Predicción de ventas futuras

Víctor Alberto Lizcano Portilla

Jimmy Alexander Romero Miranda

Monografía presentada para optar al título de

Especialista en Analítica y Ciencia de Datos

Asesor

Sebastián Rodríguez Colina

Magister (MSc) en ingeniería

Universidad de Antioquia

Facultad de Ingeniería

Especialización en Analítica y Ciencia de Datos

Medellín, Antioquia, Colombia

2021

Cita	(Lizcano Portilla & Romero Miranda, 2021)
Referencia **Estilo APA 7 (2020)**	Lizcano Portilla, V. A., & Romero Miranda, J. A. (2021). *Predicción de ventas futuras, 2021* [Trabajo de Grado Especialización]. Universidad de Antioquia, Medellín, Colombia.

Especialización en Analítica y Ciencia de Datos
Cohorte II

UNIVERSIDAD DE ANTIOQUIA
Vicerrectoría de Docencia

Sistema de Bibliotecas

Centro de Documentación de Ingeniería (CENDOI)

Repositorio Institucional: http://bibliotecadigital.udea.edu.co

Universidad de Antioquia - www.udea.edu.co

Rector: John Jairo Arboleda Céspedes

Decano/Director: Jesús Francisco Vargas Bonilla

Jefe departamento: Diego José Luis Botia Valderrama

TABLA DE CONTENIDOS

LISTA DE FIGURAS

LISTA DE TABLAS

1. RESUMEN

La finalidad del proyecto es predecir las ventas para el mes siguiente para una cadena de tiendas, haciendo uso de algoritmos de inteligencia artificial. El dataset cuenta con 4 archivos (productos, categoría de los productos, tiendas y ventas diarias) tomados de la competencia de kaggle "Predict Future Sales" que proporcionan información detallada de las ventas históricas para 22.171 productos en 34 meses, a partir del mes de enero de 2013. Se probaron diferentes metodologías para organizar la información para la generación de los modelos. Los datos se preprocesaron para tener la información organizada por ventas mensuales para cada tienda y categoría de producto y posteriormente fueron separados de acuerdo a un *lookback* para generar las series de tiempo para entrenar los modelos. Se probaron diferentes modelos y se utilizó una métrica particular de negocio para medir el error de las predicciones.

El código y los notebooks usados se puede encontrar en el siguiente repositorio de github: https://github.com/Alberto7526/MonografiaGithub

2. DESCRIPCIÓN DEL PROBLEMA

La mayoría de las empresas distribuidoras de productos sufren de manera recurrente al no conocer la cantidad o un aproximado de productos que debería mantener en stock ya que, por un lado, si el stock es demasiado grande se pueden producir pérdidas de mercancía y costos innecesarios de transporte y almacenamiento, por el contrario, si el stock es demasiado pequeño, este será insuficiente para cubrir la demanda de los diferentes productos y se verá reflejado en la pérdida de clientes. Este problema no solo afecta a la empresa en términos económicos, sino que a largo plazo en el prestigio de la misma, sin mencionar la incomodidad que esto causa a los clientes, al no poder satisfacer sus necesidades de demanda de ciertos productos de manera rápida, se ven obligados a trasladarse en busca de otro proveedor.

El problema a resolver consiste en determinar la cantidad de productos que se van a vender (por tienda y categoría) para un mes, dado el histórico de ventas para meses previos, esto permitirá a la tienda tener la capacidad de organizar mejor su presupuesto y disponibilidad en ventas.

2.1 APROXIMACIÓN DESDE LA ANALÍTICA DE DATOS

En primera instancia, antes de implementar un modelo predictivo, es de gran importancia realizar un análisis y exploración de los datos, que permite encontrar algunas relaciones entre las diferentes variables que conforman el dataset, así como identificar discontinuidades, datos faltantes e incluso valores atípicos, comúnmente denominados outliers. Luego de ello se desarrollarán modelos que permitan a la compañia estimar cuántos productos deberá mantener en stock cada mes, de este modo no tendrá pérdidas importantes de dinero o espacio por mantener existencia de productos que no se venden con tanta frecuencia, y poder suplir las necesidades de sus clientes sin retrasos en sus compras, optimizando el gasto mensual y ayudando a hacer más eficiente la compañía.

2.2 ORIGEN DE LOS DATOS

Los datos usados para generar el modelo fueron proporcionados por la empresa rusa “1C” especializada en el desarrollo, distribución, publicación y soporte a software, con mercado de producción en masa[1]. La compañía proporcionó el dataset para ser usado con fines académicos y es incluido en la competencia de Kaggle “Predict Future Sales”[2], los datos presentan las ventas diarias para los diferentes productos en las diferentes tiendas, cuenta con 34 meses de registro desde el 01-01-2013 hasta el 30-10-2015.

[1] 1C Company. https://1c.ru/eng/title.htm

[2] Predict Future Sales. Final project for "How to win a data science competition" Coursera course. https://www.kaggle.com/c/competitive-data-science-predict-future-sales

2.3 MÉTRICAS DE DESEMPEÑO

Las métricas de desempeño son de gran utilidad al momento de determinar si un modelo cumple con las expectativas de negocio o no, para el problema de predicción de ventas, el cual se trata de una tarea de regresión se tomaron en cuenta diferentes métricas:

- Raíz cuadrada del Error cuadrático medio(RMSE)
- Custom error (métrica de negocio pensada en el cliente)

Raíz cuadrada del Error cuadrático medio(RMSE):

El RMSE (Root Mean Square Error) consiste en calcular la raíz cuadrada del error cuadrático medio (ecuación (1)) (Raschka, 2019). Esta métrica es proporcionada por la competencia de kaggle y será utilizada para ajustar y evaluar el modelo.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y_{real} - y_{predic}\right)^2} \quad (1)$$

Custom error:

La métrica de negocio es una medida utilizada por las empresas para definir y evaluar el desempeño de un modelo o de un proceso en particular, generalmente, es diseñada para adaptarse al problema de negocio. En este trabajo se diseñó un *"custom error",* está métrica permite interpretar el error de predicción como el costo de mantenimiento que se genera si el modelo predice valores por arriba de los reales, o como costo de oportunidad, indicando el dinero que deja de ganar en caso de que el modelo predice valores por debajo del valor real.

En la ecuación (2) se muestra el cálculo efectuado para el costo de oportunidad, teniendo en cuenta que el error es negativo (se predice por debajo del valor real)

$$Costo_{oportunidad} = \sum_{i=0}^{N-1}\left(\left|y_{i-real} - y_{i-predic}\right| * \left(Precio_{venta} - Costo_{producto}\right)\right) \quad (2)$$

El costo requerido para mantener cada producto en almacén está definido en la Tabla 1. Los valores presentados son consideraciones tomadas para el desarrollo de los modelos y no necesariamente corresponden a los valores reales en la industria. Durante la aplicación real se deberían conocer estos valores para una mejor aproximación.

