Trabajo Fin de Máster

Detección de anomalías en operatoria bancaria mediante inteligencia artificial

Aida Vázquez Pardo

Máster en Técnicas Estadísticas

Curso 2021-2022

Propuesta de Trabajo Fin de Máster

Título en galego: Detección de anomalías en operatoria bancaria mediante intelixencia artificial

Título en español: Detección de anomalías en operatoria bancaria mediante inteligencia artificial

English title: Anomaly detection in banking operations through artificial inteligence

Modalidad: Modalidad B

Autora: Aida Vázquez Pardo, Universidad de Santiago de Compostela

Directores: Rubén Fernández Casal, Universidad de Coruña; Guillermo López Taboada, Universidad de Coruña

Tutores: Jose Jorge García Romarís, ABANCA; Beatriz Gantes Crespo, ABANCA

Breve resumen del trabajo:
En este trabajo de fin de máster, se afronta la identificación de problemas en diferentes canales corporativos de una entidad financiera. El objetivo principal es la definición de un modelo predictivo en operaciones que permita detectar anomalías en una serie temporal dada, para la identificación de dichos problemas. Para definir este modelo se realiza, en primer lugar un análisis exploratorio de los datos, con la finalidad de estudiar la naturaleza de la serie temporal. Posteriormente, se aplican diferentes métodos estadísticos de predicción en series temporales y técnicas de Inteligencia Artificial y se comparan empleando diferentes métricas de evaluación, para la elección del mejor modelo. Cabe destacar que, para la realización de este trabajo, se emplea el lenguaje de programación Python.

Recomendaciones:
Conocimientos de técnicas de Deep Learning, Machine Learning e Inteligencia Artificial.

Otras observaciones:

Don Rubén Fernández Casal, Profesor contratado doctor de la Universidad de Coruña, Don Guillermo López Taboada, Titular de universidad de la Universidad de Coruña, y Don Jose Jorge García Romarís, Director del departamento Data, Analytics & Rtech de ABANCA, y Doña Beatriz Gantes Crespo, Responsable del departamento de Monitorización y Disponibilidad (área de gobierno y operaciones TI) de ABANCA, informan que el Trabajo Fin de Máster titulado

Detección de anomalías en operatoria bancaria mediante inteligencia artificial

fue realizado bajo su dirección por Doña Aida Vázquez Pardo para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal.

En A Coruña, a 13 de Junio de 2022.

FERNANDEZ CASAL RUBEN - 33317724Q
Firmado digitalmente por FERNANDEZ CASAL RUBEN - 33317724Q
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-33317724Q, givenName=RUBEN, sn=FERNANDEZ CASAL, cn=FERNANDEZ CASAL RUBEN - 33317724Q
Fecha: 2022.06.13 08:09:52 +02'00'

El director:
Don Rubén Fernández Casal

LOPEZ TABOADA GUILLERMO - 44088774M
Firmado digitalmente por LOPEZ TABOADA GUILLERMO - 44088774M
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-44088774M, givenName=GUILLERMO, sn=LOPEZ TABOADA, cn=LOPEZ TABOADA GUILLERMO - 44088774M
Fecha: 2022.06.13 11:01:12 +02'00'

El director:
Don Guillermo López Taboada

El tutor:
Don Jose Jorge García Romarís

La tutora:
Doña Beatriz Gantes Crespo

La autora:
Doña Aida Vázquez Pardo

Declaración responsable. Para dar cumplimiento a la Ley 3/2022, de 24 de febrero, de convivencia universitaria, referente al plagio en el Trabajo Fin de Máster (Artículo 11, Disposición 2978 del BOE núm. 48 de 2022), **la autora declara** que el Trabajo Fin de Máster presentado es un documento original en el que se han tenido en cuenta las siguientes consideraciones relativas al uso de material de apoyo desarrollado por otros/as autores/as:

- Todas las fuentes usadas para la elaboración de este trabajo han sido citadas convenientemente (libros, artículos, apuntes de profesorado, páginas web, programas,...)
- Cualquier contenido copiado o traducido textualmente se ha puesto entre comillas, citando su procedencia.
- Se ha hecho constar explícitamente cuando un capítulo, sección, demostración,...sea una adaptación casi literal de alguna fuente existente.

Y, acepta que, si se demostrara lo contrario, se le apliquen las medidas disciplinarias que correspondan.

"Solo podemos ver poco del futuro, pero lo suficiente para darnos cuenta de que hay mucho que hacer."
Alan Turin (s. XX)

Agradecimientos

En primer lugar, me gustaría agradecer a mis tutores de empresa, Jose Jorge García Romarís y Beatriz Gantes Crespo, a la entidad financiera ABANCA y a la Facultad de Matemáticas de la Universidad de Santiago de Compostela por ofrecerme la oportunidad de realizar el trabajo de fin de máster en esta empresa pero, sobre todo, por ofrecerme la oportunidad de formarme como matemática. También me gustaría agradecer a David Barrientos Guillen por ofrecerme su apoyo, ayuda y consejos a la hora de realizar mi trabajo de fin de máster. Desde que empecé las prácticas, tanto él como Jorge, siempre han estado ahí para ofrecerme el material necesario y resolver cualquier problema a la hora de realizar el trabajo y se han preocupado por la evolución del mismo en todo momento.

Me gustaría agradecer, en especial, al coordinador de máster y padrino de mi promoción Julio González Díaz por animarme a escoger este trabajo de fin de máster. Mencionar también a Alejandro Arias Piñeiro, mi compañero de prácticas durante estos 5 meses. Coincidimos como compañeros de facultad, nos conocíamos de vista y el trabajo de fin de máster hizo que nos reencontrásemos y compartiésemos experiencia. Agradecer también a mis tutores académicos del trabajo, Rubén Fernández Casal y Guillermo López Taboada. Por su paciencia, su apoyo y los consejos ofrecidos para la realización de este trabajo.

Finalmente me gustaría agradecer a mis padres, Roberto Vázquez Cacabelos y Mª Oliva Pardo Puente y a mi hermana Paula Vázquez Pardo, por ser un apoyo fundamental en mis estudios y sobre todo por animarme a hacer lo que más me gusta, siempre con una muestra de cariño y de orgullo. También agradecer a mis compañeros de máster Ángel Martínez González y Sergio Mejuto Vázquez por el apoyo recibido y por todos los momentos compartidos en la carrera y en el máster. Y por supuesto, agradecer a mi compañero de la facultad y uno de mis mejores amigos Manuel Vázquez Mourazos, por estar ahí siempre para cualquier cosa, ofreciendo su ayuda en todo lo posible y, sobre todo, por ser un pilar fundamental en mi vida. Sin mi familia y mis amigos esto no sería posible, siendo un pilar fundamental en mi formación como matemática y como persona.

Índice general

Resumen

Resumen en español

Las técnicas de detección de anomalías han cobrado especial importancia debido a su diversidad de aplicaciones. Se trata de un problema cuyo objetivo es localizar patrones y comportamientos extraños en un conjunto de datos. En este trabajo de fin de máster, se afronta la identificación de problemas en diferentes canales corporativos de una entidad financiera. El objetivo principal es la definición de un modelo predictivo en operaciones que permita detectar anomalías en una serie temporal dada, para la identificación de dichos problemas. Para definir este modelo se realiza, en primer lugar un análisis exploratorio de los datos, con la finalidad de estudiar la naturaleza de la serie temporal, determinando si presenta componentes estacionales (que pueden ser anual, semanal o diaria). Posteriormente, se aplican diferentes métodos estadísticos de predicción en series temporales y técnicas de Aprendizaje Estadístico, entre las que destacan los modelos de redes neuronales de memoria a corto plazo, LSTM. Todas las técnicas empleadas se comparan empleando diferentes métricas de evaluación, para la obtención del mejor modelo. Cabe destacar que, para la realización de este trabajo, se emplea el lenguaje de programación Python.

English abstract

Anomaly detection techniques have become especially important due to their diversity of applications. It is a problem whose objective is to locate strange patterns and behaviors in a data set. In this master's thesis, the identification of problems in different corporate channels of a financial institution is addressed. The main objective is the definition of a predictive model in operations that allow detecting anomalies in a given time series, for the identification of said problems. To define this model, first an exploratory analysis of the data is carried out, in order to study the nature of the time series, determining whether it has seasonal components (which can be annual, weekly or daily). Subsequently, different statistical prediction methods are applied in time series and Statistical Learning techniques, among which the long short-term memory neural network models stand out, LSTM. All the techniques used will be compared using different evaluation metrics, to obtain the best model. It should be noted that, to carry out this work, the Python programming language is used.

Capítulo 1

Introducción

La detección de anomalías en series temporales es un problema muy importante en la actualidad. Es una rama de la estadística cuya finalidad reside en obtener patrones y comportamientos extraños en un conjunto de datos dependientes del tiempo. Este problema puede ser desafiante, en el sentido de que no hay una definición clara de anomalía, puesto que ésta depende del conjunto de datos a emplear y de la aplicación de los mismos. En este trabajo de fin de máster nos centraremos en resolver un problema de detección de anomalías en operatoria bancaria.

Para empezar, en el primer capítulo se realiza una descripción detallada del problema de detección de anomalías planteado por la entidad financiera ABANCA. Principalmente, se define el problema de detección de anomalías en operatoria bancaria, se comentan los objetivos del mismo y se ilustra un ejemplo para detectar los fallos en los terminales de punto de venta, TPV. Este ejemplo, proporcionado por la entidad financiera, servirá de referencia para la comprensión del problema. En las siguientes secciones se definen los diferentes tipos de anomalías en operatoria bancaria y los diferentes aspectos a tener en cuenta en la resolución del problema. Finalmente, se describen detalladamente las métricas de evaluación, necesarias para la obtención del mejor modelo.

En el siguiente capítulo se enuncian los conceptos y la fundamentación teórica necesaria para la realización de este trabajo. Se introducen diversos conceptos en series de tiempo, entre ellos, las componentes estacionales y tendencia y se explican detalladamente los diferentes métodos de detección de anomalías, junto a los modelos de regresión polinómica local y la validación cruzada.

En el capítulo 4 se realiza un análisis exploratorio de datos para determinar cómo se comporta la serie de tiempo del número de operaciones con la que se va a trabajar. En primer lugar, se realizan gráficos para diagnosticar la evolución del número de operaciones. El siguiente paso es representar la serie temporal en períodos de un mes, una semana y un día para determinar si existe algún tipo de componente estacional, que puede ser diaria, semanal, mensual o anual. Tras determinar las componentes estacionales, se realiza la representación gráfica de las componentes estacionales desagregadas, entre las que se incluye la componente tendencia y la componente residual de la serie temporal, además de las componentes estacionales (si existen). El principal objetivo de este gráfico es ver si la serie temporal posee tendencia además de ver cómo se comportan las componentes estacionales y la componente residual de la serie temporal. Finalmente se realizan gráficos funcionales para cada día de la semana con la finalidad de determinar si la distribución del número de operaciones realizadas varía con respecto al día de la semana.

En el próximo capítulo se presentan los resultados del problema de detección de anomalías empleando métodos autoexplicativos como los modelos TBATS (*Trigonometric seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal components*), el modelo UCM (*Unobserved Component*

Model) y el modelo Prophet. Dado la complejidad computacional del modelo TBATS, en lugar de ajustar el modelo empleando el conjunto de datos completo, se ajusta el modelo para un conjunto de datos reducido y se calculan las predicciones en períodos de de 1 día, repitiendo el proceso las veces que se considere necesario. En cuanto a los modelos UCM y Prophet se realiza el ajuste con todos los datos de los años 2020 y 2021 y se predicen los valoes de los 3 primeros meses del año 2022. Una vez que se han obtenido las predicciones, se comparan gráficamente con los valores reales y se realizan gráficos del MAE y del RMSE a lo largo del día. Se comprobará que este tipo de métodos no son muy eficientes para la detección de anomalías en operatoria bancaria.

En el capítulo 6 se presentan los resultados del problema de detección de anomalías empleando modelos de Aprendizaje Estadístico como las redes neuronales LSTM y los modelos ensamblados (*Ensemble models*). Para realizar el ajuste de estos modelos en lugar de considerar únicamente la serie temporal del número de operaciones, se añaden variables adiccionales, entre las que se incluyen la hora del día, el día del año, el día de la semana y una variable categórica indicadora de si el día es laborable o no. Se consideran como días no laborables los fines de semana y los festivos a nivel nacional, autonómico y local (ciudad de A Coruña). La lista completa de festivos puede consultarse en el Apéndice A. Además, a petición de la empresa, dado que la serie temporal es dependiente del tiempo, en el ajuste de estos modelos y la obtención de las predicciones se tendrán en cuenta los valores correspondientes a los 30 minutos anteriores (lags de 30 minutos), que deben incluirse en el conjunto de datos con el que se va a trabajar.

Finalmente, en el capítulo 7, se resuelve un problema de clasificación supervisada para la detección de anomalías. Para ello, en primer lugar, se emplea un método de regresión no paramétrica como la regresión polinómica local robusta, LOWESS, para realizar una clasificación de las anomalías. Una vez obtenido un conjunto de datos clasificado, se busca el mejor modelo de clasificación supervisada empleando la búsqueda automática implementada la intefaz H2oAutoMl de Python, se calculan las predicciones y se determina cómo evoluciona la probabilidad de anomalía a lo largo del día para un día en el que conozcamos la existencia de una anomalía.

Por mi parte, espero que la lectura de este trabajo resulte satisfactoria. Con el deseo transmitir la importancia de emplear métodos de Aprendizaje Estadístico para la comprensión del futuro mediante el cálculo de predicciones y mostrar una de sus principales aplicaciones en la banca, la detección de problemas en las herramientas empleadas por los gestores en las oficinas bancarias. Espero que su lectura transmita la enorme curiosidad que he sentido a la hora de elegir este trabajo y, sobre todo, comprender cómo funciona una entidad bancaria por detrás de las oficinas.

Capítulo 2

El problema de detección de anomalías

En este capítulo se realiza una descripción detallada del problema de detección de anomalías planteado por la entidad financiera de ABANCA. En primer lugar, se define el problema y se comentan los objetivos del mismo. Después se realiza una descripción detallada de los distintos tipos de anomalías y se comentan los diferentes aspectos a tener en cuenta a la hora de resolver el problema. Para finalizar, se da una descripción de las métricas de evaluación empleadas para la selección de los modelos.

2.1. Descripción del problema y objetivos

Desde la entidad financiera ABANCA se plantea la resolución de un problema de detección de anomalías en una serie de tiempo con datos reales, a nivel de minutos, del número de operaciones realizadas, en un período concreto determinado por la empresa, en los siguientes canales corporativos: ordenador central (oficinas), Banca Electrónica y Banca Móvil. La finalidad de este problema reside en detectar, de forma ágil, si se produce algún tipo de problema en las herramientas que emplean los gestores en las oficinas bancarias es decir, se trata obtener aquellos períodos de tiempo en los que se producen cambios significativos en el número de operaciones con respecto a los valores habituales, teniendo en cuenta características como la hora del día, el día del año, el día de la semana o si el día es laborable.

La pregunta que se puede plantear el lector es la siguiente: ¿cómo se pueden detectar estos problemas de operatoria bancaria? La detección de anomalías se realiza normalmente mediante el ajuste de diferentes modelos predictivos a tiempo real de una serie temporal a nivel de minutos para el cálculo de las predicciones en un período concreto de tiempo, que deben compararse con los valores reales, calculando la diferencia. Las anomalías serán aquellas observaciones cuya diferencia entre las predicciones y los valores reales supere o se encuentre por debajo de un umbral, que suele establecerse como ± 5 veces el Error Medio Absoluto (MAE) de entrenamiento cada 5 minutos del modelo predictivo ajustado[1]. En el caso de la entidad financiera ABANCA, los modelos predictivos ajustados se preparan para el departamento de Monitorización y Disponibilidad, del área de Gobierno y operaciones TI. Este departamento es el encargado de realizar diferentes gráficos comparativos para la detección de las anomalías mediante el uso de herramientas de monitorización como la aplicación llamada Zabbix.

Para la comprensión del lector acerca de este problema, se ilustra un ejemplo. En la entidad financiera ABANCA tienen desarrollado un modelo de detección de anomalías para detectar si existe algún tipo de problema con los Terminales de punto de Venta, comúnmente conocidos como TPV.

[1]El Error Medio Absoluto (MAE) de entrenamiento en períodos de 5 minutos se emplea como una medida robusta de la varianza del error.

Desde el departamento de monitorización y disponibilidad, se han facilitado los siguientes gráficos que comparan los valores del número de operaciones real con las predicciones del modelo, la diferencia y los límites del ±MAE y ±5 MAE, para la detección de anomalías. El objetivo principal del TFM, será realizar gráficos similares para las anomalías detectadas en el caso de las operaciones realizadas en las oficinas, en Banca Móvil y en Banca Electrónica, empleando diferentes modelos predictivos.

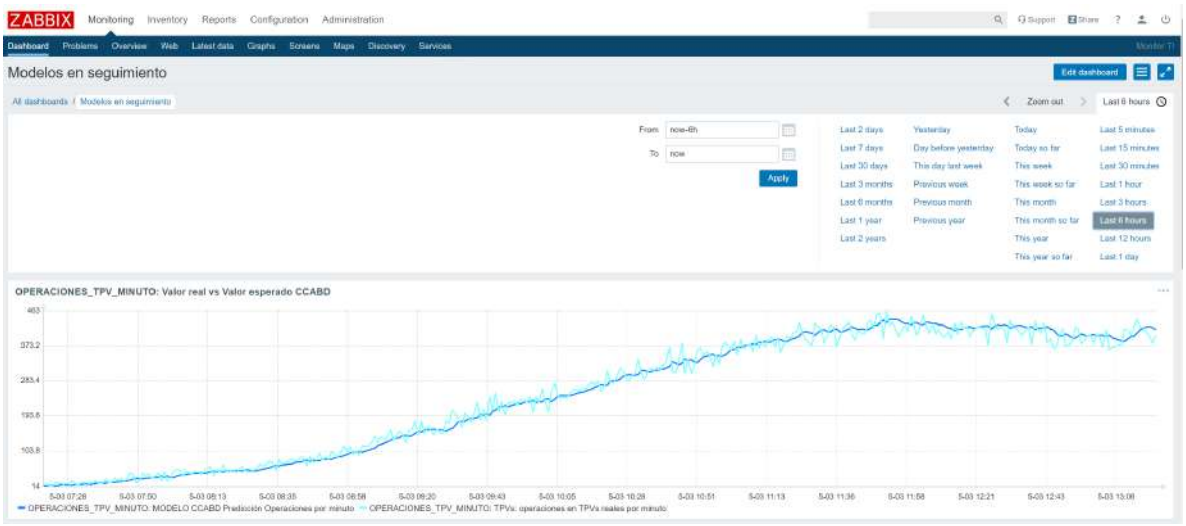


Figura 2.1: Ejemplo de gráfico de la herramienta Zabbix que compara los valores reales del número de operaciones en TPV (en azul claro) con las predicciones obtenidas mediante un modelo de regresión, en particular un modelo de redes neuronales LSTM (en azul oscuro). Se representan las operaciones el día 3 de Mayo en horario de 7:30 a 13:30.

Figura 2.2: Ejemplo de gráfico para la detección de anomalías en un modelo de TPV. En la parte superior se representa la diferencia entre las predicciones del modelo y las operaciones reales realizadas el día 3 de Mayo de 7:30 a 13:30. En la parte inferior se representa esta diferencia junto a los límites del MAE y de 5 veces el MAE. Como se puede apreciar, la diferencia entre las predicciones del modelo y las operaciones reales realizadas el día 3 de Mayo de 7:30 a 13:30 se encuentra dentro de los límites del MAE y de 5 veces el MAE, lo que indica que no se produce ningún fallo en los TPV.

2.2. Tipos de anomalías

En un problema de detección de anomalías en operatoria bancaria podemos encontrarnos los siguientes tipos de anomalías:

- **Anomalías contextuales**. Son aquellos valores de la serie temporal que toman un comportamiento diferente al resto pero que dependen de ciertas características específicas del conjunto de datos como por ejemplo, la fecha o la localización geográfica. A modo de ejemplo, consideremos el caso del problema de detección de anomalías en los Terminales de Punto de Venta (TPV). Una anomalía contextual que depende de la localización geográfica, dado que los TPV están conectados a internet, un fallo podría ser causado por una caída en el internet de cierta compañía telefónica (ejemplo, la compañía gallega R) a nivel de la provincia de A Coruña.

- **Anomalías colectivas**. Ocurre cuando un subconjunto de puntos de la serie temporal se desvía significativamente del resto de los datos, pero los datos individuales no se consideran anomalías. A modo de ejemplo, consideremos el problema de detección de anomalías en los TPV. Es evidente que en los meses del confinamiento por covid19 (Marzo, Abril y Mayo del año 2020) hubo una reducción notable del número de operaciones en los TPV, ya que sólo se empleaban en ciertos lugares como supermercados o farmacias. Este tipo de observaciones pueden considerarse anomalías colectivas ya que, en condiciones normales, el uso de TPV podría ser más reducido pero no limitado a ciertos establecimientos.

2.3. Aspectos a tener en cuenta en la resolución del problema

Para resolver el problema de detección de anomalías en operatoria bancaria debemos tener en cuenta los siguientes aspectos: la naturaleza de los datos y el etiquetado de los datos.

Con respecto a la naturaleza de los datos debemos tener en cuenta las características, los tipos de variables (categóricas, numéricas) y el tamaño de los mismos para determinar qué métodos podemos aplicar. Por ejemplo, en nuestro caso, los datos de los que disponemos son de variables numéricas indicadoras del número de operaciones realizadas por minuto en un período concreto de tiempo (por ejemplo 2 años, tendríamos un total de 1052640 observaciones). Al tener muchos datos es posible que los métodos estadísticos clásicos de predicción en series de tiempo no nos resulten de utilidad, debido a los tiempos del ajuste de los modelos. Por este motivo, en caso de que se quisiera ajustar alguno de los métodos clásicos, como un ARIMA, se podría reducir el conjunto de datos agrupando los datos por horas o por días. Además, como tenemos disponibilidad de las fechas, podríamos considerar variables adicionales como la hora del día, el día del año, el día de la semana, el mes o si es un día laborable.

Con respecto al etiquetado de los datos comentar que en la mayoría de las situaciones reales no se dispone de un etiquetado de los datos en anomalías. El propio investigador podría realizar un etiquetado de las mismas, pero esto requiere un esfuerzo muy grande, sobre todo en los conjuntos de datos masivos. Esta característica es la que nos determina el tipo de clasificación a utilizar (supervisada o no supervisada):

- Cuando se dispone de un etiquetado de los datos, se aplican métodos de **clasificación supervisada** entre los que se incluyen la regresión lineal generalizada, los árboles de decisión y las máquinas de soporte vectorial (SVM).

- Cuando no se dispone de un etiquetado de los datos, se aplican métodos de **clasificación no supervisada**. Entre ellos se incluyen los modelos de redes neuronales, el algoritmo de K medias, el Isolation Forest y los métodos basados en modelos de regresión como los modelos ARIMA, entre otros. Este tipo de métodos son los que se emplean en mayor medida en la actualidad puesto que, en la mayoría de situaciones reales, no se conocen con anterioridad las anomalías.

Según la clasificación empleada distinguiremos tres tipos de problemas de detección de anomalías:

- **Problema de detección de anomalías empleando métodos autoexplicativos**. Dada la serie temporal del número de operaciones se ajusta un modelo de de clasificación no supervisada autoexplicativo, es decir, en el ajuste de este tipo de modelos no se consideran variables auxiliares, simplemente se emplea el número de operaciones realizadas en un período concreto de tiempo. Los modelos autoexplicativos que se considerarán son los modelos TBATS (*Trigonometric seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal components*), UCM (*Unobserved Component Model*) y Prophet. Se comprobará que este tipo de métodos no son adecuados para la detección de anomalías en operatoria bancaria.

- **Problema de detección de anomalías empleando métodos con predictores externos (métodos de Aprendizaje Estadístico)**. A partir de una serie temporal con el número de operaciones, se calculan las predicciones empleando los 30 minutos anteriores (lags de 30 minutos) y considerando variables adicionales como la hora del día, el día del año, el día de la semana y una variable indicadora de si tenemos día laborable o no. Consideraremos como días no laborables los fines de semana y los festivos nacionales, autonómicos y locales (ciudad de A Coruña). La lista completa de festivos puede encontrarse en el Apéndice A. Para resolver este tipo de problemas se emplean modelos de regresión como las redes neuronales LSTM. Además, se puede emplear la interfaz H2oAutoml de Python para realizar el ajuste de los siguientes modelos y escoger el mejor entre ellos: modelos lineales generalizados (GLM), Gradient Boosting Machine (GBM), RandomForest, XGBoost y métodos ensamblados (*ensemble methods*)[2].

- **Problema de clasificación supervisada**. En los datos del número de operaciones con el que se va a trabajar no tenemos un etiquetado de los mismos en anomalías. Desde el departamento de Monitorzación y Disponibilidad se proporciona un etiquetado en función de las incidencias masivas, pero no es del todo fiable por lo que no se puede aplicar clasificación supervisada de forma directa. Entonces, a partir del ajuste de un modelo de regresión polinómica local robusta y los intervalos de confianza al 99 %, se puede obtener una clasificación de las anomalías que permita resolver un problema de clasificación supervisada. La interfaz H2oAutoMl de Python en la que se incluyen diferentes tipos de modelos, entre ellos los métodos ensamblados, nos proporcionará el mejor modelo de clasificación supervisada para resolución del problema. Cabe destacar que, en el ajuste de los modelos de clasificación, al igual que en los modelos a tiempo real, se consideran variables adicionales como la hora del día, el día del año, el día de la semana y una variable indicadora de si es día laborable o no. Además, para modelizar la dependencia del tiempo, en lugar de considerar los lags de 30 minutos se dividen los lags entre el número de operaciones real.

En los problemas de clasificación supervisada y en los problemas empleando métodos de Aprendizaje Estadístico se emplea una muestra de entrenamiento y una muestra de validación para realizar el ajuste de los modelos y escoger el mejor entre ellos para, posteriormente, calcular las predicciones del modelo evaluando sobre una muestra de test. En el caso de los modelos incluidos en la interfaz H2oAutoMl, en el ajuste se emplea validación cruzada [3]. Tras añadir las variables adicionales y modelizar la dependencia del tiempo (empleando los lags y la proporción de lags, respectivamente), los datos del número de operaciones realizadas en los años 2020 y 2021 se dividirán, de forma aleatoria, en muestras de entrenamiento y de validación (correspondientes con el 90 % y el 10 % de los datos, respectivamente), fijando una semilla para garantizar reproducibilidad. Como muestra de test se consideran los 3 primeros meses del año 2022.

[2]Los métodos ensamblados son una combinación de diferentes modelos de Aprendizaje Estadístico cuyas predicciones se obtienen combinando las predicciones obtenidas en los diferentes modelos considerados en la interfaz H2oAutoMl de Python.

[3]El procedimiento de validación cruzada se describe en el capítulo de cuestiones y fundamentos teóricos

Independientemente del problema considerado, los modelos finales deben seleccionarse en función de las métricas de evaluación, que se describen detalladamente en la siguiente sección, siguiendo como referencia Chicco et al. (2021), Tharwat (2020) y Fernández-Casal et al. (2021).

2.4. Métricas de evaluación

En esta sección, se comentan las diferentes métricas de evaluación, necesarias para la comparación de los diferentes modelos, con la finalidad de escoger el mejor de ellos. Según el tipo de clasificación empleada, distinguimos las siguientes métricas:

- En caso de que el criterio de detección esté basado en un método de regresión, se emplean el **Error Medio Absoluto (MAE)** y el **Error Cuadrático Medio (MSE)**.
- Si el método de predicción empleado es un modelo de clasificación supervisada, se suele emplear la **curva ROC** y **el área bajo la curva (AUC)**. En el caso de que la muestra sea muy desbalanceda, como criterio de comparación, se usa el **F1 Score**, la **curva Precision Recall** y **el área bajo la curva Precision-Recall (AUCPR)**.

2.4.1. Error medio absoluto (MAE)

Una métrica de evaluación clásica que se emplea en la comparación de modelos de clasificación no supervisada es el **Error Medio Absoluto**, denotado normalmente como MAE, por su abreviatura en inglés (*Mean Absolute Error*). Es una medida de error que se calcula como una media aritmética del valor absoluto de la diferencia entre las predicciones y los valores reales. Su expresión matemática es la siguiente:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |Y_i - \widehat{Y}(X_i)|$$

siendo X_i, $i = 1, \cdots, n$ las variables explicativas, $\widehat{Y}(X_i), i = 1, \cdots, n$ las predicciones del modelo, $Y_i, i = 1, \cdots, n$ los valores reales del conjunto de datos y n el número total de observaciones.

2.4.2. Error cuadrático medio (MSE)

Otra de las medidas clásicas que se usa en la comparación de modelos de clasificación no supervisada es el **Error Cuadrático Medio**, denotado como MSE por su abreviatura en inglés (*Mean Squared Error*). Es una medida de error que se calcula como una media aritmética de los cuadrados de la diferencia entre las predicciones y los valores reales. Su expresión matemática es la siguiente:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n} \left(Y_i - \widehat{Y}(X_i)\right)^2$$

siendo X_i, $i = 1, \cdots, n$ las variables explicativas, $\widehat{Y}(X_i), i = 1, \cdots, n$ las predicciones del modelo, $Y_i, i = 1, \cdots, n$ los valores reales del conjunto de datos y n el número total de observaciones.

También es habitual emplear la **Raíz Cuadrada del Error Cuadrático Medio**:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \left(Y_i - \widehat{Y}(X_i)\right)^2}$$

2.4.3. Curva ROC y AUC

Antes de proceder con la definición de la curva ROC y del área bajo la curva (AUC), se introduce la llamada **Matriz de confusión**.

En un modelo de clasificación supervisada con dos posibles categorías (en el caso particular de nuestro problema, no anomalía y anomalía), se comparan los valores reales frente a los valores de predicción en una tabla de contingencia, que llamaremos matriz de confusión:

Real\Predicción	Anomalía	No anomalía
Anomalía	Verdaderos positivos	Falsos Negativos
No anomalía	Falsos Positivos	Verdaderos negativos

Denotaremos por VP y VN a los verdaderos positivos y verdaderos negativos, respectivamente, y FP y FN a los falsos positivos y falsos negativos, respectivamente.

A partir de la anterior tabla, se definen las siguientes medidas de precisión de las predicciones:

- La **sensibilidad** o la tasa de verdaderos positivos (*Recall*), denotada por TVP y cuya expresión viene dada por:
$$TVP = \frac{VP}{P} = \frac{VP}{VP + FN}$$
- La **esfecifidad** o la tasa de verdaderos negativos, denotada por TVN y cuya expresión viene dada por:
$$TVN = \frac{VN}{N} = \frac{VP}{FP + VN}$$
- La **precisión global** o tasa de aciertos (*acuraccy*), denotada por PG y cuya expresión viene dada por:
$$PG = \frac{VP + VN}{P + N} = \frac{VP + VN}{VP + VN + FP + FN}$$
- El **índice predictivo positivo** (*Precision*), denotado por PPV y cuya expresión viene dada por:
$$PPV = \frac{VP}{VP + FP}$$
- El **índice predictivo negativo**, denotado por NPV y cuya expresión viene dada por:
$$NPV = \frac{TN}{TN + FN}$$

Una vez definidas la matriz de confusión y las medidas de precisión de las predicciones habituales, se define la curva ROC y el área bajo la curva (AUC). La **curva ROC** es una métrica de evaluación empleada para evaluar las estimaciones de las probabilidades y se define como la representación de la tasa de verdaderos positivos (TVP) frente a la tasa de verdaderos negativos (TVN), para distintos valores de un punto de corte. Sobre la curva ROC, se calcula el **área bajo la curva**, denotada habitualmente por AUC. Este área toma valores entre 0.5 y 1 y se emplea para evaluar el método de clasificación: se considera perfecto si tiene un valor de 1 y aleatorio si toma el valor de 0.5.

2.4.4. F1 Score

Cuando la muestra es muy desbalanceada, en lugar de emplear las medidas de precisión mencionadas anteriormente (sensibililidad, especifidad y precisión global), se emplea el F1 score (o puntuación F1). Esta métrica se define como la media armónica de la sensibilidad y el índice predictivo positivo:

$$F1 = \frac{2 \cdot VP}{2 \cdot VP + FP + FN}$$

2.4.5. Curva Precision Recall y AUCPR

Otra de las medidas empleadas para la evaluación de las estimaciones de las probabilidades, en el caso de trabajar con muestras desbalanceadas, es la **curva Precision Recall**. Esta métrica se define como la representación del índice predictivo positivo (PPV) frente a la tasa de verdaderos positivos (TVP), para distintos valores de un punto de corte. Sobre esta curva, se calcula el **AUCPR** (área bajo la curva Precision Recall). Toma valores entre 0.5 y 1 y, al igual que ocurría con el AUC, el clasificador es perfecto (respectivamente, aleatorio) si toma el valor máximo (respectivamente, el valor mínimo).

Capítulo 3

Cuestiones y fundamentos teóricos

En este capítulo se exponen las cuestiones y la fundamentación teórica necesarias para la resolución del problema. En primer lugar, se explican minuciosamente los principales conceptos relacionados con las series de tiempo. Se definen los llamados procesos estocásticos, estacionarios y de ruído blanco y se comentan las diferentes componentes presentes en una serie de tiempo. Posteriormente, se expone detalladamente el funcionamiento de los métodos supervisados de regresión autoexplicativos, como el modelo TBATS, y los métodos de Aprendizaje Estadístico, como el modelo de redes neuronales LSTM. Finalmente, en la última sección, se describe el modelo de regresión polinómica local robusta (LOWESS) y el procedimento de validación cruzada, que se emplea en el ajuste de los modelos de la interfaz H2oAutoMl de Python.

3.1. Conceptos relacionados con las series de tiempo

En esta sección se definen los principales conceptos relacionados con las series de tiempo, que se mencionan a la hora de definir los métodos autoexplicativos de detección de anomalías. Principalmente, se explican los conceptos de proceso estocástico, proceso estacionario, proceso de ruído blanco y las diferentes componentes estacionales presentes en las series de tiempo.

3.1.1. Proceso estocástico

Cuando un conjunto de variables aleatorias $\{X_t\}_{t\in C}$, está definido sobre el mismo espacio de probabilidad, se dice que es un **proceso estocástico**. En este trabajo nos centraremos en el proceso $\{X_t\}_{t\in\mathbb{Z}}$, siendo t el instante de observación. Un conjunto de observaciones de dicho proceso estocástico, se corresponde con una serie de tiempo. Denotaremos a la serie temporal observada con letras en minúscula: $x_1, x_2, \cdots, x_n$.

3.1.2. Proceso estacionario

Un proceso estocástico se dice **estacionario** cuando la media, la varianza y la función de autocovarianzas permanecen constantes a lo largo del tiempo. Matemáticamente, dado el proceso estocástico $\{X_t\}$, se verifica:

- $\mathbb{E}(X_t) = \mu_t = \mu$, $\forall t = 1 \cdots, n$
- $\mathrm{Var}(X_t) = \sigma_t^2 = \sigma^2$, $\forall t = 1 \cdots, n$
- $\gamma(t, t+k) = \gamma_k$, $\forall t = 1 \cdots, n$, $\forall k \in \mathbb{N}$

siendo $\gamma(s,t) = Cov(X_s, X_t) = \mathbb{E}((X_s - \mu_s)(X_t - \mu_t))$ la función de autocovarianzas, una medida del grado de dependencia temporal.

3.1.3. Proceso de ruído blanco

Un proceso estacionario $\{\epsilon_t\}$ que cumple las siguientes condiciones: $\mu = 0$, $\sigma^2 = \sigma_\epsilon^2$ y

$$\gamma_k = \gamma(t, t+k) = \begin{cases} \sigma_\epsilon^2, & k = 0 \\ 0, & k \neq 0 \end{cases}$$

se llama **proceso de ruído blanco**.

3.1.4. Componentes presentes en una serie de tiempo

En una serie de tiempo pueden presentarse los siguientes tipos de componentes: la tendencia, la estacional, la cíclica y la residual. En esta sección, se realiza un estudio detallado de las mismas. Para mayor información acerca de las componentes de una serie de tiempo, puede consultarse Jalles (2009) y Shumway et al. (2017).

Componente estacional

Una serie de tiempo presenta **componente estacional** cuando tiene un comportamiento periódico, es decir, cuando la serie presenta un patrón repetitivo cada cierto período de tiempo. Esto quiere decir que la serie $x_1, \cdots x_n$ presenta propiedades similares a la serie temporal desplazada $x_{s+1}, \cdots, x_{s+n}$, siendo s el período estacional de la serie temporal. Por ejemplo, dada una serie de tiempo con datos diarios

- Si $s = 12$, la serie presenta una componente estacional mensual.
- Si $s = 7$, la serie presenta una componente estacional semanal.
- Si $s = 1$, la serie presenta una componente estacional diaria.

En el caso de trabajar con una serie de tiempo con datos a nivel de minutos se multiplicarían los anteriores valores por 1440. Nótese que una serie de tiempo puede presentar más de una componente estacional; el número de componentes estacionales dependerá del comportamiento de los datos de la serie temporal

Componente cíclica

Una serie de tiempo presenta **componente cíclica** cuando presenta fluctuaciones, es decir, subidas y bajadas constantes que no son de período fijo. Dicho de otro modo, la variabilidad de la serie no es constante, depende de su nivel. Este comportamiento periódico se puede expresar como una combinación de lineal de funciones trigonométricas (senos y cosenos) con parámetros α y β:

$$\psi_t = \alpha \cos(\lambda \cdot t) + \beta \sin(\lambda \cdot t), \quad t = 1, \cdots, n$$

siendo λ la frecuencia, en radianes, del ciclo, es decir. Este modelo se conoce como ciclo determinista. En particular, si los parámetros α y β cambian con el paso del tiempo, el ciclo es estocástico y se formula como:

$$\psi_t = \rho\left(\psi_{t-1}\cos(\lambda) + \psi_{t-1}^* \sin(\lambda)\right) + k_t, \quad t = 1, \cdots, n$$
$$\psi_t^* = \rho\left(-\psi_{t-1}\sin(\lambda) + \psi_{t-1}^* \cos(\lambda)\right) + k_t^*, \quad t = 1, \cdots, n$$

con $\rho \in [0, 1]$ un parámetro de amortiguamiento, $\psi_0 = \alpha$, $\psi_0^* = \beta$, ψ_{t-1} el ciclo en el instante $t-1$ y ψ_{t-1}^* su conjugado. $k_t, k_t^* \sim N(0, \sigma_k^2)$ son los términos de error del modelo y se suponen que no son mutuamente correlados.

Componente tendencia

La **componente tendencia** de una serie de tiempo se define como la esperanza condicional de la serie a aumentar, disminuír o permanecer constante a largo plazo, siempre en ausencia de variables independientes. La expresión matemática del nivel local de tendencia de una serie temporal se obtiene a partir del ciclo estocástico: se toma $\rho = 1$ y $\lambda = 0$.

$$\psi_t = \psi_{t-1} + k_t, \quad t = 1, \cdots, n$$

$$\psi_t^* = \psi_{t-1}^* + k_t^*, \quad t = 1, \cdots, n$$

Componente residual

La **componente residual**, de una serie de tiempo, comúnmente conocida por ruído, resulta tras la estimación y la eliminación de las componentes estacionales y tendencia de la serie. Se corresponde con las fluctuaciones de alta frecuencia en una serie de tiempo.

3.2. Técnicas empleadas en la detección de anomalías

En esta sección se explican detalladamente los diferentes métodos no supervisados de regresión que se emplean habitualmente en la detección de anomalías. Las técnicas que se estudian son las siguientes:

- Modelo TBATS
- Modelo de componentes no observadas (UCM)
- Modelo Prophet
- Modelos de redes neuronales recurrentes: redes neuronales de memoria a corto plazo (LSTM)

Los tres primeros modelos pertenecen a los llamados métodos autoexplicativos, es decir, son modelos de regresión que no emplean variables adicionales en el ajuste (sólo emplean la serie temporal univariante), mientras que el modelo de redes neuronales LSTM es un modelo de regresión que emplea predictores externos.

3.2.1. Modelo TBATS

Los modelos TBATS (*Trigonometric seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal components*) son modelos de predicción en series de tiempo que permiten modelizar aquellos casos en los que la serie de tiempo de interés presenta más de una componente estacional, basándose en métodos de suavización exponencial. Su expresión matemática es la siguiente:

$$y_t^{(\omega)} = l_{t-1} + \varphi b_{t-1} + \sum_{i=1}^{T} s_{t-1}^{(i)} + \alpha d_t$$

siendo

$$\begin{cases} b_t = b_{t-1} + \beta d_t, & \text{la componente tendencia} \\ d_t = c + \sum_{i=1}^{p} \theta_i d_{t-i} + \epsilon_t + \sum_{j=1}^{q} \theta_j \epsilon_{t-j}, & \text{el proceso ARMA de órdenes p y q para los residuos} \\ s_t^{(i)} = \sum_{j=1}^{k_i} s_{j,t}^{(i)}, \quad i = 1, \cdots, T, & \text{la i-ésima componente estacional} \end{cases}$$

con φ un parámetro de amortiguamiento, $\alpha, \beta, \delta_1, \delta_2$ los parámetros de suavización, l_0 el nivel inicial, b_0 el parámetro pendiente, T el número total de componentes estacionales de la serie, ω una transformación de Box-Cox, m_i, k_i, $i = 1, \cdots, T$, los períodos estacionales y el número de términos de la serie de Fourier y [1]

$$s_{j,t}^{(i)} = s_{j,t-1}^{(i)} \cos(\lambda_j^{(i)}) + s_{j,t-1}^{*(i)} \sin(\lambda_j^{(i)}) + \delta_1^{(i)} d_t, \quad i = 1, \cdots, T$$

$$s_{j,t}^{*(i)} = -s_{j,t-1}^{(i)} \sin(\lambda_j^{(i)}) + s_{j,t-1}^{*(i)} \cos(\lambda_j)^{(i)} + \delta_2^{(i)} d_t, \quad i = 1, \cdots, T$$

$$\lambda_j^{(i)} = \frac{2\pi j}{m_i}, \quad i = 1, \cdots, T$$

La idea bajo los modelos TBATS es ajustar diferentes tipos modelos, teniendo en cuenta las siguientes condiciones:

- Considerando una transformación Box Cox y sin considerarla.
- Con y sin tendencia.
- Considerando procesos ARIMA(p,q) para los residuos y sin considerarlos.
- Sin considerar la estacionalidad.
- Diferentes períodos armónicos, es decir, períodos de la serie de Fourier, para modelizar los efectos de las componentes estacionales

y eligir el mejor de ellos siguiendo el llamado Criterio de Información de Akaike, denotado por AIC por su abreviatura en inglés (*Akaike Information Criterion*).

Para mayor información acerca de este tipo de modelos y su formulación puede consultarse Brozyna et al. (2018).

3.2.2. Modelo de componentes no observadas (UCM)

El **modelo de componentes no observadadas**, denotado UCM por su abreviatura en inglés (*Unobserved Component model*), realiza una descomposición de una serie de tiempo en tres componentes (tendencia, estacional y cíclica) y modela los efectos de regresión debido a la serie de predictores. Debido a su definición, este modelo puede considerarse como un modelo de regresión múltiple con coeficientes dependientes del tiempo. Su expresión matemática es la siguiente:

$$y_t = \mu_t + \delta_t + \psi_t + r_t + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{j=1}^{m} \beta_j x_{jt} + \epsilon_t$$

siendo:

- y_t, $t = 1, \cdots, n$, la serie de tiempo de interés, sobre la que se quieren realizar predicciones.
- μ_t, $t = 1, \cdots, n$, la componente tendencia de la serie.
- δ_t, $t = 1, \cdots, n$, la componente estacional de la serie.
- ψ_t, $t = 1, \cdots, n$, la componente cíclica de la serie.
- $r_t + \sum_{i=1}^{p} \phi_i y_{t-i}$, $t = 1, \cdots, n$, la parte autorregresiva de la serie, que explica los valores actuales de la serie en función de los p valores anteriores.

[1] La expresión s^* representa el conjugado de s, $\quad i = 1, \cdots T$

- $\sum_{j=1}^{m} \beta_j x_{jt}$, $t = 1, \cdots, n$, los factores causales de la serie, es decir, la contribución de las variables de regresión con coeficientes dependientes del tiempo debido a la serie de predictores.
- $\epsilon_t \sim N(0, \sigma_\epsilon^2)$, independientes e idénticamente distribuídos, $t = 1, \cdots, n$.

Para mayor información acerca del modelo UCM, puede consultarse Fomby (2008).

3.2.3. Modelo Prophet

Prophet es un software de acceso libre creado por la empresa Facebook (en la actualidad, Meta) que permite realizar predicciones en series temporales, modelizando las componentes estacionales presentes en la serie de tiempo y diferentes efectos del calendario.

