poder determinar una etiqueta para cada producto. Se utilizaron las variables “shp_lenght”, “shp_height” y “shp_height” para determinar la etiqueta correcta. *Maquinables* será igual a *1* y *No Maquinables* será igual a *0*.

A continuación, se resumen algunos de los principales descubrimientos obtenidos.

Para este conjunto de datos, la proporción de envíos de productos No Maquinables es del 3%. Por ende, se puede determinar que el conjunto de datos está muy desbalanceado. El término “desbalanceado” hace referencia a que existe una clase que presenta una gran cantidad de registros (clase mayoritaria), en términos relativos, respecto a la clase con menos registros (clase minoritaria). *Fig. 8*.

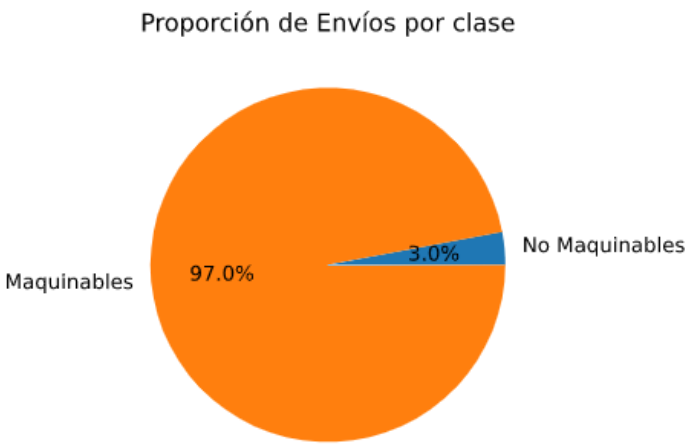


Figura 8: gráfico de torta que muestra la proporción de envíos de productos No Maquinables y Maquinables.

Siguiendo con el análisis de los envíos, se pudo detectar que alrededor del 5% de los títulos de los productos están duplicados (*Fig. 9*). Cabe destacar que dentro de la plataforma conviven diferentes publicaciones. Existen las publicaciones correspondientes a los productos llamados "productizados". Estos productos tienen la característica de tener exactamente la misma descripción estandarizada a pesar de que pueda haber más de un

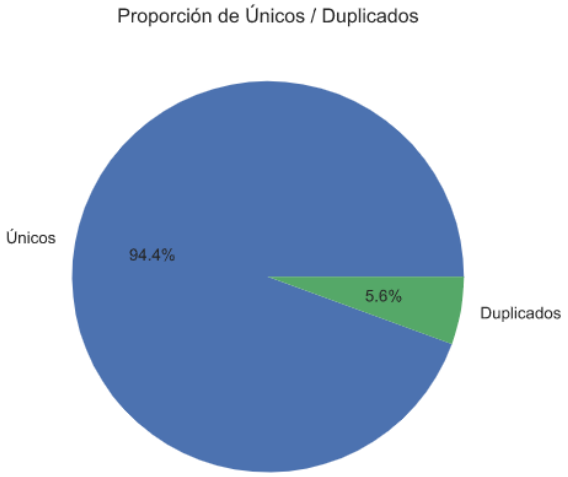


Figura 9: gráfico de torta que muestra la proporción de títulos únicos y duplicados productos enviados.

vendedor que publique dicho ítem. Por otro lado, también puede darse la posibilidad de que las publicaciones de diferentes vendedores contengan exactamente la misma descripción, aunque es menos probable que suceda. Debido a esto, vemos que los datos de entrenamiento representan un conjunto de envíos asociados a productos y no un conjunto solo de productos. Los productos y descriptores duplicados podrían generar ruido e inducir al modelo de manera errónea.

Un aspecto importante a tener en cuenta es la manera en la que están conformados los descriptores, en especial los títulos que representan al descriptor principal. Se puede ver que la conformación de los títulos de los productos es heterogénea. Según sea el caso, pueden estar conformados por texto, números y diferentes tipos de caracteres. Ningún título presenta una construcción igual.

Como se puede visualizar en la *Fig. 10*, vemos que la mayoría de los dominios contienen productos que presentan títulos que incluyen caracteres numéricos. Pueden estar identificando un modelo, una medida, o alguna referencia muy específica del producto. Si bien estos números son parte del descriptor, el objetivo no es que el modelo aprenda referencias específicas, si no, la relación entre palabras que permita identificar el tipo de producto. En este caso, el aprendizaje del modelo podría verse afectado.

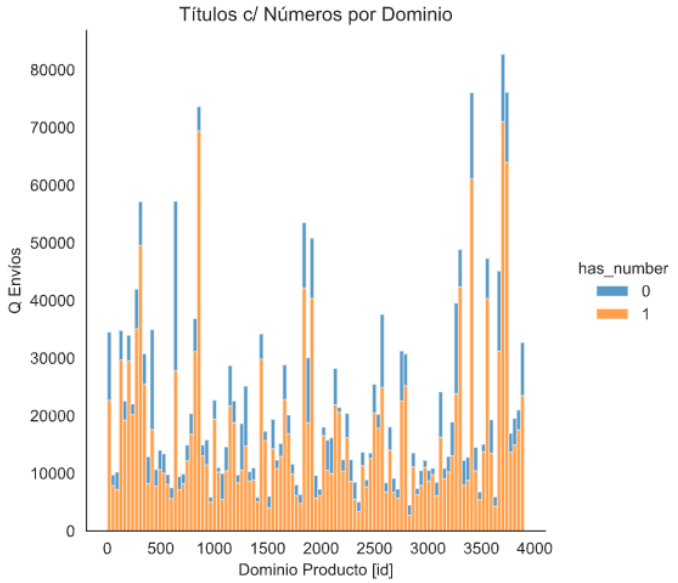


Figura 10: gráfico que muestra la distribución de los productos enviados que tienen títulos que contienen números y los que no. El color naranja indica que tiene números y el color azul indica que no.

De manera similar, en la *Fig. 11*, vemos que la mayoría de los dominios contienen productos que presentan títulos con caracteres especiales. Cuando hablamos de caracteres especiales nos referimos a cualquier carácter que no sea alfabético o numérico. Por ejemplo, @! /$”·%=) entre otros. Estos tipos de caracteres pueden afectar el rendimiento del modelo ya que incorporan información que no tiene valor descrito y ensucia el aprendizaje.

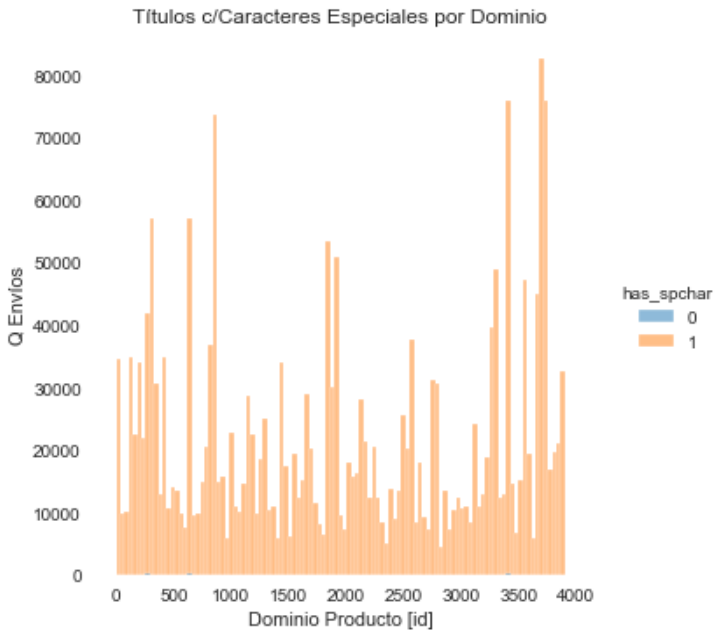


Figura 11: gráfico que muestra la distribución de los productos enviados que tienen títulos que contienen caracteres especiales y los que no. El color naranja indica que tiene números y el color azul indica que no.

Por otro lado, nos podemos encontrar con caracteres individuales, es decir, que dentro del descriptor aparecen solos. Por lo general, son conectores textuales alfabéticos que ayudan a conformar una oración. Como se puede ver en la *Fig. 12*, la mayoría de los dominios contienen productos con títulos que presentan esta característica. Este caso es similar al caso de los caracteres especiales que se mencionó antes. Más adelante, se explicarán las técnicas que nos permitirán lidiar con estas situaciones y mejorar la composición de los descriptores.

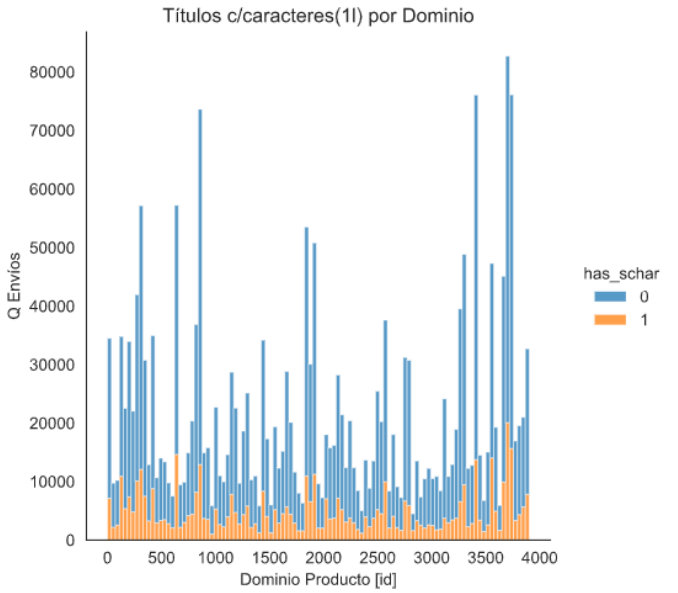


Figura 12: gráfico que muestra la distribución de los productos enviados que tienen títulos que contienen caracteres especiales y los que no. El color naranja indica que tiene números y el color azul indica que no.

Continuando con el análisis, a través de los datos se pudo observar que no hay indicios de que un grupo reducido de vendedores sea responsable de la mayoría de la venta y envíos de los productos No Maquinables.
Como se puede ver en las *Fig. 13 y 14*, se visualiza la cantidad de envíos realizados por cada vendedor. Cada punto del eje horizontal representa a un vendedor único dentro del conjunto de datos. En el eje vertical, se puede ver la cantidad de productos enviados que corresponden a cada vendedor. Los vendedores fueron ordenados de mayor a menor en función de la cantidad de productos enviada en dicha clase.
Respecto a los vendedores de los productos maquinables, como se puede ver en el gráfico, la mayoría de las ventas están concentradas en el lado izquierdo; un grupo determinado de vendedores tienen la mayoría de las ventas. No obstante, para esta muestra, el número de vendedores asciende a miles de vendedores. Respecto a los vendedores de los productos no maquinables, si bien hay algunos picos, la mayoría de las ventas están distribuidas en todo el espectro de vendedores. Cabe destacar que no es factible asegurar que las distribuciones de los productos se mantengan estables si se analiza otra muestra o el escenario real del dominio.
Ya sea para los productos maquinables como para los productos no maquinables, hay evidencia suficiente que indica que los productos no son manejados por un grupo reducido de vendedores. No obstante, es una variable que podría aportar valor dado el supuesto que un vendedor suele publicar cosas sobre algunos tipos de productos relacionados.

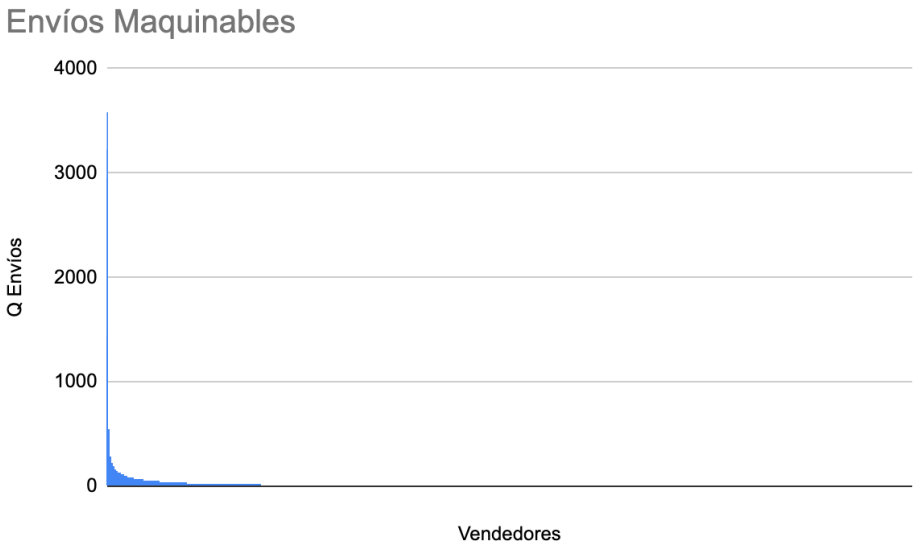
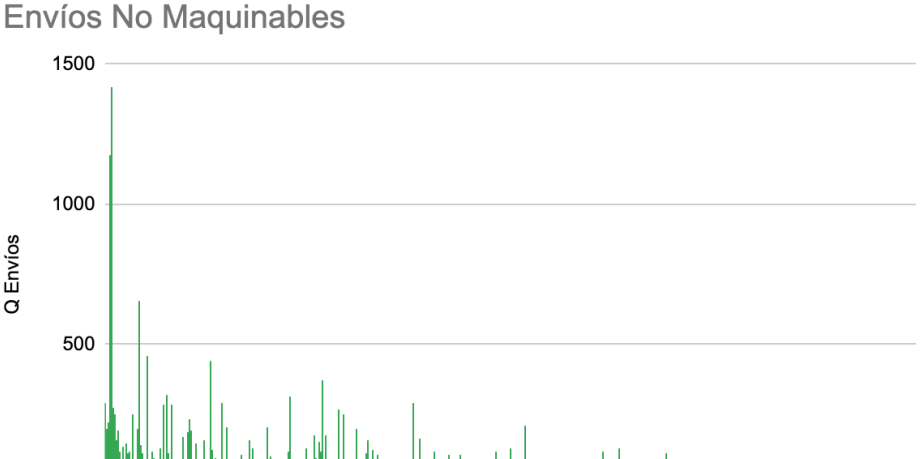


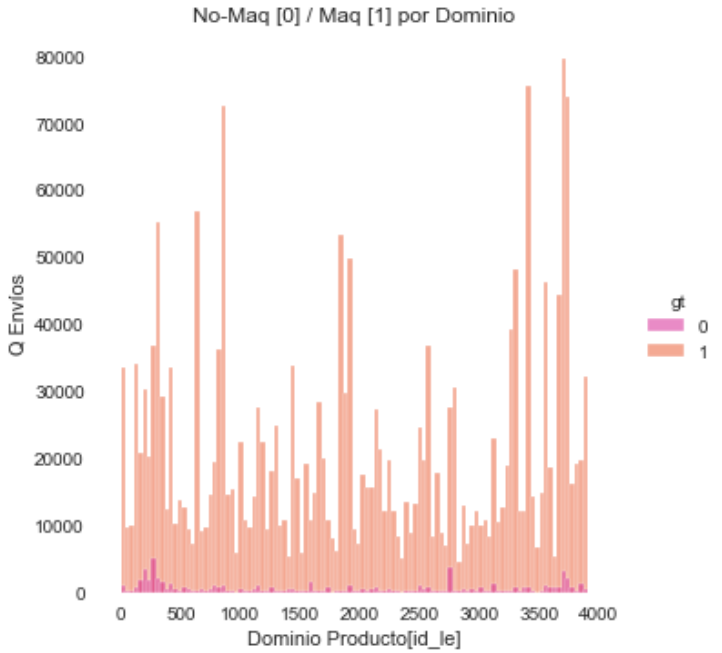
Figura 13: gráfico que muestra la distribución de los productos enviados por cada uno de los vendedores. Se puede ver la segmentación por productos maquinables.



Realizando un análisis sobre los dominios, tanto los ítems Maquinables y No Maquinables están distribuidos entre los 3903 únicos dominios sin marcar una tendencia en particular. Sin embargo, cabe destacar que hay 737 (19%) dominios que concentran alrededor del 80% de los productos. *Fig. 15*.

Figura 14: gráfico que muestra la distribución de los productos enviados por cada uno de los vendedores. Se puede ver la segmentación por productos no maquinables.

Figura 15: gráfico que muestra la distribución de los productos enviados por el tipo de dominio a la cual pertenecen. El color naranja indica que se trata de un producto maquinable y el color lila indica que no.



De la misma manera, se puede mencionar algo similar respecto a las categorías, cuyos descriptores son de un nivel más bajo (más detallado) respecto a los dominios. En este caso, de las 5637 categorías únicas, hay 836 (14%) categorías que concentran aproximadamente el 80% de los productos. *Fig. 16*.

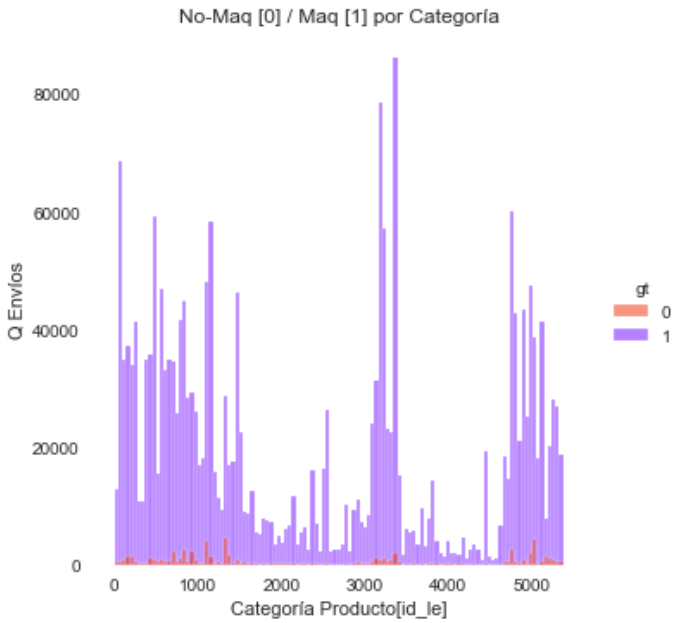


Figura 16: gráfico que muestra la distribución de los productos enviados por el tipo de categoría a la cual pertenecen. El color naranja indica que se trata de un producto maquinable y el color lila indica que no.

De manera intuitiva, se podría decir que tanto la variable domian_id y category_id podrían aportar información para predecir las clases de los productos.

Otra de las variables que presentan los productos es la condición. Esto hace referencia a si el producto es nuevo, usado o no especificado. Cómo se puede ver en el gráfico (*Fig. 17)*, tanto para los ítems maquinables como para los no maquinables, la mayoría (99%) presenta la condición de "nuevo". Dado el gran desbalance que se presenta en estas clases, no hay indicios fuertes de que esta variable podría aportar información valiosa.

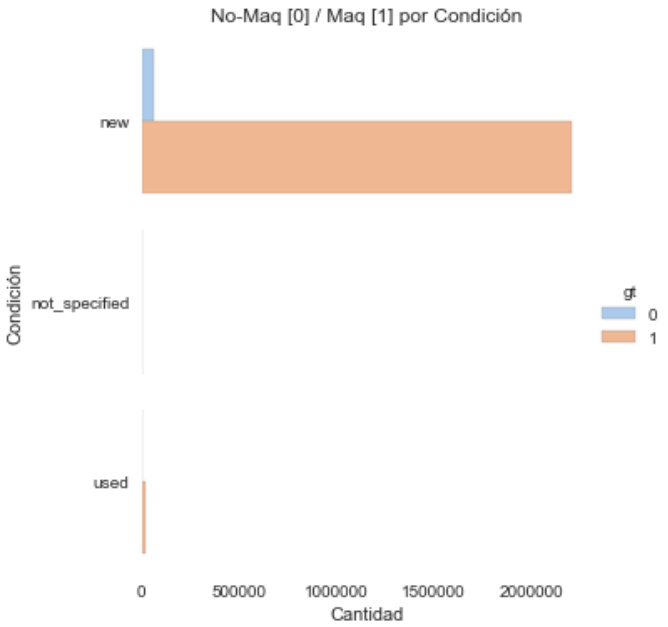
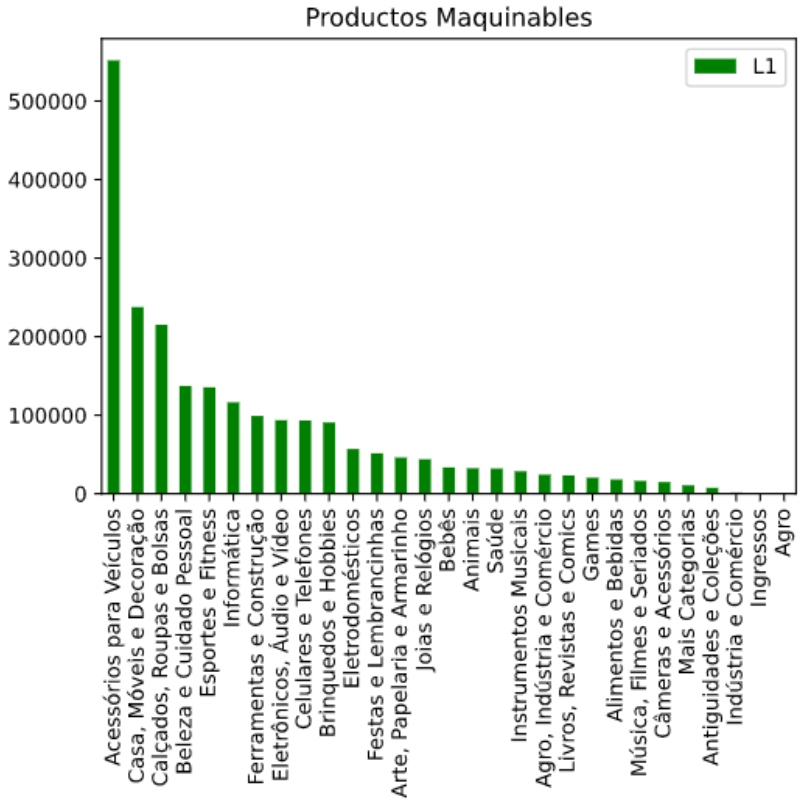


Figura 17: gráfico que muestra la distribución de los productos enviados por el tipo de condición que presentan en la publicación. El color marrón indica que es un producto maquinable y el color azul que no.

A nivel categoría L1, el 80% de los envíos está cubierto por 11 (39%) categorías de las 65 existentes para esta muestra. Se puede ver que ambas clases comparten categorías representativas (*fig. 18 y 19*). Para ambas etiquetas, las categorías de "Accesorios para Vehículos" y "Casas, Decoración", "Deportes & Fitness", entre otras, son las más representativas.

Como se puede ver, este tipo de atributos, podrían ser descriptores complementarios al título que brinden más información al momento de tener que realizar una predicción. De manera intuitiva, se podría decir que hay diferentes categorías que pueden representar un grupo de productos que difícilmente puedan ser clasificados como maquinables. Como, por ejemplo, si bien podría existir algún envío no maquinable dentro de la categoría "Celulares", sería extraño que esto ocurriese con mucha frecuencia. Como se puede ver en las visualizaciones, "Celulares" es una categoría bastante importante dentro de la clase maquinables, no así en la clase no maquinable.



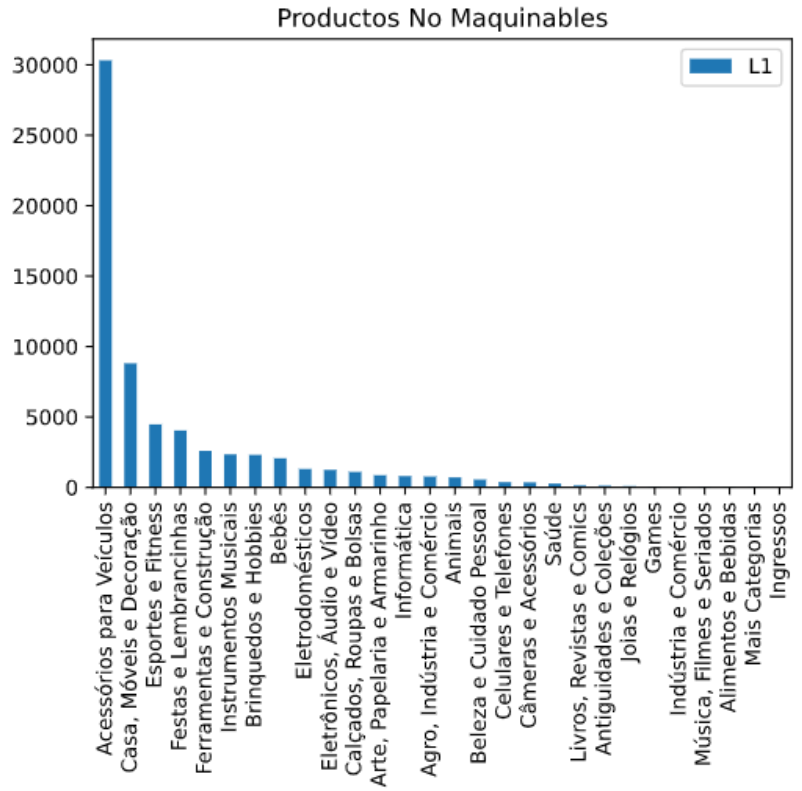


Figura 18 y 19: gráfico que muestra la distribución de los productos enviados de la categoría L1. Se puede ver la segmentación por productos maquinables y no maquinables.

A nivel categoría L2, una categoría de nivel más bajo respecto a L1, cubre el 80% de los envíos con 65 (21%) categorías de las 303 existentes para esta muestra. Se tomaron las 20 categorías relevantes para analizar. *Fig 20 y 21*.

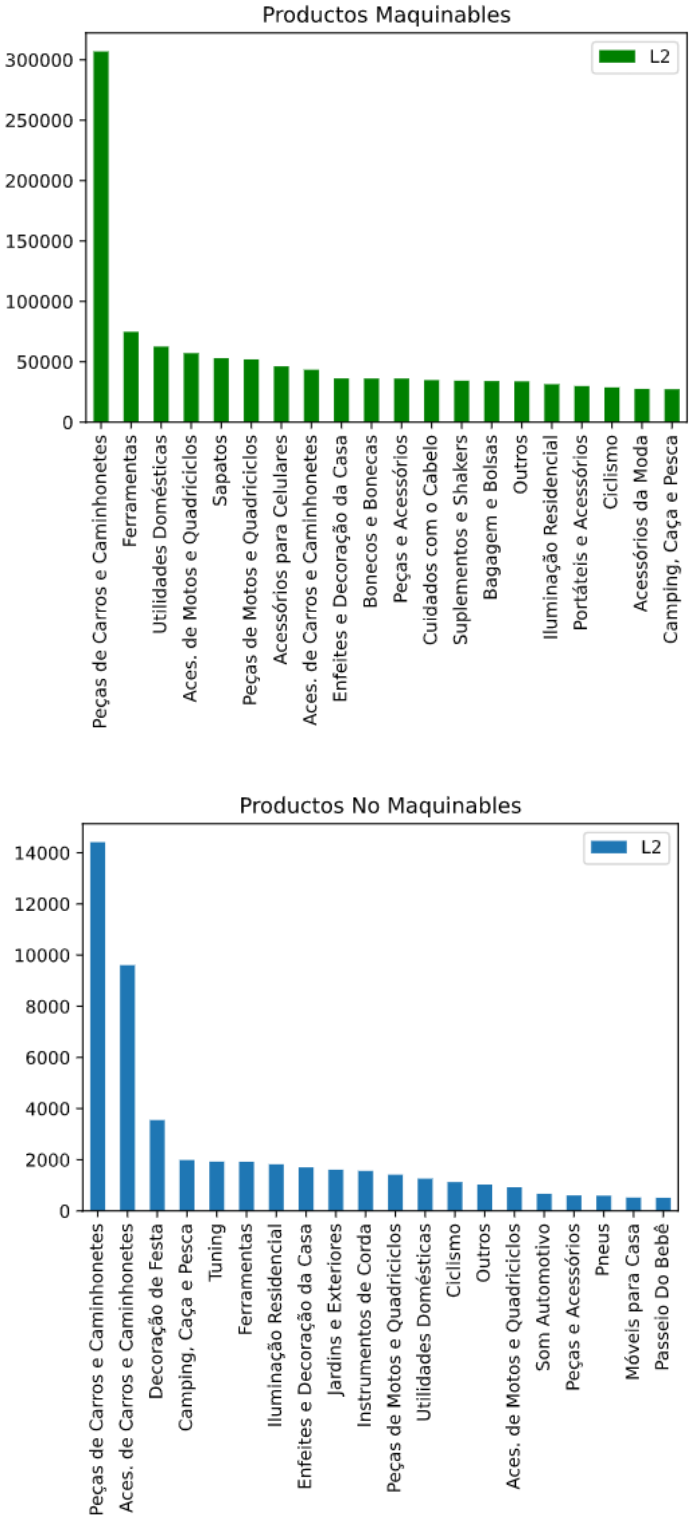


Figura 20 y 21: gráfico que muestra la distribución de los productos enviados de la categoría L2. Se puede ver la segmentación por productos maquinables y no maquinables.

Tal como se puede observar, hay una relación muy fuerte entre las categorías L1 y L2. Es obvio que este hecho suceda debido a que una categoría de predecesora siempre tendrá dependencia con su antecesora. De todas maneras, habría que experimentar para averiguar qué información podría aportar las diferentes categorías a la predicción de la clase de un producto.

- Con mayor granularidad, la categoría L3 cubre el 80% de los envíos con 197 (13%) categorías de las 1511 existentes.
- Otras variables que contienen texto, como "parent_product_id", "path_from_root", "main_picture", entre otras; no presenta información relevante que nos ayude identificar las etiquetas deseadas.

En el plan de acción de calidad de los datos se presentará el diagnóstico y las acciones a tomar según corresponda. Los detalles se encuentran en el Anexo B: Calidad de los Datos.

A continuación, se hará un resumen de las acciones principales realizadas que corresponden a la etapa de la Preparación de los Datos.

2.3. Limpieza

Con el objetivo de obtener el conjunto de datos listo para experimentar, es necesario realizar una limpieza[15] de este. Se ejecutaron las acciones detalladas en el Anexo B: Calidad de los Datos. Aquí, se resumen los puntos principales.

Se identificó un conjunto de variables que, de acuerdo con el entendimiento de la problemática, no contenían datos relevantes que puedan aportar en el desarrollo una solución predictiva. Como buena práctica, se determinó la eliminación de estas. Ejemplos, "Unnamed", "shp_service_id", "shp_quantity", "shp_shipment_id", entre otras. A su vez, esto fue consulta con el equipo de Envíos de la empresa.

En algunos casos, se decide eliminar los registros con valores nulos dado que, para ciertas variables, como por ejemplo "title", el cual se considera el descriptor más importante, podría entorpecer el entrenamiento del modelo. Claramente este valor nulo se da debido a un error en la recolección de los datos ya que no es posible que un producto no tenga título en una publicación activa. En otros casos, el hecho de que un haya un valor nulo en una variable, brinda cierta información, podría descubrirse un patrón que pueda ayudar a predecir alguna de las clases. La manera de hacerlo es crear una variable binaria que identifique ese hecho. Esto sucede con la variable "L3" por ejemplo.

Según lo analizado, el conjunto de datos representa un conjunto de envíos y no de productos. Esto implica que cierta cantidad de productos están repetidos y deben ser eliminados con el objetivo de que el modelo no incorpore a su aprendizaje patrones erróneos. En este caso, podría ponderar más los productos maquinables respecto a los no maquinables y no es lo deseado. De esta manera, el conjunto de datos podrá representar una muestra del universo de productos en vez de envíos.

Como resultado final del entendimiento y de la limpieza del conjunto de datos inicial, obtenemos finalmente una muestra que tiene las variables "gt" (etiqueta del ítem que identifica si es maquinable o no), "title" (título), "domain_id" (dominio del ítem),

“category_id” (categoría del ítem), “seller_id” (vendedor), “brand_name” (marca), “L1” (categoría nivel 1), “L2” (categoría nivel 2), “L3” (categoría nivel 3), “price” (precio) y “shp_weight” (peso del envío). Por otra parte, la cantidad final de registros asciende a 2.164.039 (94% del total inicial). La limpieza propuesta en el plan de calidad de datos asegura que el conjunto de datos represente a los artículos comercializados dentro del marketplace en vez de los envíos realizados. Debajo aparece un ejemplo del conjunto de datos que ha quedado definido y depurado. *Fig. 22*.

```
ds[['gt','title','domain_id','category_id','seller_id','brand_name','L1','L2','L3','price','shp_weight']].head(3)
```

	gt	title	domain_id	category_id	seller_id	brand_name	L1	L2	L3	price	shp_weight
0	1	Bau Tork 45 Litros + Bagageiro Titan Fan 125 2...	MLB-VEHICLE_ACCESSORIES	MLB3936	234369230	TORK/POLIMET	Acessórios para Veículos	Aces. de Motos e Quadriciclos	Outros	289.90	5870.0
1	1	Protetor De Sofá Mariana 1 Lugar Dupla-face Po...	MLB-SOFA_AND_FUTON_COVERS	MLB186804	32389645	None	Casa, Móveis e Decoração	Têxteis de Casa e Decoração	Capas	41.20	460.0
2	1	Tablet Powerpack Tela 10.1'' Ips Hd Android 7....	MLB-TABLETS	MLB99889	232562299	Powerpack	Informática	Tablets e Acessórios	Tablets	494.15	917.0

Figura 22: ejemplo de la tabla de datos que representa una muestra del conjunto de datos que será utilizado para el proyecto.

A continuación, se explicará la transformación previa y necesaria para que los datos puedan ser utilizados para los diferentes experimentos a realizar.

2.4. PreProcesamiento

Antes de que un programa de computadora pueda comprender e interpretar el lenguaje humano, es necesario realizar el preprocesamiento[16] de los datos para que estos puedan ser consumidos por el modelo. Gracias al preprocesamiento, los programas pueden transformar el lenguaje o el texto en una forma más asimilable para que los algoritmos de aprendizaje automático puedan obtener mejores resultados. Por lo general, los datos de entrada se presentan en una forma natural, que está en el formato de texto, oraciones, comentarios, párrafos, tweets, etc. Se necesita realizar un procesamiento previo para que los modelos puedan concentrarse en las palabras principales que representan el vocabulario del texto[17] analizado en lugar de todo el resto de las palabras que solo agregarían poco valor o nulo.

Ya teniendo el conjunto de datos conformado, es necesario realizar las acciones antes mencionadas y normalizar las variables textuales. Dado que todos los descriptores de un artículo están conformados por texto, cabe destacar que, uno de los mayores desafíos del modelado se presenta al descubrir la concatenación de atributos textuales como ser, título, dominio y categoría entre otros, que permita la mejor descripción de un ítem y que sea aprendible por el modelo. Al mismo tiempo, debido a la gran cantidad de signos de puntuación, conectores, caracteres especiales y recursos textuales, será necesario trabajar en la transformación de los datos para obtener la información lo mejor posible.

En esta sección, se hablará sobre la limpieza de los descriptores de los productos ya que la mayoría de estos contienen generadores de ruido. La mayoría de los textos y los documentos contienen muchas palabras que son redundantes para la clasificación del texto, como palabras vacías (artículos, pronombres, preposiciones, etc.), errores ortográficos, conectores, jergas, etc. En muchos algoritmos, como los métodos de aprendizaje estadístico y probabilístico, el ruido y ciertos componentes innecesarios pueden afectar negativamente el rendimiento general. Por lo tanto, la eliminación de estos generadores de ruido es muy importante.

2.4.1. Principales Generadores de Ruido

Stop Words

La clasificación de textos y documentos generalmente se ve afectada por la naturaleza ruidosa (abreviaturas, formas irregulares) que contienen los corpus de texto. La razón por la que las stop words[18] o palabras vacías representan poco valor para muchas aplicaciones del procesamiento del lenguaje natural es que, si eliminamos las palabras que se usan con mucha frecuencia, podemos concentrarnos en las palabras que son importantes.

Algunos ejemplos de Stop Word son: ante, antes, aún, aunque, aquí, arriba, atrás así, bajo, bastante, cabe, conmigo, bien, casi, cierto, como, debajo, ahí, ajeno, algo, algún, ambos, aquello. En muchos casos, son conjunciones, preposiciones, adverbios y artículos. No obstante, esto también depende del dominio, de la naturaleza de la problemática con la que se esté trabajando. Por ejemplo, en nuestro caso, podemos observar una visualización que refleja la frecuencia de aparición de ciertas palabras en los títulos de los ítems enviados. *Fig. 23*.

Palabras como “de”, “original”, entre otras, podrán ser palabras que presentan una alta frecuencia pero que es muy posible que no tengan gran importancia para realizar la predicción de un producto.

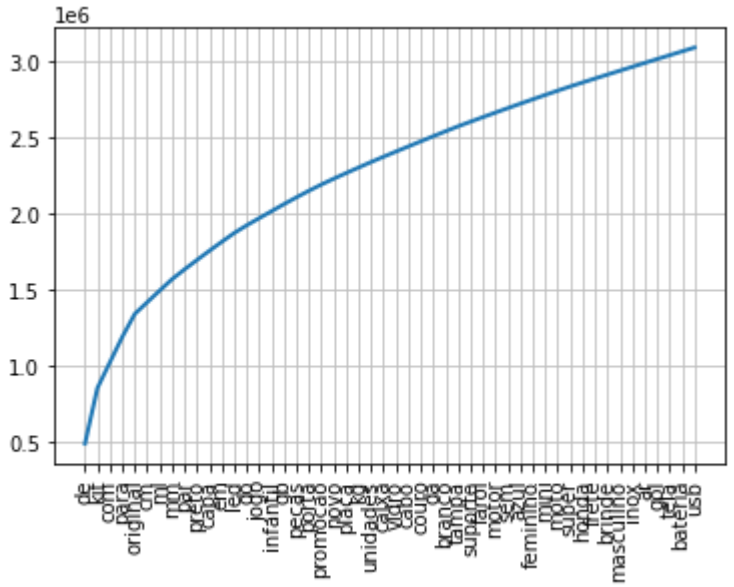


Figura 23: gráfico de distribución acumulada de las palabras que aparecen dentro de una muestra del conjunto de datos de títulos de los productos maquinables y no maquinables.

Caracteres acentuados

Los caracteres acentuados son elementos importantes que se utilizan para enfatizar una palabra, en particular durante la pronunciación o la comprensión de esta. En algunos casos, la marca del acento también aclara el significado de una palabra, que podría ser diferente sin este no estuviese. Si bien su uso puede ser limitado, dependiendo del idioma con el cual se esté trabajando, existe la posibilidad que se encuentren caracteres o letras acentuados en un corpus de texto. El manejo de caracteres acentuados se vuelve más importante, especialmente si solo desea analizar el idioma inglés. Por lo tanto, debemos asegurarnos de que estos caracteres se conviertan y se estandaricen en caracteres pertenecientes al código ASCII.

Capitalización

Los títulos pueden contener una combinación de letras mayúsculas y minúsculas. Como vimos antes, la gran mayoría de los títulos de los artículos están compuestos con palabras mayúsculas y minúsculas. Si una misma palabra aparece escrita de distinta manera, el programa reconoce cada palabra como única y por ende, identifica todas las variantes. Para reducir el espacio del problema, el enfoque más común es reducir todo a minúsculas. Esto coloca todas las palabras de un documento en el mismo espacio. Las palabras que comienzan con mayúsculas son, en su mayoría, nombres propios y su aparición puede generar un significado diferente dentro de una sentencia textual. En ocasiones, pueden ser no deseadas.

Contracciones

Las contracciones son versiones abreviadas de palabras o sílabas. Se crean eliminando una o más letras específicas de las palabras. A menudo, se combinan más de una palabra para crear una contracción. Al escribir, se usa un apóstrofe para indicar el lugar de las letras que faltan. En el idioma / texto inglés, las contracciones a menudo existen en forma escrita o hablada. Hoy en día, muchos editores inducen contracciones de forma predeterminada. Por ejemplo, “do not” a “don’t”, “I would a “I’d”, “you are” a “you’re”. En nuestro caso particular las contracciones están referidas a cosas que los artículos pueden contener o no. Por ejemplo, “con algo” a “c/algo”, “sin algo” a “s/algo”. Convertir cada contracción a su forma original expandida ayuda con la estandarización del texto; también se debe analizar qué sucede con la eliminación de estas, en muchos casos esto puede ser de mayor ayuda respecto a la expansión de la contracción.

Abreviaturas

Las jergas y las abreviaturas pueden causar ruido en el entrenamiento de un modelo Una abreviatura es una forma abreviada de una palabra. Por ejemplo, MVS significa máquina de vector soporte, en el contexto de aprendizaje automático. El método común para tratar estas palabras es convertirlas a un lenguaje formal o eliminarlas en caso de que, según el dominio del problema, no aporten valor agregado.

Caracteres especiales

Los caracteres especiales, como se sabe, son caracteres no alfanuméricos. Estos caracteres se encuentran con mayor frecuencia en comentarios, referencias, títulos, descripciones,

etc. Estos caracteres, por lo general, no agregan valor a la comprensión del texto e inducen ruido en los algoritmos.

Números

Los números pueden no tener importancia como también tenerla, depende mucho del caso de uso y de la naturaleza del texto relacionado. Si el número asociado al texto hace referencia a la dimensión del artículo, podría tener sentido conservar el número y transformarlo en caso de que sea necesario. En otros casos, el número podría ser simplemente parte de la descripción y no agregar mucha información a la comprensión del texto. Entonces, dependiendo de la acción que se quiera tomar, los números se pueden eliminar del texto o pueden ser conservados. Podemos usar expresiones regulares (RE) para eliminar o dejar los números donde se identifique un patrón determinado y realizar de manera programática.

Habiendo explicado los principales generadores de ruidos, a continuación, se hablará de las expresiones regulares; son de los recursos más utilizados para la limpieza y normalización de textos o documentos que resolverán los problemas descritos anteriormente y que permitirán obtener un conjunto de datos para ser utilizado por los modelos de aprendizaje automático.

2.4.2. Expresiones Regulares

Las expresiones regulares[19] (RE) conforman un lenguaje para especificar cadenas de búsqueda de texto. Este lenguaje práctico se utiliza en todos los lenguajes informáticos, procesadores de texto y herramientas de procesamiento de texto. Formalmente, una expresión regular es una notación algebraica para caracterizar un conjunto de tipos de cadenas de caracteres. Son particularmente útiles para la búsqueda en textos, cuando tenemos un patrón que buscar y un corpus de textos en donde buscar. Una función de búsqueda de expresiones regulares buscará en el corpus y devolverá todos los textos que coincidan con el patrón. El corpus puede ser un solo documento o una colección de textos.

Se puede diseñar un patrón de búsqueda para devolver todas las coincidencias. En los siguientes ejemplos, se resaltará el patrón que coincide con la expresión regular. Se mostrarán expresiones regulares delimitadas por barras, pero hay que tener en cuenta que las barras no son parte de las expresiones regulares. Las expresiones regulares presentan muchas variantes.

Ejemplos simples de RE

RE	Coincidencia	Sentencia
/[A-Z]/	una letra mayúscula	él se llama **P**edro
/[0-9]/	un dígito	Capítulo **1**: Introducción a

Las expresiones regulares pueden ser combinadas para describir los patrones deseados de búsqueda. Como parte del código elaborado para la ejecución del preprocesamiento de los descriptores que conforman el conjunto de datos de los productos, se desarrollaron las

funciones “normalize_text()”, “normalize_domain()”, “transform_categories_to_text()”. Todas las funciones, toman como entrada una sentencia textual y como salida, se obtiene el mismo texto pero procesado, limpio y normalizado. Las diferentes funciones de preprocesamiento son utilizadas de acuerdo con el tipo de descriptor ya que todas generan resultados distintos. En muchos casos, son ejecutadas secuencialmente. Un ejemplo del resultado se puede ver a continuación.

Habiendo realizado el preprocesamiento, es necesario realizar la separación del conjunto de datos para desarrollar el modelo y evaluar los experimentos.

2.5. Separación del Conjunto de Datos

Una vez que todos los descriptores fueron preprocesados, es momento de hacer la separación del conjunto de datos. Una parte, es utilizada para permitirle al modelo aprender de los datos y la otra parte del conjunto, es utilizada para la fase de evaluación del modelo.
La división de los datos en conjuntos de entrenamiento y test es una técnica para evaluar el rendimiento de un modelo de aprendizaje automático, se puede utilizar para problemas de clasificación (como nuestro caso) o regresión y, por ende, esto se extiende para cualquier algoritmo de aprendizaje supervisado.
Si bien existen variantes para ejecutar el procedimiento, una de las más utilizadas implica tomar un conjunto de datos y dividirlo en dos subconjuntos. El primero, es utilizado para el aprendizaje del modelo y se denomina conjunto de datos de entrenamiento; el segundo, se utiliza para evaluar la calidad predictiva del modelo; de hecho, son datos que el modelo no debe “ver” en lo absoluto. Este se denomina conjunto de datos de test y es utilizado para determinar el rendimiento del modelo en un escenario lo más real posible. El objetivo de este paso es permitirle al modelo poder aprender con un conjunto de datos disponibles y con resultados conocidos; posteriormente, en la instancia de evaluación (algo en lo cual profundizaremos luego), los datos de entrada no conocidos son proporcionados al modelo para luego realizar las predicciones y como paso final, analizar los resultados y el comportamiento obtenido. El objetivo es cuantificar, con alguna medida de error, cuántos aciertos se tuvieron al realizar las predicciones y cuántos no. Esto se puede ver representado en la *Fig 24*.