Gastos	Valor unitario	Cantidad	Valor total
Salario de los empleados	$ 5.964	2	$11.928
Alquiler del almacén	$8.000	1	$8.000
Servicios	$500	1	$500
Transporte	$500	1	$500
Total Costo de almacén			$20.928

Tabla 1. Costos mensuales de almacén por tienda

El costo total del producto se obtiene como se muestra en la ecuación (3)

$$Costo_{Producto} = Precio_{base} + \frac{Costo_{almacen}}{Total\ productos} \quad (3)$$

En la ecuación (4) se muestra el cálculo efectuado para el costo de mantenimiento, el cual se aplica si el error de predicción es positivo (si se predice por encima del valor real).

$$Costo_{mantenimiento} = \frac{Costo_{almacen}}{Total\ productos} * \left| y_{i-real} - y_{i-predic} \right| \quad (4)$$

Dada la naturaleza de la tarea de regresión, y como se pudo observar en las ecuaciones que definen las métricas anteriores, el valor ideal que se espera conseguir en cualquiera de ellas es cero; sin embargo, este objetivo es imposible de cumplir desde un punto de vista práctico.

3. DATOS

3.1 DATOS ORIGINALES

Los datos originales suministrados por la competencia de kaggle "Predict Future Sales" están conformados por los siguientes datasets:

- Items.csv
- items_categories.csv
- Shops.csv
- Sales_train.csv

Los datos vienen organizados en forma tabular, cada archivo presenta información separada relacionada con los diferentes productos, las diferentes categorías, las tiendas y el registro de ventas histórico, a continuación se describirán cada uno por separado. Para la descarga de los datasets se puede realizar a través de la competencia de kaggle[3], ya que los datos no contienen información sensible y no presentan ningún tipo de restricción.

Items.csv

Proporciona información de 22.171 ítems (productos), el dataset contiene 3 columnas: *item_name*, *item_id* e *item_category_id* (figura 1). El primero indica el nombre del producto en ruso, el cual no hace falta traducir porque no usaremos el nombre como variable en el desarrollo del modelo predictivo; el *item_id* representa un identificador único para cada producto que posteriormente me permitirá relacionarlo con otras tablas; y finalmente el *item_category_id* que indica el identificador único de la categoría a la cual pertenece cada producto.

	item_name	item_id	item_category_id
0	! ВО ВЛАСТИ НАВАЖДЕНИЯ (ПЛАСТ.) D	0	40
1	!ABBYY FineReader 12 Professional Edition Full...	1	76
2	***В ЛУЧАХ СЛАВЫ (UNV) D	2	40
3	***ГОЛУБАЯ ВОЛНА (Univ) D	3	40
4	***КОРОБКА (СТЕКЛО) D	4	40

Figura 1. Items.csv

[3] Competencia de kaggle "Predict Future Sales" https://www.kaggle.com/c/competitive-data-science-predict-future-sales/data

En la tabla 2 se muestra una descripción de cada una de las columnas

Columna	Descripción
item_name	El nombre del item, en ruso
item_id	El id del producto
item_category_id	El id de la categoría del producto relacionado

Tabla 2. Descripción columnas items.csv

Items_categories.csv

Este archivo presenta una relación entre los nombres de cada una de las categorías a la cual pertenecen los diferentes productos que se ofrecen y su correspondiente ID cuenta con 84 registros que corresponden a 86 categorías diferentes. En la figura 2 se muestra parte del contenido de este archvo.

	item_category_name	item_category_id
0	PC - Гарнитуры/Наушники	0
1	Аксессуары - PS2	1
2	Аксессуары - PS3	2
3	Аксессуары - PS4	3
4	Аксессуары - PSP	4

Figura 2. items_categories.csv

En la tabla 3 se muestra una descripción de cada una de las columnas

Columna	Descripción
item_category_name	El nombre de la categoría, en ruso
item_category_id	El id de la categoría del producto relacionado

Tabla 3. Descripción columnas items_categories.csv

Shops.csv

El archivo presenta una relación entre los nombres para cada una de las tiendas disponibles con su correspondiente id cuenta con 61 registros correspondientes a 60 tiendas, más adelante se utiliza el id de

la tienda junto con el de la categoría para poder predecir las ventas, en la figura 3 se muestran los primeros 5 registros del dataset.

	shop_name	shop_id
0	!Якутск Орджоникидзе, 56 фран	0
1	!Якутск ТЦ "Центральный" фран	1
2	Адыгея ТЦ "Мега"	2
3	Балашиха ТРК "Октябрь-Киномир"	3
4	Волжский ТЦ "Волга Молл"	4

Figura 3. shops.csv

En la tabla 4 se muestra una descripción de cada una de las columnas

Columna	Descripción
shop_name	El nombre de la tienda, en ruso
shop_id	El id de la tienda

Tabla 4. Descripción columnas shops.csv

Sales_train.csv

Cuenta con 2.935.851 registros, que corresponden a la cantidad de ventas diarias de los diferentes productos organizados por fechas desde el 01 de enero de 2013 hasta el 10 de octubre de 2015. Cabe mencionar que no todas las filas corresponden a una única fecha, sino que estas se encuentran organizadas por producto y tienda; el dataset cuenta con columnas adicionales que proporcionan información extra como lo son *date_block_num,* el cual indica a qué mes corresponden los datos, siendo 0 el primer y 33 el último mes; *item_price* indica el precio con el que se vende dicho producto, finalmente, se muestra la cantidad de ventas de cada producto por día y tienda. En la figura 4 se muestran los primeros 10 registros.

	date	date_block_num	shop_id	item_id	item_price	item_cnt_day
0	02.01.2013	0	59	22154	999.00	1.0
1	03.01.2013	0	25	2552	899.00	1.0
2	05.01.2013	0	25	2552	899.00	-1.0
3	06.01.2013	0	25	2554	1709.05	1.0
4	15.01.2013	0	25	2555	1099.00	1.0
5	10.01.2013	0	25	2564	349.00	1.0
6	02.01.2013	0	25	2565	549.00	1.0
7	04.01.2013	0	25	2572	239.00	1.0
8	11.01.2013	0	25	2572	299.00	1.0
9	03.01.2013	0	25	2573	299.00	3.0

Figura 4. Sales_train.csv

En la tabla 5 se muestra una descripción de cada una de las columnas

Columna	**Descripción**
date	La fecha, en formato dd.mm.yyyy
date_block_num	El número del mes en el histórico, inicia en 0 y termina en 33
shop_id	El id de la tienda
item_id	El id del producto
item_price	El precio de venta del producto en determinada fecha y tienda
item_cnt_day	La cantidad de producto vendida en determinada fecha y tienda. Un valor negativo representa una devolución del producto

Tabla 5. Descripción columnas sales_train.csv

3.2 ANALÍTICA DESCRIPTIVA

El objetivo principal de esta etapa es la de conocer un poco más acerca de los datos, ayudando así a comprender el problema de negocio a mayor profundidad y posteriormente poder procesar los datos y seleccionar un modelo de predicción con mayor criterio. A continuación se muestran algunas características relevantes de los datos, encontradas al momento de realizar la limpieza y preparación de los mismos.