Principalmente, el modelo Prophet es un modelo aditivo que realiza una descomposición de una serie de tiempo en:

- una curva de tendencia de crecimiento lineal o logístico.
- una curva que modela la componente estacional diaria.
- una curva que modela la componente estacional semanal.
- una curva que modela la componente estacional anual.
- efectos del calendario, como las vacaciones, días festivos o ocasiones especiales.
- curvas adiccionales, especificadas por el investigador, que modelizan la estacionalidad en función, por ejemplo, de las horas o de los días (por semana o fines de semana).

para realizar, posteriormente, predicciones sobre la misma. Cabe destactar que, las curvas que modelan las componentes estacionales son curvas de Fourier, de ahí a que las predicciones obtenidas se comporten como funciones trigonométricas. Para mayor información acerca de este modelo puede consultarse Rafferty (2021).

3.2.4. Modelo de redes neuronales de memoria a corto plazo (LSTM)

Las redes neuronales de memoria a corto plazo, denotadas por LSTM por su abreviatura en inglés (*Long Short-Term Memory*), son un tipo de redes neuronales artificiales diseñadas esencialmente para modelizar series de tiempo y sus dependencias con respecto a instantes anteriores (lags). Pertenecen a las llamadas redes neuronales recurrentes.

Para conocer el funcionamiento del modelo LSTM se necesita realizar una introdución de las redes neuronales artificiales (*Artificial Neural Networks*, ANN) y las redes neuronales recurrentes (*Recurrent Neural Networks*, RNN). Se explicará el funcionamiento de las mismas y los diferentes elementos que intervienen en los modelos.

Redes neuronales artificiales (ANN)

Las redes neuronales artificiales son modelos de aprendizaje estadístico no lineales que constan de varias estructuras conectadas entre sí, llamadas nodos (o neuronas), que permiten obtener predicciones empleando mecanismos en paralelo, es decir, mediante la construcción de capas entre las variables de entrada y salida del modelo y transformaciones de los datos. Nos centraremos en explicar el funcionamiento del modelo en el caso más sencillo, una red neuronal formada por una capa intermedia (red neuronal básica) denominada perceptrón de 1 capa (ver Figura 3.1(a)). Seguiremos como referencia el capítulo titulado *Neural Networks* del libro de Hastie et al. (2009).

En una red neuronal básica intervienen un total de 3 capas:

- Una capa de entrada (*input layer*) en la que se introducen todas las variables predictoras del modelo: $\mathbf{X} = (X_1, \cdots, X_N)$. Cada variable predictora se introduce en un nodo diferente.

- Una capa oculta o capa intermedia (*hiden layer*) formada por un número M de nodos, modelizados como una combinación lineal de las distintas variables de entrada:

$$Z_m = \sigma(\alpha_{0m} + (\boldsymbol{\alpha_m})^T \mathbf{X}), \quad m = 1, \cdots, M$$

  siendo σ la función de activación de los nodos y $\alpha_{0m}, \boldsymbol{\alpha_m} = (\alpha_{m1}, \cdots, \alpha_{mp})$ parámetros a estimar.

- Una capa de salida (*output layer*) con los valores resultantes de las predicciones del modelo, que se calculan como una transformación de una combinación lineal de los nodos de la capa oculta:

$$f_k(X) = g_k(\beta_0 + (\beta_k)^T \mathbf{Z}), \quad k = 1, \cdots, K$$

  siendo K el número de nodos, $\mathbf{Z} = (Z_1, \cdots Z_M)$ y $\beta_0, \beta_k, k = 1, \cdots, K$ parámetros a estimar. En problemas de regresión con respuesta univariante sólo se considera una capa de salida con los valores de las predicciones ($K = 1$) y g_k es típicamente la función identidad. En problemas clasificación de K clases hay un total de K variables de salida y se emplea la función softmax:

$$g_k(\mathbf{Z}) = \frac{e^{\beta_0 + \beta_k^T \mathbf{Z}}}{\sum_{l=1}^{K} e^{\beta_0 + \beta_l^T \mathbf{Z}}}, \quad k = 1, \cdots, K$$

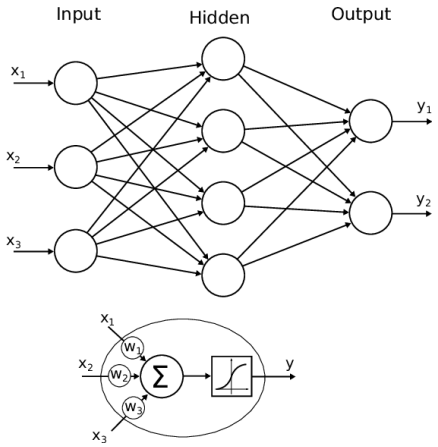
Con respecto a la función de activación, que interviene en la modelización de los nodos de la capa oculta, comentar que depende del problema. Si es un problema de clasificación se suele emplear la función de activación sigmoidal:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \in [0, 1]$$

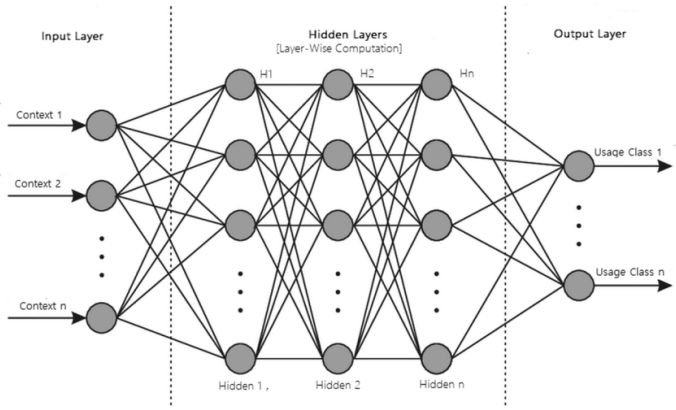
En el caso de problemas de regresión podría emplearse la anterior función, pero lo más habitual es emplear la función tangente hiperbólica:

$$\sigma(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{1 - e^{-2x}}{1 + e^{-2x}} \in [-1, 1]$$

Para ver cuáles son las diferentes funciones de activación que se emplean en los modelos de redes neuronales puede consultarse Sharma et al. (2017).



(a) Red neuronal artificial con una capa oculta. Fuente: Falch et al. (2016)



(b) Red neuronal artificial multicapa. Fuente: Sarker et al. (2021)

Figura 3.1: Esquema de diferentes tipos de redes neuronales artificiales.

La red neuronal artificial con una capa oculta con K nodos está formada por un total de $M(n+1)+K(M+1)$ parámetros desconocidos, comúnmente conocidos por pesos, que deben estimarse de forma que se ajusten de manera adecuada a los datos. Esta estimación se realiza típicamente miminizando la suma residual de cuadrados, en problemas de regresión, y la función de entropía cruzada (*cross-entropy function*), en problemas de clasificación. El problema de miminización se resuelve empleando el algoritmo *backpropagation*, que se describe a continuación para un problema de regresión. Para formular este algoritmo, se ha consultado el capítulo *Neural Networks* de Hastie et al. (2009).

Sea $\theta = \{\alpha_{0m}, \alpha_m, \beta_{0k}, \beta_k\}$, $m = 1 \cdots, M, k = 1, \cdots, K$, el conjunto de parámetros a estimar, $z_{mi} = \sigma(\alpha_{0m} + \alpha_m^T x_i)$, $z_i = (z_{1i}, \cdots, z_{Mi})$, $i = 1, \cdots N$ y

$$R(\theta) = \sum_{i=1}^{N} R_i = \sum_{i=1}^{N} \sum_{k=1}^{K} (y_{ik} - f_k(x_i))^2$$

la función a miminizar (suma residual de cuadrados), con derivadas parciales

$$\frac{\partial R}{\partial \beta_{km}} = \sum_{i=1}^{N} \frac{\partial R_i}{\partial \beta_{km}} = \sum_{i=1}^{N} -2(y_{ik} - f_k(x_i)) g_k'(\beta_k^T z_i) z_{mi}$$
$$\frac{\partial R}{\partial \alpha_{ml}} = \sum_{i=1}^{N} \frac{\partial R_i}{\partial \alpha_{ml}} = -\sum_{i=1}^{N} \sum_{k=1}^{K} 2(y_{ik} - f_k(x_i)) g_k'(\beta_k^T z_i) \sigma'(\alpha_m^T x_i) x_{il}$$

Si se aplica el método de descenso del gradiente, se obtienen los siguientes valores de los parámetros en la iteración $(r+1)$:

$$\beta_{km}^{(r+1)} = \beta_{km}^{(r)} - \gamma_r \cdot \frac{\partial R}{\partial \beta_{km}^{(r)}} = \beta_{km}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \beta_{km}^{(r)}}$$
$$\alpha_{ml}^{(r+1)} = \alpha_{ml}^{(r)} - \gamma_r \cdot \frac{\partial R}{\partial \alpha_{ml}^{(r)}} = \alpha_{ml}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \alpha_{ml}^{(r)}}$$

siendo γ_r el ratio de aprendizaje (*Learning Rate*), encargado de controlar lo que cambia el modelo en respuesta al error estimado cada vez que se actualizan los pesos del modelo. Este parámetro es

constante en cada una de las iteraciones del algortimo y debe converger a 0 a medida que aumenta el número de iteraciones.

Para llegar a las ecuaciones que describen el algoritmo *backpropagation*, es necesario reescribir las derivadas parciales del siguiente modo:

$$\frac{\partial R}{\partial \beta_{km}} = \sum_{i=1}^{N} \frac{\partial R_i}{\partial \beta_{km}} = \sum_{i=1}^{N} \delta_{ki} z_{mi}, \quad m = 1, \cdots, M, \quad k = 1, \cdots, K$$
$$\frac{\partial R}{\partial \alpha_{ml}} = \sum_{i=1}^{N} \frac{\partial R_i}{\partial \alpha_{ml}} = \sum_{i=1}^{N} s_{mi} x_{il}, \quad m = 1, \cdots M$$

siendo δ_{ki}, s_{mi}, $i = 1, \cdots N$, los errores del modelo en el instante i de las capas de salida y las capas ocultas, respectivamente. A las ecuaciones

$$s_{mi} = \sigma^{(1)}\left(\alpha_m^T x_i\right) \sum_{k=1}^{K} \beta_{km} \delta_{ki}, \quad m = 1, \cdots M, \quad i = 1, \cdots N$$

se les conoce como *backpropagation ecuations* y constituyen la base del algoritmo *backpropagation* para el cálculo de los pesos del modelo. Empleando estas ecuaciones, se implementa el algoritmo en dos pasos:

- Paso adelante (*forward pass*): se fijan los pesos actuales y se obtienen las predicciones $\widehat{f_k}(x_i)$.
- Paso hacia atrás (*backward pass*): se obtienen los errores del modelo en la capa de salida, δ_{ki}, y a partir de estos se obtienen los errores s_{mi}, empleando las ecuaciones *backpropagation*. Una vez obtenidos los dos errores, se aplica el método del descenso del gradiente.

La ventaja de este algoritmo es que puede implementarse de forma paralelizada: se seleccionan los datos de entrenamiento por lotes, que llamaremos *batch size* y se calculan los valores del ratio de aprendizaje hasta alcanzar el valor más pequeño, repitiendo el proceso un número determinado de iteraciones. Llamaremos *epochs* al número total de iteraciones que se realizan del algoritmo hasta alcanzar el óptimo, que suele ser un óptimo local (difícilmente converge a un óptimo global). Una de sus principales desventajas es que es un algoritmo lento e inestable con respecto al óptimo y se puede ver afectado negativamente por la correlación de las variables predictoras y esto se soluciona realizando un preprocesado de los datos.

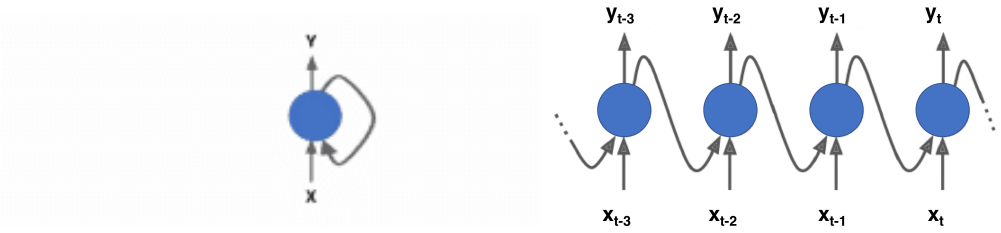
En Hastie et al. (2009) se explica detalladamente el proceso a seguir para entrenar este tipo de redes (que en la mayoría de ocasiones tienden al sobreajuste), con respecto a la elección de los parámetros de inicio del algoritmo, el número de capas intermedias a considerar (problema de redes neuronales artificiales multicapa) y el escalado de las variables de entrada del modelo. También se comenta el procedimiento a seguir para evitar el sobreajuste.

Redes neuronales recurrentes

Las redes neuronales recurrentes son una generalización de las redes neuronales artificiales, que acabamos de definir, diseñadas esencialmente para modelizar series de tiempo. A diferencia de las redes neuronales habituales, el ajuste del modelo se realiza de forma secuencial, es decir, en las capas ocultas del modelo se permite la interacción entre los distintos nodos para modelizar la dependencia de la serie de tiempo con respecto a instantes anteriores (lags). Comentar que, de entre la diversidad de aplicaciones de estos modelos, destaca el reconocimiento de voz e imágenes.

Para ilustrar la dependencia de la serie de tiempo, consideraremos el caso más sencillo, una red neuronal recurrente con una capa oculta compuesta por un único nodo, que recibe los datos de la serie

temporal como variable de entrada y produce una variable de salida, que se envía de nuevo al nodo. Si ese proceso se repite con el paso del tiempo, en cada instante se recibe el valor de la serie de tiempo y el valor correspondiente de la salida en el instante anterior, para generar una nueva variable de salida que se introducirá de nuevo en el nodo, tal y como se muestra en la siguiente figura, que puede encontrarse en Torres (2020).



(a) Nodo recurrente de la capa oculta

(b) Nodo recurrente desarrollado en el tiempo

Figura 3.2: Representación gráfica del proceso que tiene lugar en un nodo de la capa oculta de una red neuronal recurrente.

Al producirse este efecto de dependencia de los instantes anteriores en cada uno de los nodos de las capas ocultas de forma recurrente, podría decirse que cada uno de los nodos pose una capacidad de memoria. Por este motivo, al proceso de recurrencia que tiene lugar en cada uno de los nodos de la capa oculta se le suele llamar *cell memory* (o célula de memoria). En el caso de las redes neuronales recurrentes, en cada célula de memoria interviene únicamente la función de activación para el cálculo de las variables de salida. En redes neuronales recurrentes más generalizadas, como las redes neuronales de memoria a corto plazo, en la célula de memoria intervienen más elementos a parte de la función de activación, ya que se trata de redes neuronales más complejas. Más adelante se describirán detalladamente los componentes de las células de memoria del modelo LSTM.

Una vez explicado el proceso que tiene lugar en los nodos de la capa oculta en un modelo de redes neuronales recurrentes, se realiza un desarrollo matemático del proceso. Principalmente, el modelo se basa en las redes neuronales artificiales con la única diferencia de que en la capa oculta es necesario introducir los pesos del estado de la red en el instante anterior. El proceso recurrente que tiene lugar en un nodo de la capa oculta de una red neuronal recurrente en la que interviene la variable de entrada x_t y la variable de salida del instante anterior, y_{t-1} puede formularse de la siguiente forma:

$$y_t = \sigma(W_{\text{in}} x_t + W_{\text{rec}} y_{t-1} + b), \quad t = 1, \cdots p$$

siendo p el número de observaciones autocorreladas de la variable respuesta (comúnmente conocido como número de lags), W_{in} la matriz de pesos de la variable de entrada en el instante t, W_{rec} la matriz de pesos de la variable de salida en el instante $t-1$, b el sesgo del modelo y σ la función de activación que interviene en el proceso. Normalmente se suele considerar, en problemas de regresión, la función de activación tangente hiperbólica y, en problemas de clasificación, la función de activación sigmoidal, tal y como ocurría en las redes neuronales artificiales. El valor inicial de la variable de salida, y_0, suele fijarse a 0. Una formulación alternativa del proceso recurrente sería aplicar únicamente la función de activación a la variable de salida del instante $t-1$:

$$y_t = W_{\text{in}} x_t + W_{\text{rec}} \sigma(y_{t-1}) + b, \quad t = 1, \cdots, p$$

Con respecto al modelo de redes neuronales artificiales se cambia un poco la notación (en lugar de considerar parámetros en la formulación se introducen las matrices de pesos), siguiendo la formulación establecida en Pascanu et al. (2013). En este artículo también se describe el algoritmo de *backpropagation* para la estimación de las matrices de pesos en modelo de redes neuronales recurrentes, que se

explica, a continuación, de forma detallada. Recordemos que este algoritmo se desarolla en dos pasos: *forward propagation*, en el que se obtienen las predicciones del modelo y se calcula el ratio de aprendizaje, y *backward propagation*, en el que se calculan las derivadas parciales de los errores para obtener las estimaciones de los pesos empleando el método del descenso del gradiente y los ratios de aprendizaje. Este algoritmo debe generalizarse para el caso de las redes neuronales recurrentes para incluír el proceso de recurrencia con respecto al tiempo que tiene lugar en los nodos de las capas ocultas. Se denomina algoritmo *backpropagation through time*.

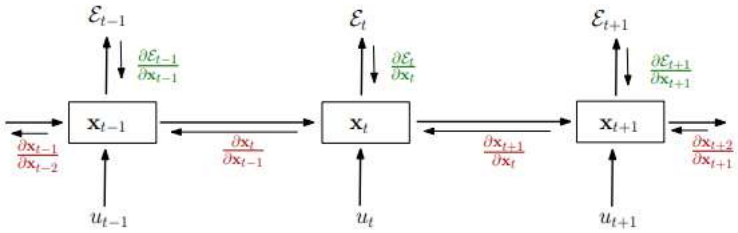


Figura 3.3: Esquema del algoritmo *backpropagation through time* en una célula de memoria. Fuente: Pascanu et al. (2013).

Para escribir las ecuaciones del algoritmo *backpropagation through time*, se reescribe la fórmula que define al proceso recurrente que tiene lugar en el nodo de una capa oculta de una red neuronal recurrente en la que interviene la variable de entrada, x_t, y la variable de salida en el instante anterior, y_{t-1}, del siguiente modo:

$$y_t = F(y_{t-1}, x_t, \theta), \quad t = 1, \cdots p$$

siendo p el número de observaciones autocorreladas de la variable respuesta (lags) y F una función parametrizada por el parámetro θ. Sea $\epsilon_t, \quad t = 1, \cdots p$, el error obtenido en el instante t y $\epsilon = \sum_{t=1}^{p} \epsilon_t$. El algoritmo *backpropagation through time* puede formularse con las siguientes ecuaciones (con los gradientes expresados como suma producto):

$$\begin{aligned}
\frac{\partial \epsilon}{\partial \theta} &= \sum_{t=1}^{p} \frac{\partial \epsilon_t}{\partial \theta} \\
\frac{\partial \epsilon_t}{\partial \theta} &= \sum_{1 \leq k \leq t} \left(\frac{\partial \epsilon_t}{\partial y_t} \frac{\partial y_t}{\partial y_k} \frac{\partial^+ y_k}{\partial \theta} \right) \\
\frac{\partial y_t}{\partial y_k} &= \prod_{k < i \leq t} \frac{\partial y_i}{\partial y_{i-1}} = \prod_{k < i \leq t} W_{\text{rec}}^T \text{diag}\left(\sigma^{'}(y_{t-i}) \right)
\end{aligned}$$

donde $\frac{\partial^+ y_k}{\partial \theta}$ representa la derivada parcial inmediata de y_k con respecto al parámetro θ, o lo que es lo mismo, y_{k-1} es una constante con respecto a θ. Esto se ve reflejado en la matriz jacobiana $\left(\frac{\partial y_k}{\partial W_{\text{rec}}} \right)$, cuyo valor para la fila i-ésima es $\sigma(y_{t-1})$. Los términos

$$\frac{\partial \epsilon_t}{\partial y_t} \frac{\partial y_t}{\partial y_k} \frac{\partial^+ y_k}{\partial \theta}, \quad k = 1, \cdots t, \quad t = 1, \cdots p$$

representan la influencia del parámetro θ en el instante k al error ϵ_t en los instantes posteriores $t > k$. Llamaremos a estos factores componentes (o contribuciones) temporales a los gradientes $\frac{\partial \epsilon_t}{\partial \theta}$. Con respeto al los factores $\frac{\partial y_t}{\partial y_k}$, comentar se distinguen dos tipos de componentes, según los valores del índice k:

- *long term contributions* (componentes a largo plazo), cuando los valores del índice k son significativamente más pequeños que el índice t ($k \ll t$).

- *short term contributions* (componentes a corto plazo), cuando los valores del índice k son cercanos los valores del índice t.

Este tipo de componentes afectan de modo importante a los gradientes $\frac{\partial \epsilon_t}{\partial \theta}$, causando los problemas de *vanishing* y *exploding* a lo largo del proceso hacia atrás (*backward pass*), que se describen a continuación. El problema de *exploding* en los gradientes ocurre cuando hay un gran incremento en la norma de los gradientes en la fase de entrenamiento de las redes neuronales recurrentes debido al incremento exponencial de las componentes a largo plazo (*long term contributions*), lo que genera un problema en el entrenamiento del modelo ya que se asigna un valor elevado a los pesos. En el problema de *vanishing* en los gradientes ocurre lo contrario, hay un decrecimiento exponencial de las componentes a largo plazo, lo que impide entrenar el modelo teniendo en cuenta las observaciones autocorreladas de la variable respuesta (lags). Este problema ocurre, principalmente, debido a la función de activación considerada en el modelo. Sin entrar en más detalles acerca de estos problemas, comentar que para mayor información acerca de este tipo de problemas, su interpretación geométrica y posibles soluciones puede consultarse Pascanu et al. (2013).

El problema de *exploding* en los gradientes, que se acaba de describir, puede solucionarse si se reduce el número total de gradientes a calcular. El caso del problema de *vanishing*, que depende de la función de activación, es mucho más complicado de resolver. Se precisa modificar la célula de memoria de forma que existan mecanismos de activación para liberar la memoria con la finalidad de actualizar las variables de salida que se van obteniendo en el proceso recurrente. Las redes neuronales de memoria a corto plazo (*Long short-term memory networks*) son una generalización de las redes neuronales recurrentes que permiten, a partir de la inclusión de puertas en las células de memoria, recordar los valores de salida a largo plazo, solucionando, de este modo, el problema de *vanishing* de los gradientes en las redes neuronales recurrentes. A continuación se explica, de forma detallada, el mecanismo que tiene lugar en las células de memoria de las capas ocultas de las redes neuronales LSTM. Seguiremos como referencia Dixon (2022).

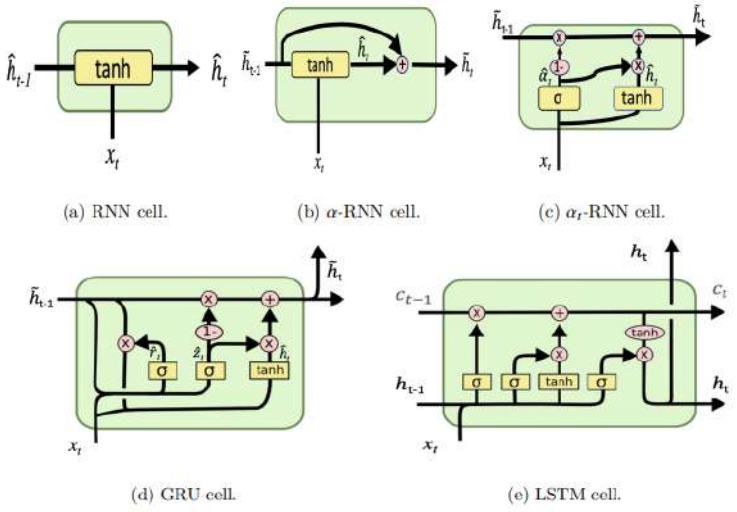


Figura 3.4: Esquema de la célula de memoria de las capas ocultas de diferentes tipos de redes neuronales recurrentes. Fuente: Dixon (2022)

La célula de memoria de una red neuronal LSTM está formada por cuatro puertas: la puerta de entrada (*input gate*), la puerta para "olvidar" (*forget gate*), la puerta de salida (*output gate*) y la puerta de estado de la célula de memoria (*cell memory state gate*), cuya formulación matemática es la siguiente: para $t = 1, \cdots, p$, con p el número de observaciones autocorreladas de la variable respuesta (lags) y $c_0 = 0$, $y_0 = 0$, los valores iniciales de la célula de memoria y la variable de salida,

$$\textit{input gate}: \quad \widehat{z_t} = \sigma\left(W_z x_t + U_z y_{t-1} + b_z\right)$$

$$\textit{forget gate}: \quad \widehat{\alpha_t} = \sigma\left(W_\alpha x_t + U_\alpha y_{t-1} + b_\alpha\right)$$

$$\textit{output gate}: \quad \widehat{r_t} = \sigma\left(W_r x_t + U_r y_{t-1} + b_r\right)$$

$$\textit{cell memory state gate}: \quad \widehat{c_t} = \tanh\left(W_c x_t + U_c y_{t-1} + b_c\right)$$

siendo x_t la variable de entrada en el instante t, y_{t-1} la variable de salida en el instante anterior, b_z, b_α, b_r, b_c los sesgos en cada una de las puertas, W_z, W_α, W_r, W_c y U_z, U_α, U_r, U_c los pesos de la variable de entrada y de la variable de salida en el instante anterior en cada una de las puertas, respectivamente, tanh es la función de activación tangente hiperbólica y σ representa a la función de activación sigmoidal.

Para cada $t = 1, \cdots, p$ se actualiza la célula de memoria mediante la siguiente expresión:

$$c_t = \widehat{\alpha_t} \circ c_{t-1} + \widehat{z_t} \circ \widehat{c_t}$$

donde c_{t-1} es la célula de memoria en el instante anterior y $\circ$ representa el producto de Hadamard (producto elemental). Esta expresión es la que permite obtener las variables de salida en en instante t:

$$y_t = \widehat{r_t} \circ \tanh(c_t)$$

Nótese que si $\widehat{r_t} = 1$, la variable de salida se calcula empleando únicamente la célula de memoria en el instante t, sin emplear la puerta de salida.

3.3. Regresión no paramétrica: regresión local

En esta sección nos centraremos en explicar de forma detallada la regresión local perteneciente a los modelos de regresión no paramétrica. Principalmente se describe la regresión polinómica local y una generalización de la misma, la regresión polinómica local robusta, LOWESS.

Los modelos de regresión local o regresión tipo núcleo, entre los que se incluyen la regresión polinómico local (*Local Smoothing*) y la regresión polinómica local robusta, denotado habitualmente por LOWESS por su nomenclatura en inglés (***Lo****cally* ***We****ighted* ***S****catter plot* ***S****moothing*), son métodos clásicos de regresión no paramétrica cuyo objetivo es ajustar, localmente, un modelo de regresión polinómica. De este modo, los valores de la variable regresora se predicen en un entorno de la misma, es decir, se toman las observaciones más cercanas a la variable regresora para predecir los valores de la misma, de ahí el nombre de regresión polinómica local. Comentar que, estos modelos asumen una función que se supone que es suave con respecto a los predictores, de forma que la variable regresora toma la siguiente expresión:

$$Y = m(X_1, \cdots, X_p) + \epsilon$$

con $X_1, \cdots X_p$ las variables predictoras y ϵ el término de error.

Una vez comentada la finalidad de la regresión local, se explican detalladamente la regresión polinómico local y la regresión polinómico local robusta (LOWESS). Para la definición de estos métodos de regresión no paramétrica se ha consultado Fan (1996).

3.3.1. Regresión polinómico local

La regresión polinómico local realiza ajustes de mínimos cuadrados ponderados con polinomios de grado arbitrario p. Consideremos el caso univariante. Supongamos que la función suavizadora m es derivable p veces. Entonces, puede realizarse su desarollo de Taylor y aproximar localmente la función $m(t)$ por un polinomio de grado p en un entorno local de x:

$$m(t) \approx \sum_{j=1}^{p} \frac{m^{(j)}(x)}{j!}(t-x)^j = \sum_{j=0}^{p} \beta_j (t-x)^j$$

La estimación de los coeficientes $\beta_j, j = 1, \cdots p$ se consigue miminizando la siguiente función:

$$\min_{\beta_0,\cdots,\beta_p} \left\{ \sum_{i=1}^{n} \left(Y_i - \sum_{j=0}^{p} \beta_j (X_i - x)^p \right)^2 K_h\left(X_i - x\right) \right\}$$

siendo $K_h(X_i - x) = \frac{1}{h} K\left(\frac{X_i - x}{h}\right)$ la función núcleo, con $h > 0$ el parámetro ventana, encargado de controlar el grado de suavidad. Luego, si denotamos por $\widehat{\beta}_j$ a la estimación de los coeficientes, teniendo en cuenta que

$$\beta_j = \frac{m^{(j)}(x)}{j!}, \quad j = 1, \cdots, p$$

la estimación de la j-ésima derivada de la función suavizadora es:

$$\widehat{m}^{(j)}(x) = j! \cdot \widehat{\beta}_j$$

A $\widehat{m}(x) = \widehat{\beta}_0$ se le denomina estimador polinómico local de grado p. En particular, si $p = 0$, se ajusta localmente un polinomio de grado 0 (constante) y se obtiene el llamado **estimador de Nadayara-Watson**, cuya expresión viene dada por

$$\widehat{m}^{(0)}(x) = \widehat{m}_h(x) = \frac{\sum_{i=1}^{n} K_h(X_i - x) Y_i}{\sum_{i=1}^{n} K_h(X_i - x)}$$

y, si $p = 1$, se realiza un ajuste local lineal y se denomina el **estimador lineal local**, cuya expresión viene dada por

$$\begin{aligned}\widehat{m^{(0)}(x)} = \widehat{m}_0(x) &= \frac{\sum_{i=1}^{n} w_i Y_i}{\sum_{i=1}^{n} w_i} \\ &= \frac{\sum_{i=1}^{n} Y_i \cdot K_h(X_i - x) \left[\sum_{i=1}^{n} K_h(X_i - x)(X_i - x)^2 - (X_i - x) \sum_{i=1}^{n} K_h(X_i - x)(X_i - x)\right]}{\sum_{i=1}^{n} K_h(X_i - x) \left[\sum_{i=1}^{n} K_h(X_i - x)(X_i - x)^2 - (X_i - x) \sum_{i=1}^{n} K_h(X_i - x)(X_i - x)\right]}\end{aligned}$$

Estos estimadores tienen un comportamiento asintótico semejante. En efecto: cuando $h \to 0$ el estimador de Nadayara-Watson converge a Y_i y cuando $h \to \infty$ converge a $\bar{Y}$, la media muestral de la variable respuesta. En el caso del estimador lineal local, a medida que aumenta h, el estimador se aproximará a una recta (ajuste lineal). En la práctica, para evitar el llamado efecto frontera, se emplea el estimador lineal local.

3.3.2. Regresión polinómica local robusta

La regresión polinómica local robusta, o LOWESS, es una generalización de la regresión polinómica local descrita anteriormente. Esta metodología de suavización, introducida por el matemático Cleveland en el siglo XX, realiza un número determinado ajustes de regresión polinómica local empleando, en cada uno de ellos, diferentes valores de los pesos. Este tipo de regresión puede considerarse como un proceso iterativo, cuyo algoritmo, desarollado por Cleveland, se describe a continuación:

- Para cada una de las observaciones, se calcula el estimador lineal local $\widehat{m^{(0)}(X_i)}, i = 1, \cdots, n$ empleando el llamado Kernel triweight

$$K(x) = \frac{70}{81}(1 - |x|^3)^3 I_{[-1,1]}(x)$$

- Una vez calculadas las estimaciones, se determinan los residuos no paramétricos, que se calculan como la diferencia entre las observaciones de la variable regresora y los estimadores lineal local:

$$\widehat{\epsilon_i}^{(1)} = Y_i - \widehat{m^{(0)}(X_i)}, \quad i = 1, \cdots, n$$

- Se calculan los pesos (llamados pesos de robustez) y se realiza un nuevo ajuste de regresión polinómica local. Los pesos se obtienen del siguiente modo:

$$w_i^{(1)} = K\left(\frac{\widehat{\epsilon_i}}{6M}\right), \quad i = 1, \cdots, n$$

  siendo M la mediana de los valores absolutos de los residuos y $K(x) = (1 - |x|^2)^2 I_{[-1,1]}(x)$ el llamado Kernel Biweight. Por la propia definición del Kernel, estos pesos tomarán valores grandes (respectivamente, pequeños) en los residuos elevados (respectivamente, bajos).

- Se repite el proceso hasta alcanzar convergencia (Cleveland sugiere emplear 3 iteraciones)

3.4. Validación Cruzada

La validación cruzada o *Cross-Validation*, CV, es una metodología empleada en Aprendizaje Estadístico para estimar el error de predicción en nuevas observaciones con la finalidad de prevenir el sobreajuste. Se distinguen tres tipos de validación cruzada:

- La validación cruzada dejando uno fuera (*Leave one-out cross validation*)
- La validación cruzada aleatoria (*Random cross validation*).
- La validación cruzada de k grupos (*k-fold cross validation*).

En este trabajo de fin de máster se emplea la validación cruzada de k grupos, que se describe a continuación siguiendo como referencia Berrar (2019).

En la validación cruzada de k grupos la muestra de entrenamiento se particiona, de forma aleatoria, en k subconjuntos de tamaño similar. Posteriormente, se ajusta un modelo empleando k-1 particiones como muestras de entrenamiento y la partición restante como muestra de validación para evaluar el modelo. El procedimiento se repite un total de k veces, correspondiente con el número total de particiones de la muestra. En cada una de las iteraciones de la validación cruzada se obtiene la métrica de evaluación correspondiente (en problemas de regresión se considera, por ejemplo, el MAE y en problemas de clasificación se considera el AUCPR). A partir de estas métricas, se obtiene la media, que será la métrica de evaluación del modelo considerado en nuevas observaciones. Cabe destacar que se suelen tomar k=5 o k=10 grupos. La siguiente gráfica ilustra un ejemplo del algoritmo de validación cruzada con k=10 grupos:

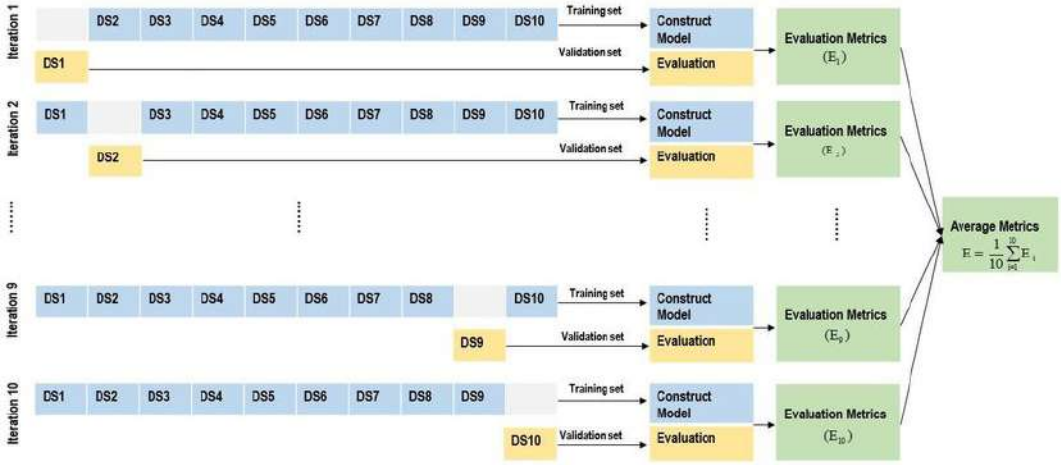


Figura 3.5: Esquema de la validación cruzada de k=10 grupos. Fuente: Sokkhey et al. (2020)

Este procedimiento se realiza para las diferentes combinaciones de hiperparámetros. Una vez terminado el proceso, teniendo en cuenta las diferentes métricas de evaluación, se selecciona el modelo que miminice el error de predicción en problemas de regresión, por ejemplo el MAE, y que maximice el error de predicción en problemas de clasificación, por ejemplo el AUCPR.

Capítulo 4

Análisis exploratorio de los datos

En este capítulo se realiza un análisis exploratorio de los datos del número de operaciones realizadas en los años 2020 y 2021 y los 3 primeros meses del año 2022. La idea es realizar diferentes tipos de representaciones gráficas para ver cómo se comporta la serie de tiempo con la que se va a trabajar:

- Gráficos para determinar la evolución del número de operaciones. Se representan, en los años 2020 y 2021, el número total de operaciones diarias realizadas mensualmente y diariamente.
- Gráficos para determinar las componentes estacionales de la serie temporal (diaria, semanal, mensual o anual). Se incluyen gráficos de la serie en períodos de un mes, de una semana y de un día.
- Representación gráfica de las componentes estacionales desagregadas, entre las que se incluye la componente tendencia y la componente residual.
- Gráficos funcionales de la serie temporal en función del día de la semana. Se representan, para cada uno de los días de la semana, las curvas del número de operaciones junto con la curva media y los intervalos de confianza al 95 %. También se representan curvas candidatas a "outliers". Estos gráficos nos sirven para comprobar la distribución del número de operaciones según los días de la semana.

4.1. Gráficos para determinar cómo evoluciona el número de operaciones

Para determinar la evolución del número de operaciones a lo largo de los años 2020 y 2021, se realizan representaciones gráficas del número total de operaciones realizadas por mes y del número total de operaciones realizadas por día en los años considerados.

En primer lugar, veamos cómo es el comportamiento mensual. Independientemente del año considerado, se aprecia un comportamiento similar: el número de operaciones sufre una caída considerable en el mes de Agosto y un repunte a final de año. Esta caída en el mes de Agosto se debe, principalmente, a que las oficinas bancarias de la entidad abren en horario de verano y la mayor parte del personal coge vacaciones. Además, cabe destacar que hay cierta diferencia en los 6 primeros meses del año. En el año 2021, el número de operaciones sube hasta el mes de Marzo, donde sufre una pequeña caída manteniéndose estable a lo largo de los meses de Abril, Mayo y Junio. Sin embargo, en el año 2020 se produce una caída en el número de operaciones a lo largo de los meses de Febrero y Marzo, debido a la pandemia del coronavirus. A lo largo de los meses siguientes el número de operaciones se recupera, volviendo a subir.

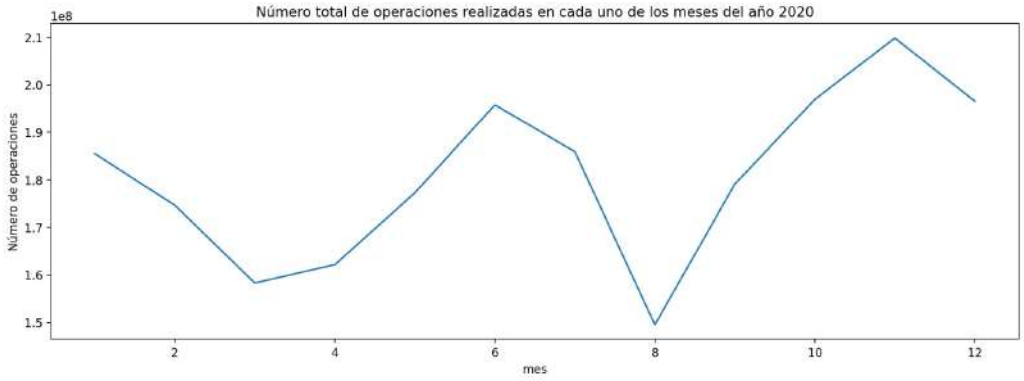


Figura 4.1: Evolución del número total de operaciones mensuales realizadas en el año 2020.

Figura 4.2: Evolución del número total de operaciones mensuales realizadas en el año 2021.

Una vez determinado el comportamiento del número total de operaciones realizadas por cada uno de los meses en los años 2020 y 2021, se realiza un gráfico del número total de operaciones realizadas por cada día, para ver su evolución. Se puede apreciar que, independientemente del año considerado, existe un patrón anual similar exceptuando los meses del confinamiento por el coronavirus (mediados del mes de Marzo, Abril y Mayo del 2020), donde el número total de operaciones diarias es inferior al comportamiento habitual.

Figura 4.3: Evolución del número total de operaciones diarias realizadas en el año 2020.

Figura 4.4: Evolución del número total de operaciones diarias realizadas en el año 2021.

Nótese que, a pesar del confinamiento, el número de operaciones realizadas en el año 2020 es, en general, superior al las realizadas en el año 2021 (es suficiente con fijarse en la escala de los ejes de las y en los gráficos).

4.2. Gráficos para determinar las componentes estacionales

El siguiente paso que se debe realizar en un análisis exploratorio de una serie temporal es determinar si posee alguna componente estacional que puede ser, por ejemplo, anual, mensual, semanal o diaria. Para ver si existe alguna componente estacional, se realizan gráficos de la serie temporal en períodos de un mes, de una semana y de un día.

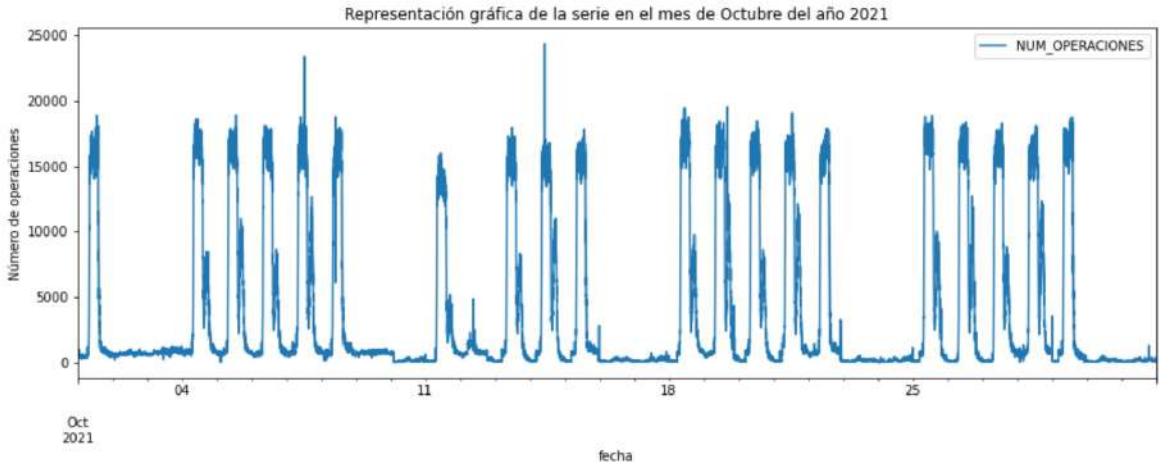


Figura 4.5: Evolución del número de operaciones realizadas en el mes de Octubre de 2021.

En primer lugar, se representa la serie en un mes, en particular se considera el mes de Octubre del año 2021. Se puede apreciar que, aparentemente, la serie temporal no posee una componente estacional mensual. Si existiese, el comportamiento mensual sería similar pero, tal y como se ha comentado en la anterior sección, en el mes de Agosto, el número de operaciones es inferior a los restantes meses ya que las oficinas abren en horario de verano y las mayor parte del personal coge vacaciones. Además, se observa que la serie temporal tiene un comportamiento similar en períodos de 7 días, lo que sugiere la existencia de una componente estacional semanal. Para determinar si realmente existe, se representa

gráficamente las serie en una semana, por ejemplo, la semana del día 24 al 31 de Octubre de 2021.

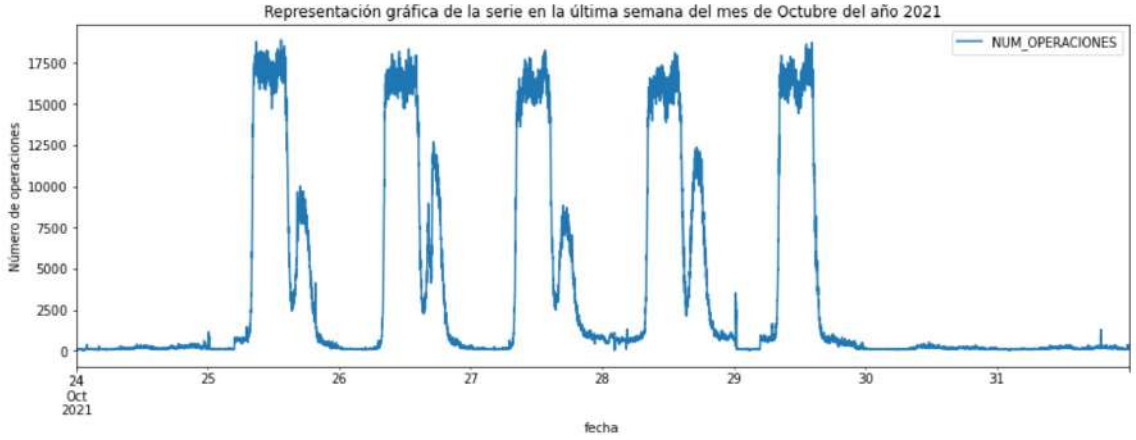


Figura 4.6: Evolución del número de operaciones realizadas en los días 24 a 31 de Octubre de 2021.