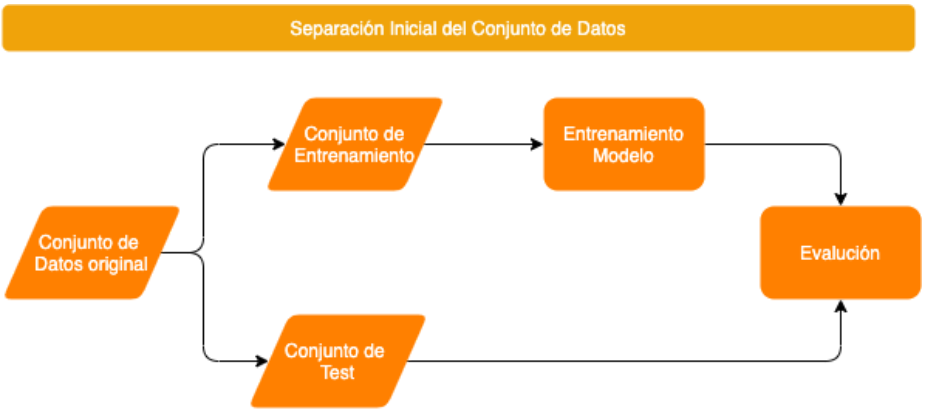


Figura 24: esquema del proceso de separación del conjunto de datos.

El procedimiento de separación de datos de entrenamiento y test es apropiado cuando hay un conjunto de datos lo suficientemente grande. La idea de "suficientemente grande" es específica para cada problemática y modelado asociado. Por otro lado, significa que debe haber suficientes datos para dividir el conjunto inicial en subconjuntos de entrenamiento y test. Cabe destacar que, cada uno de los subconjuntos de datos deben ser representaciones adecuadas del dominio del problema; esto requiere que el conjunto de datos original también sea una representación adecuada de dicho dominio. Esto implica que debe haber suficientes registros para cubrir todos los casos de uso comunes y la mayoría de los casos poco comunes dentro del dominio en cuestión.
Por el contrario, el procedimiento de separación de datos de entrenamiento y test no es apropiado cuando el conjunto de datos disponible es pequeño. La razón es que cuando el conjunto de datos se divide en los subconjuntos antes mencionados, no habrá suficientes datos de entrenamiento para que el modelo aprenda de manera eficiente y tampoco habrá suficientes datos de test para evaluar eficazmente el rendimiento del modelo. Los resultados estimados podrían ser poco confiables, en algunos casos demasiado optimistas, en otros demasiado pesimistas y solo podría generar mayor incertidumbre.

En nuestro caso, se decidió experimentar dos estrategias distintas. La primera, fue realizar una separación tradicional aleatoria y estratificada por la proporción de clases en el conjunto de datos. Esto se realizó para garantizar que los conjuntos de datos de entrenamiento y de test sean representativos del conjunto de datos original. Además, debido a que el conjunto de datos original está desbalanceado, como se vio en el análisis exploratorio, es factible configurar un parámetro que permite realizar una segmentación que respete la distribución de los datos. Es importante que tanto el subconjunto de entrenamiento como el de test, tengan la misma distribución y que reflejen la representación deseada.

Por otro lado, la segunda estrategia de separación fue generar un conjunto de entrenamiento con un cierto grupo de vendedores y un conjunto de test donde los vendedores de este último no aparezcan en el conjunto de entrenamiento. Si bien no se puede asegurar que la distribución sea totalmente representativa del conjunto original, esto se realizó dado que el escenario de predicción real se asemeja más a esta situación. Es decir, nuevos vendedores e ítems podrían ir apareciendo y estos no serían conocidos por el modelo hasta realizar un nuevo entrenamiento. De esta manera, también se previene de cualquier fuente de leakage.

Dichas estrategias son denominadas Train 1 y Train 2 respectivamente. Estas referencias son utilizadas en la sección 4 de experimentos.

Finalmente, nuestros conjuntos de datos son separados con las siguientes proporciones: 85% entrenamiento, 15% test.

3. Métodos y Procedimientos

En esta sección se dará detalle de los diferentes enfoques, técnicas, métodos y procedimientos utilizados durante el entrenamiento del modelo y el desarrollo de los experimentos.

3.1. Aprendizaje Supervisado

El aprendizaje supervisado[20] utiliza un enfoque para generar un programa que puede hacer predicciones a partir de datos conocidos. En vez de desarrollar un programa a mano, en función de requerimientos para realizar una determinada tarea; se aprende a partir de ejemplos pasados. Este proceso funciona si tenemos instancias de datos para las que sabemos exactamente cuál habría sido la predicción correcta.
Los algoritmos de aprendizaje automático supervisados están diseñados para aprender mediante ejemplos; es el proceso de enseñarle a un modelo a través de datos de entrada y con datos de salida correctos y conocidos. El nombre de aprendizaje "supervisado" se puede comparar con la idea de que entrenar este tipo de algoritmos es como tener un profesor supervisando todo el proceso de aprendizaje. Si se estuviese aprendiendo una tarea bajo supervisión, alguien estaría juzgando si está obteniendo la respuesta correcta o no. De manera similar, en el aprendizaje supervisado, eso significa tener un conjunto completo de datos etiquetados mientras se entrena un algoritmo. Completamente etiquetado implica que cada ejemplo en el conjunto de datos de entrenamiento está etiquetado con la respuesta que el algoritmo debería generar por sí solo.

Por mencionar un ejemplo, un modelo clasificador podría aprender a distinguir flores (rosas, agapantos y margaritas) a través de un conjunto de datos etiquetado que contiene registros de distintas características de las flores y su clasificación correcta (etiqueta – ground truth). El modelo podría aprender las relaciones existentes entre distintas variables predictoras y la variable target, como por ejemplo, las dimensiones y/o características de los pétalos. Luego de que el modelo haya aprendido dichas relaciones a través de los datos existentes, utilizará estas para intentar predecir la etiqueta de un nuevo registro (no conocido por el modelo) al cual no se le conoce la etiqueta y le asignará un score o probabilidad. La etiqueta que tenga mayor score o probabilidad será la seleccionada como predicción. *Fig. 25*.

El aprendizaje supervisado es un subconjunto del universo del aprendizaje automático en el que el proceso de aprendizaje está sujeto a una variable dependiente (variable a predecir), a diferencia del aprendizaje no supervisado, en el que los datos no tienen etiquetas y, por lo tanto, los algoritmos solo buscan cualquier patrón que puedan encontrar (mayormente utilizado para realizar análisis de datos y para encontrar grupos o clusters). En este sentido, el aprendizaje supervisado aparece como una oportunidad para abordar los experimentos que serán propuestos más adelante.

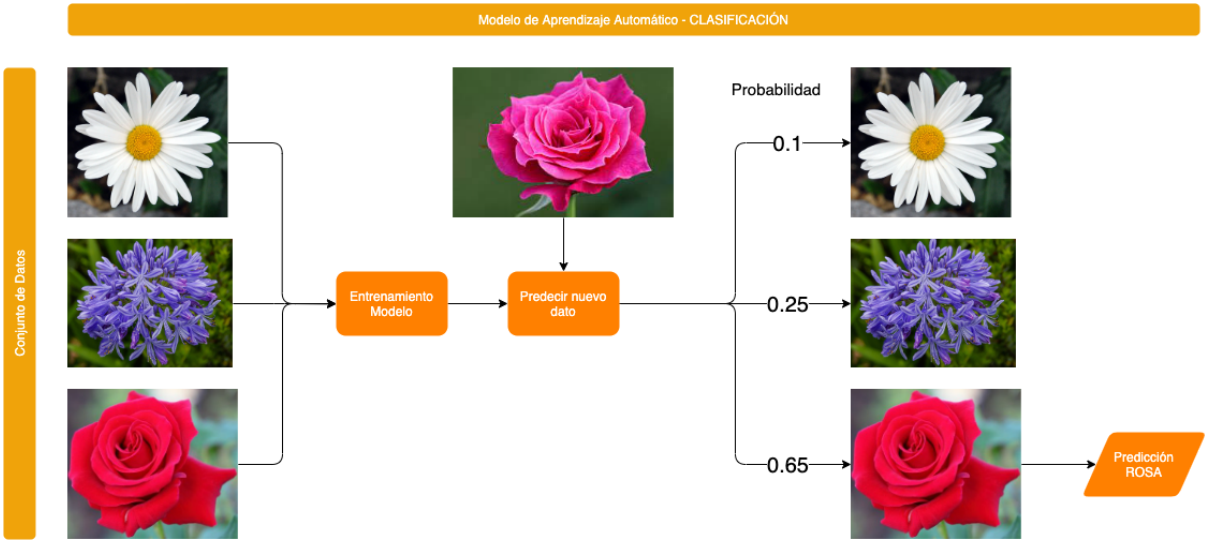


Figura 25: esquema representativo y simplificado del proceso de entrenamiento de un modelo clasificador.

El aprendizaje supervisado puede ofrecer muchas ventajas, como brindar la posibilidad de explorar grandes cantidades de datos y variables, obteniendo descubrimientos relevantes y la posibilidad de automatizar tareas. Sin embargo, existen algunos desafíos al crear modelos sostenibles de aprendizaje supervisado. Los siguientes puntos son algunos de estos:

- Los modelos de aprendizaje supervisado requieren ciertos niveles de conocimiento y experiencia para estructurarse y desarrollarse con precisión.
- El entrenamiento de los modelos de aprendizaje supervisado puede requerir mucho tiempo de cómputo.
- Los conjuntos de datos pueden tener una mayor probabilidad de error humano, lo que hace que los algoritmos aprendan incorrectamente.
- A diferencia del aprendizaje no supervisado, los algoritmos de aprendizaje supervisado requieren si o si de una etiqueta correcta y conocida (ground truth) en el caso de clasificación o de un valor numérico en el caso de un problema de regresión.

En nuestro caso, es factible realizar el etiquetado de los datos de una manera muy simple. Para datos históricos, se tienen las mediciones realizadas por los operadores logísticos. Por ende, se tiene la dimensión de cada lado de un paquete enviado. De esta manera, es factible poder identificar los artículos en función de sus dimensiones.

3.2. Clasificación

Durante el entrenamiento, el algoritmo buscará patrones en los datos que se correlacionen con los resultados deseados. Después del entrenamiento, un algoritmo de clasificación tomará nuevos datos de entradas no conocidos y determinará en qué etiqueta serán clasificados según los datos de entrenamiento anteriores. Tal como se vio en el ejemplo de la sección anterior. La clasificación se utiliza para predecir una clase o etiqueta discreta (Y).

3.2.1. Tipos de Clasificación

Según lo mencionado anteriormente, la clasificación es una subcategoría del aprendizaje supervisado en la que el objetivo es predecir las etiquetas de clase categóricas (discreta, valores no ordenados, pertenencia a grupos) de los nuevos datos u observaciones, basándonos en ejemplos pasados.

Cabe destacar que existen dos tipos principales de clasificaciones[21]:

Clasificación Binaria: es el tipo de clasificación en el que tan solo se pueden asignar dos clases diferentes (0 ó 1). Un ejemplo simple y típico podría ser la detección de email spam. En la que cada email, si es spam, será etiquetado con un 1; en caso de que no lo sea, será etiquetado con un 0.

El siguiente ejemplo es altamente ilustrativo de una clasificación binaria. *Fig. 26*.

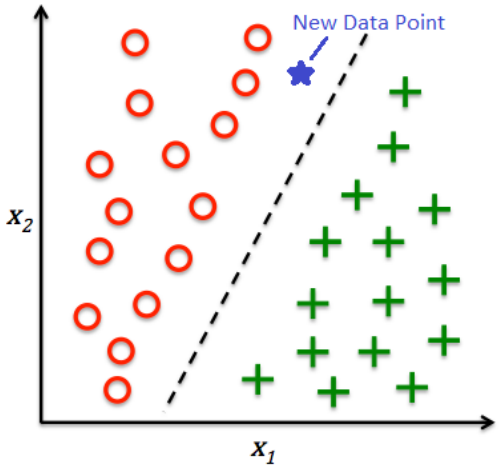


Figura 26: gráfico que representa la separación de clases de una clasificación binaria. Fuente: Medium

Como se puede ver, tenemos 2 clases, círculos y cruces. También, dos variables, X1 y X2. La visualización muestra que el modelo es capaz de encontrar la relación entre las variables de cada uno de los puntos de datos y su clase. De esta manera, puede establecer una línea de separación entre ellos, de forma tal que, al ingresar nuevos datos, es factible estimar las clases a las que pertenecen dados los valores de sus variables.

Clasificación Multi-clase: es la tarea de clasificar los datos de entrada cuando existen más de dos o más clases o grupos. Contrario a la clasificación binaria donde los datos se clasifican en una de dos clases. Algunos casos de uso de este tipo de clasificación pueden ser: clasificar noticias en diferentes categorías (deportes / entretenimiento / política), análisis de sentimientos; clasificar texto en positivo, negativo o neutral, segmentar a los clientes con fines de marketing, etc.

Clasificación de Multi-etiquetas: este problema de clasificación se puede confundir fácilmente con la clasificación de multi-clase, pero tienen una clara diferencia. Multi-etiquetas es una generalización de multi-clase (donde se asigna solo una etiqueta de múltiples clases posibles). En este caso, se tendrá más de una clase discreta que se puede predecir. Por lo tanto, es factible asignar más de una etiqueta a un mismo dato de entrada.

El presente trabajo refleja un caso de clasificación binaria. Se experimentaron diferentes modelos que permiten clasificar ciertos datos de entrada y predecir si un artículo determinado es Maquinable (menor o igual a 70 cm por lado) o No Maquinable (alguno de los lados > 70 cm).

3.2.2. Algoritmos de Clasificación

En la práctica, es recomendable probar y comparar el comportamiento de diferentes algoritmos para escoger el más adecuado en cada caso concreto. Este comportamiento estará muy influido por los datos disponibles, número de variables o descriptores textuales en nuestro caso, las diferentes clases existentes en el conjunto de datos y la relación existente entre las variables.

Los problemas de clasificación se pueden resolver con una gran cantidad de algoritmos. A continuación, se muestran algunos algoritmos de clasificación populares que fueron utilizados en los experimentos:

Clasificador Lineal

La regresión lineal se utiliza para identificar la relación entre una variable dependiente y una o más variables independientes a través de una función lineal; normalmente se implementa para hacer predicciones sobre resultados futuros, aunque también se puede utilizar como una herramienta descriptiva. Cuando solo existe una variable independiente y una variable dependiente, se conoce como regresión lineal simple. A medida que aumenta el número de variables independientes, se denomina regresión lineal múltiple[22].

Un caso de los algoritmos lineales es la regresión logística y para este proyecto será utilizada para intentar separar las clases de los productos. La regresión logística es utilizada cuando la variable dependiente es categórica, lo que significa que tienen salidas categóricas, como "verdadero" y "falso" o "sí" y "no". La regresión logística se utiliza principalmente para resolver problemas de clasificación binaria[21]. No obstante, es factible utilizar este algoritmo para predecir más de dos clases. Uno de los principales problemas en la clasificación ocurre cuando el algoritmo nunca converge en la actualización de los coeficientes de las variables dependientes mientras está siendo entrenado. Por lo general, esto ocurre cuando las clases no son separables, por lo menos, de manera lineal. Una de las razones principales por las cuales el algoritmo de regresión logística es tan popular, es porque es fácil de interpretar, es rápido computacionalmente, respecto a otros algoritmos y devuelve la probabilidad (valor entre 0 y 1) de un cierto dato de pertenecer a una clase en particular. Esto es extremadamente útil en casos como la predicción meteorológica, con la cual no solo nos gustaría saber si el tiempo va a ser lluvioso o no, además quisiéramos inferir la probabilidad de que llueva. En el ámbito médico, por ejemplo, la probabilidad de que un paciente tenga cierta enfermedad o no.

Si bien, la regresión logística es relativamente rápida en comparación con otras técnicas de clasificación supervisada, como MVS (máquina de vector soporte); dependiendo de la naturaleza del problema, podría brindar poca calidad predictiva. También, tiene los mismos problemas que la regresión lineal, ya que ambas técnicas son demasiado simplistas para relaciones complejas entre variables. La regresión logística tiende a tener un desempeño inferior cuando el límite de decisión entre clases no es lineal.

Máquinas de Vector Soporte (MVS)

El objetivo de este algoritmo está basado en la optimización de una distancia al establecer un plano de decisión que separe las clases, maximizando el margen entre el plano y los puntos de datos cercanos a este. Estos puntos se llaman vectores soporte. Es más fácil comprender esto al visualizar la siguiente figura. *Fig. 27*.

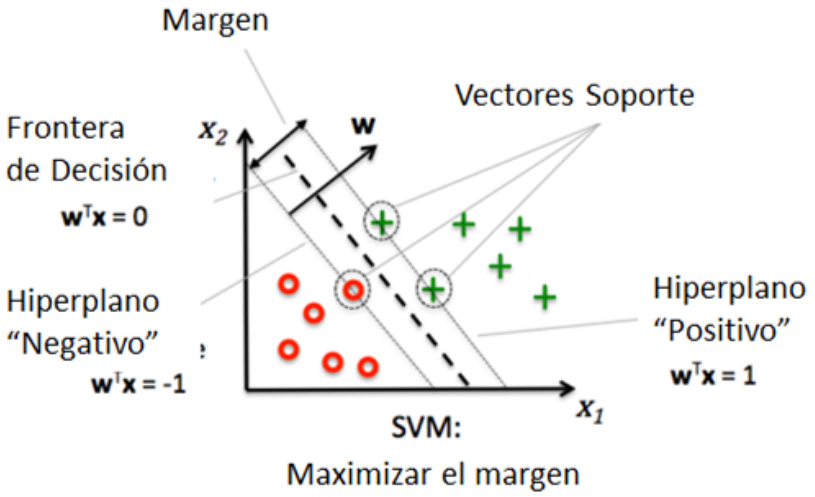


Figura 27: gráfico que representa la separación de clases de una clasificación binaria mediante una MVS. Fuente: Medium

Los vectores soporte son los puntos o datos que están más cerca del hiperplano, que influyen en la posición y orientación de este. Usando estos vectores soporte, es factible maximizar el margen del clasificador. Estos son los puntos que nos ayudan a construir la MVS[23].

Como parte del proceso de optimización, se procede a definir dos rectas paralelas, por medio de los vectores soporte, con las cuales se intentará maximizar sus distancias respecto de la frontera de decisión, tal como se puede ver en la *Fig. 27*. Se tendrán en cuenta los puntos clasificados erróneamente y también aquellos que puedan quedar entre los hiperplanos conformados. Normalmente, las líneas de decisión con márgenes grandes tienden a tener un error de generalización menor. Por otro lado, los modelos con márgenes pequeños tienen menor tendencia al sobreajuste[24] (overfitting, en inglés).

Para separar las dos clases, hay muchos hiperplanos posibles que podrían elegirse y el objetivo es encontrar un hiperplano que tenga el margen máximo, es decir, la distancia máxima entre los puntos de datos de ambas clases. El hecho de maximizar la distancia del margen brinda una mayor confianza para que los puntos de datos futuros se puedan clasificar con mayor precisión. Los puntos de datos que caen en uno de los lados del hiperplano pueden clasificarse en diferentes clases.
La dimensión del hiperplano depende del número de variables. Si el número de variables de entrada es 2, entonces el hiperplano es solo una línea en un espacio R^2. Si el número de variables de entrada es 3, entonces el hiperplano se convierte en un plano bidimensional en el espacio R^3. Se vuelve difícil imaginar cuándo el número de variables es mayor a tres. Algo destacable de las MVS es que utilizan lo que se llama el truco de kernel[25] (Kernel Trick, en inglés).

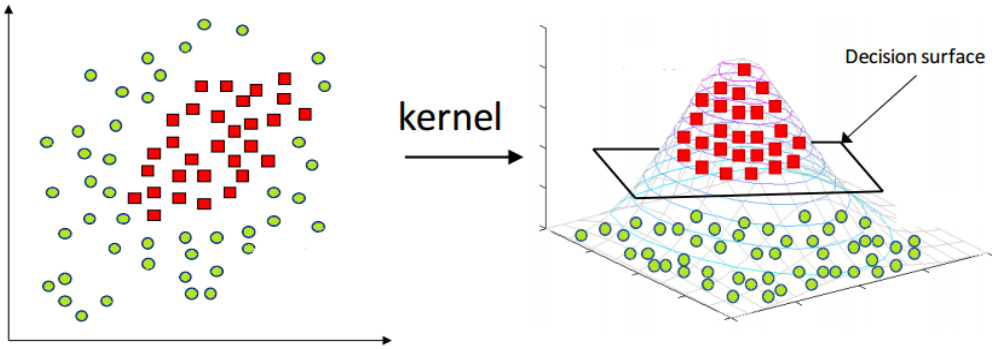


Figura 28: ejemplo de la utilización del kernel trick en una MVS. Fuente: Medium

Básicamente, el kernel es una función de transformación que altera el espacio original de los datos con el objetivo de hacer separables las clases. Cómo se puede ver en la *Fig 27*, la imagen de la izquierda muestra unos datos en el espacio R^2 que no son factibles de separar o al menos fácil determinar un hiperplano que separe las clases. En la imagen de la derecha, se puede ver que a través del truco del kernel, los datos son transformados, se obtiene un espacio de R^3 y ahora los datos se pueden separar a través de un plano.[26]

Árboles de decisión

Los árboles de decisión constituyen una familia de algoritmos de aprendizaje supervisado que implementan una estrategia de "divide y vencerás" mediante la división de los datos de entrada en diferentes regiones de manera jerárquica. Es un algoritmo que permite comprender de qué manera se van tomando las decisiones dado que se puede utilizar un árbol de decisiones para representar de forma visual y explícita el proceso de la toma de decisión. Tal como hace referencia el nombre del algoritmo, se utiliza un modelo de decisiones en forma de árbol. Un árbol de decisiones se puede representar con su raíz en la parte superior. En la *Fig. 28*, el texto en color violeta representa una variable (condición / nodo) interno, según el cual el árbol se divide en ramas. El final de la rama en donde ya no se realiza una división es la decisión-hoja (final).[27]

Consideremos un ejemplo muy básico que utiliza un famoso conjunto de datos[28] con el objetivo de predecir si una flor corresponde a una de las tres clases posibles, Setosa, Versicolor y Virginica. Este ejemplo utiliza 2 variables o atributos del conjunto de datos, el ancho (width) y el largo (length) del pétalo. La importancia de las variables es clara y las relaciones se pueden ver fácilmente. El árbol de la *Fig. 29* se llama árbol de clasificación, ya que el objetivo es clasificar el tipo de flor en función de las variables seleccionadas.[29]

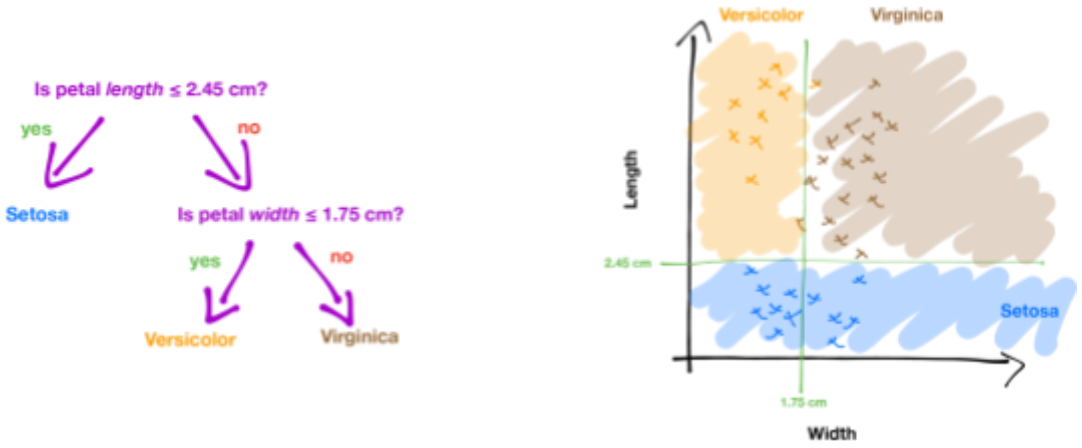


Figura 29: ejemplo de un árbol de decisión. Fuente: Berkeley Machine Learning

En el presente trabajo, el algoritmo utilizado se llama Random Forest; la traducción en español podría ser "Bosque Aleatorio". Este algoritmo utiliza como base los Árboles de Decisión. Como su nombre lo indica, Random Forest consiste en una gran cantidad de árboles de decisión individuales que operan como un solo conjunto, algo que nos introduce a las técnicas de ensamble en donde un conjunto de diferentes algoritmos es utilizado sistemáticamente para obtener un resultado único. En el Random Forest, cada árbol realiza una predicción de una clase y la clase más votada a lo largo de los diferentes modelos se convierte en la predicción de modelo conformado por todos los árboles. *Fig. 30*.

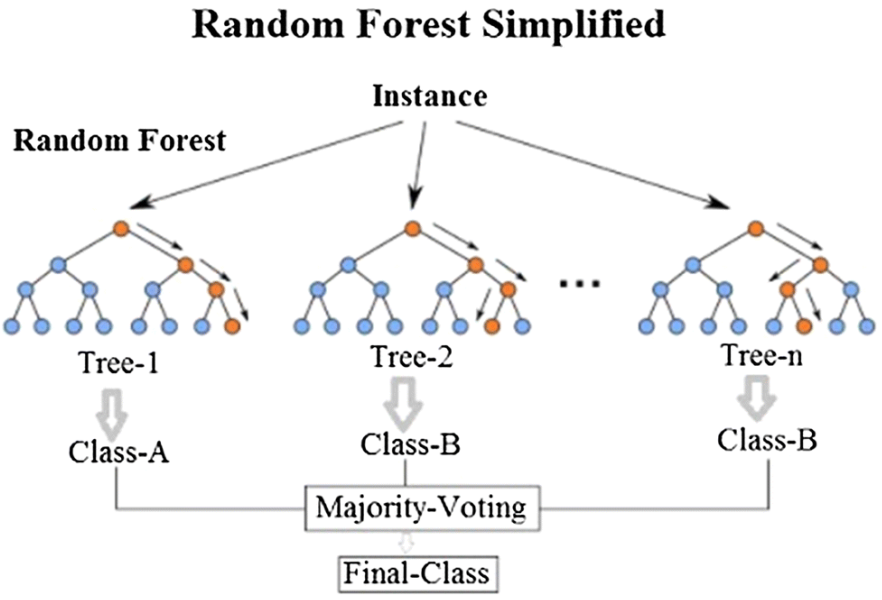


Figura 30: esquema que representa el proceso ejecutado en un modelo Random Forest. Fuente: Wikipedia

El concepto fundamental detrás de Random Forest es que la generación de un gran número de modelos (árboles) relativamente no correlacionados pueden producir predicciones, en conjunto, más precisas que cualquiera de las predicciones individuales de cada árbol. Para obtener estos árboles no correlacionados, cabe destacar que se utiliza una técnica llamada bootstrap, donde para cada árbol se genera un subconjunto de datos de entrenamiento que se obtienen realizando una selección aleatoria con repetición del conjunto de datos original.

Además, cada uno de los árboles no explora todas las variables, se realiza una selección aleatoria de estas. La razón de este efecto es que los árboles se protegen unos a otros de sus errores individuales. Si bien algunos árboles pueden estar equivocados, muchos otros árboles harán la predicción correctamente. Por lo tanto, como grupo, tienen mayor posibilidad de predecir la etiqueta correcta.[30]

Los árboles de decisiones son muy sensibles al cambio de la distribución de los datos con los que son entrenados; pequeños cambios en el conjunto de entrenamiento pueden dar como resultado estructuras de árbol significativamente distintas. El bosque aleatorio se aprovecha de esto al permitir que cada árbol individual pueda muestrear aleatoriamente el conjunto de datos, lo que da como resultado árboles con diferentes estructuras; esto se conoce como Bagging. Por otro lado, como se mencionó, la selección de variables con las que se construye cada árbol individual es aleatoria para tratar de crear un bosque de árboles no correlacionado y generar la mejor predicción posible.

Gradient Boosting Machine

Al igual que Random Forest, este algoritmo también se basa en árboles de decisión pero el enfoque de aprendizaje es totalmente diferente. El término "Boosting" hace referencia al término "incremental", esto quiere decir que el modelo aprende de manera progresiva. Por otro lado, también es considerado un método de ensamble debido a que el modelo final estará compuesto por una serie de árboles que evolucionan y mejoran los resultados de la función a optimizar[31]. Boosting es una técnica de aprendizaje por ensamble que funciona combinando varios aprendices débiles (weak learners - predictores con poca precisión) que luego darán como resultado un aprendiz fuerte (un modelo con mejor calidad predictiva). Conceptualmente, esta técnica funciona dado que cada nuevo weak learner presta atención a los errores de su predecesor. La idea detrás de Gradient Boosting es que en lugar de ajustar un predictor con los datos en cada iteración, ajusta un nuevo predictor a los errores residuales cometidos por el predictor anterior. En otras palabras, filtra las observaciones que el predictor débil anterior puede manejar y se enfoca en desarrollar nuevos predictores débiles para manejar las observaciones en las que se equivocó.[32]
Esto se representa en la *Fig. 31*.

Básicamente, Gradient Boosting está compuesto por tres elementos:

- Una función de costo o pérdida para optimizar.
- Un predictor débil para hacer predicciones de manera progresiva.
- Un modelo aditivo para agregar a cada predictor débil para minimizar la función de pérdida.

Función de pérdida: depende del tipo de problema que se quiera resolver. Debe ser diferenciable, pero se admiten muchas funciones de pérdida estándar y también se pueden definir funciones desarrolladas por el analista. Por ejemplo, si el problema está asociado a una regresión, se podría utilizar el error cuadrático medio. Para el caso de la clasificación, se podría utilizar una función basadas en la pureza de gini o entropía cruzada.[33]

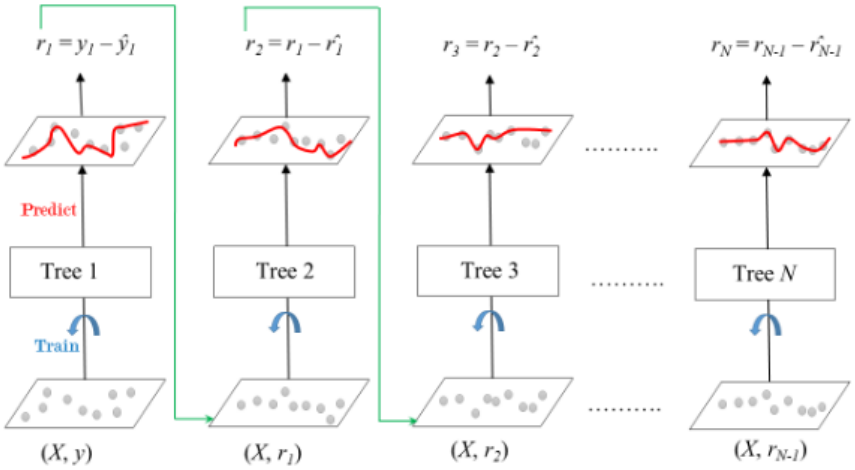


Figura 31: esquema que representa el proceso ejecutado en un modelo Gradient Boosting.
Fuente: geeksforgeeks

Predictor Débil: se utilizan árboles que generan valores reales para las divisiones y cuya salida se puede sumar, lo que permite agregar las salidas de los modelos posteriores y corregir los residuos en las predicciones. Los árboles se construyen eligiendo los mejores puntos de división basados en puntajes de pureza como Gini o Entropía.

Modelo aditivo: Los predictores débiles se agregan de uno y los árboles existentes en el modelo actual no se modifican. Se utiliza el procedimiento de descenso del gradiente de la función de costo para minimizar la pérdida al ir agregando los diferentes predictores débiles. Hay un hiperparámetro muy importante para este algoritmo que es la tasa de aprendizaje (learning rate, en inglés). Este hiperparámetro regula el paso y velocidad de aprendizaje. Se ha comprobado que una tasa de aprendizaje baja (entre 0.001 y 0.1) mejora la calidad predictiva del modelo[34]. Tradicionalmente, el descenso de gradiente se usa para minimizar un conjunto de parámetros, como los coeficientes en una ecuación de regresión o los pesos en una red neuronal. Después de calcular el error o la pérdida, los pesos se actualizan para minimizar ese error. La salida del nuevo árbol se agrega luego a la salida de la secuencia de árboles existente en un esfuerzo por corregir o mejorar la salida final del modelo. Por otro lado, se agrega una cantidad fija de árboles elegida por el científico o el entrenamiento se detiene una vez que la pérdida alcanza un nivel aceptable.[30]

3.2.3. Fast Text

FastText[35] es una librería desarrollada por Facebook Research, es utilizada para el aprendizaje de representaciones vectoriales de palabras y clasificación de texto.

Es una red neuronal conformada por tres capas; la capa lineal de entrada, una capa oculta - average pooling y la capa de salida plana que tiene una función de activación softmax[36].En la primera capa, además del vocabulario conformado por el documento, cada palabra es particionada en n-gramas que luego serán utilizados para obtener las representaciones para cada palabra y n-grama. *Fig. 32*.

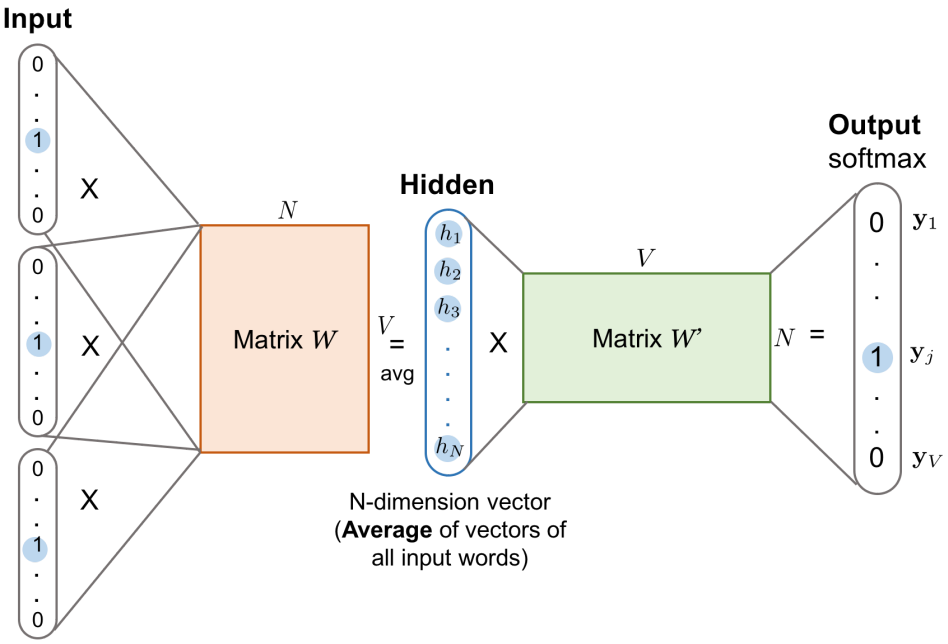


Figura 32: esquema de la arquitectura que presenta la librería Fast Text para clasificación.
Fuente: towardsdatascience

Luego, esas representaciones de palabras se promedian, para obtener una sola presentación para todo el texto; estas son procesadas por la capa oculta conformando vectores de dimensión igual a la representación que se desea obtener y por último, los datos transformados pasan a la capa de salida plana donde se obtiene la etiqueta de probabilidad máxima que se calcula mediante un función softmax. Este vector conformado contiene las palabras probables del vocabulario que podrían ser un par cercano respecto de la palabra de entrada. El objetivo es que se pueda encontrar una asociación y un contexto de todas las palabras. Se tiene la hipótesis de que las palabras que se utilizan en el mismo contexto se encuentran más cerca o tienen vectores similares. La imagen siguiente representa un ejemplo de la evaluación de un par de palabras. *Fig. 33*.

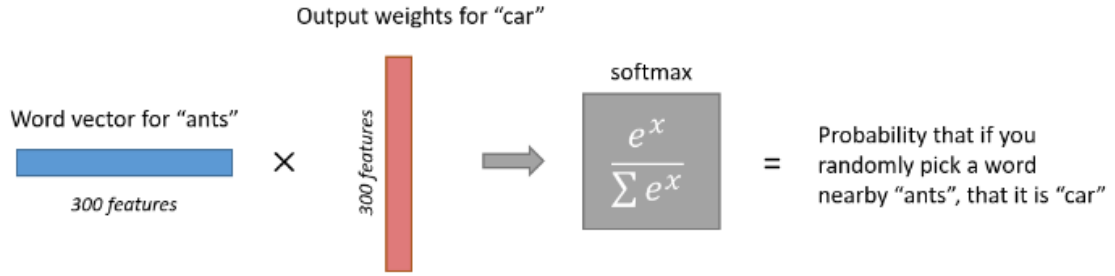


Figura 33: cálculo de ejemplo donde la función softmax se utiliza para la evaluación de un par de palabras.
Fuente: towardsdatascience

Softmax es una generalización de la regresión logística para múltiples clases. En nuestro caso, no hay diferencia dado que la problemática trata de una clasificación binaria. Sin embargo, esta función viene predeterminada por la arquitectura de Fast Text. Al margen de esto, Softmax proporciona un resultado intuitivo (probabilidades de clase normalizadas) y también tiene una interpretación probabilística.

Para brindar un ejemplo, supongamos que tenemos un problema de clasificación de tres clases (rojo, verde y azul). En la imagen que se ve a continuación, se puede ver el cálculo de los pesos y los bias para cada una de estas. Se realiza el cómputo de Y= Wi * Xi + b, luego se realiza la exponencial de dicho resultado (e^y) y se calcula el valor Softmax (e^y /sum e^y) obteniendo una probabilidad normalizada. Por último, se obtiene la pérdida (entropía cruzada) definida como -log(prob). Aquella clase que represente la menor pérdida será la clase predicha por el modelo. *Fig. 34*.

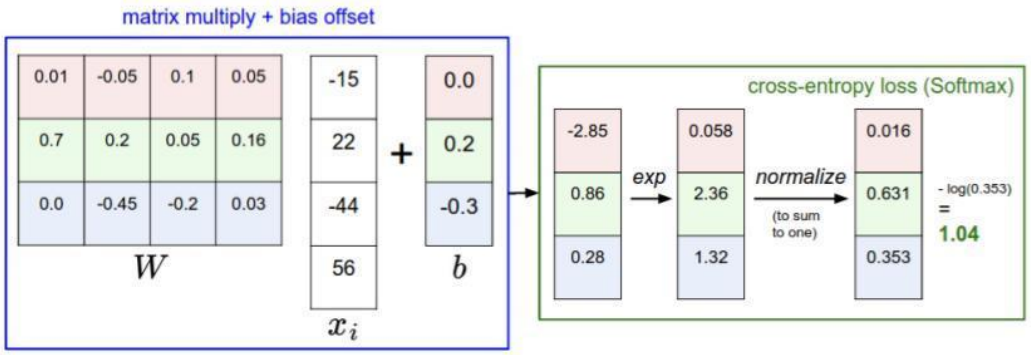


Figura 34: ejemplo de la predicción de una instancia mediante la utilización de la función de pérdida de entropía cruzada - Softmax.
Fuente: CS231n Convolutional Neural Networks for Visual Recognition (Stanford)

Debido a su estructura, Fast Text[37] es un modelo muy rápido en la fase de entrenamiento, en comparación con otros modelos de redes neuronales. Al mismo tiempo, ha logrado una alta precisión de clasificación y es ampliamente utilizado en la industria.

Fast Text, para la clasificación de texto, puede lograr un rendimiento relativamente bueno, obteniendo y utilizando las representaciones vectoriales de las palabras. Especialmente en el caso de palabras "raras", propias de un vocabulario específico, haciendo uso de la metadata a nivel de caracteres. Cabe destacar que este caso de uso representa muy bien el conjunto de datos del presente trabajo. Cuando se procesa el texto, Fast Text genera una representación de este mediante un conjunto de n-gramas además la representación vectorial de las palabras contenidas en dicho texto. El objetivo es preservar cierta información sobre las palabras circundantes que aparecen cerca de cada palabra objetivo[38].

Para dar un ejemplo de un n-grama se puede tomar la palabra "escuela" con n = 2. Las representaciones de texto para los n-gramas de caracteres serían "es, sc, cu, ue, el, la". También, es factible distinguir un n-grama respecto de una palabra en sí, como por ejemplo la palabra "es". Esto ayuda a preservar el significado de palabras más cortas que pueden aparecer como n-gramas de otras palabras. El modelo es considerado como un modelo de bolsa porque, aparte de la ventana de selección de n-gramas, no existe otro elemento interno que se utilice con el objetivo de capturar otras características del texto, es decir, siempre que los caracteres de la palabra estén comprendidos dentro de un intervalo, el orden de los n-gramas no se tiene en cuenta.
Durante la actualización del modelo, Fast Text aprende los pesos (la importancia) de cada uno de los n-gramas, como también el de la palabra completa. El vector objetivo para la

función de pérdida se calcula mediante una suma normalizada de todos los vectores de entrada. Los vectores de entrada son la representación vectorial de la palabra original y todos los n-gramas de esa palabra. Luego, se calcula y se normaliza la pérdida (entropía cruzada, cross-entropy en inglés) utilizando la función Softmax(fig), para luego actualizar los pesos para el siguiente paso, que se propagan hasta los vectores de la capa de entrada en el proceso de retropropagación[39]. Este paso es lo que nos permite aprender representaciones que maximizan la similitud. Fast Text utiliza la función Softmax dentro del arquitectura del modelo para generar la probabilidad de que un dato de entrada pertenezca a una cierta clase en problemas de clasificación. Softmax[40] suele brindar buenos resultados a pesar de que haya un desequilibrio de clases[41] presente en los datos, tal como sucede en el conjunto de datos del presente trabajo. Cabe destacar que, la utilización de esta función viene predeterminada por la misma librería.

3.3. Pipeline

3.3.1. Introducción

El objetivo de esta sección es introducir y esquematizar los diferentes pasos que atraviesan los datos como parte del proceso secuencial e iterativo del modelado, con el objetivo de desarrollar los experimentos. Este proceso tiene en cuenta las tareas mencionadas en las secciones anteriores y otros que serán explicados en las siguientes secciones. En la *Fig. 35*, están representados los pasos principales.

Los pasos de Exploración de Datos, Limpieza de Datos, Selección Inicial de Variables, Preprocesamiento de Datos y Separación de los Datos en Entrenamiento / Test, fueron desarrollados en las secciones 2.2, 2.3, 2.4 y 2.5 del presente trabajo. Como se mencionó anteriormente, el objetivo principal de esta secuencia de pasos es asegurar la obtención de un conjunto de datos que puedan ser utilizados por un modelo de aprendizaje supervisado que le permita aprender los patrones existentes en los datos, de manera tal, que sea factible poder realizar la clasificación de los productos maquinables y no maquinables que son enviados por medio la empresa. Continuando con el proceso, para cada iteración, que corresponde a cada experimento realizado, se realiza la selección de un descriptor. El descriptor puede estar conformado por uno o más de los campos textuales que se definieron como variables descriptivas de los artículos.

Luego de la selección y conformación del descriptor, comienza el proceso de entrenamiento. Esto se lleva a cabo en los distintos pipelines de entrenamiento para cada uno de los transformadores de descriptores seleccionados. Se utilizaron las transformaciones y representaciones de Fast Text, Fast Text Supervisado, Fast Text No Supervisado, Tensor Flow Multilenguaje y BERT.

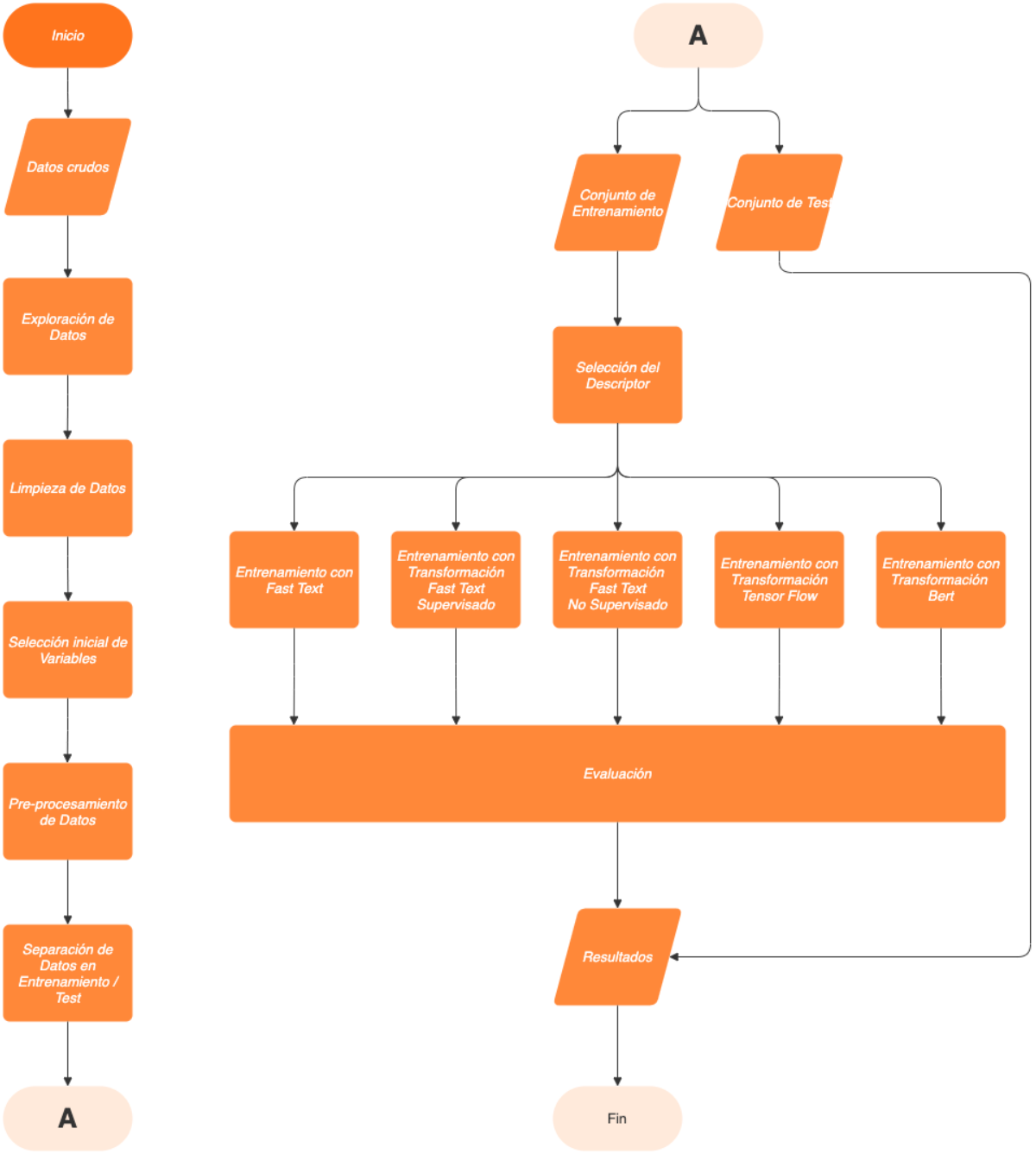


Figura 35: esquema simplificado del pipeline de procesamiento de los datos y del entrenamiento de modelo.

