Laura Jiménez Moreno

2020 - 2021

Aplicación de un modelo de

MACHINE LEARNING A LA ESTRATEGIA CRO DE UN E-COMMERCE

KSchool
Máster de Data Science
TRABAJO DE FIN DE MÁSTER

CONTENIDO

CONTENIDO 1
ANTES DE EMPEZAR 2
DIRECTORIO DE NOTEBOOKS 3
INTRODUCCIÓN 4
OBJETIVOS 4
EL E-COMMERCE 5
LOS DATOS 5
- **EXTRACCIÓN DE DATOS DE GOOGLE ANALYTICS** 5
  - **AUTENTICACIÓN EN LA API** 5
  - **REPORT REQUESTS** 6
- **EXTRACCIÓN DE DATOS DE PRESTASHOP** 11
- **FICHEROS DE DATOS FINALES** 12
- **DICCIONARIO DE DATOS (BRUTOS)** 13

METODOLOGÍA Y PLANIFICACIÓN 14
ELECCIÓN Y MODELO BASE 15
LAS VARIABLES: ANÁLISIS EXPLORATORIO Y 'FEATURE ENGINEERING' 16
- **VARIABLES NUMÉRICAS** 16
- **VARIABLES CATEGÓRICAS** 19

OPTIMIZACIÓN DEL MODELO 20
- **SELECCIÓN DE VARIABLES** 20
- **MANEJO DE CLASES DESBALANCEADAS** 21
- **OPTIMIZACIÓN DE HIPERPARÁMETROS DE RANDOM FOREST** 22
- **CALIBRACIÓN DEL MODELO** 23

INTERPRETACIÓN DEL MODELO 25
- **INTERPRETABILIDAD GLOBAL** 25
- **INTERPRETABILIDAD LOCAL** 27

MODELO FINAL 28
INTERACTIVE FRONTEND 30
CONCLUSIONES E IMPLICACIONES 31
PRINCIPALES REFERENCIAS UTILIZADAS 32

ANTES DE EMPEZAR

1. CLONE EN LOCAL EL REPOSITORIO DE GIT HUB:
   https://github.com/Laurajmoreno/kschool_masterDS_TFM

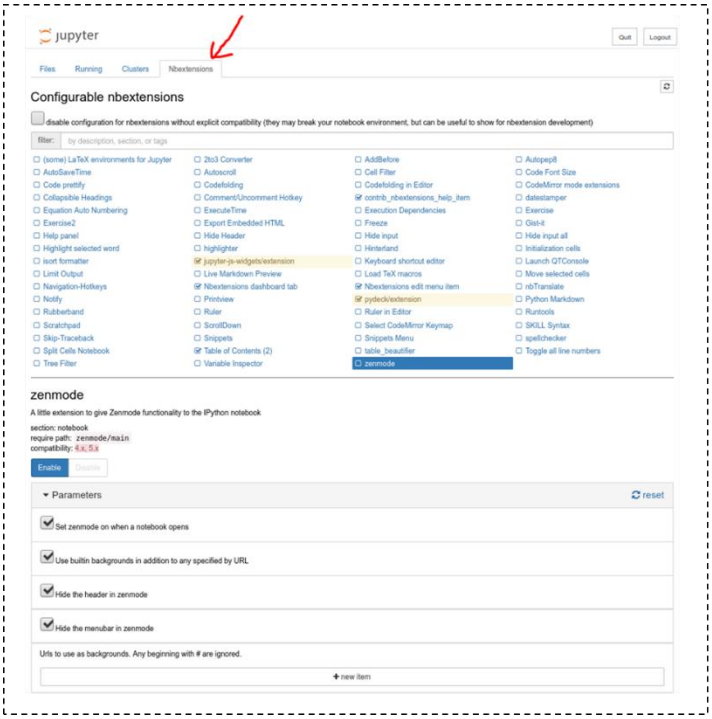
2. EN EL MISMO DIRECTORIO EN EL QUE HAYA CREADO EL REPOSITORIO, CREE UNA NUEVA CARPETA LLAMADA **`data`**
   - En el conjunto de los notebooks, ***data root*** ha sido definido como **`"../data/"`**

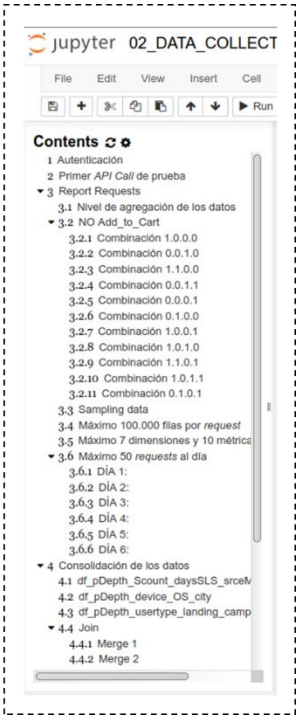
3. CONECTESE A LA **CUENTA DE GMAIL** QUE LE HA SIDO FACILITADA POR CORREO ELECTRÓNICO Y ABRA LA CUENTA DE **GOOGLE DRIVE** ASOCIADA.
   - Descargue y descomprima los ficheros de **'data'** en el interior de la carpeta creada en el apartado anterior.
   - Descargue y descomprima los ficheros de **'ficheros a incluir en el interior del repositorio'** en el interior del repositorio

4. CREE EL ENTORNO CONDA A PARTIR DEL FICHERO **environment.yml** DEL REPOSITORIO Y ACTÍVELO (todos los paquetes fueron instalados con conda y conda-forge, es posible que tarden en instalarse)

5. EJECUTE **JUPYTER NOTEBOOK**

6. PARA FACILITAR LA NAVEGACIÓN EN EL INTERIOR DE LOS NOTEBOOKS, SE RECOMIENDA ACTIVAR **NBEXTENSIONS** EN LA HOME DE JUPYTER NOTEBOOK:





DIRECTORIO DE NOTEBOOKS

01_GOOGLE_ANALYTICS_API_AUTHENTICATION

02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS

03_MODEL_SELECTION_AND_BASELINE

04_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_GA-FEATURES

05_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_PRICE-DISCOUNT

06_ DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA

07_RANDOM_FOREST_MODEL_FEATURE_BY_FEATURE

08_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA_PAGE-PATH-TEST'

09_ RANDOM_FOREST_MODEL_OPTIMIZATION

10_RANDOM_FOREST_MODEL_INTERPRETATION

11_RANDOM_FOREST_MODEL_FRONTEND

INTRODUCCIÓN

Gran parte de los esfuerzos de marketing en un *e-commerce* se centran en atraer tráfico a la web. Sin embargo, no siempre se aplican los mismos recursos para retener a esos usuarios y garantizar su conversión. Como respuesta a esta situación, en los últimos años, ha surgido un nuevo ámbito de aplicación denominado **CRO (*Conversion Rate Optimization*)**, que busca mejorar el ratio de las conversiones sobre el número de visitas. En la práctica, los profesionales dedicados a esta disciplina estudian los flujos de tráfico, los ratios de las distintas páginas vistas, establecen hipótesis, realizan A/B testing, etc. En su mayoría, siguen procesos bastante manuales y rudimentarios a partir de las herramientas predefinidas que facilitan Google Analytics y otras plataformas.

Ahora bien, la mayoría de estos procesos podrían **agilizarse, optimizarse y automatizarse** con modelos de Aprendizaje Automático. Gracias a ellos, podemos identificar patrones de conducta y medir la intención de compra a partir de aquellos factores o variables que determinen (o no) la conversión de un usuario. Mediante el uso de un proceso generativo probabilístico se puede **modelizar el comportamiento** del cliente a tiempo real, **predecir su toma de decisiones** en contextos específicos y **adoptar medidas** adecuadas para favorecer su conversión.

Con este proyecto, se busca establecer una **metodología de trabajo inicial** con la que identificar patrones de conducta que induzcan a la conversión del usuario y que puedan ser utilizados después para influir en la decisión de compra. El estudio se enmarca en el contexto de una tienda online al por menor, que opera fundamentalmente en el mercado español.

OBJETIVOS

OBJETIVO DE NEGOCIO:	• **Mejorar la tasa de conversión** del *e-commerce*.
OBJETIVOS TFM:	• **Explicar los factores** que influyen en la conversión del usuario • **Predecir la conversión** en función de las dimensiones que puedan influir y podamos capturar.

El objetivo de este proyecto es identificar los factores que explican la conversión de un cliente en un *e-commerce* en particular, con el fin de mejorar su **tasa de conversión.** Entendemos por tasa de conversión el resultado de dividir el número de conversiones por el número total de visitas para un determinado periodo:

$$\text{TASA DE CONVERSIÓN} = \frac{\text{CONVERSIONES}}{\text{VISITAS WEB}}$$

En concreto, se busca predecir, por medio de **un modelo de clasificación**, la conversión de un cliente a partir del máximo número de dimensiones que puedan influir y podamos capturar:

PRODUCTO	CARACTERÍSTICAS DE					CONVERSIÓN
	PRODUCTO	URL	TEMPORALES	USUARIO	SESIÓN	
	· Categoría · Precio · Descuento ·	· Tipo de página · Formato de página	· Mes · Día de la semana · Hora	· Ciudad · Número sesión · Dispositivo	· Fuente de tráfico · Landing	

EL E-COMMERCE

Galileo61 es una farmacia ubicada en Madrid. Además de tienda física, desde hace 3 años, cuenta con un establecimiento online **https://www.galileo61.com/** que ha ido creciendo con el transcurso del tiempo hasta alcanzar cifras muy notables que pueden ser de gran valor para el proyecto:

- 30.000/40.000 visitas de media al mes
- 400 / 800 pedidos de media al mes
- 1-2% de tasa de conversión de media al mes.

LOS DATOS

Para nuestro análisis, hemos empleado principalmente datos de **Google Analytics 4** (en adelante GA4), extraídos a partir de la API que facilita Google. Se han considerado también otros datos (precios y descuentos) procedentes del **Prestashop** de la web y fuentes privadas del establecimiento.

EXTRACCIÓN DE DATOS DE GOOGLE ANALYTICS

> Notebooks:

- 01_GOOGLE_ANALYTICS_API_AUTHENTICATION.ipynb
- 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb

AUTENTICACIÓN EN LA API

Configuración de la API de Google Analytics

> Notebook 01_GOOGLE_ANALYTICS_API_AUTHENTICATION.ipynb

En este primer notebook, se enumeran de manera detallada los pasos que se siguieron para configurar la **API de Google Analytics**. A continuación, se recopila brevemente el procedimiento que establece Google:

1. **Crear un proyecto** en Google Cloud Console y habilitar **'Analytics Report API'**
2. En el panel de administración de 'API y servicios', añadir las credenciales:
    - Configurar el tipo de **credenciales**
    - Configurar la **pantalla de consentimiento**

- Crear y descargar las credenciales (fichero ***client_secrets.json***)

3. Ya en nuestro sistema:
    - **Instalar la librería** conda install -c conda-forge google-api-python-client en el entorno.
    - Crear el fichero ***HelloAnalytics.py*** a partir del código disponible en: https://developers.google.com/analytics/devguides/reporting/core/v3/quickstart/service-py
    - Sustituir el parámetro VIEWID con el **ID de la cuenta de Google Analytics** de Galileo61 en el fichero HelloAnalytics.py
    - Guardar los ficheros ***HelloAnalytics.py*** y ***client_secrets.json*** en el directorio de trabajo del notebook en el que se vaya a llamar a la API.
    - **Ejecutar** en el notebook el fichero HelloAnalytics.py. Si el código de muestra se ha ejecutado correctamente, la celda devuelve el número total de sesiones de los últimos 7 días.

A mayores, al ejecutar el código de muestra facilitado por Google, se observó que había algunos errores de sintaxis que hubo que corregir. Asimismo, hubo que descargar una segunda librería llamada **oauth2client**.

