Tipo de documento: Tesis de maestría

Master in Management + Analytics

Estimación de demanda de tickets en el contexto gastronómico mediante aprendizaje automático

Autoría: Horgan, Juan
Fecha de defensa de la tesis: 2023

UNIVERSIDAD TORCUATO DI TELLA

Master in Management + Analytics

Estimación de demanda de tickets en el contexto gastronómico mediante aprendizaje automático

TESIS

Juan Horgan

Mayo 2023

Tutor: Enzo Ferrante

Resumen

La gran competencia existente en el mercado de comida rápida obliga a los actores a ser cada día más eficientes en su operación para poder minimizar los costos y elevar la experiencia del cliente al máximo posible. Una gran ventaja competitiva a la hora de ser eficiente es contar con una predicción de la demanda eficaz. La estimación de demanda es comúnmente utilizada por múltiples áreas de la compañía para llevar adelante su operación. Equipos como marketing, operaciones, recursos humanos y varios más, utilizan este estimador a diario, como un componente central a la hora de plantear la estrategia a seguir. En la presente tesis, se trabaja sobre un local ubicado en Brasil, perteneciente a una de las cadenas de comidas rápidas más grande del mundo y se busca superar el predictor de demanda actual, mediante la utilización de modelos de aprendizaje automático. A partir del análisis comparativo realizado, fue posible determinar que es posible mejorar la calidad de las predicciones realizadas por el predictor de demanda actual, a través de el uso de modelos de aprendizaje automático como LightGBM o Gated Recurrent Unit, reduciendo tanto el error de las estimaciones como el tiempo necesario para realizar las mismas.

Abstract

The high level of competition in the fast food market forces players to become more efficient in their operations in order to minimize costs and enhance the customer experience as much as possible. A major competitive advantage in being efficient is effective demand forecasting. Demand forecasting is commonly used by multiple areas of the company to run its operation. Teams such as marketing, operations, human resources and several others, use this estimator on a daily basis, as a central component when planning the strategy to follow. In this thesis, we work on a store located in Brazil, belonging to one of the largest fast food chains in the world, and we seek to overcome the current demand predictor, through the use of machine learning models. From the comparative analysis carried out, it was possible to determine that it is possible to improve the quality of the predictions made by the current demand predictor, through the use of machine learning models such as LightGBM or Gated Recurrent Unit, reducing both the error of the estimates and the time required to make them.

1. Introducción

1.1. Contexto

Desde su aparición en la década del 90', internet aceleró y profundizó los cambios tecnológicos iniciados durante la tercera revolución industrial. Mientras que la conectividad alcanza a cada vez más personas en el mundo, y que los avances tecnológicos son cada vez más rápidos, se estima que el 90% de los datos que existen hoy, fueron generados en los últimos dos años. Las empresas no son ajenas a esto y buscan maneras de explotar y sacar provecho de los datos que hoy están disponibles.

Las herramientas necesarias para la explotación de los datos son cada vez más accesibles. La aparición de la *nube* permite a las personas y a las empresas acceder a muy potentes servidores (como los provistos por Amazon Web Services (AWS), Microsfot Azure o Google Cloud, entre otros) para el procesamiento de los datos, pagando un precio accesible solo por el uso, sin incurrir en los enormes costos de adquirir algo equivalente *on-premise*. Costos que incluyen no solo el hardware, sino también las licencias o los recursos humanos necesarios para mantenerlo funcionando. Esto hace que muchísimas empresas que no necesiten, o no puedan costear grandes departamentos de IT, puedan acceder a los beneficios de la explotación de datos. Según una nota publicada en la revista Forbes[1] en 2016, se estima que para el año 2025 la cantidad de datos disponibles a nivel mundial va a rondar los *180 zettabytes* (unos 180.000.000.000 *terabytes*).

El mundo de la gastronomía no es ajeno a los avances tecnológicos, cada vez más disruptivos. Por nombrar algunos ejemplos, el restaurante “Niño Gordo” realizó una prueba presentando su carta de comidas con realidad aumentada y la empresa McDonald's abrió su primer restaurante 100% digital en latinoamérica. A su vez, los sistemas predictivos basados en aprendizaje automático están siendo utilizados cada vez con mayor frecuencia en esta industria, para generar información que permita la toma de decisiones basada en datos en un contexto cambiante, donde adaptarse rápidamente a las nuevas tendencias o condiciones puede ser vital para una empresa.

El concepto de *Data-Driven Decision Making* (DDD) (Provost, et. al., 2019) hace referencia a tomar decisiones basadas en datos. Por ejemplo: una persona de *marketing* puede elegir qué artículo publicar basándose únicamente en su experiencia; o bien, puede basar su elección en datos que muestran cómo los clientes reaccionan a los distintos avisos.

[1] https://www.forbes.com/sites/michaelkanellos/2016/03/03/152000-smart-devices-every-minute-in-2025-idc-outlines-the-future-of-smart-things/?sh=409766864b63

Obviamente, también puede combinar ambos enfoques. Brynjolfsson et. al. (2011) muestran que las empresas que toman decisiones basadas en datos obtienen un aumento en la productividad de un 5-6%, y que esta forma de tomar decisiones está correlacionada con incrementos en el *return on equity* (ROE).

Con el objetivo de obtener el diferencial en los beneficios mencionado anteriormente, la demanda de perfiles orientados a *ciencia de datos* crece año a año, aún más que la oferta. En el año 2012, la revista Harvard Business Review[2] hablaba de los *científicos de datos* como el trabajo más *atractivo* del siglo XXI. Según un reporte publicado por LinkedIn[3] en el año 2020, la posición cuya demanda creció más desde el año anterior fue *especialista en inteligencia artificial* (74%). En tercer lugar figura *científico de datos* (37%) e *ingeniero de datos* (33%) en la octava posición. Por su parte, la consultora PwC[4] estima que para el año 2030, la *inteligencia artificial* generará un impacto de *15,7 billones* a la economía global; donde 6,6 vendrán de aumentos en la productividad y 9,1 por el lado del consumo. Esta tesis de maestría se enmarca en el contexto de la ciencia de datos aplicada a la gastronomía.

1.2. El problema

En este trabajo, se abordará el problema de predicción de ventas en el contexto de locales gastronómicos, por medio de modelos de aprendizaje automático. Obtener una predicción de ventas acertada constituye un dato de enorme importancia para empresas de todo tipo, ya que permite generar eficiencia en múltiples actores involucrados en los distintos procesos. Una predicción eficiente puede contribuir a optimizar el uso de las maquinarias, a optimizar el stock de los productos y a mejorar los márgenes en múltiples procesos (Doganis, et. al., 2006). Por nombrar algunos ejemplos:

- El departamento de finanzas puede proyectar costos, márgenes y capital con un mayor nivel de precisión.
- El departamento de marketing puede planear mejor las estrategias.
- El departamento de logística puede mantener el stock y el flujo optimizados.
- El departamento de recursos humanos puede contar con el stock óptimo de personal que permita cumplir con los objetivos sin tener recursos ociosos.

[2] https://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century
[3] https://business.linkedin.com/content/dam/me/business/en-us/talent-solutions/emerging-jobs-report/Emerging_Jobs_Report_U.S._FINAL.pdf
[4] https://www.pwc.com/gx/en/issues/data-and-analytics/publications/artificial-intelligence-study.html

En la industria alimenticia en particular, existe un agravante relacionado con la corta vida útil de los distintos alimentos. Esto implica que muchas veces no es posible tener un sobre stock de los mismos, por el riesgo de incurrir en pérdidas por desperdicios de alimentos vencidos que deban ser desechados. Sin embargo, quedarse sin stock genera no solo la pérdida de ventas afectando el *revenue*, sino también el daño generado a la experiencia del cliente insatisfecho y el riesgo de que este se vuelque a la competencia.

Sumado a esto, los hábitos de consumo de las personas pueden cambiar rápidamente por distintos factores. Estacionalidad, precios y promociones (propias y de competidores), el clima, fechas religiosas y cambios culturales, son algunos de los tantos factores que pueden afectar las ventas de productos alimenticios (Meulstee, et. al., 2008).

Phau, et. al. (2013) muestran que el tiempo de servicio resulta una de las variables más importantes para la satisfacción de los clientes en la industria de las comidas rápidas, junto con otras variables principalmente relacionadas con el trato de los empleados del local. Se estima que durante las horas pico, el tiempo de espera de los clientes en un local de comidas rápidas es de 5,4 minutos (Dharmawirya, et. al., 2012), que se dividen en 2,42 minutos de fila y 2,98 minutos de tiempo de servicio. Contar con un pronóstico de demanda acertado permite optimizar los flujos de procesamiento de los pedidos, manteniendo estándares de calidad óptimos, generando un impacto positivo en el tiempo de servicio y manteniendo una buena experiencia del cliente.

1.3. Objetivo

En este trabajo, nos centraremos en el caso de una cadena líder en restaurantes de comida rápida donde el interés radica en anticipar cuántos tickets venderá en los próximos dos meses en un local ubicado en Brasil. En particular, se necesita saber con dos meses de anticipación y con una granularidad de predicción cada quince minutos, cuántos tickets serán vendidos en ese local. Esto permitirá a los diversos equipos (como operaciones, marketing, recursos humanos, etc) realizar planificaciones con anterioridad. Conocer las ventas futuras con alto nivel de precisión permite generar optimizaciones diversas que afectan directa o indirectamente no sólo a los costos, sino también intangibles como la experiencia del cliente. Por dar algunos ejemplos:

- Contar con la cantidad óptima de recursos humanos en el establecimiento, de manera tal que las tareas sean realizadas en tiempo y forma, sin contar con recursos ociosos.

- Contar con preparaciones semi terminadas para hacer frente a los picos de demanda, sin generar cuellos de botella o retrasos en la entrega de los pedidos.
- Un mejor manejo de los productos con vencimientos cortos, de manera de reducir al mínimo los desperdicios.

El modelo será parte del pipeline productivo de *analytics* de la empresa, permitiendo a las distintas áreas obtener con dos meses de anticipación las predicciones que les posibilite desarrollar las diversas actividades con el mayor grado de precisión posible. A tal fin, aquí se explorarán diversas técnicas de predicción, abarcando estimadores simples (como un promedio), pasando por modelos tradicionales como *Random Forest*, hasta algoritmos de aprendizaje profundo (en particular, redes neuronales recurrentes como *long short-term memory* (LSTM) o *gated recurrent unit* (GRU)).

Si bien en esta tesis se abordará sólo el módulo de predicción, cabe destacar que la aplicación completa en la que dicho módulo será utilizado corresponde a un pipeline productivo, controlado por la tecnología Control-M, que permite automatizar flujos de trabajo. Dicho pipeline será ejecutado una vez al mes, realizando los procesos de extracción, transformación y carga de datos (ETL) que se encuentran dentro del Data Lake de la empresa. Una vez completada esta fase, los datos serán utilizados por el objeto modelo para realizar las predicciones correspondientes, que a su vez serán ingestadas en un tabla delta dentro el Data Lake para su explotación. Finalizada la etapa de predicción del modelo, el pipeline continúa en la etapa de medición, donde se calculará el error medio absoluto (MAE), y otras métricas a definir, sobre la predicción realizada el mes anterior y contrastando contra la realidad. El o los modelos que superen un cierto umbral de error, serán reportados en un mail automático a los responsables del área para su reentrenamiento. Las predicciones, los datos reales y los indicadores calculados serán ingestados en un tablero de Power BI para llevar un control de la salud de todos los modelos.

2. Datos

Se cuenta con una base de datos de ventas de tickets o GC (*guest count*) para un local comercial, que abarca desde 01-01-2019 hasta 29-06-2022. Dentro de la misma, se cuenta con las siguientes variables:

Tabla 1: Descripción de las variables

Variable	**Tipo**	**Descripción**
timestamp_inicio	Fecha (yyyy-mm-dd hh:mm:ss)	Fecha
mes	Entero	Mes de la observación
dia_mes	Entero	Día de la observación
dia_semana	Entero	Número de día de la semana[5] de la observación
hora	Entero	Hora de la observación
minuto	Entero	Minuto de la observación
cant_timestep_habil_prox	Entero	Cantidad de fracciones de 15 minutos restantes para el próximo día hábil[6]
cant_tickets	Entero	Cantidad de tickets vendidos en esa observación

La variable objetivo es *cant_tickets*. Esta variable es del tipo entero y representa la cantidad de tickets vendidos en esa *observación*, es decir: en una fracción de 15 minutos específica. El contenido del ticket es indiferente para el alcance de este modelo. Un ticket que contenga un café contará como uno, de la misma manera que lo hará uno que contenga múltiples combos. En síntesis: es indiferente el contenido de un ticket o su valor, y cada ticket vendido en una fracción de 15 minutos sumará uno a esa observación. El dataset tiene la siguiente forma:

[5] Se toma como primer día de la semana al día Domingo.
[6] Según el calendario de Brasil.