Fast Text

Fast Text tiene un proceso particular de procesamiento respecto a los otros modelos. Esto se debe a que Fast Text se presenta como una librería que puede entrenar un modelo de clasificación de texto mediante la configuración de algunos parámetros que permiten desarrollar un script con muy pocas líneas de código.

Cómo primer paso, es necesario separar el conjunto de datos de entrenamiento en dos subconjuntos, uno de entrenamiento per se y otro de validación. Esto se debe realizar dado que existe un método de la librería que permite realizar la configuración automática de parámetros del modelo en base a un conjunto de datos de validación, de forma tal de obtener el mejor rendimiento de predicción posible.
Cómo próximo paso, es necesario transformar el descriptor seleccionado y la etiqueta verdadera de cada registro en el formato .txt para cada conjunto de datos. A partir de esta instancia, es factible realizar el entrenamiento. Una vez que el modelo fue entrenado, es factible obtener la etiqueta predicha y la probabilidad asociada a dicha predicción. Con estos datos como salida del modelo, es factible realizar los cálculos de las métricas para evaluar el rendimiento final del mismo. *Fig 36*.

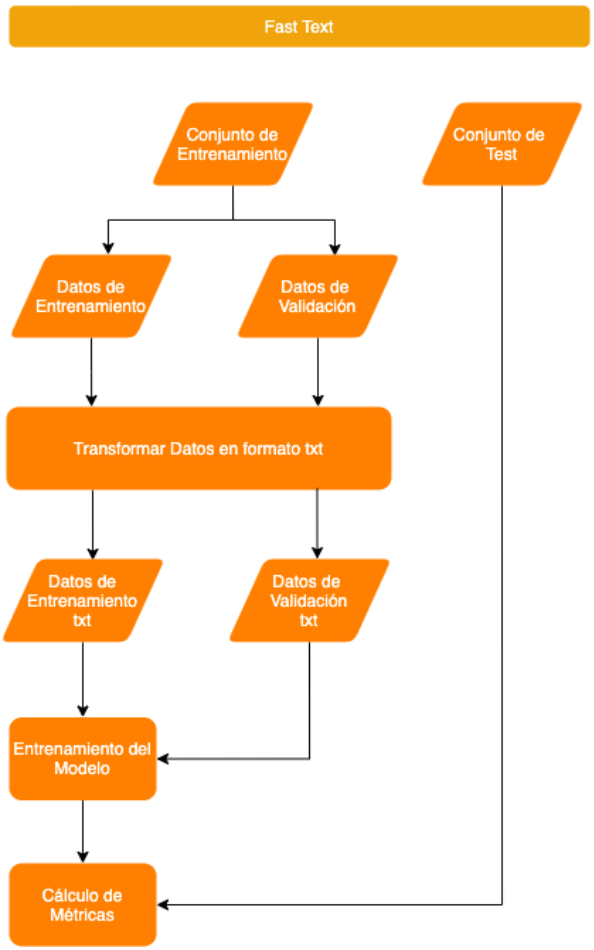


Figura 36: esquema simplificado del procesamiento de los datos y del entrenamiento del modelo Fast Text.

Resto de las Transformaciones

El procesamiento del resto de los transformadores seleccionados para realizar los experimentos, Fast Text Supervisado, Fast Text No Supervisado, Tensor Flow Multilenguaje[42] y BERT[43], recorren el mismo proceso de ejecución.
Debido a que el conjunto de datos de entrenamiento se encuentra muy desbalanceado y el objetivo es aprender a reconocer los productos de la clase minoritaria perdiendo poca precisión de la clase mayoritaria, se aplica una técnica de reducción aleatoria de datos llamada random under sampling. Esta técnica se aplica sobre la clase mayoritaria para mejorar el desbalance y tener un punto de partida menos sesgado a una clase. Posteriormente, se comienza con el proceso de entrenamiento. Se desarrolló una función que incluye cada uno de los pasos que deben seguir los datos para completar todo el proceso. Se puede ver el flujograma en la *Fig. 37*.

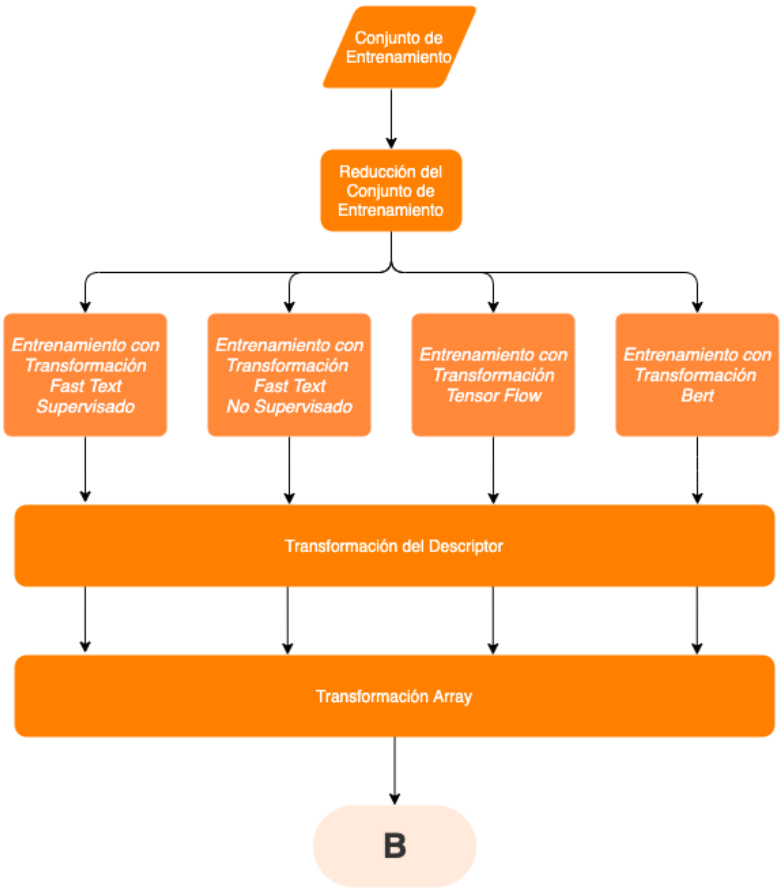


Figura 37: esquema simplificado del procesamiento de los datos en cada transformador.

El primer paso dentro de la función es la creación de una representación vectorial del descriptor seleccionado para dicha iteración. Luego, es necesario que esa representación vectorial sea transformada en un objeto de formato array para que pueda ser utilizada por la arquitectura del algoritmo predictivo (*Fig. 35*).

Además de obtener dicha transformación, se utiliza una técnica de aumento de datos sintéticos (se explicará más adelante) de la clase minoritaria (No Maquinable) con el objetivo de incrementar la cantidad de registros de dicha clase y así brindarle al modelo una mayor cantidad de datos que le permite aprender de mejor manera. Luego de este paso, se

procede a utilizar la técnica de reducción de datos para terminar de balancear el conjunto de datos en su totalidad. *Fig. 38*.

Después de estas transformaciones se obtiene un conjunto de datos y variables que deberán ser tratadas de manera independiente y que luego se unirán para ser procesadas por el clasificador. Por un lado, se tendrán las variables numéricas, conformadas por las representaciones vectoriales del descriptor y por otro lado, las variables que son consideradas categóricas, las cuales también pueden aportar valor a la predicción a realizar.

Las variables numéricas son estandarizadas y/o reescaladas para asegurar que una representación numérica sea más adecuada para los clasificadores que se utilizarán posteriormente. Existen varios algoritmos de aprendizaje automático que se benefician de la estandarización y/o re-escalamiento de los valores en un conjunto de datos. Si existen valores atípicos y están presentes en el conjunto, estas transformaciones son apropiadas para asegurar la variabilidad de los datos. Luego de realizar esta transformación, es necesario realizar una reducción de la dimensionalidad del conjunto de datos. Cuando nos enfrentamos a un gran número de variables correlacionadas, como es en el caso de las representaciones de palabras, es necesario obtener un número más pequeño de variables que, colectivamente, sean las más representativas y que expliquen la mayoría de la variabilidad del conjunto de datos original. Para realizar esto, se utiliza una técnica llamada PCA (Principal Component Analysis), que es un procedimiento de aprendizaje no supervisado que tiene el objetivo de encontrar las variables más representativas y capturar la mayor parte del valor predictivo de las variables y los datos.

Continuando con las variables categóricas, es decir, todos los campos textuales que quedaron fuera del descriptor seleccionado. Es necesario realizar una transformación para que estas puedan ser procesadas por los estimadores; la manera de realizar esto es a través de una técnica llamada One Hot Enconding, donde cada valor único dentro de una variable es representado por un valor binario.

De esta manera, se obtiene una matriz rala (matriz con una gran cantidad y dispersión de ceros) capaz de ser procesada por el clasificador. Cabe destacar que, en cada iteración se utilizan cuatro algoritmos clasificadores distintos con el objetivo de evaluar si las diferentes conformaciones y dimensiones de las representaciones modifican los resultados de los descriptores seleccionados. Todos los pasos antes mencionados son representados por la *Fig. 38*.

Al final de cada entrenamiento, se recolectan los resultados que arrojó cada modelo. De estos se puede obtener la etiqueta predicha y las probabilidades asociadas a cada clase. Por otro lado, con estos resultados, es factible realizar el cálculo de métricas que permitirá inferir el impacto de la solución ejecutada en cada iteración.

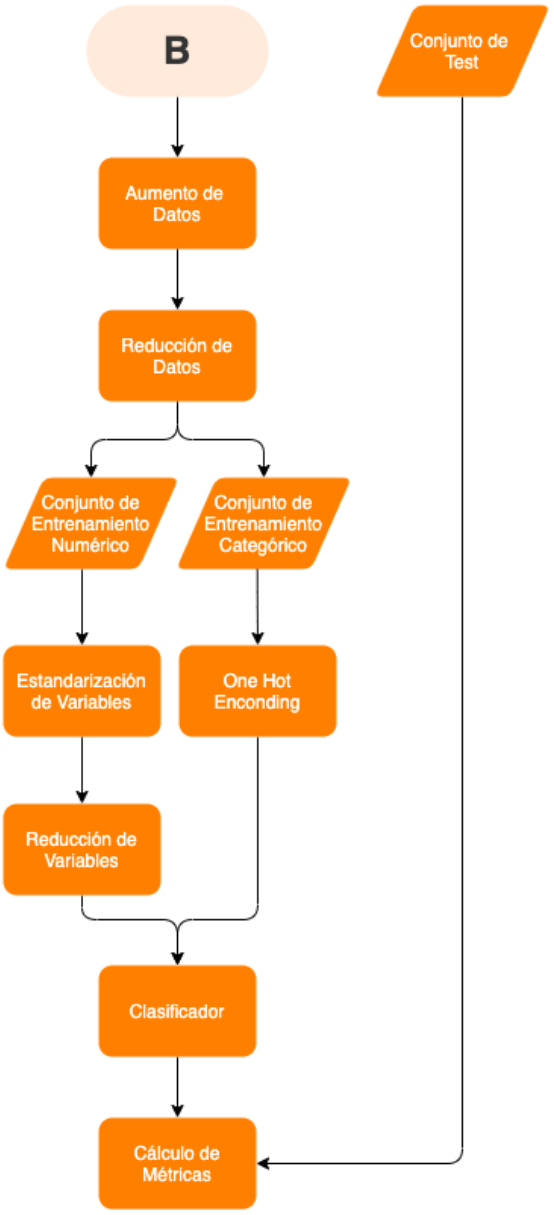


Figura 38: esquema simplificado de la transformación de los datos y entrenamiento del clasificador.

3.3.2. Ingeniería de Features

La ingeniería de features[44] es el proceso de transformar datos crudos, sin procesar, en variables que representan mejor el problema subyacente a los modelos predictivos, lo que resulta en una mayor precisión del modelo ante la predicción de datos futuros. En nuestro caso, debemos hacerlo para el escenario de clasificación[45].

El hecho de obtener mejores variables se puede traducir en obtener una mayor flexibilidad. Sin embargo, existe una relación importante entre flexibilidad y complejidad. Suele suceder que modelos más flexibles tienden a ser más complejos y muchas veces eso no es deseado. La generación de buenos features puede permitir el desarrollo de modelos menos complejos que son más rápidos de ejecutar, más fáciles de entender y más fáciles de mantener;

características muy deseables. Con variables bien diseñadas, se podrían elegir menos parámetros o no llegar a tener todos los parámetros óptimos y aún así obtener buenos resultados para el problema dado.

Gran parte del tiempo que consume un científico de datos, más del 50%, está relacionado con la obtención, organización, limpieza de datos e ingeniería de features. A continuación, se realizará una descripción de las técnicas utilizadas durante el proceso de transformación de los datos, previo al entrenamiento.

Representación de las palabras

Como se mencionó anteriormente, los humanos podemos reconocer una palabra, saber su significado, o un conjunto de palabras que forman una oración que, dado un contexto, tiene asociada una interpretación. Por otro lado, sabemos que las computadoras no pueden procesar toda esta información de la misma manera, solo pueden interpretar números. La manera de poder acercar esta tarea a las computadoras radica en crear una representación de palabras[46] que capture sus significados, relaciones semánticas y los diferentes tipos de contextos en los que se utilizan. Esto se puede realizar mediante el uso de representaciones numéricas vectoriales (*Fig. 39*), en inglés, word embeddings[47], de textos para que puedan ser procesados por modelos de aprendizaje automático.

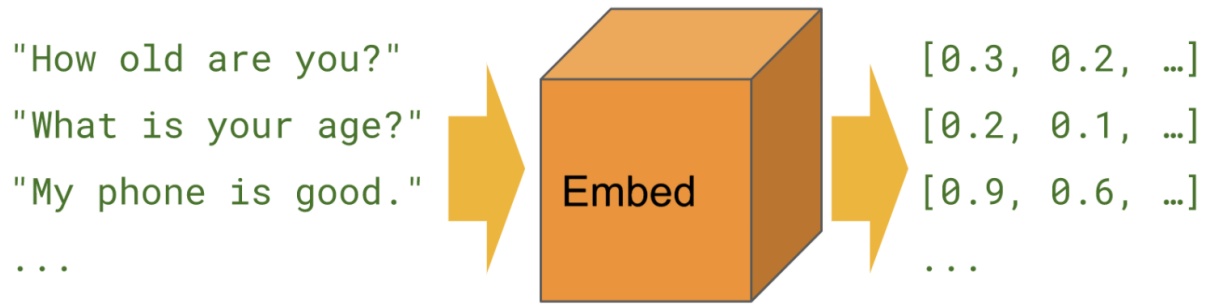


Figura 39: ejemplo de la utilización de una representación (embedding) para clasificación de texto.
Fuente: Google – Tensor Flow

Resulta que muchos, por no decir todos, algoritmos de aprendizaje automático y casi todas las arquitecturas de aprendizaje profundo son incapaces de procesar cadenas o texto en su formato original. Todos requieren números como datos de entrada para realizar cualquier tipo de operación.

En líneas generales, la representación de las palabras intenta mapear una palabra usando un vocabulario y creando un vector[48] en función a ese vocabulario. Siendo un poco más profundo, se expone el siguiente ejemplo. Se tiene la oración “Las representaciones de palabras son palabras convertidas en números”. Una palabra en esta oración puede ser “representaciones” o “números”. Por otro lado, un vocabulario puede ser la lista de todas las palabras únicas que contiene la oración ejemplo. En este caso, el diccionario puede verse así: [“representaciones”, “palabras”, “Las”, “son”, “convertidas”, “en”, “números”]. Una representación vectorial de una palabra puede ser un vector codificado con la técnica One Hot Encoding donde 1 representa la posición donde existe la palabra y 0 en cualquier

otro lugar. La representación vectorial de la palabra "números" en este formato, de acuerdo con el diccionario anterior, sería [0,0,0,0,0,1] y de "representaciones" sería [1,0,0,0,0,0]. Este es solo un método muy simple para representar una palabra en forma vectorial. Existen diversos métodos para realizar representaciones vectoriales.

En nuestro caso, las representaciones numéricas son realizadas por diferentes algoritmos. Dentro de una iteración, el descriptor es transformado de cinco maneras distintas y con diferentes dimensiones. Cuando el entrenamiento utiliza el procesamiento completo de Fast Text[49], es la propia librería la que realiza implícitamente dicha transformación y genera vectores de n dimensiones en base al aprendizaje que realiza. Este mismo vector es el que es utilizado en Fast Text Supervisado, es decir, se utiliza la misma representación pero luego todo el proceso de entrenamiento es diferente, tal como se describió en la sección 3.3.1. Por otro lado, en el caso de Fast Text No Supervisado, se utiliza la librería de manera independiente y utilizando un método de aprendizaje no supervisado sobre el corpus que genera todo el conjunto de descriptores; de esta manera se obtiene un vector con 300 dimensiones. Otras de las transformaciones son realizadas por el modelo pre-entrenado de Tensor Flow (Google) conocido como "universal sentence encoder multilingual" el cual brinda un vector de 512 dimensiones. Por último, también se utilizó el modelo pre-entrenado de Google llamado BERT (Bidirectional Encoder Representations from Transformers, sigla en inglés) que también brinda un vector de 512 dimensiones. Cabe destacar que para las transformaciones correspondientes a Fast Text Supervisado y No Supervisado se debió desarrollar una función adicional para convertir representaciones vectoriales de palabras en representaciones de sentencias textuales.

One Hot Encoding

Como se mencionó en reiteradas ocasiones, dentro de los datos disponibles existen una gran cantidad de variables categóricas y varias de estas presentan una cardinalidad significativamente alta. Como las categorías de una variable categórica no suelen ser numéricas, es necesario codificarlas de cierta forma para que el algoritmo de aprendizaje automático pueda aprovecharlas. En el ejemplo de la *Fig. 40*, se puede ver cómo está aplicado uno de los enfoques más simples y utilizados, conocido como codificación One-hot. Básicamente, consiste en generar una variable binaria para cada categoría dentro de una variable categórica. Por lo tanto, una variable categórica con n categorías únicas se codifica como un vector de características de longitud n. Una ventaja de este método, además de su simplicidad, es que se ocupa muy bien de los valores nulos dentro de la variable categórica.

Como se describe en la sección 2.3 del presente trabajo, el conjunto de datos contiene algunas variables con valores faltantes; por lo que codificar las variables categóricas que presentan ese escenario, es una forma de evaluar este problema sin perder la información derivada de esa variable por más que no haya valores para esos registros.

Color
Azul
Azul
Rojo
Verde
Rojo

Rojo	Verde	Azul
0	0	1
0	0	1
1	0	0
0	1	0
1	0	0

Figura 40: ejemplo de la codificación one-hot para el campo color, con categorías azul, rojo y verde.

La desventaja de la codificación One-hot es la matriz de salida que suele ser rala cuando este codificador se enfrenta a un gran número de categorías cuando el número de categorías es grande. Sin embargo, los algoritmos de aprendizaje automático utilizados para entrenar nuestros modelos se ocupan muy bien de este escenario.

Para todas las iteraciones, la codificación se ha realizado a través de la implementación del método OneHotEncoder perteneciente al módulo de la librería Scikit-Learn Preprocessing.

Data Augmentation

La tarea predictiva de clasificación con un conjunto de datos desbalanceados implica el desarrollo de modelos predictivos puedan manejar este tipo de escenarios. Cómo se mencionó anteriormente, el conjunto de datos para este trabajo presenta un alto desbalance de clases. El desafío de trabajar con un conjunto de datos desequilibrado es que este sesgo puede influir en el aprendizaje de un modelo, lo que lleva a ignorar por completo a la clase minoritaria. Este es un problema, ya que normalmente es la clase minoritaria la que suele ser más importante para la problemática en cuestión y debido a que el rendimiento de dicha clase suele ser parte del objetivo principal del modelo.

Un enfoque para abordar un conjunto de datos desbalanceados[50] es realizar un sobremuestreo, es decir, generar un incremento de los datos de la clase minoritaria. El enfoque más simple implica la duplicación aleatoria de ejemplos en la clase minoritaria, aunque estos ejemplos no agregan ninguna información nueva al modelo, solo le dan más peso. Por otro lado, se puede adoptar otro enfoque y realizar la creación de nuevos datos sintéticos, nuevos ejemplos a partir de los ejemplos y datos existentes.

Este es un tipo de técnica de Data Augmentation (aumento de los datos) para la clase minoritaria y se conoce como Técnica de Sobremuestreo de Minorías Sintética, o SMOTE[51] en inglés para abreviar (Synthetic Minority Oversampling Technique). Esta técnica se utiliza para incrementar datos estructurados y numéricos; también permite el manejo de variables categóricas y puede ser muy eficaz. SMOTE funciona seleccionando ejemplos cercanos en el espacio de variables y datos, dibujando una línea entre los ejemplos en el espacio de variables y generando una nueva muestra en un punto a lo largo de esa línea. SMOTE primero selecciona un registro “a” de clase minoritaria al azar y encuentra los k vecinos más cercanos. Luego, el nuevo registro sintético se crea eligiendo uno de los k vecinos más cercanos al azar “b”. Conectando “a” y “b” se forma un segmento de línea en

el espacio de variables. Los registros sintéticos se generan como una combinación convexa de las dos instancias elegidas “a” y “b”.

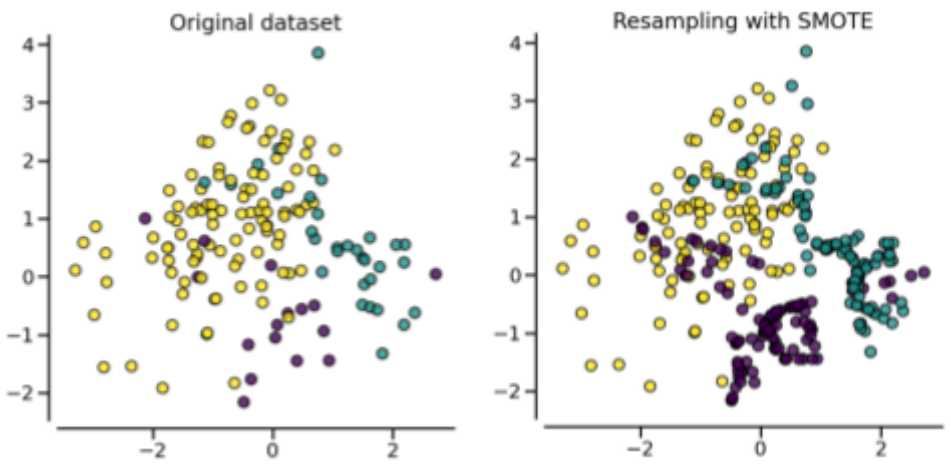


Figura 41: efecto de la utilización de SMOTE en un conjunto de datos desbalanceado.
Fuente: imbalanced-learn.org

En la *Fig. 41*, se puede visualizar un ejemplo del efecto producido sobre un conjunto de datos. El enfoque es eficaz porque se crean nuevos ejemplos sintéticos de la clase minoritaria que son plausibles, es decir, están relativamente cerca en el espacio de variables de los ejemplos existentes de la clase minoritaria. Por otro lado, una desventaja que presenta este enfoque es que los nuevos ejemplos sintéticos se crean sin considerar la clase mayoritaria, lo que posiblemente resulte en ejemplos ambiguos si hay una fuerte superposición de las clases a partir de algún punto de la distribución.

Para todas las iteraciones, la ejecución de la técnica se ha realizado a través de la implementación del método SMOTE, perteneciente al módulo de la librería Imbalance-Learn Over Sampling, compatible con la librería Scikit-Learn.

Data Reduction

Así como en el punto anterior se habló de la técnica de aumento de datos, también existe la técnica de reducción de datos. El submuestreo aleatorio es lo opuesto al sobremuestreo aleatorio[52]. Este método busca realizar la selección y eliminación aleatoria de la muestra de la clase mayoritaria, reduciendo progresivamente el número de ejemplos en dicha clase. Es una realidad que, durante la ejecución del submuestreo aleatorio, se descartan potenciales datos e información que el modelo no podría capturar. Esto podría llegar a ser problemático, ya que la pérdida de dichos datos puede dificultar el aprendizaje del límite de decisión entre las clases minoritarias y mayoritarias, lo que da como resultado una pérdida en el rendimiento de la clasificación. Este proceso puede repetirse hasta que el número de ejemplos en cada clase sea igual. El uso de este enfoque es eficaz en situaciones en las que la clase minoritaria tiene una cantidad suficiente de registros a pesar del desbalance que puede presentar el conjunto de datos en su totalidad. Además, la combinación de ambos métodos[53] (Aumento / Incremento de los datos) de muestreo aleatorio podría brindar como resultado, un rendimiento general mejorado en comparación con los métodos si estos se realizan de forma separada. El concepto detrás es que se puede aplicar una cierta cantidad de sobremuestreo a la clase minoritaria, lo que mejora el sesgo de los ejemplos de

la clase minoritaria, mientras también se realice una cierta cantidad de submuestreo en la clase mayoritaria para reducir el sesgo en los ejemplos de la clase mayoritaria. En la *Fig. 42*, se puede ver un ejemplo de un conjunto de datos conformado por tres clases donde una de ellas se podría definir como clase mayoritaria, luego de aplicar la técnica de submuestreo aleatorio se puede ver que la clase mayoritaria se ha reducido considerablemente. Cabe destacar que este procedimiento se puede aplicar a más de una clase.

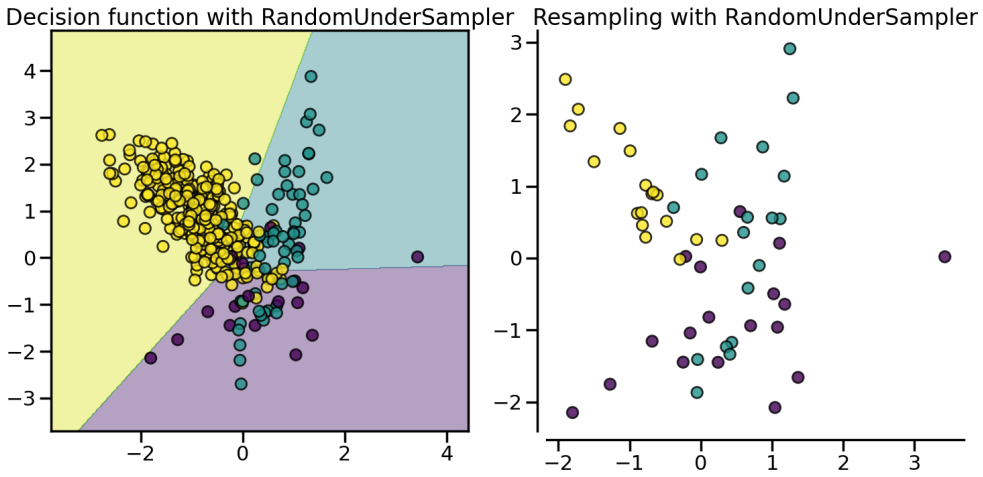


Figura 42: efecto de la utilización de under sampling en un conjunto de datos desbalanceado.
Fuente: imbalanced-learn.org

Para todas las iteraciones, la ejecución de la técnica se ha realizado a través de la implementación del método RandomUnderSampler, perteneciente al módulo de la librería Imbalance-Learn Under Sampling, compatible con la librería Scikit-Learn.

PCA (Principal Component Analysis)

PCA[54] es un algoritmo de aprendizaje automático no supervisado que intenta reducir la dimensionalidad, es decir, reducir el número de variables pertenecientes a un conjunto de datos mientras que se retiene la mayor cantidad de información posible. Esto se realiza encontrando un nuevo conjunto de variables llamadas componentes principales[55], que son combinaciones de las variables originales que no están correlacionadas entre sí.

La reducción de la cantidad de variables de un conjunto de datos se obtiene generalmente a expensas de la precisión del modelo pero el trueque en la reducción de la dimensionalidad es cambiar un poco de calidad predictiva por simplicidad. Las nuevas variables son ordenadas por la cantidad de varianza original que pueden describir. Técnicamente, PCA busca la proyección según la cual los datos quedan mejor representados, esta transformación lineal ajusta el conjunto de datos a un nuevo sistema de coordenadas de tal manera que la varianza más significativa se encuentra en la primera coordenada, y cada coordenada subsiguiente es ortogonal a la última y tiene una varianza menor. De esta manera, se convierte un conjunto de observaciones de variables posiblemente correlacionadas en un conjunto de variables con p componentes principales no correlacionados sobre el mismo conjunto de datos. Cómo se puede visualizar en la *Fig. 43*, los pequeños círculos azules representan el espacio original del conjunto de datos; por simplicidad y comprensión, se representa en dos dimensiones pero desde ya, el espacio

original de un conjunto de datos puede estar representado por muchas variables, por ende, muchas dimensiones. Cuando muchas variables se correlacionan entre sí, todas contribuirán en gran medida al mismo componente principal. Cada componente principal suma un cierto porcentaje de la variación total en el conjunto de datos. Cuando sus variables iniciales están fuertemente correlacionadas entre sí, se podrá capturar la mayor parte del comportamiento con pocos componentes principales. A medida que se agregan más componentes principales hace que su estimación del conjunto de datos total sea más precisa, pero también más difícil de manejar para el modelo.

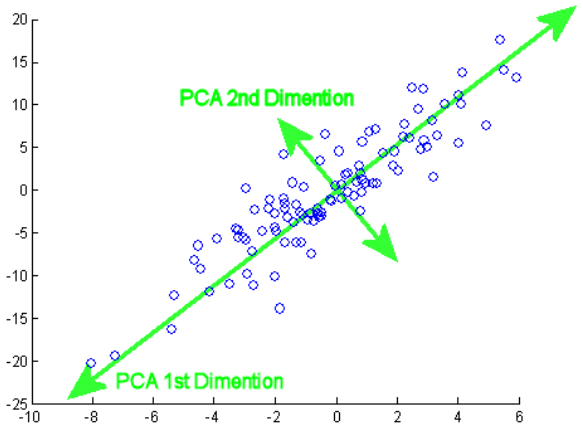


Figura 43: ejemplo ilustrativo de la aplicación de la técnica PCA. Fuente: weigend.com

Algo muy beneficioso de la técnica PCA es que, al proyectar el conjunto datos en un espacio más pequeño, estamos reduciendo la dimensionalidad de nuestro espacio de variables, lo que permite generar modelos más simples.

Para todas las iteraciones, la codificación se ha realizado a través de la implementación del método PCA perteneciente al módulo de la librería Scikit-Learn Decomposition.

A continuación, se explicarán las formas de evaluación de los modelos implementados.

3.4. Evaluación del Modelo Predictivo

Es necesario cuantificar que tan bien o que tan mal un modelo logra capturar los patrones y tendencias embebidos en los datos existentes con el objetivo de generalizar ese dominio y predecir nuevos datos correctamente. Para esto, se deben definir procesos de evaluación y métricas asociadas, piezas claves en el proceso de entrenamiento, que nos ayuden a inferir y mejorar el rendimiento de los modelos.

3.4.1. Procesos de Evaluación

Random Search (Búsqueda Aleatoria)

Se puede utilizar una variedad de algoritmos de optimización diferentes. Dos de los métodos más utilizados son la búsqueda aleatoria[56] (random search) y la búsqueda exhaustiva (grid search). La búsqueda aleatoria define un espacio de búsqueda con un dominio limitado de valores de hiperparámetros y puntos de muestreo aleatorios en ese dominio. Por otro lado, la búsqueda exhaustiva define un espacio de búsqueda como una

grilla de valores de hiperparámetros y evalúa todas y cada una de las combinaciones posibles en ese dominio. Esta última requiere un costo computacional y un uso del tiempo considerablemente mayor a otra técnica de optimización existente. En la *Fig. 44*, se puede ver una simple representación de los conceptos antes mencionados.

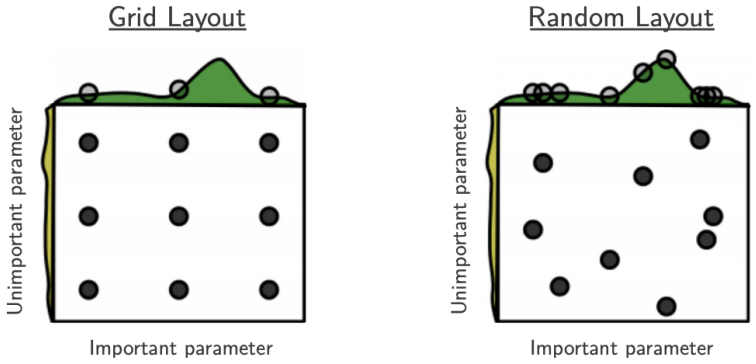


Figura 44: ejemplo ilustrativo de Grid Search y Random Search.
Fuente: Random Search for Hyper-Parameter Optimization

La técnica elegida para nuestro caso es la búsqueda aleatoria. En pocas palabras, este método consiste en generar diferentes configuraciones del pipeline de entrenamiento de un modelo de manera aleatoria. Cada uno de los pasos del pipeline, como así también los algoritmos clasificadores, contiene parámetros denominados hiperparámetros que permiten modificar diferentes características del modelo. Esto repercute directamente en el rendimiento del modelo y por ende, en los resultados obtenidos.

De forma independiente, es cierto que se conocen los efectos y el impacto que pueden tener los hiperparámetros en un modelo pero prever cuál es el impacto que tendrán de manera conjunta es un desafío y un arte dentro del aprendizaje automático. A menudo, existen algunas heurísticas generales o reglas empíricas para configurar hiperparámetros. Sin embargo, un mejor enfoque es buscar objetivamente diferentes valores para los hiperparámetros del modelo y elegir un subconjunto que dé como resultado, un modelo que obtenga el mejor rendimiento posible en un conjunto de datos determinado. Esto se denomina optimización de hiperparámetros y la búsqueda aleatoria es una de las técnicas para llevarlo a cabo.

Un procedimiento de optimización implica definir un espacio de búsqueda. Geométricamente, el espacio de búsqueda se puede considerar como un volumen de n dimensiones, donde cada hiperparámetro representa una dimensión diferente y la escala de la dimensión son los valores que el hiperparámetro puede tomar, como valores reales, valores enteros o categóricos. Un punto en el espacio de búsqueda es un vector con un valor específico para cada valor de hiperparámetro. El objetivo del procedimiento de optimización es encontrar un vector que dé como resultado el mejor rendimiento del modelo después del aprendizaje de este, minimizando la función de pérdida o maximizando la función de ganancia según sea el caso. En nuestro caso, se buscará minimizar una función de pérdida.

El inconveniente de la búsqueda aleatoria es que se produce una gran variación durante la computación debido a que la selección de parámetros es completamente aleatoria y dado que no se utiliza inteligencia para probar estas combinaciones, la suerte también juega su

papel. A medida que transcurre cada iteración, es muy probable que la selección de valores aleatorios haya cubierto gran parte del espacio de acción definido. Esta técnica funciona mejor si se adopta el supuesto de que no todos los hiperparámetros son igualmente importantes. Las posibilidades de encontrar el parámetro óptimo son comparativamente más altas en la búsqueda aleatoria debido al patrón de búsqueda aleatorio en el que el modelo podría terminar siendo entrenado con los parámetros óptimos. Además, esta técnica resulta insumir un menor tiempo computacional y tener un menor riesgo de sobreajuste sobre los datos.

En el Anexo C: Hiperparámetros, se detalla el espacio de hiperparámetros definidos dentro de nuestro pipeline de entrenamiento.

Para realizar la evaluación conjunta de todo el pipeline se ha utilizado el método RandomizedSearchCV, perteneciente al módulo de la librería Scikit-Learn Model_Selection.

A continuación, se explicará la técnica que se utiliza para evaluar si las combinaciones elegidas de los hiperparámetros permiten obtener el mejor rendimiento posible del modelo. Esta técnica está embebida en el método RandomizedSearchCV.

Validación Cruzada

Simplemente por haber desarrollado un modelo de aprendizaje automático y entrenarlo con un cierto conjunto de datos, no se puede suponer que funcionará bien con datos que todavía no han sido vistos. En otras palabras, no podemos estar seguros de que el modelo tenga la calidad predictiva deseada en un contexto real o ante datos desconocidos. Es necesario algún tipo de garantía sobre el rendimiento del modelo y las predicciones que puede realizar. Es necesario encontrar un proceso de validación que permita inferir si los resultados numéricos que cuantifican las relaciones entre variables son aceptables como representación de los datos o no.

Para evaluar el rendimiento de cualquier modelo de aprendizaje automático, se debe poder realizar un testeo en una muestra de datos (representativa del dominio) que el modelo no haya visto todavía. En base a la realización de la validación en estos datos no vistos, se podrán obtener resultados que cuantifique si el modelo tiene un ajuste insuficiente (underfitting), un ajuste excesivo (overfitting) o si realiza una buena generalización del dominio que es el resultado ideal. La validación cruzada[57] (CV) es una de las técnicas utilizadas para probar la efectividad de un modelo. Para realizar una CV, necesitamos dejar a un lado una muestra de los datos que no se usa para entrenar el modelo para que luego sea utilizada como conjunto de validación. La *Fig. 45*, representa efectivamente el proceso de validación cruzada K-Fold. En la validación cruzada de K iteraciones (K-fold cross-validation, en inglés), con K= 5; los datos del conjunto de datos de entrenamiento se dividen en K subconjuntos. Uno de los subconjuntos se utiliza como subconjunto de validación (V) y el resto (K-1) como subconjunto de datos de entrenamiento. El proceso de validación cruzada es repetido durante k iteraciones, por cada uno de los posibles subconjuntos de datos de validación. Finalmente, se realiza la media aritmética de los resultados de cada iteración para obtener un único resultado. Este método es muy preciso puesto que evaluamos a partir de K combinaciones de datos de entrenamiento y de prueba. En la práctica, la elección del número de iteraciones depende de la medida del conjunto de datos.

La técnica validación cruzada de K iteraciones es popular y fácil de entender, generalmente da como resultado un modelo menos sesgado en comparación con otros métodos porque garantiza que cada observación del conjunto de datos original tenga la posibilidad de aparecer en el conjunto de entrenamiento y prueba.

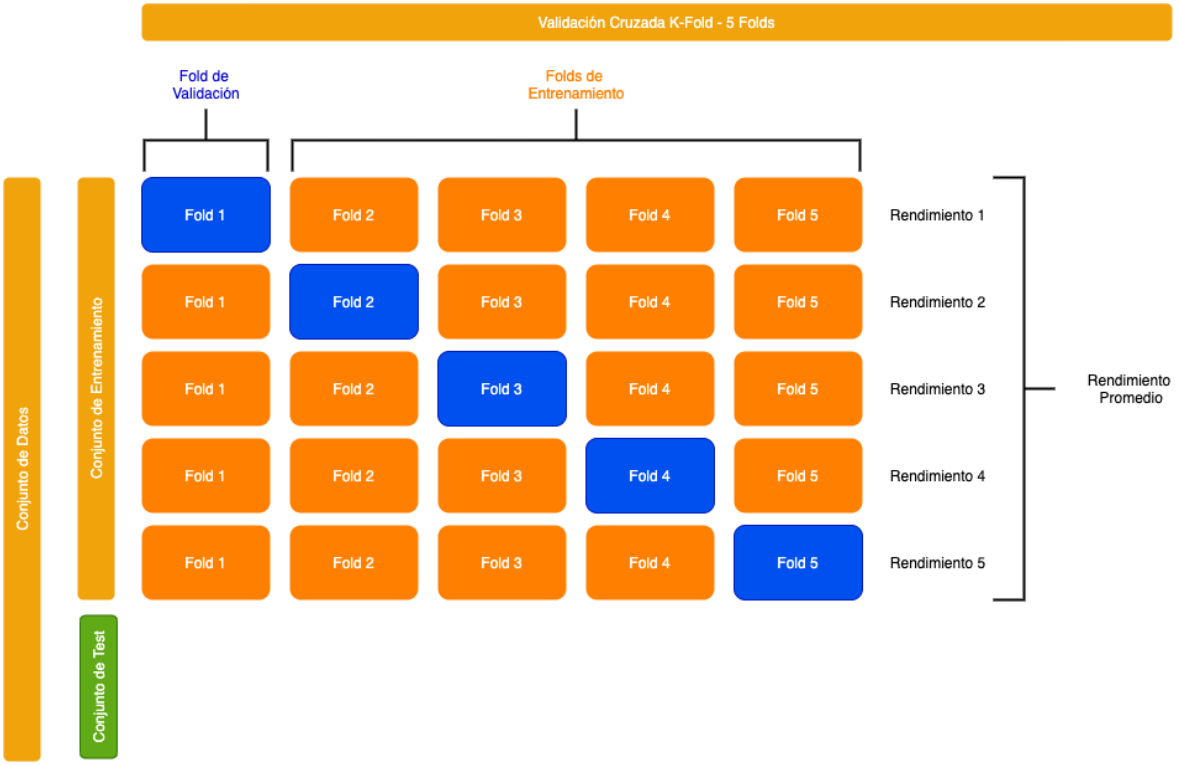


Figura 45: estructura del proceso de validación cruada k-fold, con k=5.
Fuente: Random Search for Hyper-Parameter Optimization

3.4.2. Métricas

Las diferentes métricas[58] que se adecuen mejor al proceso a mensurar, serán las que conformen la expresión que cuantifique si lo realizado hasta el paso anterior fue correcto o no. Existen distintos tipos de métricas, en este caso, se propone abordar una métrica de desarrollo, la cual es propia de los algoritmos de clasificación a evaluar y que intenta cuantificar los aciertos y errores del modelo en sí. Por otro lado, se propone abordar una métrica del proceso real, aquella que puede cuantificar el impacto en el proceso productivo o escenario real en el cual estaría integrado el modelo. En nuestro caso, esta métrica está asociada con componentes de costos.

Métrica de Desarrollo

En primera instancia, todos los algoritmos de clasificación utilizados dentro del proceso de entrenamiento y que son evaluados dentro de la búsqueda aleatoria de hiperparámetros, comparten por defecto una métrica de desarrollo denominada “Mean Accuracy”. La métrica de “Accuracy” se utiliza como una medida estadística para inferir qué tan bien una prueba de clasificación binaria identifica correctamente las clases correspondientes. En otras

palabras, la "Accuracy" es la proporción de predicciones correctas (tanto verdaderas positivas como verdaderas negativas) entre el número total de instancias predichas.

		Predicción	
		Menor_70: 1	Mayor_70 : 0
Real	Menor_70: 1	VP	FN
	Mayor_70 : 0	FP	VN

Puesto en una fórmula, "Accuracy" = (VP + VN) /(VP + VN + FP + FN)
donde: VP = verdadero positivo; FP = falso positivo; VN = verdadero negativo; FN = falso negativo.

Dado que el pipeline de entrenamiento es ejecutado reiteradas debido a la búsqueda aleatoria de hiperparámetros, diversos modelos son ejecutados y todos son evaluados con la métrica de "Accuracy". Habiendo finalizado la búsqueda, se promedian los resultados de las ejecuciones y de ahí se obtiene la "Mean Accuracy". Sin embargo, se sabe que el conjunto de datos de entrenamiento es balanceado durante el entrenamiento para darle mayor importancia a la clase minoritaria, no así al conjunto de datos de test que continúa representando la distribución original del dominio y que no forma parte del entrenamiento, solo de la evaluación final. Debido a esto, la métrica de "Mean Accuracy" podría no representar el comportamiento del modelo correctamente. Por esta razón, se escoge una métrica complementaria que será calculada en el paso de evaluación final. Esta métrica es denominada AUC-ROC y será explicada a continuación.

AUC - ROC

La métrica se denomina "Área bajo la curva ROC" (AUC - Area Under the Curve). El área bajo la curva ROC[59] (AUC-ROC) es una métrica de rendimiento ampliamente conocida en la comunidad de aprendizaje automático que ha ido ganando cada vez más adherencia en los últimos años debido a sus propiedades que la hacen especialmente útil para dominios con distribución de clases sesgada.

