Universitat Oberta de Catalunya (UOC)

Máster Universitario en Ciencia de Datos (*Data Science*)

TRABAJO FINAL DE MÁSTER

Área: *Machine Learning*

Interpretación de modelos de predicción del *Churn*.

Quién y porqué abandona una compañía de servicios.

Autor: Susana Vila Melero

Tutor: Jordi Escayola Montilla

Profesor: Antonio Lozano Bagén

Barcelona, 14 de enero de 2023

Créditos/Copyright

FICHA DEL TRABAJO FINAL

Título del trabajo:	Interpretación de modelos de predicción del *Churn*
Nombre del autor:	Susana Vila Melero
Nombre del colaborador/a docente:	Jordi Escayola Montilla
Nombre del PRA:	Antonio Lozano Bagen
Fecha de entrega (mm/aaaa):	01/2023
Titulación o programa:	Máster de Ciencia de Datos
Área del Trabajo Final:	Machine Learning
Idioma del trabajo:	Castellano
Palabras clave	*machine learning*, *churn*, interpretabilidad de modelos

Dedicatoria

A mis cuatro puntos cardinales, Héctor, Miquel, Ricard y Sara.

El tiempo tiene dos caras, se dice Jayyám, dos dimensiones;
la longitud va al ritmo del sol, la densidad al ritmo de las pasiones.
Amin Maalouf, Samarcanda.

Abstract

Churn is one of the major problems facing service companies today. We invest in customer loyalty, but the customer keeps changing companies with considerable volatility. Understanding what leads the customer to abandon us for another offer is fundamental to maintain a fixed portfolio. In this paper we apply advanced methods of predictive analytics on a data set of customers of a telecommunications company, with the intention of deepening the analysis of why and who is likely to increase the *churn* and to avoid it.

Keywords: churn, deep-learning, predictive analysis, machine learning, Data Science Master.

Resumen

El *churn* es uno de los grandes problemas que presentan actualmente las empresas de servicios. Se invierte en la fidelización del cliente, pero este sigue cambiando de compañía con notable volatilidad. El entender qué es lo que lleva al cliente a abandonarnos por otra oferta es fundamental para evitarlo y mantener una cartera fija. En este trabajo aplicamos métodos avanzados de análisis predictivo sobre un conjunto de datos de clientes de una empresa de telecomunicaciones, con la intención de profundizar en el análisis de quién es susceptible de incrementar el *churn* y por qué para así poder evitarlo.

Palabras clave: *churn*, *deep-learning*, análisis predictivo, *machine learning*, trabajo final de Máster, Máster Universitario de Ciencia de Datos.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Descripción de la propuesta

Las empresas de servicios basan la calidad de su desempeño en diferentes métricas, siendo una de las más relevantes el *churn rate* o tasa de abandono de los clientes. Esta métrica hace referencia al porcentaje de clientes que deja la compañía en un peridodo determinado de tiempo, lo que permite evaluar la calidad del servicio percibida y la capacidad de retención de los clientes que tiene la compañía.

La fórmula de cálculo del *churn* se muestra a continuación:

$$\text{churn rate} = \frac{\text{clientes perdidos durante el periodo}}{\text{clientes a principio de periodo}} * 100 \tag{1.1}$$

Las razones que llevan a un cliente a abandonar la compañía son múltiples y no siempre hay un único factor implicado a la hora de tomar la decisión. El precio del servicio, las expectativas generadas en la venta, la experiencia de usuario o la atención recibida cuando hay un problema son algunos de los factores que más impactan en el aumento del *churn rate*.

Hay muchas formas de analizar y estudiar los factores comentados, pero esta propuesta se basa en aplicar métodos avanzados de *deep learning* para profundizar en los motivos que llevan a un cliente a abandonar la compañía. Para ello, se utilizará un conjunto de datos obtenido aquí, en el que hay información sobre el churn de clientes de una compañía de servicios de telecomunicaciones. Se prepararán esos datos con técnicas de muestreo y selección de características para aplicar a continuación los modelos de aprendizaje supervisado *XGBoost* y *Deep-BP-ANN*. Por último se llevará a cabo una comparación entre LIME, SHAP y SMACE a la hora de obtener una interpretación de esos resultados.

1.2. Motivación personal

Llevo toda mi vida profesional (25 años) trabajando en operadoras de telecomunicaciones. A lo largo de estos años, una de las funciones que he desempeñado es la de análisis de las circunstancias que han llevado a los clientes a cambiar proveedor de servicios. Los métodos tradicionales de análisis me parecen insuficientes para abordar el tema desde una perspectiva que permita tanto identificar los motivos del abandono como las acciones necesarias para evitarlo. En cambio, la capacidad de cálculo y análisis que ofrecen los actuales métodos de *machine learning* son especialmente adecuados para este tipo de problemas, con multitud de factores que pueden impactar en la decisión final.

Me resulta muy interesante no solo poder predecir el *churn*, sino también poder obtener información que permita identificar acciones para reducirlo.

1.3. Objetivos

El **objetivo principal** de este trabajo es obtener un modelo predictivo para el *churn*, así como profundizar en la interpretacion de los resultados para poder ejecutar acciones que permitan reducir esta métrica.

Por otro lado, tendremos una serie de **objetivos parciales** que serán:

- Preprocesamiento del conjunto de datos:
  - Análisis exploratorio(EDA)
  - Técnicas de *Feature selection/extraction*
  - Aplicación de muestreos para balancear los datos
- Aplicacion de algoritmos supervisados
  - Modelo *baseline*: red neuronal profunda
  - Hiperparámetros: función de activación Relu, función de salida Sigmoid y optimizador Adam
  - Evaluación posterior del modelo mediante *Cross validation*
- Utilización de métodos post-hoc de interpretación de modelos
  - LIME
  - SHAP
  - SMACE

1.4. Metodología

La metodología utilizada para este trabajo será la metodología CRISP-DM, que consta de las fases que se listan a continuación:

1. Comprensión del negocio
   - Identificación de la necesidad de negocio a evaluar
   - Análisis del estado del arte
   - Determinación de los objetivos y elaboracion de plan de proyecto
2. Estudio y comprensión de los datos
   - Descripción y exploración de los datos iniciales
   - Control de calidad de los datos
3. Análisis de los datos y selección de características
   - Fase de *Feature selection/extraction*
   - Muestreo y balanceo de los datos
4. Modelado
   - Elaboración del modelo de aprendizaje supervisado
   - Aplicación de técnicas de interpretabilidad
5. Evaluación
   - Evaluacion de resultados
   - Determinación de nuevos pasos
6. Despliegue(conclusiones)
   - Elaboración de la memoria final
   - Defensa pública del TFM

En la Fig.1.1 se ve un esquema visual de la metodología, que permite observar las interacciones entre las diferentes fases del proyecto. Esta figura ha sido tomada de A visual guide to CRISP-DM methodology.

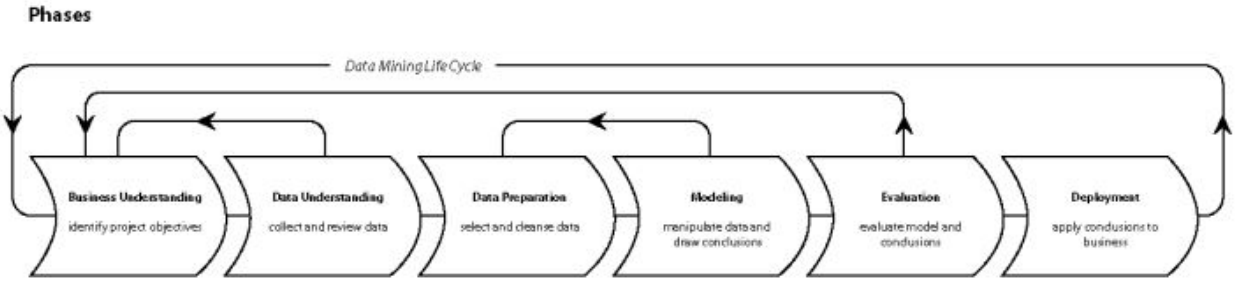

Figura 1.1: Esquema metodología CRISP-DM

1.5. Planificación

Definición y planificación del proyecto (1 semana): Definición básica del proyecto.

- Definición preliminar del proyecto
- Elección de objetivos, metodología y tecnologías utilizadas
- Planificación inicial

Estado del arte (3 semanas): Revisión sistemática de las diferentes aproximaciones a la interpretabilidad de los modelos. Identificación de referencias existentes y estudio sobre los ámbitos del proyecto.

- Obtención de información y referencias
- Lectura y revisión de la documentación
- Análisis de mercado para identificar la mejor aproximación

Diseño e implementación (9 semanas): Desarrollo del proyecto.

- Estudio y comprensión de los datos
- Análisis de los datos y selección de características
- Implementación de los modelos supervisados
- Implementación de modelos post-hoc
- Evaluación
- Conclusiones

Redacción de la memoria (2 semanas): Recopilación y revisión de la documentación del proyecto para ajustar cualquier modificación de alcance respecto a la planificación original. Generación de la memoria final.

- Primera versión/borrador
- Versión definitiva revisada

Defensa pública del TFM (1 semana): Presentación del TFM.

- Preparación del soporte documental gráfico
- Defensa pública del proyecto

En la Fig.1.2 se muestra el diagrama de Gantt correspondiente a la planificación del TFM mostrada anteriormente.

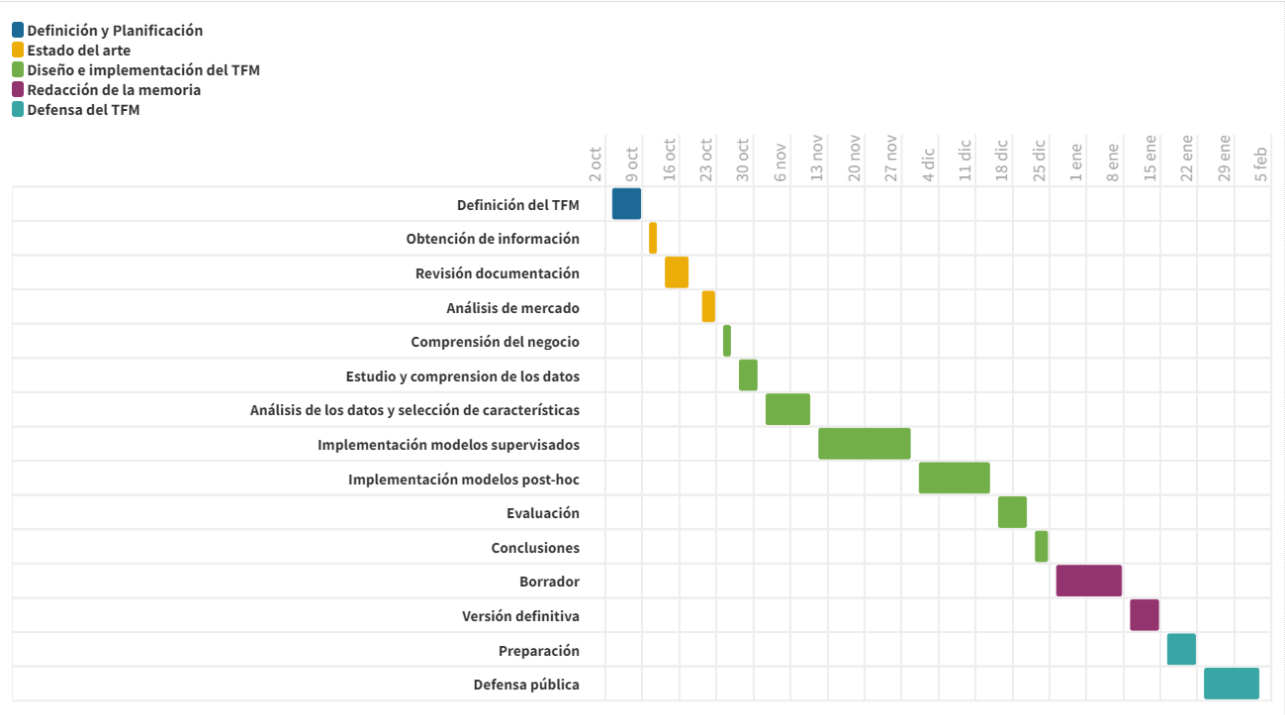

Figura 1.2: Diagrama de Gantt de este Trabajo Final de Máster

Por último, se muestra la correlación entre las diferentes fases de este TFM y la metodología CRISP-DM utilizada:

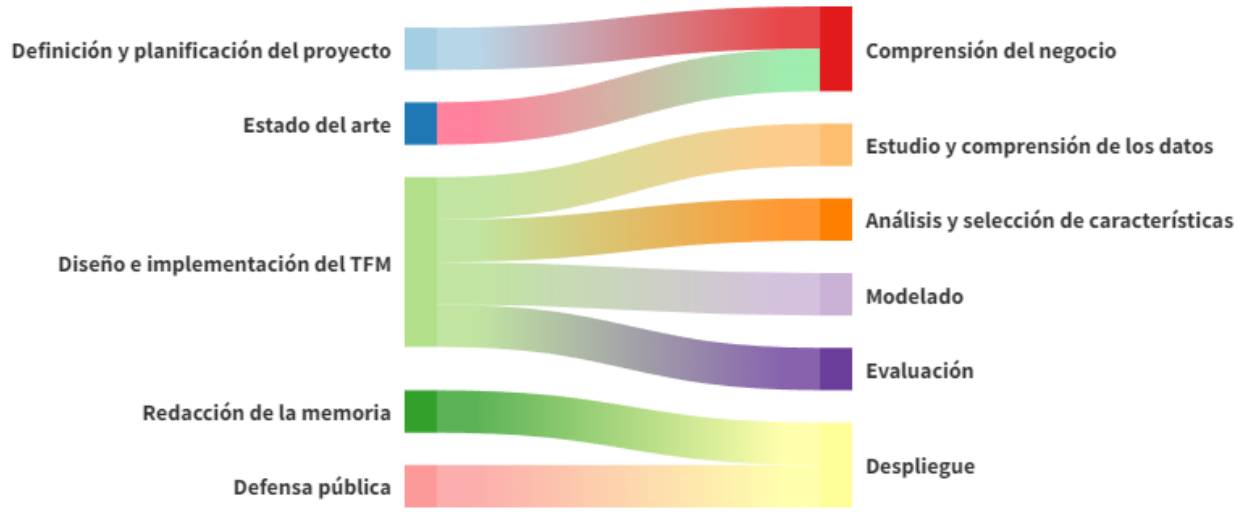

Figura 1.3: Relación Planificacion - Metodología CRISP-DM

1.6. Tecnología

A continuación se listan las tecnologías empleadas en la elaboración de este trabajo:

- Por su versatilidad y las numerosas librerías disponibles, se utilizará el lenguaje de programación **Python**, con Jupyter como IDE.
- Para el control de versiones, así como para recoger toda la información asociada al proyecto, utilizaremos este repositorio de **GitHub**.
- La redacción del trabajo se lleva a cabo con **LaTeX**, utilizando el editor online *Overleaf*.