Tabla 2: Ejemplo del set de datos

timestamp_inicio	mes	dia_mes	dia_semana	hora	minuto	cant_timestep_habil_prox	cant_tickets
2019-01-01T11:00:00.000Z	1	1	3	11	0	51	0
2019-01-01T11:15:00.000Z	1	1	3	11	15	50	4
2019-01-01T11:30:00.000Z	1	1	3	11	30	49	2
2019-01-01T11:45:00.000Z	1	1	3	11	45	48	2
2019-01-01T12:00:00.000Z	1	1	3	12	0	47	4
2019-01-01T12:15:00.000Z	1	1	3	12	15	46	1
2019-01-01T12:30:00.000Z	1	1	3	12	30	45	12
2019-01-01T12:45:00.000Z	1	1	3	12	45	44	10

En las figuras 1 y 2 observamos las ventas de tickets por cuarto de hora para los primeros dos meses (Figura 1) y los primeros diez días del 2019 (Figura 2). En esta última también notamos que existen horarios sin ventas. Esto se debe a que el horario en el que esta sucursal opera es de 10 a 23 horas, de lunes a lunes. Nótese también en el círculo rojo que existe un número positivo de tickets en un horario no operativo (en este ejemplo particular, un solo ticket). Esto se debe a que a veces los sistemas transaccionales fallan por razones varias, como puede ser cortes de luz, saturación del sistema u otros, y realizan las cargas que quedaron en cola en horarios incorrectos. Teniendo en cuenta esto, realizamos un histograma de tickets vendidos por hora (Figura 3). Lo primero que se observa en esta figura es que la mayor cantidad de tickets vendidos corresponden a las horas de entre las 18 y las 20. La cantidad de tickets vendidos alrededor de las 22 horas es similar a la cantidad vendida durante horas de la tarde temprana. No se observa un pico de ventas durante el mediodía, pero sí se observa una cantidad significativa de tickets vendidos durante las 23 y las 0 horas. Para ver esto con más detalle, realizamos un nuevo histograma de ventas entre las 23 y las 00:45 horas (Figura 4), donde podemos observar que la cantidad de tickets decrece a medida que aumenta la franja horaria. Esto puede deberse a múltiples razones: en primer lugar, existe la posibilidad de que en el pasado, este local haya operado más tarde de las 23 horas. En segundo lugar, aquellas personas que se encuentren dentro del local una vez pasada la hora de cierre, es decir, más allá de las 23 horas, deben ser atendidas.

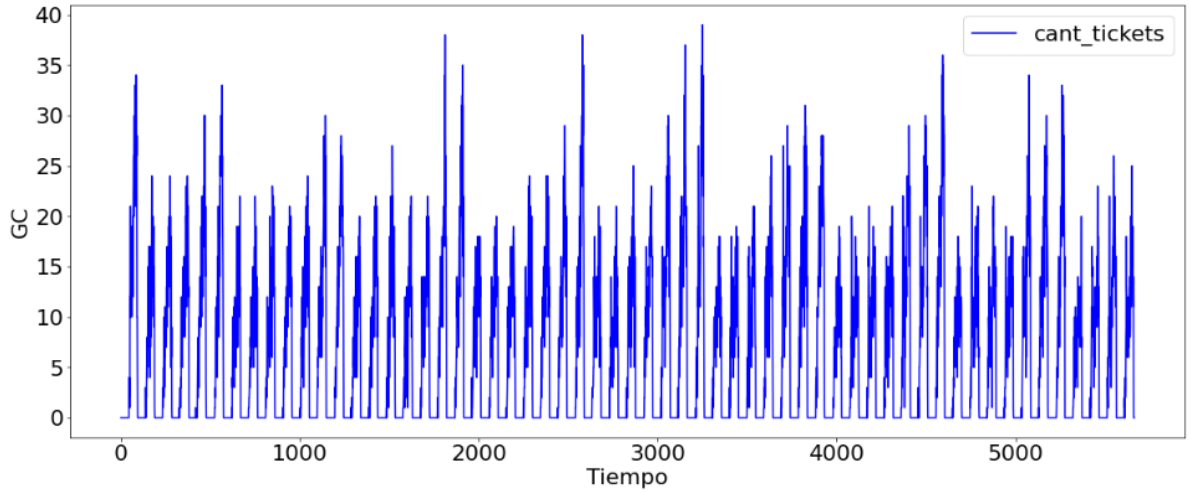

Figura 1: Cantidad de tickets facturados por cuarto de hora del 2019-01-01 al 2019-03-01.

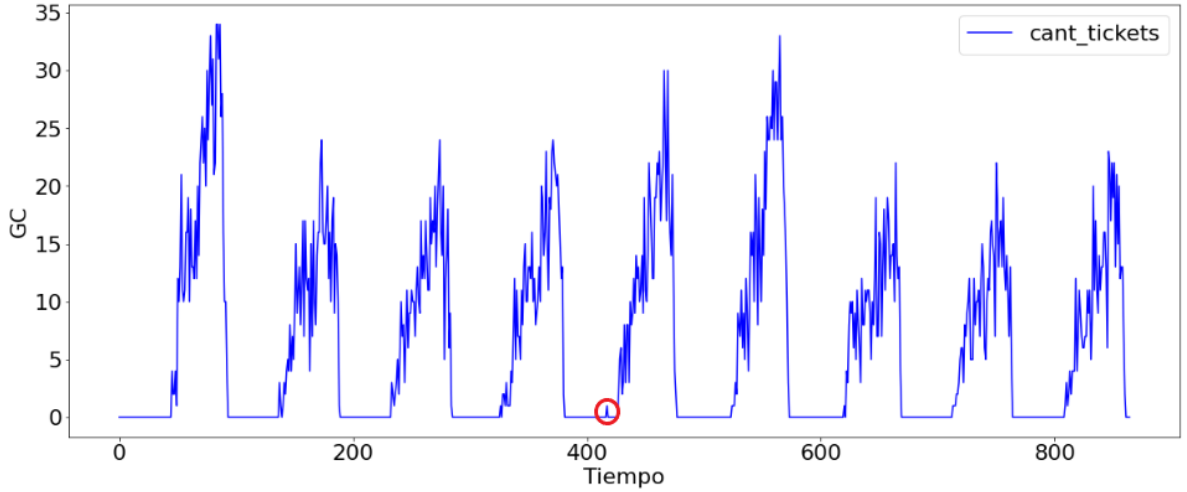

Figura 2: Cantidad de tickets facturados por cuarto de hora del 2019-01-01 al 2019-01-10.

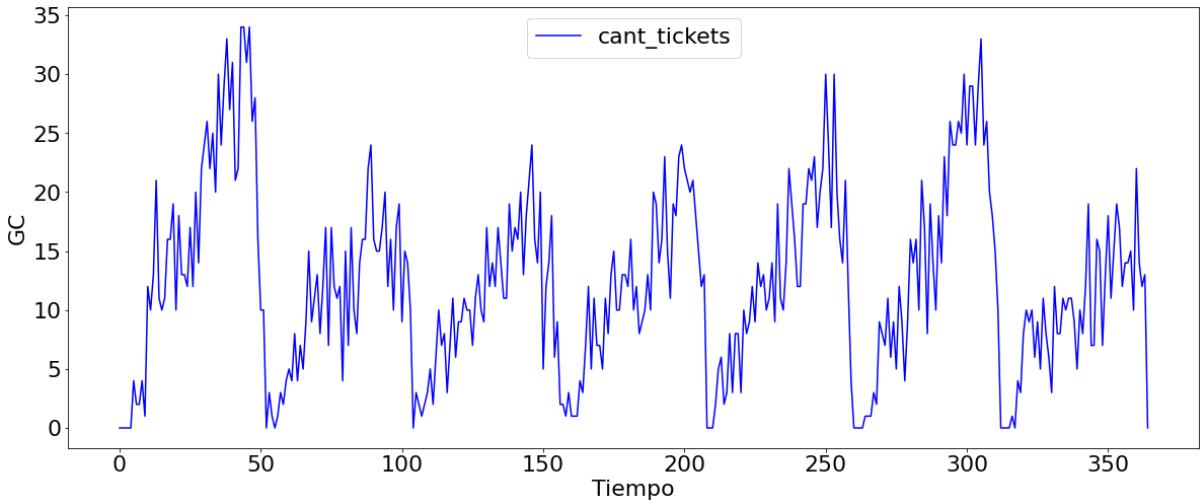

Figura 3: Cantidad de tickets facturados por cuarto de hora del 2019-01-01 al 2019-01-08 en horarios operativos.

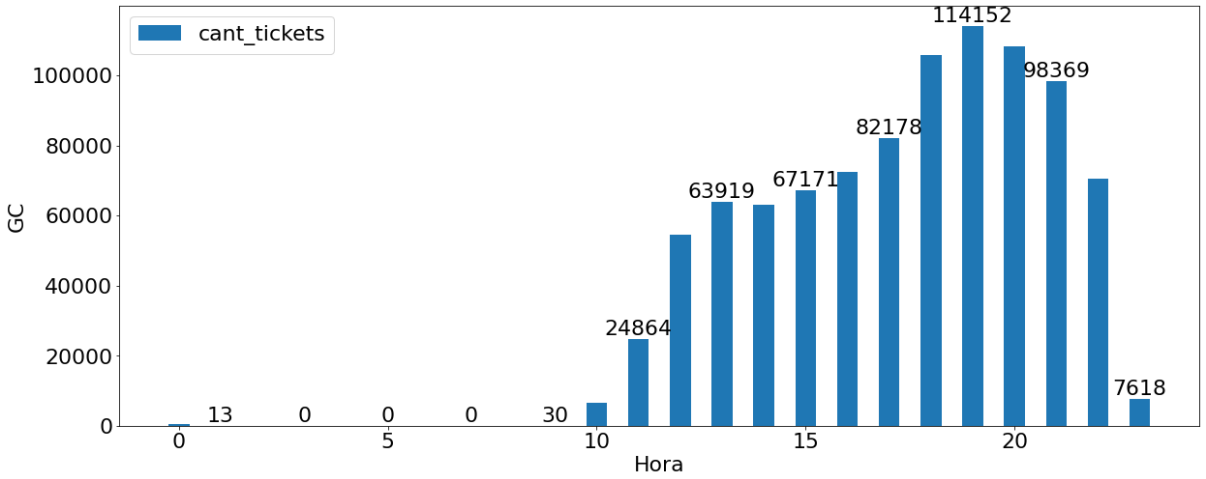

Figura 4: Histograma de tickets facturados por hora.

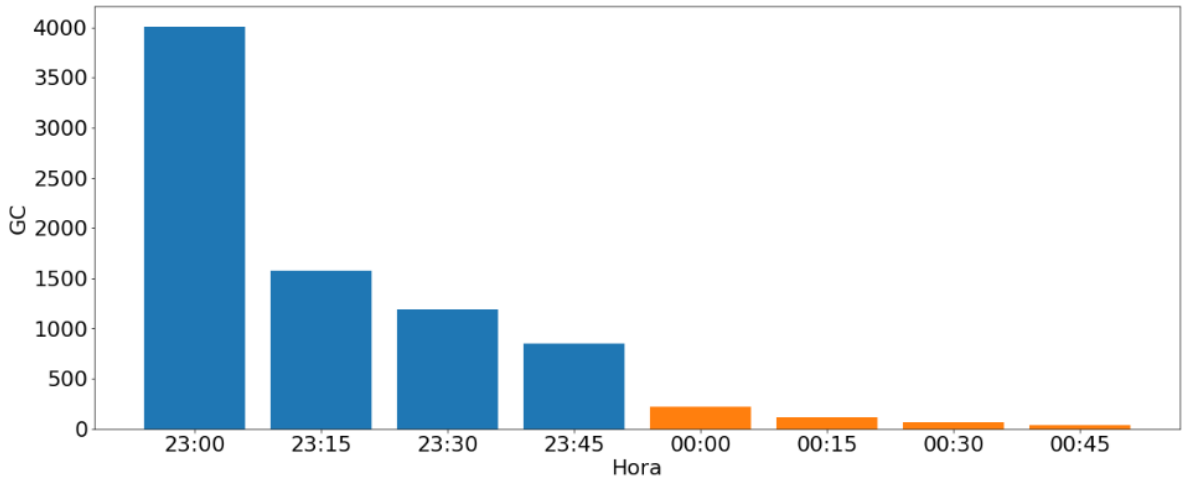

Figura 5: Histograma de tickets facturados por cuarto de hora entre las 23:00hs y las 00:45hs.