Como su nombre lo indica, esta métrica se obtiene calculando el área bajo la curva de las Características Operativas del Receptor (ROC - Receiver Operating Characteristics). La curva de características operativas del receptor es un gráfico que ilustra el potencial de un clasificador a medida que varía su umbral que discrimina una clase de otra. La curva se obtiene trazando las diferentes combinaciones de TPR (tasa de verdaderos positivos, true positive rate en inglés) en el eje "y", y FPR (tasa de falsos positivos, false positive rate en inglés) en el eje "x", donde cada punto del gráfico es derivado del uso de un cierto umbral discriminante.

El TPR es = VP / (VP + FN). Esta ratio refleja la proporción de aciertos que puede tener el modelo respecto del total de casos que son realmente positivos para un cierto umbral.

El FPR es = FP / (FP + TN). Esta ratio refleja la proporción de errores que puede tener el modelo respecto del total de casos que son realmente negativos para un cierto umbral.

Donde: VP = verdadero positivo; FP = falso positivo; VN = verdadero negativo; FN = falso negativo.

Al iterar sobre el espacio de umbral entre [0,1] y trazar sus resultados, se puede dibujar una curva. Cuanto más cerca esté la curva de la esquina superior izquierda, mayor será el potencial del modelo para lograr una separación adecuada de clases. En el extremo, la curva ROC de un clasificador aleatorio sería en realidad la diagonal donde el TPR = FPR. En la *Fig. 46*, se puede ver un ejemplo de la curva AUC -ROC, donde se representa la calidad de un modelo de regresión logística.

Si bien es interesante en términos gráficos, para comparar los diferentes modelos clasificadores es necesario reducir la medida de rendimiento a un solo valor escalar y ahí es donde AUC toma relevancia. Dado que es una parte del área de una unidad cuadrada y se encuentra en un valor entre 0 y 1. Un modelo se puede considerar muy bueno cuando tiene un AUC cercano al 1, lo que significa que tiene un buen potencial de separabilidad. Un modelo deficiente tiene un AUC que tiende al valor 0, lo que significa que tiene muy poca o nula capacidad de separabilidad. De hecho, significa que predice 0 como 1 y 1 como 0. Cuando el AUC es 0,5, significa que el modelo tiene una capacidad de separación de clases aleatoria, es como si una persona estuviera adivinando.

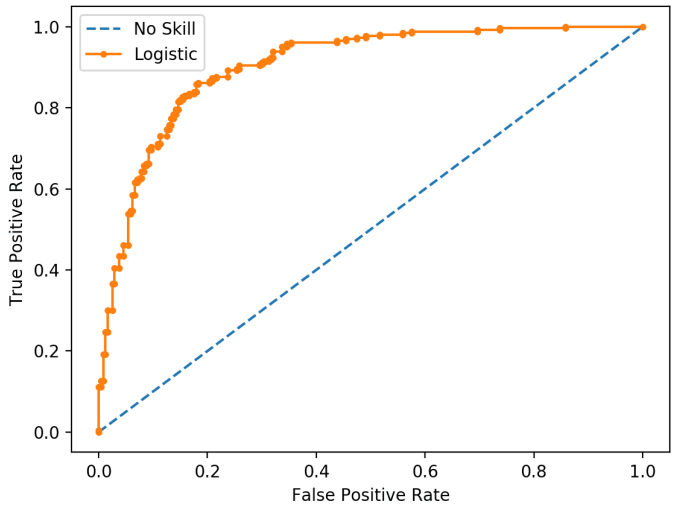


Figura 46: gráfico ejemplo de la curva ROC. El eje horizontal representa la tasa de falsos positivos y el eje vertical la tasa de verdaderos positivos. Fuente: Medium

Por otra parte, el AUC-ROC tiene una propiedad estadística importante: su valor es equivalente a la probabilidad de que un clasificador clasifique una instancia positiva elegida aleatoriamente por encima de una instancia negativa elegida aleatoriamente.
El análisis de la AUC - ROC proporciona herramientas para seleccionar los modelos en base a su rendimiento y optar por aquellos que cumplan los niveles requeridos.

De manera complementaria, se suelen utilizar las métricas conocidas como “Precision” y “Recall” para cuantificar el rendimiento del modelo en términos de aciertos y errores, para cada una de las clases.

La métrica “Precision” cuantifica la cantidad de predicciones correctas respecto del total de predicciones realizadas. Es decir, se tiene en cuenta las predicciones correctas y los errores cometidos. En una fórmula, Precision = VP / (VP + FP).
Por otra parte, “Recall” cuantifica la proporción de aciertos respecto de todas las instancias correctas. Es decir, toma en cuenta las predicciones correctas y aquellas instancias que no se predijeron con dicha clase. En una fórmula, Recall = VP / (VP + VN).

Clasificación con una Métrica Sensible al Costo

Una métrica de desarrollo puede ser un buen proxy del desempeño de un modelo de aprendizaje automático pero difícilmente puede representar concretamente el escenario real de la oportunidad o problemática a resolver. En base a la naturaleza de la oportunidad de este proyecto, es factible desarrollar una manera de evaluar el impacto que el modelo podría llegar a realizar si este estuviese productivo dentro de un proceso existente. Se decidió realizar un métrica sensible al costo[60] .En la sección 1.4.2 se explicaron los diferentes casos de uso que se podrían dar si el modelo pudiese intervenir en la decisión de ofrecer un tipo de gestión de envío u otro. Dependiendo del rendimiento del modelo, existe la posibilidad de acierto, es decir, identificar un caso verdadero cuando un producto es realmente maquinable o no, como también, existe la posibilidad de error cuando el modelo identifica un producto maquinable como no maquinable o viceversa.

Cada caso de uso representa un escenario distinto, el cual contiene variables de costos asociadas. A continuación, se presenta una tabla que resume la interacción entre los diferentes componentes. Debido a que el objetivo es obtener el resultado más bajo, las ganancias de se presentan con signo negativo y los costos o pérdidas con signo positivo.

		Predicción	
		Menor_70: 1	Mayor_70: 0
Real	Menor_70: 1	- Fee envío	Precio * Fee venta * Prob(no venta)
	Mayor_70: 0	Multa - Fee envío	- Multa + Fee envío

El escenario *Verdadero Positivo*, tendrá en cuenta solo el costo de envío aproximado del artículo multiplicado por el fee de envío que percibe la plataforma por la gestión de este y que, por ende, representa una ganancia. Cabe destacar que, solo se contabiliza el revenue obtenido.

El escenario *Verdadero Negativo*, tendrá en cuenta el costo de la multa y el costo de envío aproximado del artículo multiplicado por el fee de envío que percibe la plataforma, dado que se genera el ahorro de la multa pero por otro lado se tiene la pérdida de la ganancia por el envío no realizado.

En el escenario *Falso - Positivo*, el costo de la multa es considerada como un costo pero por otro lado, se tiene en cuenta la ganancia del precio del artículo multiplicado por el fee de envío ya que el ítem es enviado de cualquier manera.

El escenario *Falso - Negativo*, tendrá en cuenta el precio del artículo multiplicado por el fee correspondiente a la venta del ítem y por la probabilidad de no vender el ítem, debido a que el producto podría haber sido enviado por medio de la gestión de la plataforma pero dada la predicción, no se ofrece la opción.

Para cada valor posible de umbral discriminante del clasificador, habrá una cierta cantidad de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos correspondientes al conjunto de datos de testeo. Para cada punto se realizará la suma o resta de los escenarios antes mencionados y se determinará el costo asociado que tendrá ese umbral de decisión.

La ecuación de costo es = (-Fee envío)*VP + (Precio * Fee venta * Prob(no venta))*FN
+ (Multa - Fee envío)*FP + (- Multa + Fee envío)* TN

De esta manera, se podrán armar las curvas de costos en función de los resultados que podrían generar los diferentes umbrales de clasificación para cada iteración. Por lo tanto, se podrá evaluar la efectividad del modelo productivo en un escenario más cercano al real y determinar si es factible implementar la solución planteada.

Cabe destacar que los modelos no fueron entrenados teniendo en cuenta la función de costo antes mencionada. Solo se utilizó para evaluar cuál hubiese sido el rendimiento en función del conjunto de datos de test.

A continuación, se explicaran los experimentos realizados y los resultados obtenidos en las diferentes iteraciones.

4. Experimentos

En cada uno de los experimentos, el descriptor y las variables complementarias recorren el proceso de entrenamiento descrito en la sección 3.3. También, los modelos fueron entrenados de dos maneras distintas, con conjuntos de entrenamiento distintos (Train 1 y Train 2), tal como se explica en la sección 2.5.
En la parte final de cada iteración se realiza el cálculo de métricas, descrito en la sección 3.4.2. Cabe destacar que fue utilizado el mismo conjunto de datos de test para evaluar todos los experimentos. Dicho conjunto, como escenario ideal, presenta un ahorro potencial de USD 160.731. Es decir, en el caso que todas las predicciones fuesen correctas.

A continuación, se explicarán el porqué de los experimentos propuestos, la descripción de estos y los resultados obtenidos.

4.1. Motivación

Parte de la motivación para realizar los experimentos está asociada con comprobar si era factible identificar las diferentes clases (maquinables / no maquinables) de productos a través de los descriptores textuales que conforman la descripción de un producto. Datos a los cuales se puede acceder en un escenario real. Otra parte de la motivación, algo que fue mencionado en la propuesta del presente trabajo, era la aplicación de una librería capaz de resolver el problema de clasificación con muy pocos pasos, con una simple configuración y verificar su efectividad (Fast Text). De manera complementaria y para poder realizar diferentes comparaciones, se desarrolló otra arquitectura de entrenamiento con distintos transformadores de texto y algoritmos clasificadores.

4.2. Descripción

Tal como se mencionó en la sección anterior, parte de la motivación está centrada en la utilización de los diferentes descriptores textuales de un producto. Cada iteración, es una combinación única de todos estos datos y se han probado de distintas maneras para descubrir si alguna de estas aporta información valiosa que le permita reconocer la clase correspondiente a un modelo clasificador. Por conocimiento del dominio, se sabe que el título de un producto aporta mucha información de los productos, por ende, se decidió que siempre sea parte del descriptor conformado. El resto de los datos textuales son combinados de distintas maneras, inclusive los descriptores ya generados en otras iteraciones. Los experimentos realizados son:

- **Experimento 1**: el descriptor está conformado por el título.
- **Experimento 2**: el descriptor está conformado por el descriptor del experimento 1 + la variable domain_id del producto.
- **Experimento 3**: el descriptor está conformado por el descriptor del experimento 2 + la variable category_id del producto.
- **Experimento 4**: el descriptor está conformado por el descriptor del experimento 3 + la variable seller_id del producto.

- **Experimento 5**: el descriptor está conformado por el descriptor del experimento 4 + la variable brand_name del producto.
- **Experimento 6**: el descriptor está conformado por el descriptor del experimento 5 + la variable L1 del producto.
- **Experimento 7**: el descriptor está conformado por el descriptor del experimento 6 + la variable L2 del producto.
- **Experimento 8**: el descriptor está conformado por el título del producto + la variable category_id del producto.
- **Experimento 9**: el descriptor está conformado por el título del producto + la variable seller_id del producto.
- **Experimento 10**: el descriptor está conformado por el título del producto + la variable brand_name del producto.
- **Experimento 11**: el descriptor está conformado por el título del producto + la variable L1 del producto.
- **Experimento 12**: el descriptor está conformado por el título del producto + la variable L2 del producto.
- **Experimento 13**: el descriptor está conformado por el título del producto + la variable L3 del producto.

Por cada experimento se genera un gráfico que representa la curva de ahorro. Es decir, se representa la métrica sensible al costo, que se explicó en la sección 3.4.2. El gráfico representa el ahorro de cada iteración evaluada en cada punto del umbral de separación del modelo clasificador. El eje vertical representa la cantidad de dinero (USD) ahorrada en base a la ecuación de costo establecida en la sección 3.4.2. El eje horizontal representa cada umbral de decisión, es decir, cada punto en el eje representa el valor 0.01 y el eje en su totalidad contiene el intervalo [0,1]. También, se puede ver graficada una línea punteada que representa el escenario ideal de clasificación mencionado anteriormente (ahorro potencial – USD 160.731). Esta visualización toma relevancia debido a que cada escenario, que puede adoptar una predicción, tiene un costo y/o ahorro distinto. Si bien una métrica de evaluación tradicional como ROC-AUC, Precision, Recall pueden ser buenos proxys para inferir el rendimiento del modelo, no estarían reflejando el impacto real que este podría realizar.

Dada la cantidad de información, sólo se detallarán los resultados de los experimentos más relevantes. Los resultados de los mejores candidatos de cada experimento se encuentran en el anexo D: Resultados. Para el resto de los resultados se tendrá acceso mediante el repositorio mencionado en el Anexo E: Software.

A continuación, se presentarán los resultados obtenidos de las iteraciones y modelos más relevantes.

4.3. Resultados

Experimento Base

La primera iteración donde solo se utiliza el título como descriptor, se podría considerar como un experimento base sobre el cual se puedan comparar el resto de las iteraciones.

Experimento Base | sb_LR_1					
Entrenamiento	**Transformador**	**Clasificador**	**Mejor umbral**	**% de Ahorro**	**Accuracy**
Train 1	BERT	Regresión Logística	0.53	68%	89%
Train 2			0.18	62%	91%

En este caso, la iteración ganadora es la "sb_LR_1", la cual representa la iteración donde se utilizó el transformador BERT (sb) y el algoritmo clasificador Regresión Logística (LR). Vemos que al utilizar una estrategia de separación de datos Train 2, el porcentaje de ahorro potencial decrece un 4% respecto de la versión que utilizó Train 1.

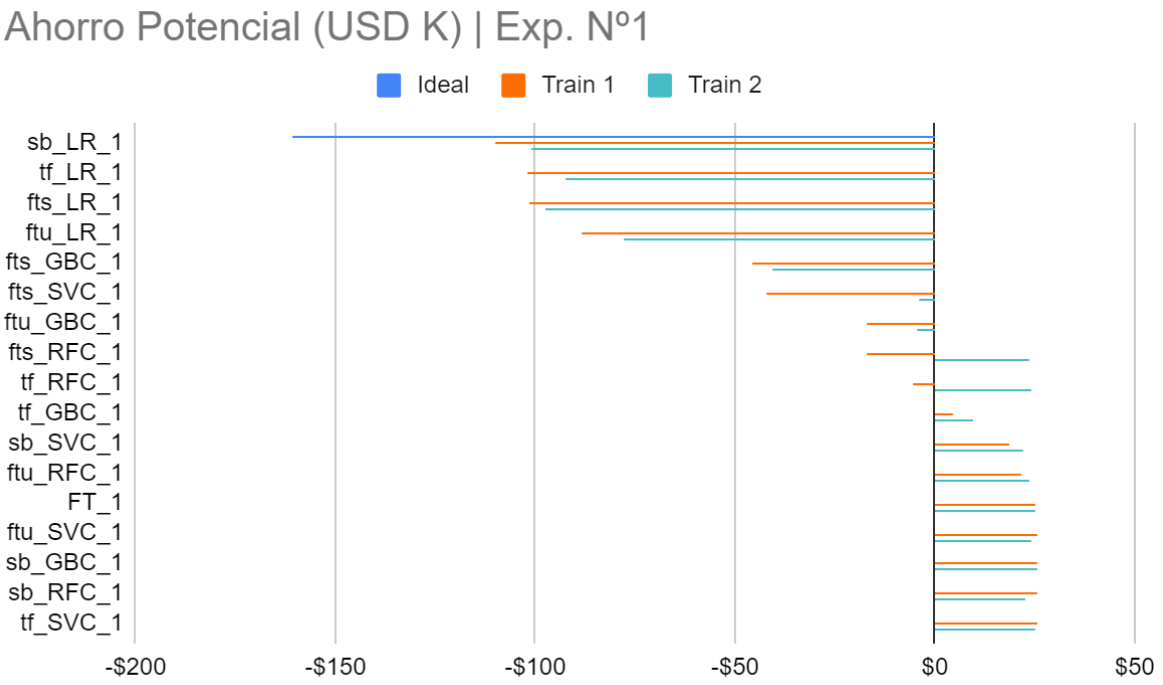


Figura 47: gráfico del ahorro potencial en base al conjunto de test. El eje horizontal representa el ahorro medido en miles de USD. El eje vertical identifica cada iteración dentro del experimento N°1.

En el caso donde se utilizó la estrategia de separación Train 1, el mejor umbral de separación obtenido es el valor 0.53, *Fig. 48*; en este valor se lograría capturar el 68% (USD 110.176) del ahorro potencial total (USD 160.731), *Fig. 47*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 89%. Para la clase No Maquinable, se logra una Precisión del 21% y un Recall del 86%. Además, se logra obtener un AUC de 94%, *Fig. 49*.

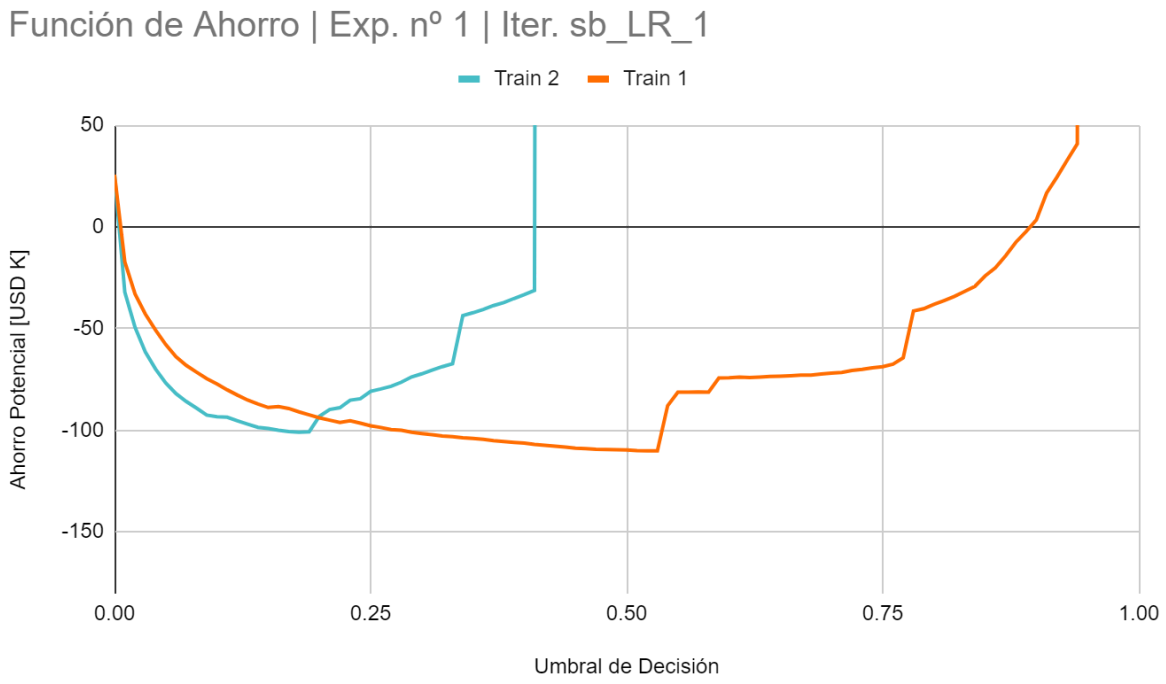


Figura 48: gráfico de la función de ahorro potencial en base al conjunto de test. El eje horizontal representa el umbral de decisión del modelo. El eje vertical presenta el ahorro potencial en miles de USD.

En el caso donde se utilizó la estrategia de separación Train 2, el mejor umbral de separación obtenido es el valor 0.18, *Fig. 48*; en este valor se lograría capturar el 62% (USD 101.191) del ahorro potencial total (USD 160.731), *Fig. 47*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 92%. Para la clase No Maquinable, se logra una Precisión del 25% y un Recall del 79%. Además, se logra obtener un AUC de 91%, *Fig. 49*.

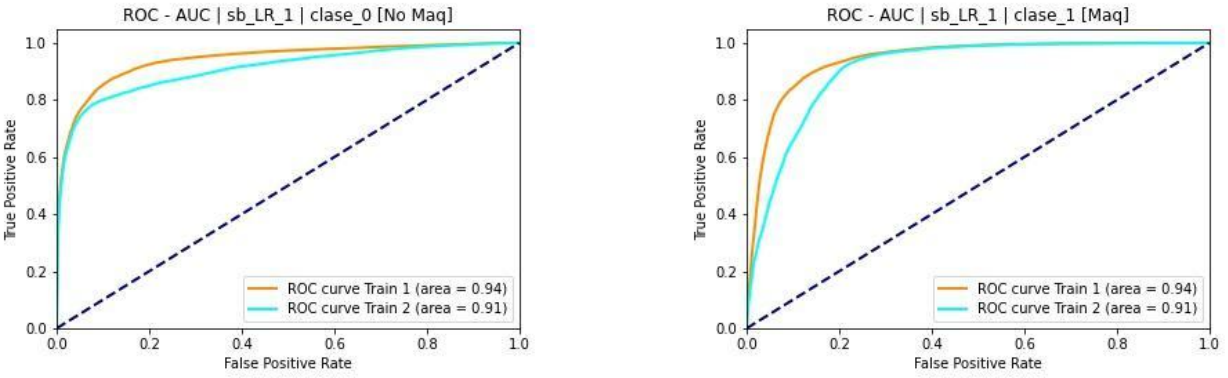


Figura 49: gráficos de las curvas ROC – AUC para cada clase y tipo de entrenamiento. Iteración sb_LR_1

Si bien se evidencia una oportunidad de mejora, es factible inferir que el título es un descriptor muy importante.

Mejores Candidatos

Al margen de toda métrica de rendimiento, el criterio adoptado para elegir los mejores candidatos es cuantificar el ahorro potencial logrado. En la siguiente imagen se muestra el ranking que obtuvieron los mejores rendimientos. *Fig. 50*.

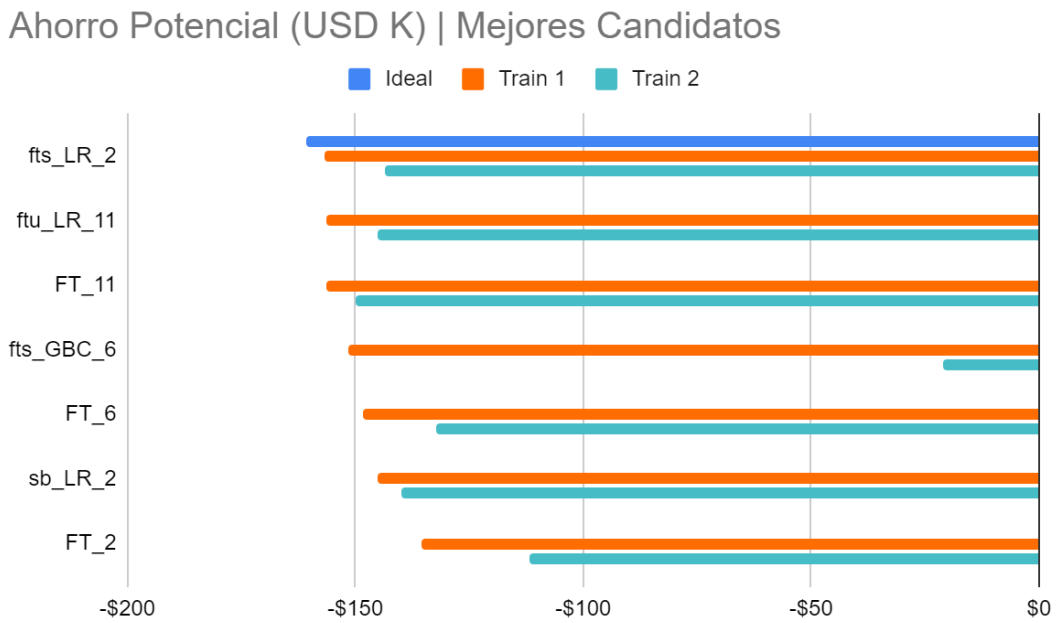


Figura 50: gráfico del ahorro potencial en base al conjunto de test. El eje horizontal representa el ahorro medido en miles de USD. El eje vertical identifica el ranking de los mejores candidatos.

El ranking está ordenado en base al ahorro potencial, de manera decreciente, y con la estrategia de entrenamiento Train 1. Los experimentos N°2 y N°11 tienen la mayoría de las mejores iteraciones. Cabe destacar el rendimiento de la iteración FT_11 que no solo obtiene muy buenos resultados ejecutando Train 1, si no también ejecutando Train 2. A esta le sigue la iteración ftu_LR_11 y en tercer lugar la iteración fts_LR_2.

A continuación, se dará detalle de los experimentos e iteraciones que obtuvieron los mejores resultados.

Experimento N°2: descriptor del experimento 1 + la variable domain_id del producto.

Experimento N°2 | fts_LR_2					
Entrenamiento	**Transformador**	**Clasificador**	**Mejor umbral**	**% de Ahorro**	**Accuracy**
Train 1	Fast Text Supervisado	Regresión Logística	0.57	97%	98%
Train 2			0.19	89%	97%

Para este experimento, la iteración ganadora es la "fts_LR_2", la cual representa la iteración donde se utilizó el transformador Fast Text Supervisado (fts) y el algoritmo clasificador Regresión Logística (LR). Se ubica en la primera posición del ranking, respecto a Train 1, y en la tercera posición respecto a Train 2, *Fig. 48*. Vemos que al utilizar una

estrategia de separación de datos Train 2, el porcentaje de ahorro potencial decrece un 8% respecto de la versión que utilizó Train 1.

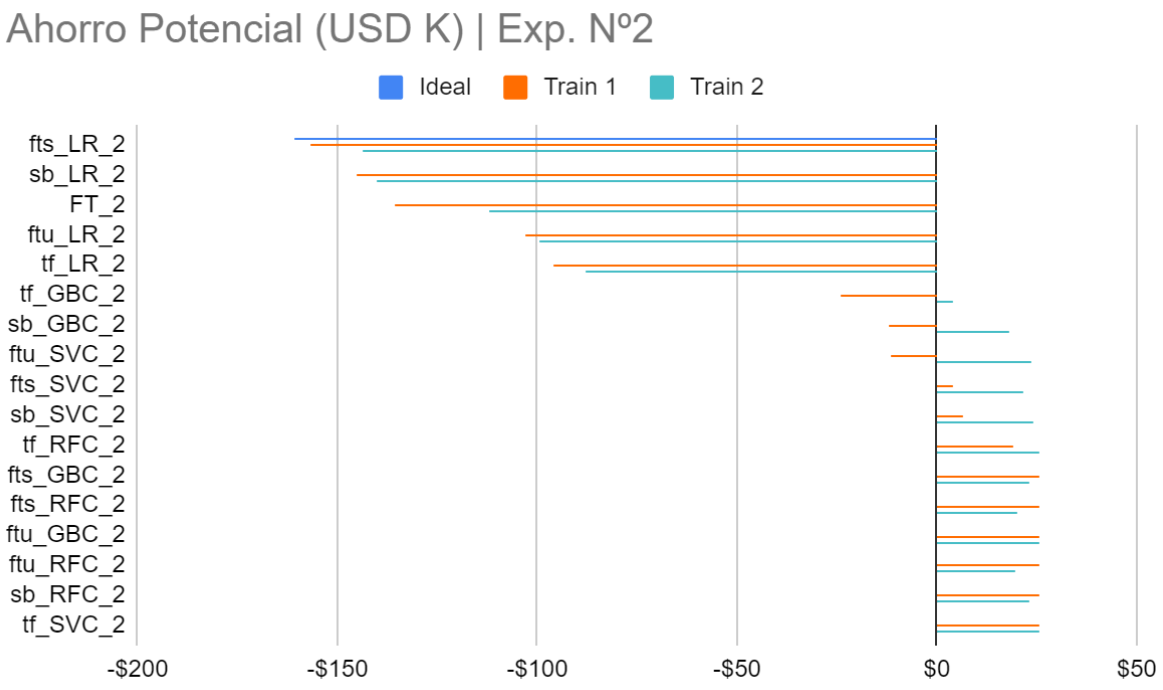


Figura 51: gráfico del ahorro potencial en base al conjunto de test. El eje horizontal representa el ahorro medido en miles de USD. El eje vertical identifica cada iteración dentro del experimento N°2.

En el caso donde se utilizó la estrategia de separación Train 1, el mejor umbral de separación obtenido es el valor 0.57, *Fig. 52*; en este valor se lograría capturar el 97% (USD 156.929) del ahorro potencial total (USD 160.731), *Fig. 51*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 99%. Para la clase No Maquinable, se logra una Precisión del 70% y un Recall del 99%. Además, se logra obtener un AUC de 99%, *Fig. 53*.

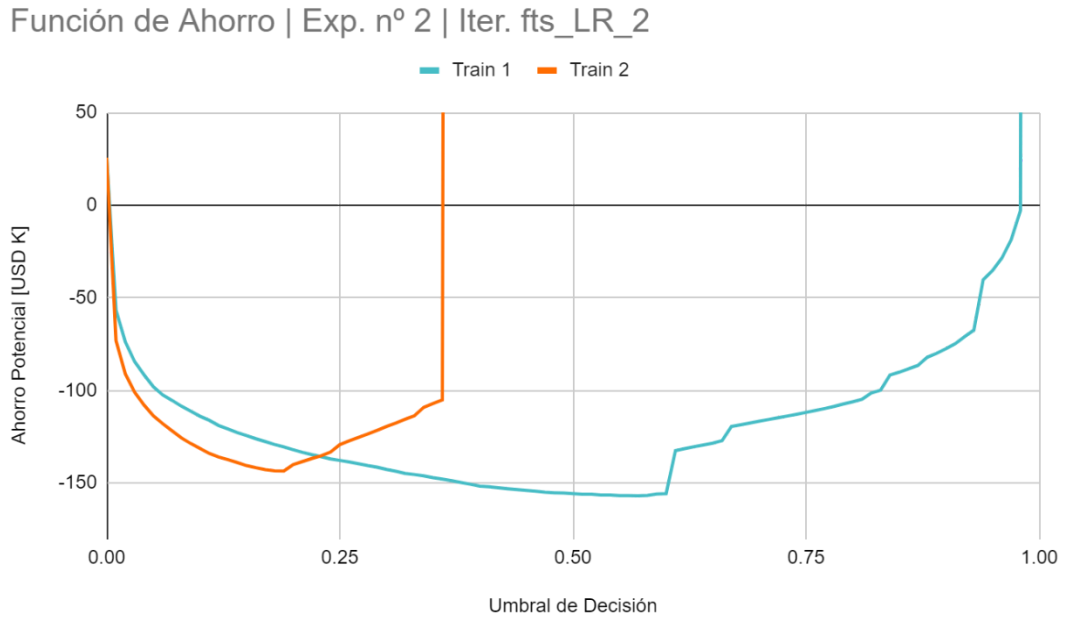


Figura 52: gráfico de la función de ahorro potencial en base al conjunto de test. El eje horizontal representa el umbral de decisión del modelo. El eje vertical presenta el ahorro potencial en miles de USD.

En el caso donde se utilizó la estrategia de separación Train 2, el mejor umbral de separación obtenido es el valor 0.19, *Fig. 52*; en este valor se lograría capturar el 89% (USD 143.524) del ahorro potencial total (USD 160.731), *Fig. 51*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 98%. Para la clase No Maquinable, se logra una Precisión del 57% y un Recall del 93%. Además, se logra obtener un AUC de 99%, *Fig. 53*.

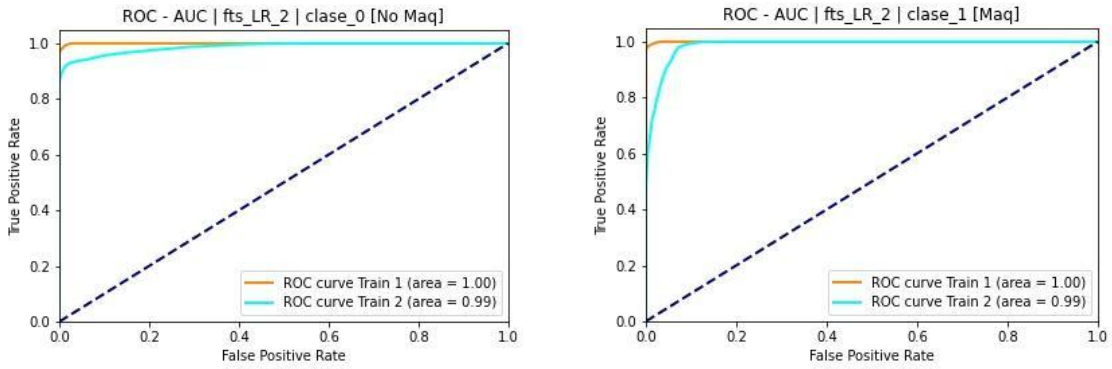


Figura 53: gráficos de las curvas ROC – AUC para cada clase y tipo de entrenamiento. Iteración fts_LR_2

Experimento N°2 | sb_LR_2					
Entrenamiento	**Transformador**	**Clasificador**	**Mejor umbral**	**% de Ahorro**	**Accuracy**
Train 1	BERT	Regresión Logística	0.38	90%	94%
Train 2			0.14	87%	96%

Estando muy cerca del rendimiento obtenido por la mejor iteración del experimento N°2, la segunda mejor iteración es la "sb_LR_2", donde se utilizó el transformador BERT (sb) y el algoritmo clasificador Regresión Logística (LR). Se ubica en la sexta posición del ranking, respecto a Train 1, y en la cuarta posición respecto a Train 2, *Fig. 51*. Vemos que al utilizar una estrategia de separación de datos Train 2, el porcentaje de ahorro potencial decrece un 3% respecto de la versión que utilizó Train 1 y un 2% respecto al mejor candidato de este experimento utilizando Train 2.

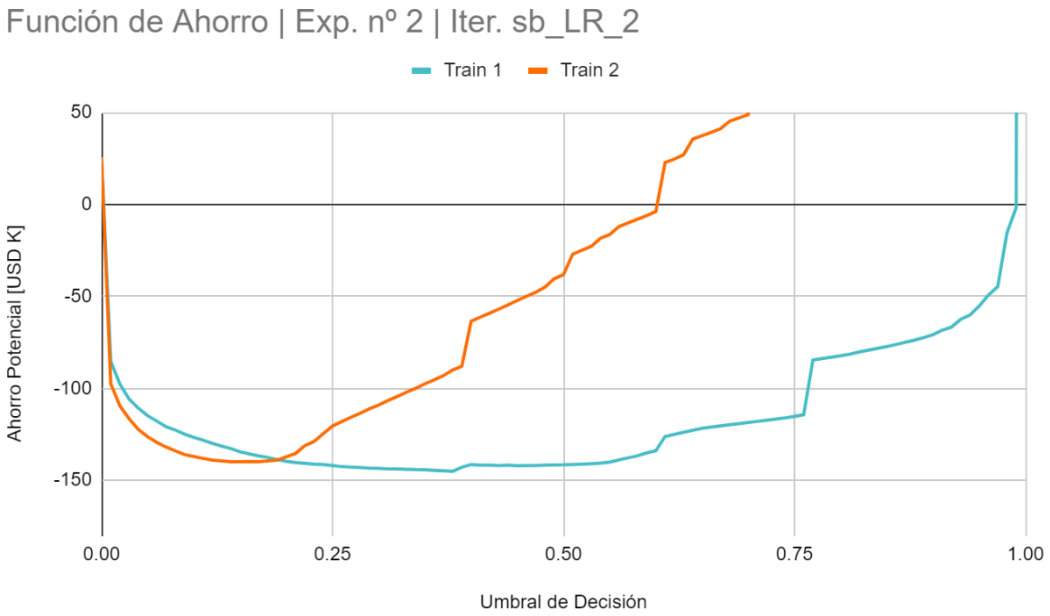


Figura 54: gráfico de la función de ahorro potencial en base al conjunto de test. El eje horizontal representa el umbral de decisión del modelo. El eje vertical presenta el ahorro potencial en miles de USD.

En el caso donde se utilizó la estrategia de separación Train 1, el mejor umbral de separación obtenido es el valor 0.38, *Fig. 54*; en este valor se lograría capturar el 90% (USD 145.139) del ahorro potencial total (USD 160.731), *Fig. 51*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 95%. Para la clase No Maquinable, se logra una Precisión del 39% y un Recall del 96%. Además, se logra obtener un AUC de 99%, *Fig. 55*.

En el caso donde se utilizó la estrategia de separación Train 2, el mejor umbral de separación obtenido es el valor 0.14, *Fig. 54*; en este valor se lograría capturar el 87% (USD 139.937) del ahorro potencial total (USD 160.731), *Fig. 51*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 97%. Para la clase No Maquinable, se logra una Precisión del 49% y un Recall del 92%. Además, se logra obtener un AUC de 99%, *Fig. 55*.

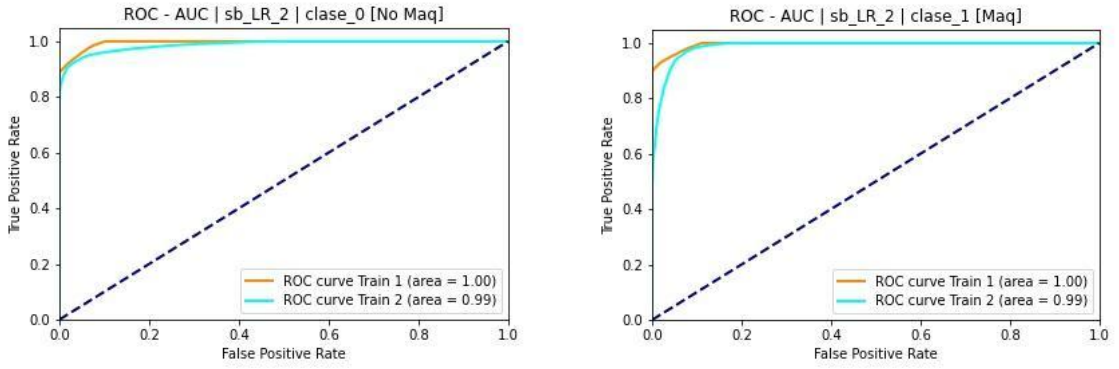


Figura 55: gráficos de las curvas ROC – AUC para cada clase y tipo de entrenamiento. Iteración sb_LR_2

Experimento N°6: descriptor del experimento 5 + la variable L1 del producto.

Experimento N°6 | fts_GBC_6					
Entrenamiento	**Transformador**	**Clasificador**	**Mejor umbral**	**% de Ahorro**	**Accuracy**
Train 1	Fast Text Supervisado	Regresión Logística	0.44	94%	96%
Train 2			0.33	13%	51%

En este experimento, la iteración ganadora es la "fts_GBC_6", la cual representa la iteración donde se utilizó el transformador Fast Text Supervisado (fts) y el algoritmo clasificador Gradient Boosting (GBC). Obtiene la cuarta posición en el ranking con la ejecución Train 1 pero por otro lado, su rendimiento se degrada considerablemente cuando el modelo es ejecutado con Train 2, *Fig. 56*. Vemos que al utilizar una estrategia de separación de datos Train 2, el porcentaje de ahorro potencial decrece drásticamente un 81% respecto de la versión que utilizó Train 1.

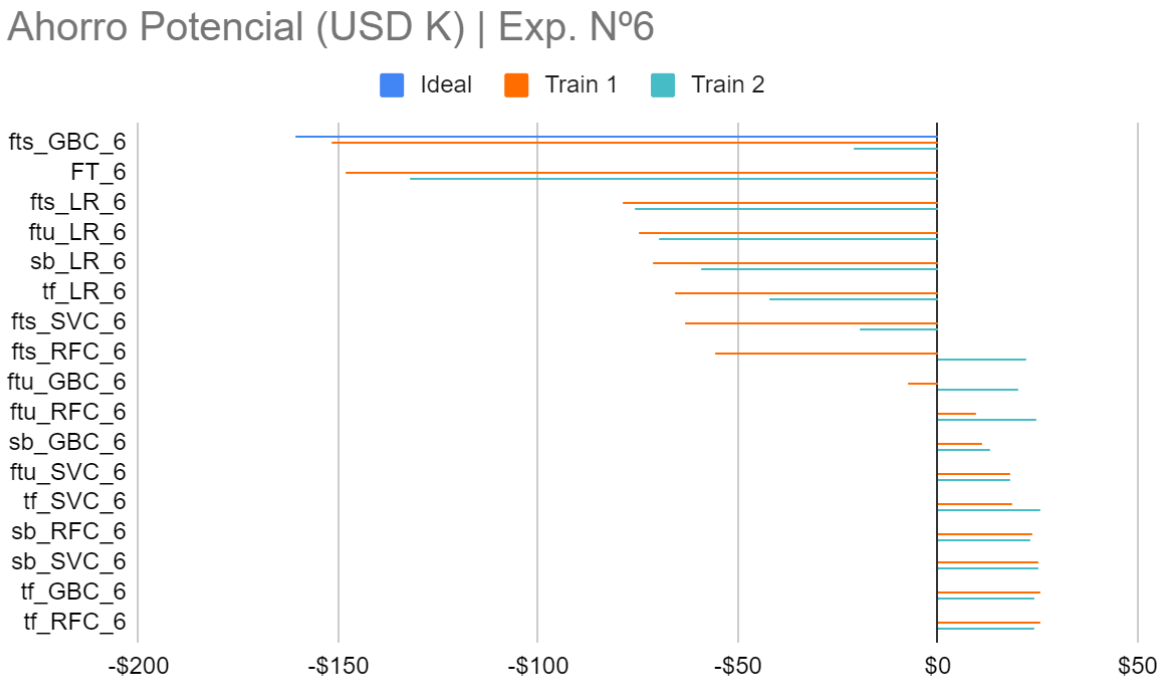


Figura 56: gráfico del ahorro potencial en base al conjunto de test. El eje horizontal representa el ahorro medido en miles de USD. El eje vertical identifica cada iteración dentro del experimento N°6.

En el caso donde se utilizó la estrategia de separación Train 1, el mejor umbral de separación obtenido es el valor 0.44, *Fig. 57*; en este valor se lograría capturar el 94% (USD 151.608) del ahorro potencial total (USD 160.731), *Fig. 56*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 96%. Para la clase No Maquinable, se logra una Precisión del 45% y un Recall del 99%. Además, se logra obtener un AUC de 98%, *Fig. 58*.

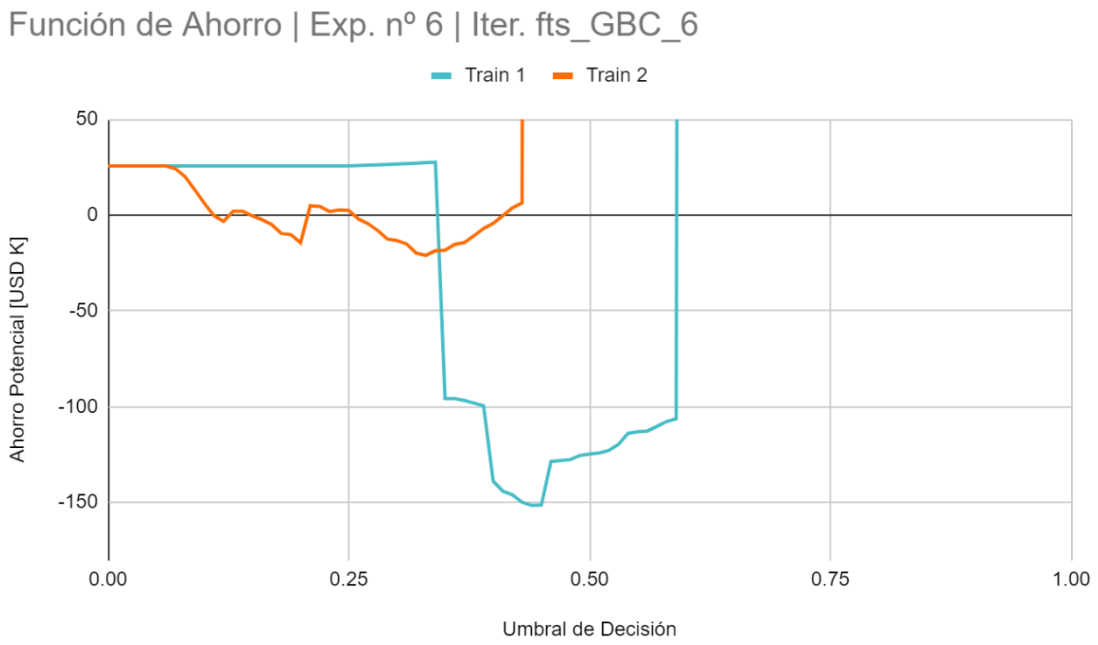


Figura 57: gráfico de la función de ahorro potencial en base al conjunto de test. El eje horizontal representa el umbral de decisión del modelo. El eje vertical presenta el ahorro potencial en miles de USD.

En el caso donde se utilizó la estrategia de separación Train 2, el mejor umbral de separación obtenido es el valor 0.33, *Fig. 57*; en este valor se lograría capturar el 13% (USD 21.036) del ahorro potencial total (USD 160.731), *Fig. 56*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 50%. Para la clase No Maquinable, se logra una Precisión del 6% y un Recall del 95%. Además, se logra obtener un AUC de 78%, *Fig. 58*.