Capítulo 2

Estado del arte

En este apartado se define el contexto de este trabajo. Para ello, se empieza analizando el concepto de interpretabilidad de modelos, su definición, taxonomía y los retos que presenta en la actualidad. A continuación, se habla sobre el *churn rate* dentro del campo del aprendizaje automático supervisado y la interpretabilidad de modelos. Entender en qué se está trabajando actualmente y qué posibilidades se ofrecen permitirá definir la propuesta de investigación. Por último, y tendiendo en cuenta todo lo anterior, se elige un conjunto de datos sobre el que diseñar e implementar dicha propuesta.

2.1. Interpretabilidad de modelos

La utilidad de los modelos de aprendizaje automático (*Machine learning*) en la predicción de fenómenos complejos es de sobras conocida y ha sido ampliamente estudiada. Además de la capacidad predictiva, estos modelos también permiten obtener información sobre las relaciones de dominio contenidas en los datos, siendo este un campo que ha despertado un gran interés en los últimos años. El conocimiento que se obtiene de estas relaciones es lo que se conoce como **interpretabilidad** del modelo. En esta sección se profundiza en la definición de este concepto, así como en la clasificación de los diferentes métodos de interpretación, siempre en el contexto del aprendizaje automático.

2.1.1. Definición de interpretabilidad

La interpretabilidad es un concepto complejo y subjetivo, lo que hace complicado encontrar una definición que sirva a todos los propósitos (Carvalho et al. [2019]). Se puede empezar considerando que interpretar es extraer conocimiento relevante de un modelo de aprendizaje automático sobre las relaciones contenidas en los datos o aprendidas por el modelo (Murdoch

et al. [2019]). A partir de aquí, es inevitable incluir en la ecuación el factor humano, la subjetividad mencionada anteriormente. Una de las definiciones más populares sería la de Doshi-Velez and Kim [2017], que afirma que, en el contexto de ML, *la interpretabilidad se define como la capacidad de explicar o presentar en términos comprensibles a un humano*. Otra alternativa sería la que presentan Biran and Cotton [2017] cuando indican que interpretabilidad es *el grado en que un observador puede entender la causa de una decisión*. No obstante, todas estas definiciones carecen de rigurosidad matemática y se basan más en la intuición que permite identificar la relación causa-efecto entre las entradas y las salidas de un modelo (Pantelis Linardatos and Kotsiantis [2020]), lo que lleva a elegir como definición para este trabajo la propuesta por Adadi and Berrada [2018]: *un sistema interpretable es un sistema en el cual el observador puede no solo ver sino entender y estudiar la forma en la que se relacionan matemáticamente las entradas y las salidas.*

2.1.2. Clasificación de métodos de interpretabilidad

A la hora de hablar de una taxonomía de los métodos de interpretabilidad, hay unanimidad (ejemplos en Carvalho et al. [2019], Díaz et al. [2021] y Pantelis Linardatos and Kotsiantis [2020]) en clasificar atendiendo a los siguentes criterios:

1. Pre-modelo vs. En-modelo vs. Post-modelo
2. Intrínseco frente a post hoc
3. Local frente a global
4. Modelo específico frente a modelo agnóstico

El **primer criterio** se refiere al momento en el que se aplican las técnicas de interpretabilidad, antes, durante o tras el modelo de aprendizaje automático. Las que se aplican antes del modelo son independientes de este, ya que se refieren a los propios datos a analizar. Estaríamos hablando por lo tanto de técnicas exploratorias de datos que irían desde la estadística descriptiva clásica hasta los métodos de visualización de datos, como el análisis de componentes principales (PCA) y el t-SNE, y también métodos de agrupación, como k-means (Carvalho et al. [2019]). La interpretabilidad en el modelo se refiere a los modelos que son intrínsecamente interpretables con estructuras (aprendidas) y parámetros (aprendidos) a los que se puede asignar una determinada interpretación (Molnar et al. [2021]), como por ejemplo los modelos de regresión lineal o los árboles de decisión. Por último, las técnicas post-modelo se refieren a las que se aplican tras el entrenamiento del modelo.

El **segundo criterio** es similar al anterior, en cuanto a que distingue si las técnicas de interpretabilidad están incluidas en el modelo de aprendizaje supervisado (lo que en el punto anterior considerábamos *en el modelo*), o se aplican tras el entrenamiento de este (*post-modelo*). Es importante destacar que algunas técnicas post-hoc pueden aplicarse también a modelos intrínsecamente interpretables al estar desvinculadas del modelo principal. En este caso, se trataría de enriquecer la interpretación propia del modelo añadiendo una capa adicional. Cuando se habla de modelos intrínsecos, también se incluiría a los modelos de caja blanca (*White box*), mientras que si el modelo dispone de múltiples capas (algunas ocultas, como en el caso de las redes neuronales), se habla de un modelo de caja negra (*Black box*)(Burkart and Huber [2021]).

El **tercer criterio** se refiere al grado de comprensión del modelo que se obtiene a partir de las técnicas. Se refiere, por lo tanto, del alcance de la interpretabilidad: entender el comportamiento completo del modelo o entender una única predicción (Adadi and Berrada [2018]). La interpretabilidad global permite entender de principio a fin la lógica de un modelo y el razonamiento que lleva a los posibles resultados; la interpretabilidad local nos ofrece una explicación individual que justifica por qué el modelo tomó una decisión específica para una instancia.

El **cuarto criterio** tiene que ver con la limitación que podemos encontrarnos a la hora de aplicar nuestras técnicas. Los métodos de interpretabilidad específicos se refieren a clases concretas de modelos, de forma que si necesitamos un tipo concreto de interpretación estamos limitados a la elección de los modelos que la proporcionan. Los métodos intrínsecos son por definición específicos del modelo. Por otro lado, los modelos agnósticos ofrecen la ventaja de no estar vinculados a un tipo concreto de modelo de aprendizaje supervisado. Las técnicas de interpretabilidad agnósticas suelen ser post-hoc, se utilizan generalmente para interpretar redes neuronales y pueden referirse a interpretabilidad local o global (Adadi and Berrada [2018]).

En la figura siguiente se muestra un resumen de diferentes técnicas de interpretabilidad (Adadi and Berrada [2018]) según los criterios anteriormente mencionados:

Técnicas	Intrínseco/ Post-hoc	Global/ Local	Model-especifico/ Model-agnostico
Árboles de decisión	I	G	SP
Rule lists	I	G	SP
LIME	H	L	AG
Shapely explanations	H	L	AG
Saliency map	H	L	AG
Activation maximization	H	G	AG
Surrogate models	H	G/L	AG
Parcial Dependence Plot (PDP)	H	G/L	AG
Individual Conditional Expectation (ACI)	H	L	AG
Rule extraction	H	G/L	AG
Decomposition	H	L	AG
Model distillation		G	AG
Sensitive Analysis	H	G/L	AG
Layer-wise Relevance Propagation (LRP)	H	G/L	AG
Feature importance	H	G/L	AG
Prototype and criticism	H	G/L	AG
Counterfactuals explanations	H	L	AG

I: Intrínseco, H: Post-hoc, G: Global. L:Local, SP: Model-specifico, AG: Model-agnostico

Figura 2.1: Características de técnicas de interpretabilidad

2.1.3. Elección de las técnicas de interpretabilidad

Teniendo en cuenta todo lo anterior, las técnicas empleadas a la hora de enfrentarse a un modelo de aprendizaje automático dependerán del modelo supervisado que utilicemos sobre el conjunto de datos. Para resumir el problema se tiene en cuenta la propuesta de (Burkart and Huber [2021]):

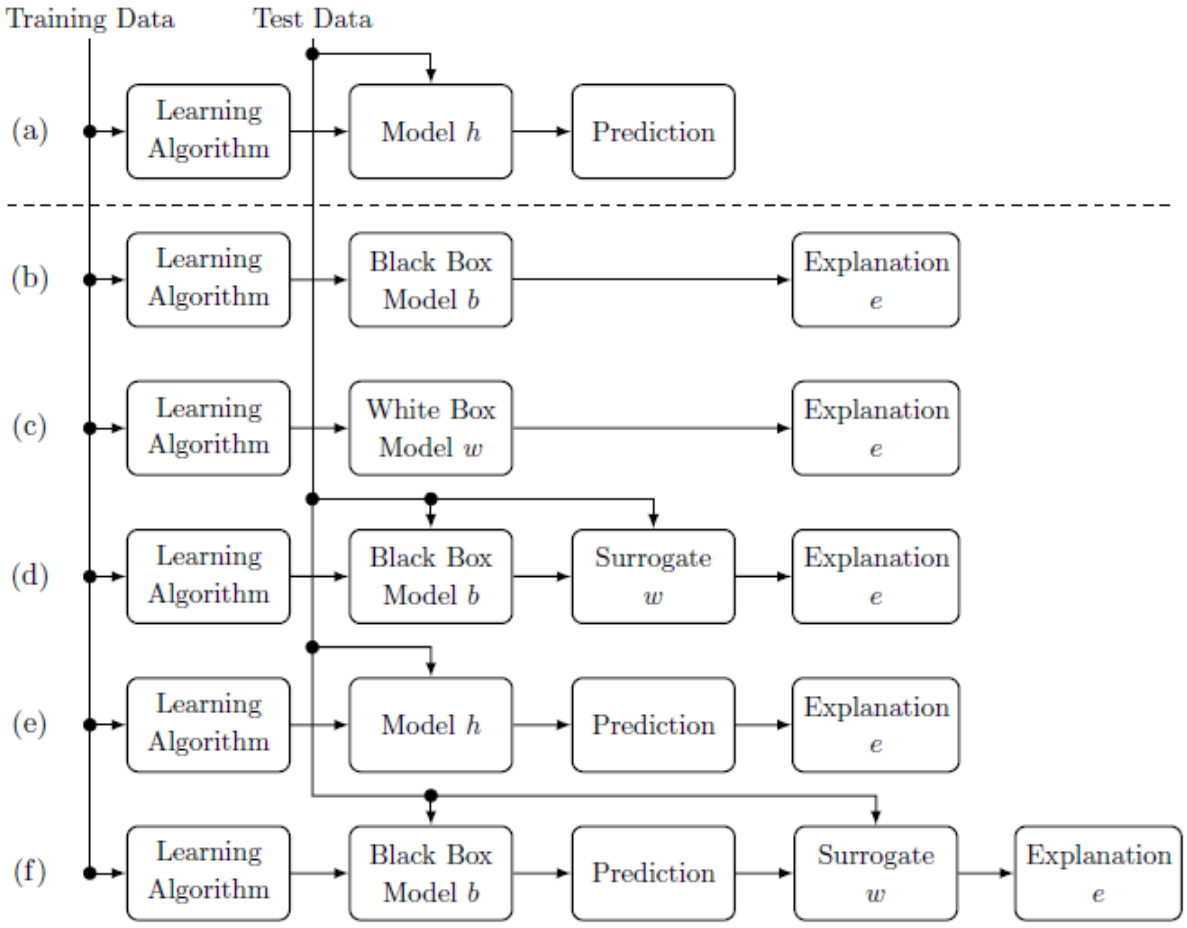

Figura 2.2: De un modelo supervisado opaco a una explicación

(a) Aprendizaje automático supervisado estándar sin explicación. (b)-(d) Explicaciones del modelo/global: (b) explicación post-hoc del modelo de caja negra, (c) interpretable por naturaleza, es decir, explicación del modelo de caja blanca y (d) explicación de un modelo de caja negra mediante un modelo global sustitutivo. (e)-(f) Explicaciones de instancia/locales: (e) directamente o (f) con un sustituto local.

No es el objeto de este trabajo hacer una revisión sistemática de las técnicas de interpretabilidad, sino fijar un contexto que permita elegir la mejor aproximación al problema planteado. No obstante, es interesante nombrar algunos ejemplos de modelos y técnicas asociados a la figura anterior:

- Modelos interpretables por naturaleza (Figura: 2.2, (c)): *decision trees*, *decision rules*, *generalized additive models (GAM)*.
- Modelos interpretables por diseño (Figura: 2.2, (c)): *decision trees*, *decision rules*, *decision sets*, modelos lineales, *surrogate fitting)*.

- Modelos sustitituvos globales (Figura: 2.2, (d)): SP-Lime, k-Lime, *decision tree extraction*, BRAINE, STARE, REX, RuleFit.

- Modelos sustitituvos locales (Figura: 2.2, (f)): Lime, MES, Kernel SHAPE, Linear SHAPE.

- Explicaciones globales (Figura: 2.2, (b)): *RF Feature Importance*, PCA, t-SNE, Residual Analysis.

- Explicaciones locales (Figura: 2.2, (e)): *border classification*, ICE plots, PVM, SVM recommendations.

2.2. Predicción del *churn rate* en el campo del aprendizaje automático

Para analizar en qué se está trabajando actualmente en cuanto a predicciones de *churn* en el campo del aprendizaje automático y la interpretabilidad de modelos, se ha limitado la búsqueda a publicaciones cuya antiguedad es inferior a dos años. Asimismo, se ha centrado en aquellos artículos que contemplen el mercado de las telecomunicaciones (en exclusiva o como parte de un análisis global del concepto en diferentes ámbitos (Ahn et al. [2020]). Esta búsqueda se ha llevado a cabo en *Google Schoolar* y *Scopus* con los resultados que se comentan a continuación.