En el set de datos no se encuentran otras variables exógenas comúnmente utilizadas para este tipo de problemas, como pueden ser variables de marketing o el clima. Esto se debe a varios motivos. Respecto a la variable clima, la naturaleza propia del problema sobre el cual trata este trabajo requiere una predicción de ventas de los próximos dos meses, lo que hace imposible tener una predicción climatológica con cierto grado de precisión en un horizonte temporal tan grande. Sin embargo, podríamos esperar que ciertas cuestiones estacionales, cómo pueden ser épocas de lluvias o una baja temperatura, generen mermas en las ventas. Como muestran Bujisic et. al. (2016), el clima impacta no solo en las ventas agregadas, sino que también provoca cambios en la demanda de ítems específicos del menú. La variación estacional tiene una gran correlación con la época del año, por ende es razonable pensar que

dicho efecto se encuentra como variable latente dentro de la variable mes. Es por ello que no se incorpora.

Respecto a las campañas de marketing, la empresa tiene todo el tiempo descuentos activos en forma de cupones. En general, estos cupones suponen un descuento sobre el precio total a pagar cuando se combinan varios ítems o combos y van cambiando continuamente. Con lo cual, la matriz de cupones se vuelve muy extensa si se quisieran incorporar como variables explicativas. El problema de esta variable radica en que las combinaciones muchas veces incluyen ítems que se descontinúan o nuevos productos que salen al mercado. Con lo cual, sería imposible incorporar a un modelo productivo una nueva variable que incluya un ítem nuevo en el mercado. Si se añadiera una variable binaria *marketing general*, que tome valor 1 cuando hay una campaña activa, la misma se encontraría "prendida" todo el tiempo ya que, como explicamos anteriormente, siempre hay cupones o descuentos activos de distintos productos.

2.1. Pre-procesamiento y extracción de características

Los modelos de aprendizaje automático son entrenados por medio de bases de datos donde, en general, cada muestra se encuentra descripta por medio de un conjunto de características. Si pensamos en una serie de tiempo de venta de tickets de comida, es lógico pensar que determinados días, determinados horarios o combinación de estos tendrán efectos distintos en las ventas. Por ejemplo, si pensamos en un local ubicado en una zona de oficinas, podríamos esperar una mayor cantidad de ventas en el horario del mediodía, durante los días hábiles. Esta información le permite al modelo entender mejor el contexto y por ende, generalizar de una manera más eficaz. Por ello, se generaron 8 nuevas características que serán usadas como entrada para algunos modelos. En particular, se generaron cuatro medias móviles exponenciales y cuatro variables rezagadas sobre la variable objetivo:

Tabla 3: Descripción de las variables generadas

Variable	**Tipo**	**Descripción**
cant_tickets_1m	Entero	Cantidad de tickets vendidos en ese día-horario del mes anterior.
cant_tickets_2m	Entero	Cantidad de tickets vendidos en ese día-horario, 2 meses atrás.
cant_tickets_3m	Entero	Cantidad de tickets vendidos en ese día-horario, 3 meses atrás.
cant_tickets_6m	Entero	Cantidad de tickets vendidos en ese día-horario, 6 meses atrás.
ema_4	Decimal	Media móvil exponencial de 4 horas
ema_12	Decimal	Media móvil exponencial de 12 horas
ema_18	Decimal	Media móvil exponencial de 18 horas
ema_24	Decimal	Media móvil exponencial de 24 horas

Estas variables serán utilizadas en los modelos de Random Forest y XGBoost (ver Sección 3: Random Forest y Extreme Gradient Boosting Machine).

3. Metodología

Se evaluarán los distintos modelos abordados siguiendo una metodología de particionado de datos en entrenamiento, validación y prueba. Los datos de entrenamiento y validación serán utilizados durante el proceso de desarrollo, y los de prueba serán sólo utilizados para reportar resultados en la etapa final, con el objetivo de evitar sobreajuste a los datos de prueba. En concreto, el período de entrenamiento irá desde el 01/01/2019 hasta el 28/02/2022 inclusive. El período de validación irá desde el 01/03/2022 hasta el 31/04/2022. Finalmente, el período de prueba irá desde el 01/05/2022 hasta el 29/06/2022:

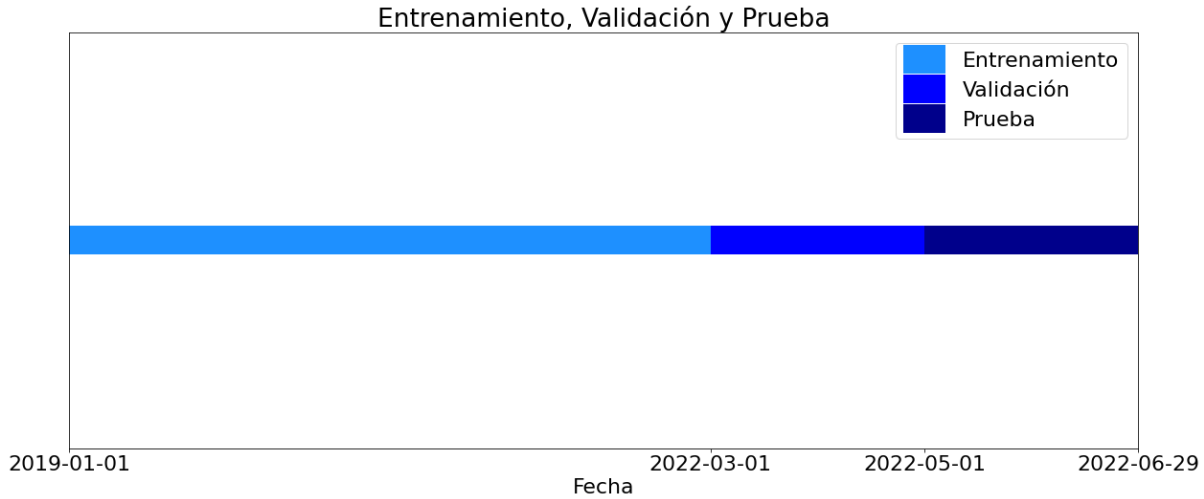

Figura 6: Períodos de entrenamiento, validación y prueba.

Se tomarán sólo los datos comprendidos entre las 10:00:00 y las 22:45:00 inclusive. Esto se debe a que el horario operativo de la sucursal en cuestión es de lunes a lunes entre las 10am y las 11pm. Las predicciones fuera del horario operativo no son relevantes, ya que se reportará como cero tickets.

Utilizaremos el error medio absoluto (MAE) de la cantidad de tickets como métrica de performance, considerando todo el período de prueba. El error medio absoluto se calcula de la siguiente forma:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Donde n es la cantidad de registros del período de prueba, y_i es el valor real del momento i e $\hat{y}_i$ es el valor predicho para el momento i. Para la búsqueda de hiper parámetros de los modelos de utilizará la técnica de *random search* para reducir así el subespacio de búsqueda, finalizando con una búsqueda manual dentro del mejor subespacio encontrado.

Adicionalmente, se considerarán los tiempos de entrenamiento de cada modelo. Esto se debe a que la generación de predicciones por parte de este local en particular es un producto viable mínimo (MVP) de un proyecto mucho más grande, que incluye cientos de locales. Por lo cual, los tiempos de entrenamiento de los modelos resultarán importantes para el escalado de la aplicación en el mundo real.

También evaluaremos la robustez del modelo sobre otros restaurantes. Dado que los restaurantes tienen distintos niveles de ventas, los mismos no pueden ser comparados entre ellos usando el error medio absoluto. En el campo del aprendizaje automático, es común

hacer uso de una métrica llamada error medio absoluto porcentual (MAPE, por sus siglas en inglés):

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

Donde n representa la cantidad de observaciones, y_i el valor real e $\hat{y}_i$ el valor predicho. El problema de este estimador es que si el valor real o el valor predicho son cero o cercanos a cero, el resultado tiende rápidamente a infinito. Por ello, el estimador elegido para comparar entre restaurantes es el error porcentual absoluto medio simétrico:

$$SMAPE = \frac{100\%}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{(|y_i| + |\hat{y}_i|) / 2}$$

Donde n representa la cantidad de observaciones, y_i el valor real e $\hat{y}_i$ el valor predicho y el estadístico está definido entre 0 y 2. Observando la distribución del valor del SMAPE para los distintos restaurantes, encontramos que la misma resulta no paramétrica. Por ello, podemos calcular el rango intercuartílico para encontrar outliers. El rango intercuartílico se calcula como:

$$RIQ = \frac{(Q_3 - Q_1)}{2}$$

Donde Q_3 representa el valor del cuartil 75 y Q_1 representa el valor del cuartil 25. Los outliers son aquellos valores menores que $Q_1 - RIQ$ y mayores que $Q_3 + RIQ$.

En las siguientes sub-secciones (3.1 a 3.5), se describirán los modelos comparados a lo largo de este trabajo. Los resultados obtenidos por cada modelo serán posteriormente presentados en la Sección 4.

3.1. Promedio simple

La primera estimación que realizaremos será un promedio simple, donde el valor predicho del momento *t*, depende del promedio de la cantidad de tickets de las últimas 4 semanas para ese día de la semana y esa hora particular:

$$\hat{y}_t = \frac{y_{t-7} + y_{t-14} + y_{t-21} + y_{t-28}}{4}$$

La elección de que sean 4 semanas se debe a que el negocio efectivamente estaba usando esta cantidad de semanas para estimar la predicción y, por lo tanto, este era la "línea

de base" (o *baseline* en inglés) a ser superada. Esta estimación nos servirá como base, ya que los siguientes modelos que abordaremos deberían ser, como mínimo, mejores que el promedio simple. Para evitar la fuga de información, a medida que vayamos avanzando en las predicciones del período de validación, las mismas se irán calculando sobre los valores predichos. Por ejemplo: el valor del día 01/03/2022 10:00:00 será el promedio de la cantidad de tickets efectivamente vendidos en los días 22/02/2022 10:00:00, 15/02/2022 10:00:00 , 08/02/2022 10:00:00 y 01/02/2022 10:00:00. Sin embargo, la cantidad estimada de tickets para el día 08/03/2022 10:00:00 será el promedio de las ventas reales de tickets para los días 22/02/2022 10:00:00, 15/02/2022 10:00:00 y 8/02/2022 10:00:00, más la estimación que realizamos para el día 01/03/2022 10:00:00.

3.2. Random Forest

Un modelo Random Forest es un conjunto o ensamble de múltiples árboles de decisión, que se entrenan mediante una técnica llamada *bagging* (Breiman, L, 2001). Un árbol de decisión, en el contexto de aprendizaje automático, es un modelo predictivo que se representa como un grafo compuesto de un nodo raíz, nodos intermedios y "hojas" o nodos terminales. Cada nodo intermedio se representa como una prueba de atributo, mientras que las hojas representan una probabilidad, asociada a una clase o valor (Rokach, L., et. al., 2015). Para ejemplificar cómo funciona un árbol de decisión, generamos un árbol simple entrenado sobre el dataset de Titanic[7], que contiene información sobre los pasajeros a bordo de dicho barco y si sobrevivieron al accidente o no. Podemos tomar un individuo al azar y recorrer el árbol de arriba hacia abajo como lo indica la Figura 6.