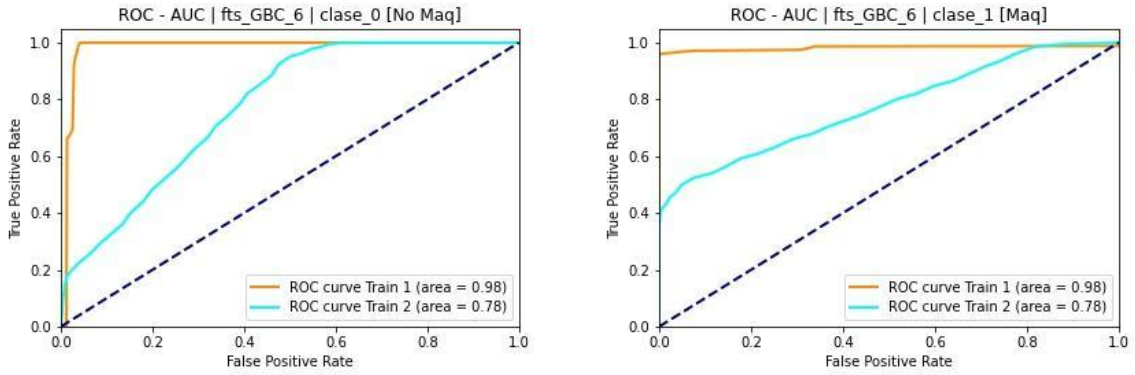


Figura 58: gráficos de las curvas ROC – AUC para cada clase y tipo de entrenamiento. Iteración fts_GBC_6

Experimento N°6 | FT_6					
Entrenamiento	**Transformador**	**Clasificador**	**Mejor umbral**	**% de Ahorro**	**Accuracy**
Train 1	Fast Text	Softmax	0.55	92%	93%
Train 2			0.19	82%	96%

En este experimento, la segunda iteración ganadora es la "FT_6", la cual representa la iteración donde se utilizó todo el pipeline de la librería Fast Text. Se ubica en la quinta posición del ranking, respecto tanto a Train 1 como a Train 2, *Fig. 56*. Vemos que al utilizar una estrategia de separación de datos Train 2, el porcentaje de ahorro potencial disminuye un 10% respecto de la versión que utilizó Train 1 pero, por otro lado, la versión ejecutada con Train 2 ofrece un 69% más de ahorro respecto a la iteración "fts_GBC_6" con la misma versión de entrenamiento.

En el caso donde se utilizó la estrategia de separación Train 1, el mejor umbral de separación obtenido es el valor 0.55, *Fig. 59*; en este valor se lograría capturar el 92% (USD 148.130) del ahorro potencial total (USD 160.731), *Fig. 56*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 94%. Para la clase No Maquinable, se logra una Precisión del 35% y un Recall del 100%. Además, se logra obtener un AUC de 99%, *Fig. 60*.

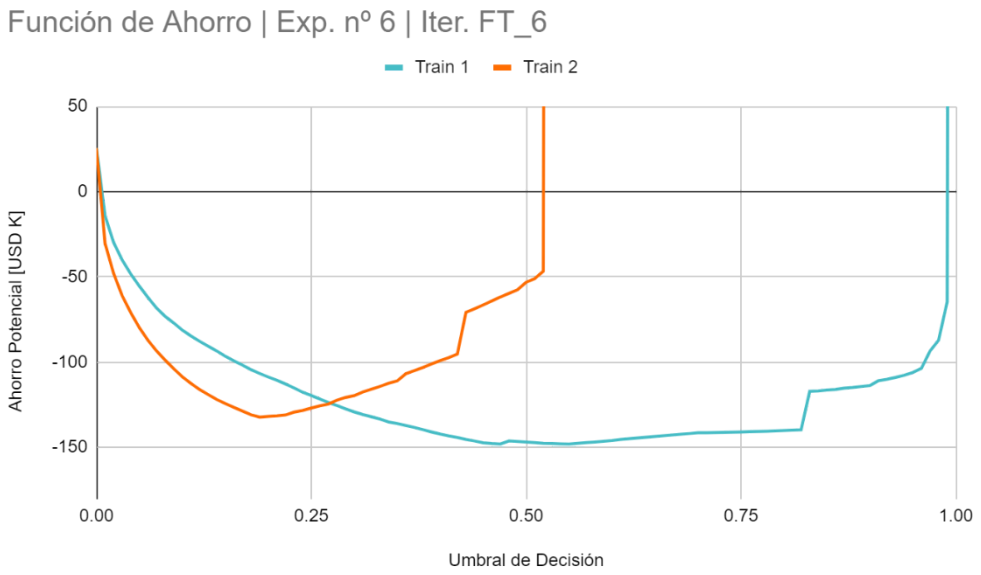


Figura 59: gráfico de la función de ahorro potencial en base al conjunto de test. El eje horizontal representa el umbral de decisión del modelo. El eje vertical presenta el ahorro potencial en miles de USD.

En el caso donde se utilizó la estrategia de separación Train 2, el mejor umbral de separación obtenido es el valor 0.19, *Fig. 59*; en este valor se lograría capturar el 82% (USD 132.279) del ahorro potencial total (USD 160.731), *Fig. 56*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 97%. Para la clase No Maquinable, se logra una Precisión del 48% y un Recall del 88%. Además, se logra obtener un AUC de 98%, *Fig. 60*.

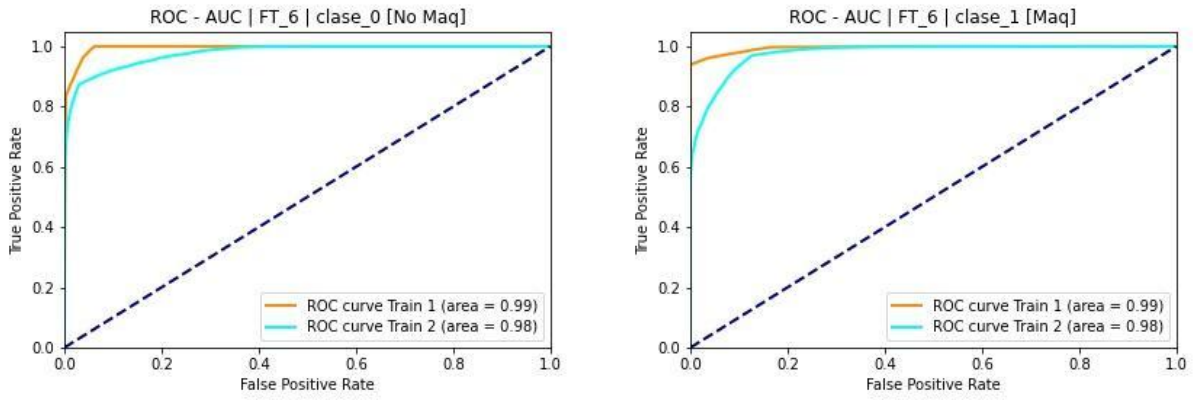


Figura 60: gráficos de las curvas ROC – AUC para cada clase y tipo de entrenamiento. Iteración FT_6

Experimento N°11: el descriptor está conformado por el título del producto + la variable L1 del producto.

Experimento N°11 | ftu_LR_11					
Entrenamiento	**Transformador**	**Clasificador**	**Mejor umbral**	**% de Ahorro**	**Accuracy**
Train 1	Fast Text No Supervisado	Regresión Logística	0.54	97%	98%
Train 2			0.19	90%	97%

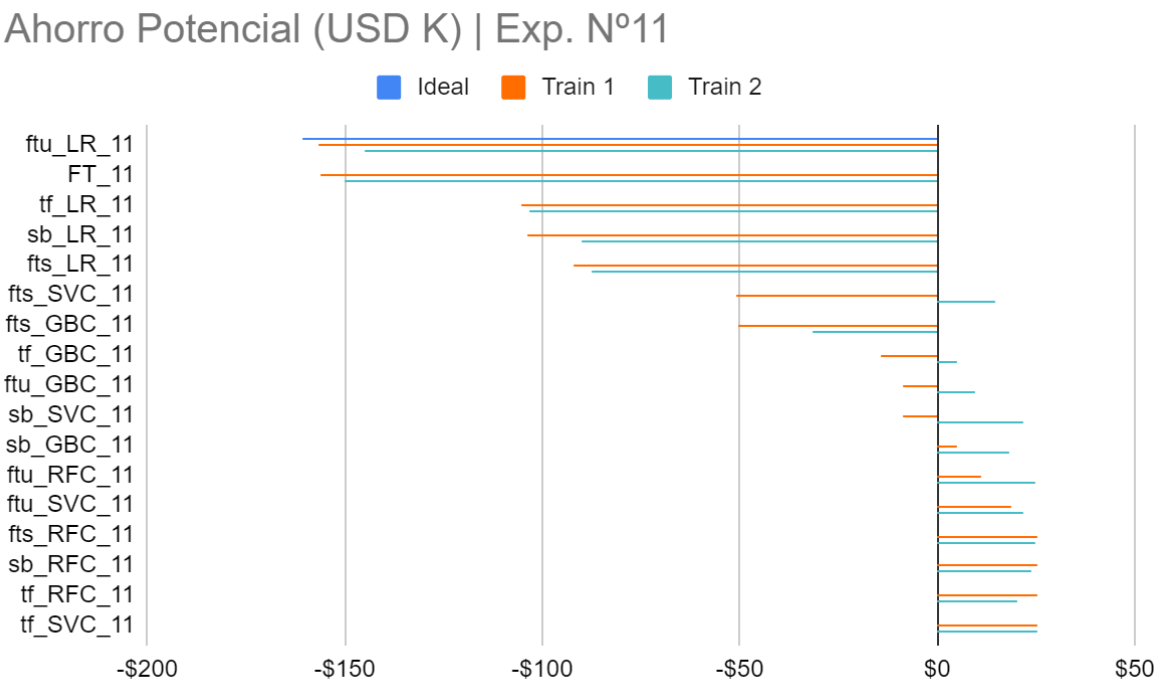


Figura 61: gráfico del ahorro potencial en base al conjunto de test. El eje horizontal representa el ahorro medido en miles de USD. El eje vertical identifica cada iteración dentro del experimento N°11.

En este experimento, la iteración ganadora es la "ftu_LR_11", la cual representa la iteración donde se utilizó el transformador Fast Text No Supervisado (ftu) y el algoritmo clasificador Regresión Logística (LR). Obtiene la segunda posición en el ranking con la ejecución Train 1 y obtiene la tercera posición respecto a la ejecución de Train 2, *Fig. 61*. Vemos que al utilizar una estrategia de separación de datos Train 2, el porcentaje de ahorro potencial decrece un 7% respecto de la versión que utilizó Train 1.

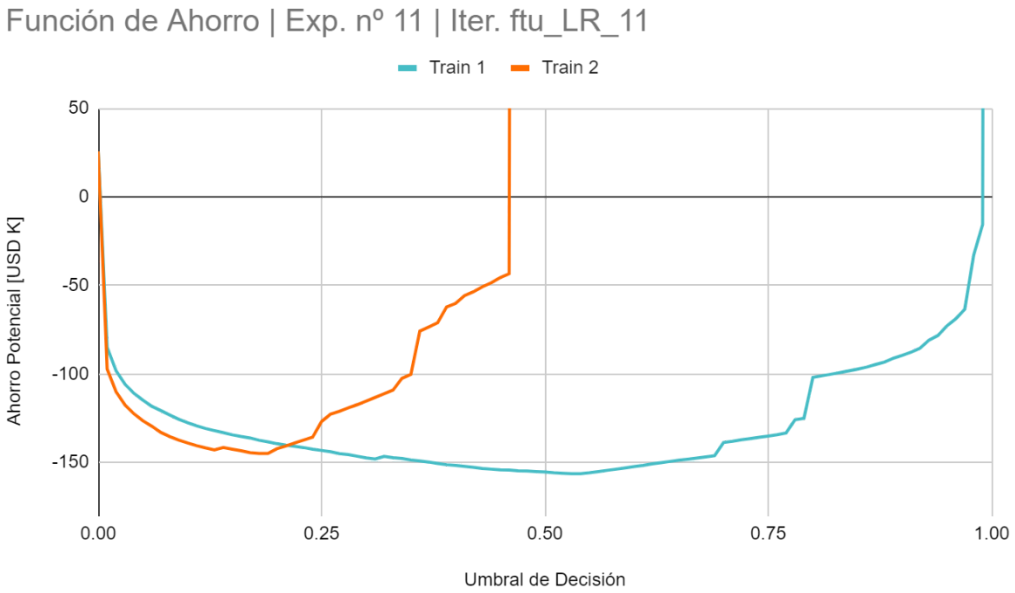


Figura 62: gráfico de la función de ahorro potencial en base al conjunto de test. El eje horizontal representa el umbral de decisión del modelo. El eje vertical presenta el ahorro potencial en miles de USD.

En el caso donde se utilizó la estrategia de separación Train 1, el mejor umbral de separación obtenido es el valor 0.54, *Fig. 62*; en este valor se lograría capturar el 97% (USD 156.398) del ahorro potencial total (USD 160.731), *Fig. 61*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 99%. Para la clase No Maquinable, se logra una Precisión del 73% y un Recall del 99%. Además, se logra obtener un AUC de 99%, *Fig. 63*.

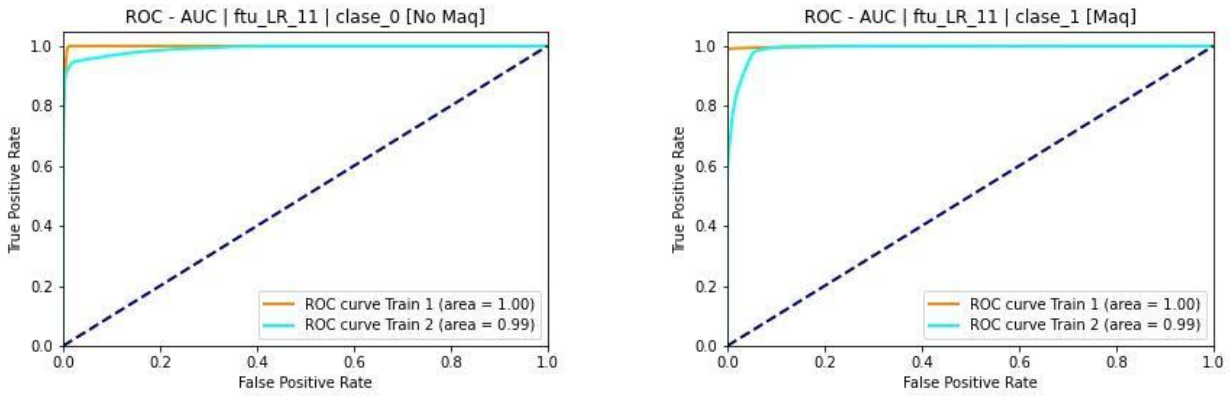


Figura 63: gráficos de las curvas ROC – AUC para cada clase y tipo de entrenamiento. Iteración ftu_LR_11

En el caso donde se utilizó la estrategia de separación Train 2, el mejor umbral de separación obtenido es el valor 0.19, *Fig. 62*; en este valor se lograría capturar el 90% (USD 144.915) del ahorro potencial total (USD 160.731), *Fig. 61*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 98%. Para la clase No Maquinable, se logra una Precisión del 57% y un Recall del 95%. Además, se logra obtener un AUC de 99%, *Fig. 63*.

Experimento N°11 | FT_11					
Entrenamiento	**Transformador**	**Clasificador**	**Mejor umbral**	**% de Ahorro**	**Accuracy**
Train 1	Fast Text	Softmax	0.45	97%	98%
Train 2			0.19	93%	98%

En este experimento, la segunda iteración ganadora es la "FT_11", la cual representa la iteración donde se utilizó todo el pipeline de la librería Fast Text. Se ubica en la tercera posición del ranking respecto a Train 1 y sorprendentemente, obtiene la primera posición respecto a la ejecución con Train 2, *Fig. 61*. Vemos que al utilizar una estrategia de separación de datos Train 2, el porcentaje de ahorro potencial disminuye un 5% respecto de la versión que utilizó Train 1 pero, por otro lado, la versión ejecutada con Train 2 ofrece un 3% de mayor ahorro respecto a la iteración "ftu_LR_11" con la misma versión de entrenamiento.

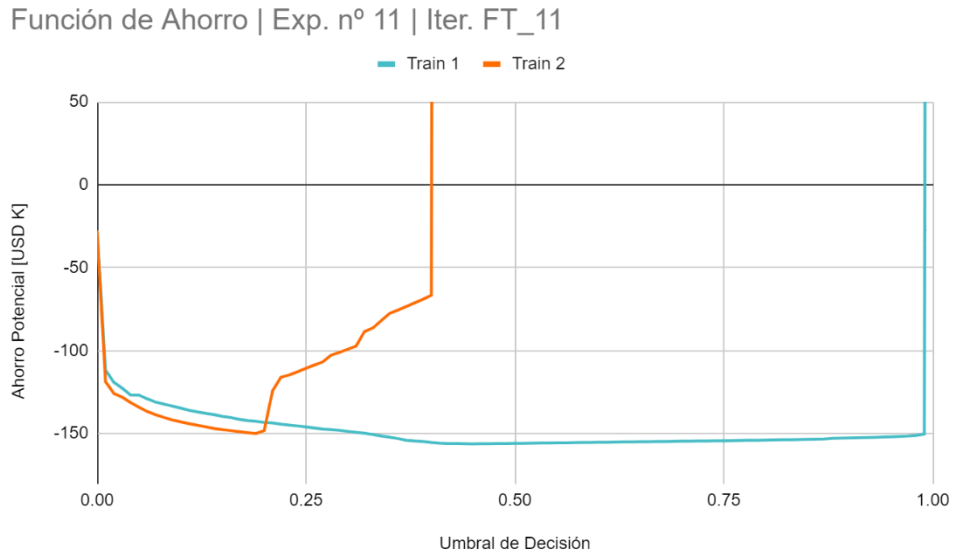


Figura 64: gráfico de la función de ahorro potencial en base al conjunto de test. El eje horizontal representa el umbral de decisión del modelo. El eje vertical presenta el ahorro potencial en miles de USD.

En el caso donde se utilizó la estrategia de separación Train 1, el mejor umbral de separación obtenido es el valor 0.45, *Fig. 64*; en este valor se lograría capturar el 97% (USD 156.209) del ahorro potencial total (USD 160.731), *Fig. 61*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 99%. Para la clase No Maquinable, se logra una Precisión del 71% y un Recall del 100%. Además, se logra obtener un AUC de 99%, *Fig. 65*.

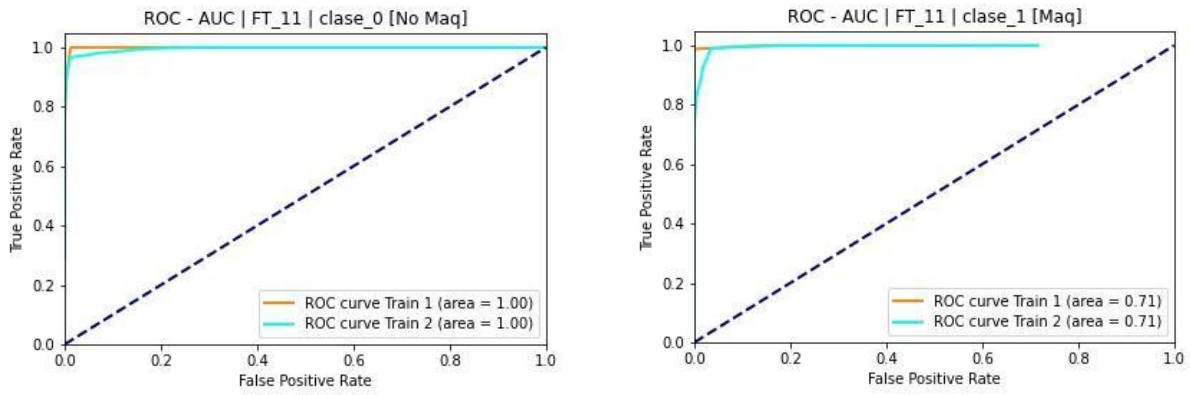


Figura 65: gráficos de las curvas ROC – AUC para cada clase y tipo de entrenamiento.
Iteración FT_11

En el caso donde se utilizó la estrategia de separación Train 2, el mejor umbral de separación obtenido es el valor 0.19, *Fig. 64*; en este valor se lograría capturar el 93% (USD 150.018) del ahorro potencial total (USD 160.731), *Fig. 61*. Por otro lado, para la clase Maquinable, se logra una Precisión del 99% y un Recall del 99%. Para la clase No Maquinable, se logra una Precisión del 72% y un Recall del 97%. Además, se logra obtener un AUC de 99%, *Fig. 65*.

4.4. Interacción de las Variables

Cabe destacar que cada experimento, refleja una selección y permutación de variables predictoras en sí. Esto se da debido a que se van probando distintas combinaciones entre estas que permiten descubrir cuál es la selección de features más adecuada. Los experimentos 1 a 7, van agregando variables predictoras de a una y por otro lado, bajo la hipótesis de que no es necesaria la utilización de todas las variables independientes, se generaron experimentos que complementan la variable predictora principal (título) con otra variable adicional (experimentos 8 a 13). Los efectos y los resultados correspondientes fueron desarrollados en la sección anterior.

Por otro lado, se intenta interpretar mejor la relación que se puede dar entre las variables predictoras y cómo estas aportan valor, en mayor o en menor medida, a la predicción de la clase de un producto. Dado que la mayoría de los mejores modelos tuvieron como clasificador un algoritmo de regresión logística, se optó por cuantificar la magnitud y la dirección de los coeficientes de las variables predictoras utilizadas. Estas son: “componente 1”, “componente 2”, “componente 3”, “componente 4”, “componente 5”, “domain_id”, “category_id”,” brand_name”, “L1”, “L2” y “L3”. Las variables componente 1, 2, 3, 4, y 5 derivan de la transformación que sufre el descriptor al reducir la dimensionalidad de la representación vectorial generada por el transformador, sección 3.3.2. Una de las formas de materializar esto es con la ejecución de un “feature importance”.

La realización del “feature importance” hace referencia a un conjunto de técnicas que asignan una puntuación a las variables predictoras de un modelo en función de su utilidad para predecir una variable objetivo. La ejecución del “feature importance” juega un papel importante en un proyecto de modelado predictivo, dado que aporta información valiosa sobre el modelo, es una base para la reducción de la dimensionalidad y la selección de

variables que pueden mejorar la eficiencia y la eficacia de un modelo predictivo sobre el problema en cuestión.

Hay varios tipos y fuentes de datos que permiten realizar dicho cálculo, aunque los ejemplos más comunes incluyen scores de correlación estadística, coeficientes calculados como parte de modelos lineales, árboles de decisión y de permutation importance. En nuestro caso, dado que el clasificador más relevante dentro los experimentos con mejores candidatos fue la regresión logística, se visualizará la importancia relativa de los coeficientes calculados de dicho modelo, *Fig. 66*.

Vemos que las componentes principales derivadas de la transformación del título juegan un rol importante en la decisión de la predicción, como así también, las variables "domain_id" y "L1". Estas aparecen en los experimentos ganadores N°2, N°6 y N°11.

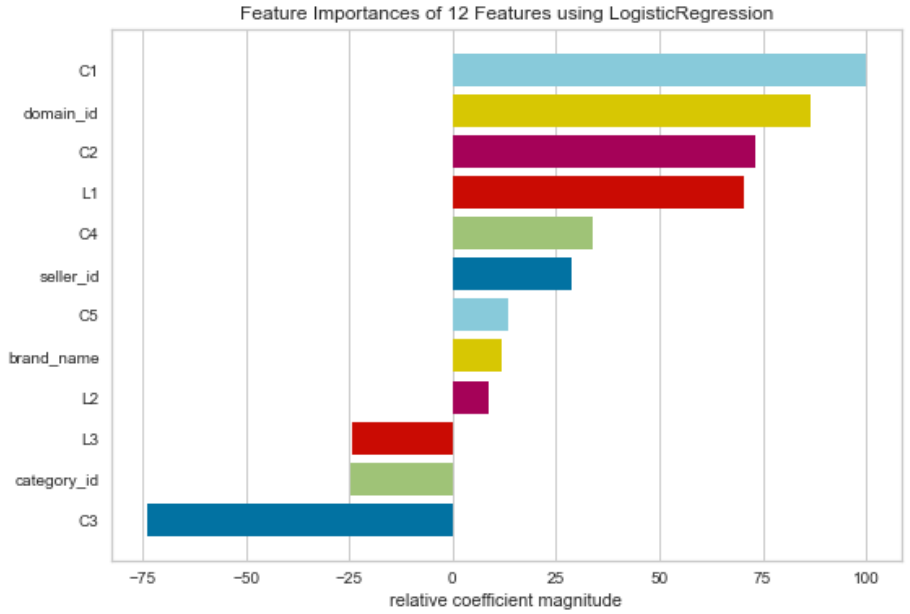


Figura 66: gráficos de las curvas feature importance para un modelo de regresión logística.

Otro método utilizado para intentar cuantificar la contribución de las variables predictoras se denomina "Shapley Additive Explanations" (SHAP). Básicamente, SHAP utiliza la teoría de juegos para capturar las contribuciones marginales de cada predictor. Se generan distintas coaliciones de predictores, se ejecuta un muestro de estas dado que sería muy costoso realizar el cómputo para todas y se calcula el cambio en el score del modelo para cada variación. Luego, estos cambios se promedian para cada variable para crear un score resumen. También brinda información e interpretabilidad a nivel local, es decir, sobre una instancia particular. SHAP es capaz de cumplir con tres características muy deseables referidas a la interpretabilidad. Por un lado, cualquier variable que no tenga ningún efecto sobre la predicción deberá tener un valor de Shapley de cero (importancia). Por otro lado, si dos covariables agregan el mismo valor a la predicción, entonces sus valores de Shapley deberían ser los mismos (sustituibilidad). Por último, si se tienen dos o más predicciones que podrían fusionarse, se debería poder simplemente agregar los valores de Shapley (sumar) que se calcularon en las predicciones individuales (Aditividad). Esta última es muy útil en nuestro caso debido a que todas las variables predictoras utilizadas en el modelo son categóricas. Por lo tanto, esta característica permite que los valores de Shapley, para cada variable categórica utilizada y transformada con la técnica "one-hot-encoding", se sumen

y conformen una representación del valor global de Shapley para toda la variable en su conjunto.

A continuación, se muestra un gráfico que resume los valores principales de las diferentes variables utilizadas en el ejemplo, están ordenados de mayor a menor en base a la contribución generada en el modelo. Cada punto en el gráfico corresponde a una observación o instancia, es decir, se visualiza la distribución del valor shap para cada predictor. Cómo se puede ver, las variables principales antes mencionadas también aparecen en la parte superior del gráfico indicando que tienen mayor ponderación, *Fig. 67*.

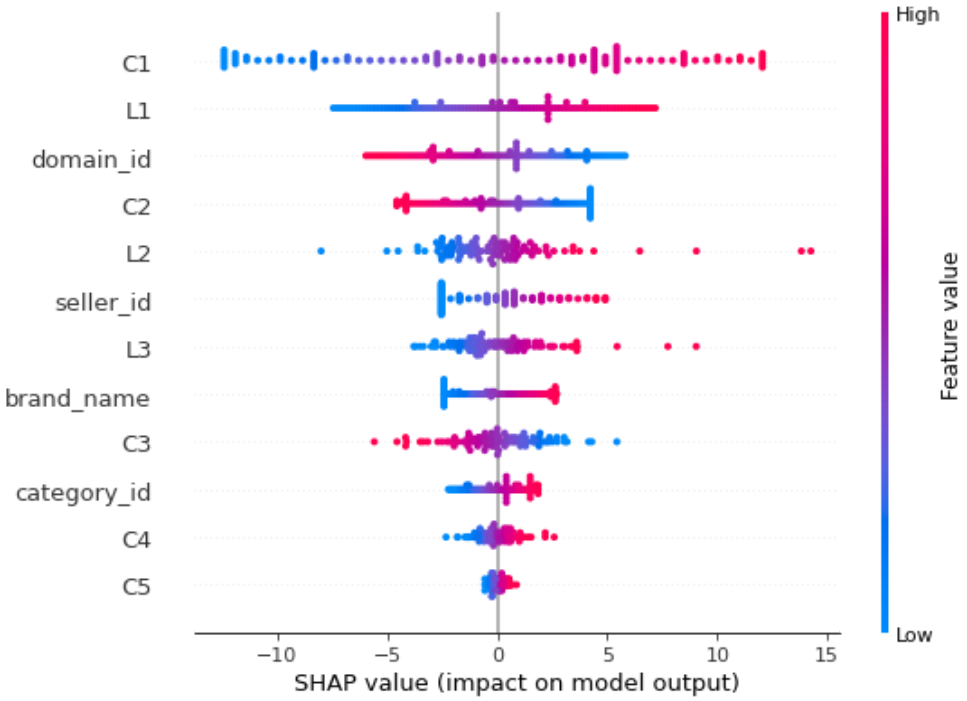


Figura 67: gráficos resumen de SHAP de las variables utilizadas en el modelo.

Mientras que un gráfico de resumen SHAP ofrece una descripción general de cada covariable, un gráfico de dependencia SHAP muestra cómo varía la salida del modelo según el valor de un variable elegida. En el eje vertical izquierdo se puede visualizar el valor SHAP y en el eje horizontal el valor de la variable en cuestión. De manera complementaria, en el vertical derecho, también se indica otra variable con la que posiblemente interactúa la variable elegida. La función utilizada para colorear se elige automáticamente para resaltar lo que podría estar impulsando estas interacciones. Algo que vale la pena aclarar es que existe un gran supuesto detrás de este gráfico. El cálculo de importancia se basa en que la variable de interés no está correlacionada con todas las demás variables.

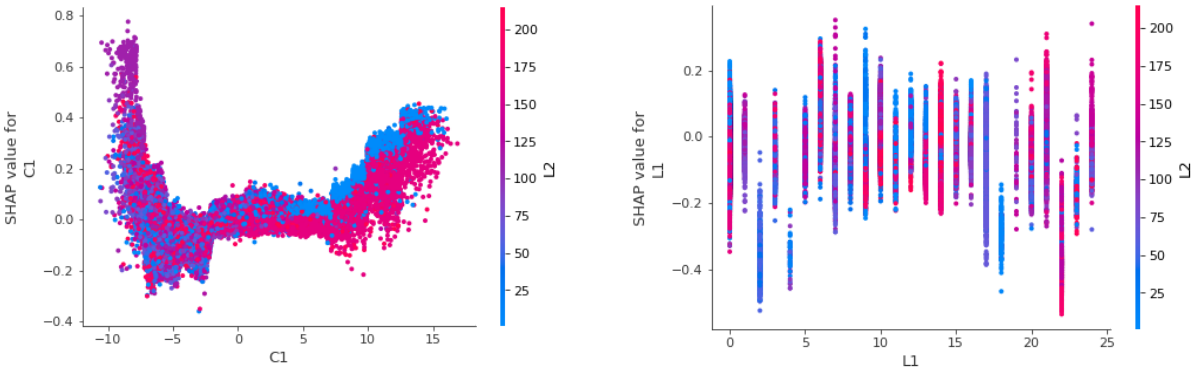


Figura 68: gráficos de dependencia SHAP de las variables "C1" y "L1" utilizadas en el modelo. Se puede ver la variación de valor SHAP de acuerdo con el valor de cada variable. Posiblemente, las variables "C1" y "L1" interactúan mayormente con la variable "L2".

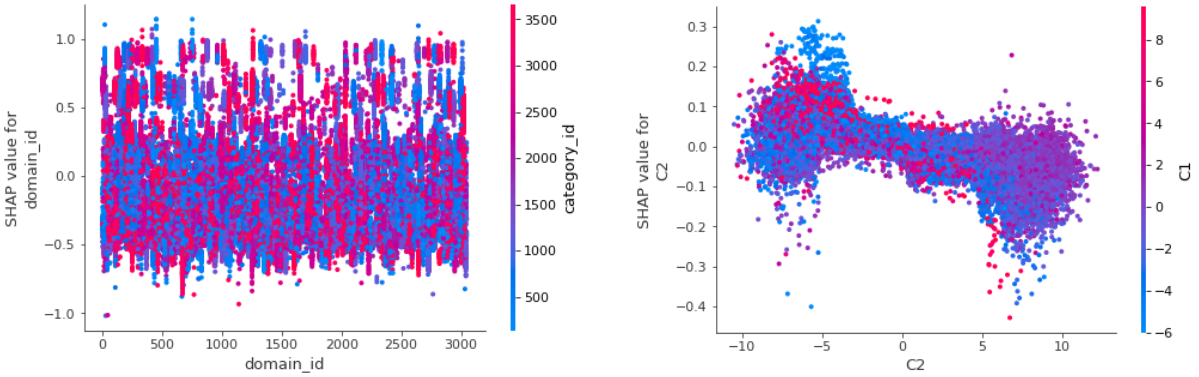


Figura 69: gráficos de dependencia SHAP de las variables "domain_id" y "C2" utilizadas en el modelo. Se puede ver la variación de valor SHAP de acuerdo con el valor de cada variable. Posiblemente, la variable "domain_id" interactúe con la variable "category_id" y "C2" con la variable "C1".

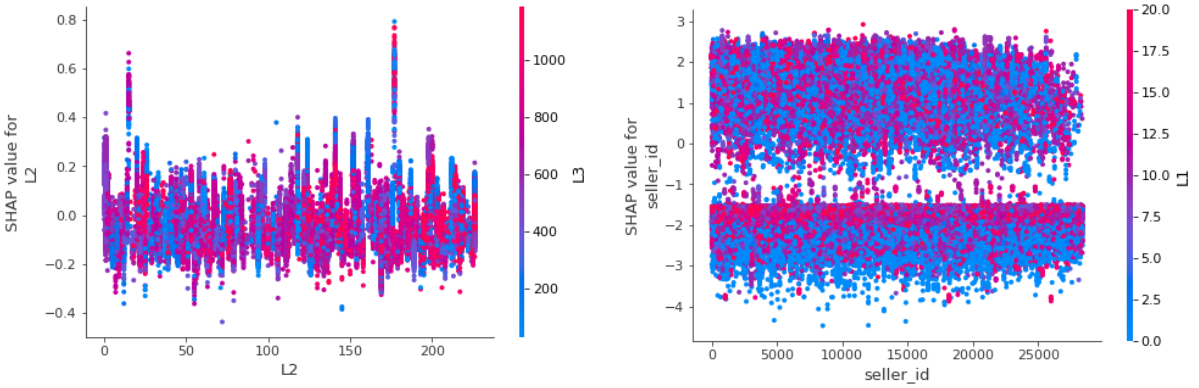


Figura 70: gráficos de dependencia SHAP de las variables "L2" y "seller_id" utilizadas en el modelo. Se puede ver la variación de valor SHAP de acuerdo con el valor de cada variable. Posiblemente, la variable "L2" interactúe con la variable "L3" y "seller_id" con la variable "L1".

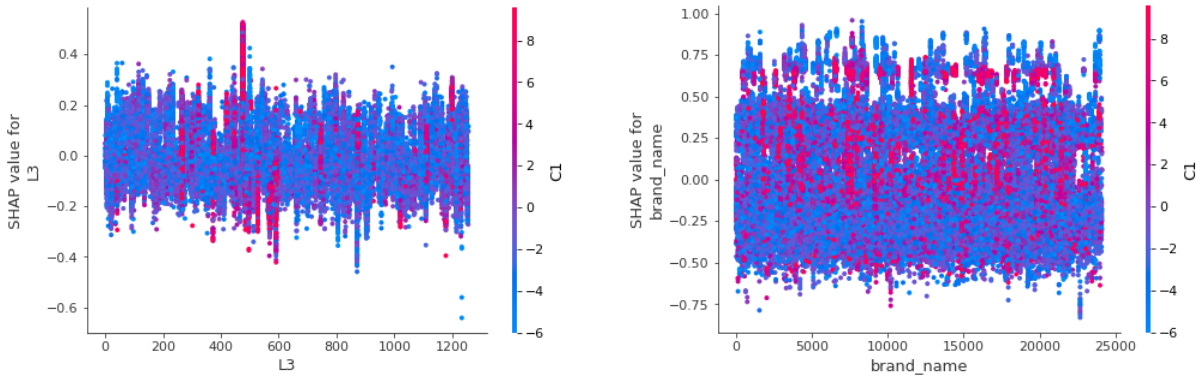


Figura 71: gráficos de dependencia SHAP de las variables "L3" y "brand_name" utilizadas en el modelo. Se puede ver la variación de valor SHAP de acuerdo con el valor de cada variable. Posiblemente, las variables "L3" y "brand_name" interactúan mayormente con la variable "C1".

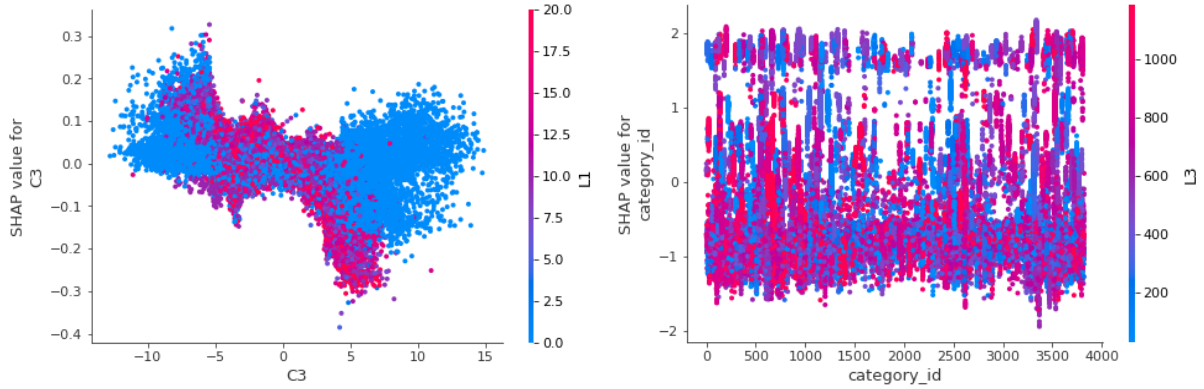


Figura 72: gráficos de dependencia SHAP de las variables "C3" y "category_id" utilizadas en el modelo. Se puede ver la variación de valor SHAP de acuerdo con el valor de cada variable. Posiblemente, la variable "C3" interactúe con la variable "L1" y "category_id" con la variable "L3".

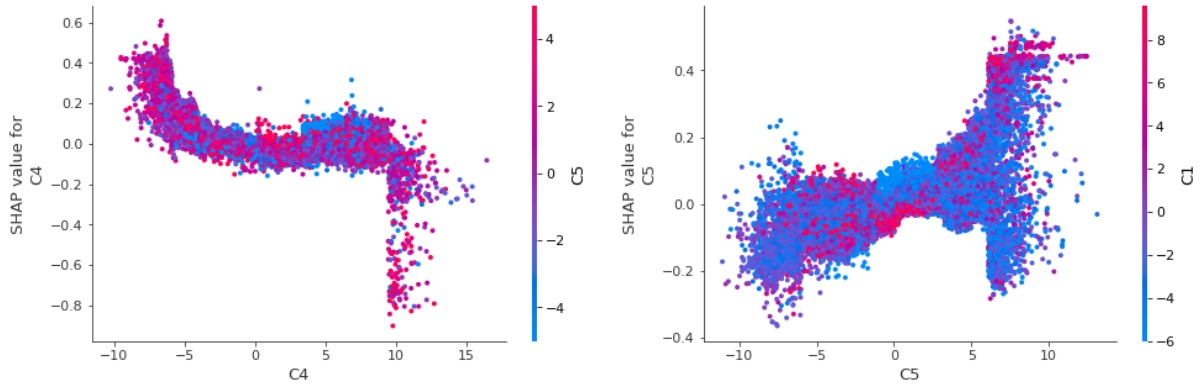


Figura 73: gráficos de dependencia SHAP de las variables "C4" y "C5" utilizadas en el modelo. Se puede ver la variación de valor SHAP de acuerdo con el valor de cada variable. Posiblemente, la variable "C4" interactúe con la variable "C5" y "C5" con la variable "C1".

Por último, daremos ejemplo de algunas instancias puntuales, una por cada clase de producto (Maquinable | No maquinable). A través SHAP también es factible cuantificar el valor agregado que tiene cada variable predictora para una instancia en particular. El gráfico muestra aquellas que son las más importantes y los valores asociados.

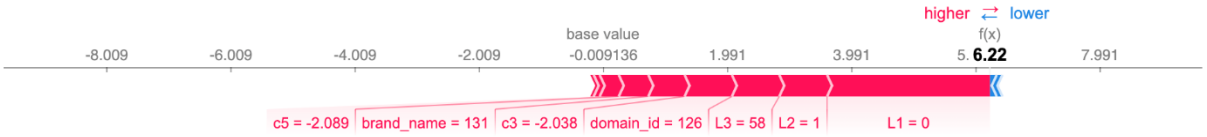


Figura 74: gráfico SHAP force. Se destacan las variables predictoras principales. Clase: Maquinable.

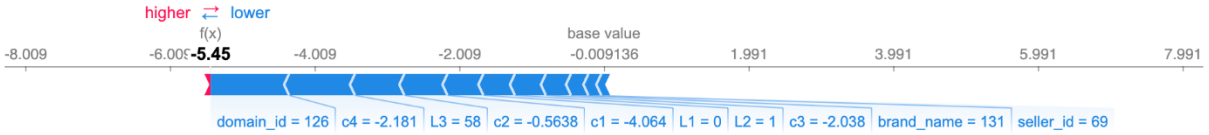


Figura 75: gráfico SHAP force. Se destacan las variables predictoras principales. Clase: No Maquinable.

5. Conclusiones

Se puede ver que los 3 experimentos explicados anteriormente han demostrado que es factible modelar el problema, obtener buenos resultados con calidad predictiva aceptable y acercándose consistentemente al valor de ahorro total en base a la función de ahorro evaluada en el conjunto de test. Se podría inferir que la variable L1, la cual representa una descripción de bajo nivel de la categoría de los productos, tuvo una importancia considerable dado que esta aparece en los experimentos 6 y 11. Por otro lado, la variable domain_id, la cual representa una descripción de alto nivel de la categoría de los productos, tuvo importancia para la iteración (fts_LR_2) donde se utilizó el transformado de texto de Fast Text Supervisado.

Las iteraciones antes descritas son parte del conjunto de las 260 iteraciones realizadas en todo el pipeline de ejecución. Fueron ejecutados 13 experimentos, con dos estrategias de entrenamiento distintas tal como se describe en la sección 2.5, y en cada uno de estos, fueron ejecutados 5 transformadores textuales distintos, tal como se describió en la sección 3.3. Además, por cada uno de los transformadores, fueron configurados y evaluados 4 clasificadores distintos mediante la técnica de Random Search descrita en la sección 3.4.1. Luego de obtener los resultados calculados con el conjunto de datos de test, cada una de las iteraciones fueron evaluadas con la métrica sensible al costo, explicada en la sección 3.4.2.

Uno de los aspectos importantes para tener en cuenta dentro de los resultados es el tiempo insumido para cada una de las iteraciones. En este caso, cabe destacar la superioridad de Fast Text (iteración FT_11) versus el resto de las iteraciones realizadas con los pipelines de entrenamiento del resto de los transformadores. Todas las iteraciones de Fast Text fueron seteadas para que el entrenamiento y la búsqueda de hiperparámetros tenga una duración exacta de 10 minutos; este parámetro es configurable en base a la decisión del analista. Haciendo hincapié en los mejores candidatos antes presentados, la iteración fts_GBC_6 realizó el entrenamiento en 28 min, la iteración fts_LR_2 realizó el entrenamiento en 22

min y la iteración ftu_LR_11 realizó el entrenamiento en 360 min. Por otro lado, el código pertinente al pipeline de Fast Text fue desarrollado con una baja cantidad de líneas respecto al resto de los pipelines ejecutados por el resto de los clasificadores. Cómo se puede ver, existe una ganancia muy importante en el tiempo invertido en el desarrollo y entrenamiento de la solución; este aspecto tiene mucha importancia si esta solución se pensara integrar en un proceso productivo.

El proyecto actual ha demostrado que es factible aplicar las técnicas de aprendizaje automático dedicadas al modelado del lenguaje natural, para abordar la oportunidad presentada en este proyecto. Al experimentar con diferentes arquitecturas de modelado, pudimos llegar a una serie de modelos que combinaban diferentes tipos de variables, los cuales permitieron representar adecuadamente el dominio del problema, generalizarlo y obtener una buena calidad predictiva. Se construyó una serie de experimentos basados en los descriptores textuales donde quedó demostrado que es posible obtener buenos desempeños en un conjunto de datos de testeo el cual conserva la distribución real del dominio.

La optimización de hiperparámetros con el enfoque de Random Search demostró ser de utilidad para obtener una configuración óptima en base a un espacio de parámetros establecido. El espacio de hiperparámetros establecido quedará asentado en el anexo C: Hiperparámetros. Si bien se ha utilizado la técnica antes mencionada dado que se conoce su rendimiento y robustez, existen varios enfoques que quedarían pendientes de ser probados durante la ejecución del entrenamiento que podrían conducir a una mejora en el rendimiento presentado.

Por otra parte, cabe destacar el rendimiento de la librería de Fast Text. Se quiso testear la simplicidad, la agilidad y la calidad predictiva que esta herramienta podría obtener. En menos de 30 líneas de código fue posible construir un programa capaz de utilizar solo los descriptores textuales para captar la información esencial de los datos que permitió identificar las clases de los productos. Con muy pocos hiperparámetros fue posible encontrar iteraciones que logren resultados muy buenos y además, esta librería contempla un método automático de optimización de hiperparámetros con el solo hecho de configurar una métrica objetivo y el tiempo de entrenamiento. Además, tal como se mencionó en la sección 4.3, se consumió muy poco de entrenamiento para obtener los resultados.