El dataset fue organizado por categoría y por tienda, separando cada mes en columnas, seguidamente se obtiene una descripción de los datos con el objetivo de verificar si en un principio existen valores nulos o si los tipos de datos no son los adecuados, Como se muestra en la figura 5 se poseen 3271 registros, no se poseen valores nulos y las ventas están representadas por datos del tipo float.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3271 entries, 0 to 3270
Data columns (total 37 columns):
 #   Column            Non-Null Count  Dtype       #   Column            Non-Null Count  Dtype
---  ------            --------------  -----      ---  ------            --------------  -----
 0   id                3271 non-null   object      19  month_16          3271 non-null   float64
 1   shop_id           3271 non-null   int64       20  month_17          3271 non-null   float64
 2   item_category_id  3271 non-null   int64       21  month_18          3271 non-null   float64
 3   month_0           3271 non-null   float64     22  month_19          3271 non-null   float64
 4   month_1           3271 non-null   float64     23  month_20          3271 non-null   float64
 5   month_2           3271 non-null   float64     24  month_21          3271 non-null   float64
 6   month_3           3271 non-null   float64     25  month_22          3271 non-null   float64
 7   month_4           3271 non-null   float64     26  month_23          3271 non-null   float64
 8   month_5           3271 non-null   float64     27  month_24          3271 non-null   float64
 9   month_6           3271 non-null   float64     28  month_25          3271 non-null   float64
 10  month_7           3271 non-null   float64     29  month_26          3271 non-null   float64
 11  month_8           3271 non-null   float64     30  month_27          3271 non-null   float64
 12  month_9           3271 non-null   float64     31  month_28          3271 non-null   float64
 13  month_10          3271 non-null   float64     32  month_29          3271 non-null   float64
 14  month_11          3271 non-null   float64     33  month_30          3271 non-null   float64
 15  month_12          3271 non-null   float64     34  month_31          3271 non-null   float64
 16  month_13          3271 non-null   float64     35  month_32          3271 non-null   float64
 17  month_14          3271 non-null   float64     36  month_33          3271 non-null   float64
 18  month_15          3271 non-null   float64    dtypes: float64(34), int64(2), object(1)
                                                   memory usage: 945.6+ KB
```

Figura 5. Descripción de los datos

Como el objetivo de este proyecto consiste en predecir las ventas de las diferentes tiendas, se realiza una gráfica (figura 6), la cual nos permite ver la serie temporal de las ventas para todas las tiendas. Es importante tener en cuenta que el mes 0 corresponde a enero de 2013 y se va incrementando en 1 por cada mes hasta llegar al mes 33 que corresponde a octubre del 2015. Nótese cómo existen unos picos en ventas muy marcados en los meses 11 y 23, los cuales corresponden a diciembre de 2013 y 2014 respectivamente, esto coincide con la época navideña, como es común en esta época se esperan mayores números en las ventas. También es de notar como las ventas para el último año presentan una tendencia decreciente.

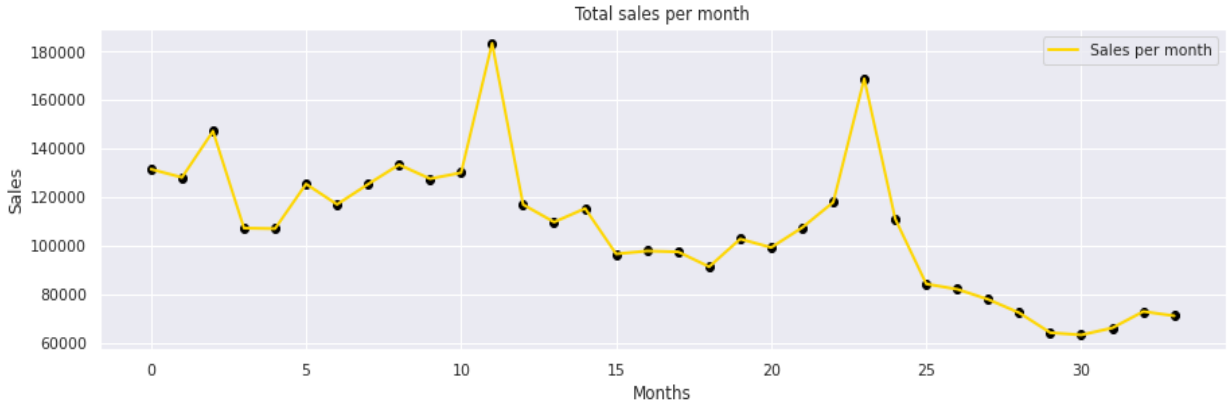