Si el individuo seleccionado fuese mujer, observamos que la probabilidad de haber sobrevivido (según este modelo muy simple) es levemente mayor que la de no haberlo hecho. Sin embargo, podemos seguir recorriendo hacia abajo para obtener una probabilidad más detallada. Como el individuo es mujer, nos movemos hacia la derecha y nos encontramos en un nodo sobre la clase en la que viaja la pasajera. Asumiremos que viajaba en primera o segunda clase, por lo que nos movemos por la rama de la izquierda. Al llegar a este nodo, notamos que la impureza de Gini es baja para la clase 0 (no haber sobrevivido). La impureza de Gini en un nodo es una medida que nos indica la probabilidad de que una muestra elegida aleatoriamente sea incorrectamente etiquetada. En este caso, podemos observar que la

[7] https://www.kaggle.com/datasets/heptapod/titanic

probabilidad de que una mujer con esas características elegida aleatoriamente no haya sobrevivido, es de 0.074. Podemos obtener más precisión contestando la pregunta de si la pasajera es mayor o menor de 27.5 años. En caso de que sí sea mayor, la probabilidad de haber sobrevivido es alta (de hecho, aproximadamente el 95% de las pasajeras que cumplen con estos criterios, efectivamente sobrevivieron).

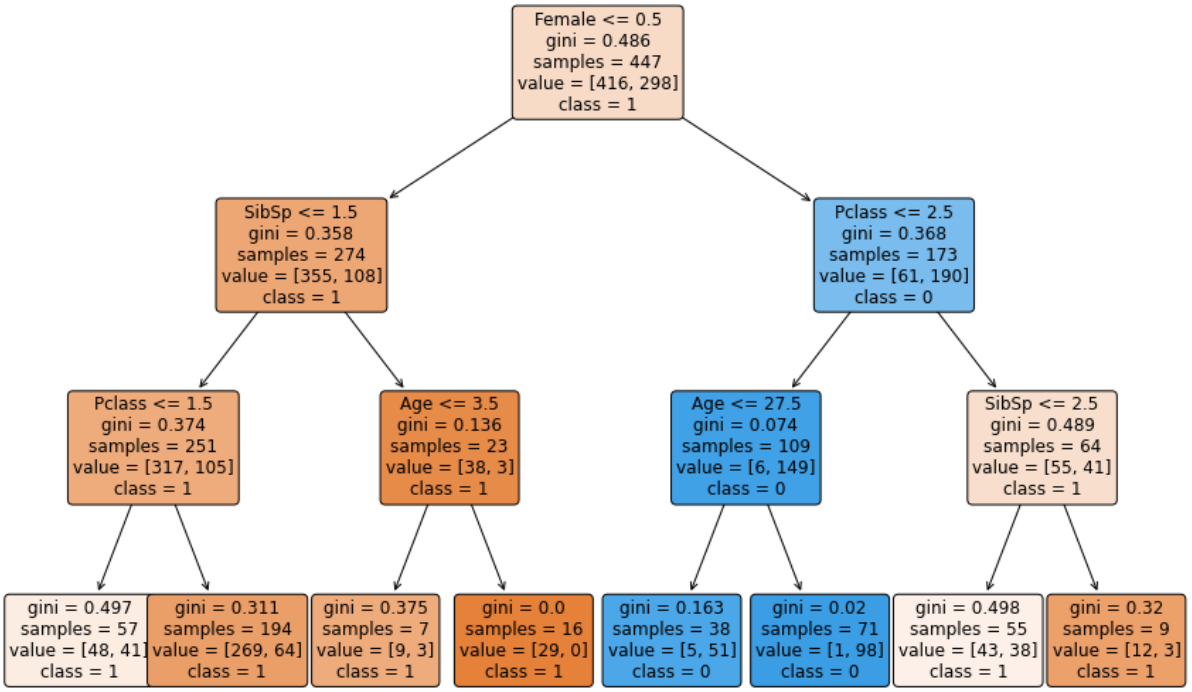

Figura 7: Árbol de decisión simple entrenado sobre el dataset de Titanic. Elaboración propia.

El método de bagging consiste en que cada uno de los árboles que integran el Random Forest es entrenado sobre un sampleo aleatorio (con reposición) de las variables del problema, de manera tal que ningún árbol puede ser entrenado sobre el set completo de datos. En este sentido, cada árbol es entrenado sobre un subespacio dentro del espacio de los datos (Figura 7). Finalmente, se toma el promedio de las predicciones de cada árbol como la predicción del modelo. El objetivo de esta técnica es evitar que el modelo realice un sobre ajuste de los datos, para obtener un mejor nivel de generalización. Por este motivo, los algoritmos de *bagging* suelen tener bajo sesgo, pero alta varianza.

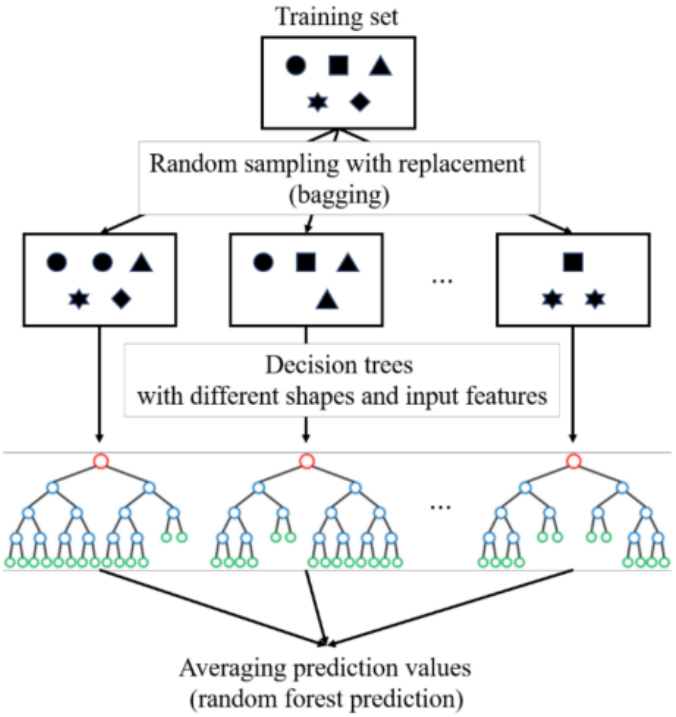

Figura 8: Esquema de funcionamiento del algoritmo de Random Forest donde cada árbol se entrena sobre un subconjunto de los datos y luego de promedian las predicciones (Lee, et. al., 2021).

El proceso de entrenamiento de Random Forest implica la creación de múltiples árboles de decisión independientes, que se entrenan sobre un subconjunto aleatorio de observaciones y variables. La importancia de cada variable puede calcularse como la disminución media de la impureza que se logra cuando se utiliza esa variable particular en diferentes nodos. Cuanto mayor sea la reducción media de la impureza, mayor será la importancia de esa variable. En casos donde los sets de datos contienen muchas características, suelen utilizarse algoritmos de árboles como una de las técnicas de selección de características, previo a generar un modelo. Las características generadas resultaron, en su mayoría, sumamente importantes para el modelo (Figura 8):

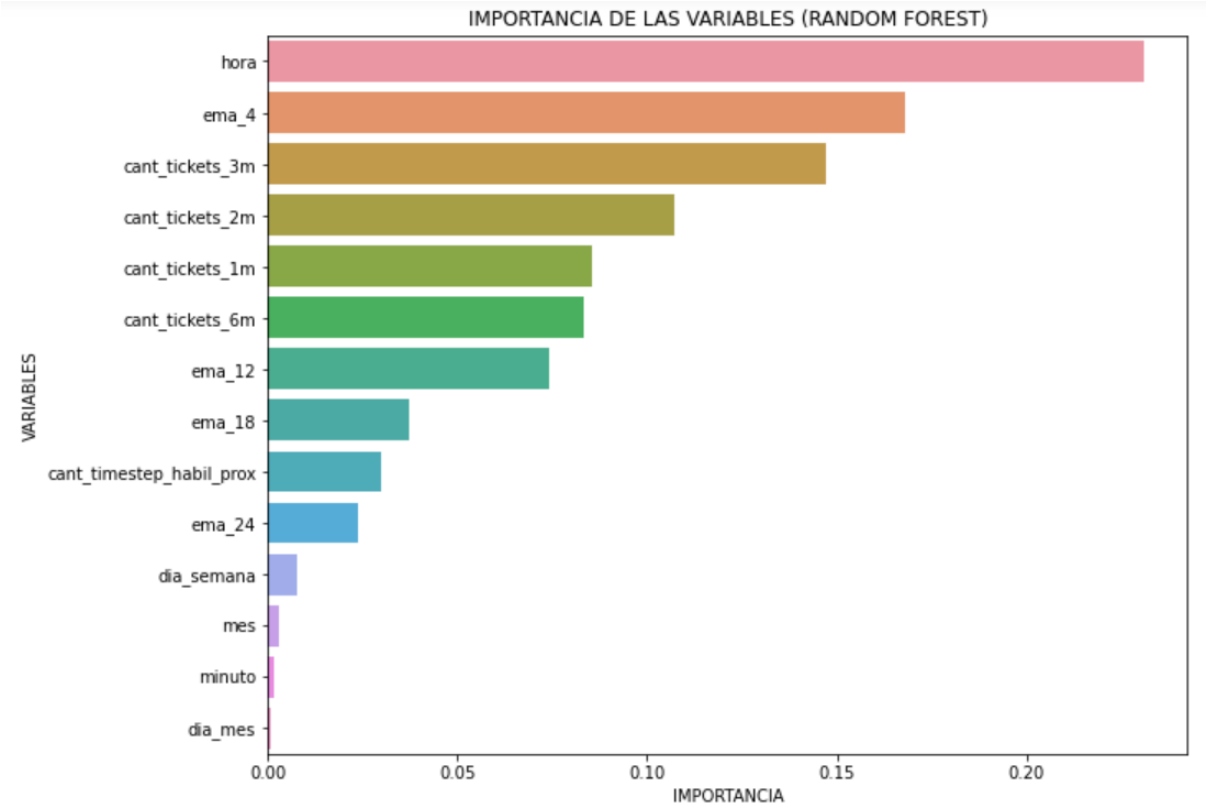

Figura 9: Importancia de las características según el modelo de Random Forest. Elaboración propia.

Las características extraídas a partir de los datos iniciales (ver Sección 2.1: Pre-procesamiento y extracción de características) fueron utilizadas tanto en Random Forest como en XGBoost (Sección 3.3). A la hora de realizar predicciones, esas variables deben ser recalculadas luego de cada predicción realizada para incluir el valor de la predicción y ser utilizadas como input en el próximo paso, lo que genera altas demoras en la generación de las predicciones.

3.3. Extreme Gradient Boosting (XGBoost)

XGBoost es un algoritmo de aprendizaje automático presentado en 2016 por T. Chen y C. Guestrin (Chen et. al., 2016). Utiliza la técnica de *gradient boosting* en árboles para resolver problemas de clasificación y regresión, y tiene amplia adopción en el mundo del aprendizaje automático.

A diferencia del método de bagging explicado anteriormente; boosting es un método que construye una secuencia de árboles, donde cada árbol corrige iterativamente los errores cometidos por el árbol anterior, buscando así generar una mejor precisión del sistema en general.

En particular, boosting combina múltiples clasificadores débiles en un único modelo robusto. Los clasificadores débiles tienden al sobre ajuste y son malos para clasificar datos que difieren mucho del conjunto de datos original. En vez de entrenar cada árbol por separado, XGBoost los entrena secuencialmente. Cada árbol trata de corregir los errores del árbol anterior asignándole una mayor importancia (Figura 9). En este sentido, si estuviéramos entrenando un clasificador de gatos, el primer árbol podría ser bueno entendiendo orejas, el segundo ojos, el tercero bigotes, etc. Luego, la predicción de cada uno de los árboles se promedian para obtener la predicción finales.