Como se comentó al inicio, todos los detalles se pueden ver en el notebook 01_GOOGLE_ANALYTICS_API_AUTHENTICATION.ipynb. Dado que se subsanaron los errores, los resultados de las celdas no son replicables. Se recomienda no ejecutarlas.

Proceso de autenticación

> Notebook 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb

Para poder replicar las consultas a la API de Google Analytics que se detallan en este segundo notebook, es necesario acreditarse antes. Para ello, siga los pasos que se especifican al inicio en el apartado *'1. Autenticación'*.

REPORT REQUESTS

> Notebook 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb

Nivel de agregación de los datos

A la hora de plantear el nivel de agregación de los datos, hubo que tener en cuenta distintas cuestiones.

Por un lado, la métrica que se iba a considerar como **'conversión'** y, por lo tanto, como *target* del modelo. Había dos opciones:

- **`ga:productAddsToCart` –** Número de veces que el producto fue añadido al carrito.
- **`ga:productCheckouts` –** Número de veces que el producto fue incluido en el proceso de *check-out*. Google entiende por *check-out* la 'acción que inicia el proceso de pago de uno o varios productos'.

Finalmente, se eligió la primera por las siguientes razones:

- Para compensar el **desequilibrio entre la clase positiva y negativa** del modelo. Las observaciones con clase positiva en `ga:productAddsToCart` representaban un 2% del conjunto de datos. En el caso de `ga:productCheckouts`, este valor era inferior.
- El proceso de *check-out* se lleva a cabo en una misma página para todos los productos, por lo que **se pierde información relevante** sobre la ubicación en la que tiene lugar la conversión.
- Conveniencia para definir **los eventos de la clase negativa** como veremos más adelante.
- Aun tratándose de una *microconversión* y no de la conversión final, se siguen preservando los objetivos del proyecto.

En segundo lugar, la versión gratuita de GA4 devuelve **datos agregados** en sus informes. Es decir, si solicitamos los eventos `ga:productAddsToCart` para los últimos siete días, nos devolverá el total de veces que se añadieron productos al carrito en ese periodo. Cabe señalar, que la versión de pago que da acceso a todos los hits de la sesión de un usuario tiene un coste mínimo anual de 150.000 $.

Para solventar este problema, hubo que **desagregar los datos**, añadiendo distintas dimensiones a los informes que dieran respuesta a las siguientes cuestiones:

PREGUNTA	DIMENSIÓN	Descripción de GA4
¿**Qué** se añadió al carrito?	`ga:productSku` `ga:productName`	• *The product SKU, defined in the ecommerce tracking application, for purchased items.* • *The product name, supplied by the ecommerce tracking application, for purchased items.*
¿**Dónde** se añadió al carrito?	`ga:pagePath`	*A page on the website specified by path and/or query parameters.*
¿**Cuándo** se añadió al carrito?	`ga:dateHourMinute`	*Combined values of ga:date, ga:hour and ga:minute formated as YYYYMMDDHHMM.*
¿**Quién** lo añadió al carrito?	**ClientID** no está disponible en la versión gratuita.	

Más información sobre dimensiones y métricas de Google Analytics en:
https://ga-dev-tools.appspot.com/dimensions-metrics-explorer/

Tal y como se aprecia, en la tabla anterior, todas las cuestiones esenciales pudieron ser tratadas salvo el **identificador de cliente**. Esta dimensión no está disponible por defecto en GA4 y, aunque puede ser incluida como CUSTOMDIMENSION, no lo estaba en el momento de la extracción de los datos. De haberse configurado en ese momento, sólo hubiera podido contemplarse para los datos que hubiera recogido Google de ahí en adelante y no en el histórico.

Para solventar este problema, se incluyeron otras dimensiones que facilitaran la **identificación de un usuario único**. Por ejemplo:

- **`ga:sessionCount`** – Índice de sesión de un usuario a modo de contador.
- **`ga:daysSinceLastSession`** – Número de días transcurridos desde la última vez que el usuario visitó la web.
- **`ga:sourceMedium`** – Fuente de tráfico y tipo (responde a la pregunta de dónde procede el usuario).

```
#DATAFRAME WITH DATA:
df = pd.DataFrame(list_)
print(df.shape)
df.head()
```

```
(2121, 7)
```

	ga:productName	ga:pagePath	ga:dateHourMinute	ga:sessionCount	ga:daysSinceLastSession	ga:sourceMedium	ga:productAddsToCart
0	▷▷HELIOCARE 360 Water Gel SPF 50+	/es/anticelulíticos/4402-heliocare-360-water-g...	202101251823	1	0	google / organic	1
1	▷ Envío Gratuito |ENDOCARE TENSAGE CONTORNO DE...	/es/contorno-de-ojos-para-bolsas/1124-endocare...	202101241058	1	0	google / cpc	1
2	▷▷ GUAM - FANGO GEL FIR ACCION 300 ml ANTICEL...	/es/anticelulíticos/4265-▷▷-guam-fango-gel-fir...	202101302229	1	0	google / organic	1
3	▷▷Emulsión limpiadora JAPONESA Kire Gema Herre...	/es/brand/356-gema-herrerias	202101241033	1	0	m.facebook.com / referral	1
4	▷▷Emulsión limpiadora JAPONESA Kire Gema Herre...	/es/module/iqitsearch/searchiqit	202101281544	2	0	google / organic	1

Al añadir todas las dimensiones anteriores, se desdobló como es lógico el número de observaciones del informe y la mayor parte de ellas pasaron a adoptar valores igual a 1 en la columna `ga:productAddsToCart`. Sólo algunas filas siguieron mostrando valores superiores pero, tras un análisis exploratorio, se asumió que cada fila representaba un único usuario.

- ver notebook '02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS > '3. Report Requests' > '3.1. Nivel de agregación de los datos')
- ver notebook 04_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_GA-FEATURES > 3. Target

Por último, faltaba por incluir al informe todas aquellas **observaciones en las que el usuario NO había convertido**. Hasta ahora, GA4 sólo devolvía eventos en los que se había añadido un producto al carrito. Para ello, fue necesario incluir nuevas métricas que registrarán el resto de comportamientos que pudiera estar realizando el cliente en su lugar:

- **`ga:productListViews`** – GA4 se refiere a esta métrica como el número de veces que el producto apareció en una lista de productos. Estas listas pueden ser de una categoría específica, de resultados de una búsqueda o de productos recomendados. En este caso, asumimos que el usuario se encuentra explorando esta lista en el que se le muestra el producto en cuestión.
- **`ga:productListClicks`** – GA4 describe esta métrica como el número de veces que los usuarios hicieron click sobre un producto que les apareció en la lista.
- **`ga:productDetailViews`** – Número de veces que el usuario vio el producto en su página específica (ficha de producto).

En el notebook (*'02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb' > '3. Report Requests' > '3.2. NO Add_to_Cart'*), se pueden ver las distintas combinaciones de eventos que resultaron y como estos capturan correctamente todos los posibles comportamientos del usuario con respecto al producto y la conversión.

Otros aspectos de interés para la extracción de los datos

Al solicitar los informes hubo que sortear y solventar otros inconvenientes que se enumeran a continuación.

DATOS SAMPLEADOS:

> Notebook 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb' > '3. Report Requests' > '3.3. Sampling Data'

En ocasiones la versión gratuita de GA4 devuelve **muestras de datos**. Esto sucede cuando:

- El informe solicitado tiene en cuenta más de 500.000 sesiones.
- El informe solicitado cuenta con más de 1.000.000 de valores en una de las dimensiones.

Para verificar que los informes descargados no correspondían a muestras, se modificó el código de la *request* para que también devolviera el número de muestras leídas (***'samplesReadCounts'***) y presentes (***'samplingSpaceSizes'***), si las había.

En ninguna de las consultas realizadas se observaron datos muestrales.

MÁXIMO 100.000 FILAS POR REQUEST

> Notebook 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb' > '3. Report Requests' > '3.4 Máximo 100.000 filas por request'

La API gratuita de GA4 sólo devuelve informes de máximo 100.000 filas. Ahora bien, se puede acceder a las distintas páginas de los informes especificando el **índice del informe** a partir del cual se desean descargar los datos. Asimismo, podemos pedirle que devuelva el ***'NextpageToken'*** a un informe para saber dónde se ha quedado y realizar la siguiente solicitud a partir de él.

Para agilizar el proceso, se incluyó en el código de la *request* un **bucle** que, con las dos funcionalidades anteriores, permitiera descargar un informe completo ejecutando una única vez la celda.

MÁXIMO 7 DIMENSIONES Y 10 MÉTRICAS POR INFORME

> Notebook 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb' > '3. Report Requests' > '3.5 Máximo 7 dimensiones y 10 métricas por request'

Hasta aquí, la mayor parte de lo considerado en el conjunto de datos se refiere al *target*. Sin embargo, para el análisis, era necesario también recabar información de **otras dimensiones** que entrarán en juego a la hora de convertir o no. Tal y como se señalaba al inicio de este documento, se esperaban estudiar características específicas del producto, de la URL, del usuario y de la sesión que influyeran en la predicción del modelo.

Ahora bien, otra de las restricciones que impone la API de GA4 es que no se pueden exportar más de **7 dimensiones** (equivalentes a variables descriptivas) y **10 métricas** (variables cuantitativas) en una misma *request*. Asimismo, algunas de las métricas y dimensiones son incompatibles entre sí.

Para solventar esto, se tuvieron que exportar distintos informes que luego se unieron por medio de columnas comunes. Tras distintas pruebas, se establecieron como campos en común:

- **`ga:productSKU`** (dimensión)
- **`ga:pagePath`** (dimensión)
- **`ga:dateHourMinute`** (dimensión)
- **`ga:pageDepth`** (dimensión)
- **`ga:productListViews`** (métrica)
- **`ga:productListClicks`** (métrica)

- **`ga:productDetailViews`** (métrica)
- **`ga:productAddsToCart`** (métrica)

Cabe señalar que se optó por utilizar el **SKU en lugar del nombre del producto** para evitar problemas si, en el transcurso de los 3 años de actividad online, se había modificado el nombre del artículo. Ya más avanzados en el análisis, se cruzaron ambas referencias con un *dataset* distinto que relacionaba el **`ga:productSKU`** con su nombre.

Una vez seleccionadas las columnas comunes y visto que el resto de métricas que se habían elegido, como `ga:timeOnPage` o `ga:pageLoadTime`, eran incompatibles con nuestro informe básico, se escogieron las siguientes **9 dimensiones adicionales** y se distribuyeron en **3 informes distintos**:

INFORME	DIMENSIONES ADICIONALES

Informe 1	`ga:sessionCount`
	`ga:daysSinceLastSession`
	`ga:sourceMedium`

Informe 2	`ga:city`
	`ga:deviceCategory`
	`ga:operatingSystem`

Informe 3	`ga:userType`
	`ga:landingPagePath`
	`ga:campaign`

MÁXIMO 50 REQUESTS AL DÍA:

> Notebook 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb' > '3. Report Requests' > '3.6_Máximo 50 requests al día'

El último reto que planteó GA4 fue la imposibilidad de realizar más de **50 *requests* diarios**. La extracción de informes se hizo de manera progresiva en el transcurso de varios días. Se optó por solicitar **periodos de un mes** con el fin de monitorizar el número de consultas empleadas.

Se empezó por descargar el **Informe 1** (ver apartado anterior) por tratarse en su mayoría de **variables numéricas** que permitirían lanzar rápidamente un primer mínimo producto viable de nuestro modelo de Machine Learning, mientras se descargaban el resto de datos.