Si bien Fast Text obtuvo buenos resultados, no hay que perder de vista lo construido en el pipeline de entrenamiento con el resto de los transformadores y algoritmos clasificadores. Se sabe que un conjunto de datos muy desbalanceado representa un desafío muy importante en términos de rendimiento. Las técnicas combinadas de Data Augmentation y Data Reduction explicadas en la sección 3.3.2, permitieron obtener un conjunto de datos de entrenamiento balanceado y además, ponderar de mejor manera los ejemplos representados en la clase de productos No Maquinables. Cómo se puede ver en las tablas de resultados de las iteraciones correspondiente a los mejores candidatos, sección 4.3, los modelos lograron obtener resultados muy buenos. Si bien para la clase de productos maquinables se obtienen rendimientos un poco más bajos, siguen siendo resultados muy buenos.

En resumen, como logros del proyecto en términos productivos, vemos que se podría alcanzar un ahorro del 97% respecto del ahorro total potencial que presenta el conjunto de datos de test.

Podemos decir que fue factible encontrar evidencia a favor de la hipótesis donde se planteó que la aplicación del aprendizaje automático era adecuada para procesar la gran cantidad de datos que persisten el sistema de publicación de los productos, de los cuales se obtienen las variables principales para impulsar una solución que permita la identificación temprana del tipo de producto a enviar. De esta manera, el proceso de envío incurriría en menores costos y podría ser más eficiente.

Para abordar esta problemática, se trabajó con un conjunto de datos perteneciente a una de las empresas de comercio electrónico más importantes de Latinoamérica, con el fin de desarrollar un clasificador que pudiera predecir si un producto es maquinable o no, con el objetivo de ejecutar el proceso logístico de manera correcta.

A través de la exploración, pudimos encontrar ciertos patrones en los datos que nos permitieron comprender con qué información debería alimentarse el modelo. El modelo construido tiene en cuenta diferentes tipos de variables que conforman el espectro de descriptores textuales que permiten la identificación de un producto.

Experimentamos con diferentes arquitecturas de modelado, técnicas de ingeniería de features, algoritmos de aprendizaje supervisado y configuración de hiperparámetros. Los clasificadores fueron optimizados utilizando un enfoque Random Search (búsqueda aleatoria) y se aplicó una estrategia de validación cruzada que combinaba los métodos de k-fold y la separación del conjunto de datos en entrenamiento, validación y test en el orden correspondiente.

El entregable de este trabajo es una serie de modelos que, dado un conjunto de características del producto, puede predecir las probabilidades de que dicho producto sea de la clase maquinable o no maquinable.

6. Recomendaciones

Si bien se obtuvieron buenos resultados en este proyecto, sabemos que es una prueba de concepto que está ligada a supuestos para facilitar el desarrollo de esta y por otra parte, está atada a una serie de limitaciones que se podrían seguir trabajando en caso que se quisiera convertir en un proyecto productivo.

6.1. Limitaciones

Rendimiento

En primer lugar, si bien el rendimiento alcanzado por los mejores modelos candidatos fueron muy prometedores, no se podría asegurar que se obtengan resultados similares en un escenario real dado que hay muchos datos que no conocemos aún y tampoco podríamos asegurar la representatividad del dominio tal como es en el conjunto de datos utilizados para este proyecto. Es posible que la calidad predictiva se pueda mantener, incorporando nuevos datos en el modelo y realizando el entrenamiento de manera periódica. Sin

embargo, algo que no estuvo dentro del alcance de este proyecto, es el análisis del cambio de distribución de los datos en una franja temporal extensa que permitiese identificar las variaciones que se pueden producir en el conjunto de datos debido al comportamiento de compra de los compradores, aparición de nuevos productos, modas, etc. Además, se podría continuar experimentando con diferentes representaciones de palabras o algoritmos clasificadores.

Respecto a los datos del modelo, hay mucha información relativa a los usuarios, compradores y vendedores, que fue descartada de antemano para este proyecto debido a las políticas de privacidad de los datos que aseguran la información privada de todos y cada uno de los usuarios que utilizan la plataforma. Existe la posibilidad y es una hipótesis, que dicha información podría aportar mucho valor predictivo.

Con respecto a la transformación de los datos, se puede hacer más en términos de ingeniería de features. Por ejemplo, probando una codificación diferente para variables categóricas. Featuring binning y hashing son técnicas que han demostrado contribuir con la minería de datos y particularmente, funcionan muy bien con algoritmos basados en árboles. Otra opción es probar distintas técnicas de representación de texto, si bien hemos aplicado cinco tipos de transformadores distintos, hay una variedad de técnicas que no han sido probadas, tales como TF - IDF, Glove, Word2vec, entre otras. Otra opción sería desarrollar un modelo de aprendizaje supervisado donde sean utilizadas las redes neuronales, dado que presentan grandes ventajas para aprender relaciones complejas de los datos. Sin embargo, esto requeriría un gran esfuerzo de desarrollo y un conjunto de datos muy grande con los que se pudiera entrenar dicho modelo. Otra opción, podría ser la utilización del aprendizaje por transferencia utilizando la capacidad predictiva de una red neuronal entrenada previamente con un conjunto de datos que presenta características similares respecto a nuestra problemática.

Además de las limitaciones mencionadas, cabe destacar que la integración de un modelo predictivo puede presentar restricciones en torno al proceso productivo debido a que este debería cumplir requisitos funcionales que aseguren la experiencia del usuario. Como, por ejemplo, el tiempo de respuestas de la predicción. Esto representa un reto de ingeniería, un análisis y un desarrollo de una arquitectura complementaria que no está contemplada en esta instancia.

Sabemos que para la búsqueda y optimización de hiperparámetros se utilizó la técnica Random Search, la cual se sigue utilizando hoy en día y presenta buenos resultados. Sin embargo, existen técnicas más avanzadas con las cuales se podrían obtener mejores resultados. Una de estas, es la optimización bayesiana[61]. El método funciona construyendo una estimación de la función de probabilidad del rendimiento del modelo dados ciertos valores de hiperparámetros. Luego, se optimiza esta función y se encuentra la combinación de hiperparámetros que minimiza la probabilidad subrogada del rendimiento. Al aplicar estos hiperparámetros a la verdadera función objetivo, se actualiza la función de probabilidad estimada incorporando estos últimos resultados y el proceso se repite hasta que se cumplen ciertos criterios de éxito.

Por último y no menos importante, la función de pérdida utilizada para ajustar los modelos durante la fase de entrenamiento utiliza como métrica la precisión media (mean accuracy), que es la métrica que está configurada por defecto. Existe la posibilidad de crear una

función de pérdida que represente un escenario sensible al costo y que, por ende, el aprendizaje del modelo también se realice de esa manera. El aprendizaje sensible al costo es un tipo de aprendizaje que toma en cuenta los costos de clasificación errónea al entrenar un modelo de aprendizaje automático, por lo tanto, el objetivo de este tipo de aprendizaje es minimizar el costo total a través de dicha función. La principal diferencia con el aprendizaje tradicional o no sensible al costo es que trata las clasificaciones erróneas de manera diferente. Es decir, el costo de etiquetar un ejemplo positivo como negativo puede ser diferente del costo de etiquetar un ejemplo negativo como positivo. Sin embargo, los métodos de clasificación tradicionales no tienen en cuenta estos criterios y suponen un costo constante respecto de clasificación errónea. Los problemas de clasificación binaria desbalanceados, siendo el caso de uso de este proyecto, suelen presentar escenarios diferentes para cada uno de los errores de clasificación que se pueden cometer. La función de pérdida a desarrollar estaría basada en la métrica planteada en la sección 3.4.2.

Modelado

La forma en la que se ha modelado el problema de clasificación, donde cada registro del conjunto de datos representa un producto que fue enviado por el proceso logístico gestionado por la empresa, tiene embebido un sesgo hacia los productos que fueron posibles de etiquetar como maquinables, ya que hay muchas más observaciones de este tipo de productos. Se intentó mitigar esto con el uso combinado de técnicas de Data Augmentation y Data Reduction, explicadas en la sección 3.3.2, implementadas en la fase de entrenamiento. Sin embargo, el algoritmo podría tender a ponderar más a los patrones encontrados en los datos de los productos maquinables, porque al hacerlo, podrá clasificar adecuadamente una fracción mayor de los datos. Si los patrones provenientes de estos productos difieren significativamente de las características de los productos que son considerados no maquinables, nuestro modelo podría quedar sesgado. Por otro lado, debido a cómo se planteó la problemática y la oportunidad encontrada, la empresa definitivamente estaría interesada en el desempeño del clasificador para los productos no maquinables pero también estaría interesada en que no se degrade la experiencia de los productos maquinables.

Para abordar este potencial problema se pensó en algunas opciones. Esto consistiría en agrupar primero a los productos en grupos relevantes para el negocio, por ejemplo, a través del tipo de categoría. Luego, construir un clasificador diferente para cada grupo. Esto mismo se podría repetir con diferentes segmentaciones que se pueden realizar sobre los productos. Por otro lado, primero se podría equilibrar el conjunto de datos original sobre los grupos de productos para que los datos de entrenamiento tengan la misma fracción de observaciones de cada grupo de productos, posteriormente, ajustar un solo clasificador con esos datos. Ambos enfoques deberían probarse para decidir cuál podría ser más conveniente midiendo el rendimiento en diferentes conjuntos de validación y test que representen la distribución y el dominio de dichos grupos.

6.2. Próximos pasos

Si bien este proyecto podría resultar en una ventaja operativa para el proceso de gestión logística de la empresa, hay que tener en cuenta las limitaciones mencionadas en la sección 6.1. Cómo curso de acción, se proponen dos caminos paralelos. Por un lado, continuar una fase de exploración que permita encontrar una evolución del trabajo propuesto. Sobre todo,

haciendo foco en el aprendizaje sensible al costo dado que este haría que el algoritmo aprenda y se optimice de manera completamente distinta. Por otro lado, las pruebas de concepto deben ser testeadas en un ambiente productivo real para poder cuantificar el impacto genuino que la solución propuesta podría brindar. Para esto, se propone la ejecución de una metodología conocida como A/B testing, donde el objetivo de la prueba es dividir la población de usuarios de forma aleatoria, en este caso compradores, en dos flujos. Por un lado, habría una rama de control en la cual caerían todos los usuarios que podrían vivir la experiencia actual, donde la posibilidad de envío gestionado no queda restringida si el vendedor no especifica de antemano el tipo de producto. Por otro lado, habría una rama variante, en la cual se integra de forma productiva el modelo predictivo donde quedaría restringida la posibilidad de envío a través de la plataforma, en caso de que el algoritmo prediga que se trata de un producto no maquinable. Dado que este proceso podría afectar el proceso y la experiencia del usuario de manera negativa, se propone generar un división progresiva de las ramas, por ejemplo, primera iteración: 10% variante - 90% control. Como se puede ver en el ejemplo, las ramas no presentan la misma proporción. Para equilibrar esto, se suelen normalizar los resultados teniendo en cuenta la cantidad de usuarios / sesiones que pasaron por cada una de las ramas.
El objetivo del A/B testing es evaluar el rendimiento de la solución predictiva por medio de una métrica de negocio asociado al proceso productivo, no evaluar una métrica de desarrollo del modelo (sección 3.4.2). En este caso, la métrica de negocio podría ser el costo / ahorro que ambas ramas podrían generar en el escenario real.
En la rama de control, los productos se gestionan de manera tradicional, con los aciertos y errores que esto conlleve.
En la rama variante, si el producto es predicho como no maquinable cuando en realidad era maquinable, podría darse el caso que el comprador no quiera el producto y cancele la venta o no. En el caso inverso, se generaría el costo adicional por enviar un producto no maquinable.

De esta manera, los tomadores de decisión podrán tener un panorama claro, real y cuantificado para decidir sobre la inclusión definitiva de la solución propuesta.

7. Anexos

Anexo A: Datos

Conjuntos de Datos

Cantidad de Registros: 2.298.887

Rango de fechas: 01/03/2020 y el 30/04/2020

Columnas de Datos: 43 en total
- float64 - 13 variables
- int64 - 2 variables
- object - 28 variables

Descripción de Variables

El conjunto de datos presenta 43 variables que contienen las principales características de los ítems que son enviados a través del proceso logístico tercerizado. Estos son:

- Unnamed: índice del registro. Variable numérica discreta. Todos los valores son únicos.
- item_id: número que representa al ítem dentro del sistema informático. El número de ítem es generado por cada producto nuevo que se registra en el sitio. Es la identificación más granular para el universo de ítems en todo el sitio. Variable alfanumérica. Todos los valores son únicos. No presenta valores nulos.
- shp_service_id: es el número que identifica el proveedor del servicio de envío que fue utilizado dentro del proceso logístico. Variable numérica discreta. Todos los valores son únicos. Presenta valores repetidos, cantidad de valores únicos 36. No presenta valores nulos
- shp_quantity: es el número que indica la cantidad de unidades dentro del mismo envío. Debido al alcance del proyecto, este número es igual para todo registro, 1.
- shp_shipment_id: es el número que identifica el envío realizado a través de la plataforma. Variable numérica discreta. No presenta valores repetidos ni valores nulos.
- shp_weight: es el peso del ítem (en gramos) que fue mensurado por la empresa logística tercerizada. Variable numérica continua. No presenta valores nulos.
- company_weight: es la medida del peso del ítem (en gramos) que fue medido por la empresa e-commerce en caso de que el producto haya pasado en algún punto por el

centro de distribución de la propia empresa. También, puede darse el caso que estas mediciones sean producto de las predicciones de las medidas que fueron realizadas por la empresa. Variable numérica continua. No presenta valores nulos.

- shp_width: representa la medida del ancho del ítem (en centímetros) que fue mensurado por la empresa logística tercerizada. Variable numérica continua. No presenta valores nulos.

- company_width: es la medida del ancho del ítem (en centímetros) que fue medido por la empresa e-commerce en caso de que el producto haya pasado en algún punto por el centro de distribución de la propia empresa. También, puede darse el caso que estas mediciones sean producto de las predicciones de las medidas que fueron realizadas por la empresa. Variable numérica continua. No presenta valores nulos.

- shp_height: representa la medida de la altura del ítem (en centímetros) que fue mensurado por la empresa logística tercerizada. Variable numérica continua. No presenta valores nulos.

- company_height: es la medida del alto del ítem (en centímetros) que fue medido por la empresa e-commerce en caso de que el producto haya pasado en algún punto por el centro de distribución de la propia empresa. También, puede darse el caso que estas mediciones sean producto de las predicciones de las medidas que fueron realizadas por la empresa. Variable numérica continua. No presenta valores nulos.

- shp_length: representa la medida del largo del ítem (en centímetros) que fue mensurado por la empresa logística tercerizada. Variable numérica continua. No presenta valores nulos.

- company_length: es la medida del largo del ítem (en centímetros) que fue medido por la empresa e-commerce en caso de que el producto haya pasado en algún punto por el centro de distribución de la propia empresa. También, puede darse el caso que estas mediciones sean producto de las predicciones de las medidas que fueron realizadas por la empresa. Variable numérica continua. No presenta valores nulos.

- site_id: identificador del país. Dado que la problemática se desarrolla en Brasil, todos los registros contienen el mismo valor.

- domain_id: es el valor que identifica el dominio al cual pertenece el ítem. Este hace referencia e identificada a la categoría de más alto nivel con la que puede ser identificado un producto. Variable categórica, está conformada por una concatenación de strings. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- attributes: cada registro contiene diferentes atributos que describen al ítem. Objeto del tipo diccionario; cada key del diccionario corresponde a un tipo de descriptor y cada valor asociado al key, tiene el valor correspondiente. Podría utilizarse realizando un parseo del diccionario. No presenta valores nulos.

- shipping_mode: es el identificador del modo de envío que se utiliza. Debido al alcance de la problemática y dado que se especifica un solo tipo de logística, el valor es el mismo para todos los registros. Variable categórica. Conformada por un string.

- title: cada registro contiene el título del producto que registra el vendedor al momento de la publicación. Está conformada por cadenas de textos y/o números que describen las principales características del ítem y que el vendedor quiere destacar. El lenguaje utilizado para cada título es portugués. Presenta estructura heterogénea. Presenta valores repetidos. Presenta valores nulos.

- model_name: es uno de los descriptores del producto publicado. Está conformado por cadena de texto y/o números. Hace referencia al modelo en caso de que corresponda. Variable categórica. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- category_id: es el valor que identifica la categoría a la cual pertenece el ítem. Este descriptor corresponde a un nivel descriptivo más bajo respecto a la variable "domain_id". Variable categórica. Conformación alfanumérica. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- condition: valor que describe la condición del artículo publicado. Variable categórica conformada por texto. Presenta valores repetidos. No presenta valores nulos.

- catalog_product_id: es el identificador del producto a nivel de catálogo. Este código es utilizado para aquellos productos que podrían ser estandarizados dentro de un conjunto de publicaciones. Variable categórica, alfanumérica. Presenta valores repetidos como así también valores nulos. Tiene alta cardinalidad.

- shp_date_created: cada registro contiene la fecha de cuando fue iniciado el proceso de envío. Formato AAAA-MM-DD. No presenta valores nulos.

- logistic_type: identifica el tipo de logística que se utilizó para realizar el envío. Variable categórica conformada por texto. No presenta valores nulos.

- permalink: cada registro contiene la dirección url de la publicación del ítem. No presenta valores nulos.

- shp_volume: cada registro contiene el valor precalculado del volumen (en cm3) del paquete. Corresponde a la multiplicación de "shp_lenght", "shp_width" y "shp_height". Variable numérica continua. No presenta valores nules.

- seller_id: es el número que identifica al vendedor, responsable del envío. Variable numérica entera. Presenta valores repetidos. No presenta valores nulos. Presenta alta cardinalidad.
- brand_name: representa la marca del producto enviado. Variable categórica conformada por texto y números según sea el caso. Presenta valores repetidos, valores nulos y alta cardinalidad.

- shp_dimensions_from: referencia por la cual se identifica donde fueron tomadas las últimas medidas de los paquetes dentro del proceso logístico. Variable categórica conformada por texto. Presenta valores repetidos. No presenta valores nulos.

- shp_picking_type_id: identifica el tipo de picking que se utilizó para realizar el envío. Variable categórica conformada por texto. No presenta valores nulos.

- price: cada registro contiene el precio del ítem en valores de la moneda de Brasil, el real. Variable numérica. No presenta valores nulos.

- parent_product_id: es el número que identifica el producto padre del ítem en cuestión. En caso de que el producto pueda ser estandarizado, habrá distintos vendedores que vendan el mismo artículo pero con diferente "item_id". Este valor agrupa todos esos ítems bajo una misma identificación. Variable categórica, alfanumérica. Presenta valores nulos.

- creation_date: cada registro contiene la fecha correspondiente a la creación de la publicación del producto. Formato AAAA-MM-DDTHH:MM:SS. No presenta valores nulos.

- path_from_root: representa parte del proceso de búsqueda que realizó el usuario para adquirir dicho artículo. Está conformado por un diccionario que contiene la secuencia de categorías, a diferentes niveles, que visitó. No presenta valores nulos.

- status: hace referencia al estado de la publicación del producto. Variable categórica conformada por texto. No presenta valores nulos.

- L1: identificador de la categoría nivel 1, a la cual pertenece el producto enviado. Variable categórica, conformada por texto. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- L2: identificador de la categoría nivel 2, a la cual pertenece el producto enviado. Variable categórica, conformada por texto. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- L3: identificador de la categoría nivel 3, a la cual pertenece el producto enviado. Variable categórica, conformada por texto. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- L4: identificador de la categoría nivel 4, a la cual pertenece el producto enviado. Variable categórica, conformada por texto. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- L5: identificador de la categoría nivel 5, a la cual pertenece el producto enviado. Variable categórica, conformada por texto. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- L6: identificador de la categoría nivel 6, a la cual pertenece el producto enviado. Variable categórica, conformada por texto. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- L7: identificador de la categoría nivel 7, a la cual pertenece el producto enviado. Variable categórica, conformada por texto. Presenta valores repetidos. Presenta valores nulos. Presenta alta cardinalidad.

- main_picture: cada registro contiene la dirección url donde se ubica la imagen principal del producto. Presenta valores nulos.

Características de las Variables

Id	Column	Dtype	Nulos
0	Unnamed: 0	int64	0
1	item_id	object	0
2	shp_service_id	float64	0
3	meli_length	float64	0
4	shp_quantity	float64	0
5	shp_weight	float64	0
6	shp_shipment_id	float64	0
7	meli_weight	float64	0
8	shp_width	float64	0
9	meli_height	float64	0
10	site_id	object	0
11	domain_id	object	5986
12	attributes	object	0
13	shipping_mode	object	0
14	title	object	11
15	shp_length	float64	0
16	model_name	object	718608
17	category_id	object	11
18	condition	object	0
19	catalog_product_id	object	1872372
20	shp_date_created	object	0
21	logistic_type	object	0
22	permalink	object	0
23	shp_volume	float64	0
24	seller_id	int64	0
25	brand_name	object	175099
26	shp_dimensions_from	object	0
27	shp_picking_type_id	object	0
28	price	float64	0
29	parent_product_id	object	1998017
30	creation_date	object	0
31	shp_height	float64	0

32	path_from_root	object	0
33	status	object	0
34	meli_width	float64	0
35	L1	object	3
36	L2	object	3
37	L3	object	230515
38	L4	object	937063
39	L5	object	1761537
40	L6	object	2022844
41	L7	object	2043217
42	main_picture	object	42575

Características de las Variables Categóricas

Variable	Conteo	Únicos	Top	Frecuencia	Nulos	% Nulos
item_id	2298887	2298887	MLB1379803419	1	0	0.0%
site_id	2298887	1	MLB	2298887	0	0.0%
seller_id	2298887	174508	53919165	3590	0	0.0%
domain_id	2292166	3903	MLB-VEHICLE_ACCESSORIES	61582	5986	0.3%
shp_shipment_id	2298887	2298887	27762098175	1	0	0.0%
shp_service_id	2298887	36	21	1369719	0	0.0%
attributes	2298887	1961444	[{'attribute_group_id': 'OTHERS', 'attribute_group_name': 'Outros', 'id': 'ITEM_CONDITION', 'name': 'Condição do item', 'value_id': '2230284', 'value_name': 'Novo', 'value_struct': None}]	40137	0	0.0%
shipping_mode	2298887	1	me2	2298887	0	0.0%
title	2298876	2170229	Xiaomi Redmi Note 8 Dual Sim 64 Gb Preto-espacial 4 Gb Ram	100	11	0.0%
model_name	1490935	590005	Universal	7737	718608	31.3%
category_id	2298876	5367	MLB3936	33871	11	0.0%
condition	2298887	3	new	2272318	0	0.0%
catalog_product_id	192550	55649	MLB12287867	492	1872372	81.4%
shp_date_created	2298887	784	2020-01-21	8225	0	0.0%
logistic_type	2298887	5	drop_off	1683766	0	0.0%
permalink	2298887	2298887	https://produto.mercadolivre.com.br/MLB-1187646828-100-uni-chaveiro-bola-de-futebol-anti-estress	1	0	0.0%

			e-__JM			
brand_name	2101962	207024	Importado	46128	175099	7.6%
shp_dimensions_from	2298887	3	carrier	2029100	0	0.0%
shp_picking_type_id	2298887	4	drop_off	1961020	0	0.0%
parent_product_id	51039	13418	MLB15188551	544	1998017	86.9%
creation_date	2298887	2133736	2018-06-28T21:32:09Z	34	0	0.0%
path_from_root	2298887	5573	[{'name': 'Acessórios para Veículos'}, {'name': 'Aces. de Motos e Quadriciclos'}, {'name': 'Outros'}]	34806	0	0.0%
status	2298887	2	active	1400328	0	0.0%
L1	2298884	29	Acessórios para Veículos	582114	3	0.0%
L2	2298884	303	Peças de Carros e Caminhonetes	321325	3	0.0%
L3	2039328	1551	Outros	120811	230515	10.0%
L4	1244631	2345	Outros	90934	937063	40.8%
L5	316903	768	Outros	21697	1761537	76.6%
L6	22979	110	Outros	2986	2022844	88.0%
L7	53	6	Rolos Isolantes	25	2043217	88.9%
main_picture	2250842	2189904	http://http2.mlstatic.com/resources/frontend/statics/processing-image/1.0.0/O-PT.jpg	760	42575	1.9%

Características de las Variables Numéricas

Variable	Conteo	Promedio	Des. Std.	Min	25%	50%	75%	Max
meli_length	2298887	30.78	13.60	1	23	26	35	150
meli_height	2298887	11.47	6.65	1	6	10	15	71
meli_width	2298887	21.33	8.86	1	15	20	25	106
meli_weight	2298887	1479.42	2289.70	1	279	664	1660	30000
shp_length	2298887	32.72	18.25	1	20	28	40	2069
shp_height	2298887	12.18	8.04	1	6	10	16	160
shp_width	2298887	22.07	10.69	1	14	19.5	27	171
shp_weight	2298887	1787.21	3052.77	0.1	260	660	1880	35000
shp_volume	2298887	14318.78	28212.35	1	2128	5580	15360	4096000
shp_quantity	2298887	1	0	1	1	1	1	1
price	2298887	6297.70	2532208.43	0.1	55.34	119.9	219.94	2861097399

Anexo B: Calidad de los Datos

Plan de acción

Problema Calidad	Descripción	Variable	Acción
Sin Información	No contiene información valiosa	Unnamed	Eliminar variable
		shp_service_id	
		shp_quantity	
		shp_shipment_id	
		site_id	
		shipping_mode	
		condition	
		catalog_product_id	
Sin Información	No contiene información valiosa	logistic_type	Eliminar variable
		permalink	
		shp_dimensions_from	
		shp_picking_type_id	
		parent_product_id	
		creation_date	
		status	
		main_picture	
Discrepancia	Muchos registros contienen información dudosa	company_length	Eliminar variable
		company_weight	
		company_height	
		company_width	
Cardinalidad	La variable presenta alta cardinalidad	item_id	No utilizar variable
		attributes	Eliminar variable
Duplicados	El conjunto representa envíos, se debe transformar para que represente un conjunto de productos	Title	Eliminar los títulos duplicados
Valores nulos	Valores nulos por falla de recolección		Eliminar los registros con valores nulos
		domain_id	Eliminar los registros con valores nulos

	Descriptor con valores nulos debido a su naturaleza. Muchos artículos no tienen modelo	model_name	Encontrar método para manejar valores nulos
	Valores nulos por falla de recolección	category_id	Eliminar los registros con valores nulos
	Descriptor con valores nulos debido a su naturaleza. Muchos artículos no tienen marca	brand_name	Encontrar método para manejar valores nulos
Valores nulos	Valores nulos por falla de recolección	L1	Encontrar método para manejar valores nulos
		L2	Encontrar método para manejar valores nulos
		L3	Encontrar método para manejar valores nulos
	Descriptor con gran cantidad de valores nulos debido a su naturaleza. Muchos artículos no tienen descripción de categoría	L4	Eliminar variable
		L5	
		L6	
		L7	
Información Duplicada	Contienen información duplicada	path_from_root	Eliminar variable

Anexo C: Hiperparámetros

Configuración del espacio de hiperparámetros

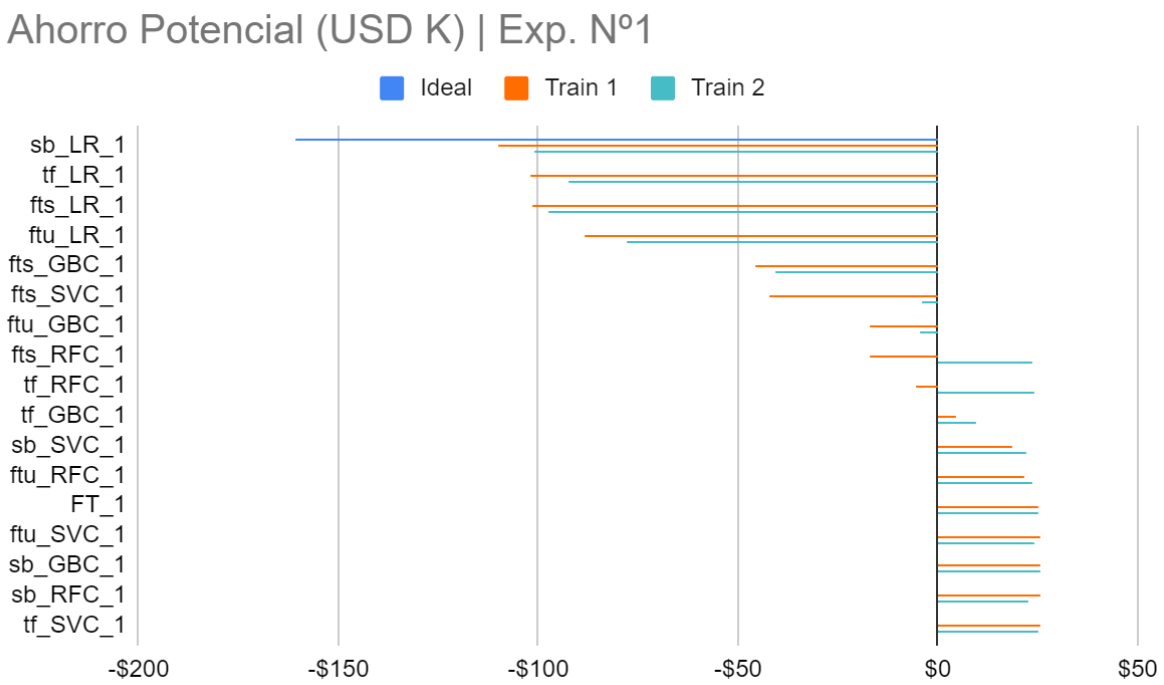
Algoritmo / Paso	Hiperparámetros	Espacio de valores
Pipeline - PCA	nº de componentes principales	{5, 10, 15, 20}
Regresión Logística	C (parámetro regularizador)	{(0,1) - Uniform - 6 valores, (0,1000) - Unidorm - 4 valores}
	penalty	{“l1”, “l2”}
	solver	“saga”
	max_iter	200
	class weight	“balanced”
MVS	C (parámetro regularizador)	{(0,1) - Uniform - 6 valores, (0,1000) - Unidorm - 4 valores}
	penalty	{“rbf”, “sigmoi”, “poly”, “linear”}
	gamma	“auto”
	probability	True
	max_iter	200
	class weight	“balanced”
Random Forest	n_estimators	{200, 300, 400, 500}
	criterion	{“entropy”, “gini”}
	min_samples_split	{0.01, 0.025, 0.05}
	min_samples_leaf	{0.1, 0.2, 0.3}
	class weight	{“balanced”, “balanced_subsample”}
Gradient Boosting	loss	{“deviance”, “exponential”}

Gradient Boosting	learning_rate	{0.001, 0.005, 0.01, 0.03, 0.05}
	subsample	{0.2, 0.4, 0.6, 0.8, 1}
	n_estimators	{200, 300, 400, 500}
	max_depth	{3, 6, 9, 12}
	min_samples_split	{0.01, 0.025, 0.05}
	min_samples_leaf	{0.1, 0.2, 0.3}

Anexo D: Resultados

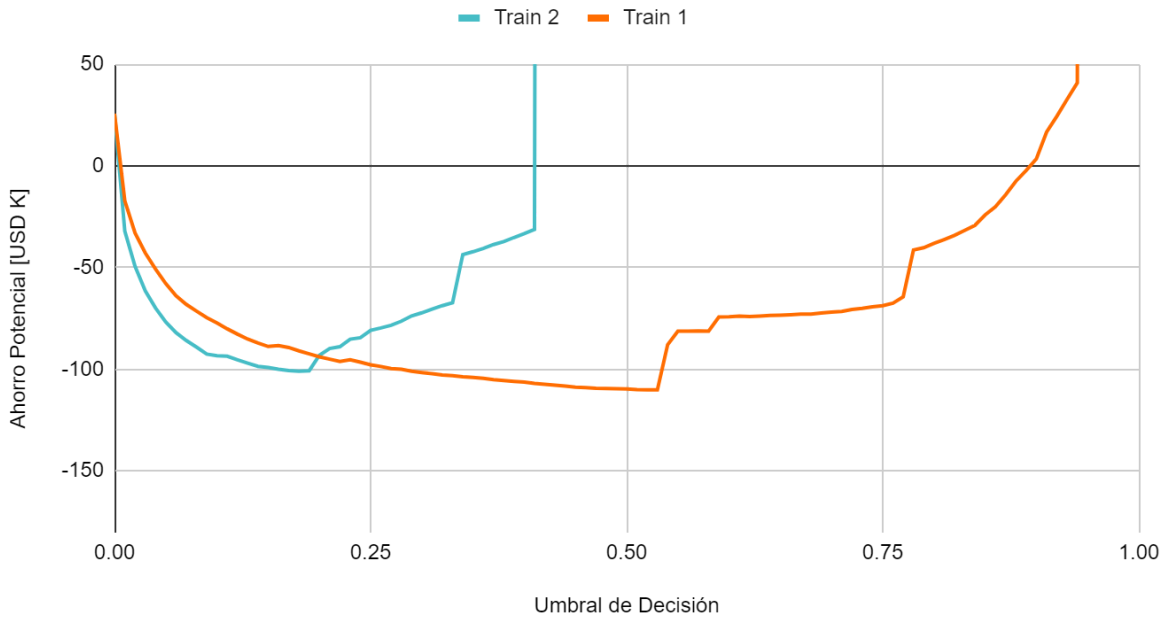
Resultados de los mejores modelos para cada experimento

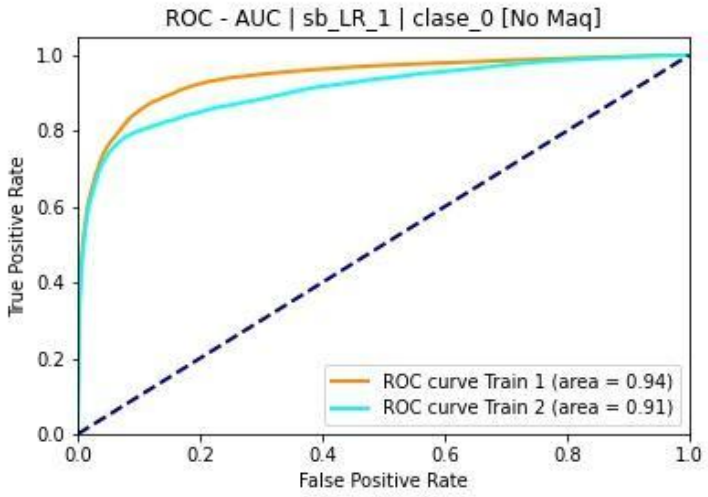
- **Experimento 1**: el descriptor está conformado por el título.

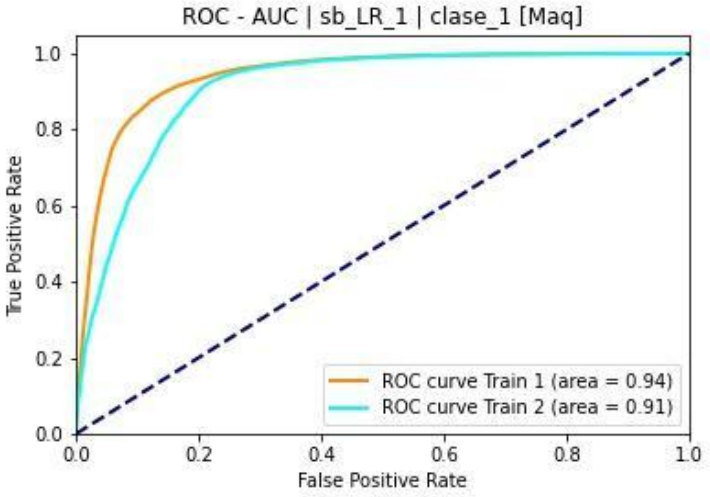


Principales Características	Entrenamiento	
	Train 1	Train 2
Transformador	BERT	
Clasificador	Regresión Logística	
Mejor Umbral	0.53	0.18
% de Ahorro	110.176 (68%)	101.191 (62%)
Precisión Maquinables	99%	99%
Recall Maquinables	89%	92%
Precisión No Maquinables	21%	25%
Recall No Maquinables	86%	79%

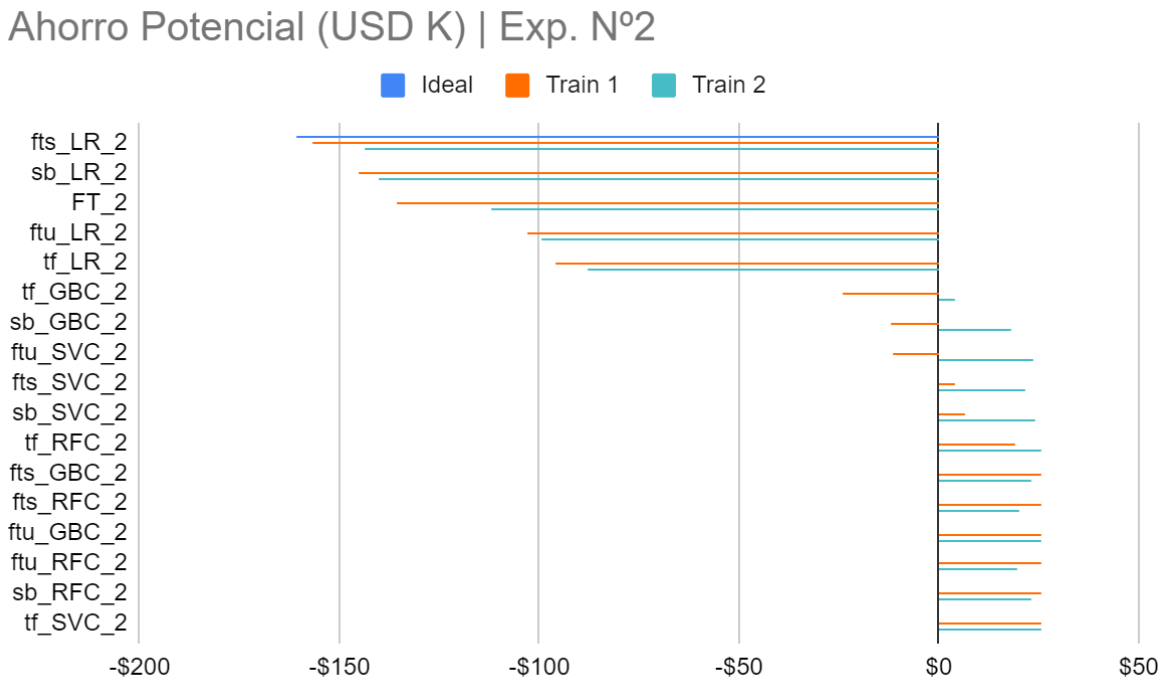
Función de Ahorro | Exp. nº 1 | Iter. sb_LR_1



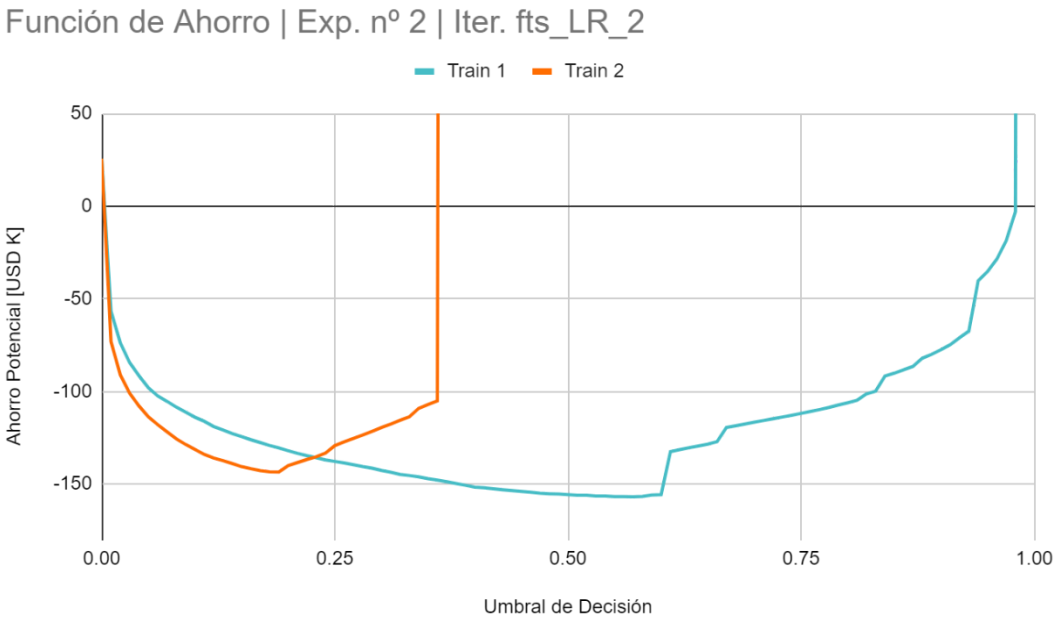


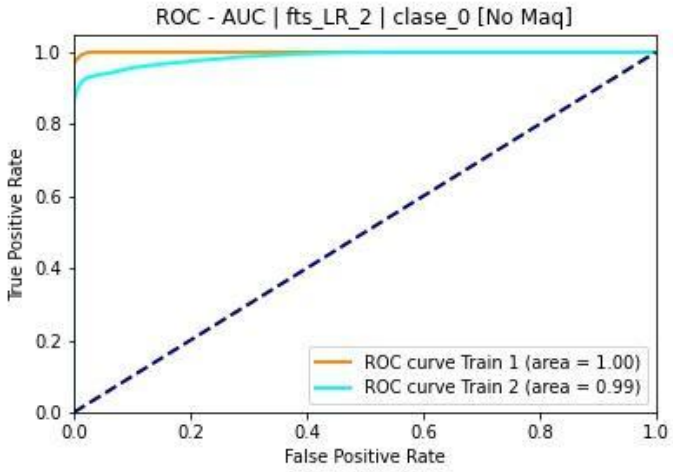


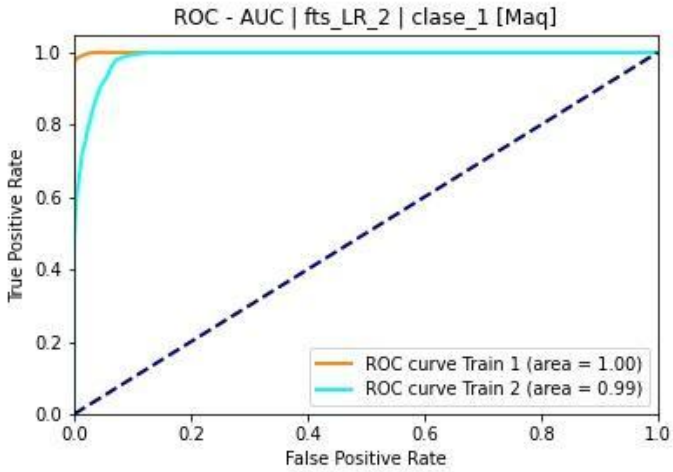
- **Experimento 2**: el descriptor está conformado por el descriptor del experimento 1 + la variable domain_id del producto.