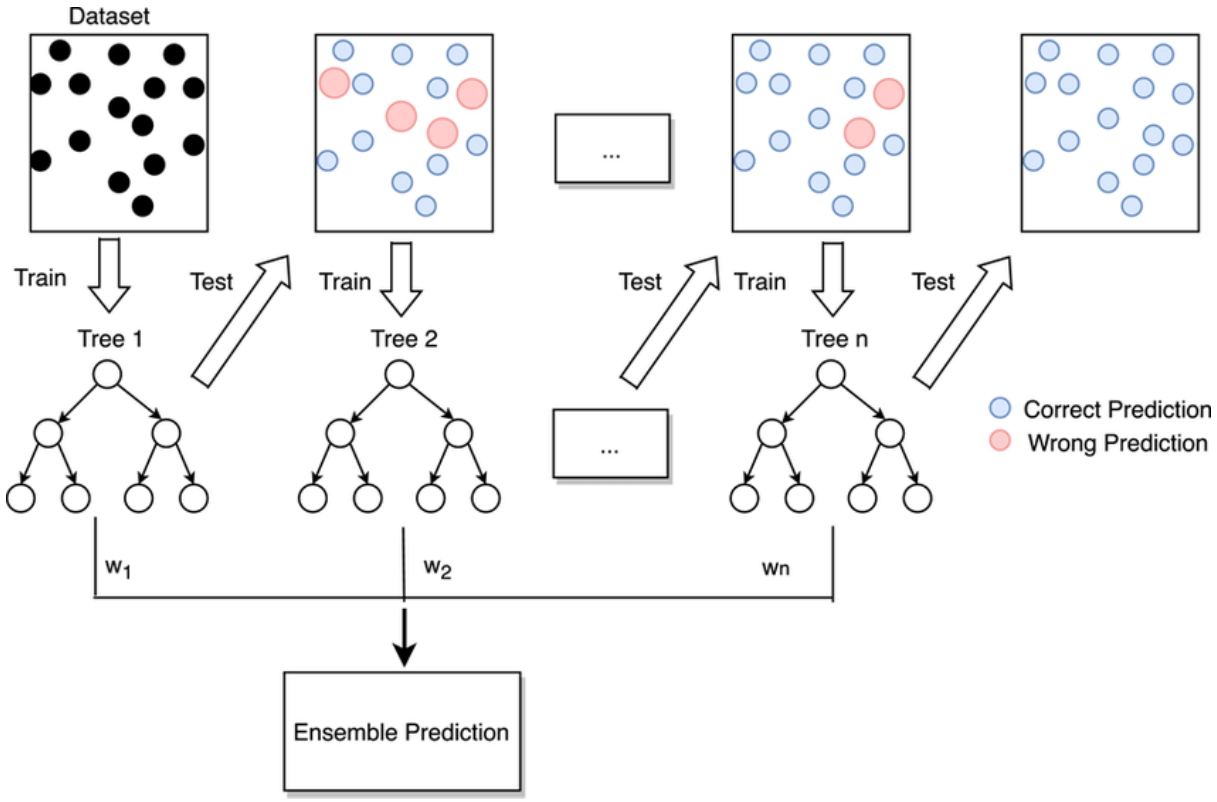

Figura 10: Ejemplo de la secuencia de entrenamiento de un modelo mediante el método de Boosting (Zhang, et. al., 2021)

3.4. Light Gradient Boosting Machine (LightGBM)

LightGBM es un algoritmo desarrollado por Microsoft en el 2017 (Guolin Ke, et. al., 2017). Posee similares características a XGBoost, ya que también utiliza el método de boosting para su entrenamiento mediante árboles de decisión. La principal diferencia con este radica en la manera en la que los árboles de decisión crecen. Mientras que en XGBoost, Random Forest y otros árboles de decisión, el crecimiento es por nivel (Figura 10); LightGBM realiza el crecimiento a nivel de hoja (Figura 11). Este tipo de enfoque, aunque

puede tender al sobreajuste, permite que LightGBM sea generalmente más rápido y preciso que XGBoost.

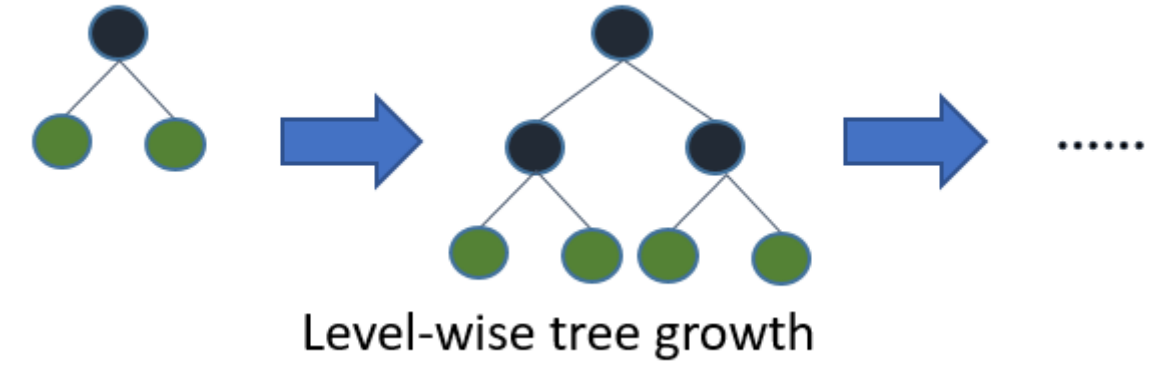

Figura 11: Crecimiento por nivel en árboles de decisión como XGBoost (fuente: https://lightgbm.readthedocs.io/en/latest/Features.html)

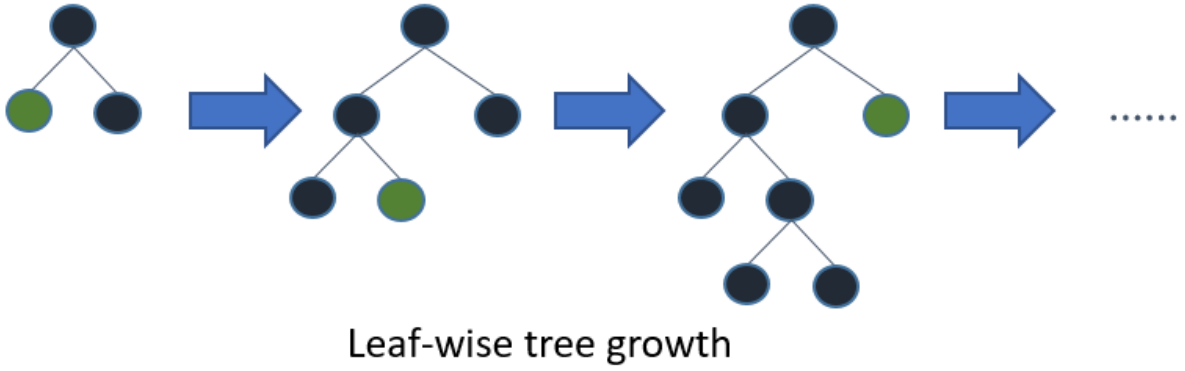

Figura 12: Crecimiento por hoja en LightXGB (fuente: https://lightgbm.readthedocs.io/en/latest/Features.html)

Además de lo mencionado, ambos tienen dos diferencias estructurales. La primera radica en cómo se realizan los cortes. Mientras que XGBoost trabaja con histogramas, LightGBM utiliza una técnica llamada Gradient Based One Side Sampling (GOSS). Esta técnica trabaja sobre los gradientes de las variables, donde un gradiente con un valor grande es entendido como una variable donde existe una importante posibilidad de ganancia en términos de información para el modelo.

La segunda diferencia se encuentra en que LightGBM trabaja internamente con una técnica llamada Exclusive Fracture Bundling (EFB), que permite reducir la dimensionalidad del problema. Parte de la premisa de que, en un espacio altamente ralo (o *sparse* en inglés), posiblemente existan variables que no toman valor cero al mismo tiempo (son mutuamente excluyentes). Una vez que se encuentra un par de esas variables, se realiza una transformación a una de ellas (por ejemplo, sumándole una constante) para cambiar el rango

y luego se las fusiona. Supongamos un problema donde se le pregunta a un grupo de personas si viven en casa o departamento (y supongamos también, que no existe otra opción que esa). A la variable casa podríamos sumarle una constante, por ejemplo 1, y combinar ambas variables para reducir la dimensionalidad del problema:

Tabla 4: Ejemplo de funcionamiento de Exclusive Fracture Bundling

Casa	Departamento	Variable EFB
1	0	2 (1 + 1)
0	1	1
0	1	1
1	0	2 (1 + 1)

3.5. Long Short Term Memory (LSTM)

Las redes neuronales artificiales fueron descriptas por primera vez en 1943 por Warren McCulloch y Walter Pitts. En su publicación llamada “Un cálculo lógico de las ideas inmanentes en la actividad nerviosa”, los autores propusieron un modelo matemático de una neurona, y mostraron cómo estas redes de neuronas artificiales son capaces de computar funciones lógicas. Actualmente existen muchos tipos de redes neuronales distintas, siendo algunas de las más conocidas las redes neuronales profundas, redes neuronales convolucionales, redes neuronales recurrentes, redes neuronales adversarias, etc. Los avances en hardware (en particular, el uso de procesadores gráficos) y la gran cantidad de datos disponibles hicieron que el área de aprendizaje profundo se hiciera muy popular sobre la década del 2010, generando enormes descubrimientos y mejoras, cuyos resultados son ampliamente conocidos por todos al día de hoy.

Las redes neuronales recurrentes (RNN) fueron descriptas en la década de 1980 y se utilizan comúnmente para el procesamiento de secuencias de datos, como ejemplo series de tiempo, audio o texto. La arquitectura de una red neuronal recurrente es la siguiente:

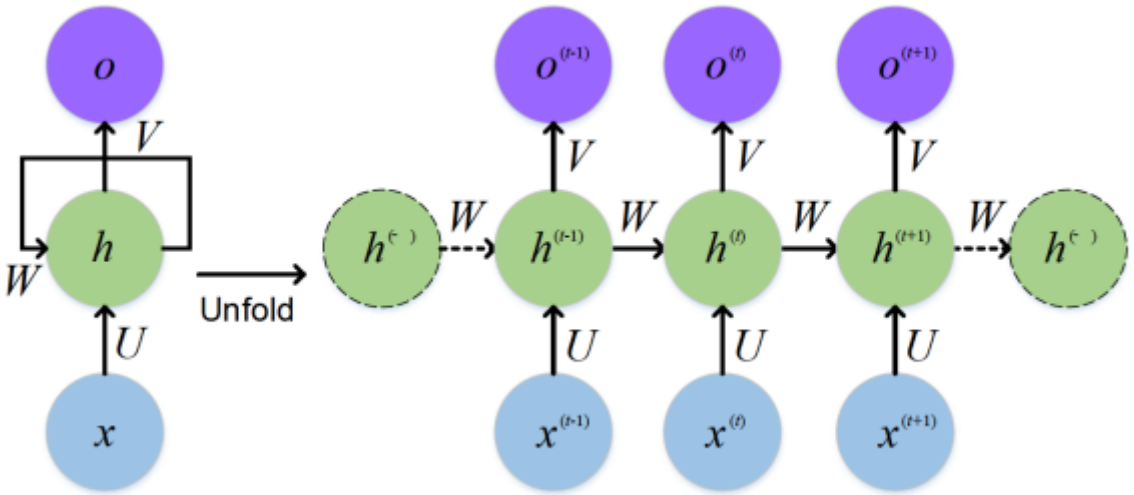

Figura 13: Arquitectura de una RNN (Weijiang, et. al., 2017)

Donde *x* representa la capa de entrada, *h* la capa oculta y *o* la capa de salida; y *U*, *W* y *V* representan matrices de pesos. La célula de una red neuronal recurrente tiene la particularidad de poseer ciclos en su capa oculta y puede ser desdoblada gráficamente como se observa a la derecha de la figura, donde también se observa cómo el estado de la capa oculta del momento *t-1* también es un input de la capa oculta del momento *t*. Este tipo de redes neuronales recurrentes posee un problema a la hora de realizar la optimización que hace que el gradiente explote o desaparezca cuando las secuencias son relativamente largas. Para solucionar el problema del gradiente explosivo, se utiliza la técnica de *clip value*, que consiste en acotar el valor máximo que puede tomar el gradiente. Para solucionar el problema de desvanecimiento del gradiente, surgieron un tipo de redes neuronales recurrentes llamadas Long Short Term Memory (LSTM).