Figura 6. Serie temporal de las ventas totales por mes

Analizando más a fondo, se decide ver el comportamiento de las ventas, pero esta vez discriminado por tiendas, el resultado se muestra en la figura 7. En primera instancia se puede observar cómo los picos en ventas para los meses 11 y 23, se siguen presentando para cada tienda, y de manera adicional podemos observar que no todas las tiendas presentan ventas para cada uno de los meses registrados, las tiendas que se encuentran enmarcadas con un cuadro de color rojo en un inicio presentaban ventas pero, al pasar el tiempo, el registro se mantiene en cero, lo que nos indica que estas tiendas fueron cerradas. Las tiendas enmarcadas con un cuadro de color verde son tiendas que llamaremos tiendas ocasionales, estas tiendas solo permanecen abiertas durante un periodo de tiempo limitado, para este caso se puede observar que solo registran ventas para el mes de septiembre, y finalmente las tiendas enmarcadas con un cuadro de color azul son otro tipo especial de tiendas que no registran ventas desde el inicio, podriamos decir que abrieron sus puertas al publico despues de cierto tiempo pero que en el mes 34 (el cual queremos predecir) permaneceran abiertas.

En la figura 8 se muestran las ventas para la tienda 37 y la tienda 50, se puede observar cómo la tienda 37, que permanece abierta en todos los meses de los que se tiene información, presenta un comportamiento similar al de las ventas totales mostrado en la figura 6, dejando muy marcados los picos para el mes de diciembre, mientras que la tienda 50 es una tienda ocasional y solo presenta ventas para 3 meses en particular.

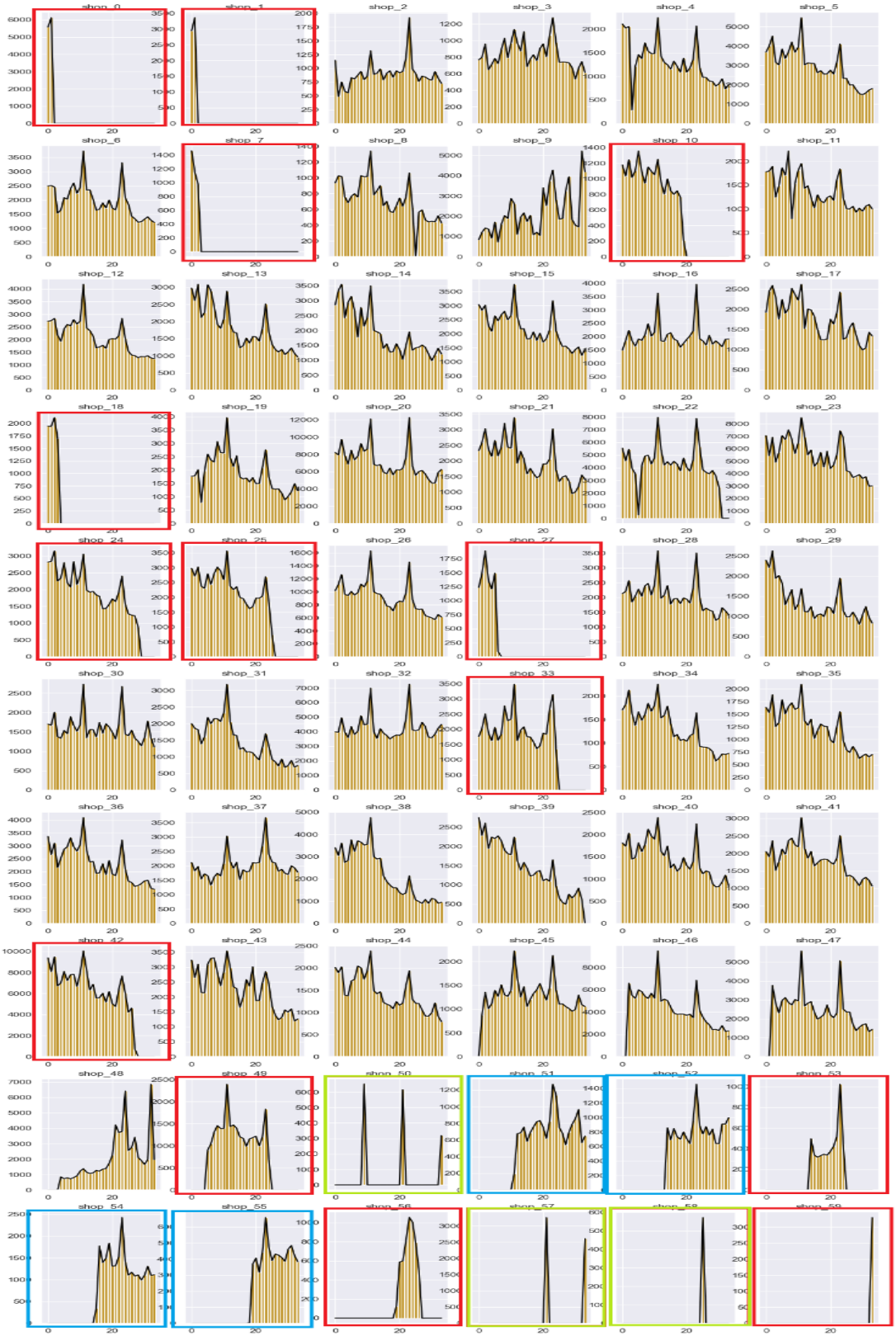

Figura 7.Ventas mensuales por tienda.

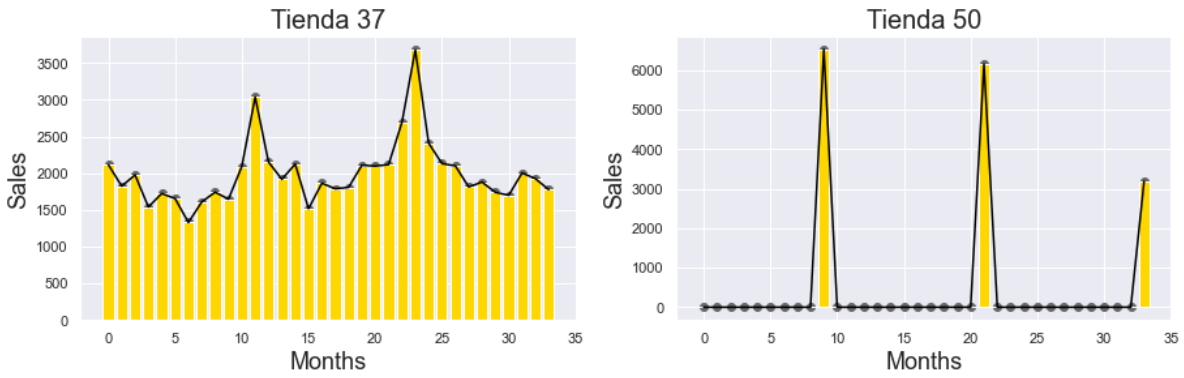

Figura 8.Ventas para las tiendas 37 y 50.

La detección de outliers es un aspecto de gran importancia al momento de generar un modelo de predicción, ya que estos valores atípicos puede ocasionar modelos con sesgo elevado y ajustes a los hyperparameters que, más que ayudar a que el rendimiento del modelo sea óptimo, harán más lenta la curva de aprendizaje (Hawkins, 2013). Para el caso de ventas es importante eliminar los outliers, ya que estos valores se pueden presentar gracias a eventos o campañas publicitarias realizadas en ciertos días del mes y no hacen parte del comportamiento habitual en ventas para determinado producto. En la figura 9 se observa los diferentes outliers discriminados por tiendas y categoría.

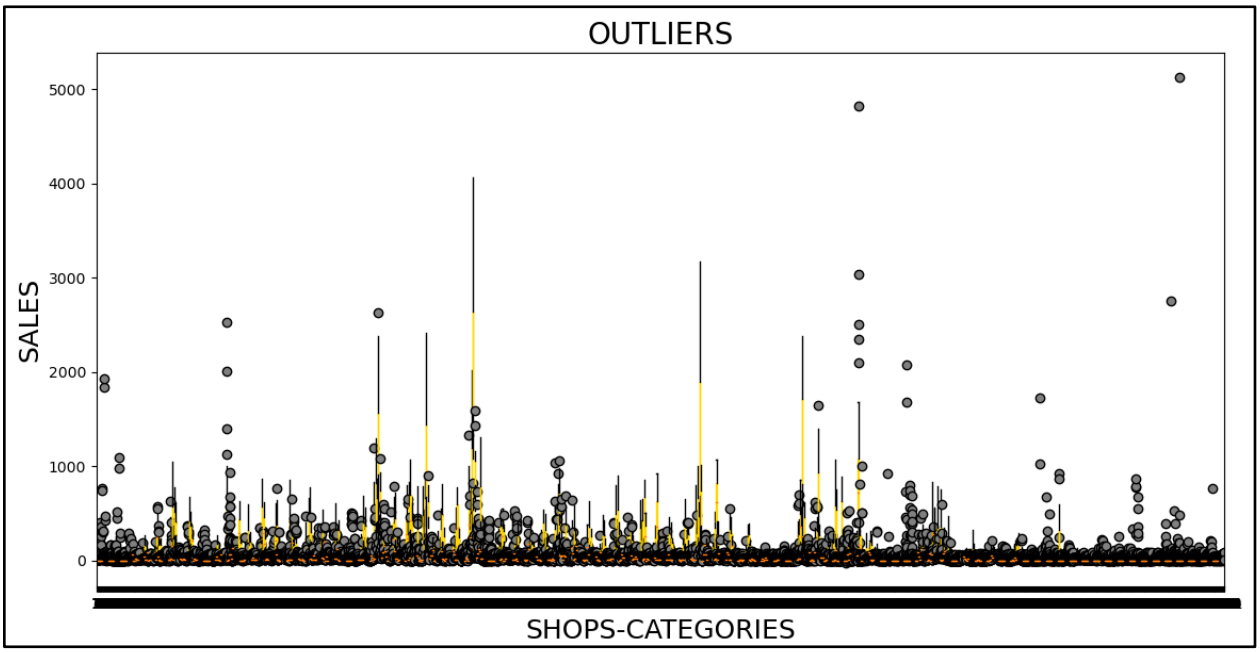

Figura 9.Outliers por tienda y categoría.

4. PROCESO DE ANALÍTICA

4.1 PIPELINE PRINCIPAL

En la figura 10, se describe el flujo de trabajo llevado para la generación del modelo. Inicialmente se realiza la limpieza y transformación de los datos, para esta transformación tiene en cuenta el objetivo final del proyecto: predecir las ventas del mes siguiente; luego se realiza feature engineering para obtener características adicionales que permitirán mejorar el rendimiento del modelo. Seguido a la preparación de los datos, se realiza la separación del dataset modificado inicialmente en dos grandes grupos, *"train"* y *"test"*, teniendo en cuenta la variable tiempo para su división; los datos seleccionados para el entrenamiento también son utilizados para la búsqueda del mejor modelo, utilizando *"grid search"* y "*validación cruzada*". Finalmente se realiza el entrenamiento del modelo con los mejores hyperparameters y se evalúa con los datos de prueba.