Se han analizado en primer lugar las publicaciones que hacen un recorrido detallado por las diferentes técnicas empleadas, conjuntos de datos y resultados obtenidos en otros estudios. Jain et al. [2020] llevan a cabo una recopilación de las investigaciones llevadas a cabo entre 2005 y 2020 y concluyen que la utilización de métodos de *Feature Selection* es muy relevante en la mejora del modelo así como utilizar datos de entrada que incluyan una componente temporal. Entre los métodos de *Feature Selection* proponen PCA y *co-relation matrix* por su efectividad. En cuanto a modelos, las redes neuronales convolucionales son las que muestran mejores resultados. Por último, proponen que para medir los resultados obtenidos no se utilice exclusivamente la *Accuracy* sino también *confusion matrix*, *precision*, *recall* y *f-score*. Por su parte, Ahn et al. [2020] llevan a cabo un recorrido por los estudios realizados en diferentes sectores con la intención de que futuros investigadores puedan realizar la mejor selección de técnicas, definiciones y modelos. Especialmente interesante es la evolución que muestran de los algoritmos utilizados en este campo en el periodo 1998-2020:

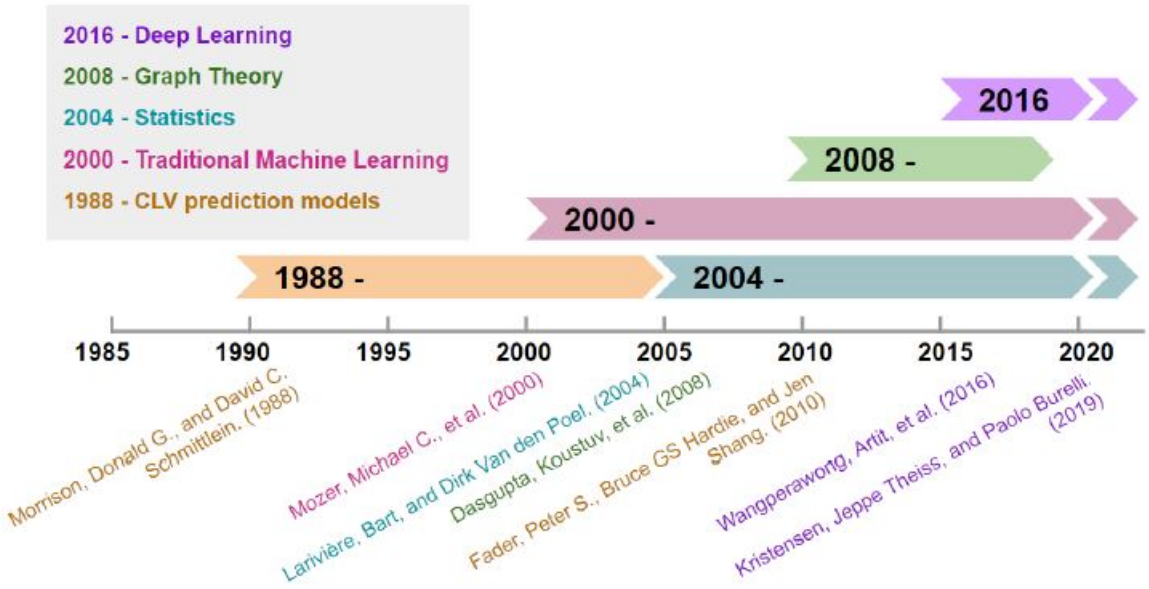

Figura 2.3: Evolución de los modelos de aprendizaje automático para la predicción del churn

En sus conclusiones, afirman que a pesar de que las técnicas de *deep learning* son las que muestran mejores resultados, el machine learning tradicional se sigue utilizando para casos concretos (según la estructura del conjunto de datos) en los que su efectividad sigue siendo elevada. Como medidas de rendimiento del modelo, proponen la curva ROC y AUC.

A continuación se resumen las publicaciones que optan por un análisis comparativo de diferentes algoritmos y métodos. Beeharry and Fokone [2022] utilizan un conjunto de dos capas de clasificadores, cada uno con cuatro algoritmos, tras el que introducen un clasificador de voto *suave* que promedia las probabilidades de las clases. Para probar el modelo, además, utilizan dos conjuntos de datos desbalanceados que procesan para obtener una versión balanceada. Sus conclusiones insisten en la mejora del modelo combinado frente a cada uno de los algoritmos por separado, y en la importancia de obtener conjuntos de datos balanceados. Geiler et al. [2022], por su parte, llevan a cabo un exhaustivo análisis de 11 algoritmos supervisados y semisupervisados, junto con siete técnicas de muestreo sobre 16 conjuntos de datos de acceso público. Utiliza además *Crosss validation*, *K-fold validation* y *Stratified k-validation* sobre los resultados, utilizando la métrica AUC para valorar y comparar todos los métodos utilizados. Sus conclusiones se basan en la comparación del AUC de los diferentes algoritmos y combinación de algoritmos utilizados en condiciones de muestreo/no muestreo. Lalwani et al. [2021] lleva a cabo un trabajo con una orientación similar, comparando 13 modelos supervisados diferentes sobre los que aplica *K-fold cross validation*. Además, introduce un preprocesado que incluye *Fearure selection*. La evaluación del rendimiento de los modelos la lleva a cabo mediante métricas AUC,

Confusion matrix, *Accuracy*, *Recall*, *Precision* y *f-score*. Sus resultados experimentales muestran que dos técnicas *ensemble*, Adaboost y XGBoost, obtienen las mejores puntuaciones en términos de AUC.

En la misma línea anterior, pero orientándose hacia el *deep learning*, Mahmood and Abdullah [2022] propone el uso de una red neuronal profunda (128-64*2-36*4-16*4-1) con función de activación Relu y función de salida Sigmoid. El resultado lo compara con otros algoritmos tradicionales de clasificación mediante las métricas habituales *Confusion matrix*, *Accuracy*, *Recall*, *Precision* y *f-score*. Sus conclusiones muestran que los resultados son mejores en todas las métricas consideradas si se aplica una red neuronal profunda. Dalli [2022] enfoca su trabajo directamente sobre *deep learning*, basándose en estudios anteriores que afirman que obtiene mejores resultados para la predicción del churn que las técnicas tradicionales de *machine learning*. Utilizando una red neuronal 10-6-6-1, se enfoca en la optimización de sus hiperparámetros. Tras llevar a cabo sus experimentos, proponen el uso de la función de activación Relu, la función de salida Sigmoid, y el optimizador RMSProp. En cuanto al *Batch size*, no parece que sus variaciones tengan un impacto relevante en los resultados, siendo mejores cuando menor es aquel. Todos los resultados se ordenan de acuerdo a la *Accuracy* obtenida.

Fujo et al. [2021] utilizan también una red neuronal profunda con *Back propagation*. En su experimento incluyen una fase extensa de EDA (análisis exploratorio de los datos), dos diferentes métodos de *Feature selection*, *Lasso Regularisation* y *Variance Thresholding* y *Random oversampling* para evitar el desbalanceo de los datos. En cuanto a métodos de evaluación encontramos *Holdout* y *K-fold cross validation*, y las métricas AUC,*Confusion matrix*, *Accuracy*, *Recall*, *Precision* y *f-score*. En sus conclusiones comparan sus resultados con los de otros estudios similares, mostrando mejores resultados en cuanto a *Accuracy*.

Por último, se listan los estudios realizados que incluyen técnicas de interpretabilidad. Nkolele and Wang [2021] proponen el uso de tres métodos tradicionales de aprendizaje automático (*Decision trees*, *Random Forest* y *Light Gradient Boosting Machine*). A continuacion, utilizan LIME y SHAP sobre sobre los dos últimos ya que consideran que el modelo *Decision tree* es un modelo intrínseco. En sus conclusiones afirman que los resultados son mejores con el uso de un algoritmo LGBM, y que el uso de técnicas de interpretabilidad permite aumentar el conocimiento extraído del análisis. Por su parte, Mirabdolbaghi and Amiri [2022] utilizan como modelo base *Light Gradient Boosting Machine*, aplicando técnicas de *Feature reduction* junto con optimización bayesiana de hiperparámetros y algoritmos de optimización genética. Para la interpretación y evaluación del impacto de las *features* utilizan SHAP. Los resultados obtenidos se comparan con otras técnicas tradicionales de aprendizaje automático resultando su aproximación la que mejores resultados obtiene. En cuanto a SHAP, su utilización ayuda al equipo

de marketing a comprender las razones subyacentes de la pérdida de clientes y les ayuda a identificar mejor los puntos de tracción. En una línea completamente diferente a lo que hemos estado viendo hasta ahora, Lopardo et al. [2022] propone y desarrolla un nuevo método de interpretabilidad: SMACE (Semi-Model-Agnostic Contextual Explainer). Además de explicar la teoría de la nueva técnica, lleva a cabo un par de análisis comparativos sobre tres conjuntos de datos diferentes (uno de ellos es sobre churn en una empresa de telecomunicaciones). En este análisis compara SMACE a LIME y SHAP, mostrando que, en el caso que nos interesa, los resultados de las tres técnica son similares, pero SMACE mejora la interpretabilidad ya que es capaz de clasificar las *features* por sensibilidad, mientras que SHAP y LIME tienden a atribuir explicaciones idénticas.

2.3. Propuesta de investigación

Tras el análisis realizado sobre los estudios llevados a cabo en los dos últimos años, la propuesta de este trabajo se centra en (1) la elección de un conjunto de datos sobre el que (2) aplicar un modelo de aprendizaje automático. A continuación, a los resultados se les aplicará (3) una técnica de interpretabilidad para entender mejor el impacto y evaluar su importancia. A continuación detallamos la propuesta de investigación para estos tres conceptos.

2.3.1. Modelo de aprendizaje automático

En la literatura revisada (Dalli [2022], Mahmood and Abdullah [2022] y Fujo et al. [2021]), se ha visto que las redes neuronales profundas son las que presentan mejores resultados en cuanto a *Accuracy* en la prediccion del *churn*. Es importante destacar, además, la importancia de que el conjunto de datos esté balanceado (Beeharry and Fokone [2022]), lo que implica un análisis exploratorio de los datos que incluya métodos de muestreo (Geiler et al. [2022]) y de *Feature reduction/selection* (Lalwani et al. [2021]). Tras la aplicacion del modelo, será necesario, asimismo, aplicar métodos de validación (Fujo et al. [2021]).

Teniendo en cuenta lo anterior, se utiliza como referencia el trabajo de Fujo et al. [2021] tanto para la preparación de los datos como para la implementación del modelo. Se incluye, no obstante, una variación en cuanto al muestreo de datos según Geiler et al. [2022], ya que se aplica su propuesta híbrida SMOTE (*oversampling*) y NCR (*undersampling*).

2.3.2. Técnicas de interpretabilidad

Si bien no se han encontrado muchos ejemplos de aplicación de técnicas de interpretabilidad a la predicción del *churn* en el sector de las telecomunicaciones, resulta interesante la propuesta

de Lopardo et al. [2022]. Se replicará, por lo tato, su análisis comparativo entre LIME, SHAP y SMACE, para ver si los resultados que se obtienen son consistentes con su propuesta. Su trabajo será, por lo tanto, la referencia principal de este proyecto.

2.3.3. Conjunto de datos

Una vez fijados los dos puntos anteriores, es necesario elegir el conjunto de datos sobre el que llevar a cabo el estudio. Los criterios para elegirlo serán, por un lado, que haya sido utilizado en anteriores estudios para poder comparar los resultados obtenidos, y por otro lado, que no sea el que utiliza el estudio de referencia, IBM Telco Churn dataset (Lopardo et al. [2022]). Teniendo en cuenta lo anterior, se ha optado por utilizar el conjuntos de datos de Cell2Cell (Beeharry and Fokone [2022] y Fujo et al. [2021]).

Capítulo 3

Estudio y comprensión de los datos (EDA)

Este apartado se ocupa del estudio y comprensión de los datos previo a la implementación del modelo. En concreto se corresponde con la fase 2 de la metodología CRISP-DM (apartado 1.4).

El conjunto de datos seleccionado presenta 58 variables (siendo una de ellas la variable objetivo) y 51.047 registros. En cuanto a las variables, 34 de ellas son numéricas y el resto categóricas. En ese apartado haremos una limpieza de datos que incluye analizar y decidir qué hacemos con los valores nulos y los outliers, estudiar el reparto de la variable objetivo, analizar las distribuciones de las variables numéricas y categóricas y, por último, llevar a cabo una matriz de correlación para determinar si existen variables que pueden descartarse directamente antes de la fase siguiente.

3.1. Registros

3.1.1. Valores nulos

El primer análisis a realizar es comprobar si nuestra información es completa. Para ello, se ha revisado la existencia de registros sin información en alguna de las variables. Se han encontrado valores *nan* en 14 variables: *MonthlyRevenue, MonthlyMinutes, TotalRecurringCharge, DirectorAssistedCalls, OverageMinutes, RoamingCalls, PercChangeMinutes, PercChangeRevenues, ServiceArea, Handsets, HandsetModels, CurrentEquipmentDays, AgeHH1 y AgeHH2*. En total, hay 1.295 registros que tienen alguna variable sin información, por lo que se ha procedido a eliminarlos dado el poco volumen que representan frente al número total de registros (2,5 %). Por lo tanto el número final de registros considerados es de 49.751.

3.1.2. Outliers

Si bien existen valores extremos en algunos registros, nos ha parecido importante no eliminarlos, ya que cada registro se corresponde a un cliente y los extremos también nos pueden proporcionar información relevante.

3.1.3. Reparto variable objetivo

En cuanto a la variable objetivo presenta un reparto no balanceado, ya que los registros con Churn = Yes no llegan al 30 %, como se muestra en la figura 3.1. Eso lleva a confirmar la necesidad de llevar a cabo un muestreo de datos en la siguente fase del proyecto (Análisis de los datos y selección de las características).

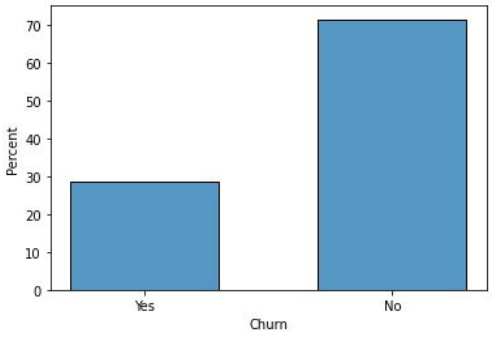

Figura 3.1: Variable objetivo en el conjunto de datos

3.2. Variables numéricas

Las variables numéricas se reparten según el tipo de datos en enteras (8 casos) y decimales (26 casos). No es necesario llevar a cabo ningún tipo de transformación sobre ellas. No presentan una distribución normal en la mayoría de los casos, mostrando además una dispersión notable en cuanto a sus valores. De la misma forma que con los registros, de este hecho se deducen acciones a realizar en fases posteriores; en concreto el llevar a cabo una normalización antes de trabajar con los datos.