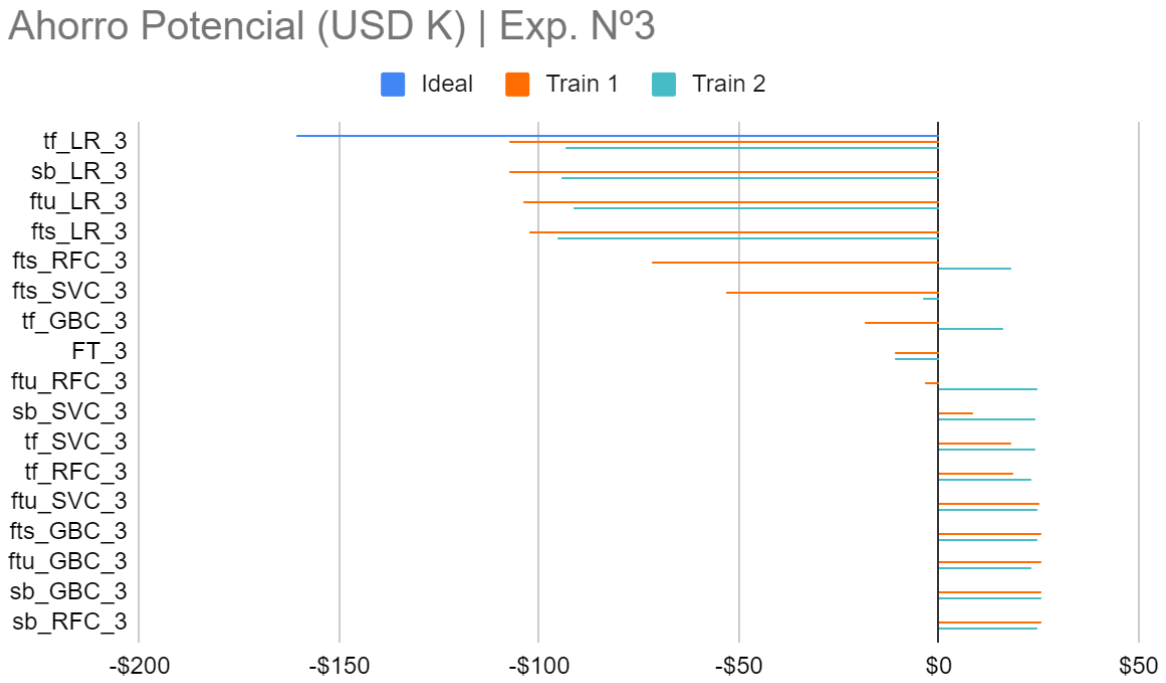
Principales Características	Entrenamiento	
	Train 1	**Train 2**
Transformador	Fast Text Supervisado	
Clasificador	Regresión Logística	
Mejor Umbral	0.57	0.19
% de Ahorro	156.929 (97%)	143.524 (89%)
Precisión Maquinables	98%	99%
Recall Maquinables	99%	98%
Precisión No Maquinables	70%	57%
Recall No Maquinables	99%	93%


Función de Ahorro | Exp. nº 2 | Iter. fts_LR_2
Train 1
Train 2
Ahorro Potencial [USD K]
50
0
-50
-100
-150
0.00
0.25
0.50
0.75
1.00
Umbral de Decisión


ROC - AUC | fts_LR_2 | clase_0 [No Maq]
True Positive Rate
False Positive Rate
ROC curve Train 1 (area = 1.00)
ROC curve Train 2 (area = 0.99)

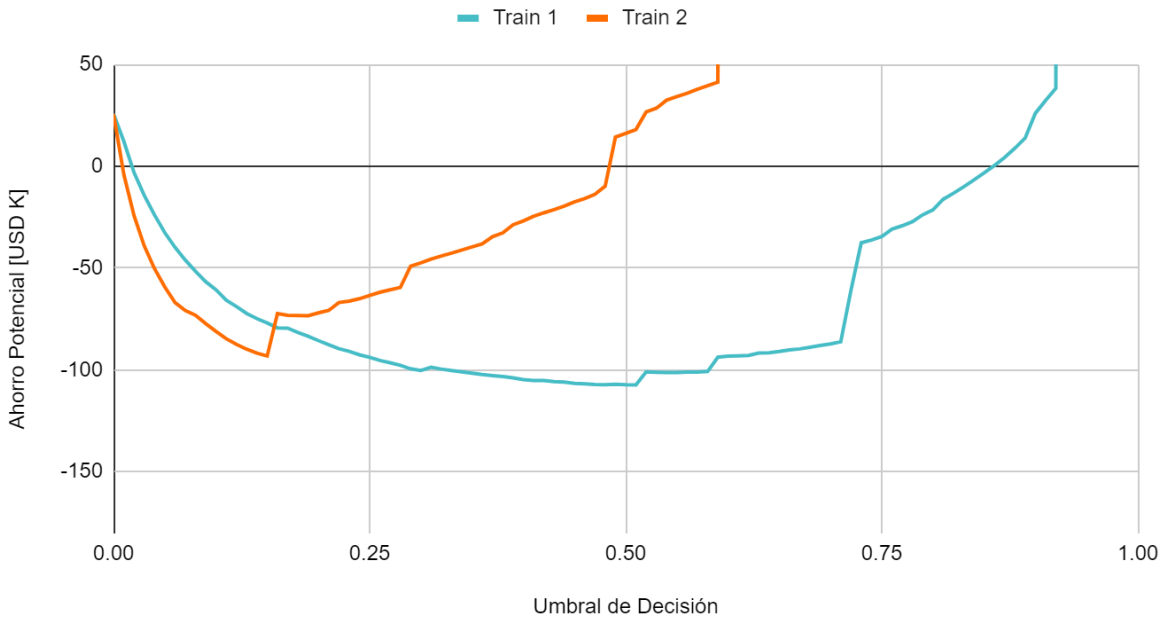

ROC - AUC | fts_LR_2 | clase_1 [Maq]
True Positive Rate
False Positive Rate
ROC curve Train 1 (area = 1.00)
ROC curve Train 2 (area = 0.99)

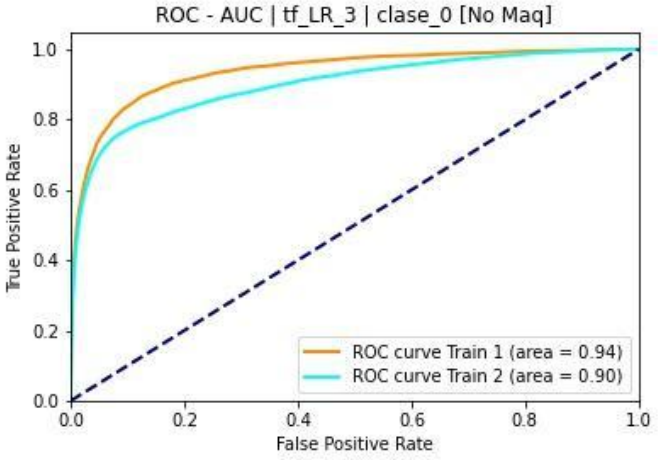
- **Experimento 3**: el descriptor está conformado por el descriptor del experimento 2 + la variable category_id del producto.

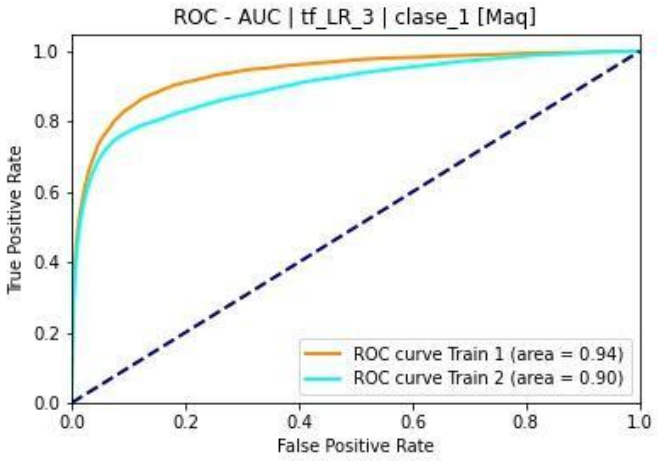


Principales Características	Entrenamiento	
	Train 1	Train 2
Transformador	Tensor Flow	
Clasificador	Regresión Logística	
Mejor Umbral	0.51	0.15
% de Ahorro	107.506 (67%)	93.246 (58%)
Precisión Maquinables	99%	99%
Recall Maquinables	89%	94%
Precisión No Maquinables	20%	28%
Recall No Maquinables	85%	72%

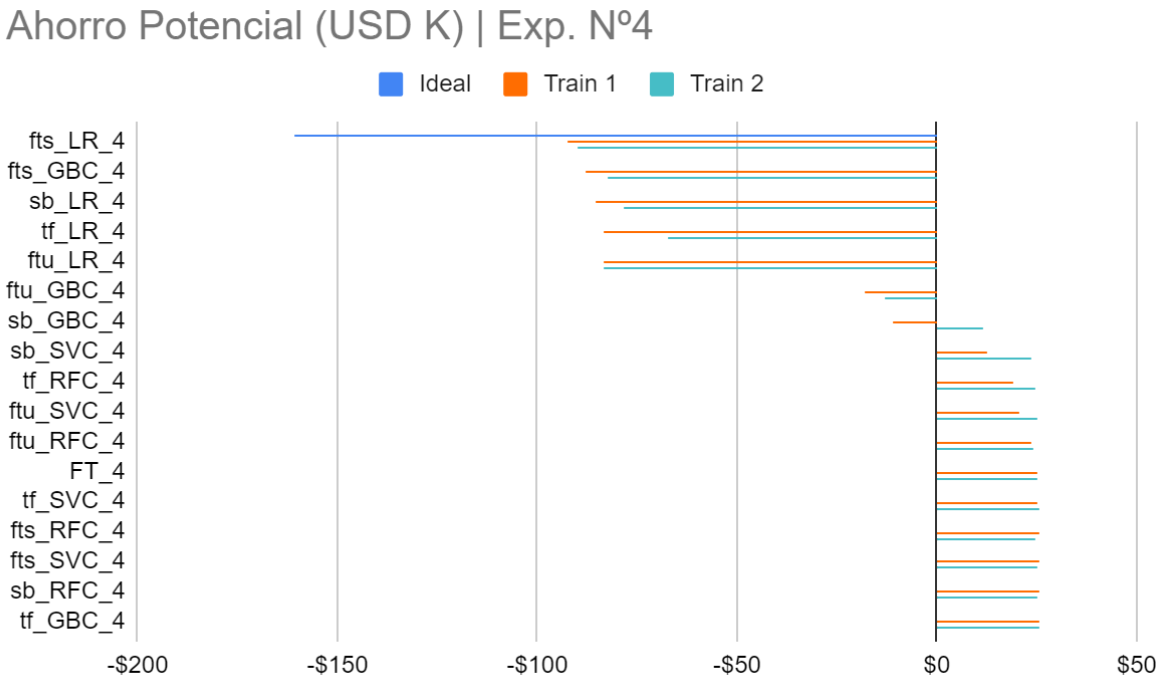
Función de Ahorro | Exp. nº 3 | Iter. tf_LR_3





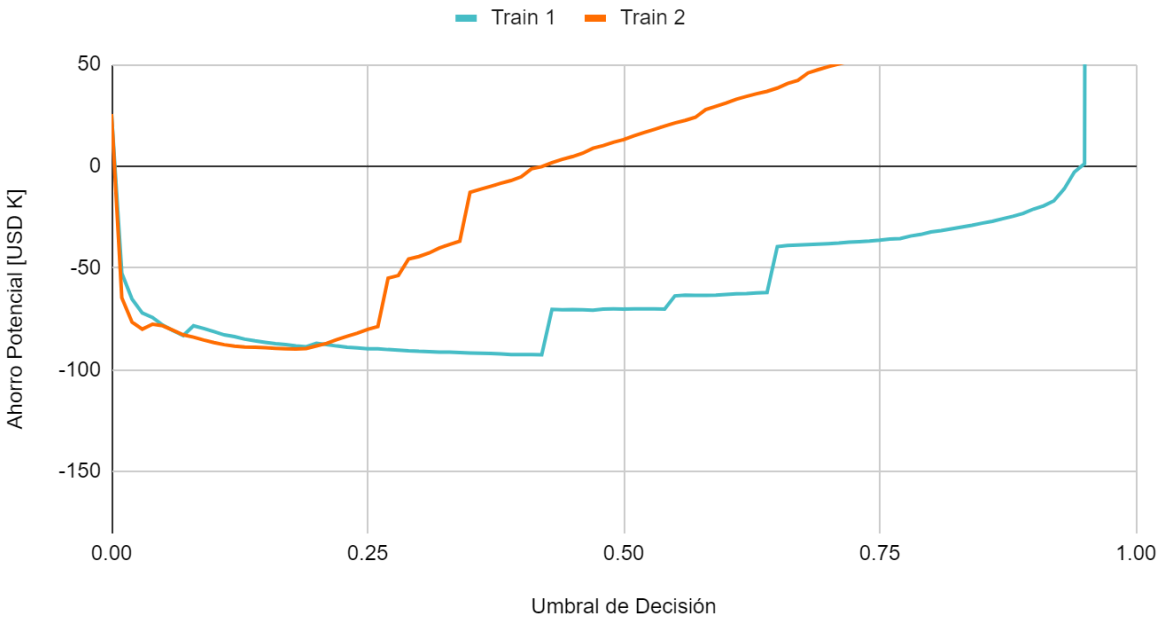


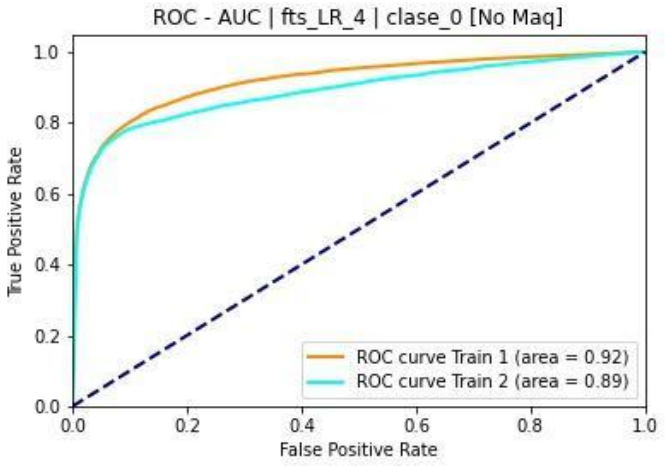
- **Experimento 4**: el descriptor está conformado por el descriptor del experimento 3 + la variable seller_id del producto.

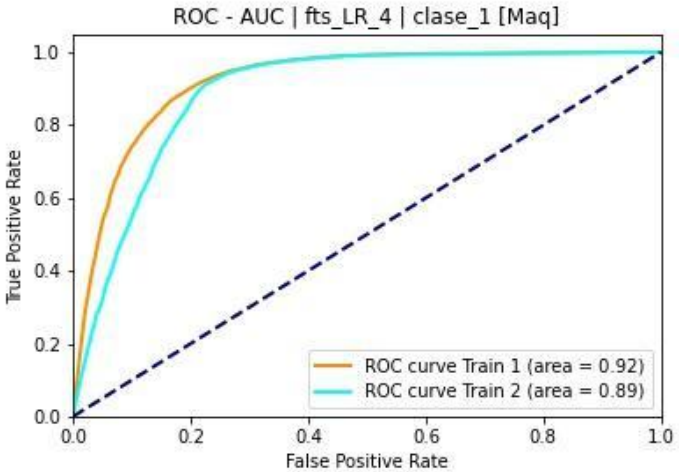


Principales Características	**Entrenamiento**	
	Train 1	**Train 2**
Transformador	Fast Text Supervisado	
Clasificador	Regresión Logística	
Mejor Umbral	0.42	0.18
% de Ahorro	92.634 (58%)	89.828 (56%)
Precisión Maquinables	99%	99%
Recall Maquinables	91%	91%
Precisión No Maquinables	23%	22%
Recall No Maquinables	79%	78%

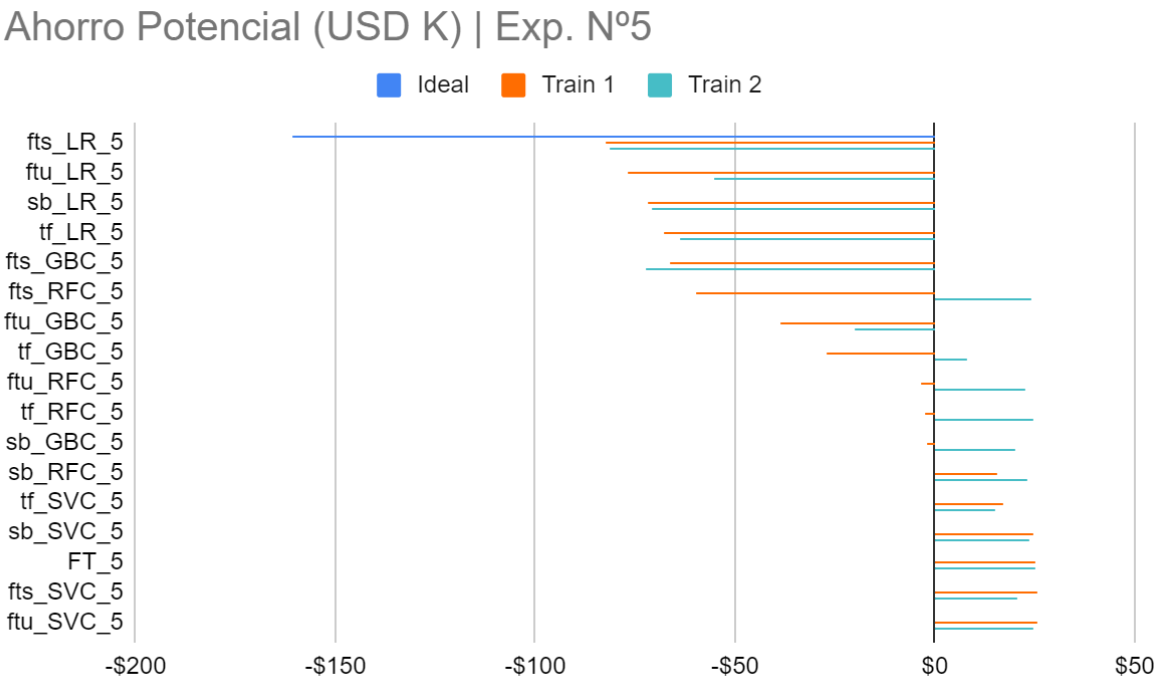
Función de Ahorro | Exp. nº 4 | Iter. fts_LR_4



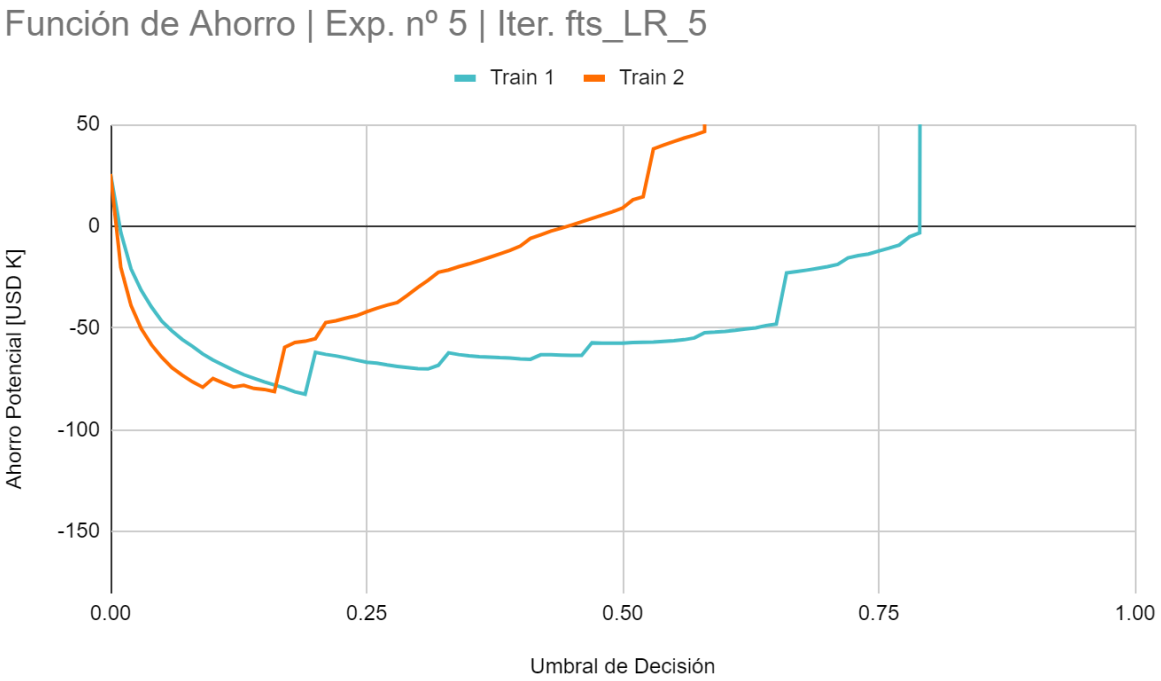


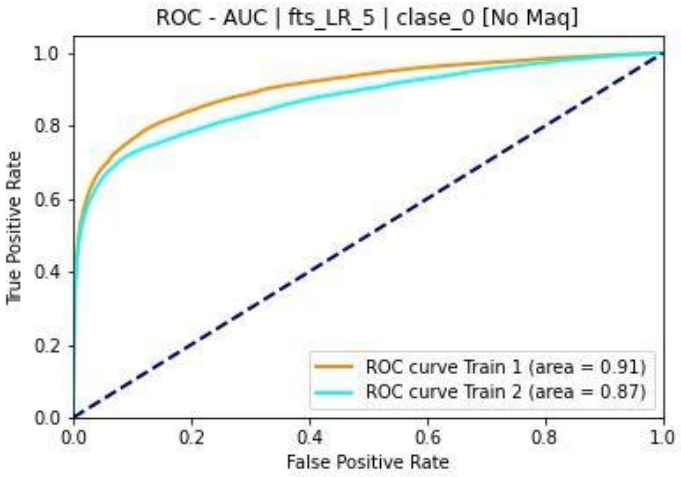


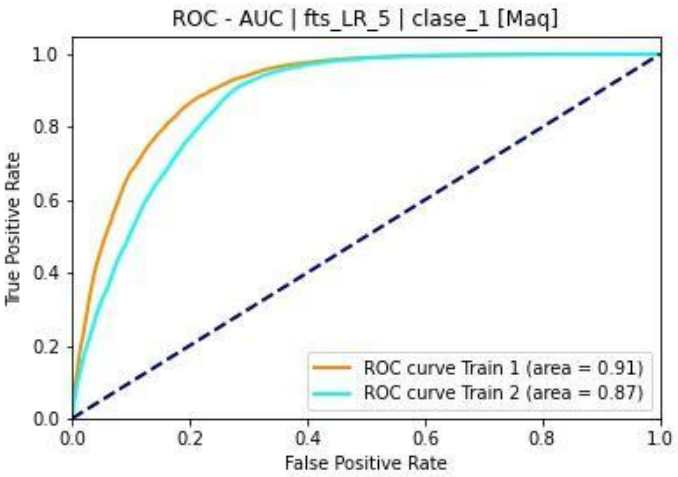
- **Experimento 5**: el descriptor está conformado por el descriptor del experimento 4 + la variable brand_name del producto.



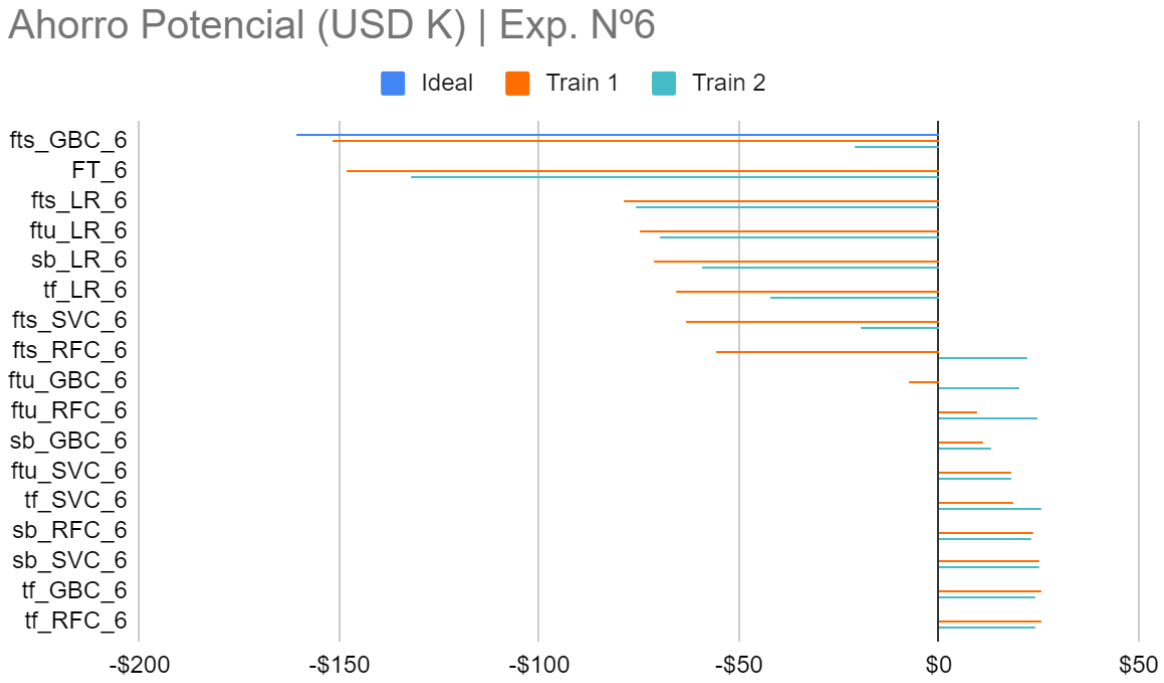
Principales Características	Entrenamiento	
	Train 1	**Train 2**
Transformador	Fast Text Supervisado	
Clasificador	Regresión Logística	
Mejor Umbral	0.19	0.16
% de Ahorro	82.555 (51%)	81.256 (50%)
Precisión Maquinables	99%	99%
Recall Maquinables	98%	92%
Precisión No Maquinables	45%	23%
Recall No Maquinables	61%	70%


Función de Ahorro | Exp. nº 5 | Iter. fts_LR_5
Train 1
Train 2
Ahorro Potencial [USD K]
50
0
-50
-100
-150
0.00
0.25
0.50
0.75
1.00
Umbral de Decisión

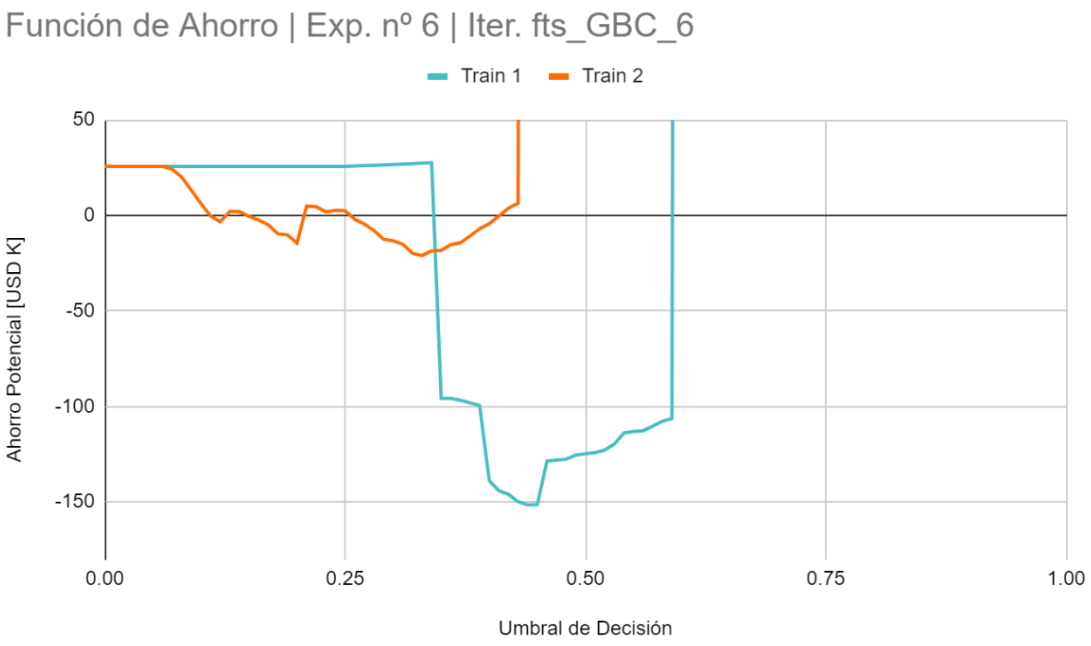

ROC - AUC | fts_LR_5 | clase_0 [No Maq]
True Positive Rate
False Positive Rate
ROC curve Train 1 (area = 0.91)
ROC curve Train 2 (area = 0.87)

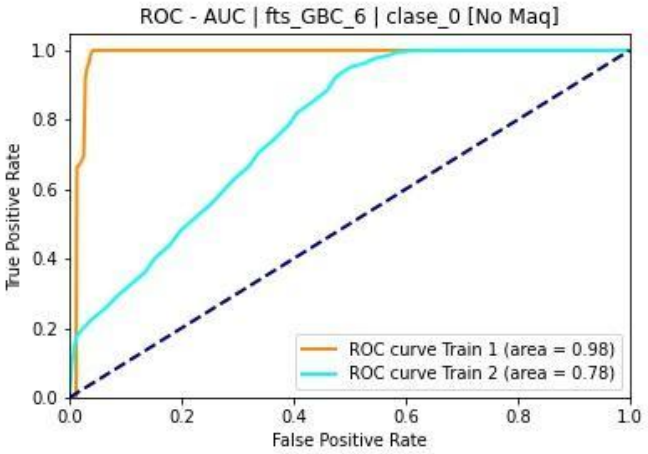

ROC - AUC | fts_LR_5 | clase_1 [Maq]
True Positive Rate
False Positive Rate
ROC curve Train 1 (area = 0.91)
ROC curve Train 2 (area = 0.87)

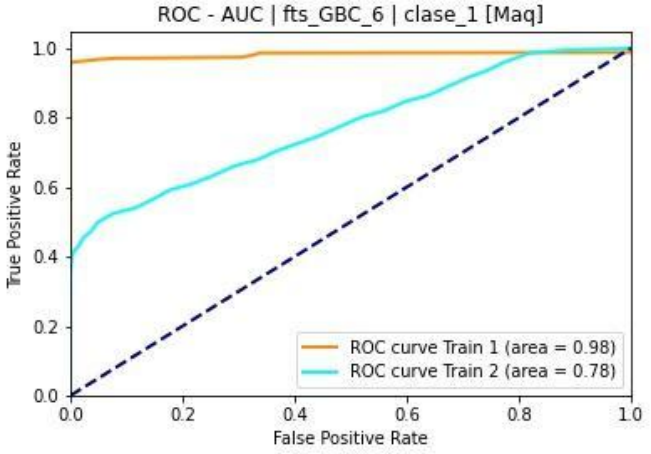
- **Experimento 6**: el descriptor está conformado por el descriptor del experimento 5 + la variable L1 del producto.



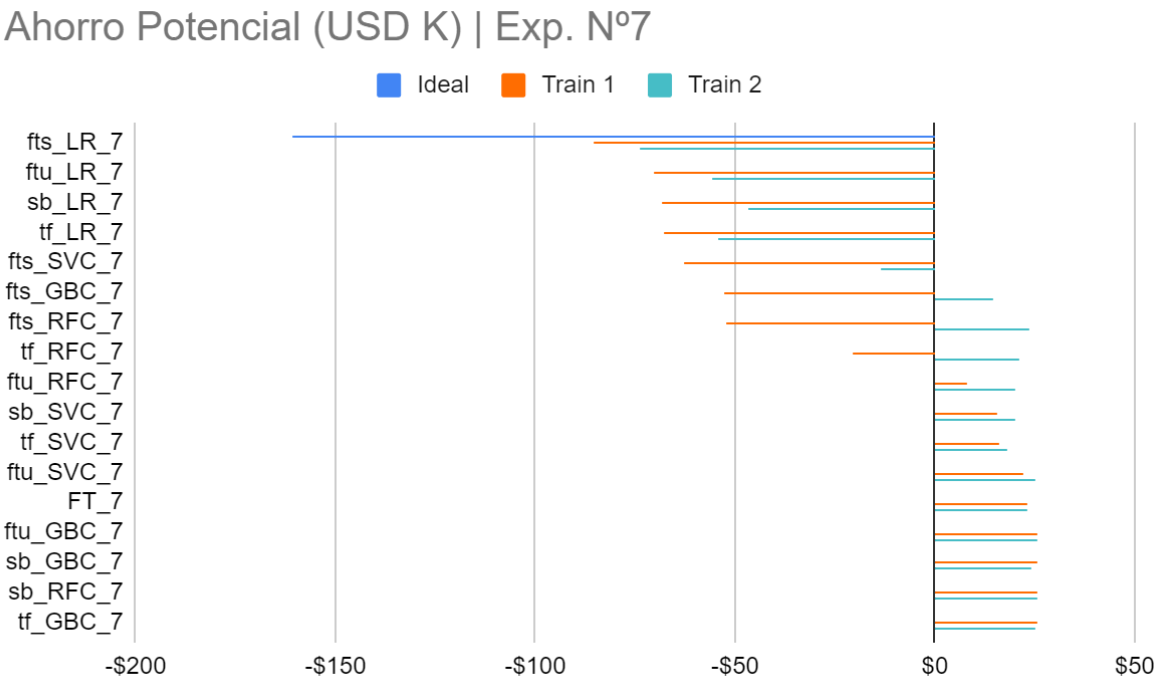
Principales Características	**Entrenamiento**	
	Train 1	**Train 2**
Transformador	Fast Text Supervisado	
Clasificador	Gradient Boosting	
Mejor Umbral	0.44	0.33
% de Ahorro	151.608 (94%)	21.036 (13%)
Precisión Maquinables	99%	99%
Recall Maquinables	96%	50%
Precisión No Maquinables	45%	6%
Recall No Maquinables	99%	95%


Función de Ahorro | Exp. nº 6 | Iter. fts_GBC_6
Train 1
Train 2
Ahorro Potencial [USD K]
50
0
-50
-100
-150
0.00
0.25
0.50
0.75
1.00
Umbral de Decisión


ROC - AUC | fts_GBC_6 | clase_0 [No Maq]
True Positive Rate
False Positive Rate
ROC curve Train 1 (area = 0.98)
ROC curve Train 2 (area = 0.78)

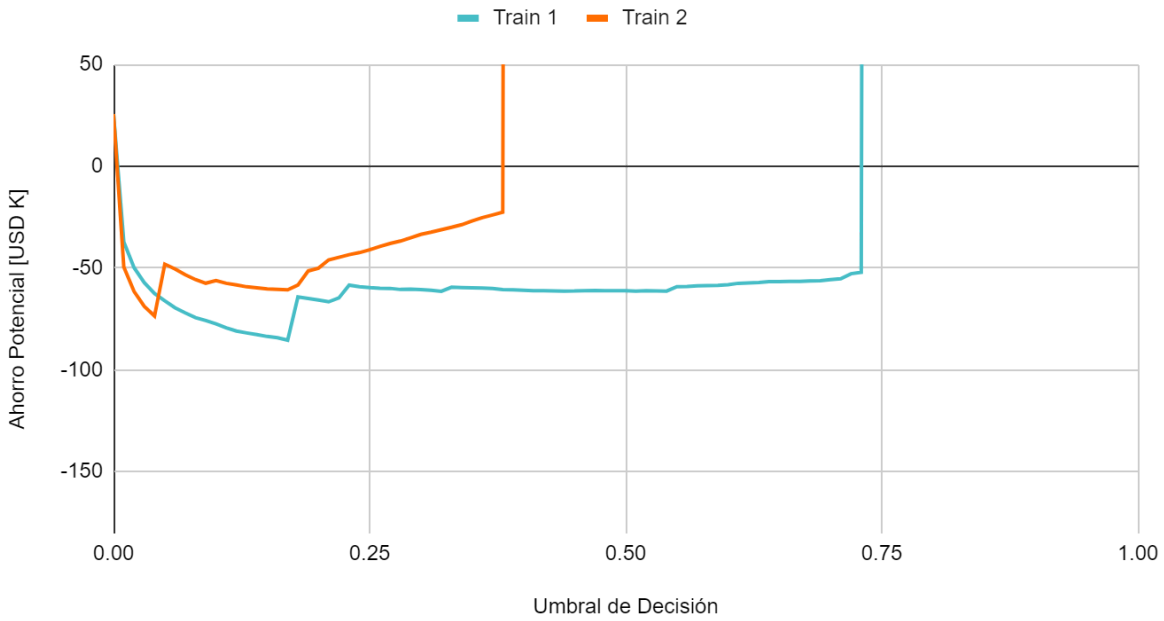

ROC - AUC | fts_GBC_6 | clase_1 [Maq]
True Positive Rate
False Positive Rate
ROC curve Train 1 (area = 0.98)
ROC curve Train 2 (area = 0.78)

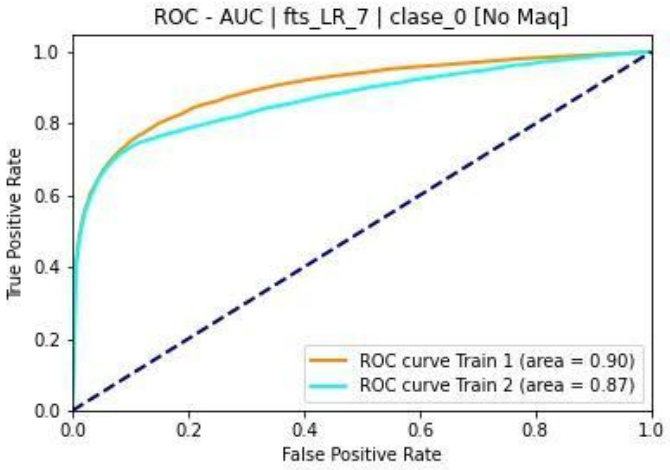
- **Experimento 7**: el descriptor está conformado por el descriptor del experimento 6 + la variable L2 del producto.

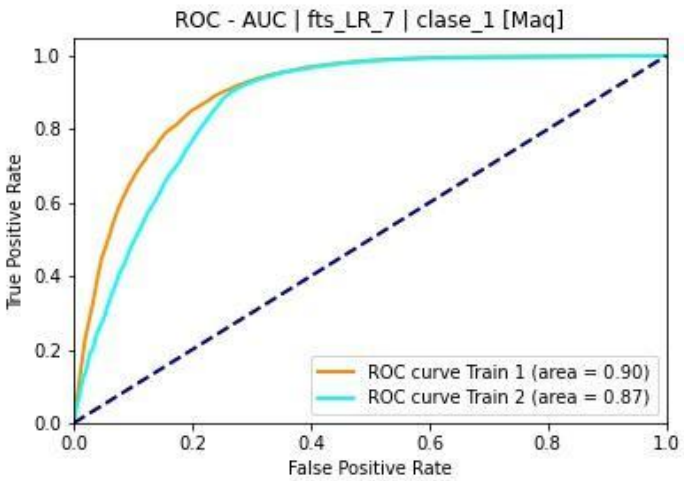


Principales Características	Entrenamiento	
	Train 1	**Train 2**
Transformador	Fast Text Supervisado	
Clasificador	Regresión Logística	
Mejor Umbral	0.17	0.04
% de Ahorro	85.466 (53%)	73.631 (46%)
Precisión Maquinables	99%	99%
Recall Maquinables	96%	98%
Precisión No Maquinables	33%	45%
Recall No Maquinables	64%	56%

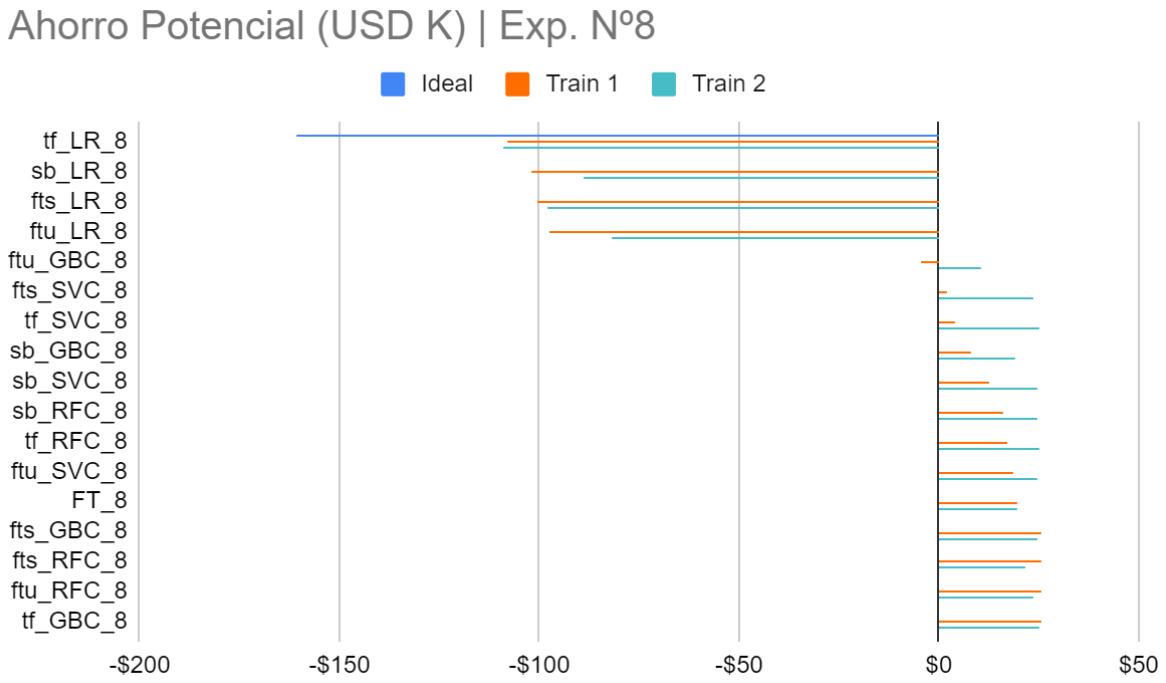
Función de Ahorro | Exp. nº 7 | Iter. fts_LR_7



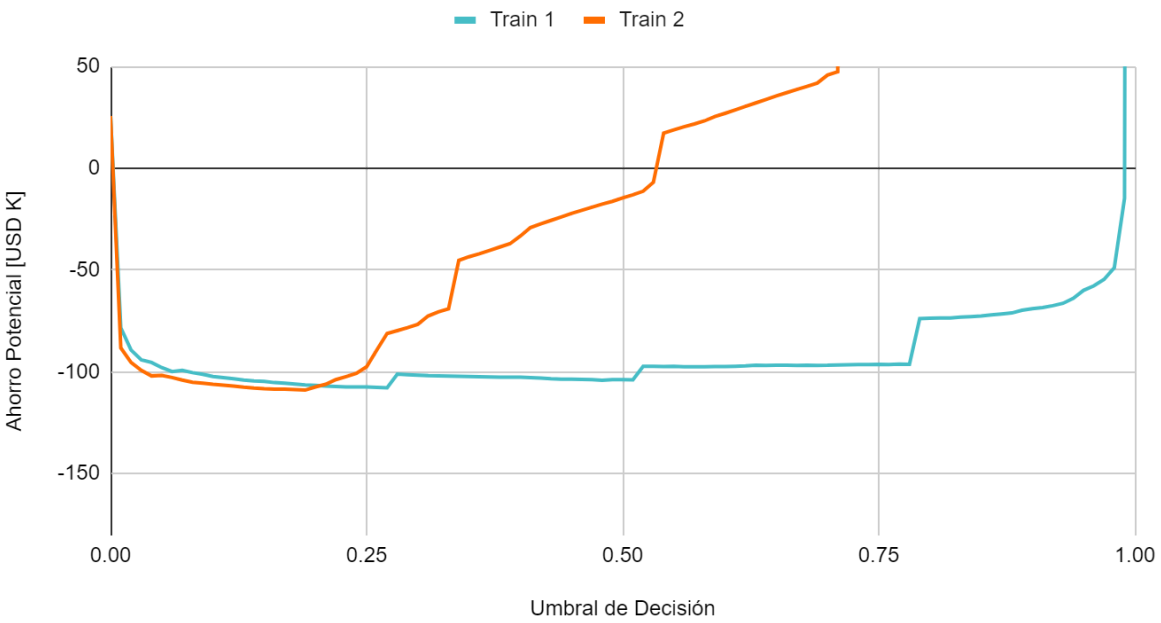


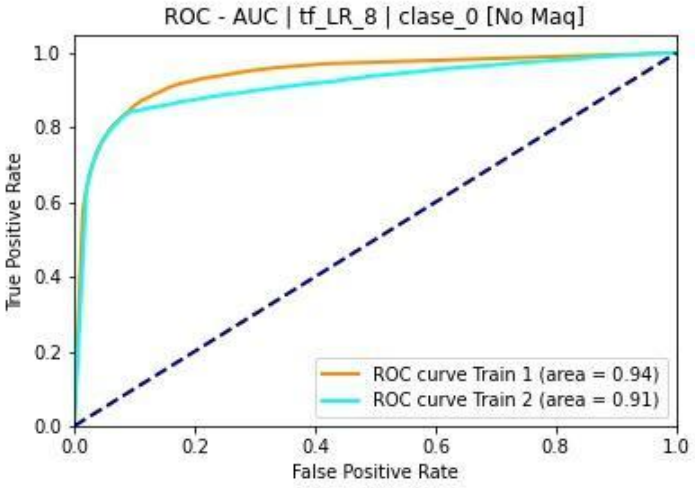


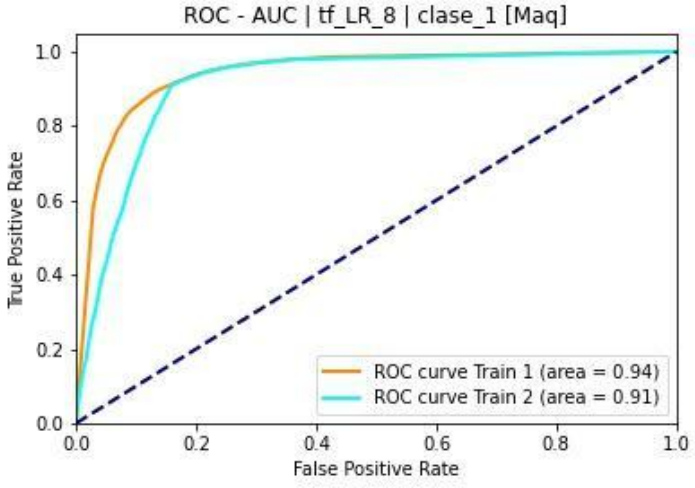
- **Experimento 8**: el descriptor está conformado por el título del producto + la variable category_id del producto.