Se puede apreciar que la serie temporal posee un comportamiento que depende del día de la semana. Los fines de semana (en el gráfico, los días 24, 30 y 31 de Octubre) el número de operaciones es cercano a 0, lo cual no es sorprendente pues las oficinas bancarias permanecen cerradas y las pocas operaciones contabilizadas se deben a las realizadas a través de la Banca Móvil y la Banca Electrónica. En el caso de los días laborables, de Lunes a Jueves se aprecia la existencia de dos picos bien diferenciados, que se corresponden con los horarios de mañana y tarde de las oficinas bancarias, respectivamente. Los Viernes sólo se aprecia un pico, pues las oficinas bancarias no abren por las tardes. Comentar que el número de operaciones realizadas en horario de tardes es menor que las realizadas en horario de mañana debido a que no todas las oficinas de la entidad bancaria abren por las tardes. Vemos, por lo tanto, que la serie del número de operaciones posee una componente estacional semanal y, debido al comportamiento que se observa los días por semana, podría presentar una componente estacional diaria. Para determinar si realmente existe se representa gráficamente la serie temporal en un día, por ejemplo, el día 27 de Octubre de 2021.

Figura 4.7: Evolución del número de operaciones realizadas el día 27 de Octubre de 2021.

En la representación gráfica de la serie en un día (en este caso un Miércoles) se aprecia muy bien el comportamiento del número de operaciones dependiendo de la hora:

- El número de operaciones es prácticamente nulo en las horas de la madrugada.
- Con la apertura de las oficinas bancarias a las 8:15 horas el número de operaciones empieza a subir y se estabiliza a lo largo de la mañana.
- Con el cierre de las oficinas bancarias a las 14:15 horas, el número de operaciones comienza a descender hasta la apertura, de las oficinas bancarias de la entidad en horario de tarde. Las operaciones se estabilizan a lo largo de la tarde y, con el cierre de las oficinas, empiezan a descender hasta alcanzar valores cercanos a 0. Se aprecia que el número de operaciones es inferior que en el horario de mañana, pues no todas las oficinas abren por las tardes, como comentamos anteriormente.

En el caso de considerar un Viernes, dado que las oficinas bancarias no abren ese día por la tarde, una vez cerradas las oficinas, el número de operaciones desciende hasta alcanzar valores cercanos a 0. Si el día es no laborable (fines de semana y festivos) el número de operaciones es cercano a 0 (sólo se contabilizan las operaciones realizadas en Banca Móvil y en Banca Electrónica). Veremos esto con más detalle en la sección dedicada a los gráficos funcionales.

4.3. Gráficos de las componentes estacionales desagregadas

Una vez determinadas las componentes estacionales de la serie del número de operaciones (recordemos que hay dos componentes: la diaria y la semanal) se realiza la representación gráfica de las componentes estacionales desagregadas que incluye la componente tendencia, la residual y las estacionales diaria y semanal.

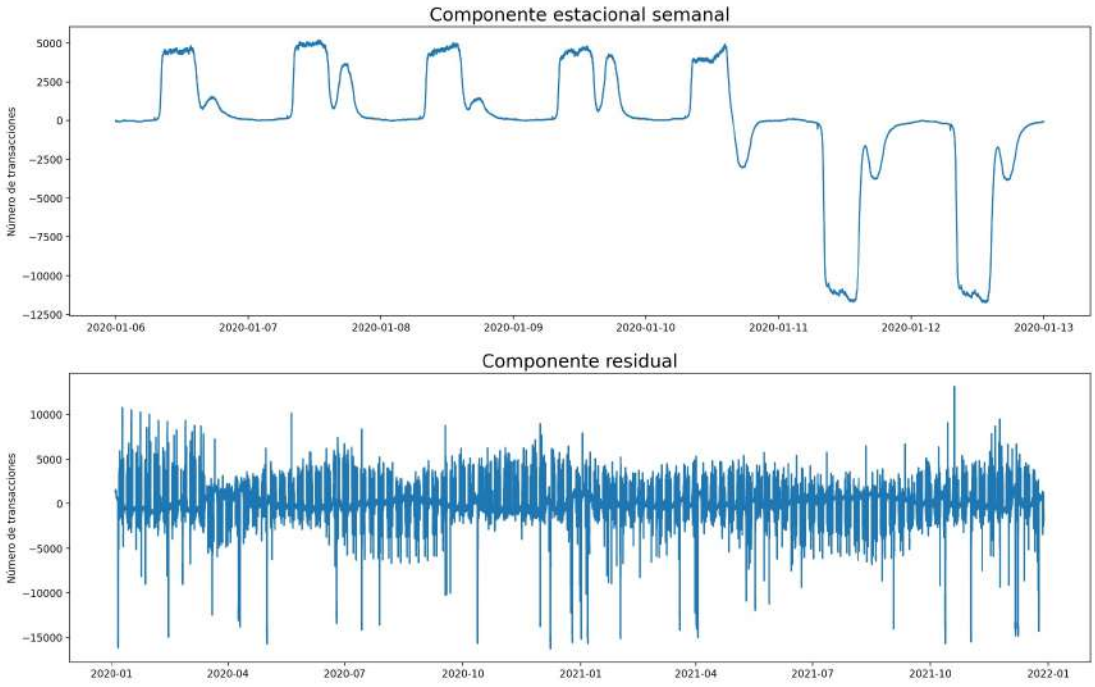


Figura 4.8: Gráficos de las componentes estacionales deagregadas: componente tendencia, componentes estacionales diaria y semanal y componente residual.

En primer lugar, aparentemente la serie temporal del número de operaciones no presenta componente tendencia. Si nos fijamos en el gráfico que representa la componente tendencia de la serie temporal, se puede apreciar que el número de operaciones realizadas no tiene un comportamiento que dependa del tiempo a largo plazo, es decir, no se observa ni crecimiento ni decrecimento del número de operaciones con el paso del tiempo sino que se aprecia variabilidad en el número de operaciones en los años 2020 y 2021, independientemente del año considerado. Además, se puede apreciar que hay ciertos períodos de tiempo en los que el número de operaciones es decreciente: ocurre a principio de año (debido al comienzo de un nuevo año) y en los meses de Abril de 2020 y 2021 (puede ser debido al efecto de la semana Santa). En el año 2020 también se aprecia una bajada notable del número de operaciones en el mes de Marzo del año 2020, época en la que empezó el confinamiento por la pandemia del coronavirus. Vemos, por lo tanto, que la serie temporal posee componente tendencia.

Con respecto a la componente estacional diaria, representada para el Miércoles 1 de Enero del año 2020, un festivo nacional, se aprecia que sigue el comportamiento de la serie del número de operaciones realizadas un Miércoles pero de forma más suavizada. Si se observa la componente estacional semanal, representada desde el día 6 al 12 de Enero del año 2020, se aprecia una caída notable los días 11 y 12 debido al efecto del fin de semana, donde el número de operaciones es prácticamente nulo. Al igual que ocurría con la componente estacional diaria, la curva es más suave que si se representa directamente la serie temporal en esa semana.

Para finalizar, nos centramos en la componente residual, también conocida como ruído, que representa las fluctuaciones de alta frecuencia en la serie temporal considerada. Como podemos apreciar, la componente residual es una serie estacionaria con media 0 y las fluctuaciones se producen en mayor medida a principio y a finales de año. Las fluctuaciones representan el efecto de los festivos y de los fines de semana sobre la serie del número de operaciones.

4.4. Gráficos funcionales

Para finalizar con el análisis exploratorio de los datos, se representan los gráficos funcionales de la serie temporal en función del día de la semana, para determinar cómo es la distribución del número de operaciones y determinar si hay diferencias notables entre los días de la semana. Se representan, para cada uno de los días de la semana, las curvas del número de operaciones junto a la curva media (en color negro) y las bandas de confianza al 50 % y al 95 % (en color azul), que son estimaciones basadas en profundidad (puede consultarse Hyndman et al. (2010)). También se representan las curvas candidatas a "outliers", es decir, aquellas curvas que presentan un patrón diferente al comportamiento habitual como, por ejemplo, los días festivos, cuyo comportamiento es similar a los fines de semana. Para evitar el comportamiento de los días festivos, se realiza la representación gráfica de los diferentes días sin considerar los festivos nacionales, autonómicos y locales (A Coruña). La lista completa de festivos puede consultarse en el Apéndice A.

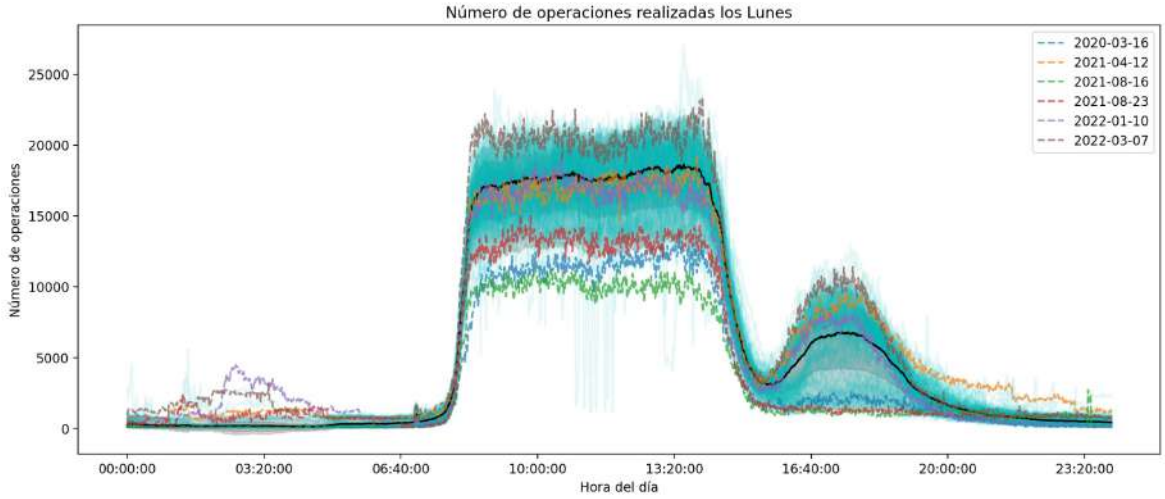


Figura 4.9: Gráfico funcional de los Lunes.

En el gráfico funcional de los Lunes se observa el siguiente comportamiento:

- El número de operaciones se mantiene cercano a 0 en las horas de la madrugada. Las escasas operaciones contabilizadas en ese horario se corresponden con operaciones realizadas en la Banca Electrónica y en la Banca Móvil.
- Con la apertura de las oficinas bancarias de la entidad a las 8:15 horas el número de operaciones aumenta y se estabiliza a lo largo de la mañana.
- Con el cierre de las oficinas bancarias de la entidad a las 14:15 horas, el número de operaciones comienza a disminuír hasta la apertura, de nuevo, de las oficinas bancarias en horario de tarde.
- El número de operaciones se mantiene a lo largo de la tarde hasta el cierre de las oficinas, donde se produce un nuevo descenso del número de operaciones, hasta alcanzar valores cercanos a 0. Al igual que ocurría en las horas de madrugada, las escasas operaciones contabilizadas se corresponden con operaciones realizadas en Banca Electrónica y en Banca Móvil.

Cabe destacar que el número de operaciones realizadas en horario de tarde es inferior a las realizadas en horario de mañana. Esto es debido a que no todas las oficinas bancarias de la entidad abren en horario de tarde.

Con respecto a las curvas con un comportamiento diferente al resto, cabe destacar los casos de los días 16/3/2020, 16/8/2021, 23/8/2021, 10/1/2022 y 7/3/2022:

- El día 16/3/2020 se produce un descenso notable del número de operaciones realizadas. Esto no es algo que deba sorprendernos pues justo coincide con el primer Lunes tras el decreto del estado de alarma por la pandemia de coronavirus. Durante el período de confinamiento, las oficinas bancarias permanecieron cerradas y, por ese motivo, el número de operaciones realizadas es inferior a la media. Nótese que apenas se contabilizan operaciones en horario de tarde.

- Los días 16 y 23 de Agosto se aprecia que el número de operaciones realizadas en horario de mañana es inferior al habitual y que por las tardes apenas se contabilizan. Esto es debido a que, en el mes de Agosto, las oficinas bancarias de la entidad sólo abren por las mañanas y en horario de verano. Además, durante ese mes la mayoría del personal coge vacaciones.

- El día 10/1/2022 se produce un ascenso del número de operaciones, superando en algunos casos los valores de la curva media. Esto es debido a que se produce la vuelta de las vacaciones de navidad.

- En el caso del día 7/3/2022 se produce un aumento en el número de operaciones con respecto a valores habituales. Probablemente, este aumento se deba a que en primeros de mes es cuando se contabiliza mayor número de operaciones.

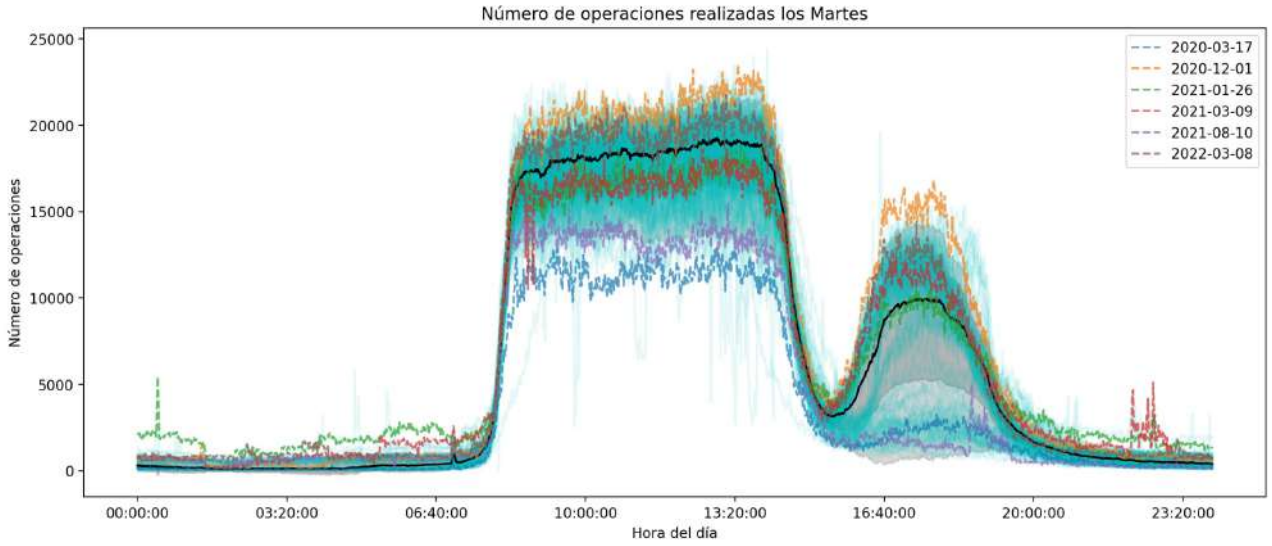


Figura 4.10: Gráfico funcional de los Martes.

Con respecto a los Martes, la distribución del número de operaciones es similar a la de los Lunes: se produce un ascenso del número de operaciones con la apertura de las oficinas bancarias a las 8:15 horas y a las 16:30 horas, se mantienen en el horario laborable y descienden tras su cierre. Además, el número de operaciones en horario de tarde es inferior a las contabilizadas en horario de mañana, pues no todas las oficinas abren en horario de tarde. Son de especial interés los días 17/3/2020, 1/12/2020, 10/8/2021 y 8/3/2022:

- El 17/3/2022 coincide con el primer Martes tras el decreto del estado de alarma por la pandemia del coronavirus. Se contabiliza un número de operaciones inferior a la media.

- Los días 1/12/2020 y 8/3/2022 se produce un aumento del número de operaciones. No es algo sorprendente, pues coincide con primeros de mes y en esos días hay mayor actividad en las oficinas.
- El día 10/8/2021 el número de operaciones realizadas en horario de mañana es inferior al habitual y por las tardes apenas se contabilizan operaciones. Esto es debido a que en el mes de Agosto las oficinas de la entidad abren en horario de verano y la mayoría del personal está de vacaciones.

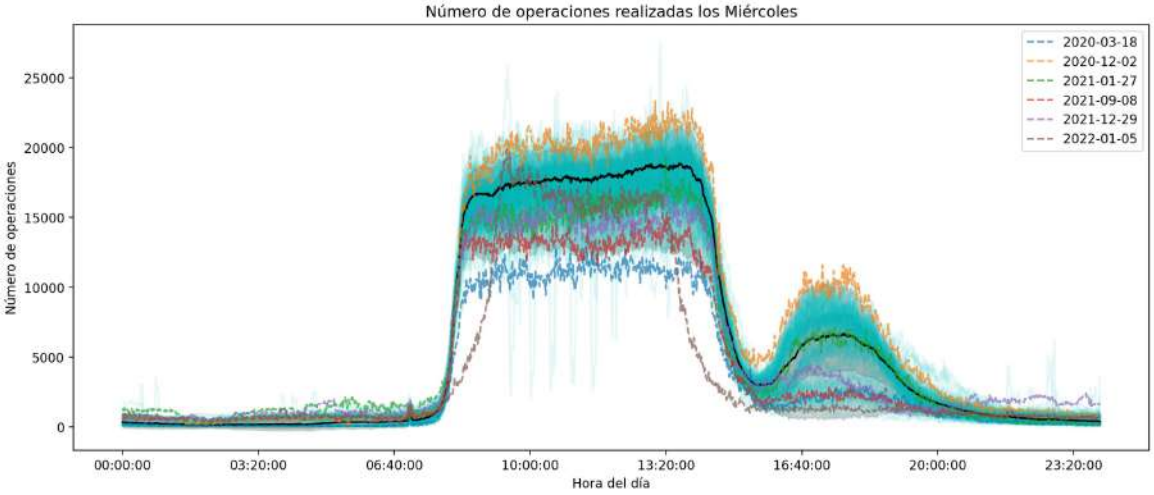


Figura 4.11: Gráfico funcional de los Miércoles.

En el gráfico funcional de los Miércoles vuelve a producirse un comportamiento en la distribución del número de operaciones realizadas similar a la distribución de los Lunes y los Martes: se aprecian las subidas y bajadas en el número de operaciones con la apertura y el cierre de las oficinas. Son de especial interés los días 18/3/2020, 2/12/2020, 8/9/2021, 29/12/2021 y 5/1/2022:

- El día 18/3/2020 se corresponde con el primer Miércoles tras el decreto del estado de alarma por la pandemia del coronavirus. Se pueden sacar conclusiones análogas a las comentadas en los casos de los Lunes y los Martes.
- El día 2/12/2020 coincide con primeros de mes, que es cuando se produce más actividad en las oficinas. Por ese motivo el número de operaciones es superior a la media y a los restantes días del año.
- Con respecto al día 8/9/2021, además de coincidir con primeros de año y con la vuelta de vacaciones, se aprecia que el número de operaciones es inferior a la media y que apenas de contabilizan en el horario de tarde. Esto es debido a que en el mes de Septiembre las oficinas bancarias no abren por las tardes.
- El día 29/12/2021 se contabiliza un número de operaciones menor que la media sobre todo en el horario de tarde, lo que puede ser debido a la época navideña, pues las oficinas no abren por la tarde.
- En cuanto al día 5/1/2022, correspondiente con primeros de mes, se observa que la actividad en cuanto al número de operaciones es elevada tras las vacaciones de Navidad. Además, el número de operaciones crece y decrece lentamente, no como en el comportamiento habitual. También se aprecia que por la tarde apenas se contabilizan operaciones, pues las oficinas no abren por la tarde en la época navideña.

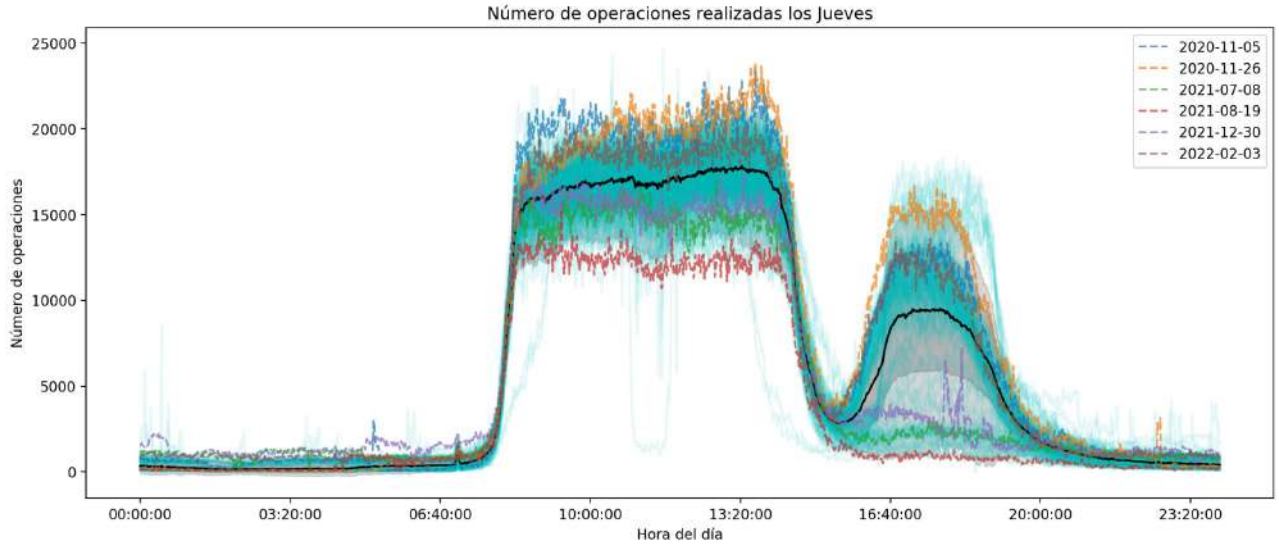


Figura 4.12: Gráfico funcional de los Jueves.

Con respecto a la distribución del número de operaciones en los Jueves, se aprecia de nuevo el comportamiento de los Lunes, Martes y de los Miércoles, por lo que no se entra en mayor detalle. En los Jueves, cabe destacar el comportamiento del número de operaciones en los días 5/11/2020, 26/11/2020, 19/8/2021, 8/7/2021, 30/12/2021 y 3/2/2022:

- En los días 5/11/2020, 26/11/2020 y 2/3/2022, la distribución del número de operaciones es superior a la media en las horas centrales del día, con mayor intensidad en el horario de tarde.
- Los restantes días apenas se contabilizan operaciones por la tarde, debido a que las oficinas no permanecen abiertas: en los meses de Julio y Agosto por el horario de verano y en Dicciembre por la época navideña.

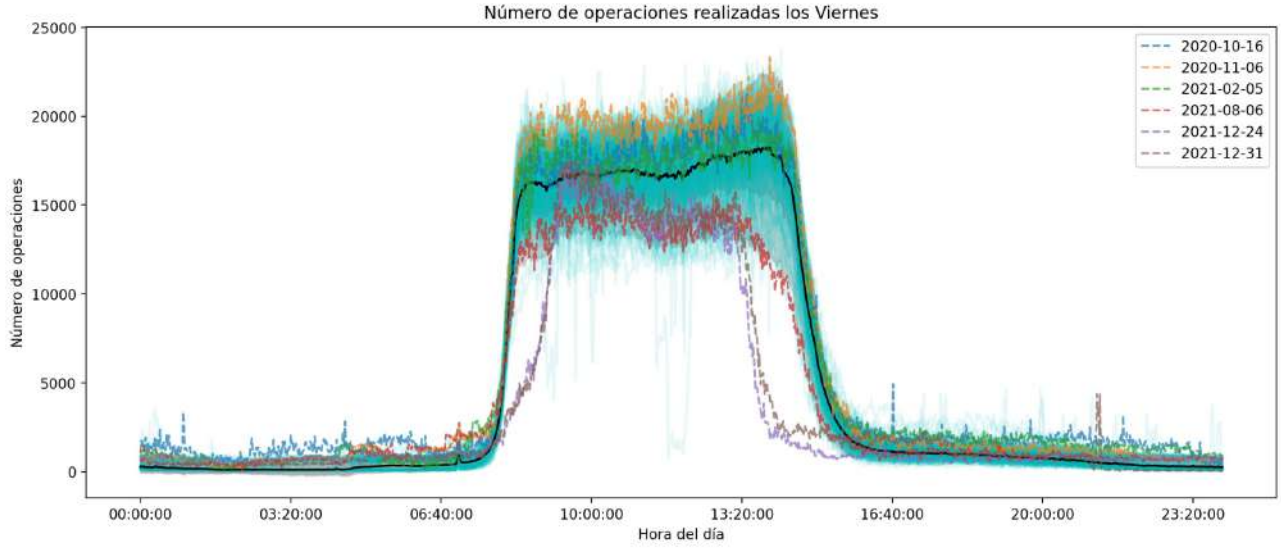


Figura 4.13: Gráfico funcional de los Viernes.

Si observamos el gráfico funcional de los Viernes, la distribución del número de operaciones es diferente a la distribución de los restantes días por semana (Lunes a Jueves). Se aprecia que el número de operaciones sube con la apertura de las oficinas bancarias a las 8:15 horas, se estabiliza a lo largo de la mañana y con el cierre a las 14:15 horas el número de operaciones desciende hasta alcanzar valores cercanos a 0. Esto es debido a que los Viernes es el único día de la semana en el que las oficinas bancarias de la entidad no abren en horario de tarde. Además, al igual que ocurría con el número de operaciones en los restantes días por semana, durante las horas de la madrugada apenas se contabilizan operaciones (son relativas a la Banca Electrónica y a la Banca Móvil). Cabe destacar el comportamiento de los días 6/11/2020, 24/12/2021 y 31/12/2021. Con respecto al día 6/11/2020, el número de operaciones es superior a la media, lo cual no es nada extraño pues a primeros de mes es cuando se produce un mayor movimiento de operaciones. Con respecto a los días 24 y 31 de Dicciembre, correspondientes con vísperas de festivo, nótese que el número de operaciones asciende más tarde y desciende mucho antes que el comportamiento habitual. Esto puede ser debido a que, en la época de Navidad, el horario de apertura de las oficinas sea más reducido con respecto a otras épocas del año.

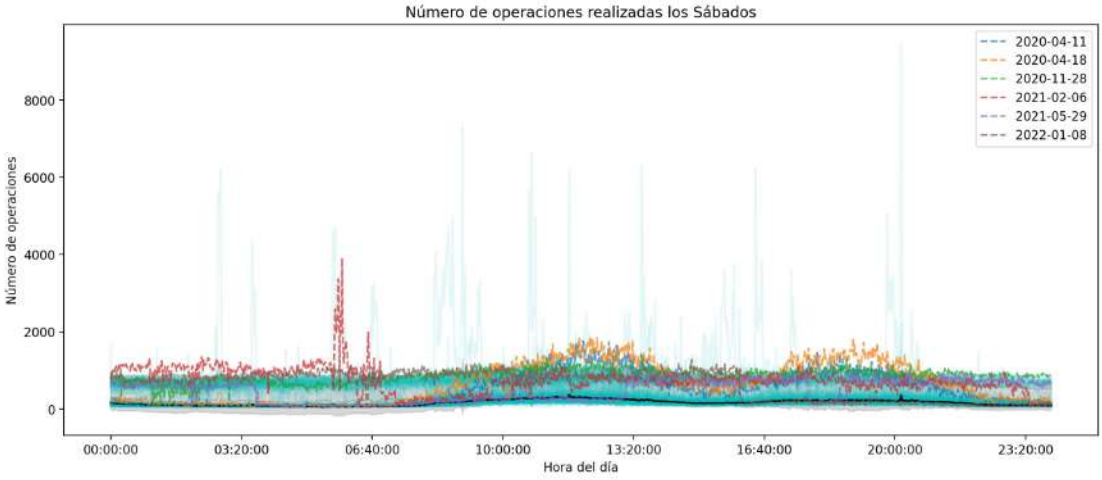


Figura 4.14: Gráfico funcional de los Sábados.

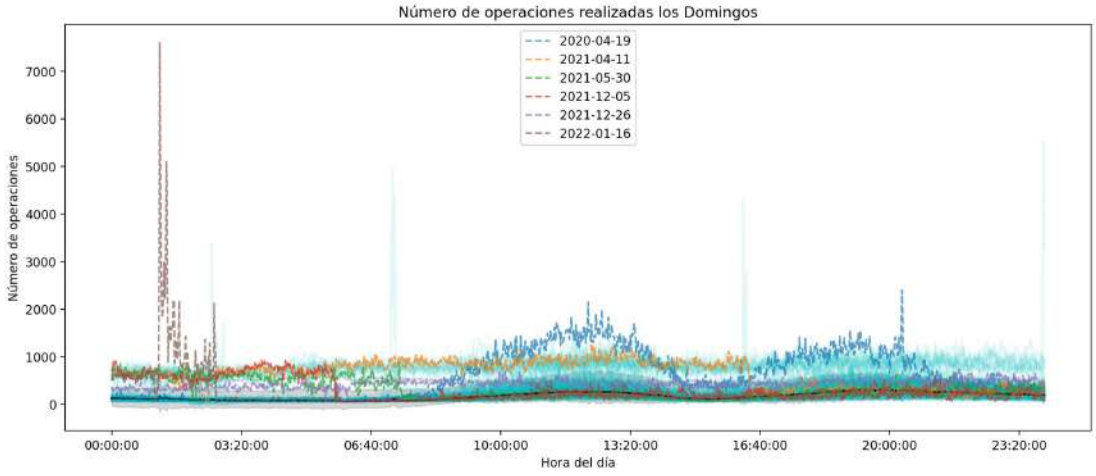


Figura 4.15: Gráfico funcional de los Domingos.

Para finalizar con el análisis de los gráficos funcionales, comentar que la distribución del número de operaciones realizadas durante los fines de semana (Sábados y Domingos) es más reducida con respecto a los restantes días de la semana. Las escasas operaciones contabilizadas se corresponden con operaciones realizadas en Banca Electrónica y en Banca Móvil, ya que en los fines de semana no se abren las oficinas bancarias de la entidad.

4.5. Conclusiones

Tras la realización de diferentes gráficos para determinar el comportamiento de la serie temporal del número de operaciones realizadas, llegamos a las siguientes conclusiones:

- La distribución del número de operaciones es diferente dependiendo del día de la semana. Hay diferencias significativas entre los días por semana y los fines de semana. Además, los Viernes a la tarde la distribución del número de operaciones es diferente a la distribución de los restantes días por semana (Lunes a Jueves), pues los Viernes por la tarde las oficinas bancarias de la entidad no abren.

- En los gráficos funcionales hemos visto que, por ejemplo, el número de operaciones realizadas en el mes de Agosto en horario de mañana es inferior al habitual y que por las tardes apenas se contabilizan operaciones. Esto es debido a que en en mes de Agosto las oficinas bancarias sólo abren por las mañanas. Entonces, esto nos sugiere la existencia de una componente estacional anual. Además, dado que a primeros de cada mes se produce un aumento en el número de operaciones con respecto a los valores habituales se puede deducir que la serie temporal posee una componente estacional mensual.

- A partir de los gráficos de la serie temporal en períodos de una semana y de un día, se sugiere la existencia de componente estacional semanal y estacional diaria.

- La serie temporal posee una componente tendencia ya que existe variabilidad en el número de operaciones realizadas a lo largo del año.

- La componente residual de la serie temporal posee numerosas fluctuaciones correspondientes, principalmente, con el inicio y el fin del año y con los días festivos y fines de semana.

Capítulo 5

Resolución del problema con métodos autoexplicativos

En este capítulo se incluyen los resultados del problema de detección de anomalías en operatoria bancaria empleando tres modelos autoexplicativos: el modelo TBATS (*Trigonometric seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal components*), el modelo UCM (*Unobserved Component Model*) y el modelo Prophet. Se comprobará que estos métodos no son adecuados para la detección de anomalías en operatoria bancaria.

5.1. Modelo TBATS

En el análisis exploratorio de datos se ha comprobado que la serie temporal del número de operaciones presenta componente estacional diaria, semanal, mensual y anual, por lo que un modelo ARIMA estacional multiplicativo no sería útil para obtener predicciones, ya que sólo permite modelizar una componente estacional. De este modo, es necesario recurrir a otro tipo de modelos, los llamados modelos TBATS. Estos modelos de predicción en series de tiempo permiten modelizar las series temporales con más de una componente estacional basándose en métodos de suavización exponencial. Además, modeliza la componente tendencia, realiza una transformación de Box-Cox e incluye los errores de un modelo ARMA. Para realizar el ajuste de este tipo de modelos se emplea la función TBATS de la librería tbats incluída en lenguaje de programación python.

A modo de ejemplo, se quieren emplear los datos desde el día 10/1/2022 al día 23/1/2022 para predecir, por horas, los valores de las dos siguientes semana (del 24/1/2022 al 4/2/2022). Dado que los datos de la serie temporal del número de operaciones están a nivel de minutos, el coste computacional del ajuste de los modelos TBATS puede ser elevado, por lo que se ha decidido agrupar los datos correspondientes por horas y realizar las predicciones por cada día (cada 24 horas). Antes de proceder con el ajuste, es necesario comprobar si la serie del número total de operaciones realizadas por cada hora presenta componente estacional semanal y diaria. Para ello, se representan gráficamente los valores del número de operaciones realizadas por cada hora en la semana del 24 al 30 de Enero y para el día 24.

Si se observa el gráfico que representa la evolución del número total de operaciones realizadas por cada hora en la semana del día 24 al 30 de Enero del 2022, se aprecia un comportamiento similar a lo que ocurría con los datos a nivel de minutos: los fines de semana el número de operaciones es prácticamente nulo y durante la semana se produce variabilidad en el número de operaciones, cuya evolución depende del horario de las oficinas bancarias de la entidad. Por lo tanto, se aprecia la existencia de una componente estacional semanal. El gráfico también sugiere la existencia de una componente estacional diaria. Para comprobarlo, debe realizarse la representación gráfica en un día.

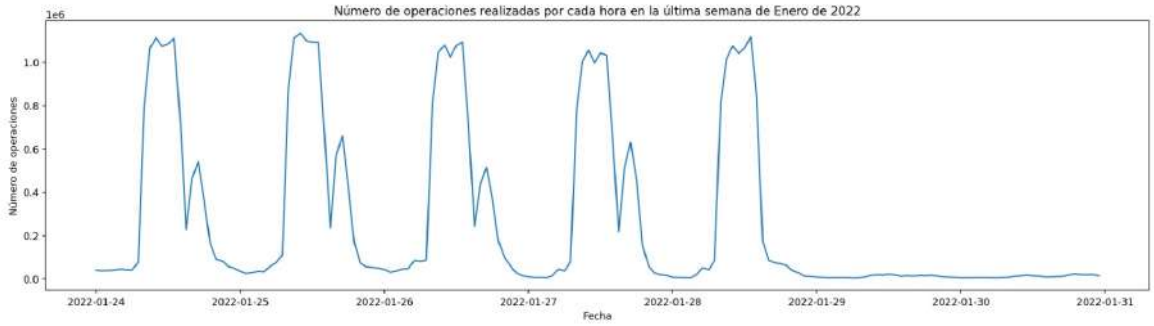


Figura 5.1: Evolución del número total de operaciones realizadas por cada hora en la semana del día 24 al 30 de Enero del 2022.

Si se observa la representación gráfica de la evolución del número total de operaciones realizadas por cada hora en el día 24 de Enero del 2022, vemos un comportamiento similar a lo que ocurría con los datos a nivel de minutos en un día por semana. El número de operaciones es prácticamente nulo hasta las 8 horas, donde el número de operaciones comienza a subir. Con el cierre de las oficinas, a las 14 horas, el número de operaciones desciende hasta la apertura, de nuevo, de las oficinas bancarias de la entidad en horario de tarde, donde vuelve a subir. Tras el cierre de las oficinas en horario de tarde, el número de operaciones desciende hasta alcanzar valores cercanos a 0. Nótese que, al igual que ocurría con los datos a nivel de minuto, el número de operaciones realizadas por las tardes es inferior a las contabilizadas a la mañana. Esto es debido a que no todas las oficinas bancarias de la entidad abren en horario de tarde. Luego, se aprecia la existencia de componente estacional diaria.

Figura 5.2: Evolución del número total de operaciones realizadas por cada hora el día 24/1/2022.

Una vez comprobada la existencia de componentes estacionales semanal y diaria, calculamos las predicciones en períodos de 24 horas y comparamos con los valores reales. La idea es la siguiente: se ajusta un modelo TBATS empleando los datos del número total de operaciones realizadas por horas los días 10 al 23 de Enero, se calculan las predicciones para el día siguiente (24 horas) y se vuelve a ajustar el modelo añadiendo los valores reales correspondientes con el día que se acaba de predecir. Se repite este proceso, calculando las predicciones por días, hasta alcanzar el último día de interés (en este caso, se pararía el proceso tras obtener las predicciones del día 4 de Febrero del año 2022 mediante el ajuste del modelo TBATS con los días 24/1 al 3/2 del 2022).

El siguiente gráfico compara las predicciones obtenidas con el modelo TBATS (siguiendo el proceso que se acaba de definir) con los valores reales del número total de operaciones realizadas por cada hora del día 24 de Enero al 4 de Febrero del año 2022. Son de especial de interés los días 28 de Enero y 2 y 4 de Febrero. En el día 2 de Febrero, las predicciones en el rango de 10 a 13 horas toman valores superiores

al número de operaciones, lo que podría indicar un aumento en el número de clientes que acuden a la oficina. En los días 28 de Enero y 4 de Febrero, correspondientes con un Viernes, las predicciones en horario de tarde toman valores similares a los restantes días laborables, lo que podría indicar un posible aumento del uso de la Banca Electrónica y de la Banca Móvil puesto que, como comentamos en el análisis exploratorio de datos, las oficinas bancarias de la entidad no abren los Viernes por las tardes. Además las predicciones obtenidas en los días 24, 25, 26 y 31 de Enero, toman valores inferiores a los valores reales llegando, en algunos casos a ser 0. Los restantes días las predicciones toman un comportamiento similar a los valores reales del número total de operaciones realizadas por cada hora.

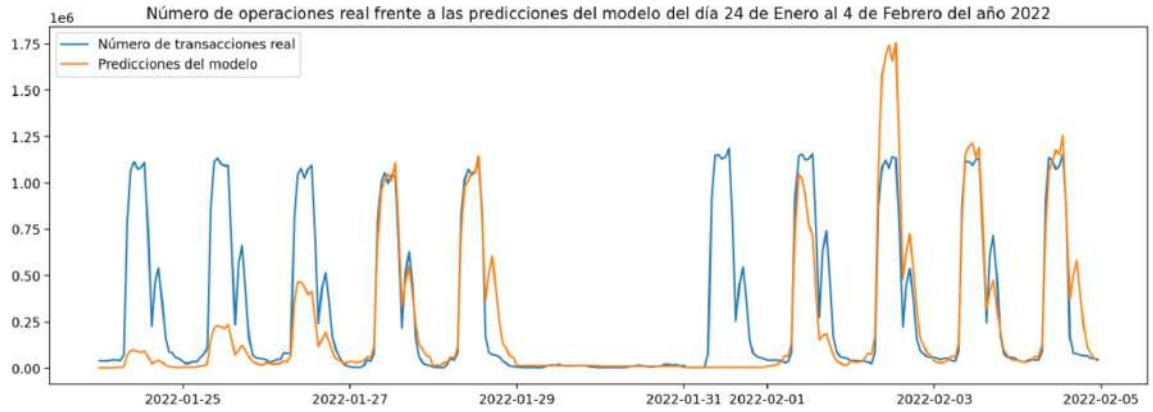


Figura 5.3: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas en períodos de 24 horas empleando el modelo TBATS.

¿Cómo podemos solucionar el problema del cálculo de predicciones en los días 24, 25, 26 y 31 de Enero? Una posible solución sería eliminar los fines de semana del conjunto de datos que se está considerando para ajustar el modelo TBATS ya que parecen tener cierta influencia en la predicción de los primeros días de la semana. Los siguientes gráficos comparan las predicciones obtenidas con el modelo TBATS con los valores reales del número de operaciones realizadas los días 24/1 al 28/1 y 31/1 al 4/2 del año 2022 tras eliminar los fines de semana del conjunto de datos inicial (15, 16, 22, 23, 29 y 30 de Enero). Las predicciones se han obtenido en períodos de 24 horas, siguiendo un proceso como el que se ha realizado anteriormente.

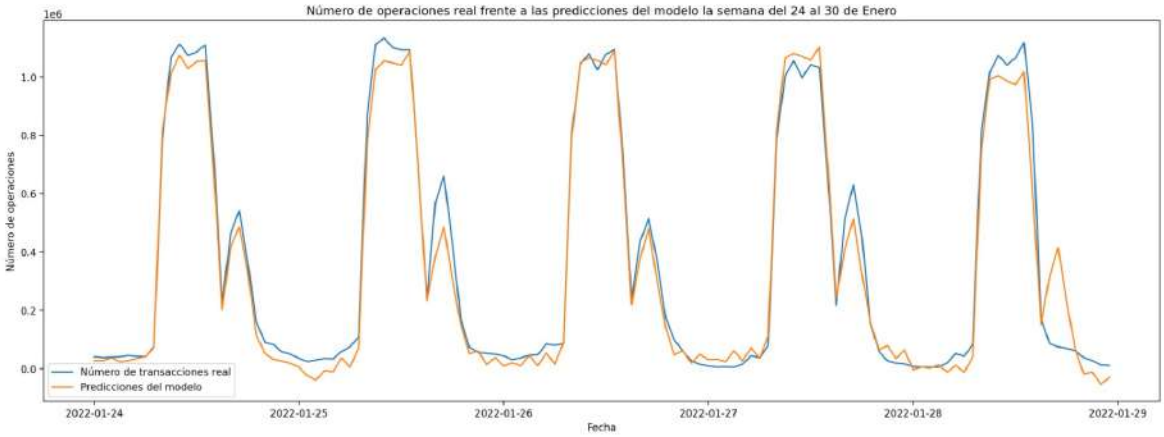


Figura 5.4: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas en períodos de 24 horas del 24/1 al 29/1 empleando el modelo TBATS, tras eliminar los fines de semana.

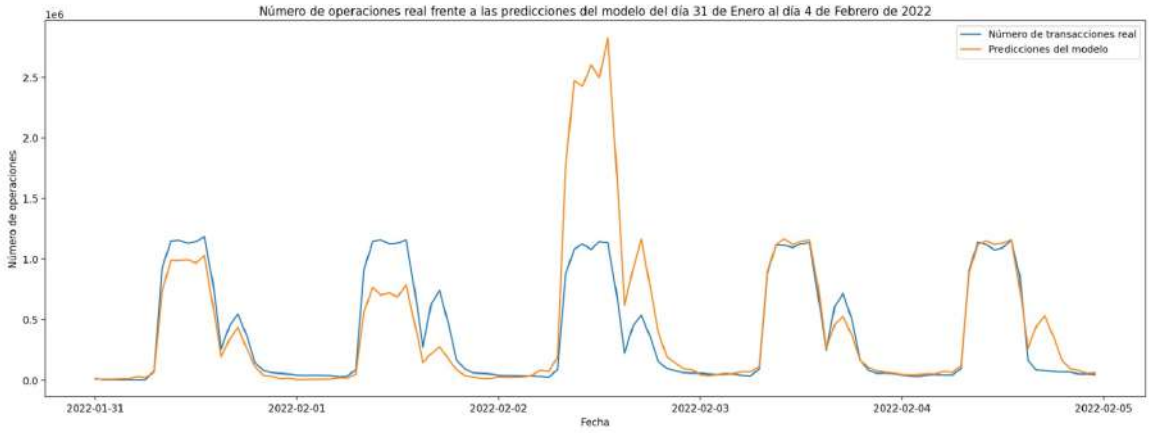


Figura 5.5: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas en períodos de 24 horas del día 31/1 al 4/2 empleando el modelo TBATS, tras eliminar los fines de semana.

Como podemos apreciar, al eliminar los fines de semana del conjunto de datos, las predicciones en los días 24, 25, 26 y 30 de Enero mejoran con respecto a las predicciones obtenidas anteriormente: los valores reales y las predicciones son similares. Con respecto a los Viernes 28 de Enero y 4 de Febrero y el Miércoles 2 de Febrero, se pueden sacar las mismas conclusiones que las comentadas anteriormente: el aumento en el uso de la Banca Móvil y la Banca Electrónica y el aumento en el número de clientes en las oficinas, respectivamente.

Para finalizar con el modelo TBATS se realiza una representación gráfica del error medio absoluto (MAE) y de la Raíz Cuadrada del Error Cuadrático Medio (RMSE) a lo largo del día para los resultados de las predicciones obtenidas con una periodicidad de 24 horas con el ajuste del modelo TBATS tras eliminar los fines de semana.