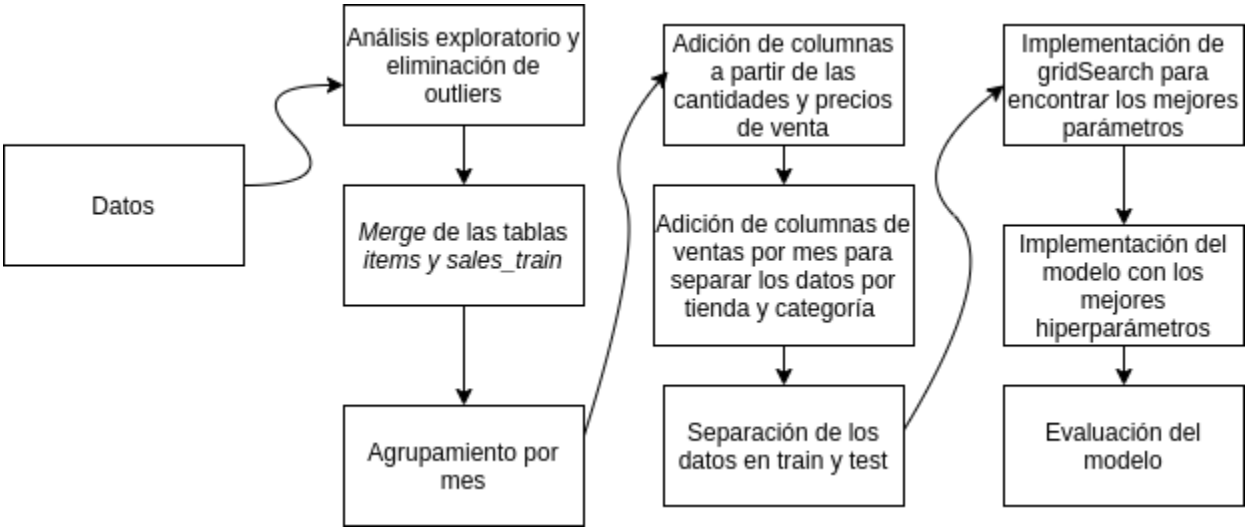

Figura 10. Pipeline de trabajo.

4.2 PREPROCESAMIENTO

Como se mencionó en el apartado de descripción del problema, el objetivo del modelo es el de predecir las ventas por tienda y categoría para el mes siguiente, partiendo de las ventas de los meses anteriores. Teniendo en cuenta esto y que nuestro dataset presenta las ventas diarias de cada producto de forma individual, se procede a combinar el dataset de "*items"* con el dataset de *"sales train"*, tomando la categoría correspondiente de cada producto y agregandola al dataset "*sales train"* tal como se muestra en la figura 11.

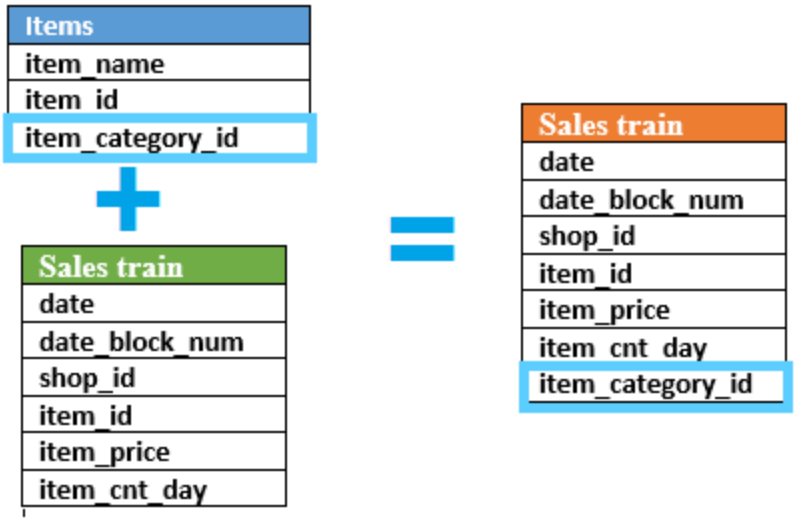

Figura 11. Merge (items, sales_train).

Seguidamente se toman las diferentes filas y se procede a agrupar de acuerdo al mes (*date_block_num*), a la tienda y a la categoría, sumando las ventas que pertenezcan a la misma tienda, la misma categoría y se encuentren en el mismo mes. Además de eliminar las columnas "*date*", "*item id*", "*item price*" y renombrar la columna "*item_cnt_day*" por "*item_cnt_month*", el resultado se muestra en la figura 12.

	date_block_num	shop_id	item_category_id	item_cnt_month
0	0	0	2	53.0
1	0	0	3	28.0
2	0	0	4	16.0
3	0	0	5	28.0
4	0	0	6	65.0
5	0	0	11	24.0
6	0	0	13	9.0
7	0	0	14	8.0
8	0	0	15	6.0
9	0	0	19	345.0

Figura 12. Dataset agrupado por mes, tienda y categoría.

Para facilitar la división del dataset en *"train"*, *"validación"* y *"test"* se decide transformar el dataset, de tal manera que cada fila contenga el id de la tienda, el id de la categoría y los meses organizados por columnas desde el mes 0 al mes 33, tal como se muestra en la figura 13.

shop_id	item_category_id	month_0	month_1	month_2	month_3	month_4	month_5	...	month_31	month_32	month_33
0	2	53.0	52.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	3	28.0	24.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	4	16.0	22.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	5	28.0	35.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	6	65.0	79.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	11	24.0	17.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	13	9.0	10.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	14	8.0	7.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	15	6.0	24.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
0	19	345.0	400.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0

Figura 13. New_dataset.csv

Con el dataset organizado por tienda y categoría, se procede a eliminar aquellos datos consideradas outliers, para este caso no se toma la totalidad de los outliers, sino, aquellos que se encuentran muy alejados del conjunto de datos , el resultado se muestra en la figura 14.

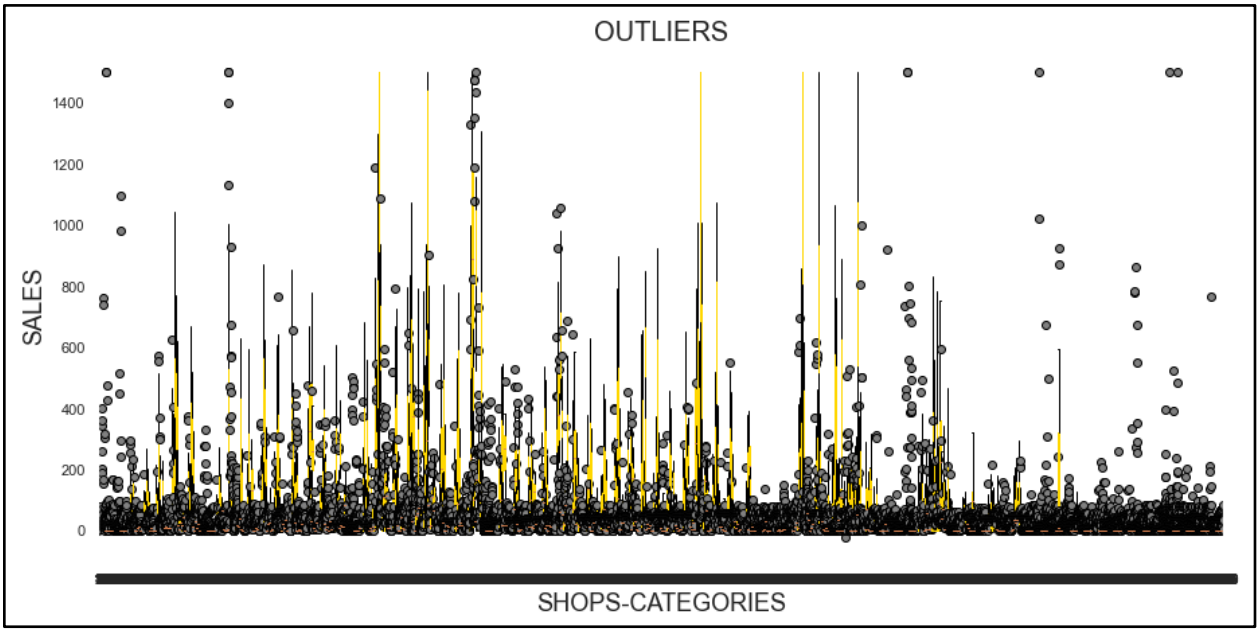

Figura 14. Distribución de las ventas por categorías y tiendas.

Posteriormente, se decide adicionar variables al dataset con el fin de proporcionar al modelo datos que permitan una mejor predicción. Las nuevas columnas corresponden a los promedios de ventas y unidades vendidas por tienda y por categoría, el resultado obtenido se puede observar en la figura 15.

	shop_id	item_category_id	average_price_per_shop	average_price_per_category	average_items_per_shop	average_items_per_category	month_0
0	0	2	563.44	2261.71	1.19	1.06	53.0