Las LSTM fueron introducidas por Sepp Hochreiter y Jürgen Schmidhuber en 1997 (Hochreiter et. al., 1997). La arquitectura de este tipo de células le permite tener memoria tanto de corto, como de largo plazo y es la siguiente:

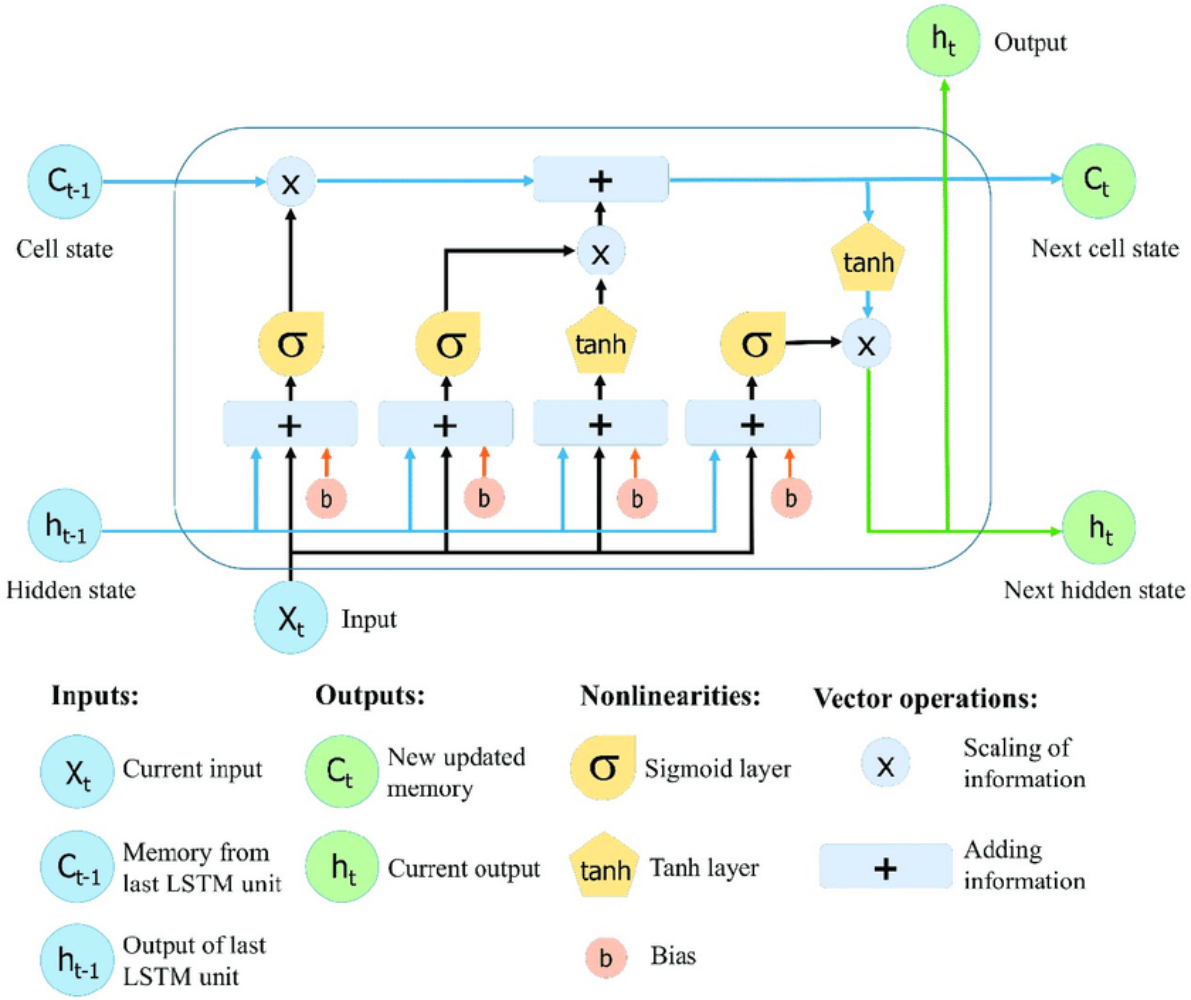

Figura 14: Arquitectura de una célula LSTM (Le, et. al., 2019)

Las LSTM cuentan con tres compuertas. La compuerta de entrada controla cuánta información debe ser procesada y cuánta agregada a su memoria de largo plazo. Se activa mediante una capa interior de neuronas que utilizan una función sigmoidea. Por esta función, los valores resultantes se encuentran acotados entre cero y uno, donde cero representa que la información no es relevante, mientras que valores cercanos a uno indican que la información es importante y debe ser agregada a su memoria de largo plazo. La puerta de olvido es responsable de decidir qué información almacenada en la memoria a largo plazo debe ser eliminada. Al igual que la puerta de entrada, la puerta de olvido se activa mediante una función sigmoidea. La puerta de salida controla cuánta información de la memoria a largo plazo se utilizará para producir la salida actual de la célula. Al igual que las otras puertas, se activa mediante una capa de neuronas que utiliza una función sigmoidea.

3.6. Gated Recurrent Unit (GRU)

Las Gated Recurrent Units (GRU) fueron introducidas por Cho et. al. en 2014 y son un tipo de redes neuronales recurrentes similares a las LSTM. La principal diferencia entre ambas se encuentra en la celda de memoria. Mientras que las LSTM tienen tres puertas (entrada, olvido y salida), las GRU tienen unidades: una de reinicio y una de actualización:

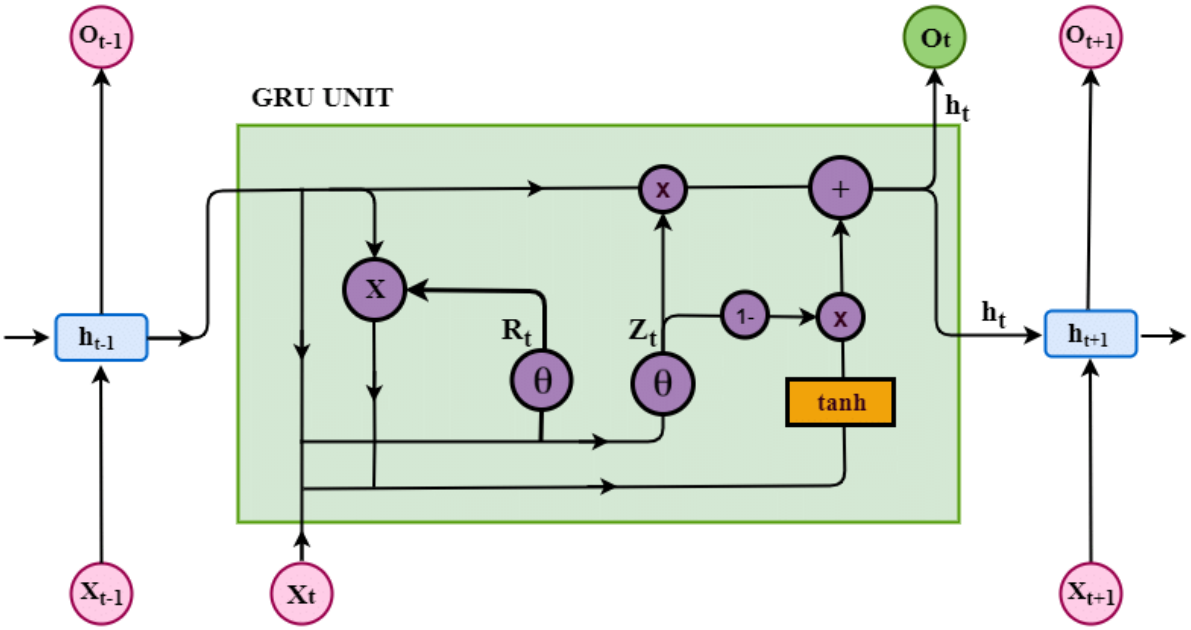

Figura 15: Arquitectura de una célula GRU (Bibi, et. al., 2020)

La unidad de actualización controla cuánta información de la entrada actual debe ser agregada a la representación de la secuencia, mientras que la unidad de reinicio controla cuánta información de la representación anterior se debe olvidar. En términos generales, las GRU son un tipo de LSTM más simple y más rápido de entrenar, ya que tienen menos parámetros. Sin embargo, las LSTM tienen un desempeño ligeramente mejor en problemas más complejos.

La capacidad de retener datos antiguos de las LSTM y las GRU, mediante el uso de compuertas que controlan el flujo de información, hacen que se mitigue el problema de desvanecimiento del gradiente, aunque no lo eliminan definitivamente. En este trabajo, se comparará el rendimiento de ambas arquitecturas recurrentes para el problema de predicción de tickets, las cuales serán entrenadas usando como función de pérdida el error cuadrático medio de las predicciones, optimizado por medio del algoritmo de gradiente descendiente[8].

[8] Ver sección 4.3 del libro "Deep Learning" (Goodfellow et. al., 2016)

4. Resultados y discusión

En esta sección se expondrán los resultados de los modelos para el período de prueba (desde el 01/05/2022 hasta el 29/06/2022) descritos en la sección anterior, junto con el tiempo de entrenamiento y predicción de los mismos. Si bien el período de prueba tiene 3120 registros, en los gráficos se muestran solo los primeros mil por cuestiones de legibilidad. Todos los cómputos se efectuaron en una computadora portátil con un procesador Intel Core i5-1145G7 y 16GB de memoria RAM.

El promedio simple no tiene tiempo de entrenamiento, y en ejecutar las "predicciones" demoró unos 10.08 segundos. El error medio absoluto (MAE) fue de 4.77:

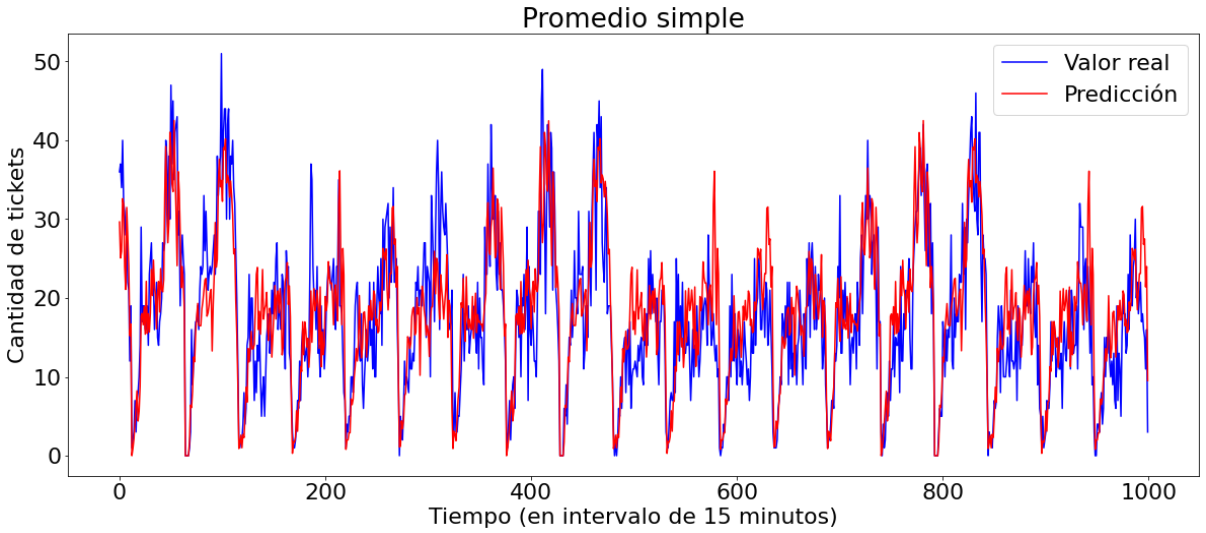

Figura 16: Muestra del promedio simple en el período de prueba. Elaboración propia.