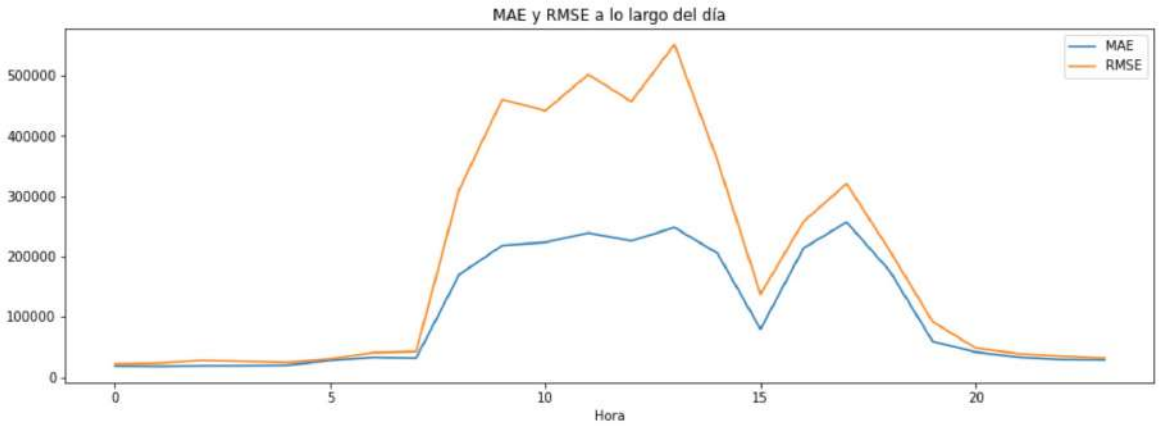


Figura 5.6: Evolución del MAE y del RMSE a largo del día.

Se puede apreciar que el MAE evoluciona según el horario de las oficinas bancarias de la entidad: toma valores cercanos a 0 hasta la apertura de las oficinas, sube a lo largo de la mañana, vuelve a descender hasta que se abren las oficinas por la tarde (produciéndose el pico a las 15 horas) y finalmente vuelve a bajar hasta tomar valores cercanos a 0 tras el cierre de las oficinas por las tardes. El caso del

RMSE es diferente: se observan picos a las 9, 10, 11, 12, 13, 15 y 17 horas ya que el RMSE se ve muy influenciado por las anomalías de la serie temporal.

5.2. Modelo UCM

El modelo UCM (*Unobserved component Model*) realiza una descomposición de la serie temporal en componente tendencia, estacional y cíclica y modeliza los efectos de regresión debido a la serie de predictores. La idea es ajustar un modelo UCM empleando la serie temporal del número de operaciones, a nivel de minutos, de los años 2020 y 2021 para predecir los 3 primeros meses del año 2022. En el lenguaje de programación python se emplea la función tsa.UnobservedComponents de la librería statmodels.

Los siguientes gráficos realizan una comparación entre las predicciones del modelo UCM y los valores reales en el mes de Enero del año 2022 y en la última semana de ese mes.

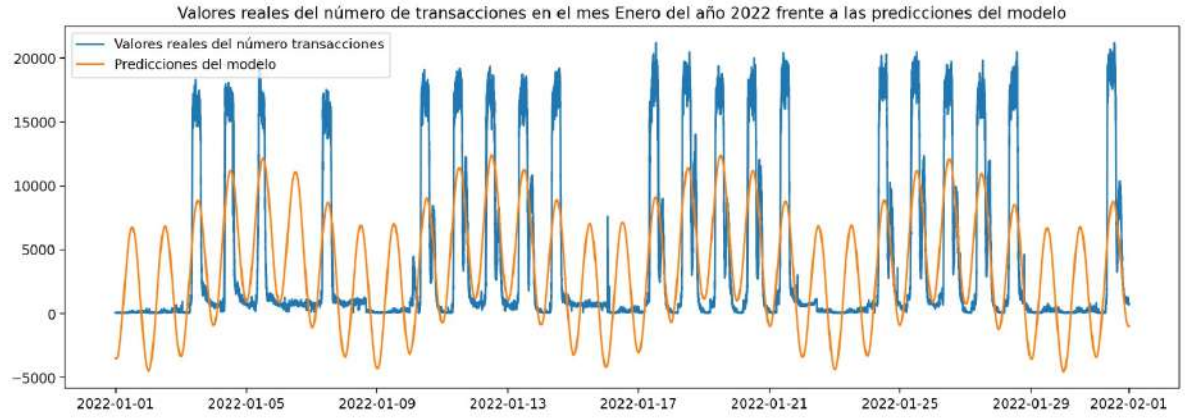


Figura 5.7: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas con el modelo UCM en el mes de Enero del año 2022.

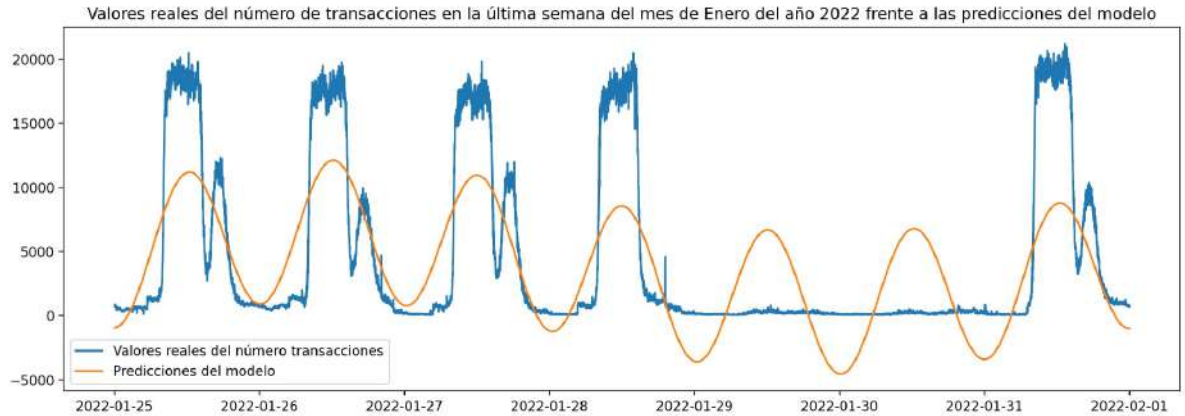


Figura 5.8: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas con el modelo UCM en la última semana de Enero del año 2022 (del 25 al 31).

En ambos gráficos las predicciones del modelo se modelizan como funciones trigonométricas llegando a tomar valores inferiores a 0 durante las horas de la madrugada, especialmente en los fines de semana, lo cual no es nada coherente. Se observa que en los días por semana, las predicciones siempre toman valores inferiores al número real de operaciones en las horas centrales del día y que, los fines de semana, las predicciones superan los valores originales en las horas centrales del día. Por lo tanto este modelo no es adecuado para la detección de las anomalías en operatoria bancaria.

Para finalizar con el estudio del modelo de componentes no observadas, UCM, se realiza una representación gráfica del Error Medio Absoluto y de la Raíz Cuadrada del Error Cuadrático Medio de evaluación a lo largo del día en períodos de 5 minutos.

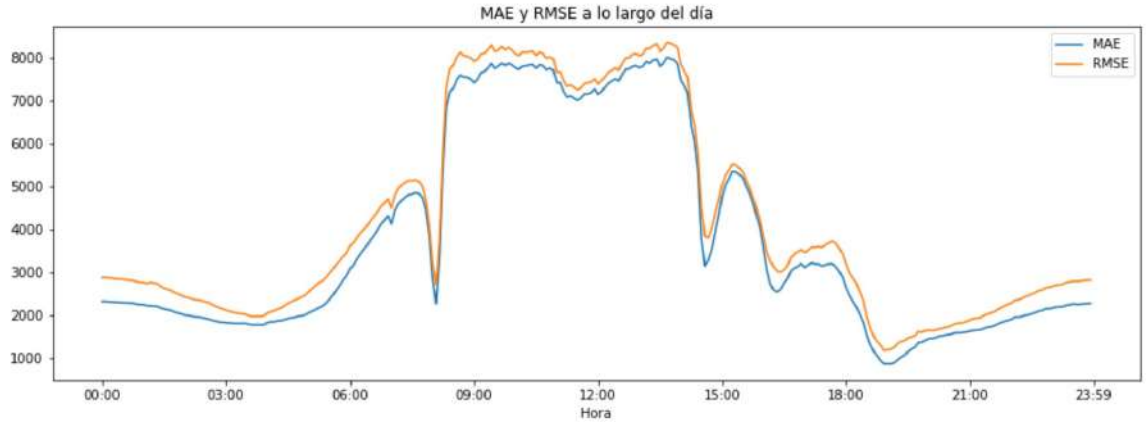


Figura 5.9: Evolución del MAE y del RMSE a lo largo del día.

En la representación gráfica del MAE y del RMSE a lo largo del día se aprecia un comportamiento similar para ambas métricas de evaluación. Se observa la presencia de picos correspondientes con la apertura de las oficinas bancarias de la entidad a las 8:15 horas y con el cierre de las mismas a las 14:15 horas. Con respecto al pico que se produce aproximadamente a las 17 horas, podemos deducir que podría ser debido a la apertura de las oficinas en horario de tarde (recordemos que abren a las 16:30 horas).

5.3. Modelo Prophet

Tras comprobar que el modelo UCM no es lo suficientemente adecuado para la detección de anomalías en operatoria bancaria, vamos a proceder a realizar un ajuste de un nuevo modelo: el llamado modelo Prophet, que modeliza las componentes estacionales de la serie temporal y diferentes efectos del calendario. Cabe destacar que este modelo está implementado en un software de acceso libre con el mismo nombre creado por la empresa Facebook y que en Python puede implementarse empleando la función Prophet de la librería fbprophet (o prophet, si la anterior librería da problemas en la instalacción) de Python.

En primer lugar se realiza el ajuste de un modelo de predicción Prophet empleaando todos los datos a nivel de minuto del número de operaciones realizadas durante los años 2020 y 2021 para predecir los valores de los 3 primeros meses de año 2022. Se realizarán dos tipos de ajustes, que se describen a continuación de forma detallada:

- Un ajuste de modelo Prophet que incluya efectos del calendario debido a los días festivos. Se considerarán como días festivos los festivos a nivel nacional, autonómico y local (ciudad de A

Coruña, puesto que la sede central de la entidad bancaria se encuentra en esta ciudad) del conjunto de datos por completo (en este caso, los años 2020, 2021 y los 3 primeros meses del año 2022). Para incluír estos festivos en el modelo es necesario definir un conjunto de datos llamado "holidays" con dos columnas: "ds", con la fecha correspondiente con el día festivo, y "holiday" con el nombre del día festivo. La lista de los días festivos puede consultarse en el Apéndice A.

- Un ajuste de modelo Prophet que incluya dos componentes estacionales semanal: una componente que dependa de los días por semana y la otra que dependa de los fines de semana. En el ajuste de este modelo no se tendrán en cuenta los días festivos.

Antes de mostrar gráficamente los resultados de las predicciones del modelo con los valores reales, comentar que para realizar el ajuste del modelo hay que modificar el conjunto de datos de forma que tenga dos columnas bien diferenciadas: una columna con la fecha correspondiente con la observación, "ds", y una columna con los valores del número de operaciones, "y". Si al realizar el ajuste no se pasa un conjunto de datos en este formato, Python nos devolverá un error.

Empezaremos analizando el primero de los ajustes. Primeramente, se representan las distintas componentes del modelo: la componente tendencia, la componente debida a los efectos de los festivos, y las componentes estacional semanal y estacional diaria.

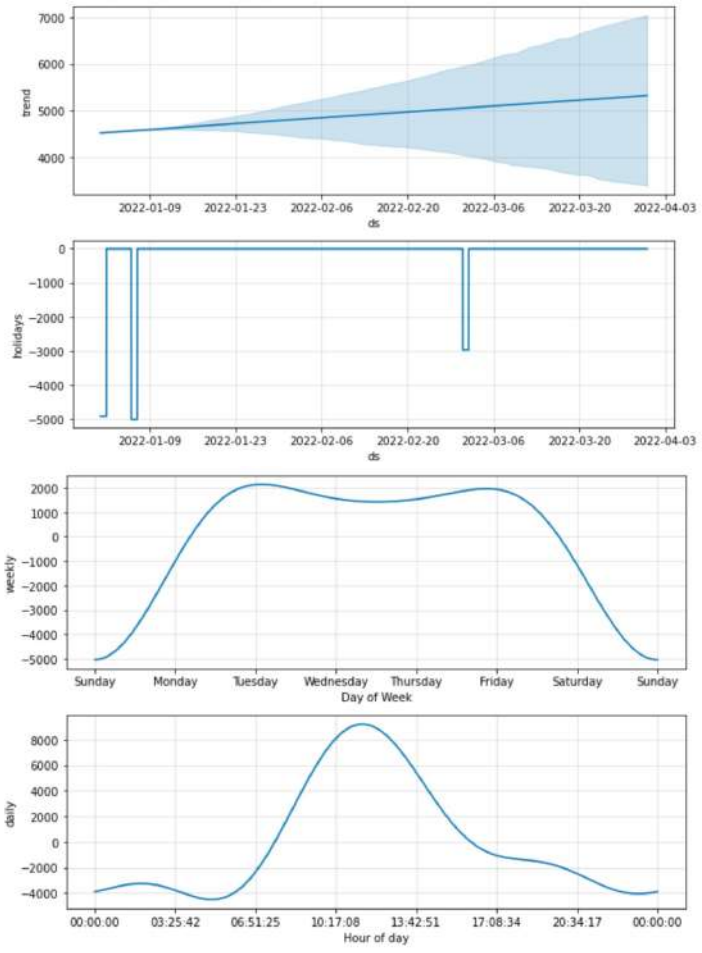


Figura 5.10: Componentes del modelo Prophet. Se incluyen los efectos de los días festivos.

La componente tendencia es creciente, lo que indica un posible aumento del número de operaciones a medida que avanza el año 2022. Con respecto a los días festivos se aprecia un notable descenso del número de operaciones. En cuanto a la componente estacional semanal, que muestra la distribución del número de operaciones en función del día de la semana, se puede apreciar que durante los fines de semana se produce un descenso notable y que, los días por semana, tiende a estabilizarse. Esto no es algo que nos sorprenda pues ya es algo que se apreciaba en el análisis exploratorio de datos cuando se verificó la existencia de componente estacional semanal. En cuanto a la componente estacional diaria, que muestra la distribución del número de operaciones en función de la hora del día, en el horario de apertura de las oficinas bancarias la distribución del número de operaciones tiende a aumentar y, con el cierre, se produce un descenso notable de la distribución. Al igual que en el caso de la componente estacional semanal, no es algo que nos sorprenda pues ya se apreciaba en en análisis exploratorio de datos, cuando se verificó la existencia de la componente estacional diaria.

Tras analizar el comportamiento de las componentes del modelo, se realiza una comparación entre las predicciones obtenidas con el modelo Prophet con los efectos de los festivos y los valores reales.

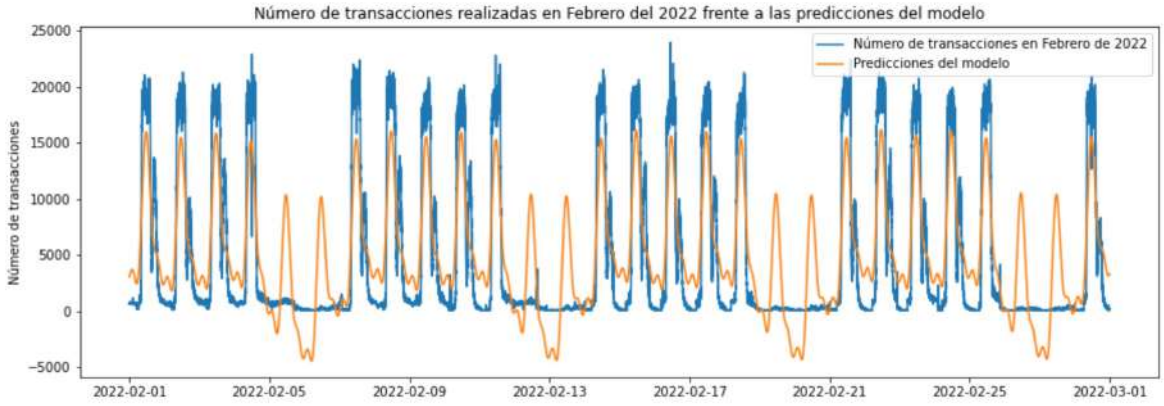


Figura 5.11: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas con el modelo Prophet incluyendo los efectos de los días festivos. Se representan los valores del mes de Febrero de 2022.

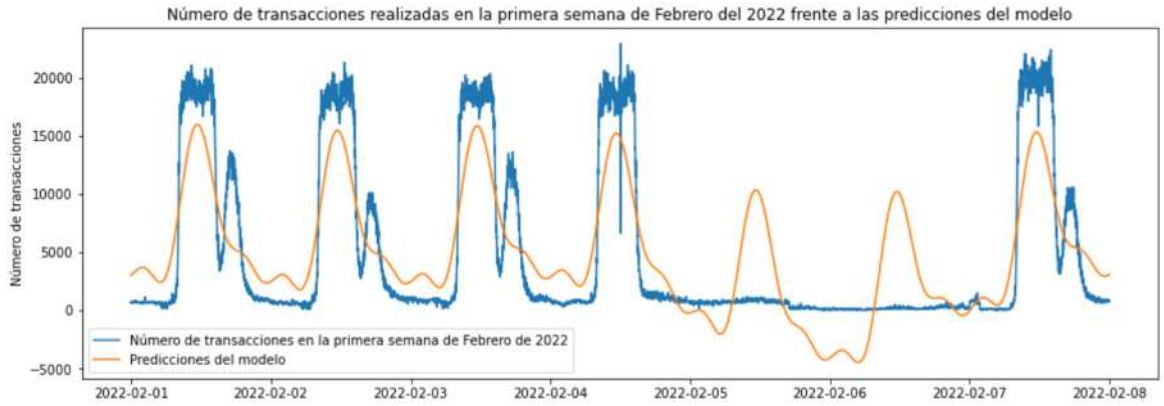


Figura 5.12: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas con el modelo Prophet incluyendo los efectos de los días festivos. Se representan los valores en la primera semana de Febrero de 2022.

En las representaciones gráficas anteriores se puede apreciar que las predicciones mejoran considerablemente con respecto al modelo de componentes no observadas UCM ajustado anteriormente. Las predicciones se siguen modelizando como funciones trigonométricas puesto que las curvas que modelan las componentes estacionales en el modelo Prophet son series de Fourier. Vemos que, en los días por semana, las predicciones en las horas centrales del día toman valores inferiores al número de operaciones real y que, sin embargo, en las horas de madrugada las predicciones superan el número de operaciones real. En cuanto a los fines de semana se aprecia que la predicción es elevada en las horas centrales del día, al igual que ocurría con el modelo UCM. Vemos, por lo tanto, que el modelo Prophet parece más adecuado que el modelo UCM para obtener las predicciones del número de operaciones.

Una vez determinado el comportamiento de las predicciones del modelo Prophet ajustado incluyendo los efectos de los días festivos, vamos a determinar cómo evoluciona el Error Medio Absoluto y la Raíz Cuadrada del Error Cuadrático Medio a lo largo del día en períodos de 5 minutos.

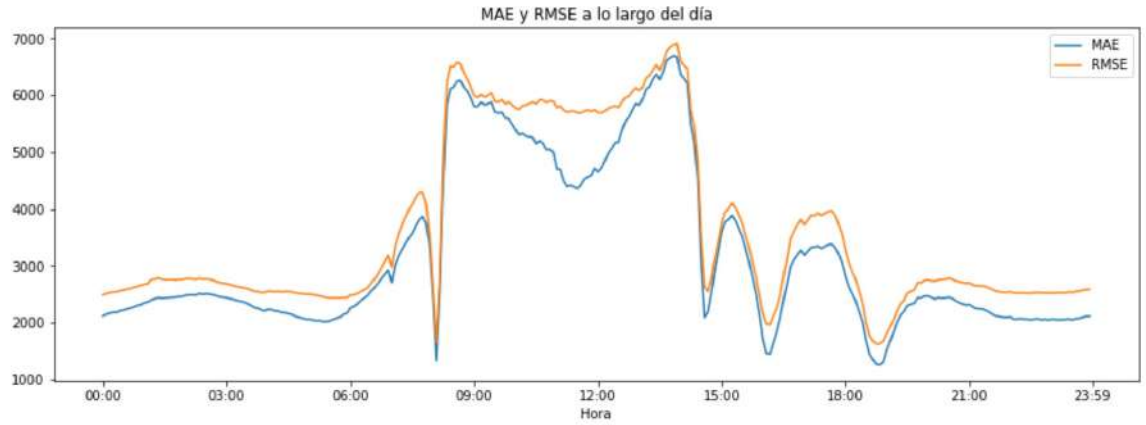


Figura 5.13: Evolución del MAE y del RMSE a lo largo del día para el modelo Prophet que incluye los efectos de los festivos.

Como podemos apreciar en la representación gráfica anterior, ambas métricas de evaluación tienen un comportamiento similar y, en el horario de apertura de las oficinas bancarias en horario de mañana hay una pequeña diferencia entre ambas: el RMSE se estabiliza a lo largo de la mañana mientras que el RMSE sufre una pequeña caída y, posteriormente, vuelve a subir. En general, el MAE toma valores inferiores al RMSE, ambas métricas superan el valor de 1000 y alcanzxan los valores más elevados a lo largo de la mañana. Nótese que se producen picos notables en el horario de apertura y cierre de las oficinas por las mañanas (a las 8:15 y 14:15 horas, respectivamente) y con el horario de apertura y cierre de las oficinas bancarias de la entidad en horario de tarde (a las 16:30 y 18:45 horas, respectivamente). Se puede apreciar que los valores del MAE y del RMSE en horario de tarde son inferiores a los valores de las métricas de evaluación en horario de mañana lo que no es sorprendente, ya que no todas las oficinas de la entidad abren en horario de tarde. Comentar que, también se produce un pico notable antes de la apertura de las oficinas en horario de tarde.

A continuación, se realiza un nuevo ajuste del modelo Prophet, sin considerar los efectos de los días festivos y distinguiendo dos tipos de componentes estacionales semanales: una componente que dependa de los días por semana (Lunes a Viernes) y otra que dependa de los fines de semana. Vamos a ver, de forma gráfica, si este nuevo ajuste mejora las predicciones del anterior modelo. El procedimiento a seguir será similar. Primeramente veremos cómo se comporta el modelo representando las distintas componentes: la componente tendencia, la componente estacional diaria y las dos componentes estacionales semanales.

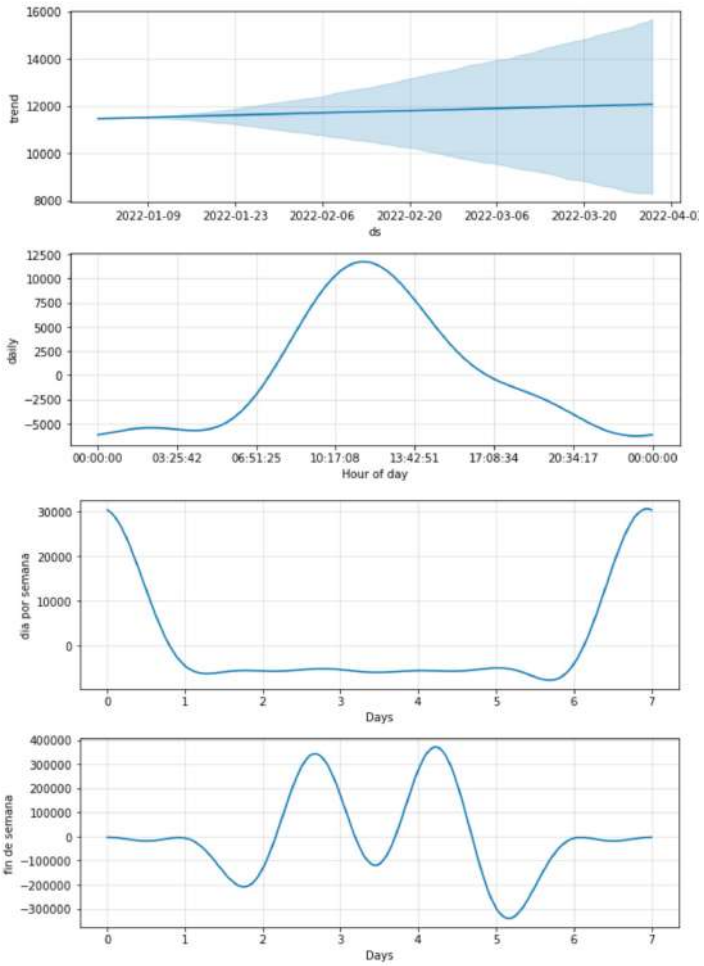


Figura 5.14: Componentes del modelo Prophet con dos componentes estacionales semanales: una que depende de los días por semana y la otra que depende de los fines de semana.

La componente tendencia es ligeramente creciente: crece en menor medida que en el anterior modelo. Con respecto a la componente estacional diaria, se obtienen las mismas conclusiones que en modelo anterior. En cuanto a la componente estacional semanal que depende de los días por semana vemos que la distribución del número de operaciones desciende los Domingos, permanece constante durante los días por semana y sube los Sábados. Finalmente, con respecto a la componente estacional semanal que depende de los fines de semana, apreciamos lo contrario a la componente semanal que depende de los días por semana: la distribución del número de operaciones permanece constante los fines de semana y tiene mayor variabilidad en los días por semana.

Una vez analizado el comportamiento de las diferentes componentes del modelo Prophet con dos componentes estacionales semanales, se comparan las predicciones obtenidas con el modelo con los valores reales del número de operaciones en el mes de Febrero del 2022 y en la última semana de ese mes.

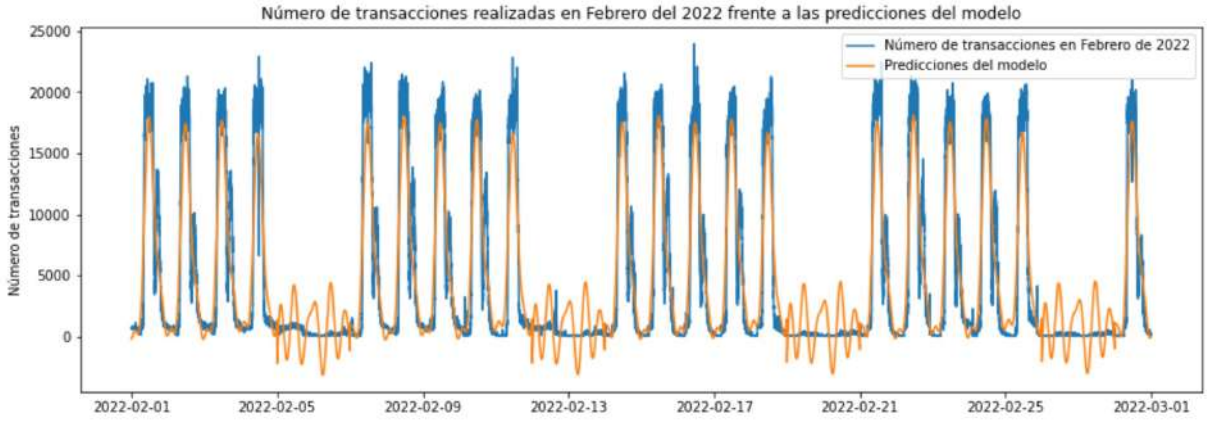


Figura 5.15: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas con el modelo Prophet con dos componentes estacionales semanales. Se representan los valores del mes de Febrero de 2022.

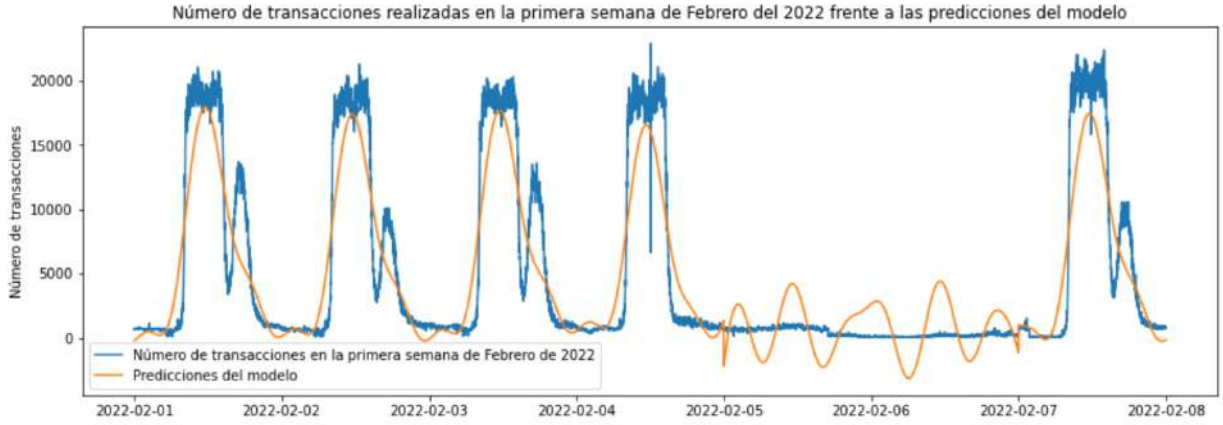


Figura 5.16: Comparación gráfica del número de operaciones real frente a las predicciones obtenidas con el modelo Prophet con dos componentes estacionales semanales. Se representan los valores en la primera semana de Febrero de 2022.

Gráficamente las predicciones de este modelo mejoran con respecto a las predicciones obtenidas con el modelo Prophet que incluye los efectos de los días festivos. En los días por semana los valores de la predicción se asemeja al número de operaciones real, exceptuando los valores en horario de tarde (las predicciones decrecen mientras que los valores reales dependen del horario de apertura de las oficinas bancarias por las tardes). En cuanto a los fines de semana, las predicciones en las horas centrales del día toman valores inferiores a 5000 operaciones, a diferencia de lo que ocurría con el anterior modelo, cuyas predicciones alcanzaban valores cercanos a 10000 operaciones. Vemos, por lo tanto, que gráficamente este modelo es mejor que el modelo Prophet ajustado incluyendo los efectos de los días festivos. De todos modos, las predicciones del modelo se siguen comportando como funciones trigonométricas, por lo que el modelo no parece adecuado para la detección de anomalías en operatoria bancaria.

Para finalizar con el estudio del modelo Prophet, al igual que se ha realizado con el ajuste del modelo Prophet incluyendo los efectos de los festivos, se realiza un estudio detallado de la evolución del Error Medio Absoluto y de la Raíz Cuadrada del Error Cuadrático Medio a lo largo del día en

períodos de 5 minutos para el modelo que se acaba de ajustar. Podemos ver representada la evolución de las métricas de evaluación en la siguiente gráfica:

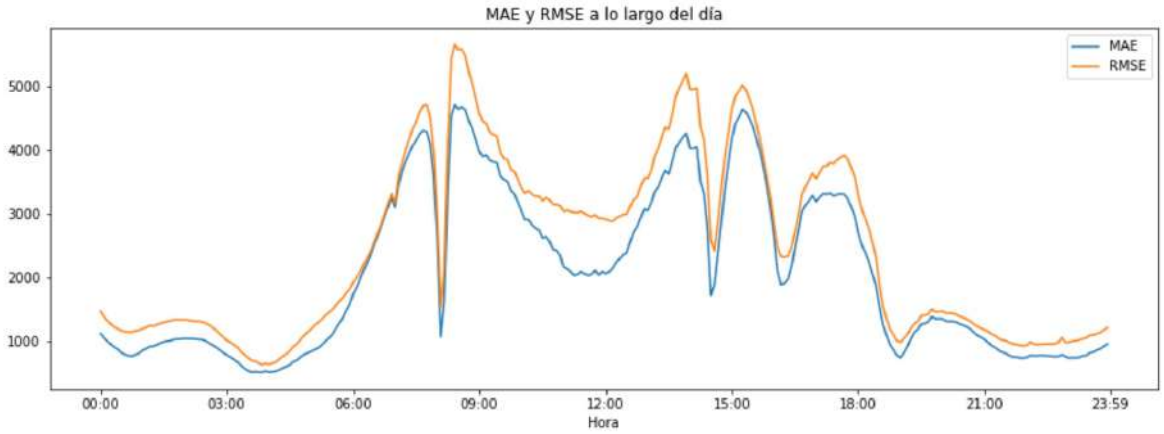


Figura 5.17: Evolución del MAE y del RMSE a lo largo del día en períodos de 5 minutos para el modelo Prophet que incluye dos componentes estacionales semanales.

Al igual que ocurría en el anterior ajuste, las métricas de evaluación tienen un comportamiento similar a lo largo del día y el valor del MAE es ligeramente inferior a los valores del RMSE. Se aprecia el siguiente comportamiento:

- En las horas de madrugada los valores de las métricas de evaluación no varían mucho a lo largo de la noche y toman valores no superiores a 1500.
- A partir de las 6:00 horas, las métricas de evaluación ascienden hasta alcanzar valores cercanos a 5000 antes de la apertura de las oficinas bancarias en horario de mañana (a las 8:15 horas). En ese instante, ambas métricas sufren una caída brusca llegando a tomar valores cercanos a 1000. Tras la apertura de las oficinas, los valores del MAE y del RMSE ascienden superando el valor de 5000 aproximadamente a las 9:00 horas.
- A lo largo de la mañana las métricas se estabilizan y se produce un descenso ligero de las mismas (más notable en el caso del MAE). Con el cierre de las oficinas a las 14:15 horas y, antes de la apertura de las mismas en horario de tarde a las 16:30 horas, observamos un pico notable en ambas métricas.
- Con la apertura de las oficinas en horario de tarde se produce un pequeño ascenso de las métricas y se estabilizan a lo largo de la tarde. Después se produce un descenso hasta el cierre a las 18:45 horas.
- Tras el cierre, ambas métricas se estabilizan y toman valores inferiores a 1500, tal y como ocurría en las horas de madrugada.

Capítulo 6

Resolución del problema con modelos de Aprendizaje Estadístico

En este capítulo se incluyen los resultados del problema de detección de anomalías empleando métodos de Aprendizaje Estadístico. Se consideran el modelo de redes neuronales LSTM y un modelo ensamblado (SackedEnsemble), el mejor modelo obtenido empleando la interfaz H2oAutoMl de Python.

Para resolver el problema de detección de anomalías en operatoria bancaria empleando modelos de predicción a tiempo real, al conjunto de datos del número de operaciones se le incluyen las siguientes variables adicionales: hora del día, día del año, día de la semana y una variable indicadora de si el día es laborable o no. Además, a petición de la empresa, dado que la serie temporal es dependiente del tiempo, deben incluirse los valores de los lags de 30 minutos, que se emplearán para calcular las predicciones (se predicen los valores en función de los 30 minutos anteriores). Estos modelos se ajustan empleando una muestra de entrenamiento y se determina la mejor combinación de hiperparámetros evaluando el modelo en una muestra de validación. En el caso de los modelos implementados en la interfaz H2oAutoMl de Python, se emplea la metodología de validación cruzada. Una vez obtenida la mejor combinación de hiperparámetros, se evalúa el modelo en la muestra de test y se calculan las predicciones. La muestra de test se corresponde con los datos del número de operaciones realizadas durante los 3 primeros meses del año 2022. Con respecto a los años 2020 y 2021, una vez añadidas las variables adicionales y los lags de 30 minutos, se divide, de forma aleatoria, la muestra en entrenamiento y validación (considerando el 90 % y el 10 % de los datos, respectivamente), fijando una semilla para garantizar reproducibilidad.

6.1. Modelo de redes neuronales LSTM

En primer lugar, cabe destacar que, para la resolución del problema empleando este tipo de modelos, se han considerado dos etapas:

- Una etapa de entrenamiento para determinar la mejor combinación de hiperparámetros del modelo de redes neuronales LSTM.
- Una etapa de evaluación para calcular las predicciones del mejor modelo obtenido en la etapa de entrenamiento.

En la etapa de entrenamiento se realiza un ajuste de varios modelos de redes neuronales LSTM y se evalúan los respectivos modelos sobre la muestra de validación. El mejor modelo será aquel cuya combinación de hiperparámetros nos proporcione el menor valor del Error Medio Absoluto (MAE). Una vez determinada la mejor combinación de hiperparámetros, se evalúa el correspondiente modelo

en la muestra de test, obteniendo las respectivas predicciones.

Los hiperparámetros que se consideran en los modelos de redes neuronales son el número de nodos (o neuronas) y el *batch size*, descritos en la teoría. A mayores se considera un parámetro con denominado *dropout* que se emplea para evitar el sobreajuste (*overfitting*) del modelo. Los valores de los hiperparámetros que se han considerado en la etapa de entrenamiento del modelo son los siguientes:

- neuronas: 15, 20 y 30.
- *batch size*: 200, 300, 500 y 1000.
- *dropout*: 0.15, 0.20 y 0.25.

resultando en un total de 45 modelos a ajustar. Los resultados del Error Medio Absoluto de entrenamiento, denotado por train MAE, y de la raíz cuadrada del error cuadrático medio de entrenamiento, denotado por train RMSE, pueden consultarse en la tabla incluida en el Apéndice B. Se puede comprobar que el modelo con menor Error Medio Absoluto, 186.02, es el modelo ajustado con 20 neuronas, 1000 de *batch size* y 0.15 de *dropout*.

Durante el proceso de entrenamiento de los modelos, además de obtener la mejor combinación de hiperparámetros, se calcula el Error Medio Absoluto y la Raíz Cuadrada del Error Cuadrático Medio a lo largo del día en períodos de 5 minutos. Los valores del MAE a lo largo del día del mejor modelo obtenido servirán para la obtención de los límites para determinar las observaciones que se consideran anomalías. La siguiente gráfica muestra la evolución del MAE y del RMSE de entrenamiento a lo largo del día en períodos de 5 minutos.

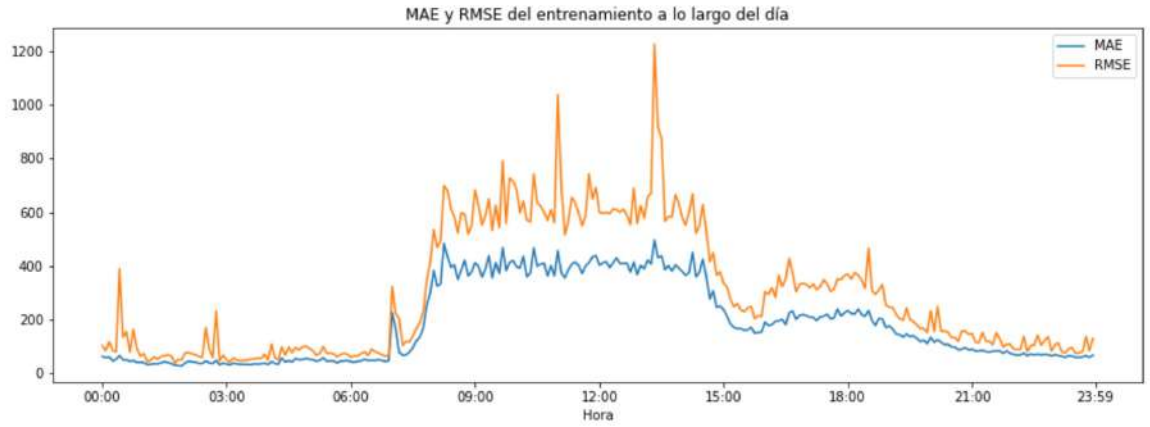


Figura 6.1: Evolución del Error Medio Absoluto (MAE) y de la Raíz Cuadrada del Error Cuadrático Medio (RMSE) de entrenamiento a lo largo del día en períodos de 5 minutos del mejor modelo LSTM.

Se puede apreciar que el comportamiento del MAE a lo largo del día es similar a la distribución del número de operaciones realizadas en un día por semana:

- El MAE toma valores cercanos a 0 las horas de la madrugada.
- Con la apertura de las oficinas bancarias en horario de mañana, el valor del MAE aumenta hasta estabilizarse en valores cercanos a 400.
- Tras el cierre de las oficinas bancarias de la entidad a las 14:15 horas, el MAE sufre una pequeña caída y sube ligeramente hasta tomar valores entre 200 y 400 durante el horario de tarde de las oficinas.

- Tras el cierre de las oficinas en horario de tarde, el MAE desciende hasta tomar valores similares a las horas de madrugada.

Con respecto al RMSE, se aprecia que toma valores superiores al MAE especialmente en las horas centrales del día y se aprecian picos con mayor intensidad que en el caso del MAE. Esto es debido a que el RMSE se ve muy influenciado por las anomalías del conjunto de datos de validación con el que se realiza la etapa de entrenamiento del modelo.

Tras la elección de la mejor combinación de hiperparámetros y el cálculo del Error Medio Absoluto a lo largo del día en períodos de 5 minutos, se pasa a la siguiente etapa, la evaluación del modelo empleando los datos de test y se calculan las predicciones, teniendo en cuenta los valores de los 30 minutos anteriores (lags). Los resultados del Error Medio Absoluto de evaluación y de la Raíz Cuadrada del Error Cuadrático Medio de evaluación para todos los modelos ajustados pueden consultarse en la tabla incluida en el Apéndice B. En esta etapa también se obtienen los valores del Error Medio Absoluto y de la Raíz Cuadrada del Error Cuadrático Medio a lo largo del día en períodos de 5 minutos, cuyos resultados se representan en la siguiente gráfica:

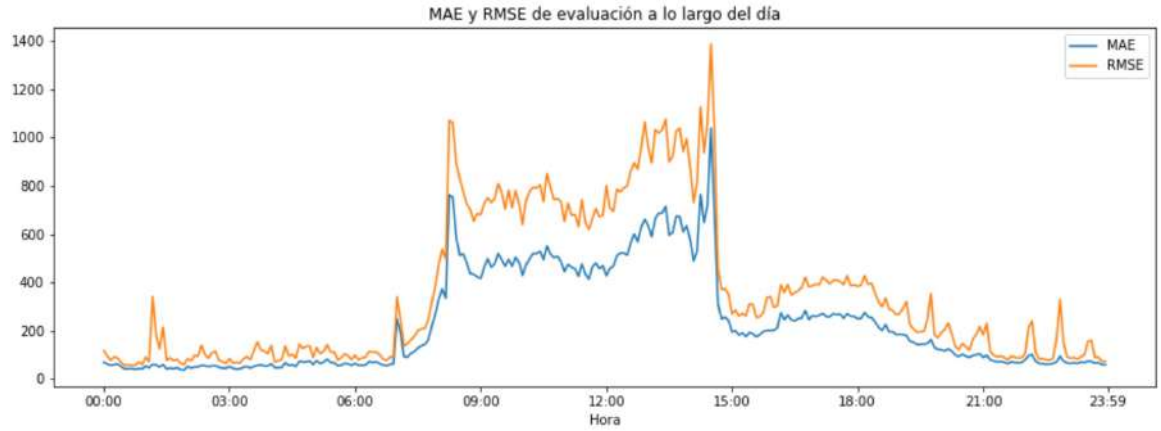


Figura 6.2: Evolución del Error Medio Absoluto (MAE) y de la Raíz Cuadrada del Error Cuadrático medio (RMSE) de evaluación a lo largo del día en períodos de 5 minutos del mejor modelo LSTM.

Se puede apreciar que el comportamiento del MAE y del RMSE de evaluación a lo largo del día es similar al comportamiento del MAE y del RMSE de entrenamiento a lo largo del día y que, especialmente en las horas centrales del día, toman valores superiores a los resultados obtenidos en la etapa de entrenamiento. Nótese que en este caso, se producen picos notables durante el horario de apertura y cierre de las oficinas bancarias a partir de las 8:15 y las 14:30 horas, respectivamente. Además, al igual que ocurría con el RMSE de entrenamiento a lo largo del día, se producen picos con mayor intensidad debido a que esta métrica de evaluación se ve muy influenciada por las anomalías que se producen en el conjunto de datos de test sobre el que se evalúa el modelo.

Una vez calculadas las predicciones del modelo y visto el comportamiento del MAE y del RMSE a lo largo del día en períodos de 5 minutos en las etapas de entrenamiento y de evaluación, se procede con el cálculo de las anomalías. Tal y como se ha comentado en el capítulo de la descripción del problema, para la obtención de las anomalías se calcula la diferencia entre las predicciones el modelo y los valores reales del número de operaciones, lo que denominaremos remainder, y se determina si supera o se encuentra por debajo de un umbral que suele establecerse como un valor de ±sensibilidad multiplicado por el MAE de entrenamiento a lo largo del día en períodos de 5 minutos. Como valores habituales para la detección de anomalías suelen establecerse los umbrales de ±3 y ±6 MAE. En particular, si se

establece un valor de sensibilidad de 7, el modelo LSTM ajustado con 20 neuronas, 1000 de *batch size* y 0.15 de *dropout* detecta un total de 11 anomalías en los 3 primeros meses del año 2022, indicadas en la siguiente tabla:

Fecha	Día de la semana	Hora de inicio	Hora de fin	Duración total anomalía
20 de Enero	Jueves	15:31	16:47	1:16
28 de Enero	Viernes	14:12	19:15	5:03
4 de Febrero	Viernes	12:04	14:35	2:31
14 de Febrero	Lunes	9:46	19:50	10:04
17 de Febrero	Jueves	15:55	16:23	0:28
22 de Febrero	Martes	16:13	17:45	1:32
24 de Febrero	Jueves	8:48	19:46	10:58
10 de Marzo	Jueves	15:54	17:27	1:33
22 de Marzo	Martes	15:51	17:12	1:21
28 de Marzo	Lunes	12:57	17:20	3:23
31 de Marzo	Jueves	8:29	18:03	9:34

Cuadro 6.1: Anomalías detectadas empleando el modelo de redes neuronales LSTM.