1	0	3	563.44	2105.68	1.19	1.18	28.0
2	0	4	563.44	554.70	1.19	1.04	16.0
3	0	5	563.44	865.31	1.19	1.03	28.0
4	0	6	563.44	1892.90	1.19	1.11	65.0

Figura 15. Dataset con nuevas columnas creadas.

4.3 MODELOS

Para el problema de regresión que se desea resolver, se decide implementar diferentes modelos, con el propósito de comparar cuál es su resultado de acuerdo a las propiedades de cada uno de ellos. En la sección anterior de preprocesamiento se hace alusión a los outliers y se muestra la eliminación de algunos de ellos, sin embargo, eliminar el 100% es una tarea bastante compleja, el modelo RANSAC permite trabajar un poco mejor con datos que presentan este problema. Debido a que el problema involucra series temporales, se decide implementar una arquitectura combinada de redes neuronales recurrentes (diseñadas especialmente para trabajar con datos dependientes del tiempo) con redes totalmente conectadas.

RANSAC:

El modelo RANSAC (RANdom SAmple Consensus) es particularmente útil para mitigar la presencia de outliers en los datos, puesto que para la generación del modelo toma varias muestras aleatorias y determina la mejor función para la mayoría de los elementos en las muestras (Raschka, 2019). En nuestro caso podría ser de utilidad para una primera aproximación para predecir las ventas en cualquier mes excepto diciembre, puesto que tenemos un fuerte cambio del comportamiento en este mes y los valores podrían ser considerados outliers.

Para determinar los mejores hiper parámetros para el modelo mediante GridSearchCV se usaron los siguientes valores:

- *max_trials:* [30,40,50,70,80,100,120,150]
- *residual_threshold:* [20,100,500,1000, 1500, 3000]

Random Forest Regression:

Otra estrategia útil para la predicción es el Random Forest. Esta estrategia permite generar múltiples árboles de decisión para finalmente obtener el consenso de todos los árboles generados. La profundidad (la cantidad de ramificaciones que puede tener) y la cantidad de estimadores se toman como los hiperparámetros a determinar mediante GridSearchCV:

- *n_estimators:* [30,40,50,70,80,100,120,150]
- *max_depth:* [20,100,500]

Redes neuronales recurrentes:

Son un tipo especial de redes neuronales que permiten procesar información, así como ajustar los parámetros del modelo teniendo en cuenta datos anteriores, su característica principal es la de poseer celdas retroalimentadas que permite separar cada elemento en cada una de ellas y crear de está manera una especie de memoria que nos permite recordar elementos anteriores (Díaz Bou, 2020).

La arquitectura de red definida para dar solución al problema se muestra en la figura 16, este tipo de redes combinadas permiten trabajar con series temporales y adicionar características que no necesariamente dependen del tiempo. La red consta de 3 capas recurrentes del tipo LSTM (Long short-term memory) con 50, 25 y 10 neuronas, donde se toma como entrada la serie temporal, una capa densa con 10 neuronas totalmente conectadas cuya entrada consiste en características propias de la tienda y la categoría, posteriormente se combinan las dos arquitecturas concatenando los resultados de las mismas, para finalmente aplicar una capa densa con 10 neuronas y una capa final para generar la predicción. En la figura 17 se muestra el resumen de la red implementada.

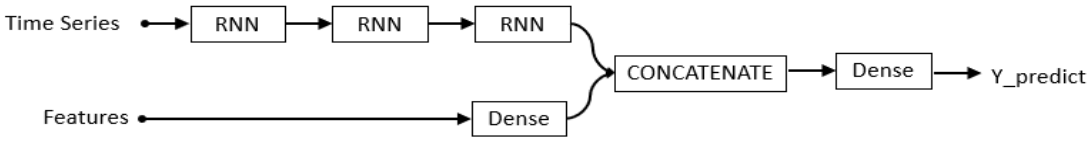

Figura 16. Arquitectura para una red neuronal combinada.