Las distribuciones comentadas se pueden ver en la figura 3.2.

Figura 3.2: Distribución variables numéricas

3.3. Variables categóricas

Las variables categóricas se reparten entre aquellas que son binarias (presentan únicamente dos valores) y las que presentan más de dos. El primer paso ha sido eliminar las variables que no ofrecen información relevante para este estudio, como el código del cliente, o el área geográfica en la que se encuentra (*CustomerID, ServiceArea, Prizmcode)*. Dado que es necesario trabajar con variables numéricas, ha habido que llevar a cabo algunas transformaciones para preparar los datos para siguientes fases:

- Variables binarias: Se ha aplicado una transformación *LabelBinarize*, que convierte los valores *Yes* a *1* y *No* a *0*.

- Variables no binarias: sobre estas, se ha aplicado una categorización previa (para reducir la dispersión de valores) en *CreditRating* y *HandsetPrice* y a continuación una transformación *getdummies* en el resto. Este último paso lleva a aumentar el número total de variables en 12: 7 debidas a la variable *Ocupation*, 3 a la variable *HandsetPriceCat* y 2 a la variable *MaritalStatus*

El resultado se muestra en la figura 3.3.

Figura 3.3: Distribución variables categóricas

Tras las transformaciones anteriormente mencionadas, el conjunto de datos en estudio mantiene el mismo número de resgitros (49.751), pero aumenta el número de características a 67 (incluyendo la variable objetivo).

3.4. Correlaciones

Se ha llevado a cabo la matriz de correlación de las 67 variables y no se ha encontrado ninguna que sea descartable de forma directa. Dado el elevado número de variables implicadas la visualización resulta compleja. No obstante, tras las transformaciones sobre las variables que se explican en el capítulo siguiente, se muestra la matriz de correlación resultante en el apartado 4.3.3.

Capítulo 4

Análisis de los datos y selección de las características

En este apartado se trata el procesado y preparación de los datos previo a la implementación del modelo (fase 3 de la metodología CRISP-DM, apartado 1.4). Se trabajará en las dos líneas que resultan óptimas según la literatura consultada: por un lado el muestreo de los datos, y por otro lado la selección de características. Las dos líneas combinadas aumentarán el número de registros, pero limitarán las variables a las más relevantes estadísticamente para este estudio.

Antes y después de la aplicación de cada una de las técnicas anteriores, se utilizará un modelo XGBoost para comparar el rendimiento de cada uno de los pasos usando como referencia la métrica **accuracy** (según el artículo de referencia). No obstante, se obtendráns también el resto de métricas más relevantes para poder llevar a cabo la comparativa con mayor detalle.

4.1. Modelos utilizados

Los modelos de aprendizaje supervisado que se van a utilizar en este proyecto serán un **XGBoost** y un **Deep-BP-ANN**, de acuerdo al artículo de referencia ya que son los dos modelos que obtienen mejores resultados. Debido al tiempo de procesado de cada uno de ellos, se elegirá XGBoost para evaluar el impacto en las métricas de los cambios tras el muestreo y la selección de las características, y la red neuronal profunda sobre los datos ya preparados en el siguiente apartado (5).

4.1.1. *XGBoost*

Este modelo es un método de aprendizaje automático supervisado para clasificación y regresión. XGBoost es la abreviatura de las palabras inglesas *extreme gradient boosting* (refuerzo

de gradientes extremo). Este método se basa en árboles de decisión y supone una mejora sobre otros métodos, como el bosque aleatorio y refuerzo de gradientes. Sus principales ventajas son: trabaja bien con grandes bases de datos con múltiples variables, puede manejar valores perdidos, sus resultados son muy precisos y tiene una excelente velocidad de ejecución.

El algoritmo XGBoost funciona de la siguiente forma (Espinosa-Zúñiga [2020]):

- Es un ensamblado secuencial de árboles de decisión (llamado CART, acrónimo de "Classification and Regression Trees"). Los árboles se agregan secuencialmente a fin de aprender del resultado de los árboles previos y corregir el error producido por los mismos, hasta que ya no se pueda corregir más dicho error (esto se conoce como "gradiente descendente", figura 4.1).

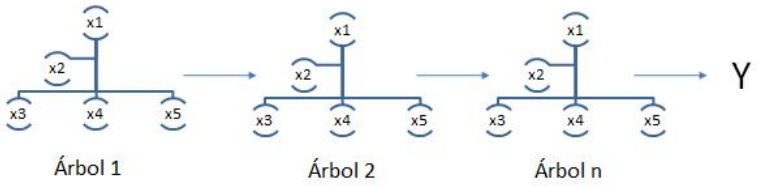

Figura 4.1: Algoritmo XGBoost

- A diferencia de otros árboles de decisión como el Random Forest, en XGBoost el usuario define la extensión de los árboles.

- Utiliza procesamiento en paralelo, poda de árboles, manejo de valores perdidos y regularización (optimización que penaliza la complejidad de los modelos) para evitar en lo posible sobreajuste o sesgo del modelo.

4.1.2. Red neuronal artificial profunda con retropropagación (*Deep-BP-ANN)*

Una red neuronal artificial consiste en la interconexión de unas partículas elementales llamadas neuronas. Cada una de estas neuronas recibe una entrada, ejecuta sobre ella una transformación y devuelve una salida como se ve en la figura 4.1(Bosch et al. [2019]).

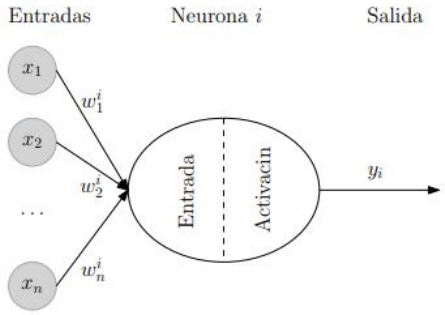

Figura 4.2: Estructura básica de una neurona

Las entradas de la neurona (x) están ponderadas por unos pesos (w). Sobre esas entradas, ejecutamos una función (*función de entrada o combinación)* y sobre el resultado una *función de activación (o transferencia)* que nos da la salida de la neurona. Las funciones de entrada pueden ser una suma ponderada, funciones máximo o mínimo, o funciones AND/OR si las entradas son binarias. En cuanto las funciones de activación, nos encontramos con las funciones escalón, lineal, sigmoide y rectificadora.

La arquitectura o topología de una red neuronal se define como la organización de las neuronas en distintas capas, así como los parámetros que afectan a la configuración de estas (funciones de entrada o activación). Según sea el tipo de problema que estemos analizando utilizaremos una arquitectura u otra. Siempre tendremos una capa de entrada y una de salida como mínimo. A partir de ahí, podemos añadir capas ocultas intermedias que nos llevarán a una configuración monocapa o multicapa.

En el caso que nos ocupa, se utilizará una arquitectura multicapa, donde la capa de entrada tendrá tantas neuronas como variables del conjunto de datos, y la capa de salida será una única neurona ya que se está frente a un problema de clasificación binaria. Asimismo, se aplicará un algoritmo de retropropagación que se basa en dos pasos (Bosch et al. [2019]):

1. Propagación hacia adelante (**feedforward**), que consiste en introducir una instancia de entrenamiento y obtener la salida de la red neuronal.

2. Propagación hacia atrás (**backpropagation**), que consiste en calcular el error cometido en la capa de salida y propagarlo hacia atrás para calcular los valores delta de las neuronas de las capas ocultas.

Para acabar con esta breve descripción de las redes neuronales artificiales, faltaría hablar sobre los hiperparámetros, que se describen brevemente a continuación brevemente:

- **Épocas**, se refiere al número de veces que se pasa el conjunto de datos completo por una red neuronal.

- **Tamaño de lote**, el tamaño del lote sobre el que se itera.
- **Iteraciones**, el número de de lotes necesario para completar una época.
- **Optimizador**, que no es más que el responsable de optimizar los valores de los parámetros para reducir el error cometido por la red.
- **Velocidad de entrenamiento**, que controla los pesos de la red neuronal con respecto al descenso de gradiente. Define la rapidez con la que la red neuronal actualiza los conceptos que ha aprendido.

4.2. Muestreo (*Balancing*)

Se ha llevado a cabo un muestreo híbrido según la propuesta de Geiler et al. [2022]. Aunque el artículo de referencia para esta fase del proyecto propone un muestreo diferente, la información recopilada sobre el muestreo híbrido presenta mejores resultados.

4.2.1. *SMOTE*

Se aplica en primer lugar la técnica *SMOTE*, que consiste en sobremuestrear la clase minoritaria (Churn = Yes) generando nuevos registros utilizando un enfoque k-nn. Frente a la duplicación simple de registros aleatorios, los que se generan mediante *SMOTE* son observaciones plausibles generadas a partir de registros existentes.

4.2.2. *NCR*

A continuación, sobre el conjunto resultante, se aplica *NCR*. Esta técnica combina dos métodos que eliminan de la clase mayoritaria (Churn = No) los registros que son (i) redundantes y (ii) ruidosos o ambiguos.

El primer método es *Condensed Nearest Neighbor (CNN)*, que selecciona un subconjunto de observaciones de la clase mayoritaria que no puede ser clasificado correctamente. Estas muestras se consideran más relevantes para el aprendizaje.

El segundo método es *Edited Nearest Neighbors (ENN)*: si un registro de la clase mayoritaria es mal clasificado por sus vecinos, se elimina del conjunto de datos. Además, si un registro de la clase minoritaria es clasificado erróneamente por sus vecinos de la clase mayoritaria, los vecinos de la clase mayoritaria también son eliminados.

4.2.3. Resultados

En la tabla siguiente se muestra paso a paso el proceso seguido al aplicar el muestreo híbrido anteriormente descrito:

Conjunto de datos	Registros (Churn = No)	Registros (Churn = Yes)
Original	35.507	14.245
SMOTE	35.507	35.507
NCR	35.507	32.402

Cuadro 4.1: Muestreo

Para calibrar el impacto del muestreo realizado utilizamos el modelo XGBoost comentado anteriormente con los resultados que se muestran a continuación:

Conjunto de datos	Características	Accuracy	F1 score	Precision	Recall	AUC
Original	66	70,88 %	32,15 %	49,00 %	23,93 %	56,92 %
SMOTE y NCR	66	82,43 %	79,58 %	89,37 %	71,73 %	81,97 %

Cuadro 4.2: Selección de Características y Muestreo

La aplicación del muestreo propuesto sobre el conjunto de datos ha supuesto una mejora muy relevante sobre las métricas obtenidas con anterioridad; no solo aumenta en 12 puntos porcentuales la *Accuracy*, sino también el resto de métricas consideradas, con incrementos aún más notables.

4.3. Selección de características (*Feature Selection*)

Tras el muestreo anterior, Se ha llevado a cabo un proceso de selección de variables de acuerdo al trabajo de Fujo et al. [2021]. Para ello se han aplicado dos técnicas consecutivas: *Lasso Regression* (Kumarappan [2020]) y *Variance Thresholding* (McCombe [2019]), que se detallan a continuación.

4.3.1. *Lasso Regression*

Esta técnica de selección de características penaliza la suma del valor absoluto de los coeficientes de regresión:

$$\Sigma_{j=1}^{n}(x_i - \Sigma_j x_i j \beta_j)^2 + \lambda \Sigma_{j=1}^{p} |\beta_j|$$

A esta penalización se le conoce como l1 y tiene el efecto de forzar a que los coeficientes de los predictores tiendan a cero. Dado que un predictor con coeficiente de regresión cero no influye en el modelo, Lasso consigue excluir los predictores menos relevantes. Es una de las técnicas embebidas más relevantes en la actualidad.

4.3.2. *Variance Thresholding*

La otra técnica utilizada para la reducción de características ha sido el umbral de varianza. La varianza de una característica determina el poder de predicción que contiene. Cuanto más baja es la varianza, menos información contiene y menos valor tiene para predecir la variable objetivo. Teniendo en cuenta este hecho, esta técnica se aplica calculando la varianza de cada característica, y luego eliminando todas las características por debajo de un determinado umbral de varianza. En el caso que nos ocupa, hemos mantenido el mismo valor que Fujo et al. [2021], fijando el umbral de selección en 0,5.

4.3.3. Resultados

Antes y después de la aplicación de cada una de las técnicas anteriores, se ha utilizado el modelo XGBoost comentado para comparar el rendimiento de cada uno de los pasos. La métrica utilizada ha sido la **accuracy**.

Los resultados son similares a los obtenidos en Fujo et al. [2021], y pueden verse en el cuadro 4.3. Se han mantenido los datos de la fase anterior para una mejor comprensión del proceso.

Conjunto de datos	Características	Accuracy	F1 score	Precision	Recall	AUC
Original	66	70,88 %	32,15 %	49,00 %	23,93 %	56,92 %
SMOTE y NCR	66	82,43 %	79,58 %	89,37 %	71,73 %	81,97 %
Lasso Regression	64	82,33 %	79,51 %	89,07 %	71,80 %	81,88 %
Variance Threshold	29	82,37 %	79,46 %	89,54 %	71,42 %	81,90 %

Cuadro 4.3: Muestreo y Selección de Características

Tras la aplicacion de las técnicas propuestas, se puede observar la notable mejora que supone la aplicación de técnicas de muestreo en cuánto las métricas obtenidas con un modelo *XGBoost* (Geiler et al. [2022]). Tras este muestreo, se puede concluir que la aplicación de los métodos de selección de características propuestos por Fujo et al. [2021] permiten reducir notablemente el número de variables del conjunto de datos con un impacto mínimo en las métricas obtenidas.

La aplicación de las técinas de preparación de los datos comentadas, reducen notablemente la carga de procesado del modelo al reducir el número de características de 66 a 29, sin com-

prometer el rendimiento de este, ya que no solo no empeora la métrica utilizada en el modelo XGBoost, sino que mejora respecto al conjunto de datos original.