Cabe señalar que el marco temporal de 3 años que inicialmente se había contemplado se redujo al periodo comprendido entre **ABRIL 2019 y ENERO 2021**. En el momento de extraer los datos, se vio que no había datos disponibles anteriores. Tras contrastarlo con el equipo técnico de Galileo61, resultó que la opción *'Enhanced Ecommerce'* de Google Analytics no se implementó en la web hasta el segundo trimestre de 2019. Este módulo es el que permite la recogida de datos esenciales para nuestro análisis, tales como `ga:productListViews`, `ga:productListClicks`, `ga:productDetailViews` y `ga:productAddsToCart`.

AGRUPACIÓN Y CONSOLIDACIÓN DE DATOS:

> Notebook 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb' > '4. Consolidación de datos'

Una vez descargados todos los datos mensuales, se procedió a consolidarlos en un único *dataframe* para, finalmente, unir los informes por medio de un **INNER JOIN**. Se eligió este tipo de unión por que existía un volumen suficiente de datos (3.654.882 observaciones) para el análisis y evitaba los inconvenientes de gestionar posteriormente valores NAN.

DATOS DE FEBRERO Y MARZO 2021:

> Notebook 02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS.ipynb' > '4. Consolidación de datos'

A posteriori y una vez se tuvo el modelo, se descargaron los datos correspondientes al periodo comprendido entre el **1 de febrero y el 31 de marzo de 2021**, con el fin de evaluar la idoneidad del uso de técnicas de *oversampling* y *undersampling*.

EXTRACCIÓN DE DATOS DE PRESTASHOP

El **Prestashop** con el que se gestiona la tienda online permite consultar y exportar una serie de informes y catálogos que son los que se han utilizado para extraer las siguientes variables:

- Nombre del producto (asociado a la referencia SKU)
- Precio del producto
- Descuentos aplicados a todos los productos de la web por rangos de fechas.

Para las dos primeras variables se exportó el **catálogo de productos** en formato *csv*. Conviene señalar que estos datos se bajan con los valores actualizados en el momento preciso de la descarga (en nuestro caso el 09/02/2021), de modo que algunos de los productos del conjunto de datos de GA4 ya no estaban disponibles. De hecho, tampoco fue posible utilizar la columna relativa a la **categoría del producto**, ya que en Prestashop se denomina así a la sección en la que se encuentra almacenado el artículo en ese instante. En ese momento, muchos de los productos se encontraban en páginas genéricas como la HOME, la página TOP VENTAS… De haber utilizado estos valores, no hubiéramos podido garantizar su validez en el histórico de datos. En cambio, sí que fue posible conservar el **PVP**, ya que según el dueño de la farmacia rara vez suele cambiar.

En el caso de los **descuentos**, se hizo un *csv* a partir del histórico disponible en Prestashop con la siguiente información:

- Nombre del descuento
- Porcentaje de descuento
- Fecha de validez (inicio y fin)

FICHEROS DE DATOS FINALES

Los ficheros resultantes y que fueron utilizados para el proyecto son:

Ficheros	Notebooks
df_pDepth_Scount_daysSLS_srceMed_2019.csv df_pDepth_Scount_daysSLS_srceMed_2020.csv df_pDepth_Scount_daysSLS_srceMed_jan21.csv	03_MODEL_SELECTION_AND_BASELINE 04_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_GA-FEATURES
dfjoin_usertype_Scount_daysSLS_landing_campaign_srceMed_city_device_OS_2019_2020_jan21.csv	05_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_PRICE-DISCOUNT 06_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA 07_RANDOM_FOREST_MODEL_FEATURE_BY_FEATURE
product_2021-02-09_103558.csv	05_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_PRICE-DISCOUNT 06_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA 07_RANDOM_FOREST_MODEL_FEATURE_BY_FEATURE 09_ RANDOM_FOREST_MODEL_OPTIMIZATION 10_RANDOM_FOREST_MODEL_INTERPRETATION 11_RANDOM_FOREST_MODEL_FRONTEND
Descuentos.csv	05_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_PRICE-DISCOUNT 07_RANDOM_FOREST_MODEL_FEATURE_BY_FEATURE 09_ RANDOM_FOREST_MODEL_OPTIMIZATION 10_RANDOM_FOREST_MODEL_INTERPRETATION 11_RANDOM_FOREST_MODEL_FRONTEND
'no_carrito_no_pedido_df_2019_2020_jan21.csv'	09_ RANDOM_FOREST_MODEL_OPTIMIZATION 10_RANDOM_FOREST_MODEL_INTERPRETATION 11_RANDOM_FOREST_MODEL_FRONTEND
DescuentosFebMar21.csv	02_DATA_COLLECTION_FROM_GOOGLE_ANALYTICS
dfjoin_usertype_Scount_daysSLS_landing_campaign_srceMed_city_device_OS_price_disc_nocarrped_feb21_mar21.csv'	09_ RANDOM_FOREST_MODEL_OPTIMIZATION

Todos ellos están disponibles en el **directorio '*data*'** que acompaña a este proyecto.

DICCIONARIO DE DATOS (BRUTOS)

VARIABLE	DESCRIPCIÓN
ga:productSKU	Referencia numérica asignada al producto para poder ser identificado en el inventario.
ga:dateHourMinute	Fecha, hora y minuto en la que tuvo lugar la observación (formato: AAAAMMDDHHMM)
ga:pagePath	Hace referencia a una página en el sitio web. Designa la ruta o los parámetros de la consulta. Al ser usado junto al dominio, obtenemos la URL completa.
ga:pageDepth	Número de páginas visitadas por el usuario durante una sesión. Su valor es un histograma que cuenta las páginas vistas a lo largo de un rango posible de valores. En este cálculo, todas las sesiones tienen al menos una página vista y un porcentaje de ellas tendrán más.
ga:userType	Indica si el usuario es nuevo o no.
ga:sessionCount	El índice de sesión de un usuario. Cada sesión de un usuario único tendrá su propio índice incremental que inicia en 1 para la primera sesión. Las sesiones siguientes no modifican los índices anteriores. Por ejemplo, si un usuario ha visitado 4 veces la web, *sessionCount* para este usuario adoptará 4 valores distintos de 1 a 4.
ga:daysSinceLastSession	Número de días transcurridos desde que el usuario visitó por última vez la web.
ga:landingPagePath	Se refiere a la primera página que visita el usuario durante una sesión.
ga:campaign	Para campañas de monitorización manual, es el valor designado en el parámetro de tracking de la campaña. En el caso de autotagging de Adwords es el nombre de la campaña online que se utilizó para el dominio. Si no se utiliza ninguno, devuelve *(not set).*
ga:sourceMedium	Devuelve la fuente y el tipo de tráfico desde el que procede el usuario.
ga:city	Ciudad del usuario que Google obtiene a través de la IP o los IDs geográficos.
ga:deviceCategory	Tipo de dispositivo desde el que se conecta el usuario: PC, Tableta o Móvil.
ga:operatingSystem	Sistema operativo del dispositivo (Ej.: Windows, Linux, Macintosh, iOS…)

ga:productListViews	Número de veces que el producto apareció en una lista de productos. Estas listas pueden ser de una categoría específica, de resultados de una búsqueda o de productos recomendados.
ga:productListClicks	Número de veces que los usuarios hicieron click sobre un producto que les apareció en la lista.
ga:productDetailViews	Número de veces que el usuario vio el producto en su página específica (ficha de producto).
ga:productAddsToCart	Número de veces que el producto fue añadido al carrito.
Web_Discount	Descuento general aplicable a todos los productos que se añaden al carrito.
Product_price	PVP del producto.

METODOLOGÍA Y PLANIFICACIÓN

Como ya se mencionó anteriormente, el objetivo del proyecto es predecir si un usuario convertirá o no con respecto a un producto específico, así como identificar los factores que contribuyen a su conversión. Estamos, por lo tanto, ante **un problema de aprendizaje supervisado**, donde tendremos un conjunto de variables *input* que presumiblemente tendrán una influencia sobre el *output*.

Llegados a este punto, hubo que decidir si se trataba como un problema de regresión o de clasificación. En el apartado anterior de los datos, vimos cómo GA4 devolvía el número total de veces que se añadía el producto al carrito en una observación. Estos valores podían adoptar valores superiores a 1 pero, dada la poca frecuencia de este tipo de eventos y la falta de transparencia de estas observaciones, **se optó por tratar el problema en términos de clasificación**. Para ello, se convirtió `ga:productAddsToCart` en una **variable binaria**. Todas aquellas observaciones que hubiesen añadido el producto al carrito se transformaron en 1 y el resto mantuvieron su valor a 0.

Los pasos que se siguieron a continuación fueron los siguientes:

- **Elección del modelo** y establecimiento de un ***baseline*** en lo que a métricas se refiere, a partir de las variables numéricas disponibles.
  - Notebook 03_MODEL_SELECTION_AND_BASELINE
- **Análisis exploratorio** de las distintas variables independientes, su distribución y su relación con la variable objetivo.
  - Notebook 04_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_GA-FEATURES
  - Notebook 05_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_PRICE-DISCOUNT
  - Notebook 06_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA
  - Notebook 08_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA_PAGE-PATH-TEST'
- **Ingeniería y preprocesado de los datos** para su incorporación al modelo

- En paralelo a lo anterior, **lanzamiento de modelos parciales** que actuaban como *baseline* a medida que se iban añadiendo nuevas variables.
    - Notebook 07_RANDOM_FOREST_MODEL_FEATURE_BY_FEATURE
- **Optimización del modelo** por medio de técnicas de selección de variables, *oversampling/undersampling*, optimización de hiperparámetros y calibración del modelo.
    - Notebook 09_RANDOM_FOREST_MODEL_OPTIMIZATION
    - Notebook 11_RANDOM_FOREST_FINALMODEL_FRONTEND
- **Interpretación del modelo** y de las contribuciones de los distintos atributos a las predicciones con la ayuda de los valores SHAP (Shapley Additive Explanations).
    - Notebook 10_RANDOM_FOREST_MODEL_INTERPRETATION
    - Notebook 11_RANDOM_FOREST_FINALMODEL_FRONTEND

Más adelante, se resumen los principales resultados obtenidos en cada una de estos procesos.

ELECCIÓN Y MODELO BASE

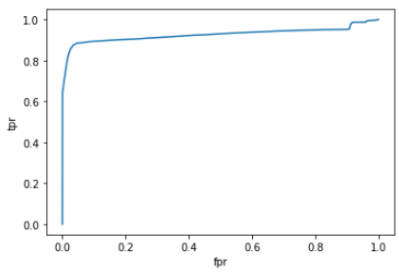
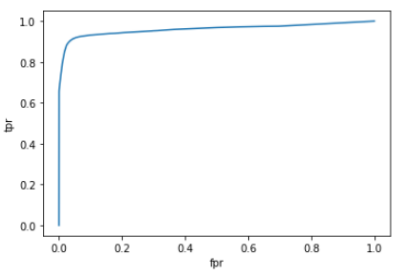
> Notebook 03_MODEL_SELECTION_AND_BASELINE.ipynb

De cara a la elección del modelo más adecuado, en un primer momento y con las variables numéricas disponibles, se realizaron diferentes pruebas con distintos algoritmos:

- Regresión Logística
- K-Nearest Neighbors
- Support Vector Machine
- Decision Tree
- Random Forest

Para la evaluación de cada uno de ellos y dado el elevado desbalanceo que existía entre la clase positiva y negativa, se tuvieron en cuenta las métricas ***Precision***, ***Recall***, ***F1-score*** y ***AUC***, además del porcentaje de aciertos (***Accuracy*** en conjunto de entrenamiento y en conjunto de test) que se utilizó para detectar posibles casos de *overfitting* y *underfitting*.