```
Model: "model_4"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_10 (InputLayer)           [(None, 3, 1)]       0
__________________________________________________________________________________________________
lstm_12 (LSTM)                  (None, 3, 50)        10400       input_10[0][0]
__________________________________________________________________________________________________
lstm_13 (LSTM)                  (None, 3, 25)        7600        lstm_12[0][0]
__________________________________________________________________________________________________
input_9 (InputLayer)            [(None, 6)]          0
__________________________________________________________________________________________________
lstm_14 (LSTM)                  (None, 10)           1440        lstm_13[0][0]
__________________________________________________________________________________________________
dense_12 (Dense)                (None, 10)           70          input_9[0][0]
__________________________________________________________________________________________________
concatenate_4 (Concatenate)     (None, 20)           0           lstm_14[0][0]
                                                                 dense_12[0][0]
__________________________________________________________________________________________________
dense_13 (Dense)                (None, 10)           210         concatenate_4[0][0]
__________________________________________________________________________________________________
dense_14 (Dense)                (None, 1)            11          dense_13[0][0]
==================================================================================================
Total params: 19,731
Trainable params: 19,731
Non-trainable params: 0
```

Figura 17. Resumen de la red neuronal combinada.

4.4 MÉTRICAS

En el inciso 2.3 se definió de forma matemática y conceptual cada una de las métricas utilizadas para ajustar y evaluar los diferentes modelos. La implementación en python se llevó a cabo con ayuda de los paquetes keras.backend de tensorflow y numpy.

Para la métrica de negocio fue necesario implementar 3 funciones diferentes, la primera denominada *"Calculate_cost_of_sales"* permite calcular el costo total del producto, recibe como parámetros, el gasto total de mantenimiento por producto y el precio base, con impuestos incluidos. La función "*calculate_expense*" permite calcular el gasto generado por cada tipo de producto (categoría), recibe como parámetros el costo por producto y lo multiplica por el inventario existente de dicho producto. Finalmente la función de costo *"eval"* utiliza las funciones mencionadas anteriormente para generar como salida el costo de mantenimiento y el costo de oportunidad, definido en la sección 2.3, la implementación python de estas funciones se encuentra en el repositorio de este trabajo, disponible en el link adjunto[4], en el script metrics.py.

5. METODOLOGÍA

5.1 BASELINE

En la primera iteración se agruparon los datos por mes, tienda y categoría, posteriormente se separaron en diccionarios para cada par producto-tienda y se generó un algoritmo para completar los datos para los meses faltantes con 0 y generar las series de tiempo que se llevarían al modelo usando un *lookback* de 4 meses. Finalmente, se empleó un algoritmo de Random Forest Regression para la predicción.

El principal inconveniente con esta aproximación fue el hecho de que hay muchos datos faltantes en el histórico, por lo que las series resultantes con la aproximación consistían principalmente en filas de ceros. Esto se vio reflejado en las salidas del modelo generado, cuyas predicciones eran en su mayoría un valor de 0.

5.2 VALIDACIÓN

Para el proceso de validación y selección del modelo se tomaron en cuenta 2 procesos importantes, la división del dataset y la selección de los mejores hiper parámetros. En el mundo de machine learning el proceso de selección de un modelo y el entrenamiento del mismo, es algo crucial, ya que si no se realiza de una manera adecuada el resultado final puede ser un modelo sobre ajustado o por el contrario un modelo con sesgo elevado, cualquiera de los dos resultados es algo que no se desea.

Para el entrenamiento de los diferentes modelos se tomó una parte del dataset como conjunto de datos de entrenamiento. Como se ha mencionado a lo largo del trabajo, se quiere un modelo que permita predecir las ventas para un mes posterior, por tanto, la división del dataset se realiza teniendo en cuenta

[4] Link repositorio Github: https://github.com/Alberto7526/MonografiaGithub

esto; en la figura 18 se muestra los primeros 30 meses (color azul celeste) utilizados para entrenamiento, luego de los meses 31 a 33 (color azul fuerte) se utilizan como entrada al modelo para predecir el mes 34, la división del dataset se hace pensando en utilizar datos nuevos (nunca antes vistos por el modelo), para de esta manera poder verificar su efectividad y robustez, al mismo tiempo de evidenciar si existe sobre ajuste.

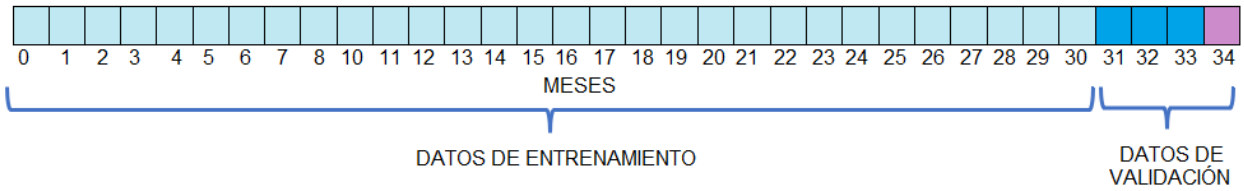

Figura 18. División del dataset en entrenamiento y validación.

Una vez dividido el dataset en el conjunto de datos de entrenamiento y el conjunto de datos de prueba, se procede a realizar el entrenamiento del modelo, sin antes realizar la selección de los mejores hiper parámetros; para ello, se utilizan las técnicas de validación cruzada y búsqueda de cuadrícula. La validación cruzada permite dividir el dataset de forma estratégica, permitiendo entrenar al modelo únicamente con una parte de los datos, y validando con un subconjunto de prueba(diferente al conjunto de validación mencionado anteriormente), esto ayuda a evitar el sobreajuste, ya que, el modelo no tendrá acceso a todos los datos, para posteriormente generar un promedio de los diferentes valores obtenidos por métrica con las diferentes divisiones y evaluar el rendimiento de un modelo en particular, la división efectuada se muestra en la figura 19