Por último, y como habíamos comentado en el apartado 3.4, añadimos a continuación la matriz de correlación de las 29 variables resultantes:

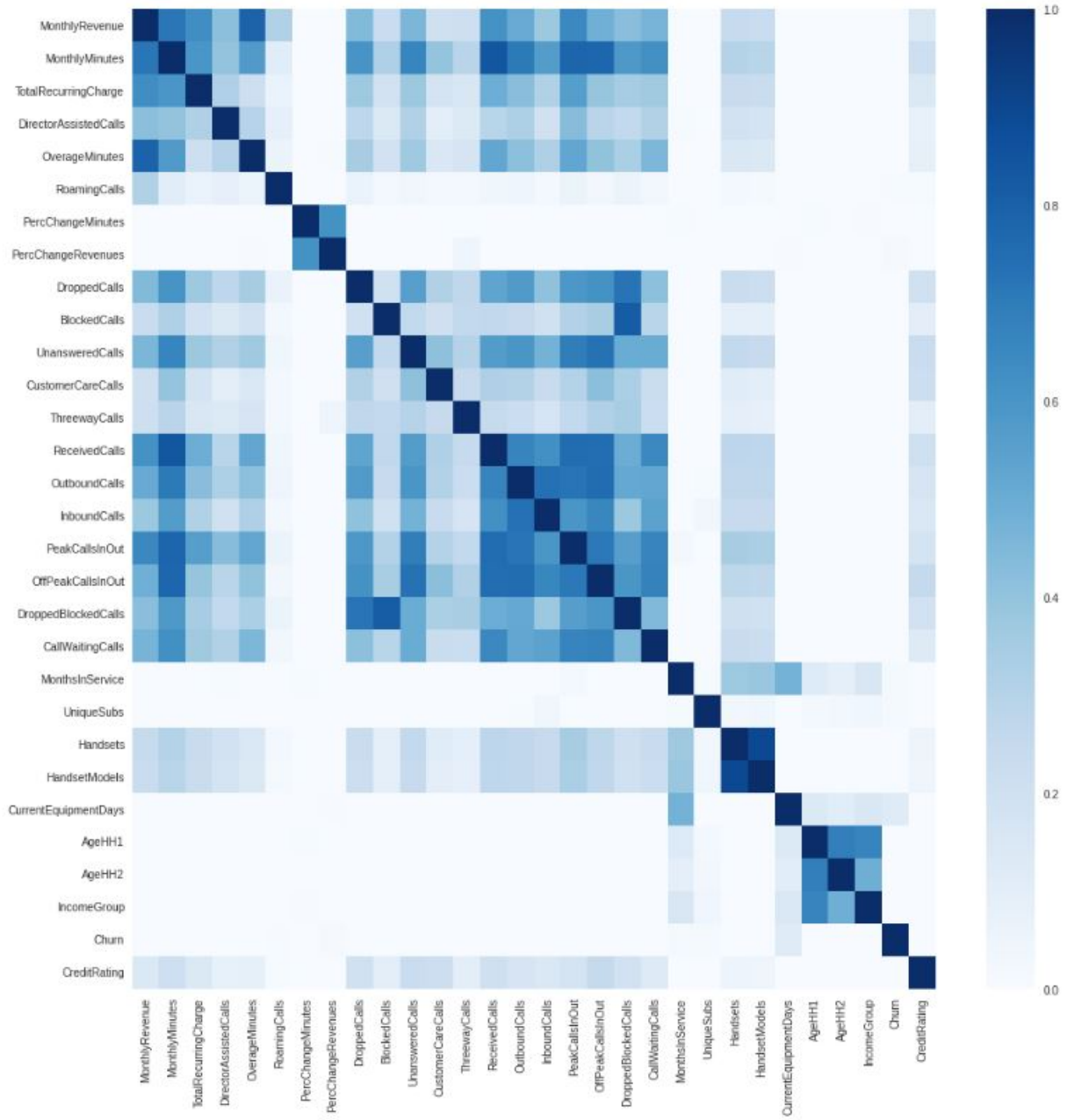

Figura 4.3: Matriz de correlación de las variables resultantes

Capítulo 5

Modelado

Una vez procesado el conjunto de datos, en este capítulo se aplicará un modelo de aprendizaje supervisado de acuerdo al modelo utilizado en Fujo et al. [2021]. A continuación, se utilizarán técnicas de interpretabilidad según Ahn et al. [2020] para confirmar si esta propuesta obtiene los mismos resultados con el conjunto de datos en estudio.

5.1. Elaboración del modelo de aprendizaje supervisado

Dado que se ha introducido un cambio en el procesado previo en cuanto a las técnicas de muestreo, los valores óptimos de algunos hiperparámetros difieren de los del artículo de referencia. A continuación se detalla el proceso llevado a cabo y los resultados obtenidos.

5.1.1. Optimización hiperparámetros

El modelo base utilizado ha sido el propuesto por Fujo et al. [2021], una red neuronal artificial profunda con retropropagación (*backpropagation*). En la capa de entrada se utiliza el mismo número de neuronas que el del número de variables obtenido en el apartado anterior y una función de activación ReLU; en las capas ocultas, el número de neuronas propuesto se mantiene en 250 con la misma función de activación, y en la capa de salida la función de activación elegida ha sido la sigmoidea.

A continuación se ha fijado el optimizador de retropropagación Adam con los valores de *learning rate* por defecto y se ha iterado sobre el número de épocas, el tamaño del *batch* y el número de capas ocultas. Una vez elegidos los hiperparámetros anteriores, se han fijado en el modelo y se ha iterado sobre el *learning rate* para elegir el valor óptimo. En todos los casos, el criterio utilizado ha sido el de maximizar la métrica *accuracy*.

Tras lo anterior los valores elegidos han sido muy similares a los del artículo de referencia,

con un tamaño del *batch* de 64, dos capas ocultas y un learning rate de 0,0001. Por otra lado, se ha introducido un *early stopping* maximizando la *accuracy* que ha reducido el número de épocas a 122.

5.1.2. Validación Cruzada

Obtenidos los hiperparámetros, y antes de entrenar el modelo, se ha aplicado *K-fold validation* para confirmar que el ajuste del modelo era correcto. Los datos muestran que no existe overfitting y que los valores del conjunto de entrenamiento y de validación presentan valores muy similares en todos los casos:

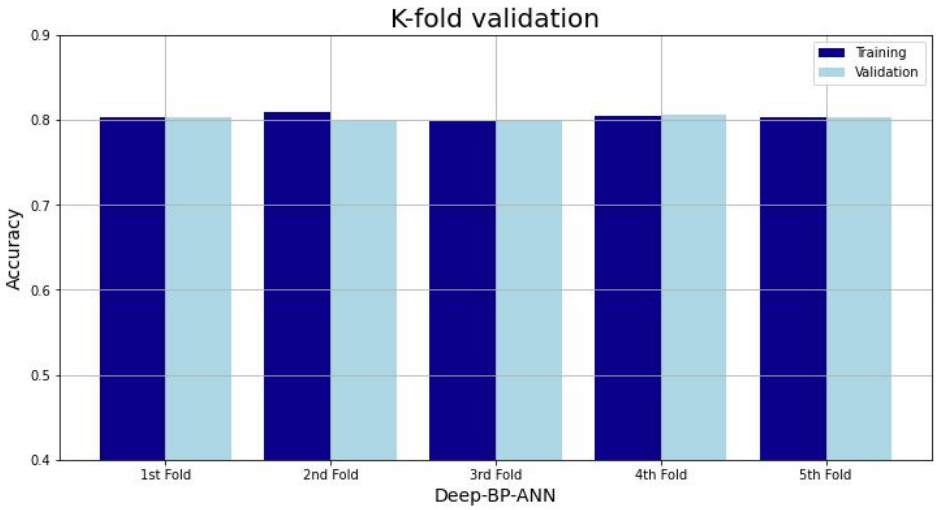

Figura 5.1: K-fold Cross Validation

5.1.3. Resultados

Con todo lo anterior fijado, se ha entrenado el modelo y se han obtenido los resultados que se muestran en la tabla a continuación 5.1. Tal y como se esperaba, los valores son ligeramente mejores que los obtenidos con el modelo utilizado en la fase previa, XGBoost.

Conjunto de datos	Características	Accuracy	F1 score	Precision	Recall	AUC
Original	66	70,88 %	32,15 %	49,00 %	23,93 %	56,92 %
SMOTE y NCR	66	82,43 %	79,58 %	89,37 %	71,73 %	81,97 %
Lasso Regression	64	82,33 %	79,51 %	89,07 %	71,80 %	81,88 %
Variance Threshold	29	82,37 %	79,46 %	89,54 %	71,42 %	81,90 %
Deep-BP-ANN	29	83,24 %	78,94 %	98,72 %	65,76 %	82,49 %

Cuadro 5.1: Evolución de las métricas tras las técnicas aplicadas

Teniendo en cuenta la premisa de partida del proyecto, interesa identificar a clientes que van a abandonar la compañía (Verdaderos Positivos) con el mayor grado de exactitud posible, pero dado que el objetivo es llevar a cabo acciones sobre ellos de fidelización, no es especialmente importante que se incluya en ese grupo algún cliente que no corresponda (Falso Positivo). Por lo tanto, el objetivo a minimizar son los Falsos Negativos, es decir, los clientes que no identificamos como *churners* a pesar de serlo. La métrica que más interesa en esta situación es el *recall*, para el que hemos obtenido un 99 % de acierto según puede verse en la matrix de confusión del modelo:

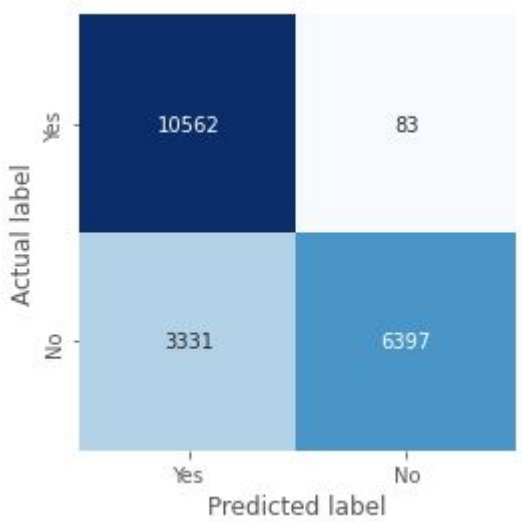

Figura 5.2: Matriz de confusión

5.2. Aplicación de técnicas de interpretabilidad

Tras la obtención de un modelo que es capaz de discriminar cuántos de los clientes de una compañía son susceptibles de abandonarla con las métricas vistas en el apartado anterior, ahora

lo que se pretende es entender por qué el modelo ha generado esas predicciones. Para ello será necesario aplicar técnicas de interpretabilidad que permitan entender qué variables y en qué medida (qué peso tienen) son consideradas a la hora de tomar una decisión.

Teniendo en cuenta que las métricas obtenidas para los modelos estudiados, *XGBoost* y *Deep-BP-ANN* han sido muy similares, y que *XGBoost* tiene unos tiempos de procesado muy inferiores, se trabajará con este último en nuestro análisis de interpretabilidad.

En cuanto a técnicas de interpretabilidad, se ha seleccionado LIME, SHAP y SMACE. Las dos primeras han sido ya suficientemente analizadas y probadas, pero esta última es de reciente publicación (julio de 2022) y se pretende confirmar que supone una mejora respecto a las dos anteriores en un conjunto de datos diferente al que se utilizó en el artículo de referencia, Lopardo et al. [2022].

5.2.1. LIME

LIME es el acrónimo de *Local Interpretable Model-agnostic Explanations*, técnica propuesta por Ribero et al. [2016]. Como su propio nombre indica, es un algoritmo agnóstico y por lo tanto aplicable a cualquier tipo de modelo. La "L"de Local, se refiere a que nos da interpretaciones de casos puntuales, respondiendo a preguntas del estilo de ¿por qué el cliente X abandona la compañía y el cliente Y no lo hace?

LIME se basa en el análisis de instancias individuales sobre el modelo de caja negra que se quiere explicar. Pretende entender por qué el modelo de aprendizaje automático ha hecho una predicción determinada. Para ello, esta técnica genera un nuevo conjunto de datos formado por muestras *fake* cercanas a la instancia en estudio y se generan las predicciones correspondientes del modelo de caja negra. Sobre este nuevo conjunto de datos, LIME entrena un modelo interpretable, ponderado por la proximidad de las instancias muestreadas a la instancia de interés. Se determina el mínimo número de variables m que genera la máxima probabilidad de acierto y esto es lo que se considera una explicación del porqué fue clasificada dicha instancia con la clase asignada. El modelo obtenido será una buena aproximación de las predicciones del modelo en estudio en un entorno local, pero no tiene porqué ser una buena aproximación global.

Se puede resumir lo anterior en la siguiente secuencia:

1. Se selecciona la instancia para la que se quiere obtener una explicación de la predicción del modelo de caja negra.

2. Se generan las muestras *fake* cercanas a la instancia y se obtienen las predicciones del modelo en análisis.

3. Se atribuyen pesos a las nuevas muestras en función de su proximidad a la instancia de interés.

4. Se entrena un nuevo modelo ponderado e interpretable en el conjunto de datos modificado.

5. Se explica la predicción interpretando el modelo local.

5.2.1.1. Resultados

Una vez visto como funciona LIME, aplicaremos la técnica a dos instancias el modelo de caja negra. Por un lado se analizará un cliente que abandona la compañía, y por otro uno que no lo hace, para ver qué diferentes criterios han influido en la decisión. Se ha limitado a 10 el número de variables a considerar para obtener las más relevantes de cara al análisis.

Se observa un cliente con una probabilidad de continuar en la compañía del 98 % (según el modelo de caja negra utilizado). En primer lugar se analiza el impacto de cada variable en la predicción (fig.5.3). De la información obtenida mediante LIME se pueden ver en azul las variables que contribuyen a que el cliente abandone la compañía, junto con el peso de cada una de ellas (*UniqueSubs* pesaría un 2 % y las cinco restantes un 1 % cada una). Por otro lado, en naranja estarían las variables que contribuyen a que el cliente continúe en la compañía (*MonthsInService* e *IncomeGroup* con un 2 % de peso y las dos restantes con un 1 %). Leyendo estos datos, ya sabemos sobre qué variables se debe actuar para conseguir que el cliente siga en la compañía.