Aquellos que devolvieron los **mejores resultados** fueron la Regresión Logística y el Random Forest:

REGRESIÓN LOGÍSTICA	RANDOM FOREST
TRAINING SCORE0.9940669026764586 ACCURACY: 0.9940482959538337 PRECISION: 0.9954404346138921 RECALL: 0.6424769895435477 F1 SCORE: 0.780927736976293 AUC: 0.9264391652120458 	TRAINING SCORE: 0.9944142664507589 ACCURACY: 0.9939655903217869 PRECISION: 0.9712611607142857 RECALL: 0.6538726441675536 F1 SCORE: 0.7815739250832615 AUC: 0.9605868642082936 

En ambos casos, se observan **resultados similares** con ligeros matices. Mientras que, en la regresión, la PRECISION es prácticamente 1, se observan ciertas inconsistencias y peores resultados en el área bajo la curva (AUC). Por su parte, el modelo de Random Forest parece mejorar ligeramente la métrica RECALL, aunque a costa de PRECISION.

Ante estos resultados, se optó en un primer momento por un modelo de Regresión Logística que privilegiara la interpretabilidad de los resultados y optimizara los tiempos de entrenamiento. Sin embargo, una vez se hubo profundizado en el análisis de las variables (ausencia de relaciones lineales entre algunas variables independientes y la variable target) y se lanzaron los primeros modelos parciales (inconsistencia en los resultados), se optó por un modelo de clasificación **Random Forest**.

LAS VARIABLES: ANÁLISIS EXPLORATORIO Y 'FEATURE ENGINEERING'

> Notebooks:

- 04_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_GA-FEATURES
- 05_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_PRICE-DISCOUNT
- 06_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA
- 08_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA_PAGE-PATH-TEST'

VARIABLES NUMÉRICAS

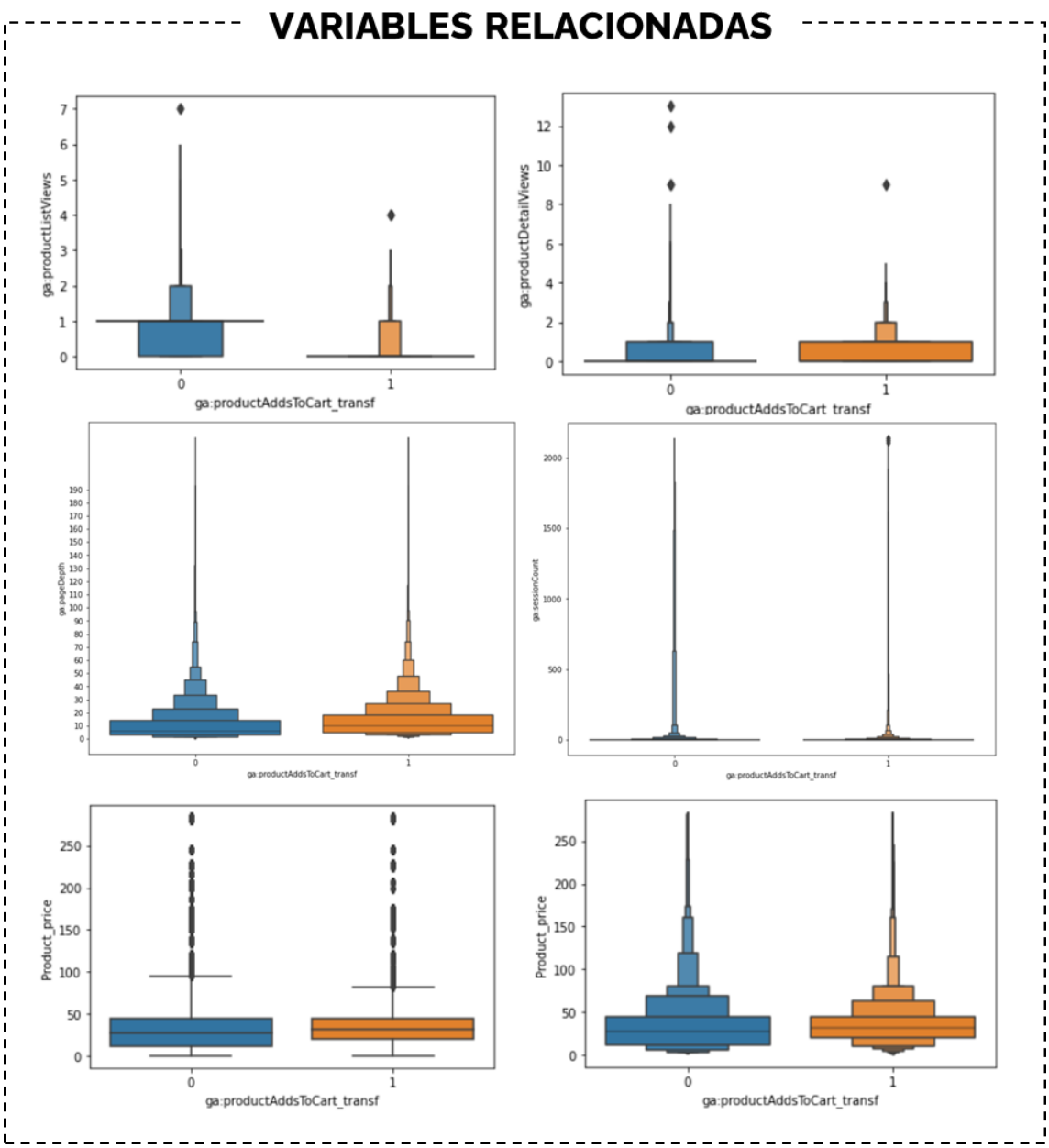
> Notebooks:

- 04_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_GA-FEATURES
- 05_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_NUMERICAL_DATA_PRICE-DISCOUNT

Al hacer el análisis exploratorio de las variables numéricas se observaron **diferencias significativas entre la clase 0 y la clase 1** para:

- `ga:productListViews`
- `ga:productDetailViews`
- `ga:pageDepth`
- `ga:sessionCount`
- `Product_price`

En todos los casos, las diferencias se hicieron evidentes al visualizar el ***boxplot*** (ver figura). Sin embargo, también se contrastaron estas hipótesis por medio de tests estadísticos. Dado que ninguna de las variables seguía una distribución Normal y las desviaciones típicas eran diferentes para las muestras de la clase 1 y clase 0 (distintas dispersiones de los datos), se optó por aplicar el ***test Mam Withney U*** que establece como hipótesis nula la igualdad de las medianas. Para todas las variables, *p-value* fue inferior a 0,05 y, por lo tanto, se pudo demostrar que las medianas eran significativamente diferentes.



En el caso de las dos primeras, se vio cómo **`ga:productListViews`** tiende a adoptar `0` cuando el producto se añade al carrito y cómo existe una **mayor probabilidad de convertir del cliente cuando se encuentra en su página específica**. Ahora bien, al comparar las conversiones en páginas tipo lista y fichas de producto, se identificó un **problema de atribución temporal**. Se estaban registrando conversiones que no tenían contabilizadas ningún tipo de visualización, ni en una lista, ni en una ficha de producto. En estos casos, se vio como la vista del producto y su conversión se estaban registrando con un minuto de diferencia y, por lo tanto, en observaciones distintas. Para paliar este problema, se optó por crear una nueva **variable *dummy*** a partir de `ga:pagePath` (`Detail_View` en el notebook 06_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA), que sustituyera a ambas y que indicara si la observación tenía lugar en una ficha de producto o no.

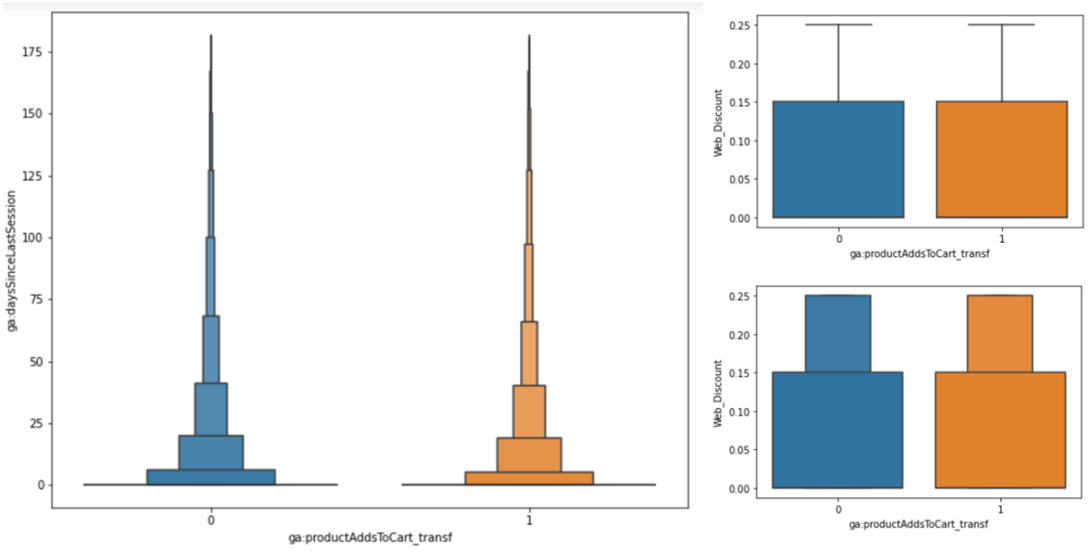
En cuanto a **`ga:pageDepth`**, **`ga:sessionCount`** y **`Product_price`**, no hubo duda a la hora de incluirlas al modelo. Dado que la idea era implementar un Random Forest y una de sus principales ventajas es que **no requiere que las variables estén estandarizadas**, no se contempló aplicar ninguna transformación de este tipo.

Únicamente, en el caso de **`Product_price`**, hubo que definir una estrategia de **imputación de NANs**, para todos aquellos productos que ya no estaban disponibles en el catálogo. Lo ideal en este caso hubiera sido calcular el precio medio (o mediana) por categoría. Es previsible que el rango de precios no sea el mismo para una crema que para unas tiritas. Ahora bien, como ya se señaló en el apartado *'Extracción de datos de Pestashop'* de este documento, la categoría disponible en el catálogo no resultaba fiable a la hora de indicar la familia a la que pertenecía el producto y su rango de precios. En este sentido, se optó finalmente por imputar **la mediana global** (en lugar de la media), para contrarrestar el sesgo que producía la larga cola que se apreciaba en el histograma y el *boxplot*.

En cuanto al resto de variables numéricas (aparentemente, no significativas para explicar la variable objetivo), estos fueron los pasos que se siguieron:

- **`ga:productListClicks`** – Dado el problema de atribución temporal descrito anteriormente y su poca explicabilidad, descartamos su incorporación al modelo.
- **`ga:daysSinceLastSession`** – A pesar de no guardar una relación aparente con la variable objetivo y dado que no exigía ningún trabajo de preprocesado, se optó por incluirla y valorar su posible influencia en el rendimiento del clasificador. Más adelante, se evaluaría su conveniencia en la selección de variables para la optimización del modelo.
- **`Web_Discount`** – Tampoco se encontraron diferencias significativas en cómo se distribuían los descuentos en la clase `1` y la clase `0`. Sin embargo, se observaron niveles de prevalencia mayores en el conjunto de observaciones que tenían descuentos superiores al 10%. En este sentido, se optó por incorporar esta variable al modelo para comprobar su efecto en el rendimiento.

VARIABLES NO RELACIONADAS



VARIABLES CATEGÓRICAS

> Notebooks:

- 06_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA
- 08_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA_PAGE-PATH-TEST
- 07_RANDOM_FOREST_MODEL_FEATURE_BY_FEATURE

En el notebook 06_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA, se exploraron las distintas etiquetas correspondientes a cada variable categórica, en términos de **interacciones totales, conversiones y prevalencia**, así como se fueron tomando decisiones de ***feature engineering*** y haciendo pruebas en este sentido.

En paralelo, en el notebook 07_RANDOM_FOREST_MODEL_FEATURE_BY_FEATURE, se fueron incorporando, una a una, las distintas variables al **modelo base** (con los atributos seleccionados en el análisis de las variables numéricas) para conocer su impacto en el rendimiento.