Random Forest demoró 5.76 segundos en entrenar y 53.64 segundos en realizar las predicciones. Como hiper-parámetros del modelo se utilizaron 300 estimadores, con una profundidad máxima de 7, entre otros. El MAE del modelo fue de 4.76:

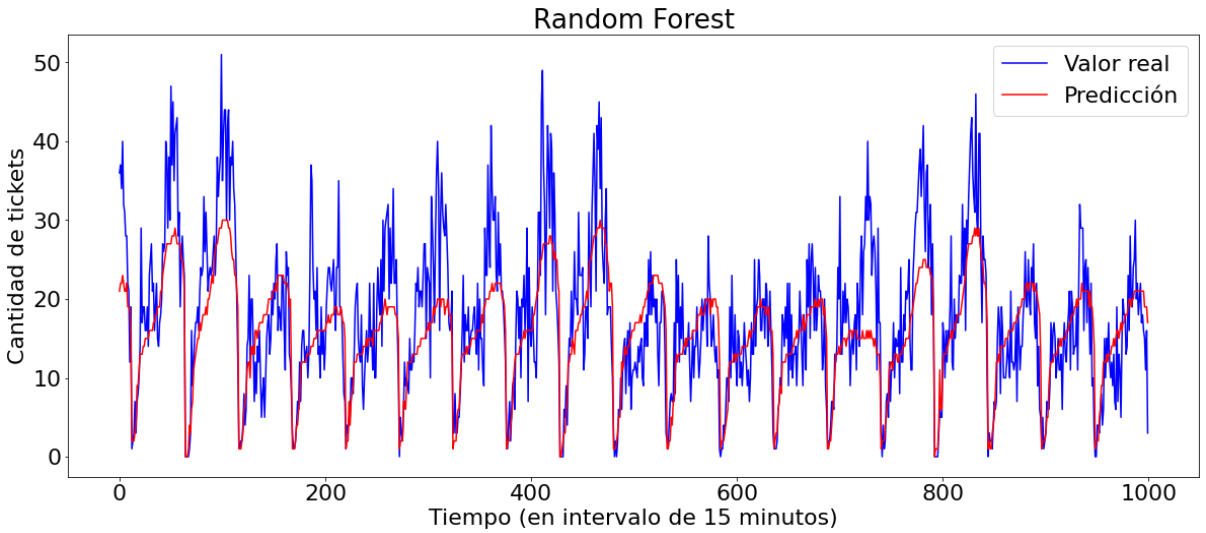

Figura 17: Muestra de Random Forest en el período de prueba. Elaboración propia.

XGBoost demoró 1.79 segundos en entrenar y 25.03 segundos en realizar las predicciones. Como hiper-parámetros del modelo se utilizaron 100 estimadores, con una profundidad máxima de 5 y un *learning rate* de 0.05, entre otros. El MAE del modelo fue de 4.90:

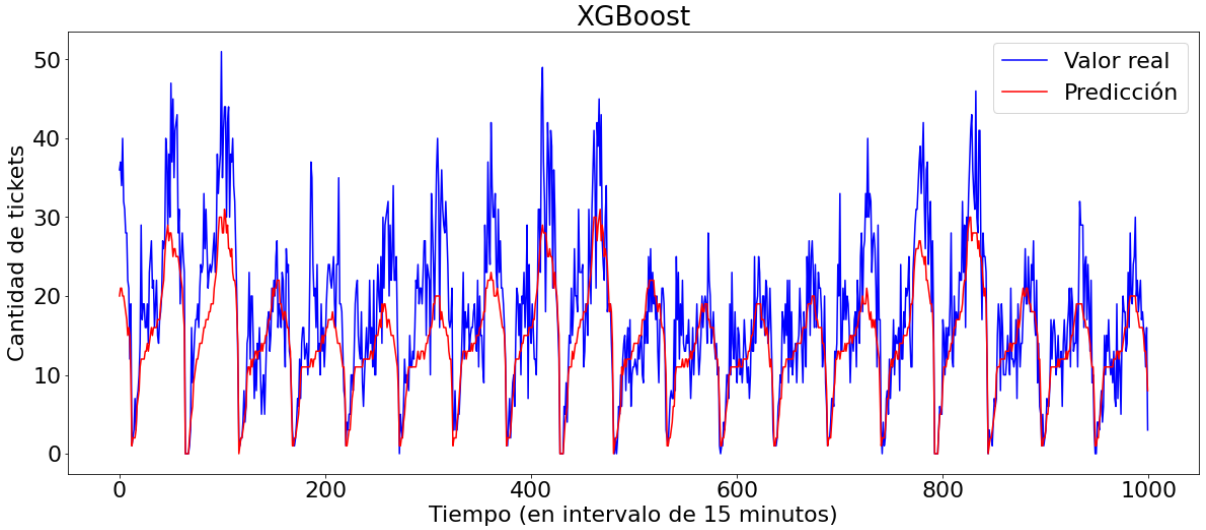

Figura 18: Muestra de XGBoost en el período de prueba. Elaboración propia.

LightGBM demoró 34.81 segundos en entrenar y 1.26 segundos en realizar las predicciones. Como hiper-parámetros del modelo se utilizaron 1000 estimadores, con una profundidad máxima de 8 y un *learning rate* de 0.1, entre otros. El MAE del modelo fue de 4.31:

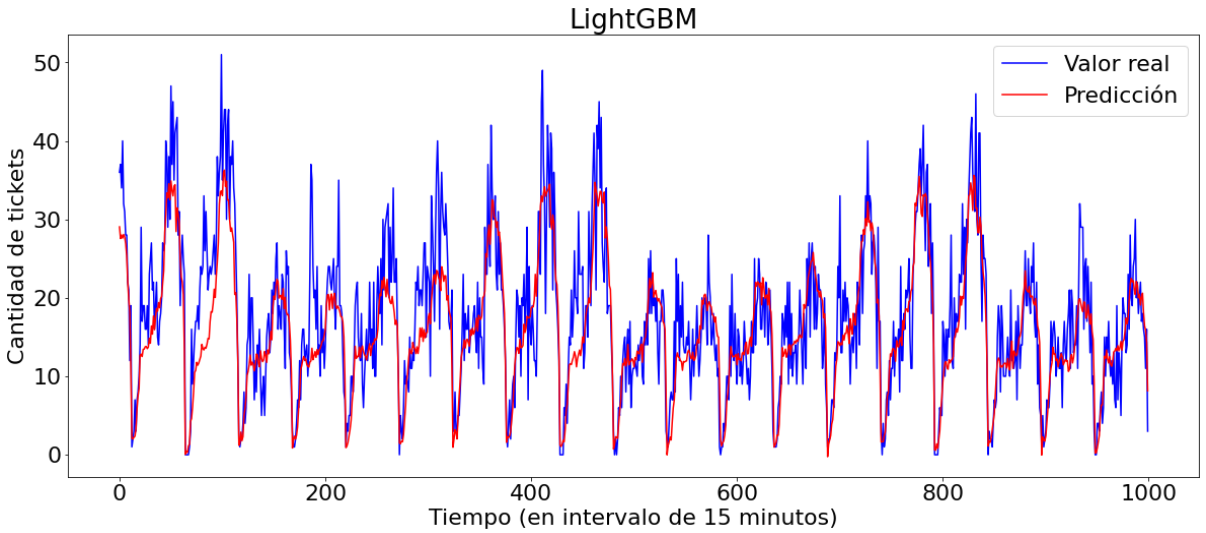

Figura 19: Muestra de LightGBM en el período de prueba. Elaboración propia.

LSTM demoró 84.07 segundos en entrenar y 0.51 segundos en realizar las predicciones. Como hiper-parámetros del modelo se utilizaron dos capas unidireccionales de 128 y 16 células respectivamente, activación ReLU, optimizador "Adam", *learning rate* 0.0002 y *clip value* de 1, entre otros. El MAE del modelo fue de 4.78:

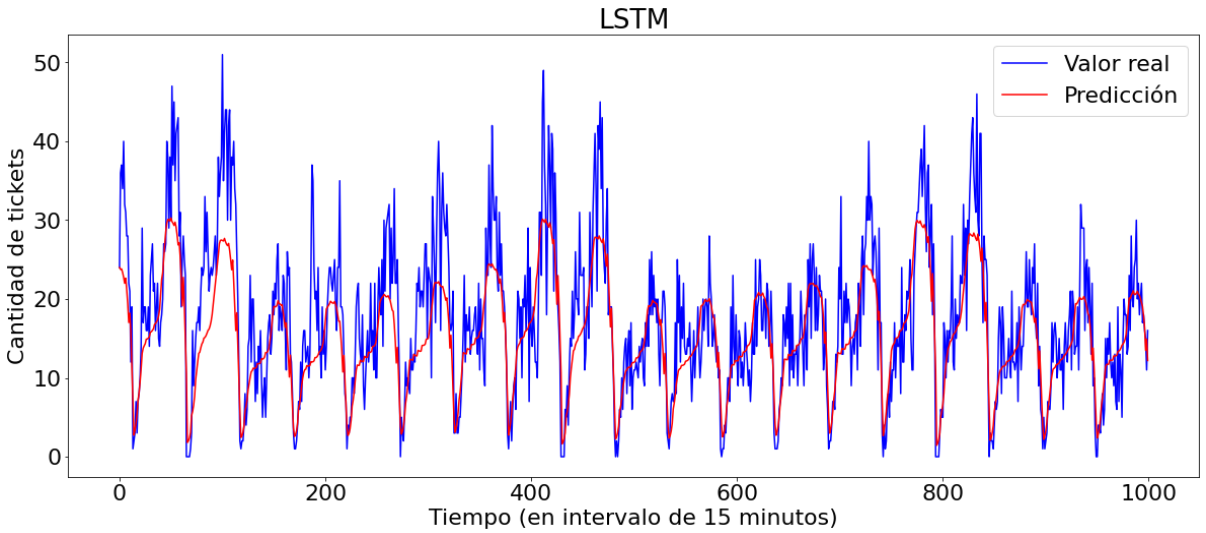

Figura 20: Muestra de LSTM en el período de prueba. Elaboración propia.

GRU demoró 69.24 segundos en entrenar y 0.41 segundos en realizar las predicciones. Como hiper-parámetros del modelo se utilizaron dos capas unidireccionales de 128 y 16 células respectivamente, activación ReLU, optimizador "Adam", *learning rate* 0.0002 y *clip value* de 1, entre otros. El MAE del modelo fue de 4.63:

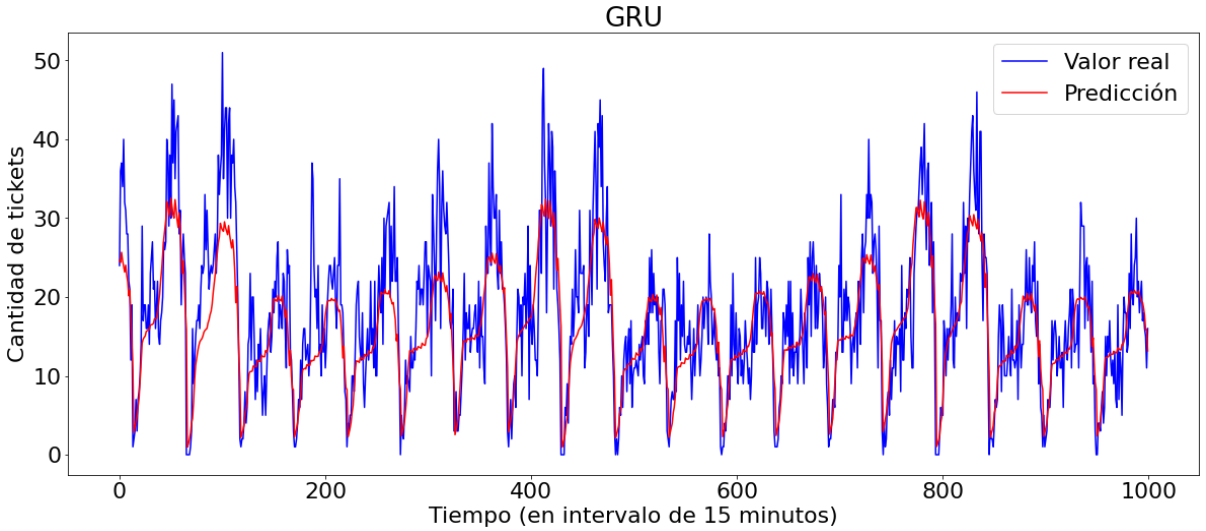

Figura 21: Muestra de GRU en el período de prueba. Elaboración propia.