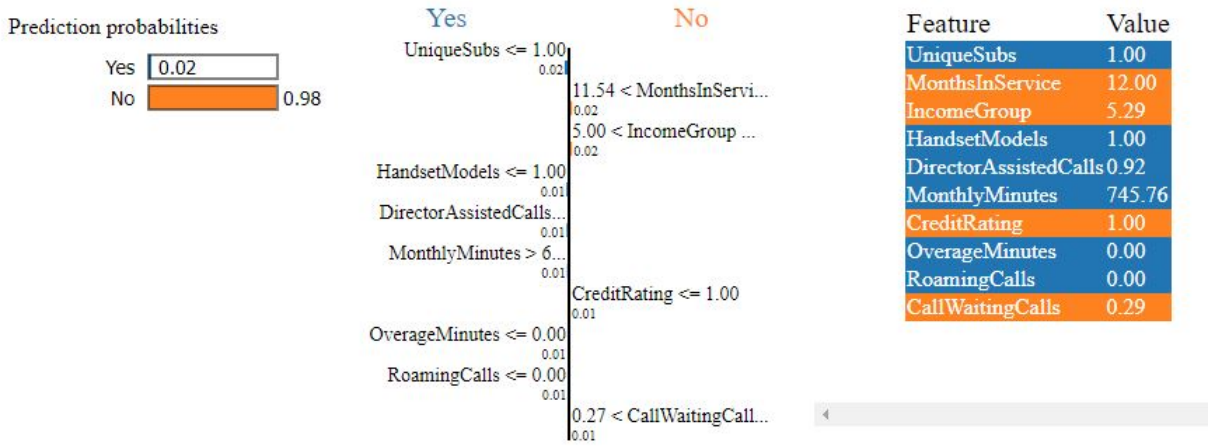

Figura 5.3: Impacto de cada variable en la predicción para Churn=No

No obstante, ya se ha comentado que LIME es un modelo **local**, lo que significa que las explicaciones que ofrece se circunscriben a un entorno cercano al de la instancia (cliente) seleccionado. Para entender el rango en el que dichas explicaciones son consistentes, se debe observar la fig.5.4 que muestra el rango de interpretabilidad local. Así, habrá que tener en cuenta los

impactos antes comentados siempre que el cliente lleve entre 11,54 y 16,78 meses en la compañía, esté en un IncomeGroup superior a 5 y menor o igual que 7, y sus minutos mensuales sean superiores a 679,29.

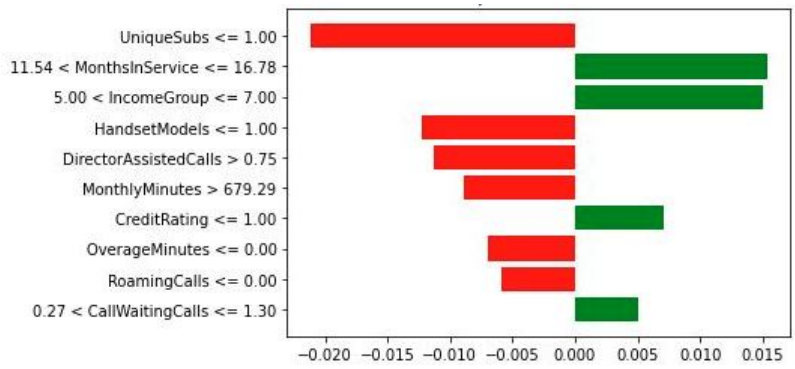

Figura 5.4: Rango de interpretabilidad local para Churn=No

Se obtiene ahora el caso contrario, un cliente para el que el modelo de caja negra afirma que va a abandonar la compañía con una probabilidad del 93 %. Se analiza en primer lugar el impacto de cada variable en la predicción (fig.5.5). En este caso las variables que contribuyen a que el cliente abandone la compañía aumentan a 8 (*UniqueSubs* e *IncomeGroup* pesarían un 2 % y las seis restantes un 1 % cada una). Por otro lado, en naranja se encuentran las variables que contribuyen a que el cliente continue la compañía (*PercChangeminutes* y *CreditRating* con un 1 % de peso).

Figura 5.5: Impacto de cada variable en la predicción para Churn=Yes

Se obtiene de nuevo el rango de interpretabilidad local (fig.5.6). Según este, los impactos antes comentados serán válidos siempre que el cliente lleve menos de 11,54 meses en la compañía, el equipo tenga menos de 219 días y el valor de PercChangeMinutes sea inferior o igual a -81,23.

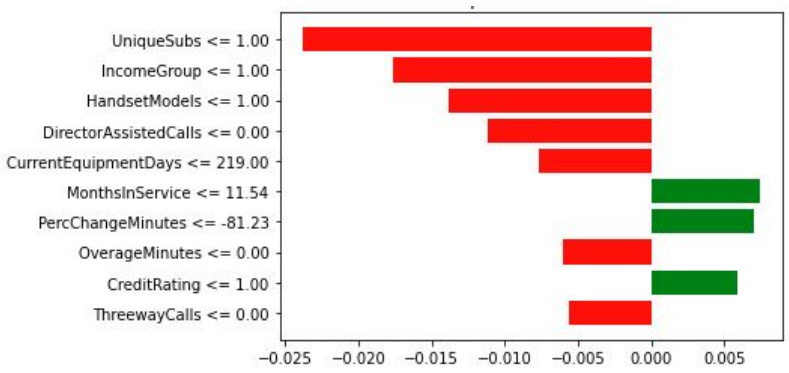

Figura 5.6: Rango de interpretabilidad local para Churn=Yes

5.2.2. SHAP

De la misma forma que LIME, SHAP es una técnica de interpretabilidad local aplicable a cualquier modelo (*model-agnostic*). Su nombre se debe a las siglas de *SHapley Additive eXplanations* y se basa en la asignación óptima de créditos usando los valores clásicos Shapley de teoría de juegos. El origen de la técnica está en el artículo de Lundberg and Lee [2017].

Lloyd Shapley, premio Nobel de economía, estudió como tratar de distribuir la ganancia obtenida por un equipo de forma de que tenga las siguientes propiedades:

1. Si un jugador no aporta ninguna ganancia, su contribución debe ser 0.
2. La suma de las contribuciones individuales tiene que ser igual a la ganancia total.
3. Consideremos dos jugadores que no pertenecen a una coalición. Si cuando se unen individualmente a ésta generan la misma ganancia, entonces deben tener las mismas contribuciones.
4. Si un juego tiene más de una ganancia, la contribución de un jugador en la ganancia total debe ser igual a la suma de las contribuciones de cada una de las ganancias del juego.

Con el fin de lograr una distribución de la ganancia que cumpla con las propiedades anteriores, Lloyd propone la fórmula siguiente:

$$\varphi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!\ (n - |S| - 1)!}{n!} (\nu(S \cup \{i\}) - \nu(S))$$

Figura 5.7: Distribución de Shapley

Esta fórmula calcula para un jugador **i** su contribución $\boldsymbol{\varphi_1}$, donde:

- **n** es la cantidad total de jugadores.
- **S** son las coaliciones que no incluyen al jugador **i**.
- $\boldsymbol{\nu}$ es la función que devuelve la ganancia de cada coalición.

Lloyd demuestra que esta distribución no solo cumple las propiedades deseadas sino que además es la única que puede cumplirlas. En su honor, a esta distribución se la llama el **valor de Shapley**.

Traduciendo a lenguaje de modelos de aprendizaje, los jugadores se corresponderían a variables, y las coaliciones a agrupaciones de variables. Teniendo esto en mente, SHAP seguiría el siguiente proceso:

1. Se selecciona la instancia para la que se quiere obtener una explicación de la predicción del modelo de caja negra (se trata de una técnica local).
2. Se generan las muestras *fake* cercanas a la instancia y se obtienen las predicciones del modelo en análisis.
3. Se ponderan las instancias muestreadas de acuerdo con el peso que obtendría la coalición en la estimación del valor de Shapley.
4. Se entrena un nuevo modelo ponderado e interpretable en el conjunto de datos modificado.
5. Se explica la predicción interpretando el modelo local.

Puede verse que la principal diferencia entre LIME y SHAP reside en la ponderación de las instancias de muestreo alrededor de la instancia en estudio: mientras uno utiliza el factor de la distancia, el otro utiliza los valores de Shapley.

5.2.2.1. Resultados

A continuación, se aplica SHAP sobre los mismos dos clientes usados con anterioridad para ver cómo explica esta técnica la predicción del modelo del estudio.

Empezando por el cliente que quiere continuar en la compañía, el resultado de aplicar SHAP en esa muestra se aprecia en las figuras 5.9 y 5.8. El valor promedio de todo el conjunto de datos es de -0.084 (valor base o E[f(X)]). Y el valor obtenido para este cliente en concreto es de 3.816 (f(x)), con las variables en rojo contribuyendo a aumentar ese valor y las variables en azul disminuyéndolo.

Figura 5.8: Gráfica lineal para Churn=NO

Se puede observar a partir de las gráficas mostradas, que el mayor impacto positivo está asociado a la variable *IncomeGroup* (valor de 5,286 y contribución de +1,19) mientras que el mayor impacto negativo lo ofrece *MontlyMinutes* con un valor de de 745,764 y una contribución de -0,18.

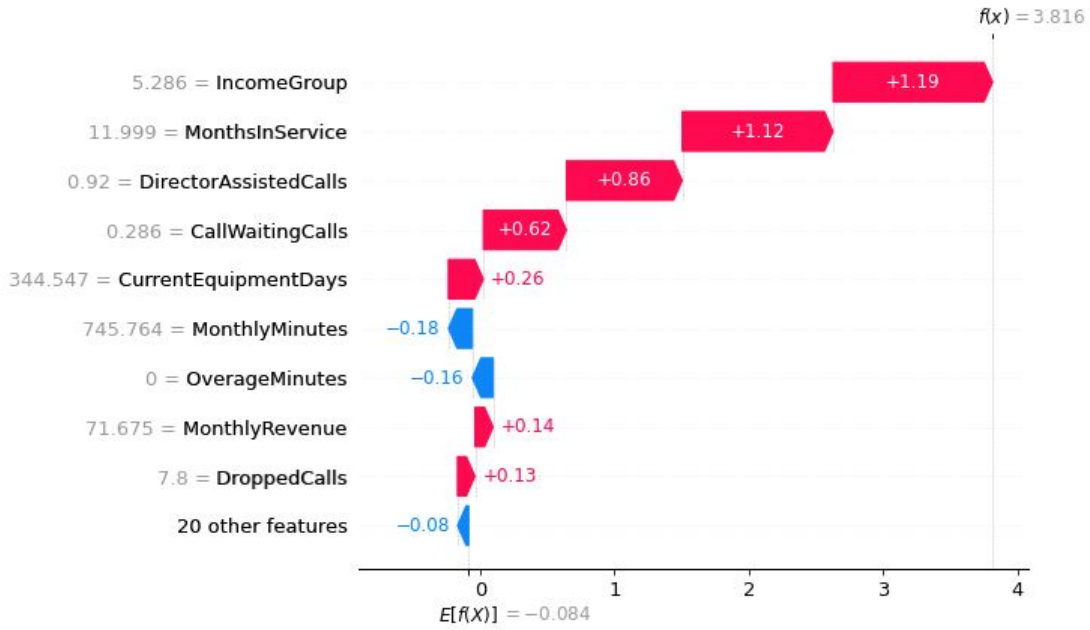

Figura 5.9: Gráfica *Waterfall* para Churn=No

Continuando con el cliente que quiere abandonar la compañía, el resultado de aplicar SHAP en esa muestra se puede ver en las figuras 5.11 y 5.10. En este caso, mientras que el valor promedio de todo el conjunto de datos es de -0.084 (valor base o E[f(X)]), el valor obtenido para este cliente en concreto es de -2.536 (f(x)).

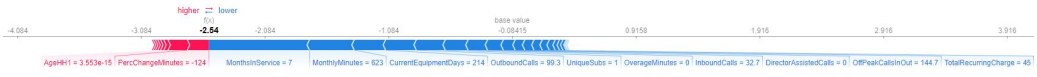

Figura 5.10: Gráfica lineal para Churn=Yes

Analizando en detalle las variables que contribuyen a dar ese valor, puede verse que el impacto negativo principal se debe a *MonthsInService* (valor de 7 y contribución de -0,81),

mientras que el impacto positivo se debe a *PerChangeminutes* con un valor de -124 y una contribución de +0,17.