Además de identificar categorías con mayores probabilidades de convertir, se halló que había un porcentaje muy elevado de observaciones que tenían lugar en **páginas vinculadas al carrito o a la tramitación del pedido** y cuya prevalencia era del 100%. En este sentido, la variable `ga:pagePath` resultaba determinante en el rendimiento del modelo y su impacto en todas las métricas consideradas (*accuracy*, *precision*, *recall* y *AUC*) era muy significativo. Tras estudiar la conveniencia de incluir o no estas observaciones en el entrenamiento del modelo (ver notebook 08_DATA_EXPLORATION_AND_FEATURE_ENGINEERING_CATEGORICAL_DATA_PAGE-PATH-TEST), se decidió eliminarlas de cara a su optimización (notebook 09_RANDOM_FOREST_MODEL_OPTIMIZATION). Dado que el objetivo del proyecto era predecir el que se añadiera el producto o no al carrito y, en este caso, el usuario en cuestión ya se encontraba en él, se eligió centrar el análisis en el resto de rutas donde, desde el punto de vista de negocio, era más determinante actuar.

En términos de ***feature engineering***, en general, se optó por codificar las variables con alta cardinalidad con *TargetEncoder()* y aquellas con baja cardinalidad con *OneHotEncoder()*, en algunos casos creando nuevas categorías miscelánea que agruparan las etiquetas más residuales. En el caso de la variable temporal `ga:dateHourMinute`, se separaron sus componentes (mes, día de la semana y hora del día) y se incluyeron al modelo como el seno y el coseno de su valor para conservar su naturaleza cíclica y continua.

Para evitar que hubiera fugas de información (*data leakage*) al implementar algunas de estas transformaciones, se utilizó el ***Pipeline*** de SKLEARN (ver detalles en el notebook 07_RANDOM_FOREST_MODEL_FEATURE_BY_FEATURE).

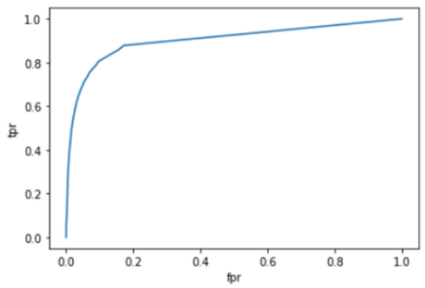
OPTIMIZACIÓN DEL MODELO

> Notebook 09_RANDOM_FOREST_MODEL_OPTIMIZATION

Como resultado de incorporar todas las variables y el conjunto de datos sin observaciones vinculadas al carrito o a la tramitación del pedido, se obtuvo el siguiente modelo base:

MODELO BASE

```
TRAINING SCORE: 0.9971983144318778
ACCURACY: 0.9875776397515528
PRECISION: 0.4017115529826328
RECALL: 0.15218842376275388
F1 SCORE: 0.22074688796680497
AUC: 0.9022823588343651
```



Para su optimización, se aplicaron técnicas de:

- Selección de variables
- *Oversampling/Undersampling*
- Optimización de hiperparámetros
- Calibración de las probabilidades

Tras consultarlo con negocio, se escogió ***Recall*** como métrica de referencia. Preferían adoptar, en un primer momento, una postura conservadora y prudente, que no les supusiera costes innecesarios. Al minimizar los falsos negativos, evitaban invertir en usuarios que hubiesen convertido de todos modos, aún asumiendo el coste de oportunidad de los falsos positivos. Si ganaban, ganaban pero no perdían.

SELECCIÓN DE VARIABLES

Se empezó por reducir el número de atributos del modelo no sólo para mejorar su rendimiento sino también para facilitar su interpretación y reducir los tiempos de entrenamiento de cara al resto de procesos de optimización.

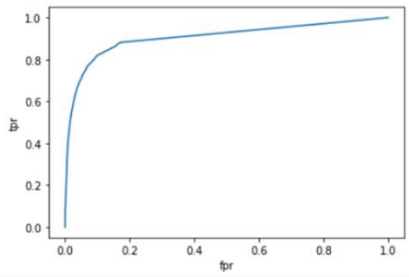
Se seleccionaron **28 de los 44** que había disponibles, en base a su importancia y contribución (superior a 0.004) a *Recall,* aplicando permutación. No sólo esta métrica se vio mejorada sino también el resto (*Precision*, evidentemente, *F1-score* y *AUC*), a excepción del porcentaje de aciertos (*Accuracy*) que bajó muy ligeramente.

VARIABLES SELECCIONADAS

Weight	Feature
0.1188 ± 0.0004	ga:pagePath
0.0740 ± 0.0037	ga:productSKU
0.0544 ± 0.0021	ga:landingPagePath
0.0469 ± 0.0032	ga:pageDepth
0.0432 ± 0.0008	Product_price
0.0406 ± 0.0022	Detail_View
0.0335 ± 0.0020	Web_Discount
0.0286 ± 0.0030	month_cos
0.0281 ± 0.0021	month_sin
0.0170 ± 0.0029	ga:city
0.0167 ± 0.0018	hour_cos
0.0154 ± 0.0010	hour_sin
0.0146 ± 0.0017	dayofweek_sin
0.0142 ± 0.0012	dayofweek_cos
0.0140 ± 0.0018	x0_google
0.0138 ± 0.0008	ga:sessionCount
0.0132 ± 0.0010	x0_Android
0.0125 ± 0.0008	x0_cpc
0.0115 ± 0.0017	x0_desktop
0.0109 ± 0.0016	x0_organic
0.0102 ± 0.0007	x0_iOS
0.0095 ± 0.0015	Returning_Visitor
0.0091 ± 0.0011	x0_mobile
0.0082 ± 0.0007	x0_Windows
0.0082 ± 0.0013	x0_(direct)
0.0068 ± 0.0007	ga:daysSinceLastSession
0.0048 ± 0.0006	x0_sendinblue
0.0045 ± 0.0010	x0_referral
0.0044 ± 0.0011	x0_email

RESULTADOS DEL MODELO RESULTANTE

```
TRAINING SCORE: 0.9971990493966123
ACCURACY: 0.9878940417209622
PRECISION: 0.4421274601686973
RECALL: 0.17993706493754172
F1 SCORE: 0.2557777024737377
AUC: 0.9057728186092732
CPU times: user 1h 24min 59s, sys: 2min 50s, total: 1h 27min 50s
Wall time: 25min 18s

Text(0, 0.5, 'tpr')
```



| NOTA: En el modelo final (notebook 11_RANDOM_FOREST_FINALMODEL_FRONTEND), se hicieron distintas pruebas para seleccionar el número óptimo de atributos y finalmente se seleccionaron 9.

MANEJO DE CLASES DESBALANCEADAS

En el conjunto de datos, las **clases negativa y positiva estaban claramente desbalanceadas** (98,85% vs 1,14%, respectivamente), especialmente después de eliminar las observaciones vinculadas al carrito o a la tramitación del pedido. Dado que esto podía estar ocasionando que el modelo tuviera **dificultades para aprender de la clase minoritaria**, se decidió probar a crear observaciones sintéticas a partir de distintas técnicas de *oversampling*.

Las técnicas que se evaluaron fueron:

- **SMOTE** - Sintetiza observaciones de la clase minoritaria (al margen de la mayoritaria) escogiendo puntos próximos entre sí en el espacio muestral.
- **SMOTE + Random Undersampling** - En la literatura, se suele recomendar el uso de SMOTE en combinación con técnicas de *undersampling* que reduzcan el peso de la clase mayoritaria.
- **Bordeline SMOTE** - Una extensión de SMOTE que selecciona únicamente las observaciones que se solapan con la clase mayoritaria para sintetizar nuevas e ignorar aquellas que se encuentran alejadas de la zona de riesgo.
- **ADASYN** - Este enfoque crea muestras sintéticas en aquellas regiones donde la densidad de los puntos de la clase minoritaria es baja, es decir, aquellas observaciones que al modelo le pueden resultan más difíciles de aprender.

Tras lanzar distintos modelos, entrenados con conjuntos de datos balanceados con cada una de estas técnicas, se eligió **SMOTE + Random Undersampling** por ser la metodología que obtenía los mejores resultados para *Recall* e, incluso, para *AUC*.

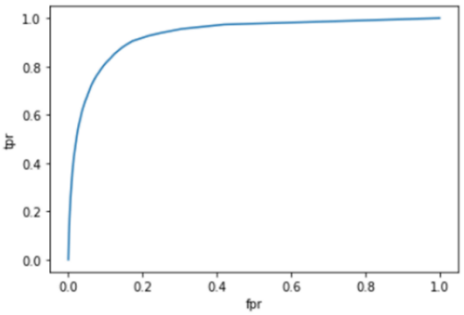
Una vez elegida, se optimizaron los parámetros *sampling_strategy* y *k* de SMOTE. La combinación ganadora en términos de Recall resultó ser:

- ***sampling_strategy*=0.1** (la clase minoritaria debía representar el 10% de la clase mayoritaria)
- ***k*=4** (número de vecinos que debía considerar SMOTE para crear las observaciones sintéticas)

A continuación, se muestran los resultados del **modelo entrenado con los datos balanceados** de manera óptima:

MODELO CON CLASES BALANCEADAS

```
TRAINING SCORE: 0.987114230791421
ACCURACY: 0.9780304583749506
PRECISION: 0.24595016414617082
RECALL: 0.43577762944598075
F1 SCORE: 0.3144351176551535
AUC: 0.9316110517577915
CPU times: user 25min 2s, sys: 15.9 s, total: 25min 17s
Wall time: 7min 4s
```



OPTIMIZACIÓN DE HIPERPARÁMETROS DE RANDOM FOREST

Dada la controversia existente en el uso de técnicas de desbalanceo, se planteó hacer la optimización de los hiperparámetros del modelo para dos escenarios distintos:

- Sin aplicar técnicas de desbalanceo
- Aplicando técnicas de desbalanceo (SMOTE/Undersampling optimizados de acuerdo a los valores obtenidos en el apartado anterior)

En ambos casos, se probaron los siguientes valores con la ayuda de ***GridSearchCV*** de SKLEARN:

- **n_estimators** (número de árboles) = [50 , 100 , 200]
- **max_depth** (profundidad máxima de cada árbol) = [None , 5 , 10 , 20]
- **criterion** (criterio para medir la pureza ponderada de los nodos) – ['*gini*' , '*entropy* ']

Una vez entrenados ambos modelos con sus respectivos valores óptimos, se evaluó la conveniencia de usar técnicas de desbalanceo no sólo con el conjunto de test que habíamos separado del *dataset* principal, sino también con un nuevo conjunto de datos descargado de Google Analytics (febrero y marzo 2021).

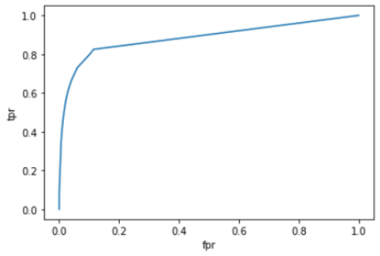
Como resultado, se obtuvo que el mejor modelo en términos de rendimiento de ***Recall*** seguía siendo aquel que había sido entrenado con el conjunto de datos balanceado y para unos valores óptimos de:

- ***n_estimators*=200**
- ***max_depth*=20**
- ***criterion*='gini'** (por defecto, en Random Forest de SKLEARN)

Cabe señalar que, ante la imposibilidad de ejecutar dicho código en el ordenador, se creó una instancia de *Jupyter Notebook* con 16 CPUs y 60 GB de RAM en *Google Cloud Console*.