Se puede apreciar que las anomalías detectadas se producen en días laborables: la mayoría de ellas se inician en horario de apertura de las oficinas por las mañanas, a lo largo de la mañana y con el cierre de las mismas mientras que, otras se inician en horas cercanas a la apertura de las oficinas en horario de tarde. Recordemos que los horarios de apertura de las oficinas por las mañanas es de 8:15 a 14:15 horas y, por las tardes, de Lunes a Jueves, de 16:30 a 18:45 horas.

Dado que el número de anomalías es elevado, en lugar de realizar todas las representaciones gráficas de las anomalías, se realiza la representación de una de las más relevantes como, por ejemplo, la detectada el día 28 de Marzo. La representación gráfica de todas las anomalías detectadas con el modelo LSTM se encuentra en en el Apéndice C.

En la representación gráfica de la anomalía detectada el día 28 de Marzo, se aprecia que las predicciones (en color naranja) son mucho más suaves que los valores del número de operaciones real (en color azul). La diferencia entre las predicciones y los valores reales, remainder, es bastante notable en la hora de inicio y en la hora de fin de la anomalía (12:57 y 17:20 horas, respectivamente) superando el umbral ± 7 MAE. Se aprecia que la predicción se desvía bastante de los valores reales después de la hora de inicio de la anomalía y, con el cierre de las oficinas la diferencia disminuye hasta mantenerse cercana a 0. Cuando se produce la apertura de las oficinas en horario de tarde, a las 16:30 horas, el remainder aumenta lentamente hasta alcanzar una diferencia notable en la hora de fin de la anomalía.

En resumen, se aprecia cómo la predicción es más suave que los valores reales de anomalía y que, en las horas de inicio y fin de la anomalía es cuando el valor del remainder supera el umbral de ±7 MAE. En particular, el valor del remainder se mantiene cercano a 0 y dentro del umbral en las horas de cierre de las oficinas. Estas conclusiones son análogas para las restantes anomalías detectadas. Nótese que en la hora de inicio de la anomalía se produce una caída notable que podría interpretarse como una caída en las herramientas que emplean los gestores para contabilizar las operaciones.

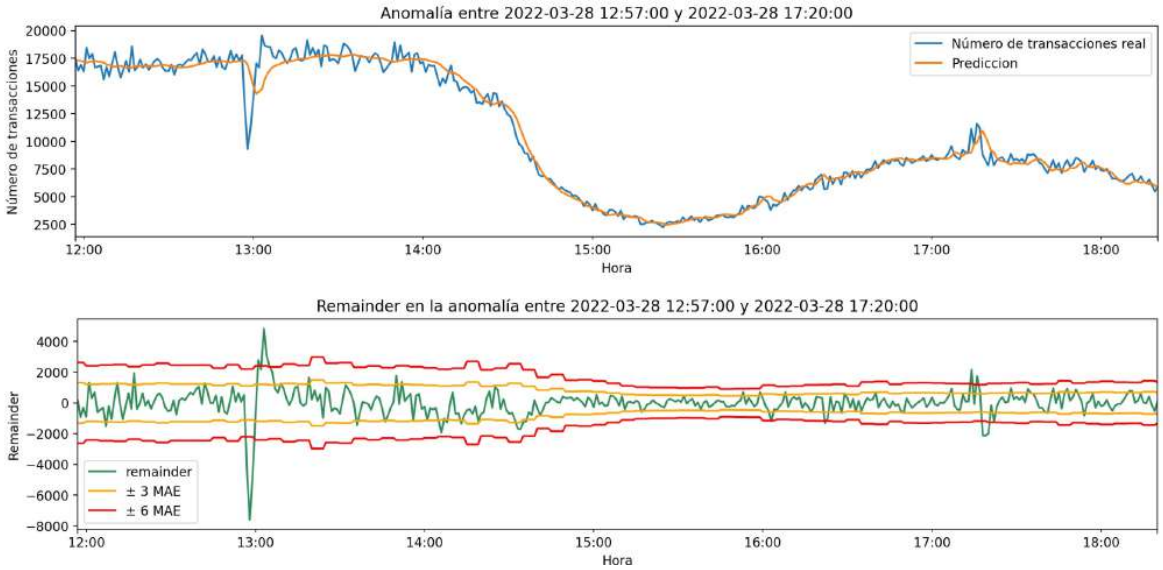


Figura 6.3: Anomalía detectada el Lunes 28 de Marzo del año 2022 de 12:57 a 17:20 horas

Para finalizar con la resolución del problema de detección de anomalías en operatoria bancaria empleando el modelo de redes neuronales LSTM, se comprueba si las incidencias masivas reportadas por el departamento de Monitorización y Disponibilidad de la entidad bancaria se detectan realmente como anomalías empleando este modelo. Las incidencias reportadas en los tres primeros meses del año 2022 se muestran en la siguiente tabla:

Fecha	Día de la semana	Hora de inicio	Hora de fin	Duración total	Tipo de incidencia
17 de Enero	Lunes	8:00	10:30	2:30	Fallos de aplicación
28 de Febrero	Lunes	11:05	11:48	0:43	Bloqueo aplicación

Cuadro 6.2: Incidencias masivas reportadas por el departamento de Monitorización y Disponibilidad en los 3 primeros meses del año 2022.

Veamos que ocurre, en primer lugar, con la incidencia reportada el día 17 de Enero. Si observamos la representación gráfica del remainder, es decir, de la diferencia entre las predicciones del modelo LSTM y los valores reales del número de operaciones realizadas, se aprecia que en ningún caso el valor del remainder supera el umbral de ±6 MAE. En el caso de considerar el umbral de ±3 MAE, se puede apreciar que hay algunas observaciones que superan un poco el umbral. Las anomalías se consideran aquellas observaciones que superan, habitualmente, los límites de ±6 MAE, siendo MAE el Error Medio Absoluto de entrenamiento a lo largo del día en períodos de 5 minutos. En este caso,

dado que los valores del remainder nunca superan el umbral de ±6 MAE, la indicencia reportada no se detecta como anomalía.

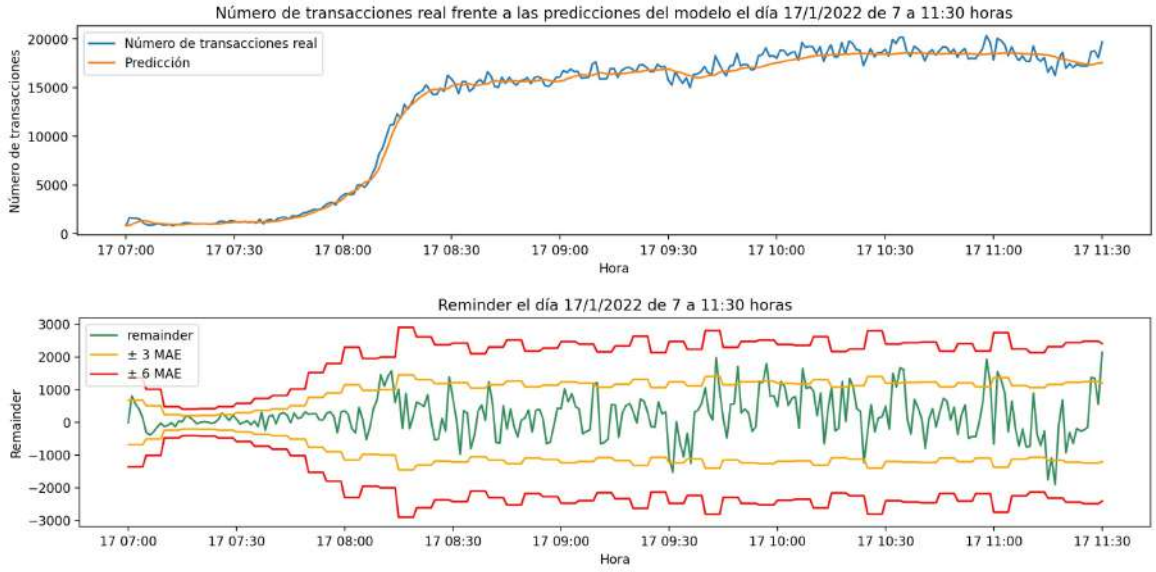


Figura 6.4: Incidencia reportada el día 17 de Enero de 8:00 a 10:30 horas.

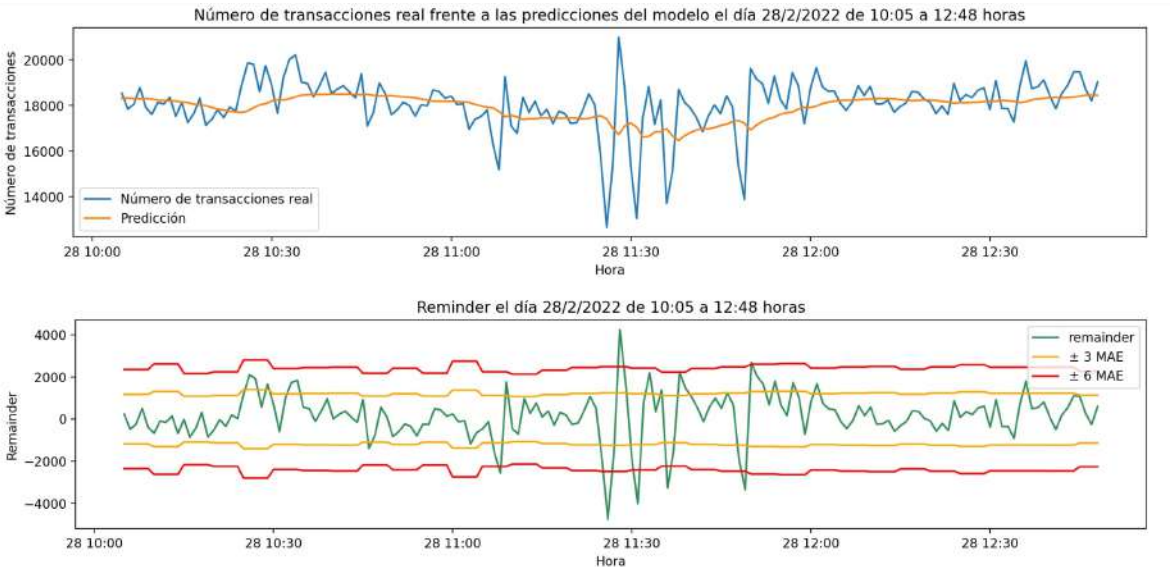


Figura 6.5: Incidencia reportada el día 28 de Febrero de 11:05 a 11:48 horas.

En cuanto a la incidencia reportada el día 28 de Febrero, en la representación gráfica del número de operaciones real frente a las predicciones obtenidas con el modelo LSTM se observa una gran diferencia entre ambos valores, en especial durante las horas de duración de la incidencia, con picos en los valores reales destacados entre las 11:20 y las 11:45 horas. Si nos fijamos en la representación gráfica del remainder, entre las 11:20 y las 11:48 horas, la diferencia entre las predicciones y los valores reales supera el umbral de ±6 MAE, lo que nos indica que la incidencia reportada sí se podría clasificar como

anomalía. Lo que ocurre es que, como hemos considerado que una anomalía es aquella que supere el umbral ±7 MAE, esta incidencia no se detecta como anomalía.

6.2. Modelo StackedEnsemble

Los modelos StackedEnsemble o modelos ensamblados son una combinación de diferentes modelos de Aprendizaje estadístico. La finalidad de estos modelos reside en mejorar la calidad de las predicciones finales. Para ello, se combinan las predicciones de los diferentes modelos considerados. Esta metodología está implementada en la interfaz H2oAutoMl de Python. Los modelos que se consideran en la construcción de estos métodos ensamblados son los que se incluyen en esa interfaz, considerando diferentes combinaciones de hiperparámetros para cada tipo de modelo. En este trabajo de fin de máster se considerarán los modelos lineales generalizados (GLM), los modelos Gradient Boosting Machine (GBM), los RandomForest y los modelos XGBoost[1]. Además de buscar modelos ensamblados, la interfaz H2oAutoMl de Python ajusta los modelos que se acaban de mencionar con diferentes hiperparámetros. El mejor modelo de la interfaz H2oAutoMl se selecciona empleando la metodología de validación cruzada. En particular, para el conjunto de datos que se emplea en este trabajo de fin de máster, se obtiene un modelo StackedEnsemble. Estos modelos ensamblados emplean los modelos lineales generalizados (GLM) con ponderaciones no negativas como el algoritmo de Aprendizaje combinador predeterminado. En la siguiente gráfica se muestran la importancia de los algoritmos de Aprendizaje Estadístico sobre los que se construye el modelo StackedEnsemble:

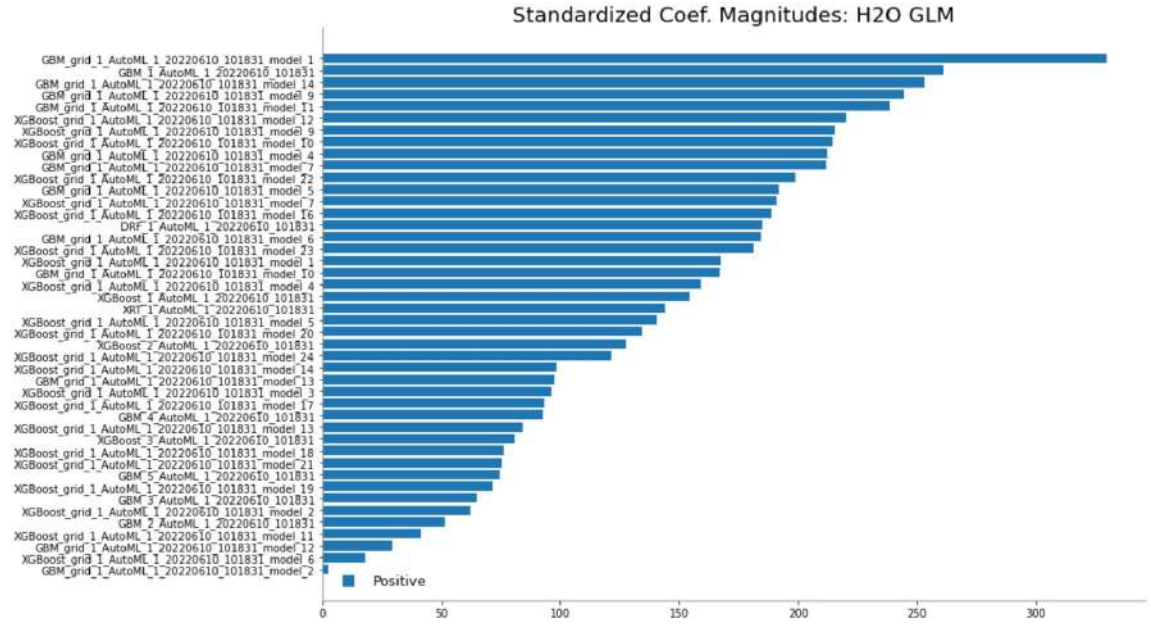


Figura 6.6: Modelos que intervienen en el modelo ensamblado obtenido con la interfaz H2oAutoMl de Python.

Como podemos apreciar, el modelo StackedEnsemble está formado por diferentes modelos XGBoost, Gradient Boosting Machine (GBM) y un modelo Random Forest con los parámetros por defecto. Vemos que el modelo de mayor importancia es un GBM obtenido con una malla de hiperparámetros (grid). No entraremos en detalle acerca de los parámetros de los diferentes modelos ajustados que se combinan para obtener las predicciones, pues no resulta de especial importancia para la obtención

[1]En el Apéndice E se describen de forma detallada estos métodos

de las anomalías. Cabe destacar que el Error Medio Absoluto en la etapa de entrenamiento para la selección del mejor modelo toma un valor de 165.21.

Al igual que en el modelo de redes neuronales LSTM, en la etapa de entrenamiento de los diferentes modelos, tras la obtención del modelo óptimo es de especial interés el cálculo del Error Medio Absoluto y de la Raíz Cuadrada del Error Cuadrático Medio a lo largo del día en períodos de 5 minutos. Los valores del MAE de entrenamiento a lo largo del día se emplearán para la detección de las anomalías. La siguiente gráfica muestra la evolución del MAE y del RMSE de entrenamiento a lo largo del día en períodos de 5 minutos del mejor modelo StackedEnsemble obtenido con la interfaz H2oAutoMl de Python.

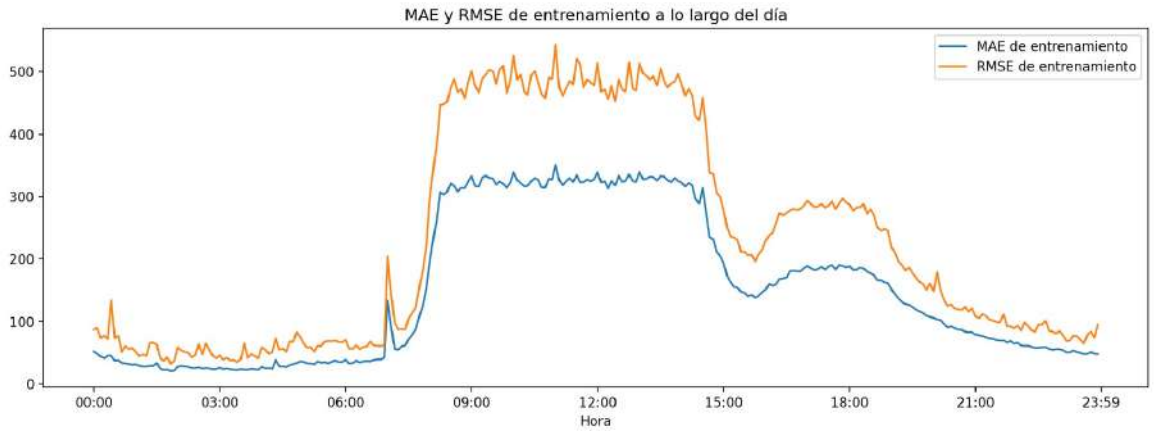


Figura 6.7: Evolución del Error Medio Absoluto (MAE) y de la Raíz Cuadrada del Error Cuadrático Medio (RMSE) de entrenamiento a lo largo del día en períodos de 5 minutos del mejor modelo ensamblado obtenido con la interfaz H2oAutoMl de Python.

En la representación gráfica se aprecia un comportamiento similar a lo que ocurría en el modelo de redes neuronales LSTM:

- El MAE toma valores cercanos a 0 en las horas de madrugada.
- Con la apertura de las oficinas bancarias a las 8:15 horas el MAE empieza a subir y se estabiliza a lo largo de la mañana tomando valores cercanos a 400.
- Con el cierre de las oficinas a las 14:15 horas, el MAE sufre una pequeña caída y se estabiliza en el horario de apertura de las oficinas por las tardes, tomando valores cercanos a 200.
- Tras el cierre de las oficinas bancarias en horario de tarde, el MAE desciende hasta tomar valores similares a las horas de la madrugada.

Con respecto al RMSE se aprecia que, en general, toma valores superiores al MAE y presenta picos más notables. Al igual que en el modelo de redes neuronales LSTM, esto es debido a que esta métrica de evaluación se ve muy influenciada por las anomalías del conjunto de datos de validación que se emplea en la etapa de entrenamiento.

Tras la elección del mejor modelo y visto el comportamiento del MAE a lo largo del día en períodos de 5 minutos, necesario para la obtención de las anomalías, se pasa a la etapa de evaluación del modelo. En esta etapa se emplean los datos de test para calcular las predicciones, teniendo en cuenta los valores de los 30 minutos anteriores. El MAE obtenido en la etapa de evaluación del modelo ensamblado es

177.95 y los valores del MAE y del RMSE de evaluación a lo largo del día en períodos de 5 minutos se representan en la siguiente gráfica.

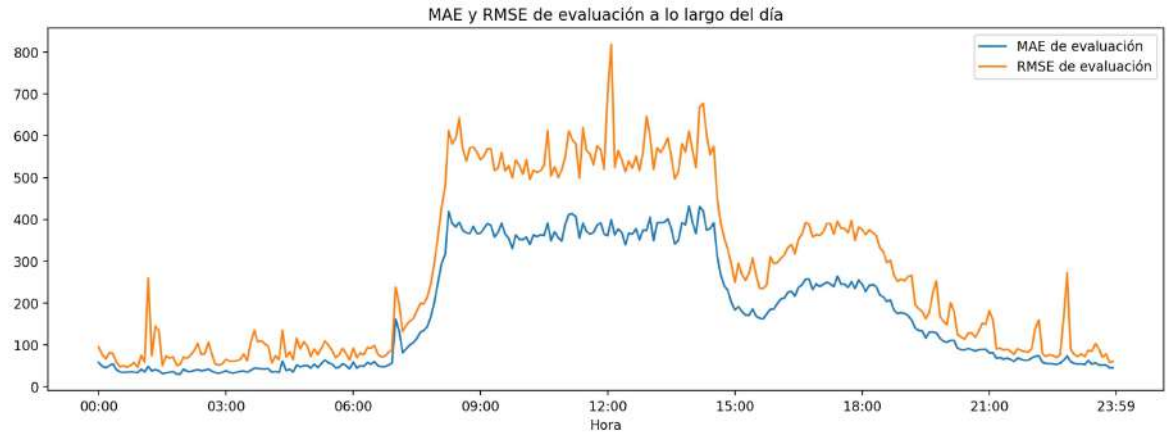


Figura 6.8: Evolución del Error Medio Absoluto (MAE) y de la Raíz Cuadrada del Error Cuadrático Medio (RMSE) de evaluación a lo largo del día en períodos de 5 minutos del mejor modelo ensamblado obtenido con la interfaz H2oAutoml de Python.

Se puede apreciar un comportamiento análogo a lo que ocurría con el MAE y el RMSE de entrenamiento a lo largo del día en períodos de 5 minutos. Se producen picos notables a la hora de apertura y cierre de las oficinas en horario de mañana, aunque de forma menos notable que en el caso del modelo LSTM, ya que en el modelo de ensamblaje los valores del MAE son inferiores a los del modelo LSTM. Nótese que en el RMSE se producen picos más notables que en el MAE, sobre todo en las horas centrales del día. Este comportamiento es debido a que esta métrica de evaluación se ve muy influenciada por las anomalías del conjunto de datos de test.

Una vez que se ha evaluado el modelo, obtenidas las predicciones y visto el comportamiento del MAE y del RMSE de entrenamiento y de evaluación a lo largo del día en períodos de 5 minutos, se procede con el cálculo de las anomalías. Recordemos que para determinar las anomalías se calcula la diferencia de las predicciones con los valores reales del número de operaciones, lo que se conoce como remainder, y se determinan las observaciones que superan el umbral de ±sensibilidad·MAE, siendo MAE el Error Medio Absoluto de entrenamiento a lo largo del día en períodos de 5 minutos. Estas observaciones serán clasificadas como anomalías. En particular, si se establece un valor de sensibilidad de 7, el mejor modelo StackedEnsemble obtenido empleando la interfaz H2oAutoMl de Python detecta un total de 15 anomalías en los tres primeros meses del año 2022, que se incluyen en la siguiente tabla:

Fecha	Día de la semana	Hora de inicio	Hora de fin	Duración total anomalía
20 de Enero	Jueves	12:30	16:47	4:17
28 de Enero	Viernes	14:12	19:12	5:00
1 de Febrero	Martes	12:22	17:18	5:04
4 de Febrero	Viernes	12:03	14:32	2:29

Fecha	Día de la semana	Hora de inicio	Hora de fin	Duración total anomalía
11 de Febrero	Viernes	11:13	14:28	3:15
14 de Febrero	Lunes	9:46	19:47	10:01
15 de Febrero	Martes	9:56	17:28	7:32
16 de Febrero	Miércoles	16:13	17:45	1:32
23 de Febrero	Miércoles	13:46	14:12	0:22
24 de Febrero	Jueves	8:37	17:41	8:53
2 de Marzo	Miércoles	11:15	17:15	6:00
10 de Marzo	Jueves	9:55	17:27	4:32
21 de Marzo	Lunes	17:36	18:01	0:25
22 de Marzo	Martes	17:12	18:01	0:53
28 de Marzo	Lunes	12:57	17:48	4:51

Cuadro 6.3: Anomalías detectadas empleando el modelo StackedEnsemble.

Se puede apreciar que las anomalías detectadas se producen en días laborables: la mayoría de ellas se inician en horas cercanas a la apertura de las oficinas en horario de tarde. Recordemos que los horarios de apertura de las oficinas por las mañanas es de 8:15 a 14:15 horas y, por las tardes, de Lunes a Jueves, de 16:30 a 18:45 horas.

En comparación con las anomalías detectadas con el modelo de redes neuronales LSTM, se aprecia que el modelo ensamblado detecta mayor número de anomalías y que la mayoría coinciden con las detectadas con el modelo LSTM, en particular, los días 20/1, 28/1, 4/2, 14/2, 24/2, 10/3, 22/3 y 28/2. Las restantes anomalías no se detectan con el modelo de redes neuronales LSTM.

Al igual que ocurría en el modelo de redes neuronales LSTM, dado que tenemos un número elevado de anomalías, en lugar de realizar la representación gráfica de todas las anomalías, se representa la más relevante como, por ejemplo, la detectada el día 28 de Enero y se comenta el porque se detecta esa observación como anomalía. Todas las anomalías detectadas por el modelo ensamblado pueden consultarse en el Apéndice D.

En la representación gráfica de la anomalía detectada el día 28 de Enero de 13:04 a 19:12 horas se aprecia que las predicciones obtenidas con el mejor modelo ensamblado son más suaves que los valores del número de operaciones real, en especial durante la hora de inicio de la anomalía, donde la diferencia entre la predicción y el valor real es bastante notable. Además, se puede observar que, tras el cierre de las oficinas bancarias a las 14:15 horas, las predicciones del modelo se asemejan bastante a los valores reales pero, a medida que nos acercamos a la hora de fin de anomalía, la predicción se desvía un poco más del valor real. Estos efectos de las predicciones y los valores reales que se acaban de comentar se aprecian muy bien en el gráfico que representa el remainder, que representa la diferencia entre las

predicciones y los valores reales. El remainder supera el umbral de ±6 MAE en las horas de inicio y fin de la anomalía mientras que, con la apertura de las oficinas bancarias en horario de tarde hasta aproximadamente las 19:00 horas, el remainder toma valores cercanos a 0. Como en las horas de inicio y fin de la anomalía los valores del remainder superan el umbral de ±7 MAE, se deduce que el día 28 de Enero en horario de 13:04 a 19:12 horas se produce una anomalía. Para las restantes anomalías, las conclusiones son análogas.

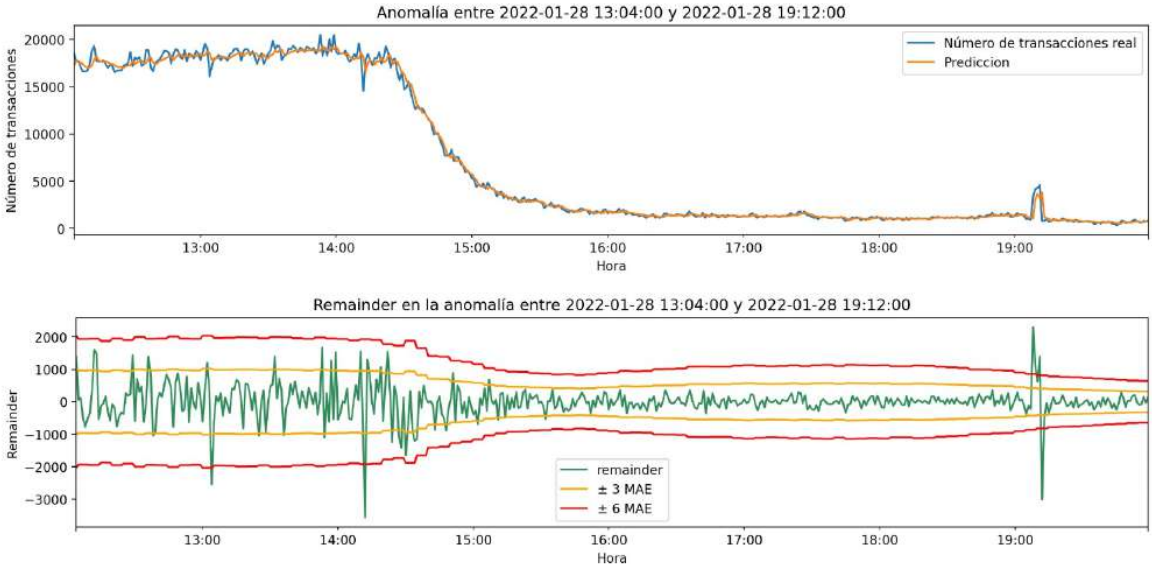


Figura 6.9: Anomalía detectada el Viernes día 28 de Enero de 14:12 a 19:12 horas con el mejor modelo ensamblado obtenido con la interfaz H2oAutoMl de Python.

Para finalizar con la resolución del problema de detección de anomalías en operatoria bancaria empleando el mejor modelo ensamblado obtenido con la interfaz H2oAutoMl de Python, al igual que se ha realizado con el modelo de redes neuronales LSTM, puede resultar de especial importancia determinar si las incidencias masivas reportadas por el departamento de Monitorización y Disponibilidad se detectan como anomalías empleando el modelo StackedEnsemble. Recordemos que las incidencias reportadas por este departamento en los tres primeros meses del año 2022 se producen los días 17 de Enero de 8:00 a 10:30 horas y 28 de Febrero de 11:05 a 11:48 horas. Las representaciones gráficas de las incidencias reportadas, en las que se incluye una comparativa de las predicciones y los valores del número de operaciones real junto al remainder con los umbrales de ±3 y ±6 MAE pueden encontrarse en la siguiente página.

Veamos que ocurre, en primer lugar, con la incidencia reportada el Lunes 17 de Enero de 8:00 a 10:30 horas. Al igual que ocurría con el modelo de redes neuronales LSTM, el remainder del modelo se encuentra dentro del umbral de ±6 MAE, siendo MAE el Error Medio Absoluto de entrenamiento a lo largo del día en períodos de 5 minutos. Además, hay ciertas observaciones en las que el remainder supera un poco el umbral de ±3 MAE. Recordemos que las anomalías son aquellas observaciones que superan el umbral ±6 MAE. Entonces, como los valores del remainder del modelo ensamblado no superan ese umbral, se deduce que la incidencia reportada no se clasifica como anomalía.

Con respecto a la incidencia reportada el Lunes 28 de Febrero de 11:05 a 11:48 horas, al igual que ocurría con el modelo LSTM, se aprecia que hay ciertas observaciones en las que el remainder supera el umbral de ±6 MAE, en especial entre las 11:20 y las 11:48 horas, la hora de finalización de la incidencia. Al superar el umbral, la incidencia se clasificaría como anomalía pero el modelo ensamblado

no la detecta pues, para la detección de las anomalías, se considera un umbral de ±7 MAE. Nótese que, a diferencia de lo que ocurría con el modelo de redes neuronales LSTM cuyas predicciones son más suavizadas (en el modelo ensamblado las predicciones no son tan suaves en el sentido de que intentan asemejarse más a los valores reales, con picos menos pronunciados).

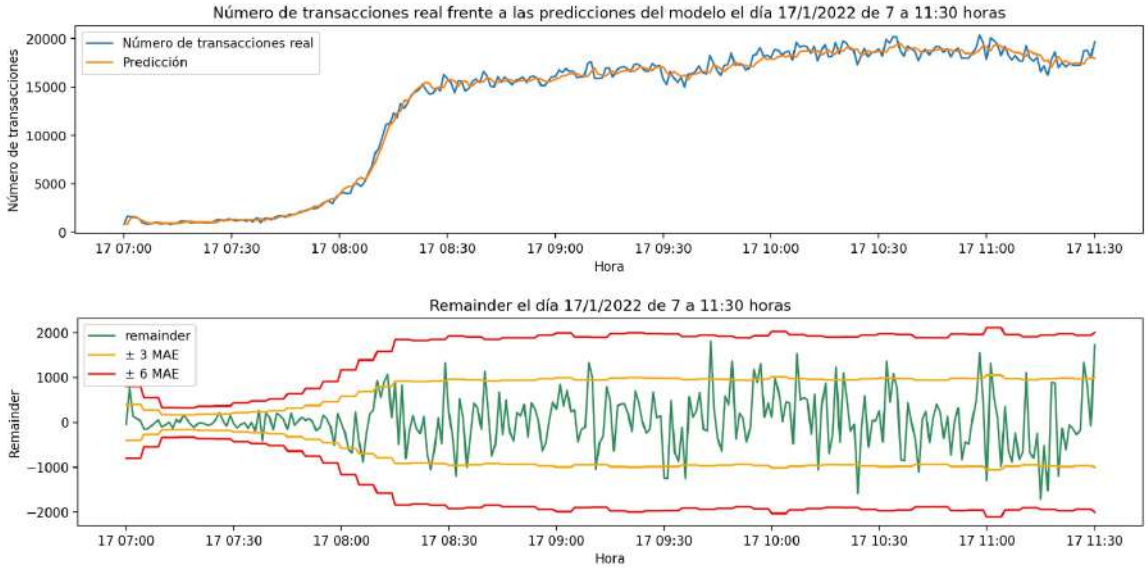


Figura 6.10: Incidencia reportada el día 17 de Enero de 8:00 a 10:30 horas.

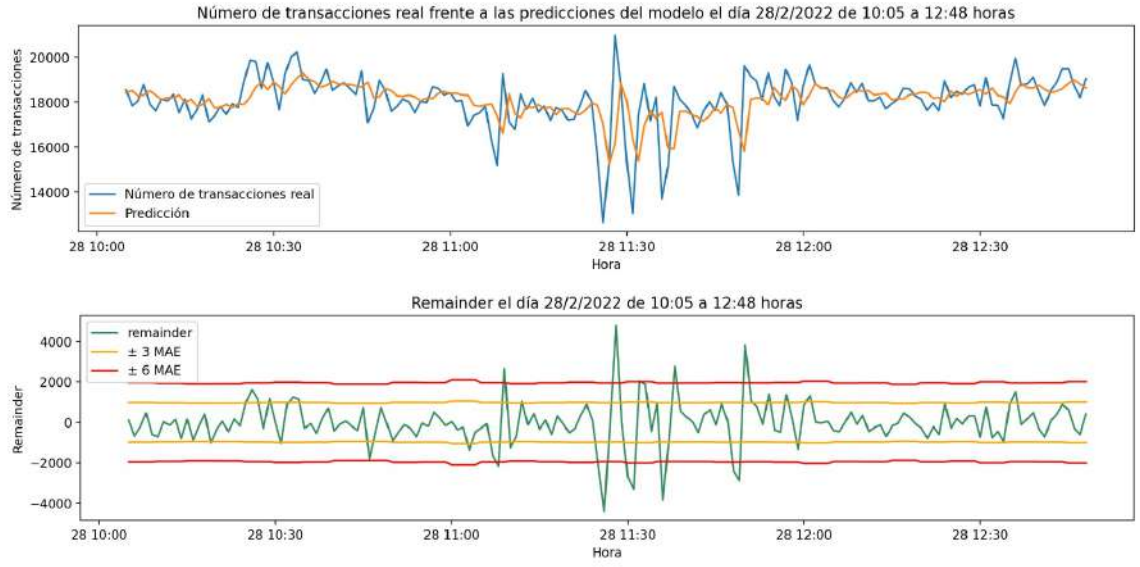


Figura 6.11: Incidencia reportada el día 28 de Febrero de 11:05 a 11:48 horas.

6.3. Comparativa de los modelos

Para finalizar este capítulo, tras realizar un análisis detallado de las anomalías detectadas, empleando los modelos de redes neuronales LSTM y el mejor modelo StackedEnsemble obtenido con la interfaz H2oAutoMl de Python, y un análisis de las incidencias reportadas por el departamento de Monitorización y Disponibilidad, se realiza una comparativa de los modelos empleados, en la que se incluye tiempo de computación (aproximado) en la etapa de entrenamiento y los gráficos comparativos del Error Cuadrático Medio, de los modelos considerados, en las etapas de entrenamiento y de evaluación a lo largo del día en períodos de 5 minutos.

Con respecto al tiempo de ajuste de los modelos en la etapa de entrenamiento, destacamos los siguientes aspectos:

- El tiempo de ajuste de los diferentes modelos de redes neuronales LSTM depende de las diferentes combinaciones de hiperparámetros. En particular, en este trabajo de fin de máster, en la etapa de entrenamiento, al considerar 3 valores de neuronas, 4 de *batch size* y 3 de *dropout*, se ajustan un total de 45 modelos. El tiempo total de ejecución en la fase de entrenamiento para la obtención de la mejor combinación de hiperparámetros ha sido de, aproximadamente 36 horas.
- En la interfaz H2oAutoMl de Python, el tiempo de ejecución de los modelos depende del usuario y del número de modelos a ajustar. Por ejemplo, para la obtención del mejor modelo ensamblado, se han considerado un total de 50 modelos a ajustar en la etapa de entrenamiento y se ha establecido como tiempo máximo de ejecución 6 horas, obteniendo resultados, aproximadamente, a las 4 horas de poner en producción.

Vemos, por lo tanto, que H2oAutoMl de Python realiza el ajuste de un número elevado de modelos en un tiempo más reducido que los modelos de redes neuronales LSTM. En el caso del modelo LSTM, podría reducirse el tiempo de computación si se considera un menor número de modelos en la etapa de entrenamiento. Aún así, si se reduce el número de modelos a ajustar, el tiempo de ejecución de la etapa de entrenamiento seguiría siendo elevado ya que el ajuste de este tipo de modelos también depende del conjunto de datos con el que se trabaja. En particular, en este trabajo, al considerar un total de 947349 observaciones en la muestra de entrenamiento, el tiempo total de ajuste de un único modelo LSTM y el cálculo de predicciones sobre la muestra de validación (105261 observaciones) puede ser de, aproximadamente, 1 hora.

Una vez que se ha determinado cuales son los tiempos de ejecución, aproximados, en la etapa de entrenamiento de los dos modelos considerados para resolver el problema de detección de anomalías en operatoria bancaria, se realiza una comparación del Error Medio Absoluto en las etapas de entrenamiento y de evaluación de los modelos de redes neuronales LSTM y el mejor modelo ensamblado.

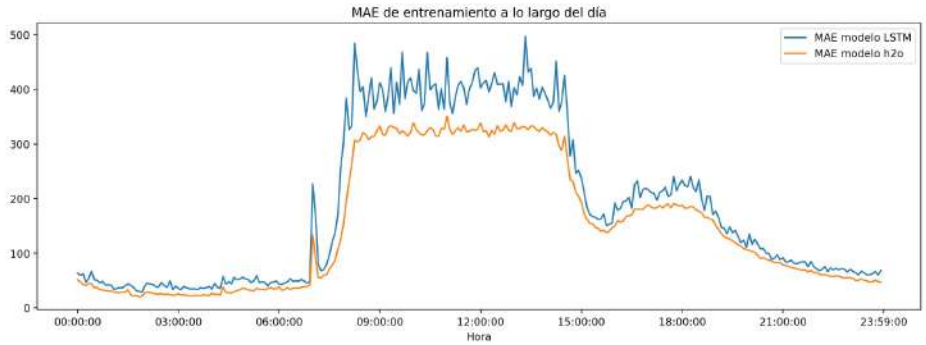


Figura 6.12: Evolución del Error Medio Absoluto de entrenamiento a lo largo del día en períodos de 5 minutos de los modelos LSTM y StackedEnsemble (etiqueta h2o).

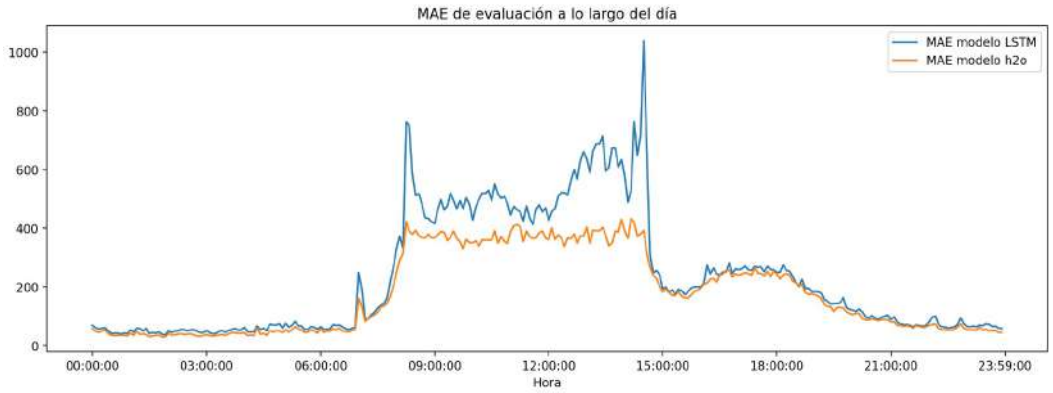


Figura 6.13: Evolución del Error Medio Absoluto de evaluación a lo largo del día en períodos de 5 minutos de los modelos LSTM y StackedEnsemble (etiqueta h2o).

En las representaciones gráficas anteriores se aprecia que, en general, los valores del MAE en ambos modelos son similares. En particular, el MAE del modelo ensamblado es ligeramente inferior al MAE del modelo LSTM. Cabe destacar que en las horas centrales del día, es decir, entre la apertura y el cierre de las oficinas en horario de mañana, es cuando se produce mayor diferencia entre ambos modelos, en especial, en la etapa de evaluación, donde se observa que el MAE del modelo LSTM tiene mayor variabilidad que el MAE del modelo ensamblado. Además, se aprecian picos notables en el MAE de evaluación del modelo LSTM con valores superiores a 800, debidos a las horas de apertura y cierre de las oficinas, respectivamente. Estos aspectos que se acaban de comentar no es algo que nos sorprenda pues el Error Medio Absoluto global en la etapa de evaluación del modelo de redes neuronales LSTM toma un valor superior al del modelo ensamblado: un MAE de 233.61 frente a un MAE de 177.95.

Capítulo 7

Resolución del problema de clasificación

En este capítulo se incluyen los resultados relativos a un problema de clasificación supervisada para la detección de anomalías. Primeramente, se realiza una clasificación de las anomalías empleando un modelo de regresión no paramétrica como la regresión polinómica local robusta (LOWESS) y posteriormente se obtiene el mejor modelo de clasificación empleando la interfaz H2oAutoML de Python. Además, se determinará cómo evoluciona la probabilidad de anomalía en un día en el que se conoce la existencia de una anomalía.

7.1. Clasificación de las anomalías

En el capítulo anterior hemos visto cuáles son las anomalías detectadas por los modelo de redes neuronales LSTM y un modelo de ensamblado y se ha determinado si las anomalías reportadas por el departamento de Monitorización y Disponibilidad se clasifican realmente como anomalías. Para ambos modelos, las dos incidencias reportadas no se detectan como anomalías, lo que nos da sospecha de que las incidencias reportadas por este departamento no son fiables, por lo que no nos sirve para realizar una clasificación de las anomalías.

Dado el especial interés por la realización de un modelo de clasificación para la detección de anomalías, como las incidencias reportadas por el departamento de Monitorización y Disponibilidad no son fiables, para obtener un conjunto de datos clasificado en anomalías se emplean modelos de regresión no paramétrica como, por ejemplo, la regresión LOWESS, comúnmente conocida por regresión polinómica local robusta. El procedimiento a seguir es el siguiente:

- Se selecciona la serie temporal del número de operaciones realizadas a nivel de minuto desde el 1/1/2020 hasta el día 31/3/2022 y se divide en ventanas temporales de 1 semana.
- Para cada una de las semanas consideradas se realiza un ajuste de regresión polinómica local robusta empleando un parámetro de ventana (o de suavización) $h = 0.01$, seleccionado a mano tras la realización de diferentes pruebas, y realizando el remuestreo de los pesos una vez.
- Tras el ajuste del polinomio se calculan los intervalos de confianza al 99 %, obtenidos mediante remuestreo boostrap, los que servirán de umbral para determinar qué observaciones se consideran anómalas: si una observación se encuentra fuera de los intervalos de confianza, se considera que es una anomalía.

En el lenguaje de programación de Python, para realizar el ajuste del modelo se emplea la función smoother.LowessSmoother de la librería tsmoothie.

Llegados a este punto el lector puede preguntarse el porque no se emplean ventanas temporales de un día en lugar de emplear ventanas temporales de una semana. Lo que ocurriría, si se empleasen ventanas temporales de un día, es que en los fines de semana y en los días festivos se tendrían muchas observaciones clasificadas como anomalías, lo que podría influenciar en el cálculo de las predicciones del modelo de clasificación supervisada. Sin embargo, si se emplean ventanas temporales de una semana, en los Sábados, Domingos y Festivos ya no habría tantas observaciones clasificadas como anomalías. A continuación, veremos gráficamente lo que se acaba de comentar.