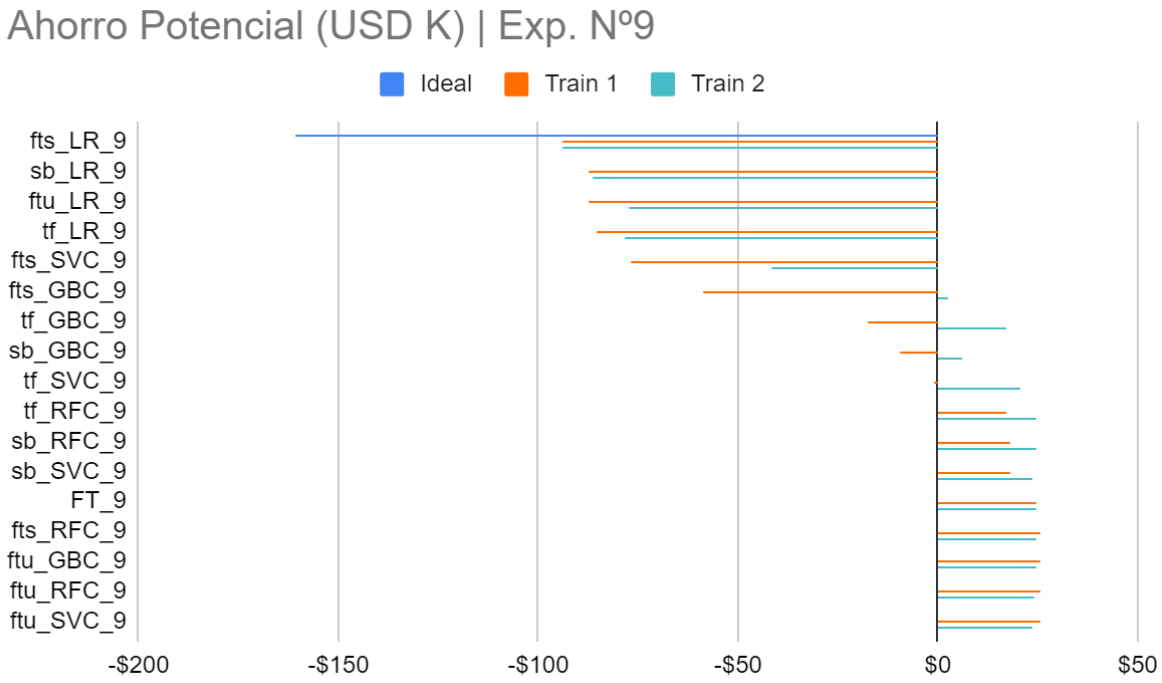
Principales Características	Entrenamiento	
	Train 1	**Train 2**
Transformador	Tensor Flow	
Clasificador	Regresión Logística	
Mejor Umbral	0.27	0.19
% de Ahorro	107.893 (67%)	109.017 (68%)
Precisión Maquinables	99%	99%
Recall Maquinables	93%	91%
Precisión No Maquinables	27%	23%
Recall No Maquinables	82%	84%


Función de Ahorro | Exp. nº 8 | Iter. fts_LR_8
Train 1
Train 2
Ahorro Potencial [USD K]
50
0
-50
-100
-150
0.00
0.25
0.50
0.75
1.00
Umbral de Decisión


ROC - AUC | tf_LR_8 | clase_0 [No Maq]
True Positive Rate
False Positive Rate
ROC curve Train 1 (area = 0.94)
ROC curve Train 2 (area = 0.91)

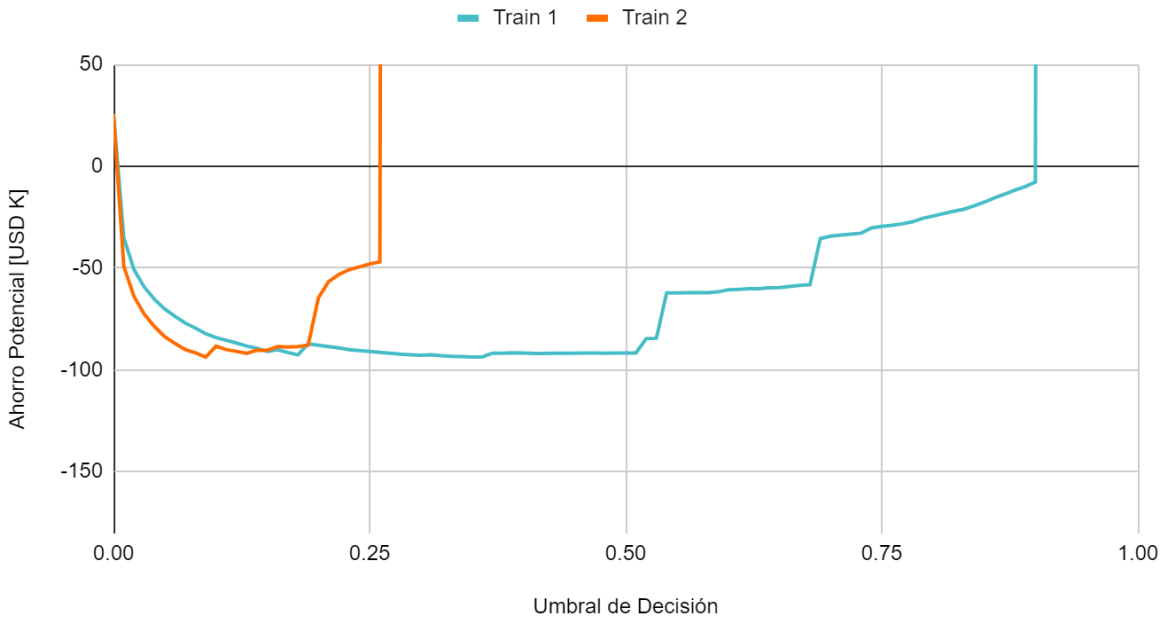

ROC - AUC | tf_LR_8 | clase_1 [Maq]
True Positive Rate
False Positive Rate
ROC curve Train 1 (area = 0.94)
ROC curve Train 2 (area = 0.91)

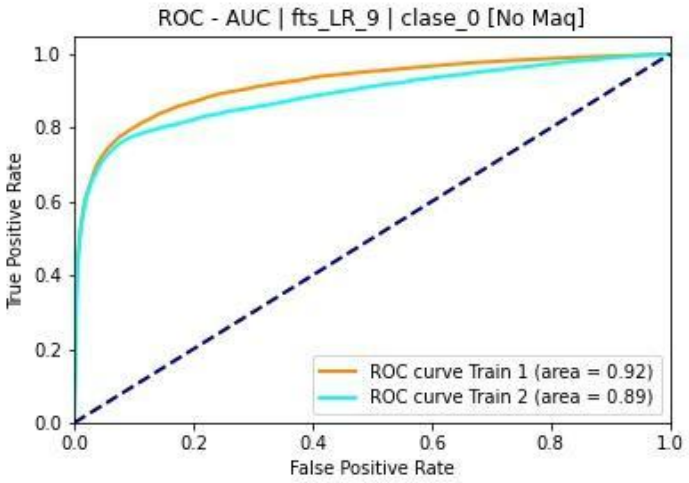
- **Experimento 9**: el descriptor está conformado por el título del producto + la variable seller_id del producto.

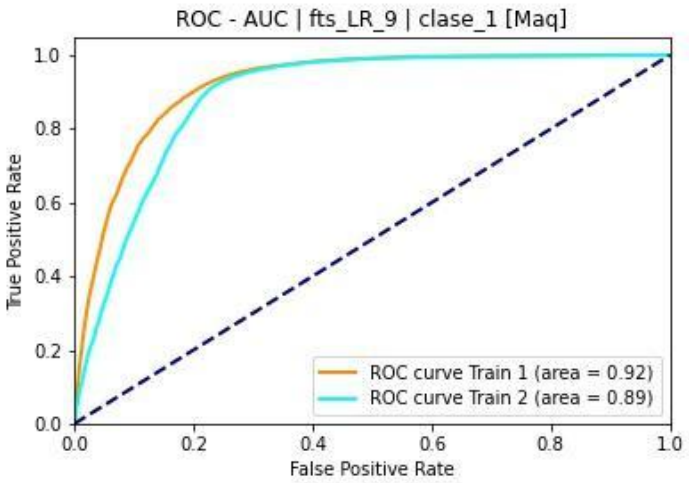


Principales Características	**Entrenamiento**	
	Train 1	**Train 2**
Transformador	Fast Text Supervisado	
Clasificador	Regresión Logística	
Mejor Umbral	0.35	0.09
% de Ahorro	93.712 (58%)	93.788 (58%)
Precisión Maquinables	99%	99%
Recall Maquinables	93%	96%
Precisión No Maquinables	26%	37%
Recall No Maquinables	77%	68%

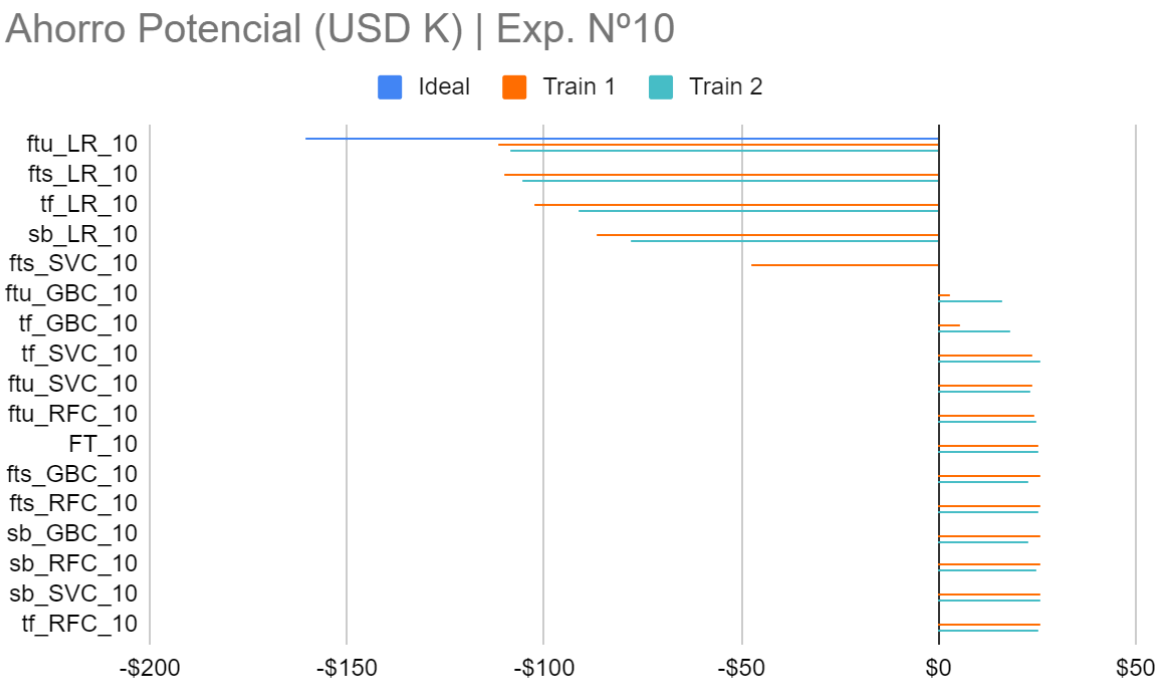
Función de Ahorro | Exp. nº 9 | Iter. fts_LR_9





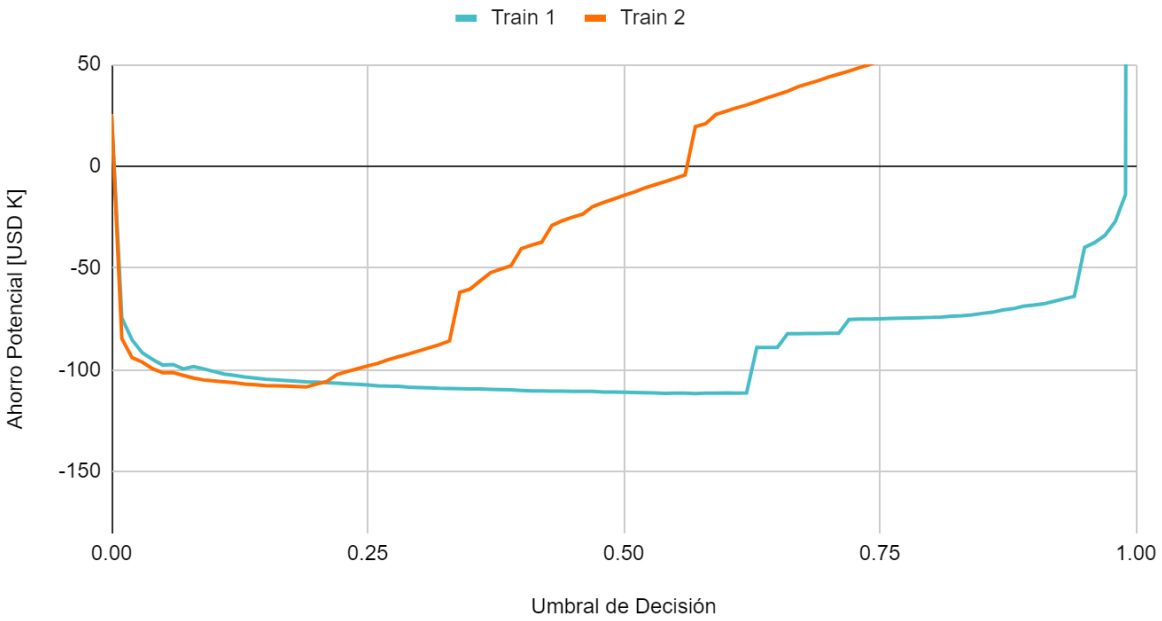


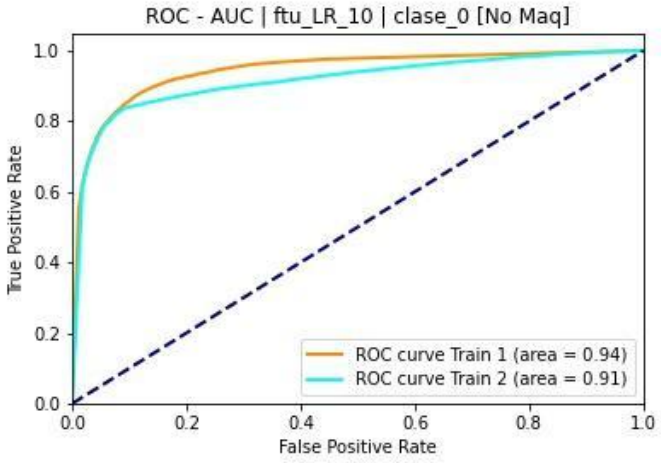
- **Experimento 10**: el descriptor está conformado por el título del producto + la variable brand_name del producto.

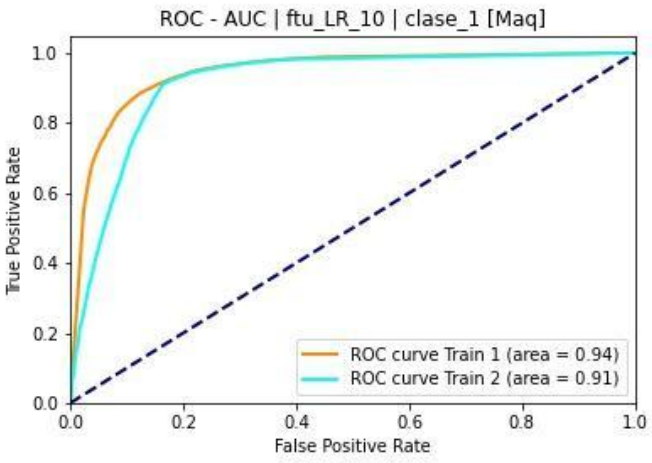


Principales Características	Entrenamiento	
	Train 1	**Train 2**
Transformador	Fast Text No Supervisado	
Clasificador	Regresión Logística	
Mejor Umbral	0.57	0.19
% de Ahorro	111.676 (69%)	108.480 (67%)
Precisión Maquinables	100%	99%
Recall Maquinables	89%	91%
Precisión No Maquinables	20%	23%
Recall No Maquinables	88%	84%

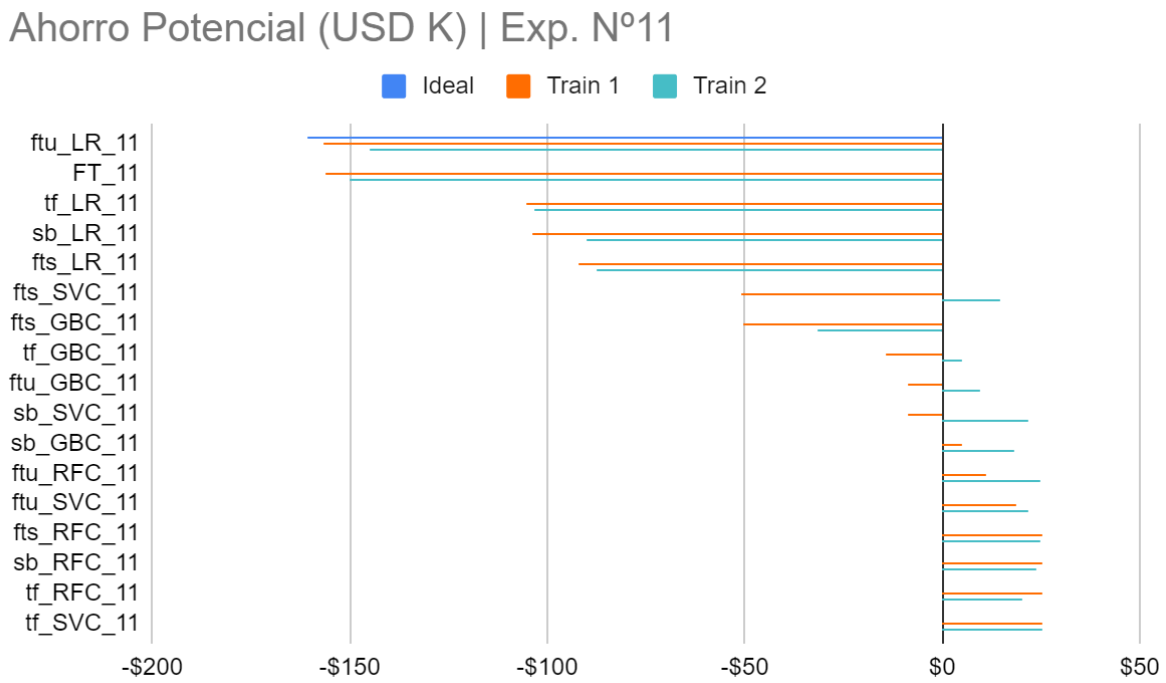
Función de Ahorro | Exp. nº 10 | Iter. ftu_LR_10





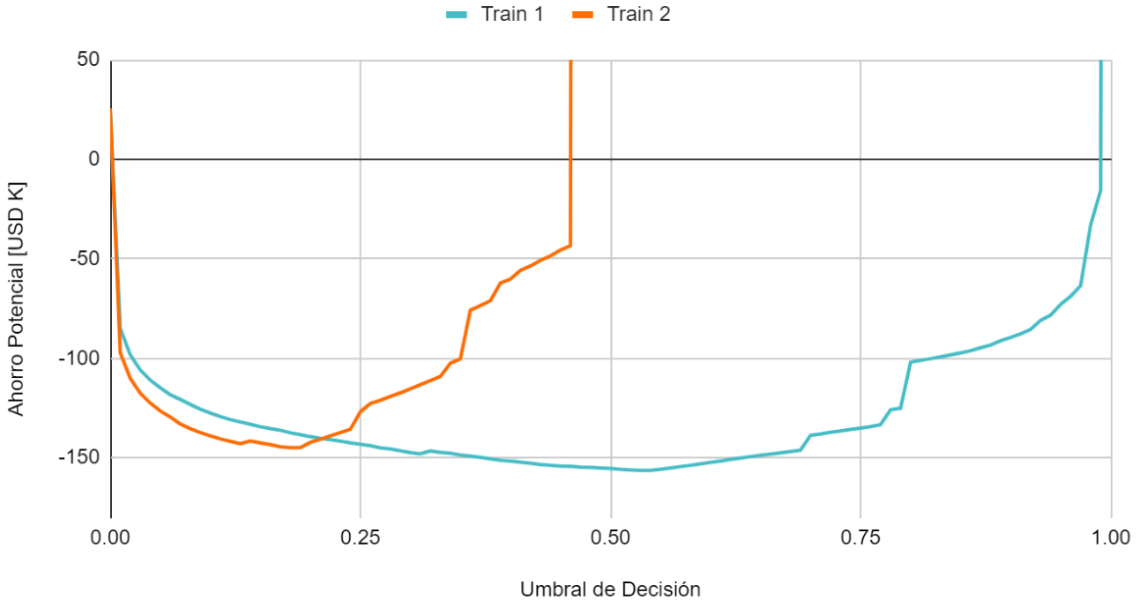


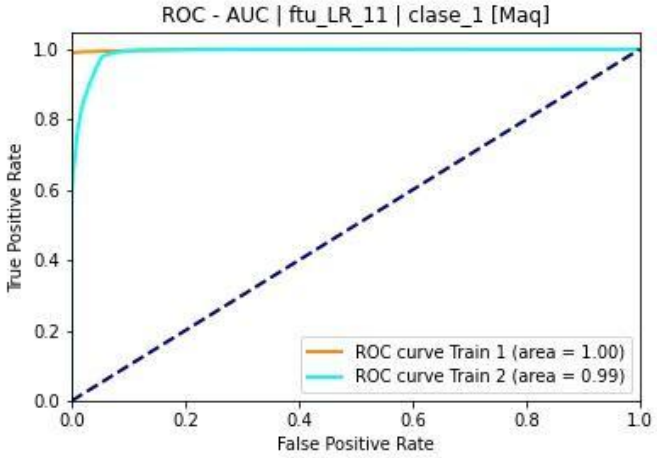
- **Experimento 11**: el descriptor está conformado por el título del producto + la variable L1 del producto.

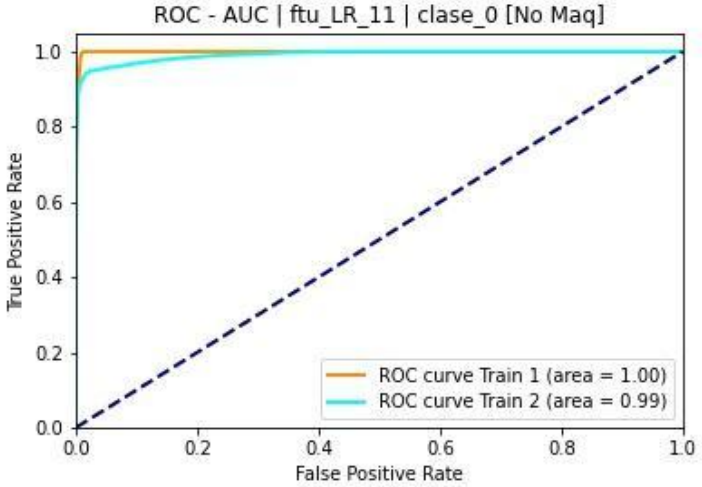


Principales Características	Entrenamiento	
	Train 1	**Train 2**
Transformador	Fast Text No Supervisado	
Clasificador	Regresión Logística	
Mejor Umbral	0.54	0.19
% de Ahorro	156.398 (97%)	144.915 (90%)
Precisión Maquinables	99%	99%
Recall Maquinables	99%	98%
Precisión No Maquinables	73%	57%
Recall No Maquinables	99%	95%

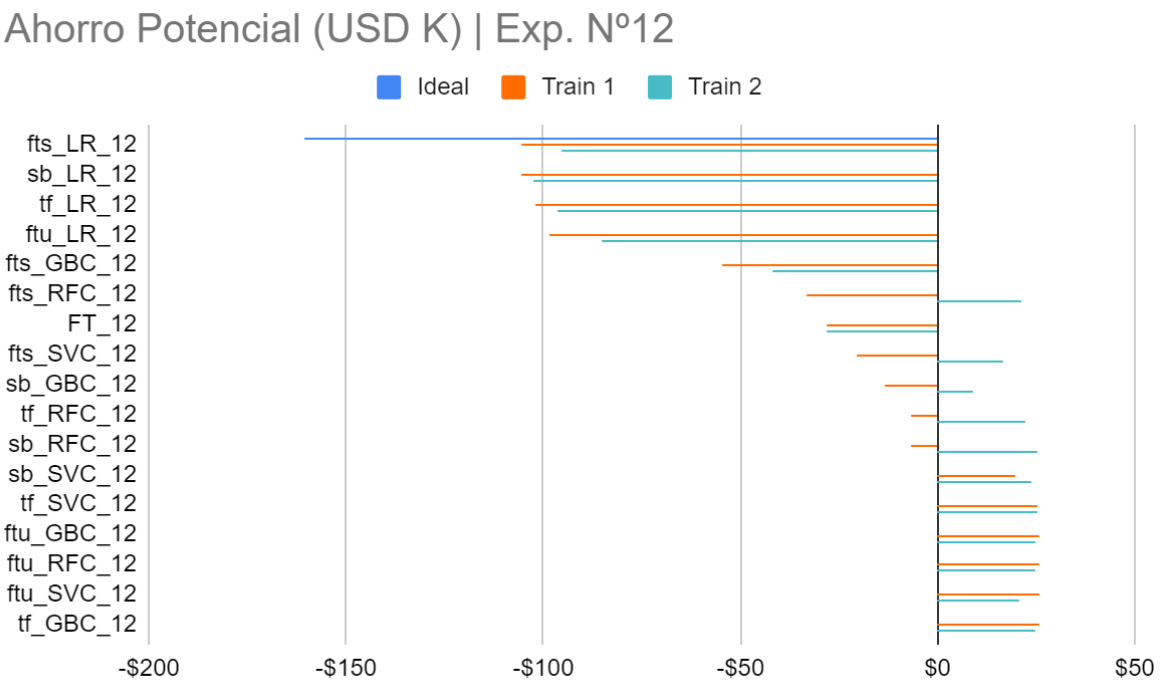
Función de Ahorro | Exp. nº 11 | Iter. ftu_LR_11





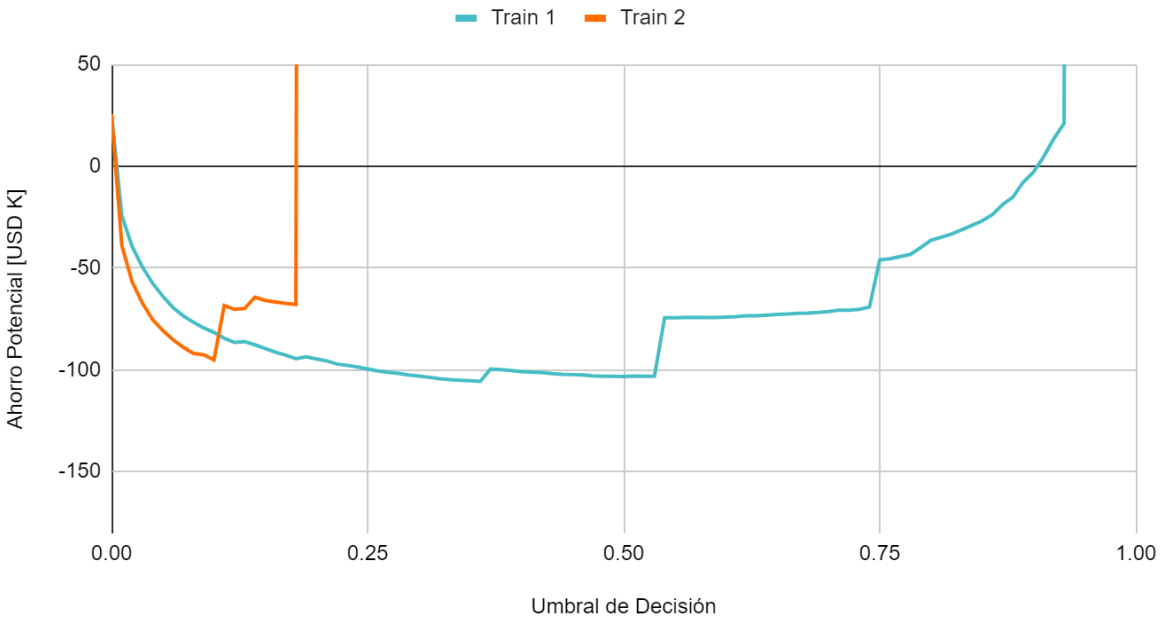


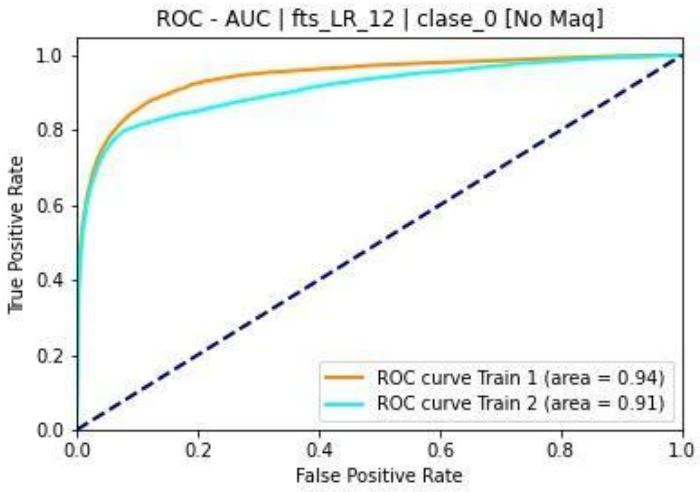
- **Experimento 12**: el descriptor está conformado por el título del producto + la variable L2 del producto.

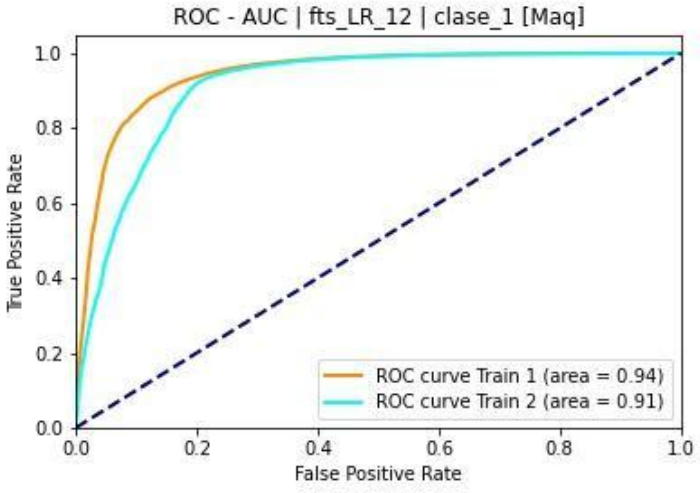


Principales Características	**Entrenamiento**	
	Train 1	**Train 2**
Transformador	Fast Text Supervisado	
Clasificador	Regresión Logística	
Mejor Umbral	0.36	0.1
% de Ahorro	105.728 (66%)	95.133 (59%)
Precisión Maquinables	99%	99%
Recall Maquinables	94%	97%
Precisión No Maquinables	30%	41%
Recall No Maquinables	80%	70%

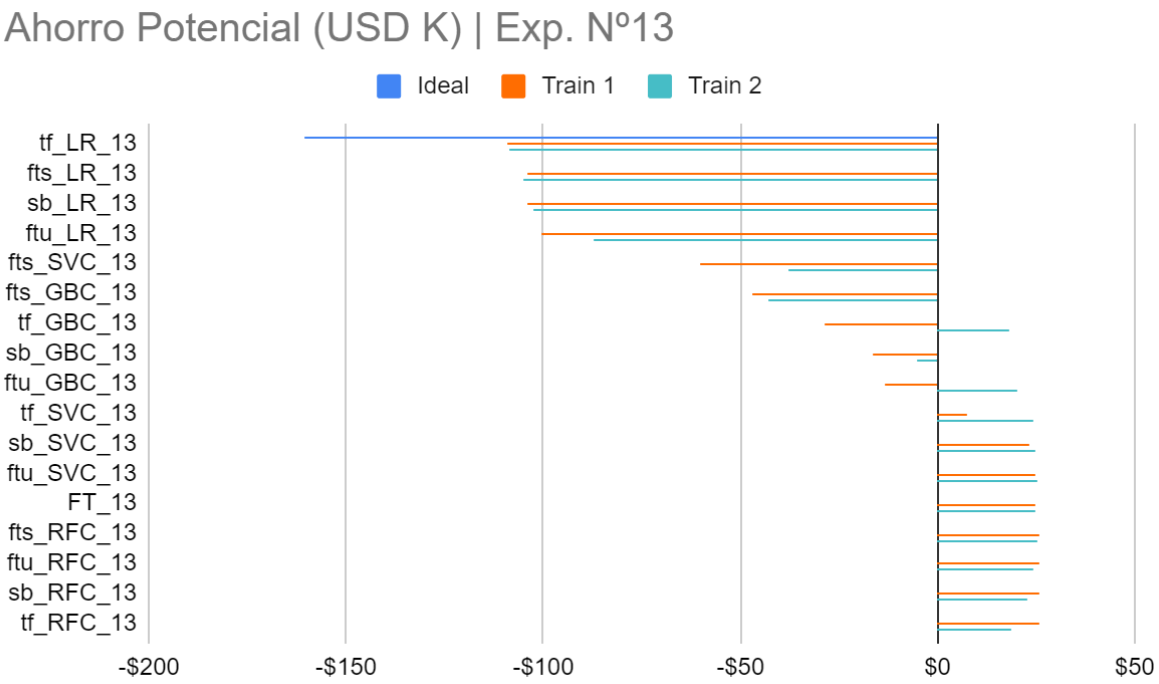
Función de Ahorro | Exp. nº 12 | Iter. fts_LR_12





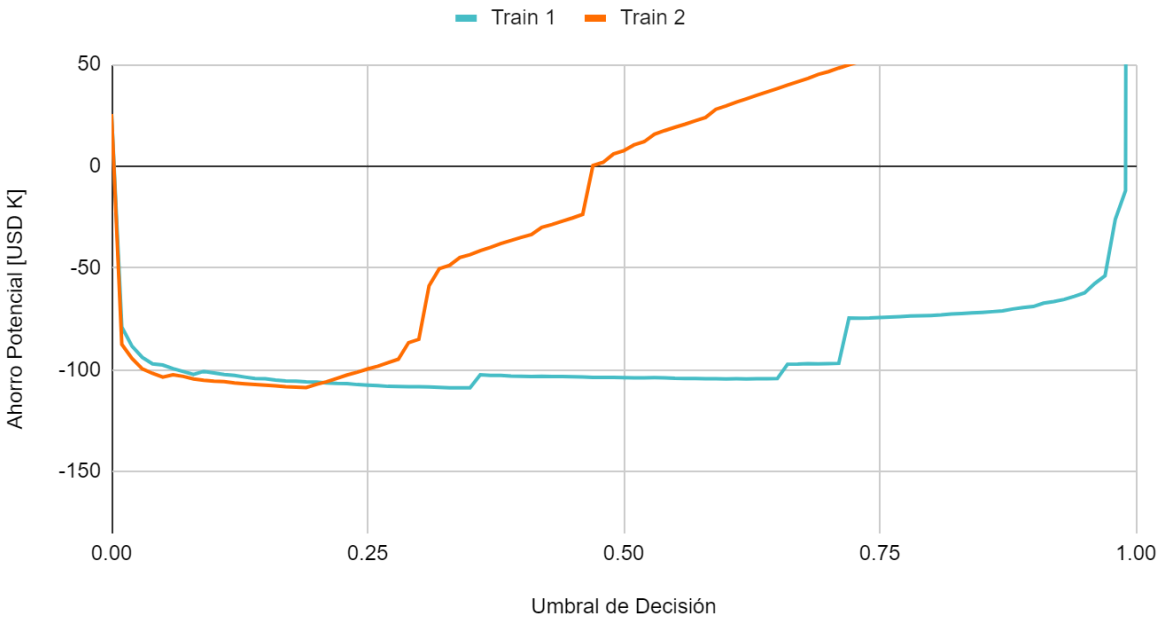


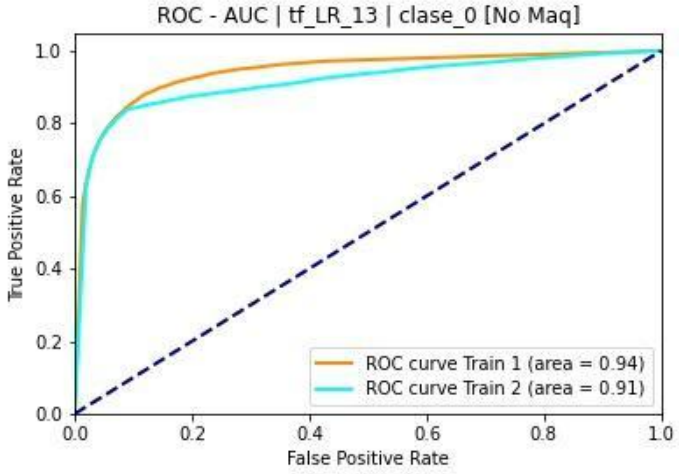
- **Experimento 13**: el descriptor está conformado por el título del producto + la variable L3 del producto.

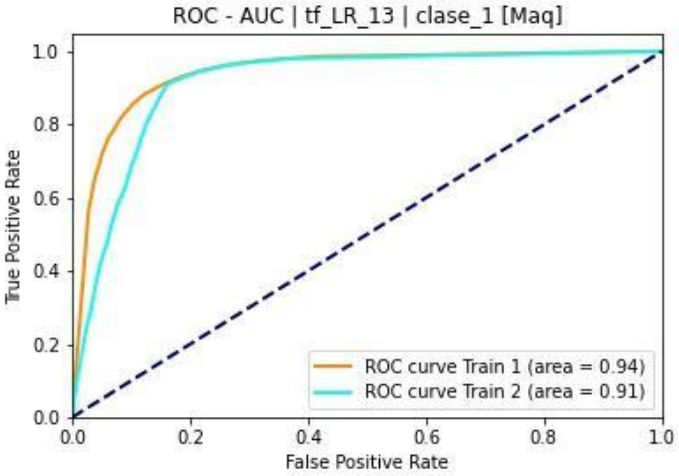


Principales Características	Entrenamiento	
	Train 1	**Train 2**
Transformador	Tensor Flow	
Clasificador	Regresión Logística	
Mejor Umbral	0.34	0.19
% de Ahorro	108.883 (68%)	108.747 (68%)
Precisión Maquinables	99%	99%
Recall Maquinables	92%	91%
Precisión No Maquinables	26%	23%
Recall No Maquinables	83%	84%

Función de Ahorro | Exp. nº 13 | Iter. tf_LR_13







Anexo E: Software

Este proyecto se ha desarrollado completamente utilizando el lenguaje de programación Python. Se han desarrollado varias funciones personalizadas para abordar tareas específicas del problema. El código y los archivos complementarios están disponibles en el repositorio.

https://github.com/tesis-utdt-envios/tesis-envios

Librerías Open Source para Python

- Pandas
- Numpy
- Fasttext
- Scikit Learn
- Imbalance Learn
- Unidecode
- Re
- Tensor Flow
- Sentence Transformer
- Matplotlib
- Seaborn
- Yellowbrick
- Shap

Otros Softwares

- Jupyter Notebooks
- Visual Studio Code
- Google Drive

8. Bibliografía

[1] Aranlund, A., & Wiktorsson, M. (2014). Automation in Internal Logistics: Strategic and Operational Challenges. En I. J. Management.

[2] Falcon Autotech. (2020). Obtenido de Falcon Autotech: falconautoonline.com/dimensioning-systems/

[3] Clark, J. (2018). Automatic Dimensioning: A Game Changer across the Supply Chain. Obtenido de DC Velocity: https://blogs.dcvelocity.com/one_off_sound_off/2018/09/automatic-dimensioning-a-game-changer-across-the-supply-chain.html

[4] Ship Bob. (2020). Obtenido de Ship Bob: https://www.shipbob.com/blog/how-to-calculate-shipping-costs/

[5] Nasteski, V. (2017). Faculty of Information and Communication Technologies. An overview of the supervised machine learning.

[6] Kozyrkov, C. (2019). Linkedin. Obtenido de The simplest explanation of machine learning you'll ever read: https://www.linkedin.com/pulse/simplest-explanation-machine-learning-youll-ever-read-cassie-kozyrkov

[7] Gareth, J., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning with R. Sección 1.2.4. Springer.

[8] Burkov, A. (2019). The Hundred-Page Machine. Sección 1. Free Edition.

[9] Akter, S., & Fosso Wamba, S. (2016). Electronic Markets: Big data analytics in E-commerce: a systematic review and agenda. Springer.

[10] Chopra, A., Prashar, A., & Sain, C. (2013). Natural Language Processing. International Journal of Technology Enhancements and Emerging Engineering Research.

[11] Código ASCII. (s.f.). Obtenido de https://elcodigoascii.com.ar/

[12] Wirth, R., & Hipp, J. (2000). CRISP-DM: Towards a standard process model for data mining. In Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining.

[13] Wongsuphasawat, K., Liu, Y., & Heer, J. (2019). Goals, process, and challenges of exploratory data analysis: an interview study.

[14] Molin, S. (2019). Hands-On Data Analysis with Pandas: Efficiently perform data collection, wrangling, analysis, and visualization using Python. Packt Publishing Ltd.

[15] Rahm, E., & Do, H. H. (2000). Data cleaning: Problems and current approaches. IEEE Data Engineering.

[16] Vijayarani, S., Ilamathi, M. J., & Nithya, M. (2015). Preprocessing techniques for text mining-an overview. International Journal of Computer Science & Communication

[17] Bird, S., Klein, E., & Loper, E. (2009). Natural language processing with Python: analyzing text with the natural language toolkit. Sección 3. O'Reilly Media.

[18] El-Khair, I. A. (2006). Effects of stop words elimination for Arabic information retrieval: a comparative study. International Journal of Computing & Information Sciences.

[19] Daniel J., James M. (2020). Speech and Language Processing 3rd edition. Sección 2.

[20] Hastie, T., Tibshirani, R., & Friedman, J. (2015). The elements of statistical learning: data mining, inference, and prediction. Sección 2. Springer.

[21] Burkov, A. (2019). The Hundred-Page Machine. Sección 2.6. Free Edition.

[22] Alpaydin, E. (2020). Introduction to machine learning. Sección 10.7. MIT press.

[23] Alpaydin, E. (2020). Introduction to machine learning. Sección 13. MIT press.

[24] Burkov, A. (2019). The Hundred-Page Machine. Sección 5.4. Free Edition.

[25] Hofmann, M. (2006). Support vector machines-kernels and the kernel trick.

[26] Hastie, T., Tibshirani, R., & Friedman, J. (2015). The elements of statistical learning: data mining, inference, and prediction. Sección 12. Springer.

[27] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning. Sección 8. Springer.

[28] Machine Learning @Berkeley, (2017). Decision Trees and Ensemble Models. Obtenido de Medium: https://medium.com/@ml.at.berkeley/machine-learning-crash-course-part-5-decision-trees-and-ensemble-models-dcc5a36af8cd

[29] Burkov, A. (2019). The Hundred-Page Machine. Sección 3.3. Free Edition.

[30] Hastie, T., Tibshirani, R., & Friedman, J. (2015). The elements of statistical learning: data mining, inference, and prediction. Sección 15. Springer.

[31] Burkov, A. (2019). The Hundred-Page Machine. Sección 7.5.2. Free Edition.

[32] Alpaydin, E. (2020). Introduction to machine learning. Sección 17.7. MIT press.

[33] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning. Pág. 314-318, 323, 337. Springer.

[34] Friedman, J. H. (2002). Stochastic gradient boosting. Computational statistics & data analysis.

[35] Fast Text Librería - https://fasttext.cc/

[36] Zolotov, V., & Kung, D. (2017). Analysis and optimization of fasttext linear text classifier. IBM T. J. Watson Research Center.

[37] Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. Facebook AI Research.

[38] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics.

[39] Rojas, R. (1996). The backpropagation algorithm. In Neural networks. Springer.

[40] Daniel J., James M. (2020). Speech and Language Processing 3rd edition. Sección 5.6.

[41] Cui, Y., Jia, M., Lin, T. Y., Song, Y., & Belongie, S. (2019). Class-balanced loss based on effective number of samples. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.

[42] Google. (s.f.). Google Tensor Hub. Obtenido de Text Embedding: https://tfhub.dev/google/universal-sentence-encoder-multilingual/3

[43] Google. (s.f.). SBERT. Obtenido de Sentence Transformer: https://www.sbert.net/

[44] Burkov, A. (2019). The Hundred-Page Machine. Sección 5.1. Free Edition.

[45] [34] Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E. B., & Turaga, D. S. (2017). Learning Feature Engineering for Classification.

[46] Scott, S., & Matwin, S. (1999, June). Feature engineering for text classification. ICML.

[47] Daniel J., James M. (2020). Speech and Language Processing 3rd edition. Sección 6.

[48] Schnabel, T., Labutov, I., Mimno, D., & Joachims, T. (2015). Evaluation methods for unsupervised word embeddings.

[49] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching Word Vectors with Subword Information. Facebook Research AI.

[50] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Learning from imbalanced data sets (Vol. 11). Springer.

[51] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research.

[52] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Learning from imbalanced data sets. Secciones 5.1 y 5.2. Springer.

[53] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Learning from imbalanced data sets. Sección 5.6. Springer.

[54] Raschka, S. (2014). Implementing a Principal Component Analysis (PCA). Obtenido de SebastianRaschka: https://sebastianraschka.com/Articles/2014_pca_step_by_step.html

[55] Alpaydin, E. (2020). Introduction to machine learning. Sección 10.5. MIT press.

[56] Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of machine learning research.

[57] Alpaydin, E. (2020). Introduction to machine learning. Sección 5. MIT press.

[58] Burkov, A. (2019). The Hundred-Page Machine. Sección 5.6. Free Edition.

[59] Fan, J., Upadhye, S., & Worster, A. (2006). Understanding receiver operating characteristic (ROC) curves.

[60] Correa Bahnsen, A. (2015). Example-dependent cost-sensitive classification with applications in financial risk modeling and marketing analytics.

[61] Frazier, P. I. (2018). A tutorial on Bayesian optimization.