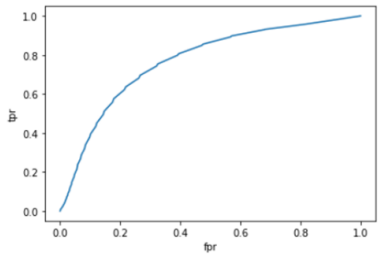
MODELO OPTIMIZADO CON CLASES DESBALANCEADAS

```
TRAINING SCORE: 0.7948759110026647
ACCURACY: 0.9881145308982509
PRECISION: 0.46793193717277487
RECALL: 0.20453895298941546
F1 SCORE: 0.2846526441510185
AUC: 0.883979405845898

Text(0, 0.5, 'tpr')
```



EN X_TEST

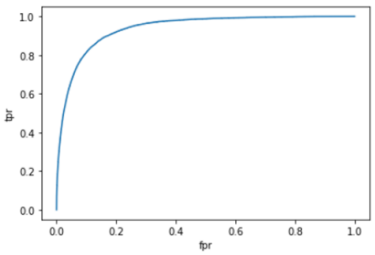
```
ACCURACY: 0.9835602206047315
PRECISION: 0.028170220053086736
RECALL: 0.02007934086054318
F1 SCORE: 0.023446408209806154
AUC: 0.7680179488383737

Text(0, 0.5, 'tpr')
```



EN FEBRERO Y MARZO 2021

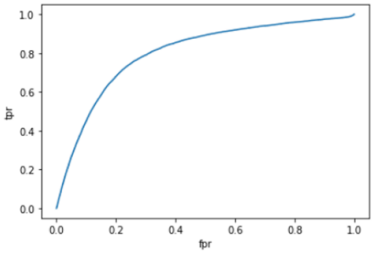
MODELO OPTIMIZADO CON CLASES BALANCEADAS

```
TRAINING SCORE: 0.9760786894021345
ACCURACY: 0.9693497994650933
PRECISION: 0.19233092860442802
RECALL: 0.5160675121579098
F1 SCORE: 0.2802257546730182
AUC: 0.9375789899166102

Text(0, 0.5, 'tpr')
```



EN X_TEST

```
ACCURACY: 0.8940076326261777
PRECISION: 0.04256217684590186
RECALL: 0.45517241379310347
F1 SCORE: 0.0778452176545188
AUC: 0.8009227549278027

Text(0, 0.5, 'tpr')
```



EN FEBRERO Y MARZO 2021

CALIBRACIÓN DEL MODELO

Uno de los inconvenientes de Random Forest es que no predice realmente la probabilidad de cada clase, sino que devuelve más bien una puntuación (estimada a partir de todos los árboles) sin interpretación estadística en este sentido. Esta es válida para estimar el *AUC* pero no para negocio. El modelo sabe lo que sabe pero desde su visión limitada.

Para poder extrapolar sus resultados al mundo real había que calibrar el modelo, ajustando las probabilidades predichas para que correspondieran con la proporción de casos reales.

Tras evaluarlo, el modelo de referencia hasta el momento hacia predicciones demasiado optimistas. Para calibrarlo, se utilizó la clase ***CalibratedClassifierCV*** de SKLEARN y se valoraron los 2 métodos disponibles para hacer las correcciones:

- ***'sigmoid'*** que utiliza una Regresión Logística para transformar las probabilidades
- ***'isotonic'*** que utiliza un modelo de Regresión Isotónica.

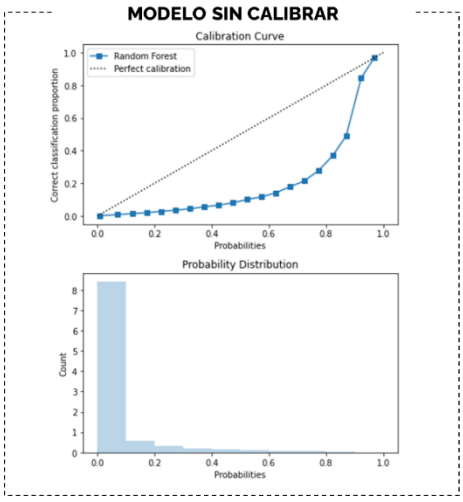
Asimismo, con el fin de evitar el *overfitting,* se utilizaron distintos datos de los de entrenamiento. En este sentido, se siguieron 2 estrategias distintas:

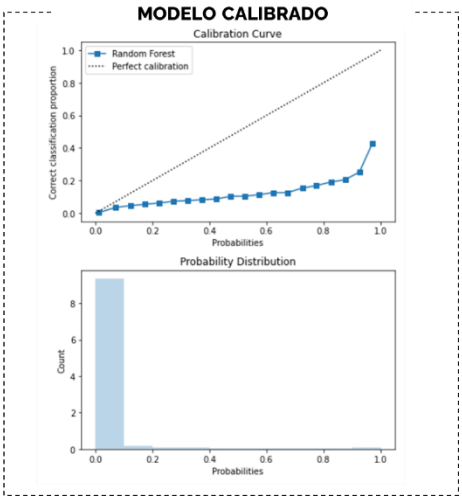
- Entrenar la calibración con el **conjunto de test** y evaluar los resultados con los datos de febrero y marzo 2021.
- Entrenar la calibración con el **conjunto de entrenamiento aplicando técnicas de validación cruzada**.

Una vez se obtuvieron los resultados para cada una de estas combinaciones, se conservó el modelo calibrado con el conjunto de entrenamiento al que se le habían aplicado técnicas de validación cruzada con los siguientes parámetros optimizados con respecto a ***brier score***:

- ***cv*=3**
- ***method*='sigmoid'**

Si bien este clasificador no estaba realmente calibrado y seguía devolviendo predicciones demasiado optimistas, a diferencia del resto, conservaba la **capacidad predictiva** y reducía ligeramente el valor de ***brier score* (de 0,23 a 0,19)** con respecto al modelo de referencia.

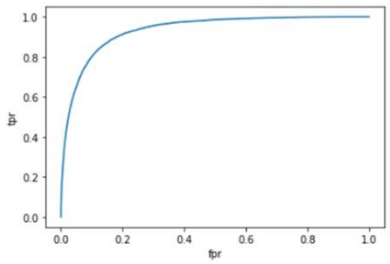




RESULTADOS DEL MODELO CALIBRADO SOBRE EL CONJUNTO DE TEST

```
BRIER SCORE=  0.01995399592216733

              precision    recall  f1-score   support

           0       0.99      0.98      0.99    896587
           1       0.21      0.46      0.29     10487

    accuracy                           0.97    907074
   macro avg       0.60      0.72      0.64    907074
weighted avg       0.98      0.97      0.98    907074


array([[878434,  18153],
       [  5619,   4868]])
```

```
ACCURACY: 0.9737926563874613
PRECISION: 0.2114591025585335
RECALL: 0.4641937637074473
F1 SCORE: 0.29055747881103017
AUC: 0.9331499765533461

Text(0, 0.5, 'tpr')
```

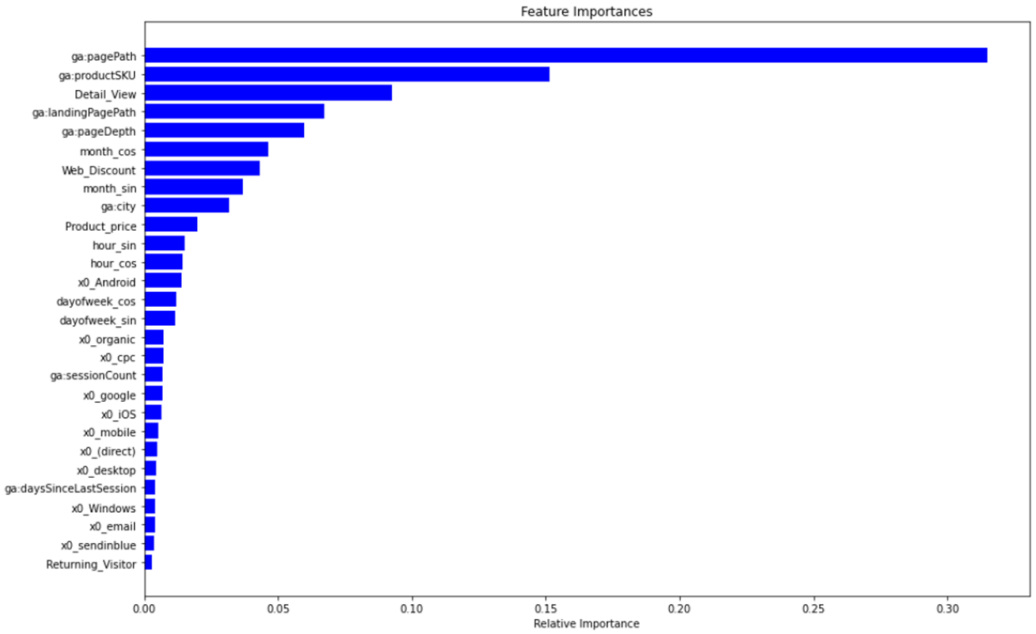


INTERPRETACIÓN DEL MODELO

> Notebook 10_RANDOM_FOREST_MODEL_INTERPRETATION

Al atender a la importancia de los distintos atributos, en base a la pureza de los nodos del Random Forest, se obtuvo que los *features* más relevantes para el modelo eran **el producto** y **la localización** (`ga:pagePath` y `Detail_View`) en la que tenía lugar la observación, seguidos de **la página en la que aterrizaban** los usuarios (`ga:landingpagepath`), **el número de páginas vistas** (`ga:pageDepth`), el **mes** (`month_cos` y `month_sin`) y **el descuento disponible** en la web (`Web_Discount`).

IMPORTANCIA DE LOS ATRIBUTOS EN BASE A LA PUREZA DE LOS NODOS



Con el fin de **entender las predicciones del modelo** y, aún sin establecer relaciones causales, comprender los predictores que en mayor medida contribuyen a la conversión del usuario, se empleó el módulo SHAP. La idea original era explicar las predicciones del modelo final calibrado, sin embargo, esta librería no soporta modelos del tipo *CalibratedClassifierCV* o *GridSearchCV*. De modo que se optó por recrear de nuevo el modelo Random Forest con los parámetros resultantes de la optimización (empleando el mismo conjunto de datos y preprocesado).

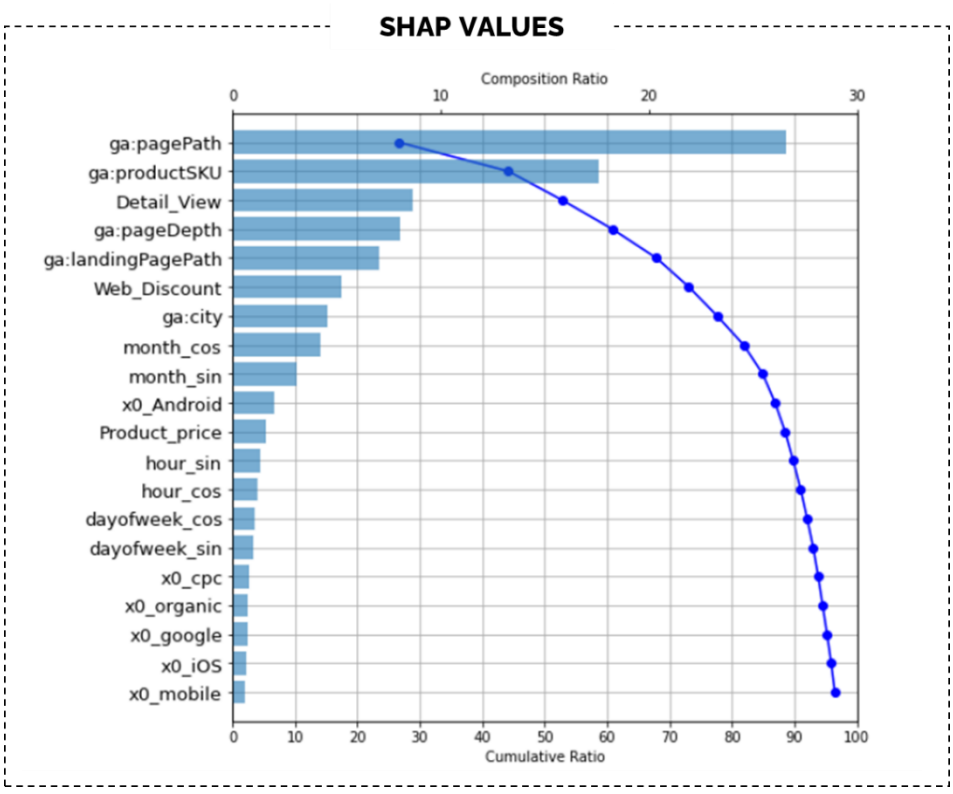
INTERPRETABILIDAD GLOBAL

En una primera **aproximación global** al modelo, empezamos por calcular los *shap values* con los datos de entrenamiento. Los resultados fueron bastante similares (especialmente en el TOP 3) a los obtenidos con el método predeterminado de Random Forest, aunque con ligeras diferencias:

- **`ga:pageDepth`** adquirió predominancia frente a **`ga:landingPagePath`**
- **`Web_discount`** y **`ga:city`** escalaron puestos y se colocaron a la cola del TOP 5.
- **`xo_Android`** se coló en el TOP 10 mientras que **`Product_price`** salió.
- **`ga:sessionCount`** dejó de estar entre los 20 predictores más importantes.