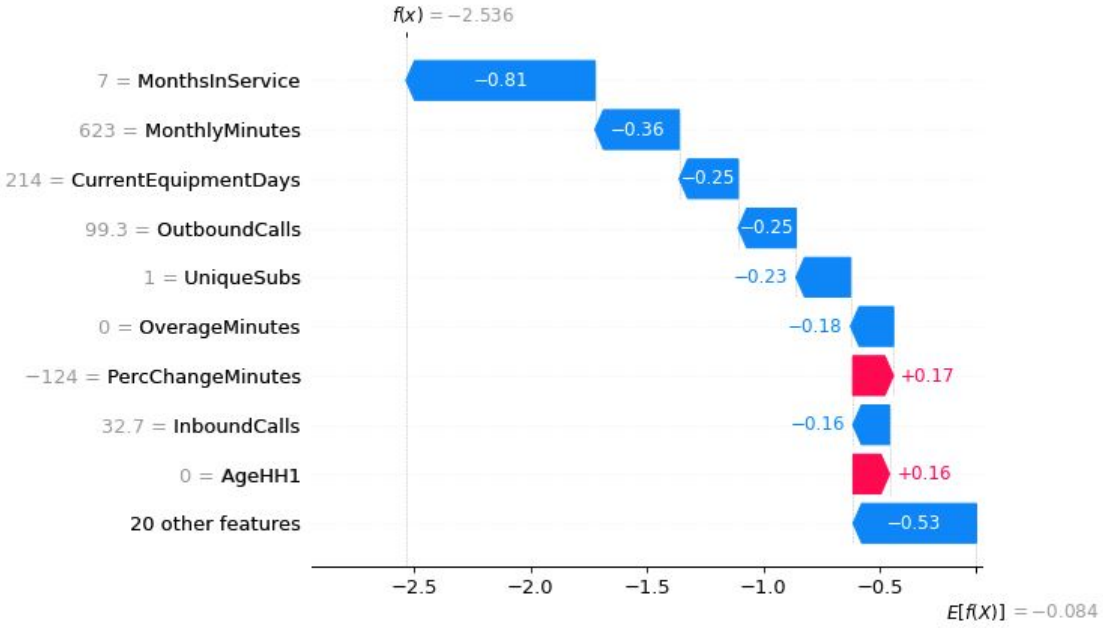

Figura 5.11: Gráfica *Waterfall* para Churn=Yes

SHAP ofrece asimismo gráficas que incluyen la totalidad de las variables y su impacto relativo en el modelo analizado. Por un lado se obtiene una gráfica de barras que lista la contribución de cada variable en promedio como se muestra en la figura 5.12:

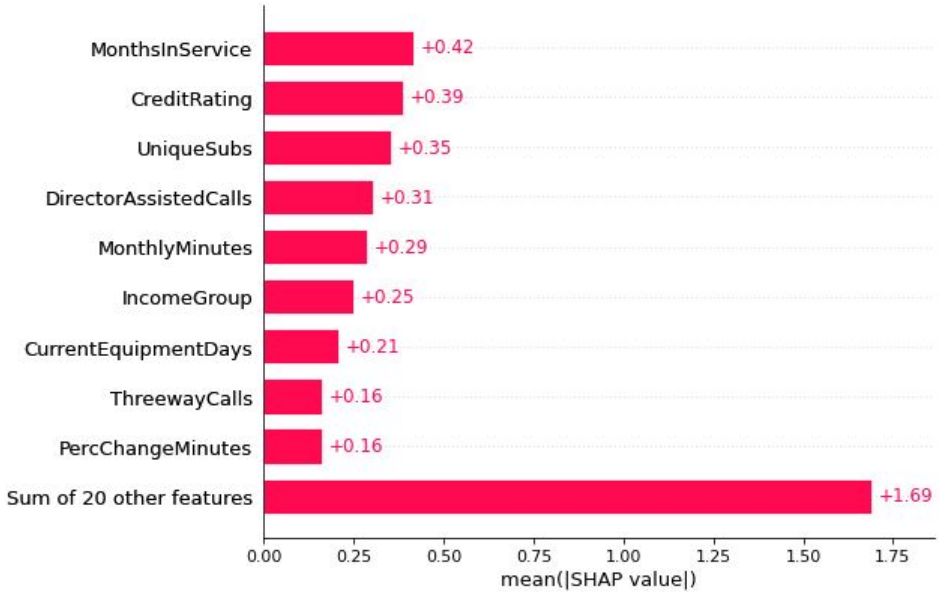

Figura 5.12: Gráfica de barras de contribución promedio de cada variable

Por otro lado, y esta gráfica es la que aporta más información, una gráfica que recopila las variables por importancia según su posición en el eje de las ordenadas, pero que además añade con colores rojo o azul las constribuciones de todas las instancias (clientes) con su valor shap.

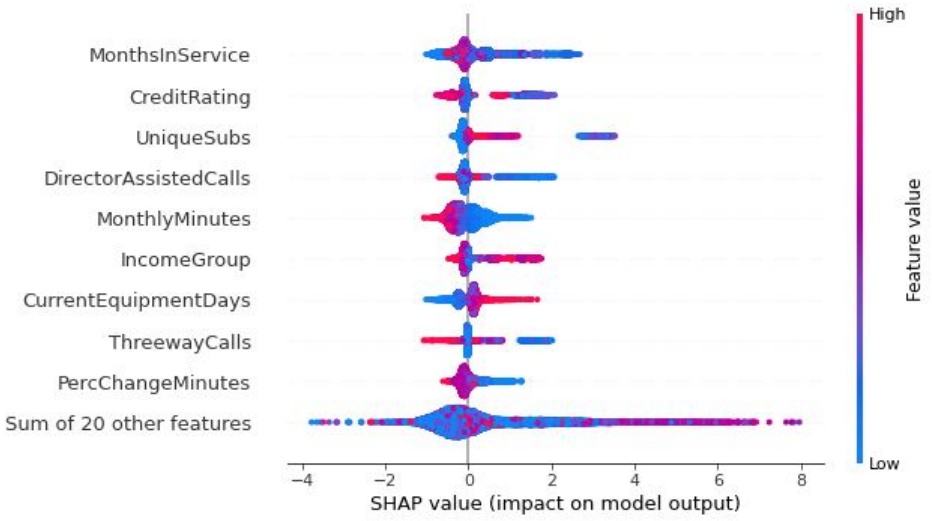

Figura 5.13: Impacto del valor SHAP en el modelo de salida

5.2.3. SMACE

SMACE es la propuesta de Lopardo et al. [2022] y sus siglas se refieren a *Semi-Model-Agnostic Contextual Model*. La idea que hay detrás de esta técnica es crear "*un nuevo método de interpretabilidad que combina un enfoque geométrico con métodos de interpretabilidad existentes para modelos de aprendizaje automático para generar una clasificación de características intuitiva y adaptada al usuario final*". Teniendo en cuenta lo anterior, el enfoque geométrico se materializa en un conjunto de reglas de decisión definidas *ad-hoc* por el usuario que pretende interpretar los resultados de su modelo de aprendizaje automático.

La definición de SMACE se basa en tres supuestos:

1. Las reglas de decisión solo aplican a valores numéricos.

2. Cada regla de decisión se refiere a una única variable, sin tener en cuenta las interacciones con otras variables.

3. Los modelos de aprendizaje automático solo trabajan con variables de entrada que provengan de un conjunto de datos, es decir, no contempla la posibilidad de aplciarla en modelos cuya entrada sea la salida de otro modelo.

Para cada instancia que se pretende explicar, SMACE (1) calcula la contribución de cada variable al resultado final, (2) analiza el impacto de las reglas de decisión sobre cada variable y (3) agrega las explicaciones parciales mediante la fórmula:

$$e_j = r_j + \sum_{m \in M} r_m \hat{\phi}_j^{(m)} .$$

Figura 5.14: Contribución total de la variable de entrada j

Es decir, se pondera la contribución de las características de entrada a cada modelo con la contribución de ese modelo en la regla de decisión, y se añade la contribución directa de la característica j a la regla de decisión (si una característica no participa directamente en una regla de decisión, su contribución es cero).

5.2.3.1. Resultados

De la misma forma que con las dos técnicas anteriores, se aplicará SMACE sobre los clientes ya vistos para poder comparar la información que ofrece con LIME y SHAP.

En primer lugar hay que definir el conjunto de reglas de decisión que se van a aplicar. De forma similar a Lopardo et al. [2022], Las reglas se definen sobre tres variables de entrada, las que han mostrado mayor relevancia en los apartados previos. Se considerarán pues:

- $x_1 >= 1$, siendo x_1 la variable *CreditRating*
- $x_2 >= 1$, siendo x_2 la variable *UniqueSubs*
- $x_3 >= 1$, siendo x_3 la variable *DirectorAssistedCalls*

Con este conjunto de reglas, lo que se pretende en enfatizar la contribución de las variables en los límites de la regla de decisión. Se añade al conjunto,además, la predicción del modelo XGBoost (cr), fijando $cr >= 0,5$.

Con los datos anteriores, y para el cliente que quiere permanecer en la compañía, los resultados que se obtienen son:

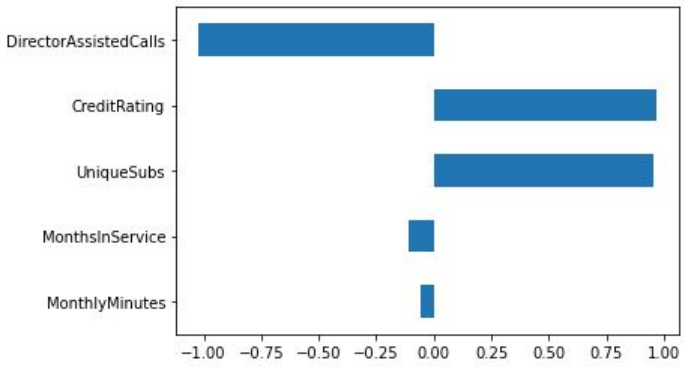

Figura 5.15: Gráfica de la contribución de cada variable a la predicción

	Example	Contribution
DirectorAssistedCalls	0.0	-1.020306
CreditRating	1.0	0.966509
UniqueSubs	1.0	0.952749
MonthsInService	7.0	-0.108364
MonthlyMinutes	623.0	-0.055392

Figura 5.16: Tabla de la contribución de cada variable a la predicción

Se puede observar el gran impacto que tienen en la predicción las variables sobre las que se ha incluido una regla de decisión, positivo en el caso de que el valor de la variable esté por encima del umbral fijado y negativo en caso contrario. Para las variables que no están dentro de las reglas de decisión, los valores obtenidos son bastante inferiores.

Para el cliente que quiere abandonar la compañía, los resultados son similares:

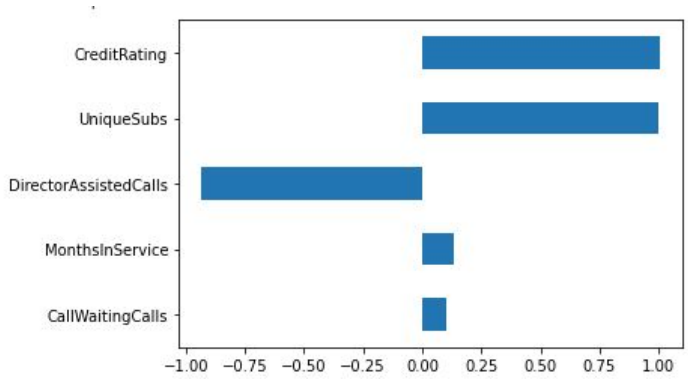

Figura 5.17: Gráfica de la contribución de cada variable a la predicción

	Example	Contribution
CreditRating	1.000000	1.003952
UniqueSubs	1.000000	1.000000
DirectorAssistedCalls	0.919646	-0.931641
MonthsInService	11.999230	0.131347
CallWaitingCalls	0.285604	0.102937

Figura 5.18: Tabla de la contribución de cada variable a la predicción

No parece que la información que se puede obtener del código propuesto sea muy relevante. Para intentar entender mejor la técnica se ha reproducido uno de los ejemplos de Lopardo et al. [2022] a partir del enlace que se proporciona en el artículo. En cuanto a los parámetros de entrada, se han utilizado los mismos que en el resto del apartado: el conjunto de datos procesado de Cell2cell en el apartado 4, el modelo XGBoost, y las reglas de decisión para SMACE ya comentadas.

A continuación se muestra la comparativa del cálculo de la contribución de cada variable incluida en las reglas de decisión para SMACE, LIME y SHAP para dos instancias concretas: una en la que los valores de la variables se encuentran cercanos umbral de decisión (5.2), y otra en las que los valores están alejados (5.3).

Variable	Condición	Valor	SMACE	SHAP	LIME
UniqueSubs	$x >= 1$	1	0,96	-0,02	-0,12
DirectorAssistedCalls	$x >= 1$	1.24	0,97	0,15	0,20
CreditRating	$x >= 1$	2	0,8	-0,02	0,07

Cuadro 5.2: Instancia con valores cercanos al umbral de decisión

En la tabla anterior se observa como SMACE presenta valores cercanos al 1 para *UniqueSubs* que está en el límite de decisión, mientras que para *CreditRating* esos valores son de 0,8 ya que está más alejado del umbral. Para las dos variables, SHAP presenta el mismo comportamiento al margen de la distancia del umbral de decisión, mientras que LIME sí que presenta algo más de variación. Teniendo en cuenta la información de esta instancia en concreto, se podría afirmar que SMACE tiene mayor sensibilidad a la distancia al umbral de decisión. Veamos qué ocurre ahora cuando se observa una instancia con valores alejados de ese umbral:

Variable	Condición	Valor	SMACE	SHAP	LIME
UniqueSubs	$x >= 1$	5	0,97	-0,02	0,1
DirectorAssistedCalls	$x >= 1$	2,23	0,98	0,16	0,19
CreditRating	$x >= 1$	3	0,6	-0,03	0,07

Cuadro 5.3: Instancia con valores alejados del umbral de decisión

En este caso, aunque la contribución de la variable *CreditRating* presenta un comportamiento consistente (disminuye su contribución a medida que se aleja del umbral), las otras dos variables presentan contribuciones muy similares a los que presentaban en la instancia anterior a pesar de estar más lejos del umbral. Lo que si podemos afirmar es que tanto SHAP como LIME no presentan variaciones significativas.

Capítulo 6

Conclusiones

Una vez finalizadas todas las tareas implicadas en el proyecto, es el momento de extraer conclusiones sobre el trabajo realizado. El objetivo planteado era entender quién y porqué abandona una compañía de servicios. Para ello, y partiendo de un conjunto de datos de clientes de una empresa de telecomunicaciones con información sobre churn, se han preparado los datos, se ha aplicado un modelo de aprendizaje supervisado y sobre este se han probado diferentes técnicas de interpretabilidad.

6.1. Procesado de los datos

En esta fase del proyecto se ha analizado el conjunto de datos realizando diferentes transformaciones sobre él para obtener la información más completa posible sin exceso de datos irrelevantes o con poco impacto para el propósito planteado. En esa línea, ha resultado especialmente útil el introducir una etapa de muestreo (véase 4.2). El primer paso fue introducir un muestreo para fijar el marco de referencia, que ya supuso una mejora destacable en las métricas. A continuación, y aunque el artículo de referencia para este apartado proponía otro enfoque, se decidió seguir la propuesta de Geiler et al. [2022]. Así, se ha utilizado en este trabajo una etapa híbrida *SMOTE* y *NCR* y el impacto en cuanto a la mejora de las métricas se puede apreciar de forma notable, como se puede ver en la tabla 6.1:

Conjunto de datos	Características	Accuracy	F1 score	Precision	Recall	AUC
Original	66	70,88 %	32,15 %	49,00 %	23,93 %	56,92 %
SMOTE y NCR	66	82,43 %	79,58 %	89,37 %	71,73 %	81,97 %

Cuadro 6.1: Conclusiones Procesado de Datos

El siguiente paso fue trabajar en la selección de características. Aquí se ha seguido la propuesta del artículo de referencia, aplicando en primer lugar *Lasso Regression* y a continuación *Variance Thresholding* (véase 4.3.2). Aunque los dos métodos empleados han permitido reducir el número de estas, es la segunda la que mayores beneficios aporta, como se puede apreciar en la tabla 6.1. En ese sentido, aplicando únicamente esta última ya se obtienen los resultados deseados.