Tabla 5: Sintetización de resultados[9] (ganador en verde):

Modelo	**Tiempo entrenamiento**	**Tiempo predicción**	**MAE[10]**	**MAE Extra Features[11]**
Promedio simple	-	10.08	4.77	-
Random Forest	5.76	53.64	5.05	4.76
XGBoost	**1.79**	25.03	5.13	4.90
LightGBM	34.81	1.26	**4.31**	13.12
LSTM	84.07	0.51	4.78	5.94
GRU	69.24	**0.41**	4.63	6.06

El modelo elegido como ganador fue LightGBM por tener el menor error (10% menor que el promedio simple) y ser de los más rápidos para predecir, aún cuando su tiempo de entrenamiento no fue el mejor. Para testear esta hipótesis se realizó un test de Wilcoxon sobre los errores absolutos de predicción de cada uno de los modelos comparándolos contra el

[9] Los tiempos de entrenamiento y de predicción reportados corresponden a la versión con mejor MAE.
[10] Sin variables generadas en la sección 2.1
[11] Con variables generadas en la sección 2.1

modelo de LightGBM, resultando en todos los casos diferencias estadísticamente significativas.

4.1. Evaluación de la capacidad de generalización del modelo propuesto a otros centros gastronómicos

Dado el modelo ganador, se decidió evaluar si dicha arquitectura resulta robusta para generar predicciones en otros restaurantes. Los mismos están ubicados en distintas zonas de Brasil, por lo cual tienen diversas características. Por dar un ejemplo, mientras que muchos están ubicados en grandes centros urbanos, otros están ubicados en pequeñas ciudades turísticas; y también poseen distintos climas y variaciones de estacionalidad.

Para comparar los distintos restaurantes entre sí y, dado que tienen diferentes niveles de ventas, utilizaremos el estadístico de SMAPE (ver sección 3). Solo el 4% de los restaurantes poseen un SMAPE mayor que $Q_3 + RIQ$. Mientras que el 16% resultaron menores que $Q_1 - RIQ$. A continuación se presenta una tabla que contiene el valor del SMAPE para el restaurante utilizado durante este trabajo (restaurante 0), y otros 10 restaurantes a modo de ejemplo. Tal como puede apreciarse, la tasa de error SMAPE se mantiene en valores adecuados.

Tabla 6: Resultados del modelo sobre otros restaurantes

Restaurante	**SMAPE**
Restaurant 0	34.4
Restaurant 1	30.73
Restaurant 2	35.39
Restaurant 3	40.09
Restaurant 4	36.26
Restaurant 5	34.54
Restaurant 6	21.2
Restaurant 7	38.3
Restaurant 8	40.66
Restaurant 9	26.85
Restaurant 10	31.64

4.2. Discusión y puesta en producción del modelo

El clima es un factor importante que influye en el comportamiento de las personas en general, y en la demanda de alimentos en particular. Resulta lógico pensar que días de extremo calor disminuya la demanda de alimentos calientes (como hamburguesas) y aumente la de alimentos fríos (como helados). Respecto a la lluvia, se podría esperar una reducción en la demanda dentro del local, pero posiblemente aumente la demanda mediante delivery. En este trabajo en particular, los efectos del clima no pueden ser incluídos en el análisis, ya que la predicción requerida es para dos meses en adelante, lo que imposibilita tener datos medianamente certeros del clima a futuro. Por lo tanto, variables relacionadas al clima no fueron tomadas en cuenta. Al momento de finalización de este trabajo, el proyecto sobre el cual se basó se encuentra terminado y los modelos en producción. Efectivamente, LightGBM fue el modelo seleccionado para generar la predicción sobre los más de 600 restaurantes que se encargaron en una primera etapa, previendo ampliarlo a muchos más. El restaurante elegido como base de comparación (y sobre el cual versa este trabajo), no fue elegido aleatoriamente. Fue elegido particularmente por ser el restaurante donde mejor funcionaba el método anterior (promedio simple móvil de 4 semanas). Los resultados del modelo LightGBM, en comparación con el resto de los restaurantes fueron aún mejores del resultado que se observa en este trabajo, donde pareciera que la mejora en rendimiento se justifica principalmente en que este modelo parece capturar mejor los picos, mientras que el resto de los modelos tiende a suavizarlos. En particular, y como se mencionó anteriormente (ver sección 4.1), la arquitectura del LightGBM elegida generalizó muy bien en otros dominios o restaurantes, donde solo el 4% de ellos se consideran como outliers “negativos” respecto a la performance del resto. A pesar de ello, el rendimiento del modelo en estos dominios es lo suficientemente bueno para que no se considere necesario generar modelos personalizados o particulares para estos restaurantes.

Sin embargo, se encontró durante el desarrollo del proyecto que el modelo se deprecia rápidamente. Esto significa que, una vez construido el objeto modelo, entrenado sobre un set de datos fijo, a medida que pasa el tiempo las predicciones tienden a empeorar rápidamente. Dado esto, se decidió no poner en producción objetos modelos, sino reentrenar con una arquitectura fija cada vez que se ejecuta el pipeline del proyecto; con lo cual el modelo siempre se entrena sobre los datos actualizados. Esto no solo resultó en un mejor rendimiento

sostenido en el tiempo, sino que acortó los tiempos de puesta en funcionamiento del proyecto sustancialmente. Es necesario tener en cuenta que el tiempo de entrenamiento del modelo fue de casi 35 segundos en una computadora personal, pero que se reduce sustancialmente cuando se realiza en un servidor potente, por lo que el tiempo de ejecución del pipeline completo para todos los restaurantes es relativamente corto.

A futuro, una de las posibles mejoras que se podrían implementar en nuestro modelo es cambiar el enfoque de predicción por ticket, a predicción por ítem vendido. Este enfoque más específico podría proporcionar una visión más detallada de las tendencias de ventas. Al analizar las ventas a nivel de ítem, podríamos obtener información más específica sobre qué productos estarían impulsando las ventas y cómo se comportan a lo largo del tiempo. A su vez, esta información más detallada sería de gran ayuda a equipos como *supply chain* a tener mayor precisión de las materias primas y tiempos que necesitaría cada restaurante para operar de la manera más eficiente posible.

5. Conclusión

El objetivo de este trabajo fue generar un modelo de aprendizaje automático que permitiera predecir cuántos tickets serán vendidos en un restaurante de comida rápida ubicado en Brasil, considerando una ventana futura de dos meses y con una granularidad de quince minutos. Dicho sistema permite a los distintos equipos de la empresa trabajar más eficientemente al tener una mejor precisión de la demanda futura.

Para poder resolver este problema, se propuso la utilización de distintos algoritmos de aprendizaje automático (Random Forest, XGBoost, LightGBM, Long Short Term Memory y Gated Recurrent Unit), los cuales fueron entrenados sobre datos de ventas desde el 2019 hasta el 2022. También fueron imputadas algunas variables no contenidas en el dataset original, que se evaluaron como características de entrada adicionales para los modelos estudiados. Se realizó una búsqueda exhaustiva de hiper parámetros mediante distintas técnicas, lo que llevó a un modelo ganador que mejora en un 10% el modelo base que era utilizado hasta el momento.

La realización de esté trabajo permitió obtener experiencia práctica sobre predicción de series de tiempo en un ámbito de datos reales, solucionando un problema concreto. Partir un problema de gran tamaño (múltiples restaurantes) en uno más pequeño (un solo restaurante), permitió entender características del problema que se replican en casi todos (o

todos) los demás. En este sentido, la arquitectura del modelo elegido resultó ser ampliamente robusta a la hora de realizar predicciones sobre otros restaurantes.

El trabajo realizado sobre la selección de variables y el estudio sobre los diferentes modelos de aprendizaje automático en el marco de una única serie de tiempo permitió ahorrar una gran cantidad de tiempo a la hora de escalar esta solución a los demás restaurantes del problema. Esto se debe principalmente a que, tanto el método como la solución, resultaron robustos a la hora de aplicarlos sobre otros restaurantes que no fueron el restaurante en estudio.

6. Referencias

1. Bibi, Iram and Akhunzada, Adnan and Malik, Jahanzaib and Iqbal, Javed and Musaddiq, Arslan & Kim, Sung, A Dynamic DL-Driven Architecture to Combat Sophisticated Android Malware, 2020, IEEE Access.

2. Breiman, L. Random Forests. Machine Learning, 2001, 45, 5–32.

3. Brynjolfsson, Erik and Hitt, Lorin M. and Kim, Heekyung Hellen. Strength in Numbers: How Does Data-Driven Decision Making Affect Firm Performance?. April 22, 2011.

4. Dharmawirya, Mathias and Oktadiana, Hera and Adi, Erwin. Analysis of Expected and Actual Waiting Time in Fast Food Restaurants. Industrial Engineering Letters, 2012, Vol. 2, No. 5.

5. Doganis, Philip and Alexandridis, Alex and Patrinos, Panagiotis and Sarimveis Haralambos. Time series sales forecasting for short shelf-life food products based on artificial neural networks and evolutionary computing. Journal of Food Engineering., July 2006, Volume 75, Issue 2, pp.196-204.

6. Feng, Weijiang and Guan, Naiyang and Li, Yuan & Zhang, Xiang & Luo, Zhigang. Audio visual speech recognition with multimodal recurrent neural networks, 2017.

7. Foster Provost and Tom Fawcett. Data Science and its Relationship to Big Data and Data-Driven Decision Making. Big Data. Mar 2013.51-59.

8. Goodfellow Ian and Bengio Yoshua and Courville Aaron, Deep Learning, 2016, MIT Press.

9. Guolin Ke and Qi Meng and Thomas Finley and Taifeng Wang and Wei Chen and Weidong Ma and Qiwei Ye and Tie-Yan Liu. LightGBM: a highly efficient gradient boosting decision tree. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17), 2017, Curran Associates Inc., Red Hook, NY, USA, 3149–3157.

10. Hochreiter, Sepp and Schmidhuber, Jürgen. Long Short-term Memory, 1997, Neural computation. 9. 1735-80.

11. Ian Phau and Graham Ferguson. Validating the Customer Satisfaction Survey (CSS) Scale in the Australian fast food industry, Australasian Marketing Journal (AMJ), 2013, Volume 21, Issue 3, pp. 147-154.

12. Kyunghyun Cho and Bart van Merriënboer and Caglar Gulcehre and Dzmitry Bahdanau and Fethi Bougares and Holger Schwenk and Yoshua Bengio, Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation, 2014, In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

13. Le, Xuan Hien and Ho, Hung and Lee, Giha and Jung, Sungho, Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting, 2019, Water. 11. 1387.

14. Lee, Hang-Lo and Kim, Jin-Seop and Hong, Chang-Ho and Cho, Dong-Keun. Ensemble Learning Approach for the Prediction of Quantitative Rock Damage Using Various Acoustic Emission Parameters, 2021, Applied Sciences. 11. 4008. 10.3390/app11094008.

15. McCulloch, W.S. and Pitts, W. A logical calculus of the ideas immanent in nervous activity, 1943, Bulletin of Mathematical Biophysics 5, 115–133.

16. Milos Bujisic and Vanja Bogicevic and H. G. Parsa. The effect of weather factors on restaurant sales. Journal of Foodservice Business Research, 2017, 20:3, 350-370.

17. P. Meulstee and M. Pechenizkiy, "Food Sales Prediction: "If Only It Knew What We Know"," 2008 IEEE International Conference on Data Mining Workshops, 2008, pp. 134-143.

18. Rokach, L. and Maimon, O. Decision Trees. In: Maimon, O., Rokach, L. (eds) Data Mining and Knowledge Discovery Handbook, 2005, Springer, Boston, MA.

19. Schmidt, Robin M. Recurrent neural networks (RNNs): A gentle introduction and overview, 2019, arXiv preprint arXiv:1912.05911.

20. Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). Association for Computing Machinery, New York, 2016, NY, USA, 785–794.

21. Zhang, Tao and Lin, Wuyin and Vogelmann, Andrew and Zhang, Minghua and Xie, Shaocheng and Qin, Yi and Golaz, Jean‐Christophe. Improving Convection Trigger Functions in Deep Convective Parameterization Schemes Using Machine Learning. Journal of Advances in Modeling Earth Systems, 2021.