En concreto, se extrajeron las siguientes conclusiones:

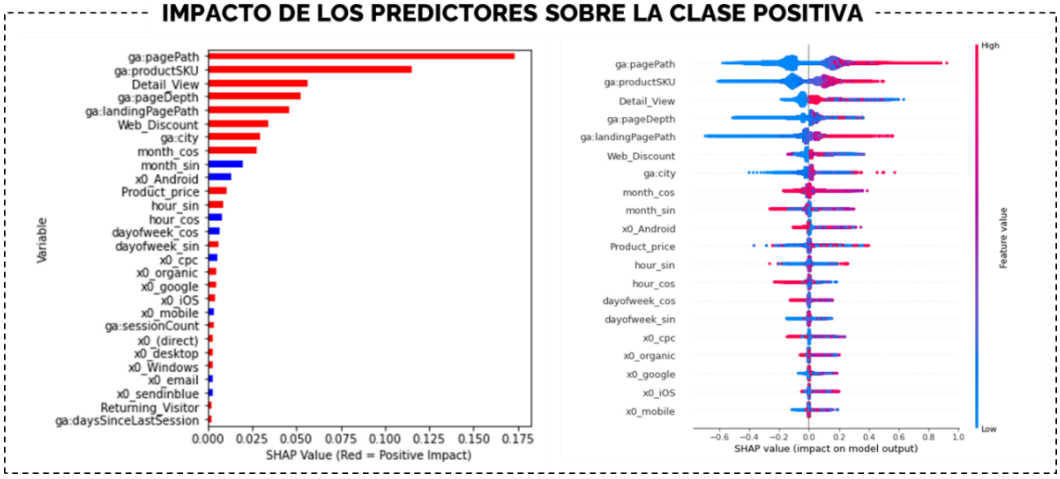
- La URL en la que tiene lugar la observación (**`ga:pagePath`**) es el atributo más importante y tiene más de un 25% de explicabilidad.
- Unida a **`ga:productSKU`** y a **`Detail_View`** se explica más de la mitad del modelo.
- En conjunto, estos 20 predictores explican prácticamente la totalidad del modelo y sólo el TOP 10 ya cubre el 90% de explicabilidad.



Al atender al detalle de su **impacto en la clase positiva**, se obtuvo que todos los *inputs* seguían una correlación positiva con la variable target, salvo:

- `month_sin`
- `x0_Android`
- `hour_cos`
- `day_of_week_cos`
- `x0_cpc`
- `x0_mobile`
- `x0_email`
- `x0_sendinblue`

En especial, **`ga:pagePath`**, **`ga:productSKU`** y **`ga:landingPagePath`** impactaban significativamente y de forma positiva sobre `ga:productAddsToCart_transf`, al adoptar valores muy elevados. Al contrario, valores bajos en estas variables influían muy negativamente.



También se estimó el **efecto marginal** de cada unas de las variables independientes sobre las predicciones de la clase 1, así como sus principales interacciones con otros atributos. Puede verse el detalle en el notebook (10_RANDOM_FOREST_MODEL_INTERPRETATION).

INTERPRETABILIDAD LOCAL

Por último, se analizó cómo se comportaba el modelo con las observaciones del conjunto de test, diferenciando las categorías en el interior de la matriz de confusión (Verdaderos Positivos, Falsos Negativos, Falsos Positivos y Verdaderos Negativos) y concluyendo lo siguiente:

El modelo tiende a estimar como Clase 1 las observaciones con:

- **`ga:pagePath`** > 0.036023 (páginas con alta probabilidad de convertir en el conjunto de entrenamiento)
- **`ga:productSKU`** > 0.017587 (productos con alta probabilidad de convertir en el conjunto de entrenamiento)
- **`Detail_View`** = 1 (productos vistos en su página específica)
- **`ga:landingPagePath`** > 0.015280 (páginas de llegada a la web con alta probabilidad de convertir en el conjunto de entrenamiento)

El modelo tiende a estimar como Clase 0 las observaciones con:

- **`ga:pagePath`** < 0.036023 y/o **`ga:productSKU`** < 0.017587 (páginas y productos con baja probabilidad de convertir en el conjunto de entrenamiento)
- **`Detail_View`** = 0 (productos vistos en páginas tipo lista)
- **`ga:pageDepth`** < 3 (sesiones con pocas visitas)
- **`ga:landingPagePath`** < 0.015280 (páginas de llegada a la web con baja probabilidad de convertir en el conjunto de entrenamiento)

Otros de los principales atributos que tienden a penalizar la conversión según el modelo son:

- **`Web_Discount`** = 0 (descuento no disponible)
- **`ga:city`** < 0.011734 (ciudades con baja probabilidad de convertir en el conjunto de entrenamiento)
- **`x0_Android`** = 1 - Dispositivos Android

MODELO FINAL

> Notebook 11_RANDOM_FOREST_FINALMODEL_FRONTEND

En base a las conclusiones extraídas en las etapas anteriores, se formuló un último modelo final para presentar, a través de un *frontend* interactivo, las **posibilidades del uso de metodologías de Machine Learning** en la optimización del ratio de conversión de un ecommerce.

Para su diseño, se siguieron las mismas pautas que en el notebook 09_RANDOM_FOREST_MODEL_OPTIMIZATION pero con ligeras modificaciones:

- Uso del precio final como *feature* (PVP - Descuento)
- Supresión de la variable `ga:pageDepth`
- Menor número de predictores

En primer lugar, optamos por no considerar el PVP y el Descuento por separado, sino crear una variable combinada que considerara el **precio final a pagar por el usuario**. Conservamos igualmente la columna **`Web_Discount`** por haber demostrado ser un atributo importante en modelos anteriores y por el efecto psicológico que pudiera tener en el usuario.

Por otro lado, se eliminó la variable `ga:pageDepth`. En modelos anteriores, se asumió que era el número de páginas vistas por el usuario hasta el momento de la observación. Sin embargo, se descubrió que en realidad correspondía al **número total de páginas vistas en el transcurso de la sesión**. Resultaba útil para identificar al usuario único/sesión, pero no debía ser incluida en el modelo final, por no ser una variable que fuera a estar disponible si se quisiera hacer más tarde **un análisis en tiempo real**.

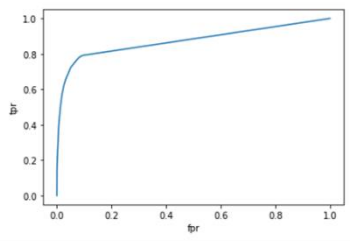
Por último, a la hora de hacer la selección de los predictores más importantes, se realizaron distintas pruebas. El mejor resultado para ***Recall*** (0,2713) se obtuvo con los **9 atributos más importantes**. Tras optimizar los hiperparámetros de Random Forest para este modelo se alcanzó 0,2736.

TOP 9 ATRIBUTOS Y SUS CONTRIBUCIONES A RECALL (POR PERMUTACIÓN)

Weight	Feature
0.1195 ± 0.0026	ga:pagePath
0.0795 ± 0.0019	ga:productSKU
0.0593 ± 0.0015	ga:landingPagePath
0.0439 ± 0.0027	Product_price
0.0417 ± 0.0027	Detail_View
0.0343 ± 0.0018	month_sin
0.0338 ± 0.0026	Web_Discount
0.0259 ± 0.0006	month_cos
0.0188 ± 0.0019	ga:city

MODELO OPTIMIZADO CON TOP9 ATRIBUTOS

```
TRAINING SCORE: 0.626545092625293
ACCURACY: 0.9887252859193406
PRECISION: 0.5237312887915297
RECALL: 0.2735768093830457
F1 SCORE: 0.35941121202630755
AUC: 0.8734769672012491
Text(0, 0.5, 'tpr')
```

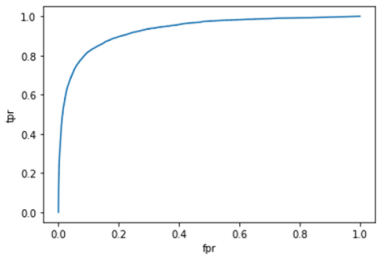


Cabe destacar que estos resultados ya eran significativamente mejores a los obtenidos en esta misma etapa en el notebook de optimización, donde ***Recall*** ascendía a 0,1799.

También se realizaron las mismas pruebas para el desbalanceo de las clases. Tras su evaluación, se optó por el uso de **SMOTE** (*sampling_strategy=0.1* y *k=3*) y **Random Undersampling**, resultando en un clasificador para el que se habían optimizado los parámetros de **Random Forest** (*criterion='gini', max_depth=20, n_estimators=200*) y para el que se obtuvo un **Recall de 0,70 en el conjunto de test y 0,52 en los datos de febrero y marzo 2021** (este mismo modelo en el notebook de optimización obtenía 0,52 y 0,45, respectivamente).

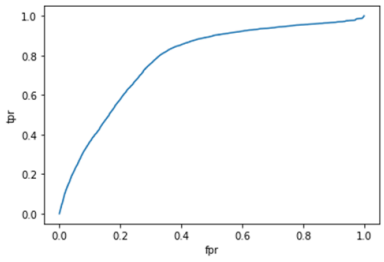
MODELO FINAL OPTIMIZADO CON CLASES BALANCEADAS

```
TRAINING SCORE: 0.9752794536945135
ACCURACY: 0.9511208567327473
PRECISION: 0.1520496690103203
RECALL: 0.7052541241537141
F1 SCORE: 0.2501648937069797
AUC: 0.9309970386050841

Text(0, 0.5, 'tpr')
```



EN X_TEST

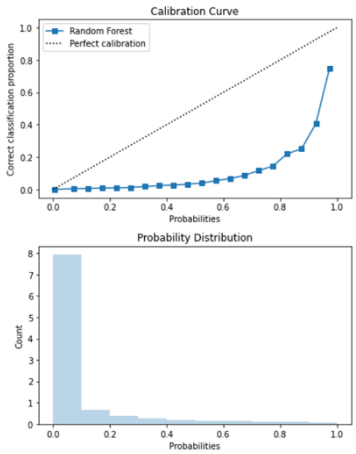
```
ACCURACY: 0.8248311092502103
PRECISION: 0.029136927844420832
RECALL: 0.5204760451632591
F1 SCORE: 0.055184552466739145
AUC: 0.777935795164294

Text(0, 0.5, 'tpr')
```



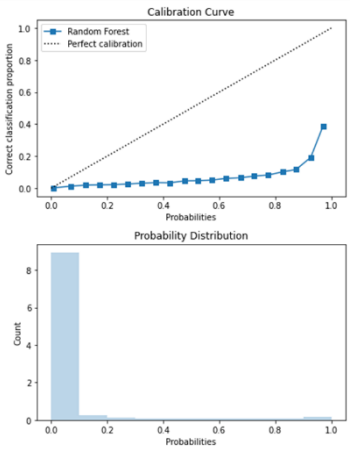
EN FEBRERO Y MARZO 2021

A pesar de estos mejores resultados, las **predicciones volvían a ser excesivamente optimistas** y, aunque se calibró el modelo, seguimos sin poder ajustar correctamente las probabilidades. En cualquier caso, como en el notebook de optimización, se conservó el modelo calibrado por haber reducido ligeramente el ***brier score*** de 0,036 a 0,032.