Los siguientes gráficos muestran el ajuste polinómico local robusto realizado en un fin de semana empleando ventanas de un día junto a los intervalos de confianza al 99 %. Como podemos apreciar, hay bastantes observaciones que se salen de los intervalos de predicción, sobre todo en el Domingo. Estas observaciones se clasificarían como anomalías. Debido a que los fines de semana y festivos el comportamiento del número de operaciones es diferente a los días por semana, que exista un número elevado de anomalías podría influenciar en el cálculo de las predicciones en un modelo de clasificación supervisada.

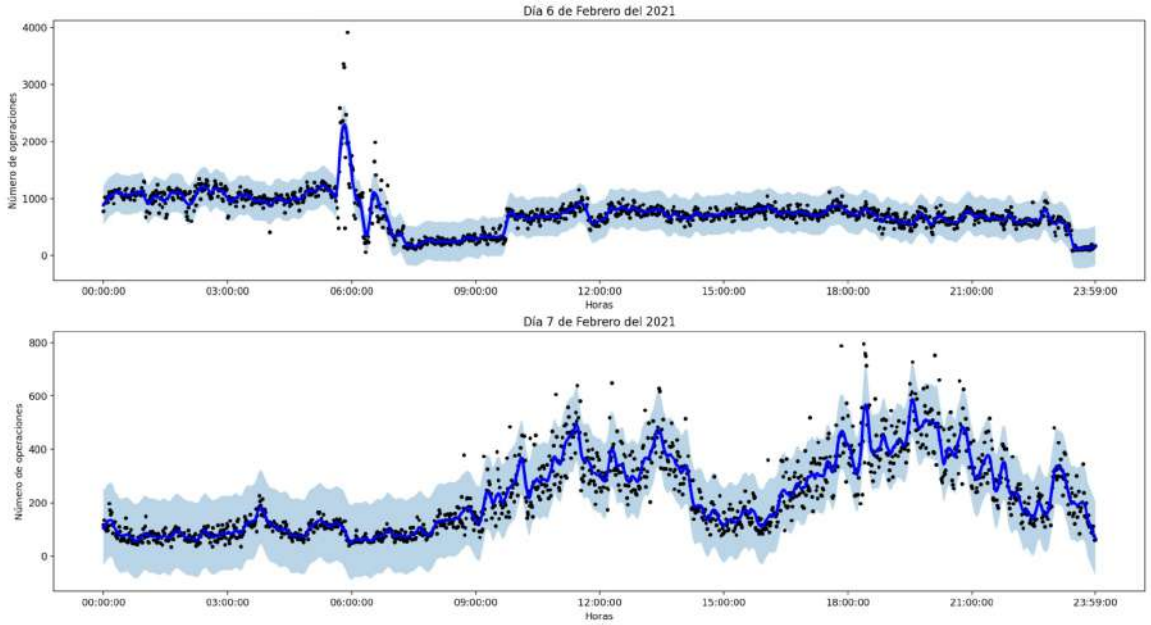


Figura 7.1: Representación gráfica del número de operaciones realizadas los días 6 y 7 de Febrero del año 2022, junto al ajuste polinómico local robusto (en azul oscuro) y los intervalos de predicción (en azul claro)

Si realizamos el ajuste polinómico local robusto en ventanas de una semana y representamos lo que ocurre, por ejemplo, en la primera semana de Enero del año 2020 y en la primera semana de Febrero de 2021, se aprecia la diferencia que existe en los fines de semana y festivos con respecto a ajustar el modelo LOWESS empleando ventanas de un día. Se puede observar que apenas hay observaciones que se salen de los intervalos de predicción en estos días, por lo que habría un menor número de observaciones clasificadas como anomalías en festivos y fines de semana. Nótese que en los dos gráficos se muestran dos casos diferentes, una semana en la que se consideran dos festivos (1 de Enero y 6 de Enero) y una semana con un comportamiento habitual. Cabe destacar que, el día 1/2/2021 se produce una anomalía destacada en las horas del mediodía y se observa cómo la mayoría de esas observaciones no se sitúan en el intervalo de predicción. Vemos, por lo tanto, que ajustando el modelo empleando

ventanas de una semana, clasificamos de forma correcta las anomalías.

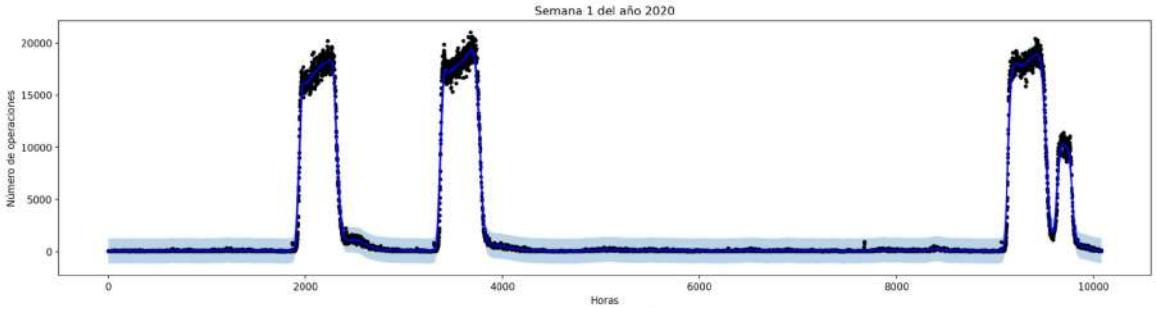


Figura 7.2: Representación gráfica del número de operaciones realizadas en la primera semana de Enero del año 2020, junto al ajuste polinómico local robusto (en azul oscuro) y los intervalos de predicción (en azul claro)

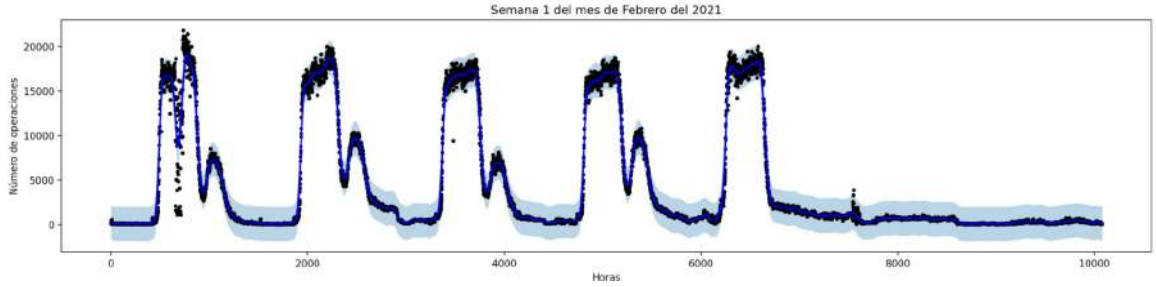


Figura 7.3: Representación gráfica del número de operaciones realizadas en la primera semana de Febrero del año 2021, junto al ajuste polinómico local robusto (en azul oscuro) y los intervalos de predicción (en azul claro)

Comentar que, tras realizar el ajuste de regresión polinómica local robusta, LOWESS, empleando ventanas de una semana, de un total de 1182240 observaciones, hay 42834 clasificadas como anomalías, es decir, aproximadamente el 3.623 % de los datos se clasifican como anomalías.

7.2. Modelo StackedEnsemble

Una vez realizada la clasificación de la serie temporal del número de operaciones realizadas en los años 2020, 2021 y los tres primeros meses del año 2022, se resuelve un problema de clasificación supervisada para la detección de las anomalías, realizando un procedimiento similar a la resolución del problema de clasificación no supervisada realizada en el capítulo anterior.

Para resolver el problema de clasificación supervisada, al conjunto de datos del número de operaciones realizadas clasificadas en anomalías se le añaden cuatro variables adicionales: la hora del día, el día del año, el día de la semana y una variable indicadora de si el día es laborable o no. Además, para modelizar la dependencia del tiempo debemos incluír los valores de los lags entre el número de operaciones real, que serán de utilidad para el cálculo de las predicciones que, en modelos de clasificación supervisada, se obtienen en función de una probabilidad.

Una vez añadidas las variables adicionales y la proporción de lags de 30 minutos al conjunto de datos con el que se va a trabajar, se realiza una búsqueda automática del mejor modelo para la re-

solución del problema de clasificación supervisada de las anomalías empleando la interfaz H2oAutoMl de Python. Recordemos que en esta interfaz están implementados una gran variedad de modelos de Aprendizaje Estadístico y que se incluye el ajuste de diferentes modelos ensamblados. En este trabajo de fin de máster se consideran todos los modelos de la interfaz, a excepción de los algoritmos de redes neuronales, y diferentes combinaciones de los mismos. Para la elección del mejor modelo, durante la etapa de entrenamiento, en lugar de emplear una muestra de entrenamiento en la que se realiza validación cruzada y una muestra de validación para obtener el MAE para la elección del mejor modelo, se ajustan diferentes modelos empleando una muestra de entrenamiento y se emplea la metodología de validación cruzada para la elección de la mejor combinación de hiperparámetros. Recordemos que en modelos de clasificación supervisada, para la elección del mejor modelo se tienen en cuenta los valores del Área bajo la curva (AUC) y, en caso de tener muestras desbalanceadas, el Área bajo la curva Precision Recall (AUCPR). En particular, para el conjunto de datos que estamos empleando, como tenemos muestras desbalanceadas, se emplea la métrica de evaluación AUCPR. El mejor modelo obtenido es un StackedEnsemble, un modelo ensamblado formado, principalmente, por combinaciones de diferentes modelos GBM y XGBoost. Al igual que en el problema de clasificación no supervisada, no entraremos en detalle en los hiperparámetros de estos modelos pues no resultan de especial importancia para la obtención de las predicciones.

Tras la obtención del modelo ensamblado, el mejor modelo obtenido con la interfaz H2oAutoMl de Python, se pasa a la etapa de evaluación del modelo en la que se calculan las predicciones y se comparan con los valores de la muestra de test. En problemas de clasificación supervisada, lo más habitual es realizar esta comparativa en una tabla de contingencia, conocida habitualmente con el nombre de matriz de confusión. Los resultados de la tabla de contingencia para el problema de clasificación de las anomalías se muestran en la siguiente tabla:

Real\Predicción	Anomalía	No anomalía
Anomalía	3766	869
No anomalía	1274	123661

Cuadro 7.1: Matriz de confusión para el problema de clasificación de las anomalías.

Si observamos la matriz de confusión, se aprecia que se comete un mayor número de errores al clasificar observaciones anómalas como no anómalas que clasificando observaciones no anómalas como anómalas: hay un total de 1274 observaciones mal clasificadas como no anómalas frente a un total de 869 observaciones clasificadas como anómalas. Esto que acabamos de comentar se refleja muy bien en los valores de Sensibilidad (tasa de verdaderos positivos) y Especifidad (tasa de verdaderos negativos):

$$VP = \frac{3766}{3766 + 869} = 0.81251$$
$$VN = \frac{123661}{123661 + 1274} = 0.98980$$

y en la tasa global de aciertos (Accuracy) que toma un valor de 0.98415. Este valor nos indica que el 98.415 % de las predicciones obtenidas con el modelo están bien clasificadas. Lo que ocurre es que, como la muestra de test que empleamos en la evaluación de este modelo de clasificación es desbalanceada, el accuracy puede dar lugar a equivocación, por lo que es necesario acudir a otro tipo de métricas de evaluación como el F1 Score y el índice predictivo positivo (Precision):

$$\text{F1} = \frac{2 \cdot 3766}{2 \cdot 3766 + 1274 + 869} = 0.77850$$

$$PPV = \frac{3766}{3766 + 1274} = 0.74722$$

Se aprecia que tanto el F1 Score como el índice predictivo positivo no superan el valor de 0.8. Lo ideal sería que estos valores se acercasen lo más posible al valor de 1.

Además de las métricas que se acaban de obtener, en la evaluación de un método de clasificación supervisada, cuando las muestras son desbalanceadas, se suele calcular el Área bajo la Curva Precision Recall (AUCPR), que se calcula como el área bajo la curva que representa el índice predictivo positivo (PPV) frente a la tasa de verdaderos positivos (TVP). Esta métrica de evaluación determina lo "bueno" que es el clasificador del método empleado: si el AUCPR toma el valor 1, el clasificador es perfecto y si el AUCPR toma el valor 0.5, el clasificador se considera aleatorio. En particular, considerando el problema de clasificación de anomalías, se obtiene un AUCPR de 0.85532, que no es un valor muy cercano a 1 ni a 0.5. Esto nos indica que el clasificador que se está empleando ni es perfecto ni es aleatorio. De todos modos, a pesar de tener muestras desbalanceadas es un valor bastante elevado por lo que el clasificador podría ser adecuado para clasificar las anomalías.

Métrica de evaluación	Valor
Precision (índice predictivo positivo)	0.74722
Recall o Sensibilidad (tasa de verdaderos positivos)	0.81251
Especifidad (tasa de verdaderos negativos)	0.98980
Accuracy (tasa global de aciertos)	0.98415
F1 Score	0.77850
Área bajo la curva Precision Recall (AUCPR)	0.85532

Cuadro 7.2: Métricas de evaluación del problema de clasificación de las anomalías.

Cabe destacar que los valores de las métricas de evaluación, recogidas en la tabla anterior, podrían mejorarse si se emplean métodos de *undersampling* o *oversampling* para equilibrar las muestras modificando las distribuciones de los datos de la clase minoritaria [1] y de la clase mayoritaria [2], respectivamente. En este trabajo de fin de máster no entraremos en detalle con respecto a la resolución del problema de clasificación de anomalías con estos métodos.

Tras calcular la matriz de confusión y comentar las diferentes métricas de evaluación, para finalizar la resolución del problema de clasificación se determina cómo evoluciona la probabilidad de anomalía a lo largo de un día. Para ello, se escogen las observaciones correspondientes a un día en el que se conozca la existencia de una anomalía destacada, como por ejemplo el día 4 de Febrero, y se representa gráficamente el número de operaciones realizadas con las observaciones clasificadas como anomalías con el ajuste de regresión polinómica local junto con la probabilidad de la clase minoritaria (anomalía) a lo largo del día obtenida en la etapa de evaluación del modelo ensamblado.

[1] La clase minoritaria son las observaciones anómalas
[2] La clase mayoritaria se corresponde con las observaciones no anómalas

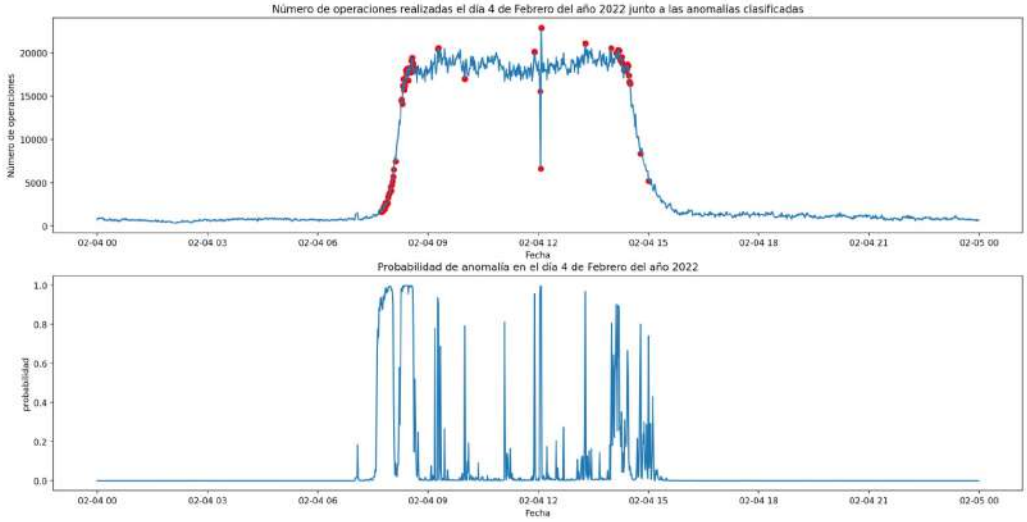


Figura 7.4: Evolución de la probabilidad de anomalía el Viernes día 4 de Febrero, día también detectado como anomalía con el modelo de redes neuronales LSTM.

Se puede apreciar que la probabilidad de anomalía toma valores cercanos a 0 durante las horas de la madrugada y en el horario de tarde. Durante las horas centrales del día la probabilidad de anomalía es variable y toma, en la mayoría de los casos valores superiores a 0.5. Cabe destacar que, entre las 12:00 y las 12:05 horas, donde se produce la hora de inicio de la anomalía detectada con el modelo de redes neuronales LSTM, la probabilidad de anomalía es bastante elevada, lo que indica que el modelo de clasificación clasifica correctamente dicha anomalía. Nótese que, al considerar un Viernes, dado que las oficinas bancarias no abren por las tardes, es bastante evidente que la probabilidad de anomalía sea casi nula en el horario de tarde. Si se considera otro día de la semana, como por ejemplo, un Lunes, puede ocurrir que la probabilidad de anomalía sea superior a 0.5 en horario de tarde. Este efecto se ve muy bien en el siguiente gráfico:

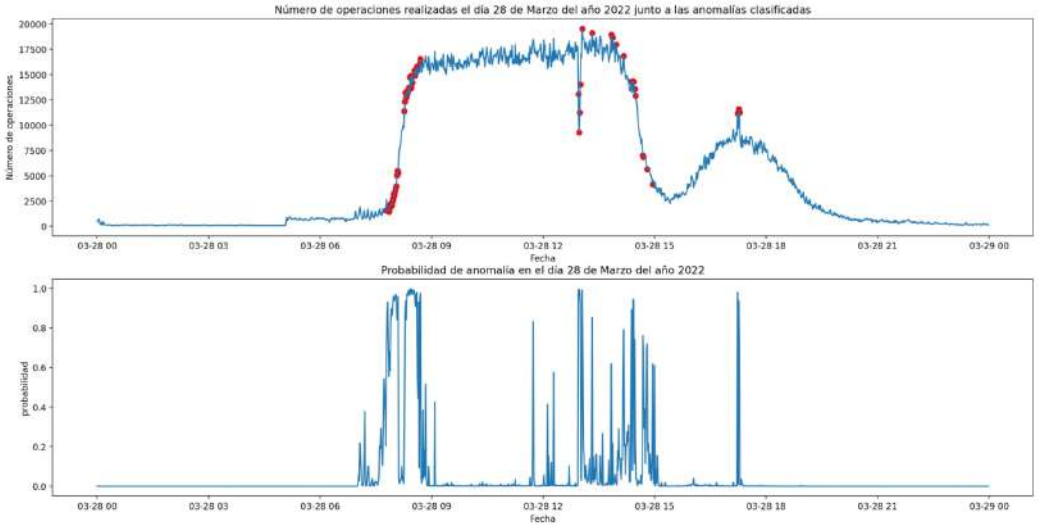


Figura 7.5: Evolución de la probabilidad de anomalía el Lunes día 28 de Marzo, día también detectado como anomalía con el modelo de redes neuronales LSTM y en el modelo StackedEnsemble.

En efecto, la probabilidad de anomalía es prácticamente nula y alcanza valores superiores a 0.5 entre las 17:10 y las 17:20 horas, coincidiendo con la hora de finalización de la anomalía detectada con los modelos LSTM y StackedEnsemble.

Capítulo 8

Conclusiones

El objetivo principal del trabajo de fin de máster era el de aplicar diferentes técnicas de Aprendizaje Estadístico para la detección de anomalías en operatoria bancaria con la finalidad de determinar cuándo podría existir un fallo en las herramientas que emplean los gestores en las oficinas para contabilizar las operaciones realizadas. Para ello, se ha empleado una serie temporal, a nivel de minutos, del número de operaciones realizadas en las oficinas bancarias de la entidad, en la Banca Electrónica y en la Banca Móvil, durante los años 2020, 2021 y los 3 primeros meses de 2022. Principalmente, las anomalías se han detectado en los 3 primeros meses de este año.

En este trabajo de fin de máster se ha realizado la resolución de tres problemas bien diferenciados: un problema empleando métodos autoexplicativos, un problema de detección de anomalías empleando métodos de Aprendizaje Estadístico y un problema de clasificación supervisada. Principalmente, se ha comprobado cuál es la metodología más adecuada de resolución del problema de detección de anomalías en operatoria bancaria.

En cuanto a la resolución del problema empleando métodos autoexplicativos, se ha comprobado que estos modelos no son adecuados para la detección de anomalías en operatoria bancaria, principalmente por la distribución del número de operaciones realizadas, que es diferente dependiendo del día de la semana[1]. Las predicciones obtenidas con los modelos UCM y Prophet se comportan como funciones trigonométricas y no se adecuan a la distribución de los datos. En el caso del modelo TBATS, cuyas predicciones se han obtenido con los datos en horas y en períodos de 1 día, el comportamiento de las predicciones es similar al número de operaciones real, pero este modelo tiene el inconveniente de que el tiempo de ejecución es elevado por lo que, si se quisiese ejecutar en infraestructura del banco, se necesitaría un tiempo elevado para la obtención de resultados. La idea es ajustar modelos de Aprendizaje Estadístico que permitan obtener resultados adecuados en un tiempo no muy elevado.

Con respecto a la resolución del problema empleando modelos de predicción a tiempo real, en lugar de emplear únicamente los datos de la serie temporal del número de operaciones, se han incluido variables adicionales como la hora del día, el día del año, el día de la semana y una variable indicadora de si el día es laborable o no. Además, para modelizar la dependencia del tiempo y calcular las predicciones, se emplean los datos del número de operaciones realizadas en los 30 minutos anteriores. Los modelos que se han empleado son las redes neuronales de memoria a corto plazo, LSTM, y los modelos GLM, GBM, XGBoost y modelos ensamblados, incluídos en la interfaz H2oAutoMl de Python. Independientemente del modelo, LSTM o el modelo StackedEnsemble (el mejor modelo de la interfaz H2oAutoMl),

[1]Recordemos que hay diferencias significativas en la distribución del número de operaciones realizadas los días por semana y los fines de semana. Además, los Viernes a la tarde la distribución del número de operaciones es diferente a la distribución de los restantes días por semana, ya que los Viernes es el único día de la semana en el que las oficinas no abren por las tardes.

se ha comprobado que se detectan correctamente anomalías con un valor de sensibilidad de 7 y que las incidencias reportadas por el departamento de Monitorización y Disponibilidad no son del todo fiables ya que no se detectan como anomalías empleando estos métodos. Se ha comprobado que los valores de las métricas de evaluación en la etapa de entrenamiento de ambos modelos son inferiores a 200, con valores respectivos de 180.02 y 165.05. En cuanto a los tiempos de ejecución de los modelos, se ha comprobado que los modelos de la interfaz H2oAutoMl de Python son más rápidos computacionalmente ya que el ajuste de los modelos depende del tiempo máximo de ejecución y del número máximo de modelos a ajustar, prefijados por el usuario. En cuanto a los modelos de redes neuronales LSTM, el tiempo de ajuste aproximado de un modelo puede ser de 1 hora. En el caso de ajustar varios modelos, el tiempo de ejecución puede ser elevado, por ejemplo, de 36 horas en el caso de considerar 45 combinaciones de hiperparámetros. Entonces, si se quisiese ejecutar este tipo de modelos para la detección de anomalías en operatoria bancaria tendríamos que tener en cuenta el número de modelos a ajustar: en los modelos de redes neuronales lo más adecuado sería escoger una combinación de hiperparámetros similar a la que obtiene el mejor MAE de entrenamiento (recordemos que se emplearon 20 neuronas, 1000 de *batch size* y 0.15 de *dropout* en el ajuste de la red neuronal LSTM) y, en los modelos de la interfaz H2oAutoMl de Python escoger un valor adecuado para el tiempo de ejecución y un número elevado de modelos (por ejemplo, ajustar 50 modelos en un tiempo máximo de 3 horas).

En cuanto a la resolución del problema de clasificación supervisada de las anomalías, dado que las incidencias reportadas no son del todo fiables como para clasificarlas como anomalías, se precisa obtener un conjunto de datos clasificado en dos categorías: anomalía o no anomalía. Para obtener dicho conjunto de datos se ha empleado un modelo de regresión no paramétrica, en particular la regresión LOWESS. El procedimiento que se ha seguido es el siguiente: se ha divido la serie temporal en ventanas temporales de 1 semana y, para cada una de las semanas, se ha realizado un ajuste LOWESS con un parámetro de ventana h=0.001 y se han calculado los intervalos de confianza al 99 % que se emplearon como umbral para determinar las observaciones anómalas. Una vez que se ha obtenido una clasificación de los datos en anomalías, se ha realizado el ajuste de un modelo de clasificación no supervisada, en particular, se han ajustado modelos Random Forest, GLM, GBM, XGBoost y modelos ensamlados empleando la interfaz H2oAutoMl de Python. Al igual que se ha realizado en el ajuste de estos modelos empleando clasificación no supervisada[2], se han considerado variables adicionales como la hora del día, el día del año, el día de la semana y una variable indicadora de si el día es laborable o no. Además, para modelizar la dependencia del tiempo, se ha incluido la proporción de lags de los 30 minutos, es decir, se han dividido los valores de los 30 minutos anteriores entre los valores reales del número de operaciones. Una vez realizado el ajuste de los anteriores modelos, se ha comprobado que el mejor modelo de clasificación de las anomalías era un modelo ensamblado formado principalmente por diferentes combinaciones de modelos XGBoost y GBM. Tras la obtención del mejor modelo, se han calculado las predicciones, la matriz de confusión y las diferentes métricas de evaluación. Se ha comprobado que el Área bajo la Curva Precision Recall, AUCPR, toma un valor de 0.85, un valor ni muy cercano a 1 ni muy cercano a 0.5, lo que indica que el modelo de clasificación no es ni perfecto ni aleatorio. El modelo de clasificación obtenido se ha considerado que es adecuado pero, cabe destacar que, como se ha trabajado con muestras desbalanceadas[3] los valores obtenidos de las métricas de evaluación podrían mejorarse si se empleasen métodos de balanceado de clases, entre las que se incluyen el *undersampling* y el *oversampling*. Ambas técnicas equilibran las muestras modificando la distribución de la clase minoritaria[4] y mayoritaria[5], respectivamente. Como trabajo futuro, podrían aplicarse estas técnicas de balanceo de clases, resolver el problema de clasificación supervisada y comprobar si se mejoran las métricas de evaluación obtenidas con la resolución del problema de clasificación supervisada con las muestras desbalanceadas.

[2]Modelos predictivos a tiempo real

[3]En los datos que se han clasificado como anomalías, hay un mayor número de observaciones clasificadas no son anómalas.

[4]La clase minoritaria son las observaciones anómalas

[5]La clase mayoritaria se corresponde con las observaciones no anómalas

Para finalizar comentar que la detección de anomalías, en general, es un problema muy abierto en el sentido de que existe una gran variedad de métodos que se pueden aplicar aunque hay que tener en cuenta que la metodología a aplicar depende de la serie temporal con la que se trabaje y de las características que nos pueda aportar. En este trabajo de fin de máster, nos hemos centrado en trabajar con la serie temporal, a nivel de minutos, del número de operaciones realizadas en las oficinas, en Banca Móvil y en Banca Electrónica. Se ha comprobado que los modelos autoexplicativos no son adecuaados para la detección de anomalías y que los métodos más complejos de predicción a tiempo real como las redes neuronales LSTM y los modelos ensamblados detectan correctamente anomalías. Como trabajo futuro podrían emplearse otro tipo de técnicas novedosas, más complejas y muy recientes, que modelizan la dependencia temporal tales como las redes neuronales generativas antagónicas, GAN (*Generative Adversarial Networks*), o las redes neuronales convolucionales, CNN (*Convolutional Neural Networks*):

Técnicas	Referencias
Redes Neuronales Adversariales (GAN)	Geiger et al. (2020)
Unidades recurrentes cerradas (GRU)	Qu et al. (2018)
Redes Neuronales Recurrentes (CNN)	Wen et al. (2019)
Redes Neuronales Convolucionales temporales (TCN)	He et al. (2019)
Redes neuronales LSTM convolucionales (ConvLSTM)	Kim et al. (2018)
Memoria temporal Jerárquica (HTM)	Bamaqa et al. (2020)

Cuadro 8.1: Técnicas de detección de anomalías que modelizan la dependencia temporal de la serie de tiempo con la que se trabaja. Esta tabla se ha elaborado siguiendo como referencia Choi et al. (2021).

Apéndice A

Lista de los festivos nacionales, autonómicos y locales

En este apéndice se incluye una tabla con los días festivos nacionales, autonómicos y locales (ciudad de A Coruña, pues la sede central de la entidad se encuentra en esta ciudad) de los años 2020, 2021 y los 3 primeros meses del año 2022.

Día	Nombre	Tipo
1/1/2020	Año nuevo	Nacional
6/1/2022	Epifanía del Señor	Nacional
25/2/2020	Martes de Carnaval	Local
19/3/2020	San José	Autonómico
9/4/2020	Jueves Santo	Autonómico
10/4/2020	Viernes Santo	Nacional
1/5/2020	Día del Trabajo	Nacional
24/6/2020	San Juan	Autonómico
25/7/2020	Santiago Apóstol (Día de Galicia)	Autonómico
15/8/2020	Asunción de la Virgen	Nacional
7/10/2020	Virgen del Rosario	Local
12/10/2020	Día del Pilar	Nacional
8/12/2020	Inmaculada Concepción	Nacional

Día	Nombre	Tipo
25/12/2020	Navidad	Nacional
1/1/2021	Año nuevo	Nacional
6/1/2021	Epifanía del Señor	Nacional
16/2/2021	Martes de Carnaval	Local
19/3/2021	San José	Autonómico
1/4/2021	Jueves Santo	Autonómico
2/4/2021	Viernes Santo	Nacional
1/5/2021	Día del Trabajo	Nacional
17/5/2021	Día de las letras gallegas	Autonómico
24/6/2021	San Juan	Local
12/10/2021	Día del Pilar	Nacional
1/11/2021	Día de Todos los santos	Nacional
6/12/2021	Día de la constitución	Nacional
8/12/2021	Inmaculada Concepción	Nacional
25/12/2021	Navidad	Nacional
1/1/2022	Año nuevo	Nacional
6/1/2022	Epifanía del Señor	Nacional
1/3/2022	Martes de Carnaval	Local

Apéndice B

Resultados del MAE y RMSE de los modelos LSTM

En este apéndice se incluyen los resultados del Error Medio Absoluto (MAE) y de la Raíz Cuadrada del Error Cuadrático Medio (RMSE) del entrenamiento (train) y de la evaluación de los diferentes modelos de redes neuronales LSTM ajustados.

Neuronas	Batch size	Dropout	train MAE	train RMSE	MAE evaluación	RMSE evaluación
15	100	0.15	223.24	377.45	234.07	409.56
15	100	0.20	216.20	395.34	241.48	472.66
15	100	0.25	247.67	407.79	235.58	455.44
15	200	0.15	242.53	387.73	247.79	491.29
15	200	0.20	202.17	369.99	252.17	511.53
15	200	0.25	234.11	400.67	291.10	523.29
15	300	0.15	314.21	496.87	323.83	582.74
15	300	0.20	231.04	417.38	330.13	540.98
15	300	0.25	360.78	469.34	300.58	604.60
15	500	0.15	245.03	420.50	256.15	516.44
15	500	0.20	352.52	461.95	263.07	491.71
15	500	0.25	252.36	461.44	268.35	510.45
15	1000	0.15	254.16	461.16	298.07	570.73

Neuronas	Batch size	Dropout	train MAE	train RMSE	MAE evaluación	RMSE evaluación
15	1000	0.20	312.05	502.43	309.71	589.15
15	1000	0.25	234.30	410.44	273.53	532.70
20	100	0.15	229.48	403.37	241.82	463.28
20	100	0.20	251.04	440.87	225.35	408.74
20	100	0.25	232.68	425.51	313.43	621.26
20	200	0.15	227.31	368.79	215.58	398.21
20	200	0.20	224.07	374.23	242.86	466.81
20	200	0.25	256.85	396.61	254.49	507.47
20	300	0.15	218.11	408.27	251.46	490.87
20	300	0.20	257.80	436.07	263.15	522.42
20	300	0.25	233.45	419.83	276.48	530.72
20	500	0.15	227.16	415.86	229.06	438.98
20	500	0.20	270.29	455.35	255.54	512.91
20	500	0.25	299.50	489.93	266.36	537.53
20	1000	0.15	186.02	374.98	233.61	473.24
20	1000	0.20	235.85	442.09	309.59	552.96
20	1000	0.25	198.84	395.13	276.09	517.41
30	100	0.15	207.57	362.88	205.64	392.35
30	100	0.20	201.70	366.65	219.72	431.01
30	100	0.25	188.87	359.86	228.91	442.88
30	200	0.15	227.68	379.69	203.99	390.58
30	200	0.20	214.28	373.12	261.02	538.14
30	200	0.25	286.62	482.63	255.14	481.80
30	300	0.15	211.56	379.12	231.77	434.68
30	300	0.20	214.35	389.98	281.39	484.01
30	300	0.25	207.25	369.39	250.01	475.61

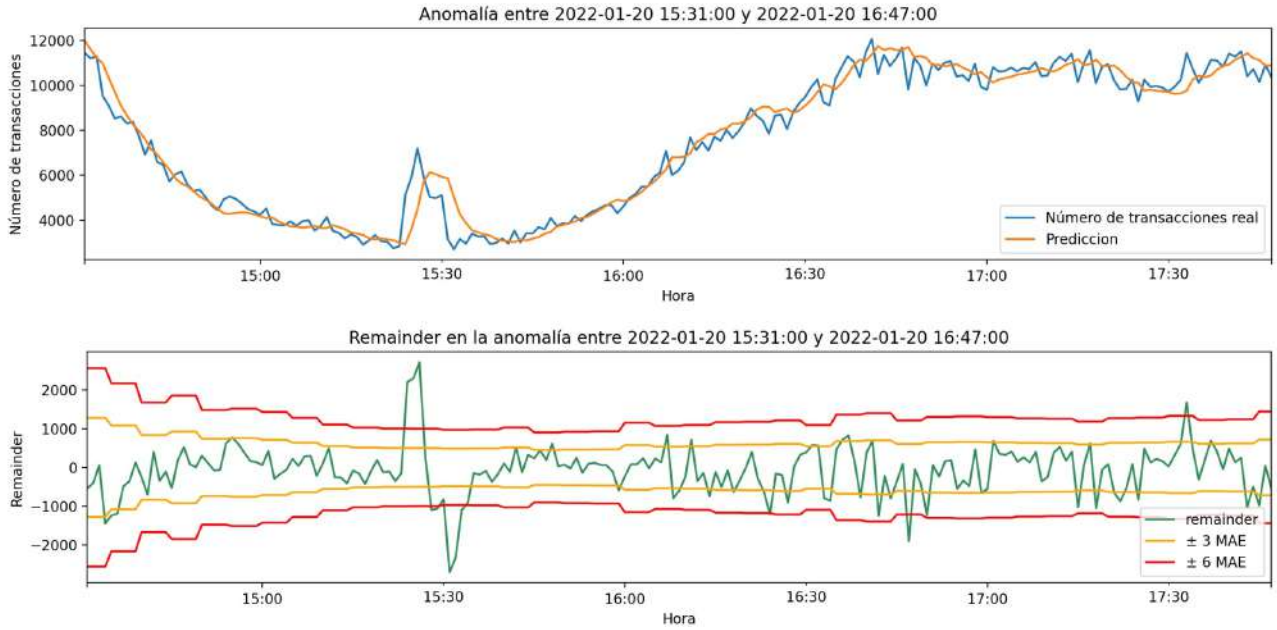
Neuronas	Batch size	Dropout	train MAE	train RMSE	MAE evaluación	RMSE evaluación
30	500	0.15	224.95	389.21	229.65	427.91
30	500	0.20	206.69	386.21	238.26	468.18
30	500	0.25	192.71	384.12	260.42	516.18
30	1000	0.15	192.73	376.77	209.93	396.55
30	1000	0.20	214.14	422.52	281.09	484.29
30	1000	0.25	222.61	424.38	238.38	451.91

Apéndice C

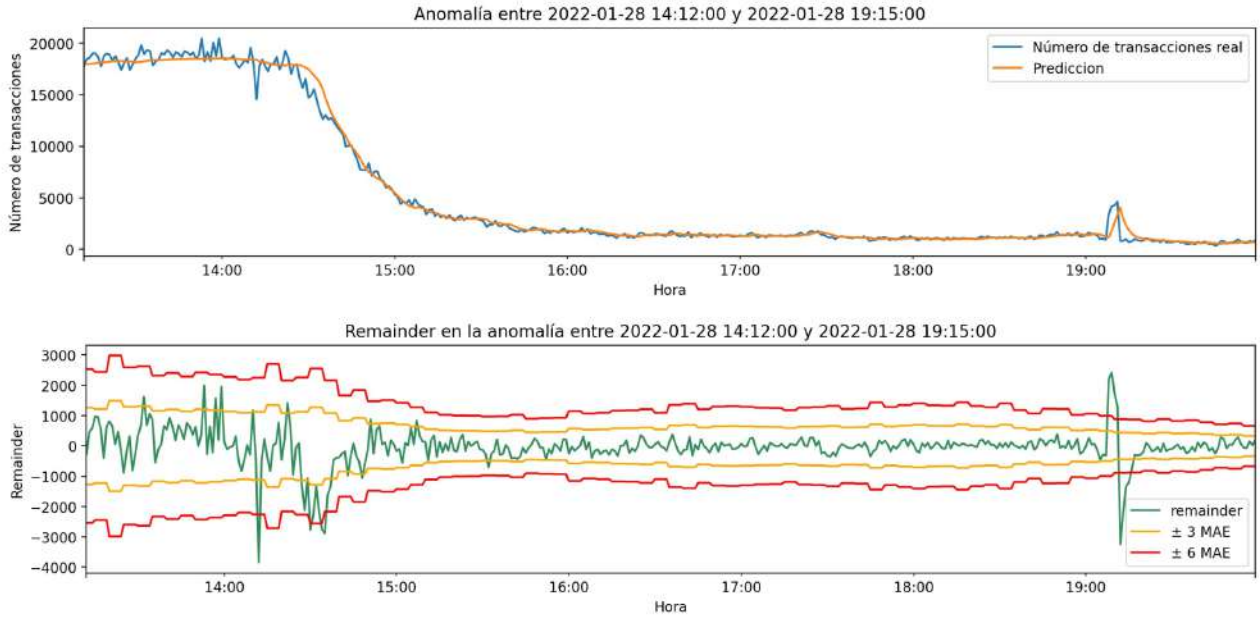
Anomalías detectadas en el modelo LSTM

En este apéndice se incluyen las representaciones gráficas de las anomalías detectadas junto con el remainder y los límites ±3 MAE y ±6 MAE, siendo MAE el Error Medio Absoluto de entrenamiento del modelo de redes neuronales LSTM a lo largo del día obtenido en períodos de 5 minutos. Estas anomalías se calculan en función de una sensibilidad, de forma que una observación se considera anomalía cuando la diferencia de las predicciones con los valores reales (remainder) supere el umbral de ±sensibilidad·MAE, siendo MAE el error medio absoluto de entrenamiento obtenido en períodos de 5 minutos. En particular, estas anomalías se han obtenido con una sensibilidad de 7.