Conjunto de datos	Características	Accuracy	F1 score	Precision	Recall	AUC
Original	66	70,88 %	32,15 %	49,00 %	23,93 %	56,92 %
Lasso Regression	64	82,33 %	79,51 %	89,07 %	71,80 %	81,88 %
Variance Threshold	29	82,37 %	79,46 %	89,54 %	71,42 %	81,90 %

Cuadro 6.2: Conclusiones Selección de Características

6.2. Modelos de aprendizaje automático

En cuanto a los modelos de aprendizaje automático supervisado, se han utilizado los propuestos por el artículo de referencia Fujo et al. [2021], *XGBoost* y *Deep-BP-ANN*. La selección de hiperparámetros ha sido notablemente más compleja en el caso de la red neuronal artificial, en gran parte debido a los tiempos de entrenamiento. En cuanto a los resultados obtenidos, las métricas de ambos resultan muy similares como puede verse en la tabla 6.3. Teniendo en cuenta tanto los tiempos de ejecución como las métricas, se puede concluir que el rendimiento del modelo *XGBoost* es superior, motivo por el cual se ha elegido para el apartado de interpretabilidad.

Modelos	Características	Accuracy	F1 score	Precision	Recall	AUC
XGBoost	29	82,37 %	79,46 %	89,54 %	71,42 %	81,90 %
Deep-BP-ANN	29	83,24 %	78,94 %	98,72 %	65,76 %	82,49 %

Cuadro 6.3: conclusiones Modelos de Aprendizaje automático

6.3. Técnicas de interpretabilidad

LIME (5.2.1) y SHAP (5.2.2) han demostrado ser dos técnicas muy útiles para profundizar en los resultados de un modelo *black box*, mientras que SMACE (5.2.3) no supone una mejora respecto a aquellas. El apartado gráfico está notablemente mejor resuelto en las dos primeras técnicas como puede apreciarse en las figuras siguientes (6.1, 6.2, 6.3) en la que se muestran los resultados para el mismo cliente. En el caso de SMACE, quizá por el poco tiempo que lleva la propuesta publicada, las gráficas incluidas en el código son muy sencillas, y aportan poca información. Además, es necesario un conocimiento adicional del conjunto de datos para definir unas reglas de decisión que resulten adecuadas. Por último, su mayor ventaja, la sensibilidad respecto a la distancia de la variable al umbral de decisión, no ha podido ser replicada consistentemente.

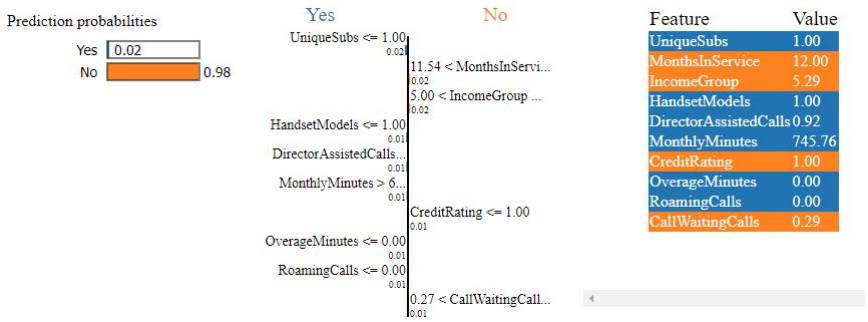

Figura 6.1: Resultados gráficos LIME

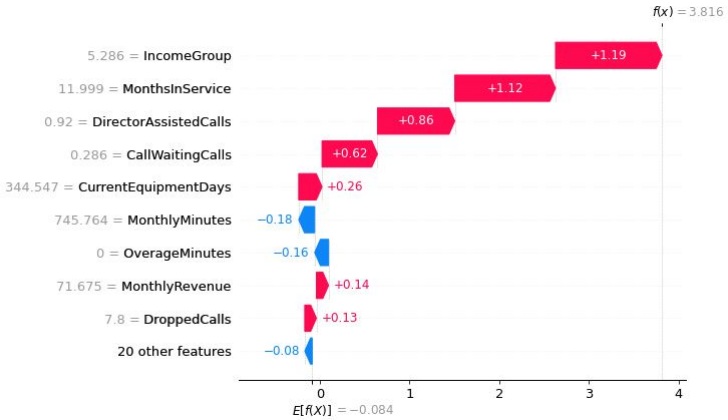

Figura 6.2: Resultados gráficos SHAP

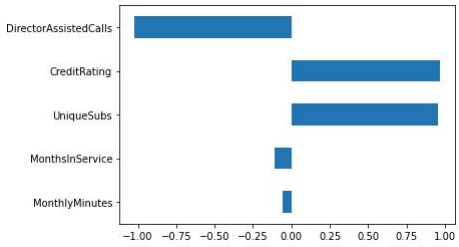

Figura 6.3: Resultados gráficos SMACE

6.4. Conclusiones finales

Se inició este proyecto pretendiendo responder a la pregunta de quién y por qué abandona una compañía de servicios.

Para la primera pregunta se propone trabajar con el conjunto de datos aplicando técnicas de muestreo híbrido para balancear los datos y aplicar *Variance Thresholding* para la selección de características. A continuación un modelo de aprendizaje supervisado XGBoost entrenado ofrece unos buenos resultados con un tiempo de ejecución muy correcto.

Para responder al porqué, la propuesta es utilizar tanto *SHAP* como *LIME* por su versatilidad a la hora de ofrecer respuestas tanto a instancias concretas como a la totalidad del modelo utilizado. La nueva técnica analizada, *SMACE*, es demasiado reciente y aún no está convenientemente desarrollada ni testada.

Capítulo 7

Próximos pasos

A la hora de enfocar los próximos pasos, se proponen dos líneas de trabajo. Una de ellas orientada a la primera parte del proyecto, la que se refiere a los modelos de aprendizaje automático; la otra a las técnicas de interpretabilidad.

En cuánto a la primera línea, sería interesante replicar el actual trabajo con otros conjuntos de datos para ver si se repiten los resultados. En la parte de preparación de los datos por las mejoras que suponen el balanceo de datos y la aplicación del *Variance Thresholding*. En la parte del modelo de aprendizaje, para confirmar el buen rendimiento del *XGBoost* frente a una *Deep-BP-ANN*.

Si nos centramos en las técnicas de interpretabilidad, la solvencia (y las limitaciones) de *LIME* y *SHAP* no necesitan de mayor incidencia, no obstante es interesante considerarlas como muestra comparativa frente a *SMACE*. Es aquí dónde se debería hacer mator incidencia: la propuesta de introducir un enfoque geométrico basado en reglas de decisión es muy interesante, pero no está suficientemente desarrollada. El artículo de referencia es muy reciente, por lo que sería necesario llevar a cabo un estudio más amplio para garantizar que los beneficios de esa técnica son consistentes. Se propone, como en la línea anterior, replicar los resultados obtenidos por Lopardo et al. [2022] con otros conjuntos de datos. En cuanto al código compartido, existen además algunos errores de carga de librerías que deberían corregirse y las opciones gráficas son muy reducidas. Sería conveniente generar un código que proporcionara mayor información.

Bibliografía

Amina Adadi and Mohammed Berrada. **Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)**. *IEEE Access*, 6:52138 – 52160, September 2018. URL https://ieeexplore.ieee.org/document/8466590.

Jaehyun Ahn, Junsik Hwang, Doyoung Kim, Doyoung Kim, Hyukgeun Choi, and Shinjin Kang. **SMACE: A New Method for the Interpretability of Composite Decision Systems**. *IEEE Access*, 8:220816 – 220839, December 2020. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9281029.

Yogesh Beeharry and Ristin Tsokizep Fokone. **Hybrid approach using machine learning algorithms for customers' churn prediction in the telecommunications industry**. *Concurrency and Computation. Practice and Experience*, 34:1 – 15, February 2022. URL https://onlinelibrary.wiley.com/doi/10.1002/cpe.6627.

Or Biran and Courtenay Cotton. **Explanation and justification in machine learning: A survey**. *IJCAI-17 Workshop on Explainable AI*, pages 8–13, 2017. URL https://www.scopus.com/record/display.uri?eid=2-s2.0-85046967682&origin=inward.

Anna Bosch, Jordi Casas, and Toni Lozano. ***Deep Learning. Principios y fundamentos***. UOC, 2019.

Nadia Burkart and Marco F. Huber. **A Survey on the Explainability of Supervised Machine Learning**. *Journal of Artificial Intelligence Research (JAIR)*, 70:245 – 317, January 2021. URL https://arxiv.org/abs/2011.07876.

Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. **Machine Learning Interpretability: A Survey on Methods and Metrics**. *Electronics*, 8:832–866, July 2019. URL https://www.mdpi.com/2079-9292/8/8/832.

Anouar Dalli. **Impact of Hyperparameters on Deep Learning Model for Customer Churn Prediction in Telecommunication Sector**. *Mathematical Problems in Enginee-*

ring, 2022:1 – 11, February 2022. URL https://www.hindawi.com/journals/mpe/2022/4720539/.

Finale Doshi-Velez and Been Kim. **Towards A Rigorous Science of Interpretable Machine Learning**. *Cornell University*, pages 1–13, March 2017. URL https://arxiv.org/abs/1702.08608.

Gabriel Marín Díaz, Ramón Antonio Carrasco González, and Daniel Gómez González. ***Interpretability Challenges in Machine Learning Models***, volume 6. Universidad Complutense de Madrid, September 2021. URL https://dialnet.unirioja.es/servlet/articulo?codigo=8036858.

Javier Jesús Espinosa-Zúñiga. **Aplicación de algoritmos Random Forest y XGBoost en una base de solicitudes de tarjetas de crédito**. *Ingeniería, investigación y tecnología*, 21(3), Jul/sept 2020. URL https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-77432020000300002.

Samah Wael Fujo, Suresh Subramanian, and Moaiad Ahmad Khder. **Customer Churn Prediction in Telecommunication Industry Using Deep Learning**. *Information Sciences Letters*, 11:185 – 198, October 2021. URL https://www.naturalspublishing.com/Article.asp?ArtcID=24434.

Louis Geiler, Séverine Affeldt, and Mohamed Nadif. **A survey on machine learning methods for churn prediction**. *International Journal of Data Science and Analytics*, 14:217 – 242, March 2022. URL https://link.springer.com/article/10.1007/s41060-022-00312-5.

Hemlata Jain, Ajay Khunteta, and Sumit Srivastava. **Telecom churn prediction and used techniques, datasets and performancemeasures: a review**. *Telecommunication Systems*, 76:613 – 630, October 2020. URL https://link.springer.com/article/10.1007/s11235-020-00727-0.

Sabarirajan Kumarappan. **Feature Selection by Lasso and Ridge Regression-Python Code Examples**. *Medium*, 2020. URL https://medium.com/@sabarirajan.kumarappan/feature-selection-by-lasso-and-ridge-regression-python-code-examples-1e8ab451b94b.

Praveen Lalwani, Manas Kumar Mishra, Jasroop Singh Chadha, and Pratyush Sethi. **Customer churn prediction system: a machine learning approach**. *Computing*, 104:271 – 294, February 2021. URL https://link.springer.com/article/10.1007/s00607-021-00908-y.

Gianluigi Lopardo, Damien Garreau, Frederic Precioso, and Greger Ottosson. **SMACE: A New Method for the Interpretability of Composite Decision Systems**. *ECML PKDD 2022, the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 9:1 – 20, July 2022. URL https://arxiv.org/abs/2111.08749.

Scott Lundberg and Su-In Lee. **A Unified Approach to Interpreting Model Predictions**. *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Nov 2017. URL https://arxiv.org/abs/1705.07874.

Israa N. Mahmood and Hasanen S. Abdullah. **Telecom Churn Prediction based on Deep Learning Approach**. *Iraqi Journal of Science*, 63:2667 – 2675, June 2022. URL https://ijs.uobaghdad.edu.iq/index.php/eijs/article/view/4280.

Madeleine McCombe. **Intro to Feature Selection Methods for Data Science. A guide to making data more manageable**. *Medium*, 2019. URL https://towardsdatascience.com/intro-to-feature-selection-methods-for-data-science-4cae2178a00a.

Seyed Mohammad Sina Mirabdolbaghi and Babak Amiri. **Model Optimization Analysis of Customer Churn Prediction Using Machine Learning Algorithms with Focus on Feature Reductions**. *Discrete Dynamics in Nature and Society*, 2022:1 – 20, June 2022. URL https://link.springer.com/article/10.1007/s11235-020-00727-0.

Christoph Molnar, Giuseppe Casalicchio, and Bernd Bischl. **Interpretable Machine Learning – A Brief History, State-of-the-Art and Challenges**. *ECML PKDD 2020 Workshops. ECML PKDD 2020. Communications in Computer and Information Science*, 1323:417 – 431, February 2021. URL https://link.springer.com/chapter/10.1007/978-3-030-65965-3_28.

W. James Murdoch, Chandan Singh, Karl Kumbiera, Reza Abbasi-Asl, and Bin Yu. **Definitions, methods, and applications in interpretable machine learning**. *PNAS*, 116:22071–22080, October 2019. URL http://portal.acm.org/citation.cfm?id=774544.774552.

Risuna Nkolele and Hairong Wang. **Explainable Machine Learning: A Manuscript on the Customer Churn in the Telecommunications Industry**. *2021 Ethics and Explainability for Responsible Data Science (EE-RDS)*, pages 1 – 7, February 2021. URL https://ieeexplore.ieee.org/document/9708561.

Vasilis Papastefanopoulos Pantelis Linardatos and Sotiris Kotsiantis. **Explainable AI: A Review of Machine Learning Interpretability Methods**. *Entropy*, 23(1):1–45, December 2020. URL https://www.mdpi.com/1099-4300/23/1/18.

Marco Tulio Ribero, Sameer Singh, and Carlos Guestrin. **"Why Should I Trust You?"Explaining the Predictions of Any Classifier**. *KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1146, Aug 2016. URL https://dl.acm.org/doi/10.1145/2939672.2939778.