MODELO FINAL SIN CALIBRAR



MODELO FINAL CALIBRADO

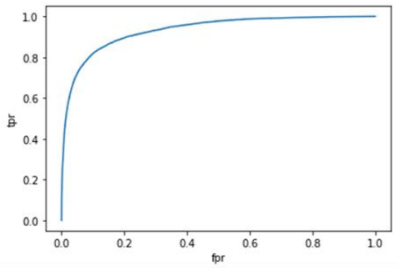


El modelo final calibrado obtenía un **Recall de 0,69** frente al 0,46 que se había obtenido en el notebook de optimización.

RESULTADOS DEL MODELO FINAL CALIBRADO SOBRE EL CONJUNTO DE TEST

```
BRIER SCORE=  0.03258619991898625

              precision    recall  f1-score   support

           0       1.00      0.96      0.98    896587
           1       0.17      0.69      0.27     10487

    accuracy                           0.96    907074
   macro avg       0.58      0.82      0.62    907074
weighted avg       0.99      0.96      0.97    907074

array([[860348,  36239],
       [  3283,   7204]])
```

```
ACCURACY: 0.9564291336759735
PRECISION: 0.16582648527956173
RECALL: 0.6869457423476686
F1 SCORE: 0.2671611348043761
AUC: 0.9322565339615351

Text(0, 0.5, 'tpr')
```



INTERACTIVE FRONTEND

> Notebook 11_RANDOM_FOREST_FINALMODEL_FRONTEND

Como último paso del proyecto, se desarrolló un *frontend* interactivo para ilustrar la aplicación práctica del modelo final y demostrar **cómo podemos influir en las conversiones**, aplicando ligeros cambios en los predictores. En particular, se esperaba mostrar cómo la implementación de Real Time Reporting API en combinación con nuestro clasificador, permitiría hacer descuentos específicos a los indecisos en el momento y lugar adecuados, resultando en un mayor número de conversiones.

La app, diseñada a partir del módulo **Streamlit**, se compone de 3 páginas:

- **Home** donde se introduce brevemente el proyecto
- **Ejemplo de maximización de las conversiones** mediante la aplicación de un descuento específico y adicional dirigido a los indecisos (predicciones de la clase 0) en el momento y lugar adecuados. Para unas predicciones (relativamente) más realistas, se utilizó el modelo calibrado.
- **¿Cómo funciona?** Mediante la cumplimentación de un formulario, el usuario puede ver el detalle de las predicciones del modelo para distintos niveles de descuento. En este caso, se utilizaron tanto el modelo preentrenado sin calibrar (para mostrar los resultados de shap), como el modelo calibrado para unos resultados más ajustados y menos optimistas.

El fichero .py (**interactive_frontend.py**) está disponible en la página principal del repositorio. Para jugar con la aplicación, basta con ejecutar la última celda del notebook Notebook 11_RANDOM_FOREST_FINALMODEL_FRONTEND (**!streamlit run interactive_frontend.py**). Si lo prefiere también puede sobrescribir el fichero.py ejecutando la penúltima celda. Si tiene algún problema confirme el directorio raíz de los datos (**data_root**) al inicio del código fuente, tras las librerías.

En la segunda página (‘**Maximizar las conversiones**`), se le pedirá que cargue un fichero. Este campo está pensado para subir un informe de Google Analytics (con los predictores finales) al que se le hayan

añadido la información de Precios y Descuentos. Para facilitar su prueba, se creó un fichero con 1.000 observaciones de febrero y marzo 2021 (**prueba_streamlit.csv**), alojado en el directorio de *data* que acompaña a este proyecto. También puede ser creado directamente desde el apartado *'CSV de prueba'* del notebook.

En la tercera página ('¿Cómo funciona?') basta con completar el formulario para obtener las predicciones. Para los campos que implican introducir una URL, copie y pegue cualquier página de https://www.galileo61.com/es/.

CONCLUSIONES E IMPLICACIONES

En general, podemos decir que se han cumplido las expectativas del proyecto, enunciadas al inicio de este documento. Se ha logrado crear un **marco de trabajo inicial** para la implementación de un modelo de Machine Learning en el desarrollo de la estrategia de CRO del *e-commerce* en cuestión. Tal y como se demuestra en el *frontend* interactivo, desde el punto de vista de su aplicación práctica y tras aplicar ciertas correcciones que se enumeraran a continuación, el clasificador podría ser utilizado para **aplicar incentivos en tiempo real** entre los indecisos e incrementar el número de conversiones.

Ahora bien, los resultados todavía están lejos de ser **fiables y óptimos**. Existen una serie de áreas que deben seguir estudiándose y mejorándose.

En primer lugar, conviene configurar una *CUSTOM DIMENSION* de GA4 con la que **identificar al usuario** y poder **eliminar el problema de atribución temporal** de las observaciones.

Por otro lado, como se vio en el modelo final al modificar los términos de la variable precio, más trabajo en el área de ***feature engineering*** puede ayudar a mejorar el rendimiento del modelo, especialmente en variables tan determinantes como **`ga:pagePath`** y **`ga:landingPagePath`**. No sólo deberían valorarse otras técnicas de codificación, sino que también habría que hacer un mayor esfuerzo por limpiar sus rutas. Existen muchas de ellas duplicadas, anidadas bajo distintas páginas, como consecuencia de los cambios que se han aplicado a la arquitectura web en el transcurso del tiempo.

Asimismo, igual que se creó la variable `Detail_View` para identificar si el usuario se encontraba en la ficha del producto y dada la prevalencia de este tipo de páginas, se podría incorporar una variable que considerara si el usuario está realizando una **búsqueda en la web**.

Convendría incorporar además nuevas variables que no sólo ayudaran a **reducir el error estocástico** del modelo sino también, por ejemplo, como en el caso de la categoría del producto, ayudarán a optimizar la imputación de *NANs*. En lo que respecta, a nuevos atributos parece ser que factores meteorológicos pueden ser determinantes.

Por otra parte, como hemos visto, la actividad del *e-commerce* ha experimentado numerosos cambios en los últimos tiempos y algunos meses o no están suficientemente representados o lo están excesivamente, como consecuencia de eventos externos excepcionales. En este orden de ideas, el modelo no debería perder tanto rendimiento (con respecto a test) al ser aplicado a los nuevos datos de

Febrero y Marzo 2021. Asimismo, se debería trazar una **estrategia de actualización** regular del modelo para capturar los posibles cambios de tendencia que se puedan producir en el tiempo.

Otro ámbito que debe ser estudiado es el uso de una métrica más adecuada a los objetivos de negocio. Tal y como se ha visto, el uso de ***Recall*** como métrica de referencia ha conducido a **resultados excesivamente optimistas**.

Del mismo modo, aunque en este proyecto se han considerado diferentes algoritmos, deberían probarse otros tipos de modelos que puedan ajustarse mejor como, por ejemplo, **modelos de redes neuronales**.

PRINCIPALES REFERENCIAS UTILIZADAS

Google Analytics 4: Reporting API v4
https://developers.google.com/analytics/devguides/reporting/core/v4

Google Analytics 4: Core Reporting API
https://developers.google.com/analytics/devguides/reporting/core/v3/quickstart/service-py

Google Analytics 4: Analytics Reporting API
https://developers.google.com/analytics/devguides/reporting/core/v4/rest

Google Analytics: Dimensions and Metrics Explorer
https://ga-dev-tools.appspot.com/dimensions-metrics-explorer/

Google Analytics: Query explorer
https://ga-dev-tools.appspot.com/query-explorer/

Vieira, Armando. (2015). Predicting online user behaviour using deep learning algorithms.
https://www.researchgate.net/publication/284219029_Predicting_online_user_behaviour_using_deep_learning_algorithms

Random Forest con Python by Joaquín Amat Rodrigo, available under a Attribution 4.0 International (CC BY 4.0) at https://www.cienciadedatos.net/documentos/py08_random_forest_python.html

Regresión logística simple y múltiple by Joaquín Amat Rodrigo, available under a Attribution 4.0 International (CC BY 4.0) at
https://www.cienciadedatos.net/documentos/27_regresion_logistica_simple_y_multiple.html

Categorical data encoding techniques
https://www.analyticsvidhya.com/blog/2020/08/types-of-categorical-data-encoding/

Machine Learning with Datetime Feature Engineering
https://towardsdatascience.com/machine-learning-with-datetime-feature-engineering-predicting-healthcare-appointment-no-shows-5e4ca3a85f96

Encoding cyclical continuous features – 24 hour time
https://ianlondon.github.io/blog/encoding-cyclical-features-24hour-time/

Feature engineering: handling cyclical features
http://blog.davidkaleko.com/feature-engineering-cyclical-features.html#:~:text=Hours%20of%20the%20day%2C%20days,each%20other%20and%20not%20far.

Automate Machine Learning Workflows with Pipelines in Python and scikit-learn
https://machinelearningmastery.com/automate-machine-learning-workflows-pipelines-python-scikit-learn/

Pipelines & Custom Transformers in scikit-learn: The step-by-step guide (with Python code)

https://towardsdatascience.com/pipelines-custom-transformers-in-scikit-learn-the-step-by-step-guide-with-python-code-4a7d9b068156

Scikit-learn Pipelines: Custom Transformers and Pandas integration
https://queirozf.com/entries/scikit-learn-pipelines-custom-pipelines-and-pandas-integration

How to Choose a Feature Selection Method For Machine Learning
https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/

How to Perform Feature Selection with Categorical Data
https://machinelearningmastery.com/feature-selection-with-categorical-data/

How to Calculate Feature Importance With Python
https://machinelearningmastery.com/calculate-feature-importance-with-python/

Explaining Feature Importance by example of a Random Forest
https://towardsdatascience.com/explaining-feature-importance-by-example-of-a-random-forest-d9166011959e

How to Use scikit-learn 'eli5' Library to Compute Permutation Importance?
https://pub.towardsai.net/how-to-use-scikit-learn-eli5-library-to-compute-permutation-importance-9af131ece387

SMOTE for Imbalanced Classification with Python
https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

Deriving true probability from an ensemble classifier
https://medium.com/bcggamma/deriving-true-probability-from-an-ensemble-classifier-4417f7a67ac4

Calibrar probabilidades en modelos de machine learning by Joaquín Amat Rodrigo, available under a Attribution 4.0 International (CC BY 4.0) at https://www.cienciadedatos.net/documentos/py11-calibrar-modelos-machine-learning.html

How to Calibrate Probabilities for Imbalanced Classification
https://machinelearningmastery.com/probability-calibration-for-imbalanced-classification/

Explain Your Model with the SHAP Values
https://towardsdatascience.com/explain-your-model-with-the-shap-values-bc36aac4de3d

Explaining Random Forest Model With Shapely Values
https://www.kaggle.com/vikumsw/explaining-random-forest-model-with-shapely-values

How to understand your customers and interpret a black box model
https://aigerimshopenova.medium.com/random-forest-classifier-and-shap-how-to-understand-your-customers-and-interpret-a-black-box-model-6166d86820d9