Anomalía detectada el Jueves día 20/1/2022 de 15:31 a 16:47 horas

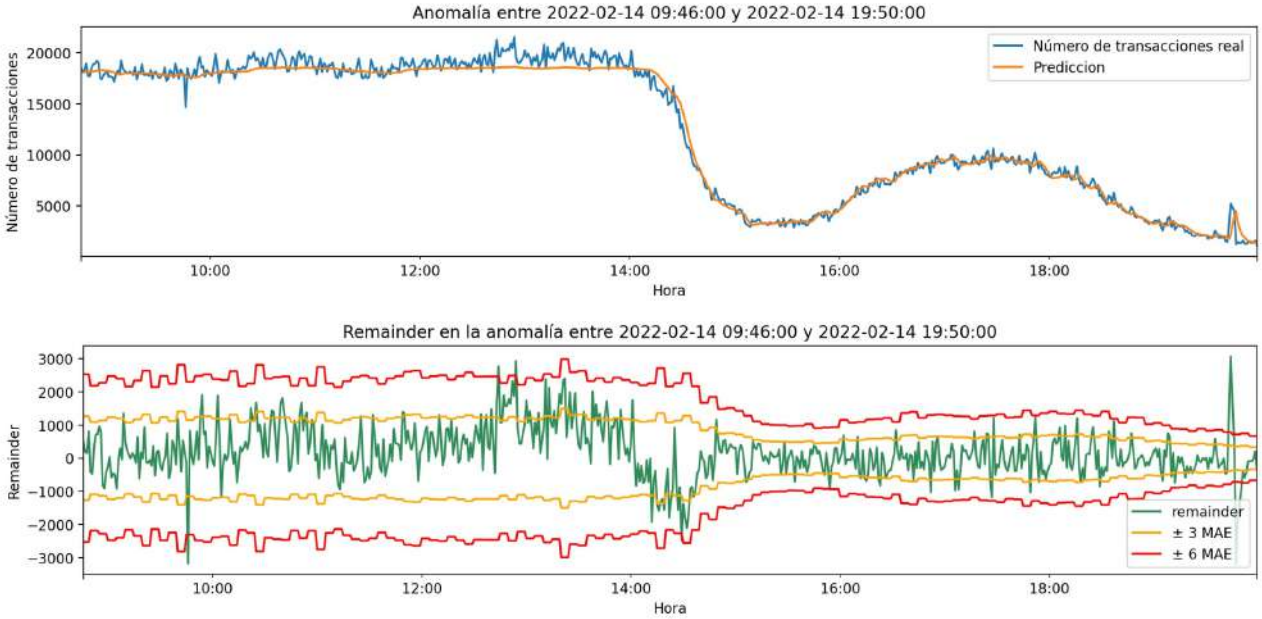


Anomalía detectada el Viernes día 28/1/2022 de 14:12 a 19:15 horas

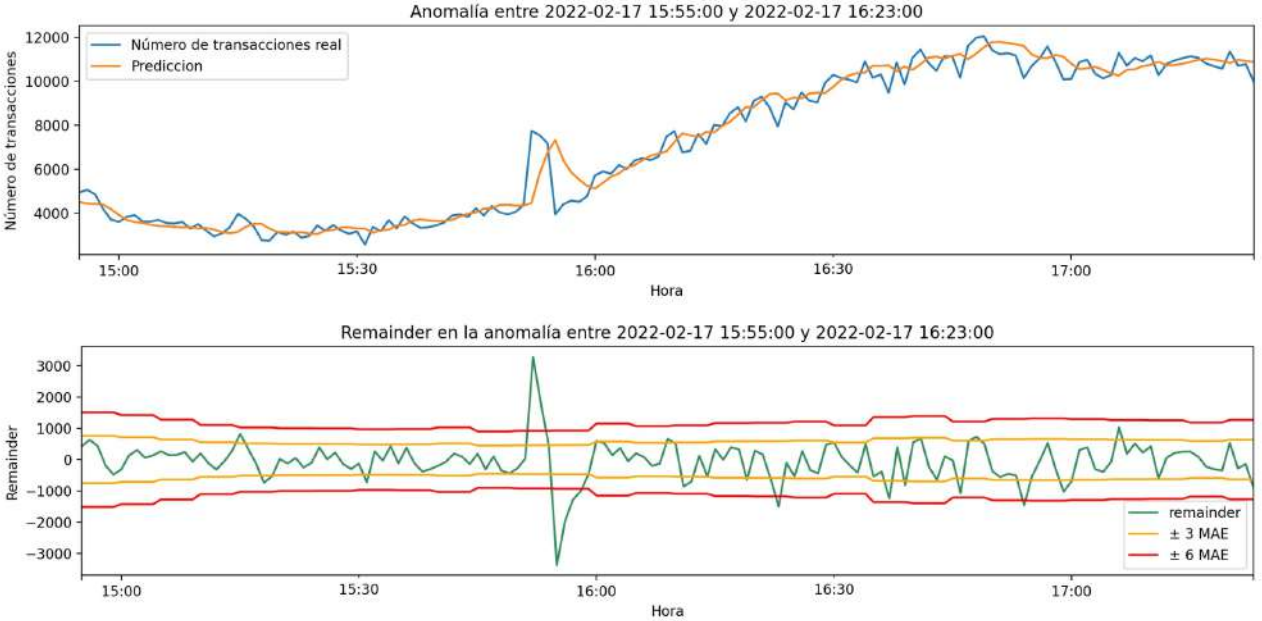


Anomalía detectada el día Viernes 4/2/2022 de 12:04 a 14:35 horas

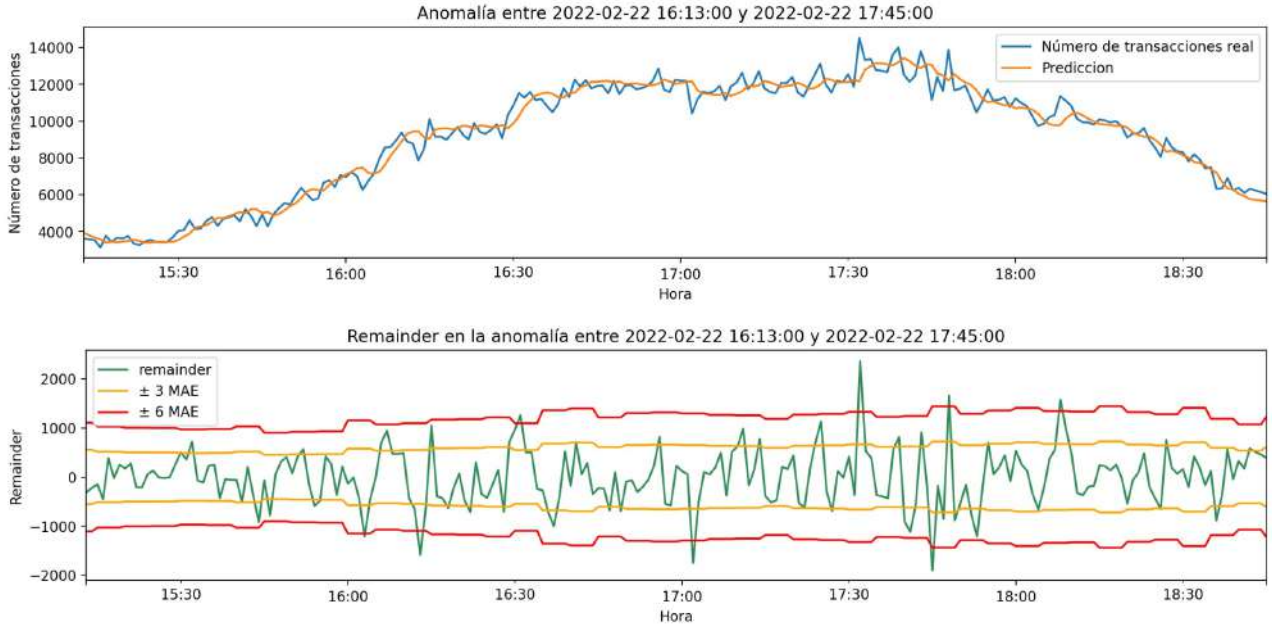
Anomalía detectada el Lunes día 14/2/2022 de 9:46 a 19:50 horas



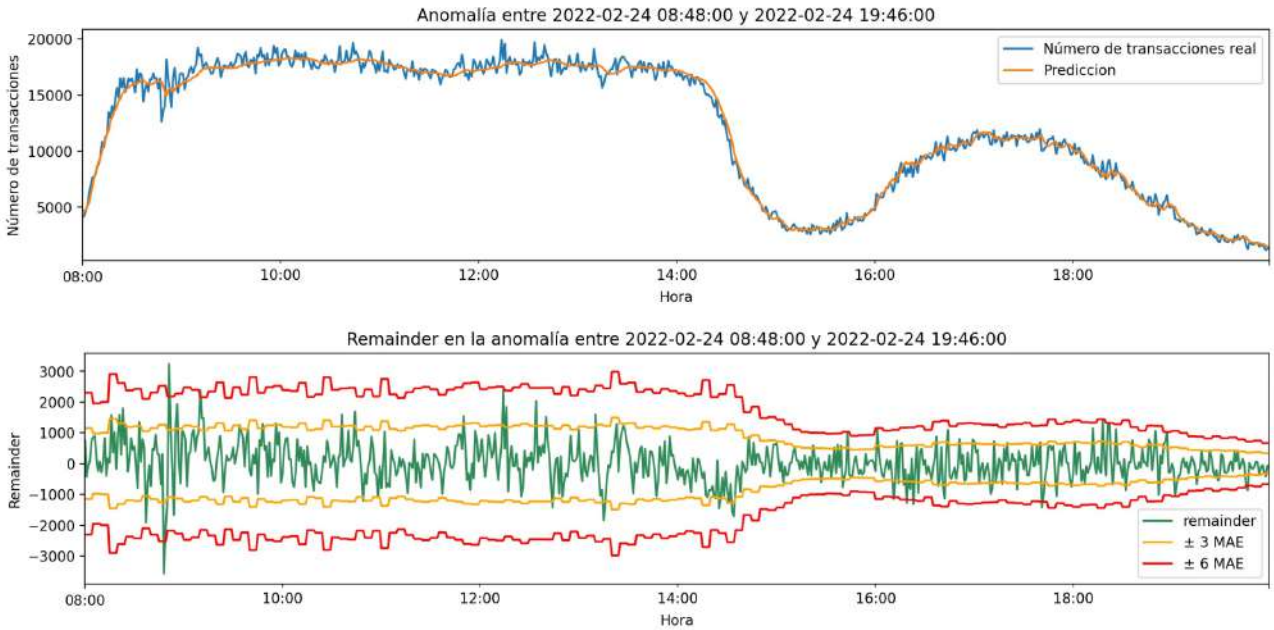
Anomalía detectada el Jueves día 17/2/2022 de 15:55 a 16:23 horas



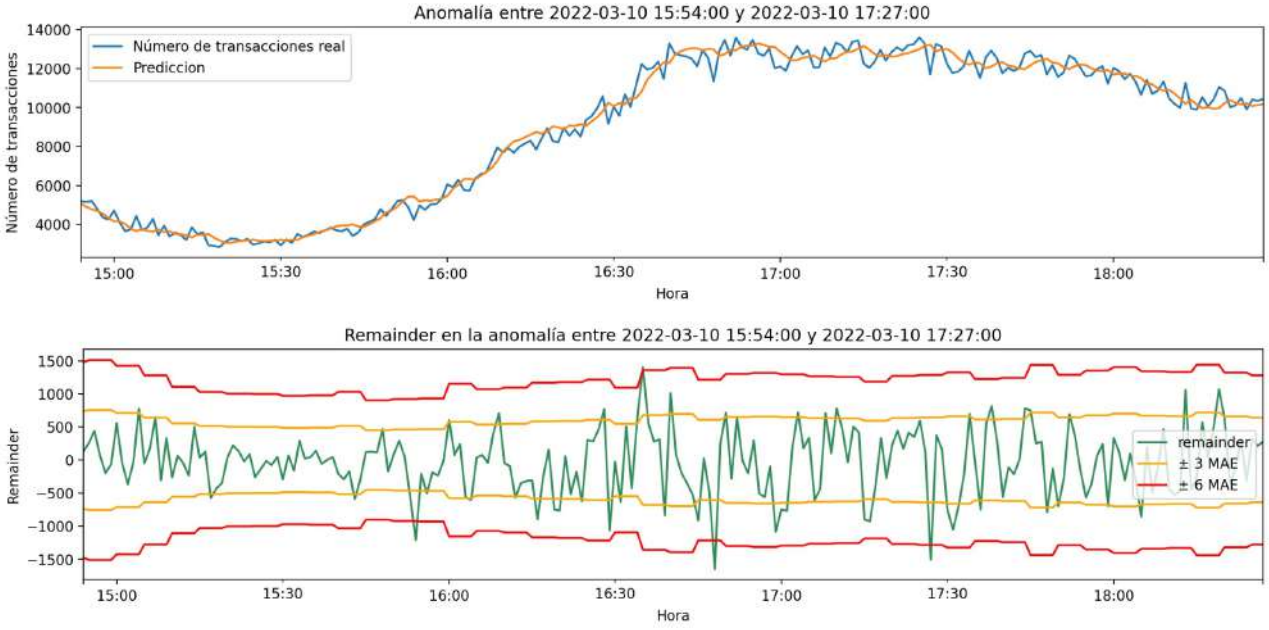
Anomalía detectada el Martes 22/2/2022 de 16:13 a 17:45 horas



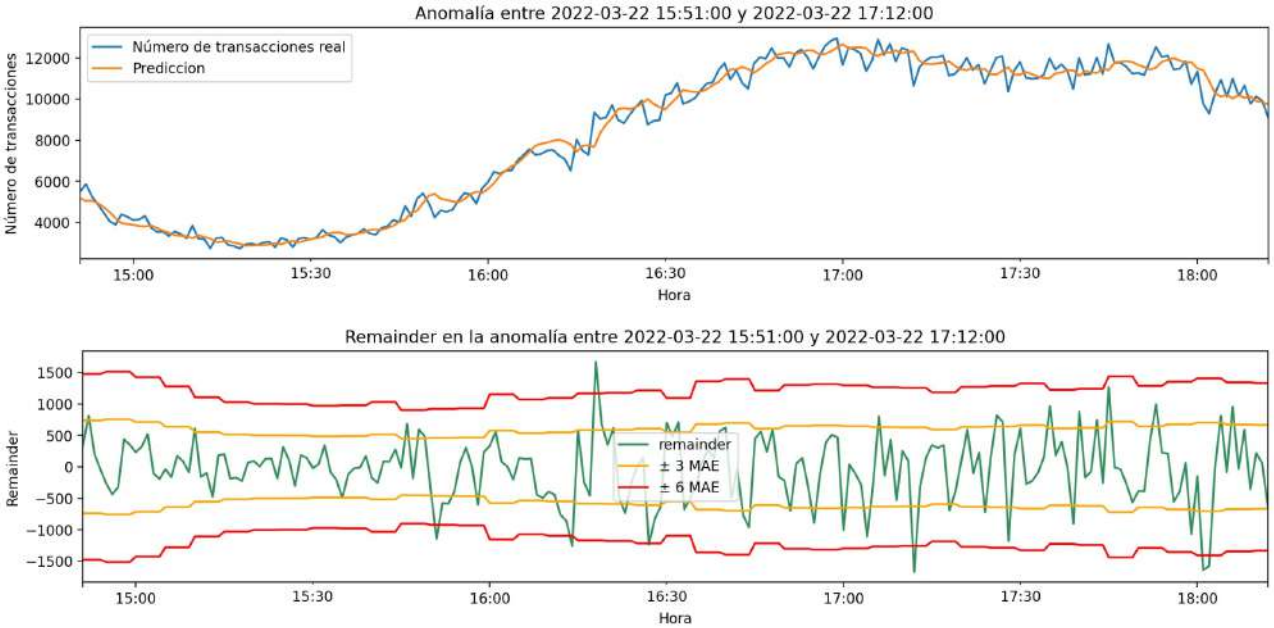
Anomalía detectada el Jueves día 24/2/2022 de 8:48 a 19:46 horas



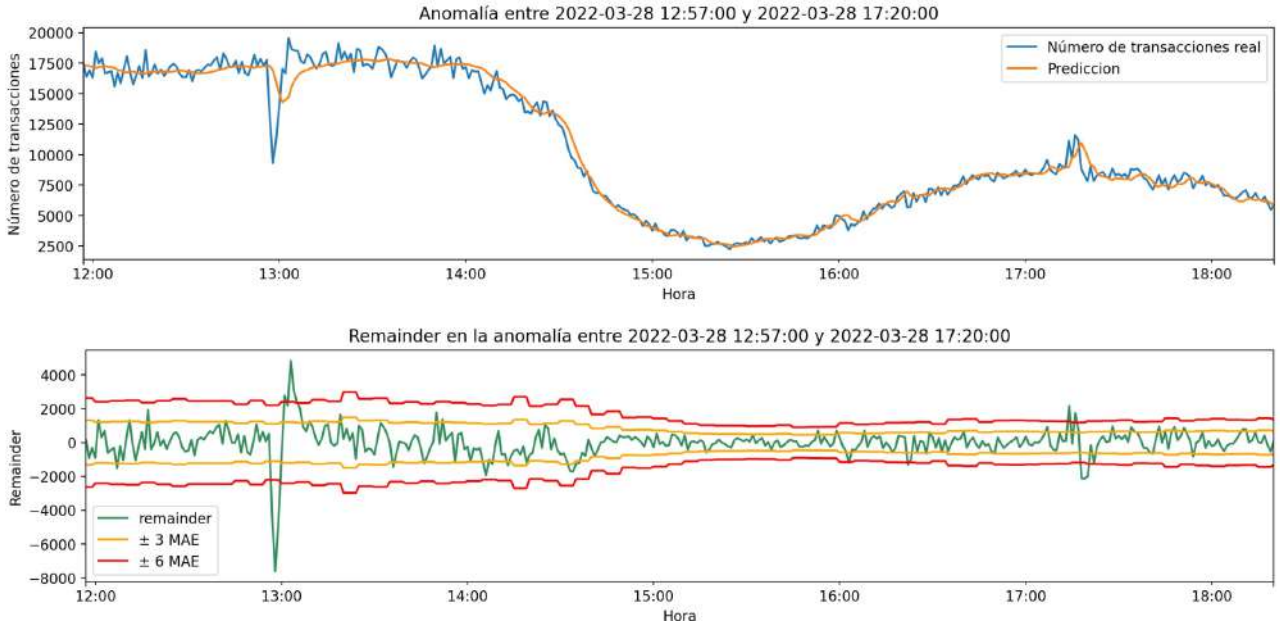
Anomalía detectada el Jueves día 10/3/2022 de 15:54 a 17:27 horas



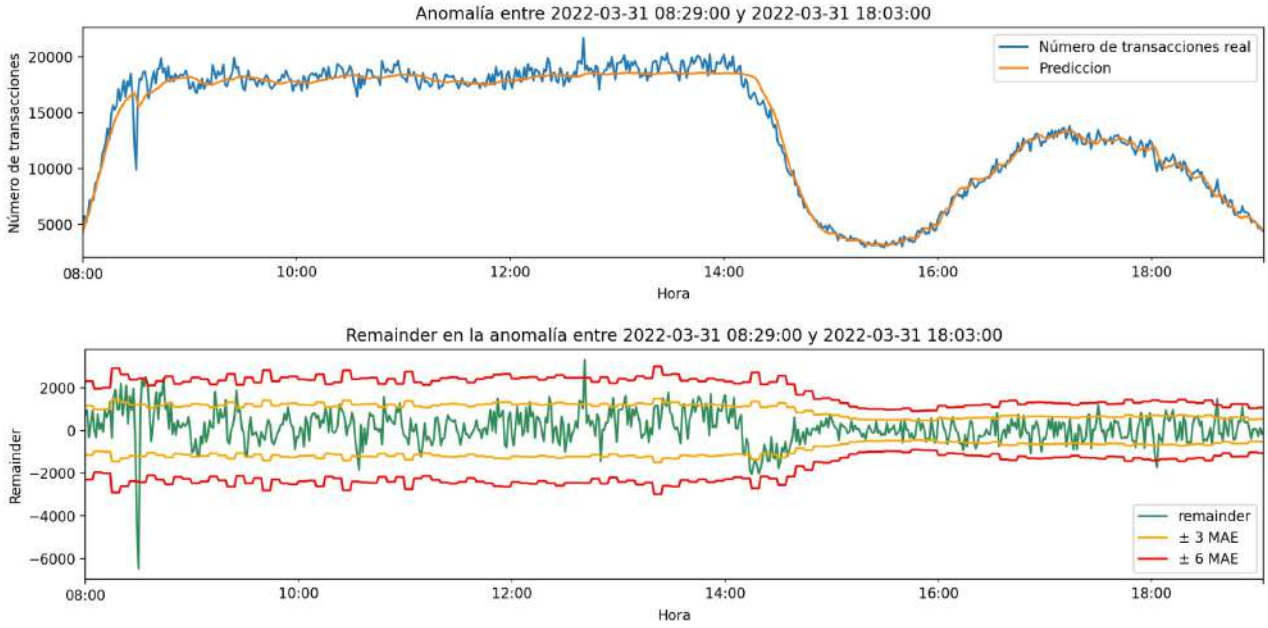
Anomalía detectada el Martes día 22/3/2020 de 15:51 a 17:12 horas



Anomalía detectada el Lunes día 28/3/2022 de 12:57 a 17:20 horas



Anomalía detectada el Jueves día 31/3/2021 de 8:29 a 18:03 horas

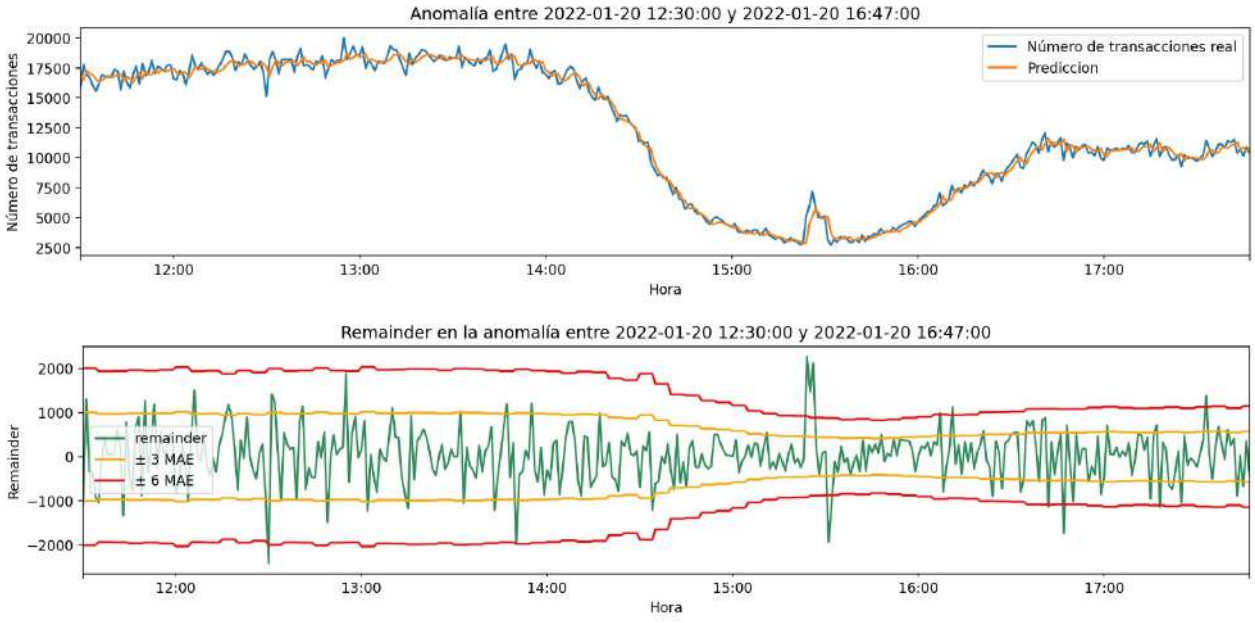


Apéndice D

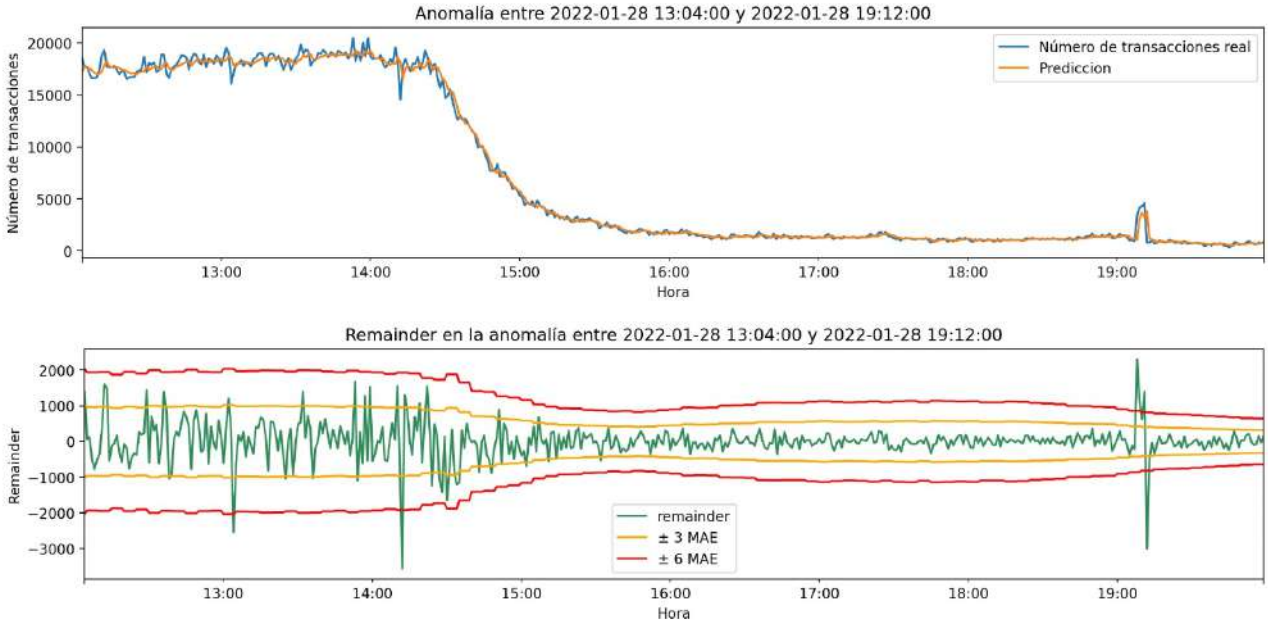
Anomalías detectadas con el modelo StackedEnsemble

En este apéndice se incluyen las representaciones gráficas de las anomalías detectadas junto al remainder y los límites ±3 MAE y ± 6 MAE, siendo MAE el error medio absoluto del entrenamiento del mejor modelo StackedEnsemble (mejor modelo ensamblado) a lo largo del día obtenido en períodos de 5 minutos. Al igual que en el modelo LSTM, estas anomalías se calculan en función de una sensibilidad de forma que una observación se considera anomalía cuando la diferencia de las predicciones con los valores reales (remainder) supera el umbral de ±sensibilidad· MAE, siendo MAE el error medio absoluto del modelo StackedEnsemble obtenido en períodos de 5 minutos. En particular, estas anomalías se han obtenido con una sensibilidad de 7.

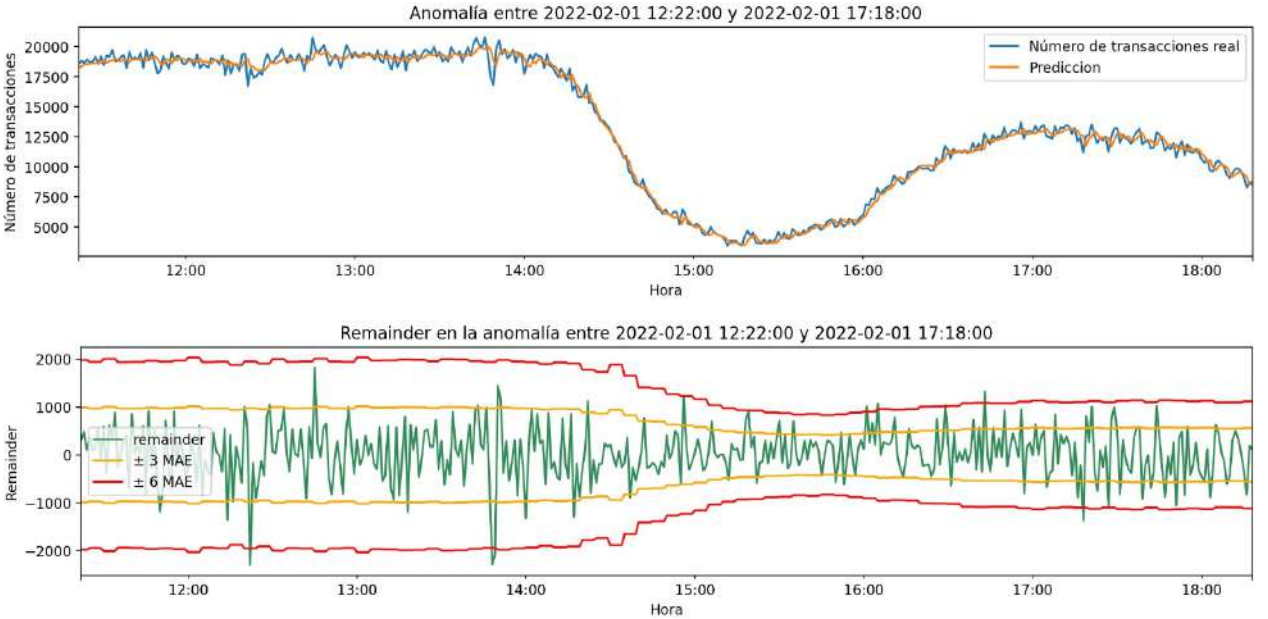
Anomalía detectada el Jueves día 20/1/2022 de 12:30 a 16:47 horas



Anomalía detectada el Viernes día 28/1/2022 de 13:04 a 19:12 horas

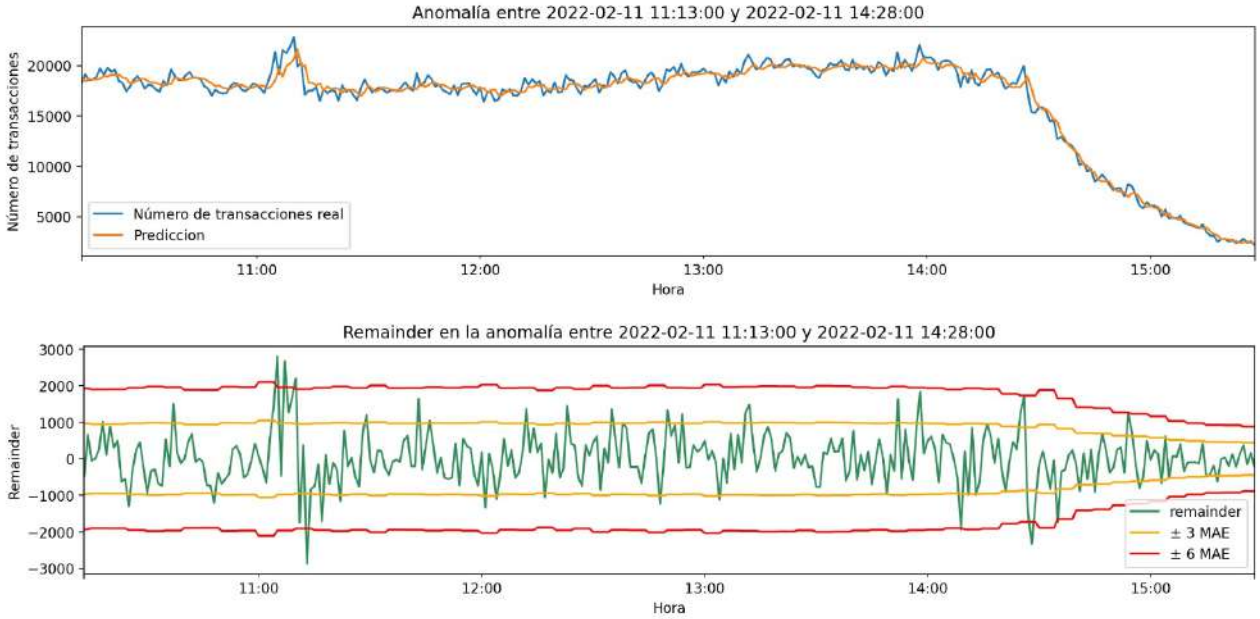


Anomalía detectada el Martes día 1/2/2022 de 12:22 a 17:18 horas

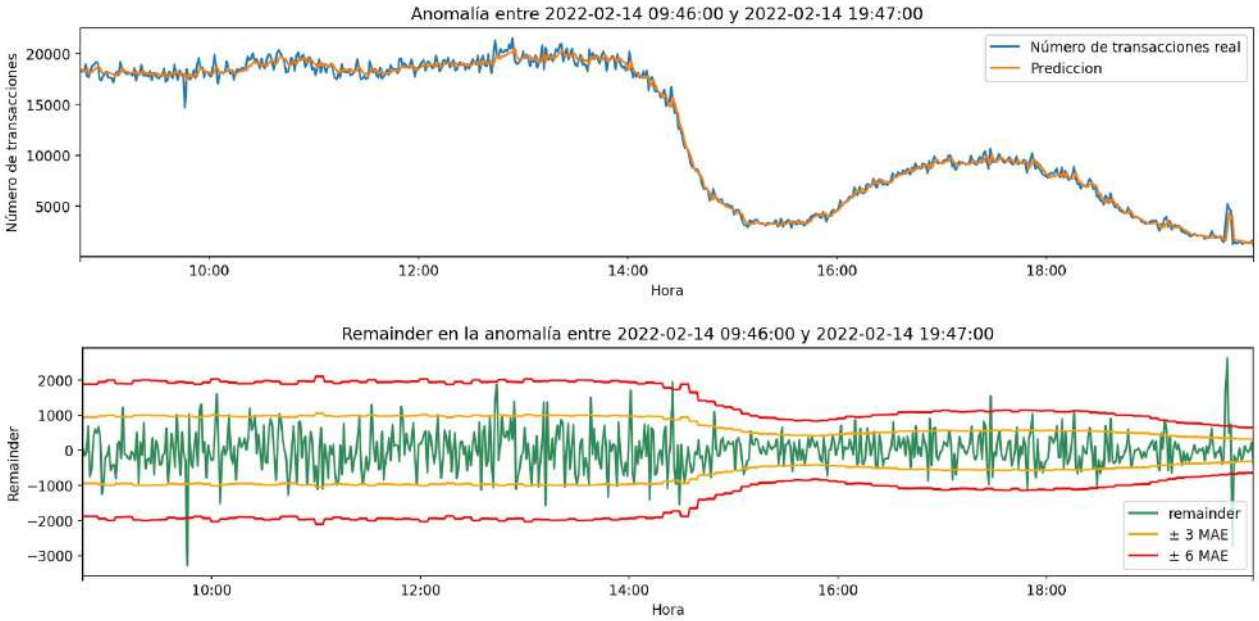


Anomalía detectada el Viernes día 4/2/2022 de 12:03 a 14:32 horas

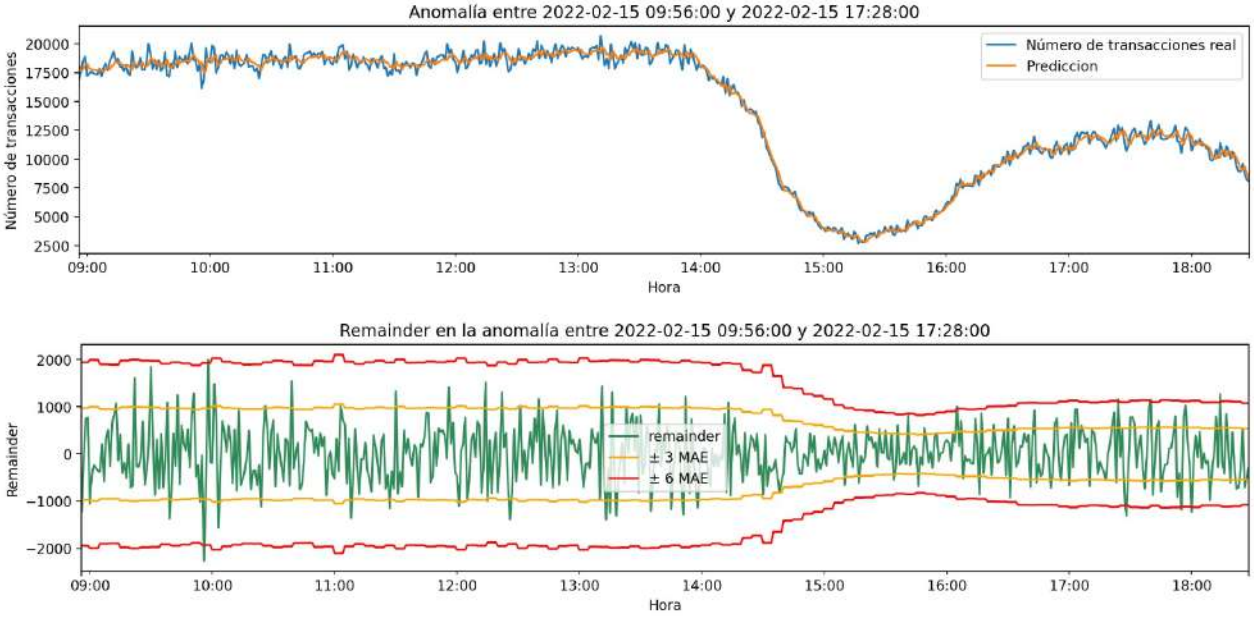
Anomalía detectada el Viernes día 11/2/2022 de 11:13 a 14:28 horas



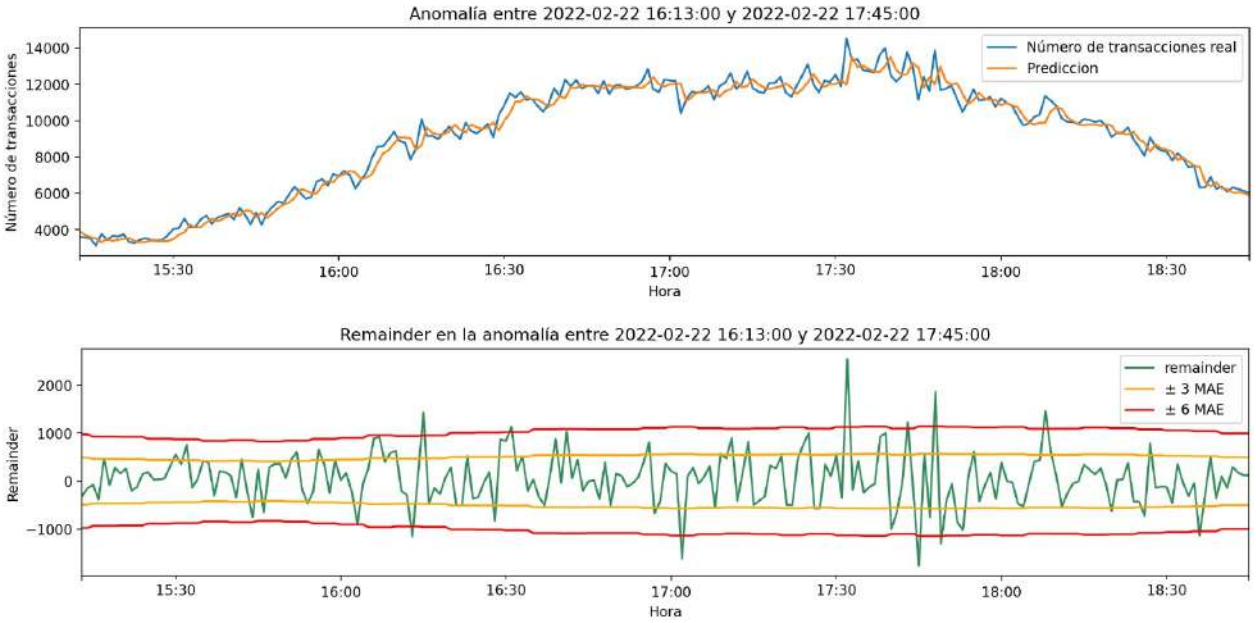
Anomalía detectada el Lunes día 14/2/2022 de 9:46 a 19:47 horas



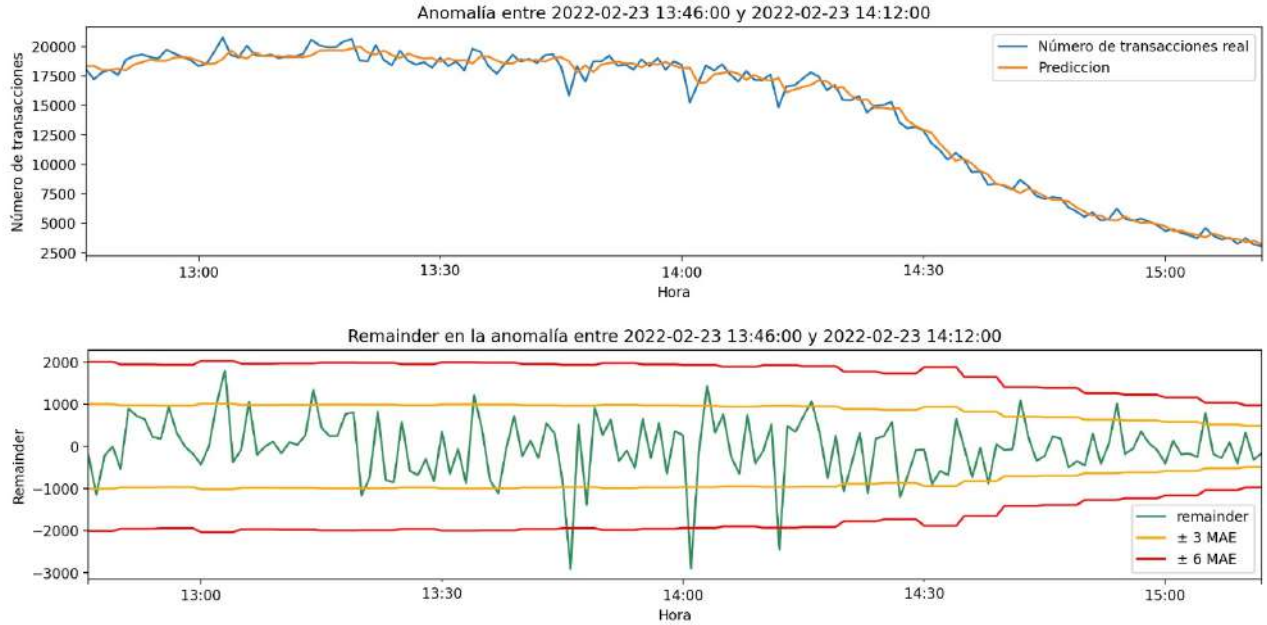
Anomalía detectada el Martes día 15/2/2022 de 9:56 a 17:28 horas



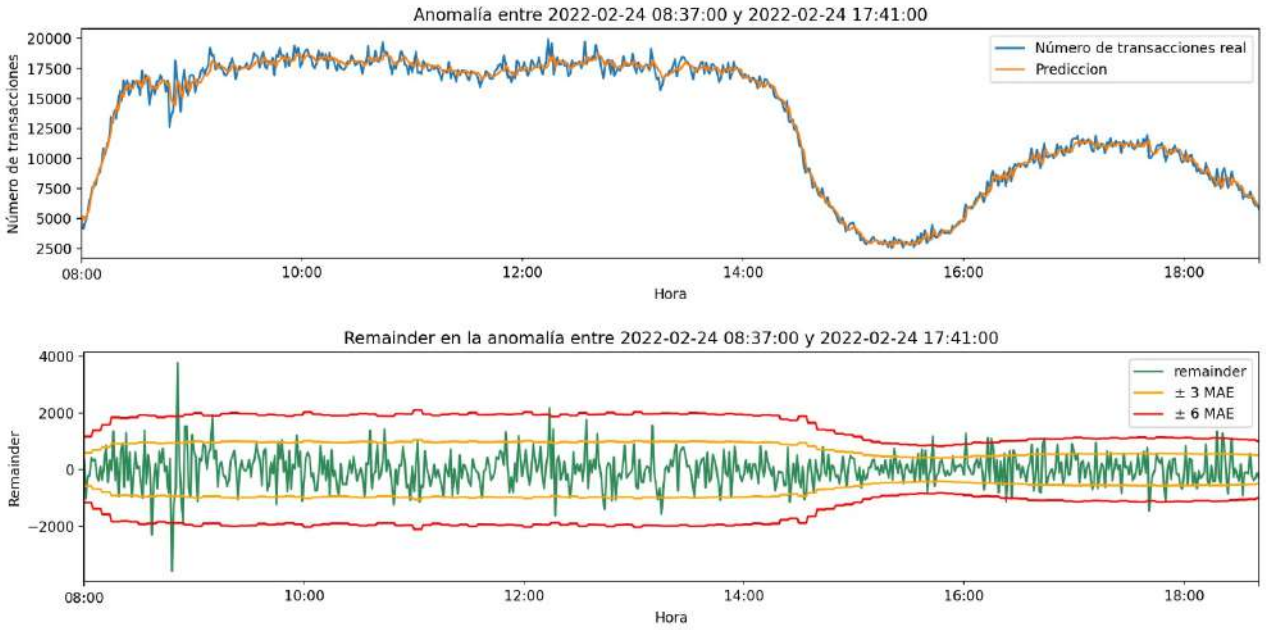
Anomalía detectada el día Miércoles 16/2/2022 de 16:13 a 17:45 horas



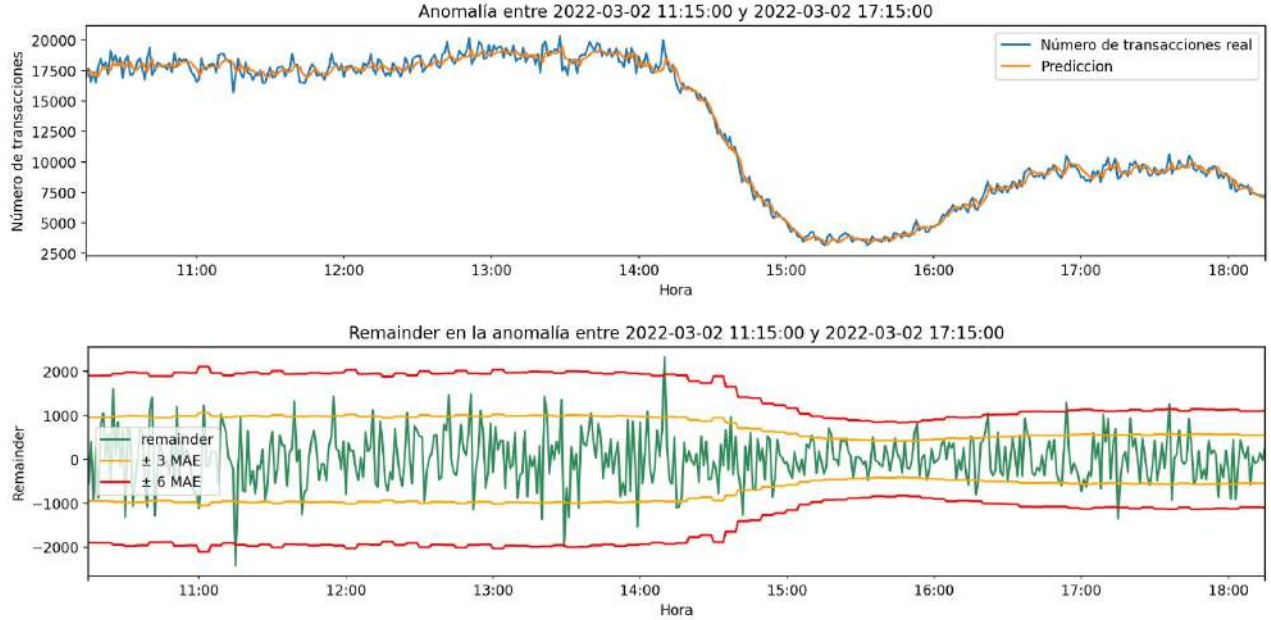
Anomalía detectada el Miércoles día 23/3/2022 de 13:46 a 14:12 horas



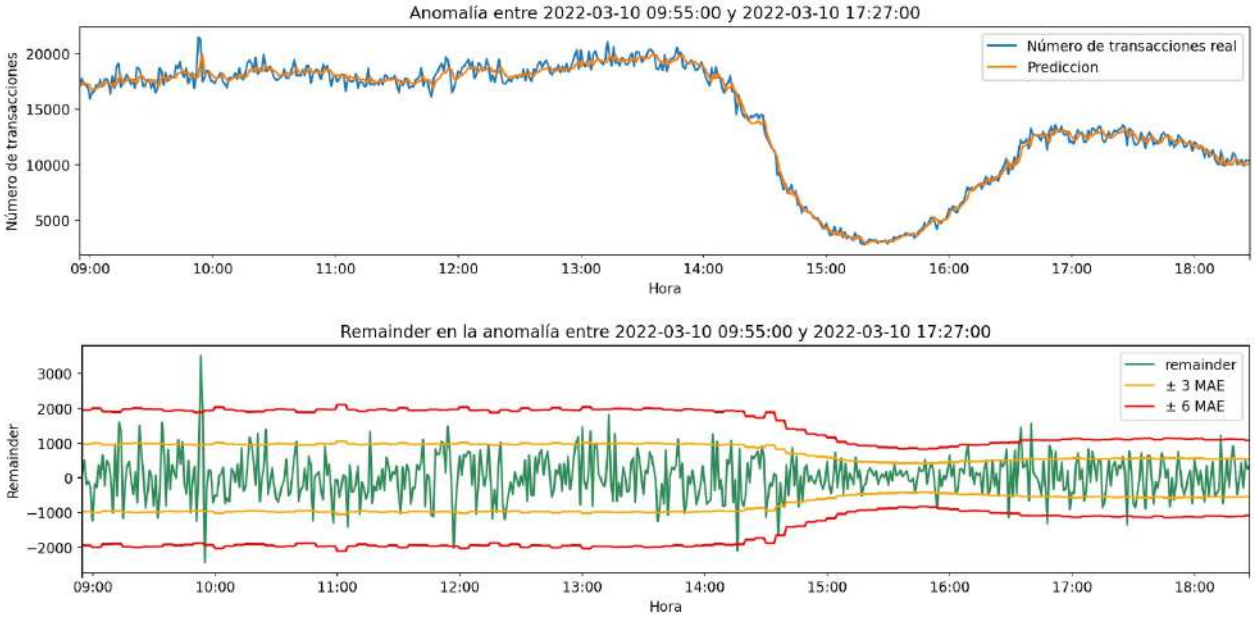
Anomalía detectada el día Jueves 24/2/2022 de 8:37 a 17:41 horas



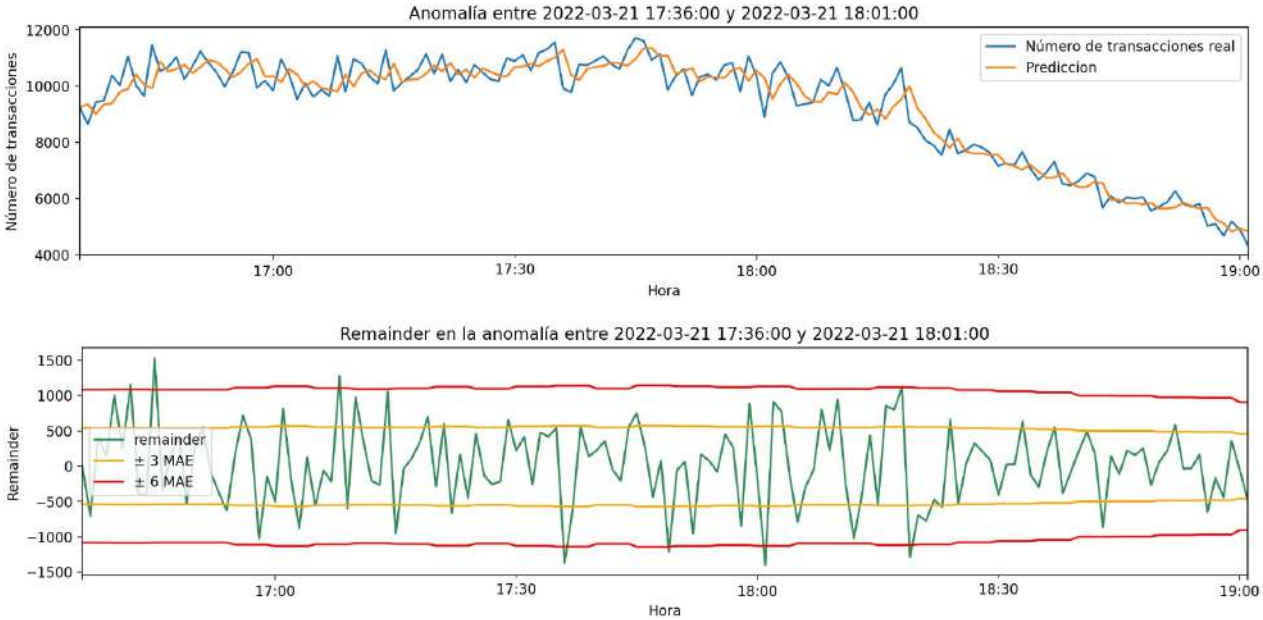
Anomalía detectada el Miércoles día 2/3/2022 de 11:15 a 17:15 horas



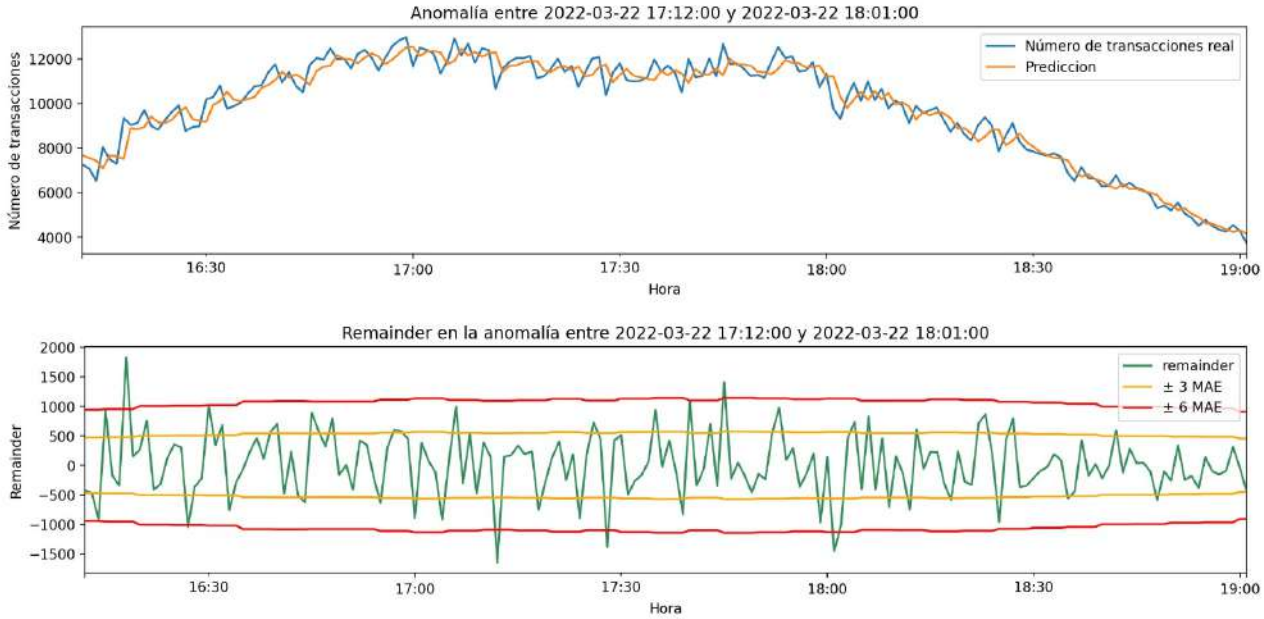
Anomalía detectada el día Jueves 10/3/2022 de 9:55 a 17:27 horas



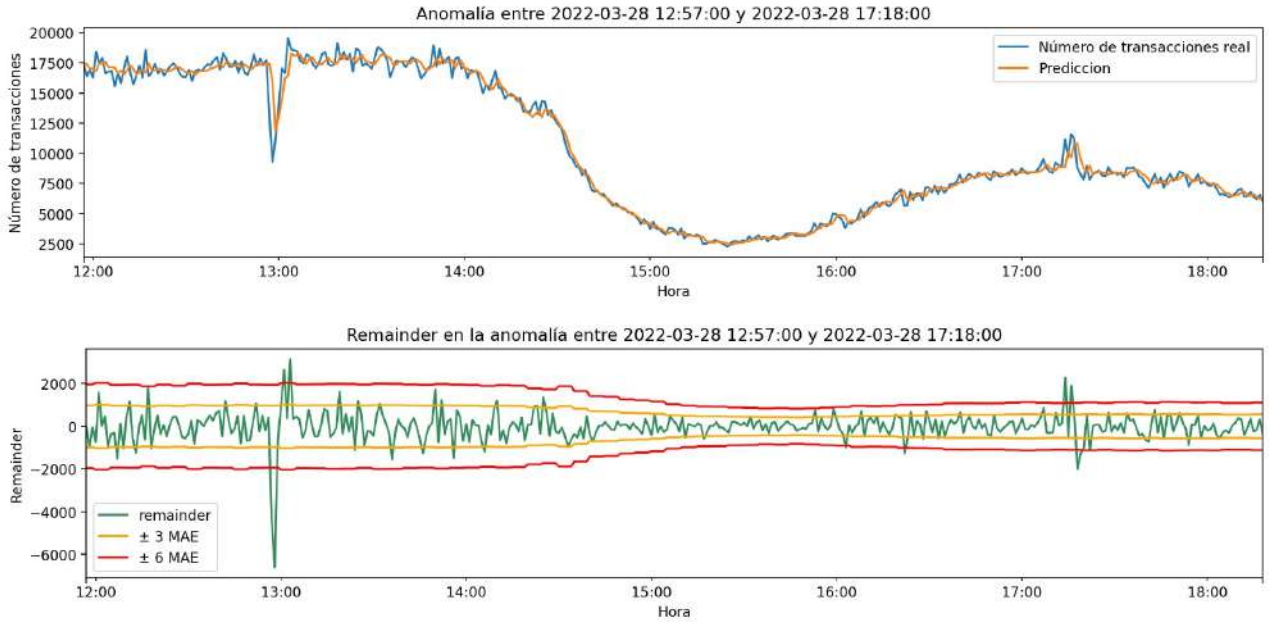
Anomalía detectada el Lunes día 21/3/2022 de 17:36 a 18:01 horas



Anomalía detectada el día Martes 22/3/2022 de 17:12 a 18:01 horas



Anomalía detectada el Lunes día 28/3/2022 de 12:57 a 17:48 horas



Apéndice E

Otros aspectos teóricos

En este apéndice se incluyen otros aspectos teóricos que se han estudiado en el Máster y que no se consideran necesarios en el capítulo de conceptos y fundamentación teórica. Los aspectos teóricos que se incluyen son los siguientes:

- Modelo autorregresivo integrado de medias móviles (ARIMA).
- Modelo ARIMA estacional multiplicativo.
- Random Forests.
- Modelos lineales generalizados (GLM).
- Gradient Boosting Machine (GBM).
- Extreme Gradient Boosting (XGBoost).

Los dos primeros son modelos clásicos de series de tiempo y los restantes son métodos de Aprendizaje Estadístico que se incluyen en la interfaz H2oAutoMl de Python.

E.1. Modelo autorregresivo integrado de medias móviles

Los modelos autorregresivos inclusivos de medias móviles, denotados por ARIMA por su abreviatura en inglés (*Autoregressive Integrated Moving Average*), son los modelos estadísticos clásicos empleados en la predicción en series temporales. Se trata de modelos que emplean los datos del pasado para predecir valores futuros. Estos modelos, desarrollados en el siglo XX, también se conocen con el nombre de modelos de Box-Jenkins por sus creadores, los estadísticos George Edward Pelham Box y Gwilym Jenkins.

Antes de proceder con la definición de los modelos ARIMA, se definen un par de modelos que sirven de base para su constucción: los modelos Autorregresivos (AR), los modelos de Medias móviles (MA) y los modelos Autorregresivos de Medias móviles (ARMA). Para la definición de los modelos, se ha consultado Shumway et al. (2017).

Modelo Autorregresivo (AR)

Un proceso estacionario que pueda representarse como

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_{p-1} X_{t-(p-1)} + \phi_p X_{t-p} + \epsilon_t$$

con $c, \phi_1, \phi_2, \cdots, \phi_{p-1}, \phi_p$ constantes, $\phi_p \neq 0$ y $\{\epsilon_t\}$ un proceso de ruído blanco, se denomina modelo autorregresivo de orden p o AR(p), por su abreviatura en inglés (*Autoregressive*). Este modelo también admite la siguiente expresión:

$$(1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_{p-1} B^{p-1} - \phi_p B^p) X_t = \epsilon_t + c$$

siendo B el operador retardo, tal que $BX_t = X_{t-1}$. Como se puede apreciar, por la formulación, se trata de un modelo que expresa los valores actuales en función de los p valores del pasado.

Modelo de Medias Móviles (MA)

Un proceso estacionario que pueda representarse como

$$X_t = c + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_{q-1} \epsilon_{t-(q-1)} + \theta_q \epsilon_{t-q}$$

con $c, \theta_1, \theta_2, \cdots, \theta_{q-1}, \theta_q$ constantes, $\theta_q \neq 0$ y $\{\epsilon_t\}$ un proceso de ruído blanco, se denomina modelo autorregresivo de orden q o MA(q), por su abreviatura en inglés (*Moving Average*). Este modelo también admite la siguiente expresión:

$$X_t = c + (1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_{q-1} B^{q-1} + \theta_q B^q) \epsilon_t$$

siendo B el operador retardo, tal que $B\epsilon_t = \epsilon_{t-1}$. Como se puede apreciar, por la formulación, se trata de un modelo que expresa el instante actual del proceso de ruído blanco como combinación lineal de los q instantes anteriores.

Modelo Autorregresvivo de Medias Móviles (ARMA)

Un proceso estacionario que pueda representarse como

$$X_t = c + \sum_{i=1}^{p} \theta_i X_{t-i} + \epsilon_t + \sum_{j=1}^{q} \theta_j \epsilon_{t-j}$$

con $c, \phi_1, \phi_2, \cdots, \phi_{p-1}, \phi_p, \theta_1, \theta_2, \cdots, \theta_{q-1}, \theta_q$ constantes, $\phi_p, \theta_q \neq 0$ y $\{\epsilon_t\}$ un proceso de ruído blanco, se denomina modelo autorregresivo de medias móviles de orden p y q o ARMA(p,q), por su abreviatura en inglés (*Autoregressive Moving Average*). Este modelo también admite la siguiente expresión:

$$\left(1 - \sum_{i=1}^{p} \phi_i B^i\right) X_t = c + \left(1 + \sum_{j=1}^{q} \theta_j B^j\right) \epsilon_t$$

Se puede apreciar que, este modelo se construye combinando el modelo Autorregresivo y el modelo de Medias móviles. En particular, si $p = 0$, se tiene un modelo de Medias móviles y, si $q = 0$, se tiene un modelo Autorregresivo.

Tras la definición de los anteriores modelos, se definen los llamados modelos ARIMA, que dan título a esta sección. Este tipo de modelos son los que se emplean cuando la serie de tiempo presenta tendencia. Un proceso estocástico $\{X_t\}$ se denomina **modelo autorregresivo inclusivo de medias móviles de orden p, d y q**, ARIMA(p,d,q), cuando el proceso $(1-B)^d X_t$ es un modelo ARMA(p,q), es decir, es aquel modelo que, tras aplicarle d diferencias regulares, resulta en un modelo ARMA. Su formulación matemática es la siguiente:

$$(1 - B)^d X_t = c + \sum_{i=1}^{p} \theta_i X_{t-i} + \epsilon_t + \sum_{j=1}^{q} \theta_j \epsilon_{t-j}$$

De forma equivalente:

$$\left(1-\sum_{i=1}^{p}\phi_i B^i\right)(1-B)^d X_t = c+\left(1+\sum_{j=1}^{q}\theta_j B^j\right)\epsilon_t$$

El orden d representa el número de veces que hay que diferenciar la serie para eliminar la tendencia.

E.2. Modelo ARIMA estacional multiplicativo

Supongamos, a continuación, que la serie de tiempo que ha sido generada a partir de un proceso estocástico posee algún tipo de componente estacional (que puede ser anual, o semanal, o diaria, entre otras). Los modelos ARIMA que se han definido anteriormente modelizan únicamente la dependencia entre observaciones pasadas y no tienen en cuenta la componente estacional. Por lo tanto, en estos casos, es necesario introducir un nuevo tipo de modelos que permitan incluír la componente estacional: los llamados **modelos ARIMA estacionales multiplicativos**. Antes de proceder con la definición de estos modelos, introducimos los modelos ARMA y ARIMA estacionales.

Modelo Autorregresivo de Medias móviles estacional

Un proceso estacionario $\{X_t\}$ que pueda representarse como

$$X_t = c+\sum_{i=1}^{P}\Phi_i X_{t-is}+\epsilon_t+\sum_{j=1}^{Q}\Theta_j\epsilon_{t-js}$$

con $c,\Phi_1,\Phi_2,\cdots,\Phi_{P-1},\Phi_P,\Theta_1,\Theta_2,\cdots,\Theta_{Q-1},\Theta_Q$ constantes, $\Phi_P,\Theta_Q\neq 0$ y $\{\epsilon_t\}$ un proceso de ruído blanco, se denomina modelo autorregresivo de medias móviles estacional de orden P, Q y s, ARMA(P,Q)$_s$ (o ARMA(P,Q) estacional de orden s). Este modelo también admite la siguiente formulación:

$$\left(1-\sum_{i=1}^{P}\Phi_i B^{is}\right)X_t = c+\left(1+\sum_{j=1}^{Q}\Theta_j B^{js}\right)\epsilon_t$$

siendo B el operador retardo estacional, tal que $B^s X_t = X_{t-s}$ y $B^s\epsilon_t=\epsilon_{t-s}$. Se puede apreciar que, en particular, si $P=0$, se tiene un modelo de medias móviles estacional de orden s (MA(Q)$_s$) y, si $Q=0$, se tiene un modelo autorregresivo estacional de orden s (AR(P)$_s$). Por su formulación, estos modelos modelizan la componente estacional presente en la serie de tiempo.

Modelo Autorregresivo inclusivo de Medias móviles estacional

Un proceso estocástico $\{X_t\}$ se denomina modelo autorregresivo inclusivo de medias móviles estacional de orden P, D, Q y s, ARIMA(P,D,Q)$_s$, cuando, tras aplicarle D diferencias estacionales de período s (para eliminar la componente estacional), resulta en un modelo ARMA(P,Q)$_s$. Su formulación matemática es la siguiente:

$$(1-B^s)^D X_t = c+\sum_{i=1}^{P}\Phi_i X_{t-is}+\epsilon_t+\sum_{j=1}^{Q}\Theta_j\epsilon_{t-js}$$

De forma equivalente:

$$\left(1-\sum_{i=1}^{P}\Phi_i B^{si}\right)(1-B^s)^D X_t = c+\left(1+\sum_{j=1}^{Q}\Theta_j B^{sj}\right)\epsilon_t$$

siendo B el operador retardo estacional, tal que $B^s X_t = X_{t-s}$ y $B^s \epsilon_t = \epsilon_{t-s}$.

Si se combinan los modelos ARIMA, definidos en la anterior sección, y los modelos ARIMA estacionales que acabamos de definir, se obtiene un nuevo modelo que modeliza la componente estacional y la dependencia entre las observaciones del pasado. Este modelo se llama habitualmente **modelo autorregresivo inclusivo de medias móviles estacional multiplicativo**. Su expresión matemática es la siguiente:

$$\left(1 - \sum_{i=1}^{p} \phi_i B^i\right)\left(1 - \sum_{i=1}^{P} \Phi_i B^{si}\right)(1-B)^d(1-B^s)^D X_t = c + \left(1 + \sum_{j=1}^{q} \theta_j B^j\right)\left(1 + \sum_{j=1}^{Q} \Theta_j B^{sj}\right)\epsilon_t$$

Habitualmente se denota por ARIMA(p,d,q)$\times$(P,D,Q)$_s$, siendo (p,d,q) los órdenes de la parte que modeliza la dependencia entre las observaciones del pasado y (P,D,Q)$_s$ los órdenes de la parte estacional. Comentar que d representa el número de veces que se aplica diferenciación regular (para eliminar la tendencia de la serie), s representa la componente estacional de la serie y D es el número de veces que se aplican diferencias estacionales de período s (para eliminar la componente estacional de la serie).

Para mayor información acerca de la construcción de estos modelos puede consultarse Shumway et al. (2017).

E.3. Random Forests

Los bosques aleatorios, comúnmente conocidos por su nombre en inglés (*Random Forests*), son una modificación de la técnica Bagging (o *Bootstrap Aggregation*) para emplear árboles de decisión. Su metodología se basa en la construcción de un gran número de árboles de decisión tratando de que sean no correlacionados para luego promediarlos. La gran ventaja de estos modelos es la facilidad en el ajuste y en el entrenamiento.

Para explicar el funcionamiento de estos modelos, se introducen los conceptos de árboles de decisión y la técnica *Bagging*. Seguiremos como referencia el capítulo titulado *Tree-Based Methods* de Hastie et al. (2009).

Árboles de decisión

Los árboles de decisión son los métodos más sencillos y de fácil interpretación para realizar predicciones en problemas de regresión y clasificación considerando tanto variables predictoras numéricas como categóricas. Se trata de métodos que particionan el conjunto de variables predictoras en una serie de rectángulos con la finalidad de ajustar un modelo sencillo en cada uno de ellos (por ejemplo, una constante). La partición se representa mediante un árbol binario. Pongamos, por ejemplo, el caso de la regresión con variables predictoras X_1 y X_2 y variable respuesta Y. La idea bajo los árboles de decisión es la siguiente:

- Se parte de un nodo inicial que represente a toda la muestra de entrenamiento.
- Del nodo inicial parten dos ramas bien diferenciadas, que dividen la muestra en dos subconjuntos. En cada una de esas ramas se considera, si es necesario, una nueva partición. Por ejemplo, en la Figura E.1 hay una la rama indicando el caso $X_1 \geq t_1$ y la otra indicando el caso $X_1 < t_1$. En ambas ramas se realiza otra partición de la muestra.
- El proceso se repite un número finito de veces hasta alcanzar los nodos terminales, denominados podas, que se denotan habitualmente por $R_i, i = 1, \cdots, n$. Por ejemplo, en la Figura E.1, se consideran 3 particiones diferentes y se obtiene un total de 5 nodos terminales: $R_i, i = 1, \cdots, 5$.

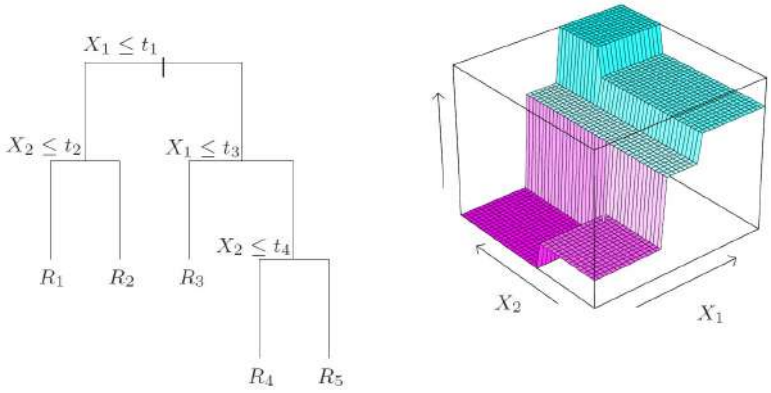


Figura E.1: Esquema del árbol de decisión correspondiente a las particiones descritas anteriormente. A la derecha se representa gráficamente la superficie de predicción de la variable respuesta Y correspondiente al árbol de decisión que se acaba de definir. Fuente: Hastie et al. (2009)

La metodología más conocida para modelizar los árboles de decisión es la metodología llamada ***C**lasification **A**nd **R**egression **T**rees* (CART). Con respecto a la formulación, no se entra en detalle (puede consultarse Hastie et al. (2009)), pero sí se realiza una introducción del cálculo de las constantes para realizar las predicciones en los diferentes modelos. En el caso de los árboles de regresión, una vez particionada la muestra en R_i, $i = 1, \cdots, M$ regiones, se calcula, en cada una de las regiones, la media de la variable respuesta Y en la región. En el caso de los modelos de clasificación, se toma como constante, en cada una de las regiones, la categoría más frecuente de la muestra de entrenamiento (categoría modal).

Bagging (Bootstrap Agregation)

Bagging (o *Bootstrap Agregation*) es un método de reducción de la varianza de un modelo de árbol de decisión. Es un método llamado de ensamblaje, pues combina las predicciones obtenidas con múltiples modelos, empleando para ello métodos bootstrap. Pongamos el caso de que se ha ajustado un modelo a los datos de entrenamiento $Z = \{(x_1, y_1), \cdots (x_N, y_N)\}$, con predicciones $\widehat{f}(x)$ para la variable de entrada x. El método *Bagging* (o *Boostrap Aggregation*) promedia estas predicciones sobre un conjunto de muestras Bootstrap, es decir, para cada Z^{b*}, $b = 1, \cdots, B$, con B el número de muestras Bootstrap, dada $\widehat{f}^{*b}(x)$ la predicción en cada una de las muestras Bootstrap, el estimador *Bagging* viene dado por

$$\widehat{f}(x) = \frac{1}{B} \sum_{b=1}^{B} \widehat{f}^{*b}$$

En el caso particular de considerar B árboles de decisión, el estimador *Bagging* sería la media de las predicciones en la variable x de los B árboles. Este caso se corresponde con los llamados *Random Forests*, que se describen a continuación.

Tal y como se comentó en la introducción de estos modelos, los *Random Forests* son una modificación del método de ensamblaje *Bagging* para emplear árboles de decisión. La idea de estos métodos es reducir la correlación entre los diferentes árboles de decisión (sin aumentar la varianza) con la finalidad de mejorar y reducir la varianza obtenida con el método *Bagging*. Para ello se incluye aleatoriedad en las variables de entrada para realizar la construcción de los árboles y así evitar la correlación entre los mismos, puesto que si se promedian variables altamente incorreladas, se consigue una reducción de la varianza considerable. En efecto, dadas B muestras bootstrap idénticamente distribuídas con varianza

σ^2 y correlación positiva entre dos muestras ρ, la varianza de la media de las muestras es

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

Nótese que, a medida que aumenta el número de muestras Bootstrap B, el término $\frac{1-\rho}{B}\sigma^2$ tiende a 0, lo que indica que la correlación entre árboles beneficia la reducción de la varianza (término $\rho\sigma^2$).

Para evitar la correlación entre árboles de decisión, en la construcción de cada uno de ellos, antes de realizar cada una de las particiones, dado p el número de variables predictoras, se deben seleccionan al azar $m \leq p$ predictores candidatos para la partición. Los valores habituales del hiperparámetro m suelen ser valores cercanos a 1. Habitualmente se considera $m = \sqrt{p}$ en problemas de clasificación y $m = 1/3$ en problemas de regresión. El número de árboles a tener en cuenta para la construcción del modelo es, en general, superior al número de árboles del método *Bagging*, para garantizar convergencia.

Algoritmo E.1 (*Random Forest*). [1] *Sea B el número de árboles de decisión:*

- *Para cada $b = 1, \cdots, B$, se considera una muestra Bootstrap de tamaño N de la muestra de entrenamiento, Z^*, y se construye el árbol de decisión correspondiente, f^{*b}, siguiendo los siguientes pasos, de forma recursiva, hasta alcanzar el número mínimo de nodos:*

  (1) Se seleccionan al azar $m \leq p$ predictores para la partición.

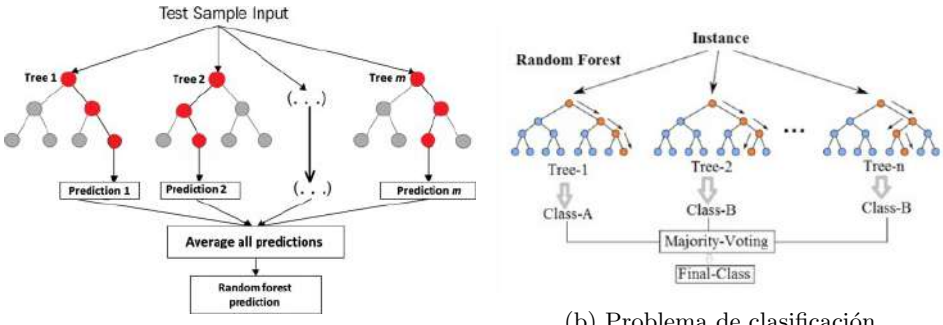
  (2) Se escoge la mejor partición de entre los m predictores considerados.

  (3) Se particiona el nodo en dos ramas bien diferenciadas.

- *Se obtiene el conjunto de árboles de decisión $\{f^{*b}\}_{b=1}^{B}$ y se calculan las predicciones correspondientes. En problemas de clasificación se toma la predicción según el criterio del voto mayoritario (majority voting): asignadas las etiquetas del nodo terminal en cada uno de los árboles, se toma como predicción aquella etiqueta que ocurre un mayor número de veces. En el caso de problemas de regresión, la predicción se calcula promediando las predicciones en los nodos terminales de cada uno de los árboles:*

$$\frac{1}{B}\sum_{b=1}^{B} f^{*b}(x)$$

siendo $f^{*b}(x)$ la predicción en el nodo terminal del árbol de decisión b.



(a) Problema de regresión.

(b) Problema de clasificación

Figura E.2: (a) Esquema del algoritmo *Random Forest* para un problema de regresión. Fuente: Afdal et al. (2020). (b) Esquema del algoritmo de Random Forest para un problema de clasificación. Fuente: Raja et al. (2019).

[1] Puede encontrarse en el capítulo titulado *Random Forests* de Hastie et al. (2009).

E.4. Modelos lineales generalizados

Los modelos lineales generalizados, denotados por GLM por su abreviatura en inglés (*Generalized Linear Models*), son una extensión de los modelos de regresión lineal para aquellos casos en los que la distribución condicional de la variable respuesta Y no sea normal.

Los modelos lineales expresan la variable respuesta Y como combinación lineal de las variables predictoras:

$$Y = \beta_0 + \sum_{i=1}^{n} \beta_i X_i + \epsilon$$

con $\epsilon \sim N(0, \sigma^2)$ el término de error. De forma equivalente:

$$\mathbb{E}(Y|X) = \beta_0 + \sum_{i=1}^{n} \beta_i X_i$$

El término de error tiene una distribución continua, por lo que, necesariamente, si la variable respuesta admite la expresión anterior, su distribución debe ser continua. En los casos en los que la distribución de la variable respuesta no sea continua, por ejemplo con una variable respuesta con distribución binomial, los modelos de regresión lineal pueden no ser adecuados, por lo que es necesario introducir los modelos lineales generalizados. Estos modelos, desarrollados por John Nelder y Robert Wedderburn, expresan la distribución condicional de la variable respuesta Y del siguiente modo:

$$G(\mathbb{E}(Y|X)) = \beta_0 + \sum_{i=1}^{n} \beta_i X_i$$

siendo G la función enlace (*link function*), que debe ser invertible para garantizar que

$$\mathbb{E}(Y|X) = G^{-1}(\beta_0 + \sum_{i=1}^{n} \beta_i X_i)$$

Estos modelos se emplean en los casos en los que la distribución condicional de la variable respuesta pertenezca a la familia exponencial, que engloba distribuciones discretas como la binomial o Poisson, entre otras, y distribuciones continuas, tales como la normal. Esto permite realizar una estimación por máxima verosimilitud de los parámetros del modelo. Para mayor información acerca de los modelos lineales generalizados y la estimación de los parámetros puede consultarse Müller (2012).

E.5. Gradient Boosting Machine

Las máquinas de potenciación del gradiente, comúnmente conocidas por *Gradient Boosting Machines* (GBM), son métodos predictivos de aprendizaje automático que pertenecen a los llamados modelos de ensamblaje, ya que están basados en la combinación de modelos predictivos débiles (como, por ejemplo, árboles de decisión) con la finalidad de obtener un modelo predictivo más eficiente (o lo que es lo mismo, más potente). Son métodos válidos tanto para problemas de regresión como para problemas de clasificación y su metodología es una extensión del llamado *Boosting*, que se describe a continuación.

Boosting

Boosting es una metodología de aprendizaje automático que combina diferentes modelos con poca capacidad predictiva con la finalidad de obtener un predictor mucho más potente y reducir la varianza.

Fue introducida a lo largo de estos últimos 20 años e inicialmente fue desarrollada para emplear en problemas de clasificación, aunque también se puede extender a problemas de regresión. Esta metodología puede implementarse a través de diferentes tipos de algoritmos. El más conocido de ellos es el algoritmo de *AdaBoost*, desarrollado por Freund y Schapire, que se explica, a continuación, de forma detallada para el caso de un problema de clasificación de dos clases, con la variable $Y \in \{1, -1\}$, indicando el éxito y el fracaso, respectivamente. Seguiremos como referencia el capítulo titulado *Boosting and Additive Trees* de Hastie et al. (2009).

Dado un vector de variables predictoras $(x_1, \cdots, x_N)$, se calculan las predicciones $G(x_i) \in \{-1, 1\}$, $i = 1, \cdots N$ y se obtiene la proporción de errores en la muestra de entrenamiento:

$$e_b = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(y_i \neq G(X_i))$$

El objetivo de la metodología *Boosting* es generar una secuencia de clasificadores débiles $G_1(x), \cdots, G_M(x)$ con la finalidad de combinarlos en un voto mayoritario con pesos para obtener las predicciones finales:

$$G(x) = \text{signo}\left(\sum_{b=1}^{B} s_b G_b(x)\right)$$

siendo signo la función que nos devuelve el signo del sumatorio. Si la suma es positiva (respectivamente negativa), la función devuelve predicción de la clase +1 (respectivamente -1). El efecto de los pesos s_b, $b = 1, \cdots B$ es dar mayor influencia a los clasificadores más precisos en la secuencia. Estos pesos se calculan con el algoritmo *AdaBoost*, que se describe a continuación de forma detallada.

Algoritmo E.2 (*AdaBoost*). [2] *Sea B el número de iteraciones y sean $w_i = 1/N$, $i = 1, \cdots, N$ los pesos iniciales en la muestra de entrenamiento.*

- *Para cada $b = 1, \cdots, B$:*

  (1) Se calcula la predicción $G_b(x)$ empleando los pesos iniciales.

  (2) Después se obtiene la proporción de errores en la muestra de entrenamiento, e_b y, con esta proporción, se calculan los pesos $s_b = \log((1 - e_b)/e_b)$.

  (3) Para finalizar se actualizan los pesos $w_i, i = 1, \cdots N$, considerando los pesos del instante anterior:

  $$w_i = w_{i-1} \exp((1 - s_b)/s_b)$$

  de forma que las observaciones clasificadas correctamente no se vean afectadas (es decir, no cambian sus pesos: $w_i = w_{i-1}$) y las clasificadas de forma incorrecta reciban mayor peso.

- *Se calculan las predicciones finales empleando un voto mayoritario: $G(x) = signo\left(\sum_{b=1}^{B} s_b G_b(x)\right)$.*

Se puede apreciar que, a diferencia de lo que ocurre en los modelos *Random Forest*, en el que se combinan diferentes tipos de árboles según las particiones aleatorias de los predictores de la muestra de entrenamiento, en el algoritmo *AdaBoost* para la metodología *Boosting* se emplea la muestra de entrenamiento al completo y se genera una secuencia con pesos en función de las predicciones obtenidas en el instante anterior, de forma que se asigna un peso elevado a observaciones mal clasificadas con la finalidad de obtener la clasificación correcta en instantes posteriores.

Los modelos *Gradient Boosting Machine*, introducidos por Friedman, son una extensión de la metodología *Boosting* que acabamos de definir para el caso particular de problemas de regresión. Esta

[2] Puede encontrarse en el capítulo titulado *Boosting and Additive Trees* de Hastie et al. (2009)

metodología tiene como principal objetivo encontrar un método aditivo que miminice una función de pérdida empleando métodos de descenso del gradiente. A continuación se describe, de forma detallada, el funcionamiento de este algoritmo siguiendo como referencia Bentéjac et al. (2021) y Touzani et al. (2018).

Sea $D = \{x_i, y_i\}_{i=1}^N$ el conjunto de datos de entrenamiento y sea $F^*(\boldsymbol{x})$ una función que asigna las variables predictoras x_i con sus correspondientes variables de salida y, $i = 1, \cdots, N$. Tal y como se ha comentado anteriormente, la idea bajo el método *Gradient Boosting Machine* es buscar una aproximación de la función $F^*(\boldsymbol{x})$ mininizando una función de pérdida $L(y, F(\boldsymbol{x}))$, por ejemplo, la suma residual de cuadrados. Para ello, se realiza un proceso iterativo en el que se construye una aproximación de la función como una suma de funciones con pesos ρ_b:

$$F_b(\boldsymbol{x}) = F_{b-1}(\boldsymbol{x}) + \nu\rho_b h_b(\boldsymbol{x}), \quad b = 1, \cdots, B$$

siendo $\nu \in (0, 1)$ el parámetro de regularización y $F_0(\boldsymbol{x}) = argmin_\alpha \sum_{i=1}^{N} L(y_i, \alpha)$ la predicción inicial constante (suele fijarse a 0). Los valores de los pesos ρ_b, $b = 1, \cdots B$, se calculan siguiendo un proceso *greedy* de descenso del gradiente y las funciones h_b son los correspondientes árboles de decisión ajustados sobre el conjunto de datos $D = \{x_i, r_{bi}\}_{i=1}^N$, siendo r_{bi} , $b = 1, \cdots B$, los residuos (gradientes) de cada uno de los ajustes realizados en las B iteraciones:

$$r_{bi} = \left[\frac{\partial L(y_i, F(\boldsymbol{x}))}{\partial F(\boldsymbol{x})}\right]_{F(\boldsymbol{x})=F_{b-1}(\boldsymbol{x})}, \quad b = 1, \cdots B$$

Con respecto al parámetro de regularización ν, comentar que sirve para controlar el proceso aditivo de la metodología *Gradient Boosting* y que, cuanto más pequeño sea el modelo,será más preciso y evitará problemas de sobreajuste de los datos (*overfitting*). Además, dado que ν es inversamente proporcional al número de iteraciones, un valor pequeño de ν conlleva a un número elevado de iteraciones con la finalizad de alcanzar convergencia. Suele fijarse, habitualmente, $\nu = 0.01$.

Algoritmo E.3 (*Gradient Boosting Machine*). [3] *Sea B el número de iteraciones (número de árboles de decisión), $\nu \in (0, 1)$ el parámetro de regularización y $d \geq 1$ la profundidad del árbol de decisión o número de cortes en cada árbol, los diferentes parámetros que intervienen en el modelo GBM, previamente fijados por el usuario. Sea $F_0(\boldsymbol{x}) = 0$ la predicción inicial (constante) y $r_{0i} = y_i$, para $i = 1, \cdots N$, los residuos iniciales de la muestra de entrenamiento.*

- *Para cada una de las iteraciones $b = 1, \cdots B$:*

  (1) Se realiza un ajuste de un árbol de decisión con d cortes, $h_b(\boldsymbol{x})$, sobre la muestra $\{x_i, r_{ib}\}_{i=1}^N$ y se calcula la versión regularizada del árbol $\nu h_b(\boldsymbol{x})$.

  (2) A partir del árbol de decisión obtenido en el anterior paso se calcula el valor del peso ρ_b siguiendo un proceso greedy de descenso del gradiente:

  $$\rho_b = argmin_\rho(L(F_{b-1}(\boldsymbol{x}) - \rho h_b(\boldsymbol{x})))$$

  (3) Una vez obtenida la versión regularizada del árbol y los pesos ρ_b, se actualiza el valor de la función $F_b(\boldsymbol{x})$, empleando la siguiente fórmula:

  $$F_b(\boldsymbol{x}) = F_{b-1}(\boldsymbol{x}) + \nu\rho_b h_b(\boldsymbol{x})$$

[3] En el capítulo *Boosting Methods* de Hastie et al. (2009) puede encontrarse una versión análoga de este algoritmo, que se ha redactado empleando Bentéjac et al. (2021) y Touzani et al. (2018).

(4) Finalmente, se actualizan los valores de los residuos:

$$r_{(b+1)i} = r_{bi} - \nu h_b(\boldsymbol{x}), \quad i = 1, \cdots N$$

Nótese que, en la actualización de los residuos, no interviene el valor del peso ρ_b, sólo interviene la versión regularizada del árbol de decisión.

- *Una vez terminado el proceso, se aproxima la función $F^*(\boldsymbol{x})$ mediante la suma de las funciones $F_b(\boldsymbol{x})$:*

$$F^*(\boldsymbol{x}) \cong \sum_{b=1}^{B} (F_{b-1}(\boldsymbol{x}) - \nu\rho_b h_b(\boldsymbol{x}))$$

E.6. Extreme gradient Boosting

Los modelos XGBoost (o *Extreme Gradient Boosting*) son métodos de aprendizaje automático pertenecientes a la familia de los métodos llamados de ensamblaje. Son una mejora de otros métodos como el *Gradient Boosting Machine*, que se acaba de definir, o el *Stocastic Gradient Boosting* (SGB). Fueron diseñados esencialmente para ser altamente escalables y su principal ventaja es que son métodos de aprendizaje rápidos computacionalmente ya que los modelos se ajustan de forma paralelizable.

Al igual que en los métodos *Gradient Boosting Machine*, los modelos XGBoost combinan métodos predictivos débiles (árboles de decisión) para obtener, de forma iterativa, métodos más eficientes. Para ello, se construye un modelo adictivo en base a una función objetivo que miminice una función de pérdida $L_{XGB}(y, F(x))$. Esta función, además de depender de un parámetro de regularización para evitar el sobreajuste (*overfitting*) debe incluír un término que controle la complejidad del modelo, es decir, la complejidad de los árboles de decisión:

$$L_{XGB}(y, F(x)) = \sum_{i=1}^{N} L(y_i, F(x_i)) + \sum_{b=1}^{B} \Omega(h_b(x))$$

siendo $\Omega(h(x)) = \gamma \cdot T + \frac{1}{2}\lambda||\omega||^2$. T es el número de árboles de decisión, $\gamma \in (0, 1)$ el parámetro de regularización y ω los scores (o puntaciones) de cada una de las hojas del árbol de decisión. Para mayor información acerca de este novedoso modelo puede consultarse Bentéjac et al. (2021) y Chen et al. (2016).

Bibliografía

Afdal A, Aabid A, Khan A, Khan SA, Rajak U, Verma TN, Kumar R (2020). Response surface analysis, clustering, and random forest regression of pressure in suddenly expanded high-speed aerodynamic flows. *Aerospace Science and Technology*, 107: 106318.

Bamaqa A, Sedky M, Bosakowski T, Bastaki BB (2020). Anomaly detection using hierarchical temporal memory (HTM) in crowd management. *International Conference on Cloud and Big Data Computing*: 37-42.

Bentéjac C, Csörgo A, Martínez-Muñoz G (2021). A comparative Analysis of XGBoost. *Artificial Inteligence Review*, 54: 1937-1967.

Berrar D (2019). Cross Validation.

Brozyna J, Mentel G, Szetela B, Strielkowski W (2018). MULTI-SEASONALITY IN THE TBATS MODEL USING DEMAND FOR ELECTRIC ENERGY AS A CASE STUDY. *Economic Computation & Economic Cybernetics Studies & Research*, 52.

Chen T, Guestrin C (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*: 785-794.

Chicco D, Warrens MJ, Jurman G (2021). The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *PeerJ Computer Science*, 7: e623. https://peerj.com/articles/cs-623/methods

Choi K, Yi J, Park C, Yoon S (2021). Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines. *IEEE Access*.

Dixon M (2022). Industrial Forecasting with Exponentially Smoothed Recurrent Neural Networks. *Technometrics*, 64: 114-124.

Falch TL, Elster AC (2016). An image processing language for performance portability on heterogeneous systems. *International Conference on High Performance Computing & Simulation (HPCS)*.

Fan J (1996). *Local Polynomial Modelling and Its Applications*. Chapman & Hall.

Fernández-Casal R, Costa-Bouzas J, Oviedo M (2021). Aprendizaje Estadístico. https://rubenfcasal.github.io/aprendizaje estadistico/

Fomby T (2008). The Unobservable Components Model.

Geiger A, Liu D, Alnegheimish S, Cuesta-Infante A, Veeramachaneni K (2020). TadGAN: Time series anomaly detection using generative advesarial networks. *2020 IEEE International Conference on Big Data (Big Data)*: 33-43

Hastie T, Tibshirani R, Friedman J (2009). *The Elements of Statistical Learning. Data Mining, Inference and Prediction*. Springer Series in Statistics.

He Y, Zhao J (2019). Temporal convoltional networks for anomaly detection in time series. *Journal of Physics: Conference Series*, 1213: 042050.

Hyndman RJ, Shang HL (2010). Rainbow plots, bagplots, and boxplots for functional data. *Journal of Computational and Graphical Statistics*, 19: 29-45.

Jalles JT (2009). Structural time series models and the kalman filter: a concise review.

Kim TY, Cho SB (2018). Web traffic anomaly detection using C-LSTM neural networks. *Expert Systems with Applications*, 106: 66-76.

Müller M (2012). Generalized Linear Models. *Handbook of Computational Statistics*: 681-709.

Pascanu R, Mikolov T, Bengio Y (2013). On the difficulty of training Recurrent Neural Networks. *International conference on machine learning*: 1310-1418.

Qu Z, Su L, Wang X, Zheng S, Song X, Song X (2018). A unsupervised learning method of anomaly detection using gru. *2018 IEEE International Conference on Big Data and Smart Compting (BigComp)*: 685-688.

Rafferty G (2021). *Forecasting Time Series Data with Facebook Prophet*, Packt Publishing Ltd.

Raja S, Fokoué E (2019). MULTI-STAGE FAULT WARNING FOR LARGE ELECTRIC GRIDS USING ANOMALY DETECTION AND MACHINE LEARNING. *Mathematics for Application*, 8: 115-130.

Sarker IH, Abushark Y, Khan AI, Alam Md M, Nowrozy R (2021). Mobile Deep Learning: Exploring Deep Neural Network for Predicting Context-Aware Smartphone Usage. *SN computer Science*, 2: 1-12.

Sharma S, Sharma S, Athaiya A (2017). Activation functions in neural networks. *towards data science*, 6: 310-316.

Shumway RH, Stoffer DS (2017). *Time Series Analysis and its applications with R examples*, Springer New York Dordrecht Heidelberg London.

Sokkhey P, Okazaki T (2020). Hybrid machine learning algorithms for predicting academic performance. *International Journal of Advanced Computer Science and Applications*, 11: 32-41.

Tharwat A (2020). Classification assessment methods. *Applied Computing and Informatics*.

Torres J (2020) *Deep Learning. Introducción práctica con Keras y TensorFlow2*. Marcombo.

Touzani S, Granderson J, Fernandes S (2018). Gradient boosting machine for modelling the energy consumption of commercial buildings. *Energy and buildings*, 158: 1533-1543.

Wen T, Keyes R (2019). Time series anomaly detection using convolutional neural networks and transfer learning.