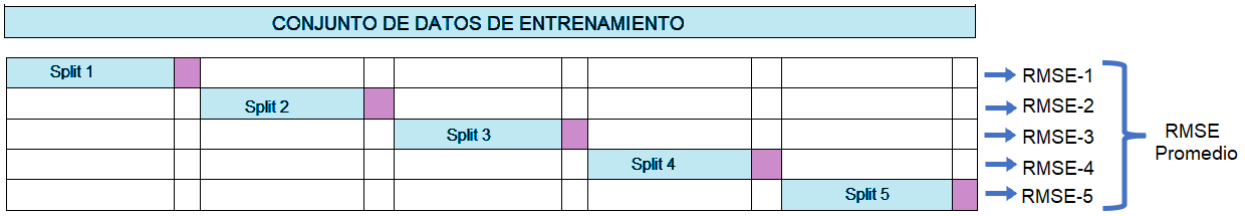

Figura 19. Validación cruzada.

Nótese como la división del dataset se ha realizado teniendo en cuenta la variable tiempo, para este caso los meses, la partición se realizó utilizando 4 meses para entrenar y un quinto mes para validar, sin embargo, estos valores pueden cambiar, puesto que se diseñó una función que genera la división con el el look back deseado.

La técnica de búsqueda de cuadrícula (o *grid search*) se implementó utilizando la librería sklearn, que permite ingresar los diferentes hiper parámetros que podría llegar a tener el modelo, para posteriormente entrenar el mismo con todas las posibles combinaciones. Además de esto, cada submodelo utiliza validación cruzada para hacer más eficiente el proceso.

5.3 ITERACIONES y EVOLUCIÓN

Una vez definido el baseline y la metodología de validación, lo siguiente consistió en buscar disminuir el error de predicción en el conjunto de datos de validación, para ello a lo largo del proceso de perfeccionamiento y analizando cada uno de los errores presentes, se tuvo en cuenta los siguientes consideraciones:

- ❖ Disminuir la cantidad de ceros presentes en el dataset:

Como se menciona en el baseline, el dataset presenta muchos valores en cero y por ende las predicciones generadas por el modelo presentaban muchos ceros y el error era grande. Para eliminar este problema se consideró a los productos con un promedio de ventas inferior a 5 ventas por mes, como un producto "nuevo" para el sistema y se excluyeron de los datos utilizados para entrenar el modelo, después de ello, se unió la tabla de "*items"* a la tabla de "*sales_train"* para tener la relación de las categorías de los ítems en el dataset, luego se agruparon los datos por tienda y categoría (no por producto, como en la primera iteración).
Con este procedimiento la cantidad de ceros se vio disminuida en un 50%, sin embargo, fue necesario realizar un análisis de las ventas por tienda y categoría tal como se observó en la figura 7 mostrando que algunas tiendas son consideradas como tiendas ocasionales y otras fueron cerradas, así que finalmente fueron eliminadas del dataset.

- ❖ Selección del look back para la predicción del modelo:

Una vez eliminados los valores de cero, la siguiente pregunta consiste en saber ¿cuántos meses necesito para predecir el siguiente mes? Utilizando un modelo basado en redes neuronales recurrentes, se realizaron diferentes iteraciones con diferentes look back, comparando la curva de aprendizaje de los diferentes modelos con 50 épocas.

- ❖ Ajuste de hyperparametros:

Finalmente, para optimizar y disminuir el error, se considera la búsqueda de cuadrícula y la validación cruzada mencionada anteriormente, permitiendo disminuir el error del modelo en gran medida.

5.4 HERRAMIENTAS

Para el presente proyecto se usó Python como lenguaje de programación, específicamente se utilizaron las librerías pandas y numpy para la preparación de los datos; sklearn y tensorflow para los modelos de aprendizaje y para validar; matplotlib y seaborn para la construcción de gráficos; Visual studio code y jupyter notebook para desarrollar los diferentes scripts.
Cabe mencionar que todo el proceso se realizó en una computadora personal con un procesador Intel core i5 de 9th generación, el cual cuenta con una memoria ram de 12 Gb.

6. RESULTADOS

6.1 MÉTRICAS

En la tabla 6 se presentan los resultados con la métrica de negocio implementada para los diferentes modelos

Modelo	**Costo de oportunidad**	**Costo de mantenimiento**
Red neuronal combinada	640.5	72
RANSAC	844.30	120.76
Random Forest Regression	1108.86	113.22

Tabla 6. Comparación entre los costos de oportunidad y mantenimiento para los modelos entrenados.

Se observa que el costo de mantenimiento para el Random Forest Regression es inferior al obtenido mediante el modelo RANSAC, sin embargo, el costo de oportunidad es mucho mayor. En ambos casos el costo es superior a los obtenidos mediante el uso de redes neuronales, por lo que consideramos que ambos modelos son menos aptos para este sistema.

6.2 EVALUACIÓN CUALITATIVA

Con el objetivo de seleccionar el *lookback* más adecuado, se realizó el entrenamiento del modelo con valores de 1, 2, 3, 6 y 12; cada modelo se entrenó 3 veces para los diferentes *lookbacks* y se promediaron los resultados, la curva de aprendizaje resultante para cada valor se muestra en la figura 20, nótese cómo los diferentes valores tiene un comportamiento similar, bastante aceptable y con tendencia al mismo punto. En la figura 21 se muestra el resultado aplicando la métrica de negocio en el conjunto de validación, obsérvese cómo el costo de mantenimiento casi no presenta variaciones, lo que permite inferir que el *lookback* no es un parámetro relevante cuando se trabaja con redes neuronales recurrentes, el costo de oportunidad presenta más variaciones, sin embargo, no son significativas. Para el modelo final se decide trabajar con un *look back* de 3, pero se cree que los resultados serían similares con *lookbacks* diferentes.

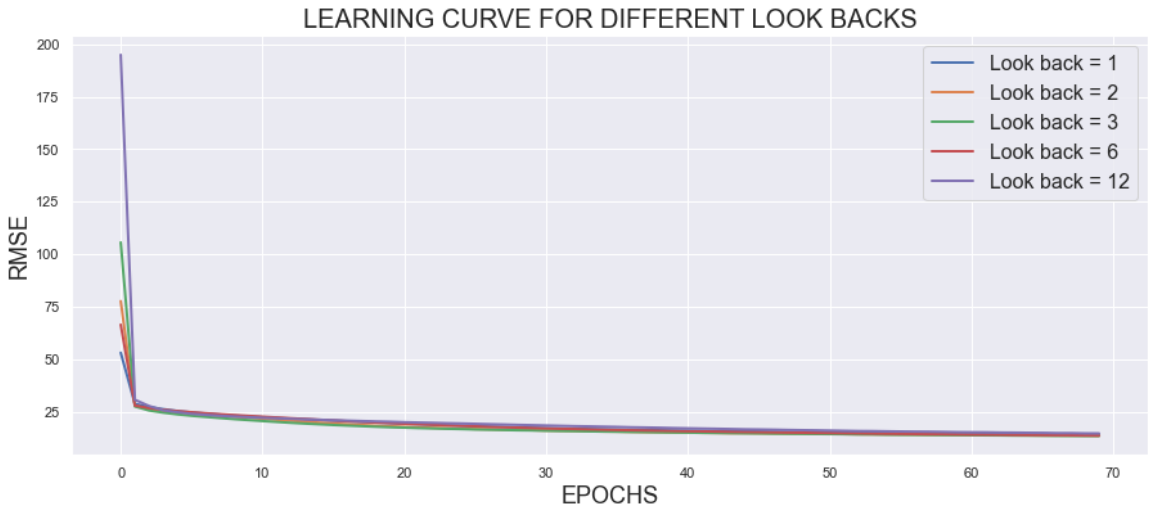

Figura 20. Curva de aprendizaje generado para una red neuronal recurrente para diferentes look backs.

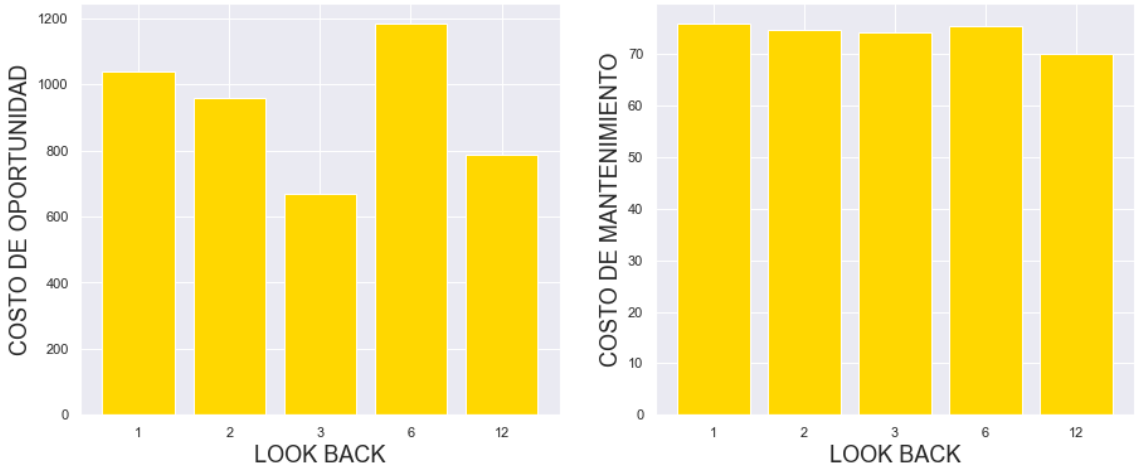

Figura 21. Evaluación del conjunto de datos de validación para diferentes look backs.

Modelo RANSAC:

Como resultado del Grid Search se obtuvo que los mejores valores eran un *max_trial* de 100 y un *residual_threshold* de 500. Se entrenó y se probó el modelo con estos valores y con un *look back* de 3, se obtuvieron los resultados presentados en la figura 22.

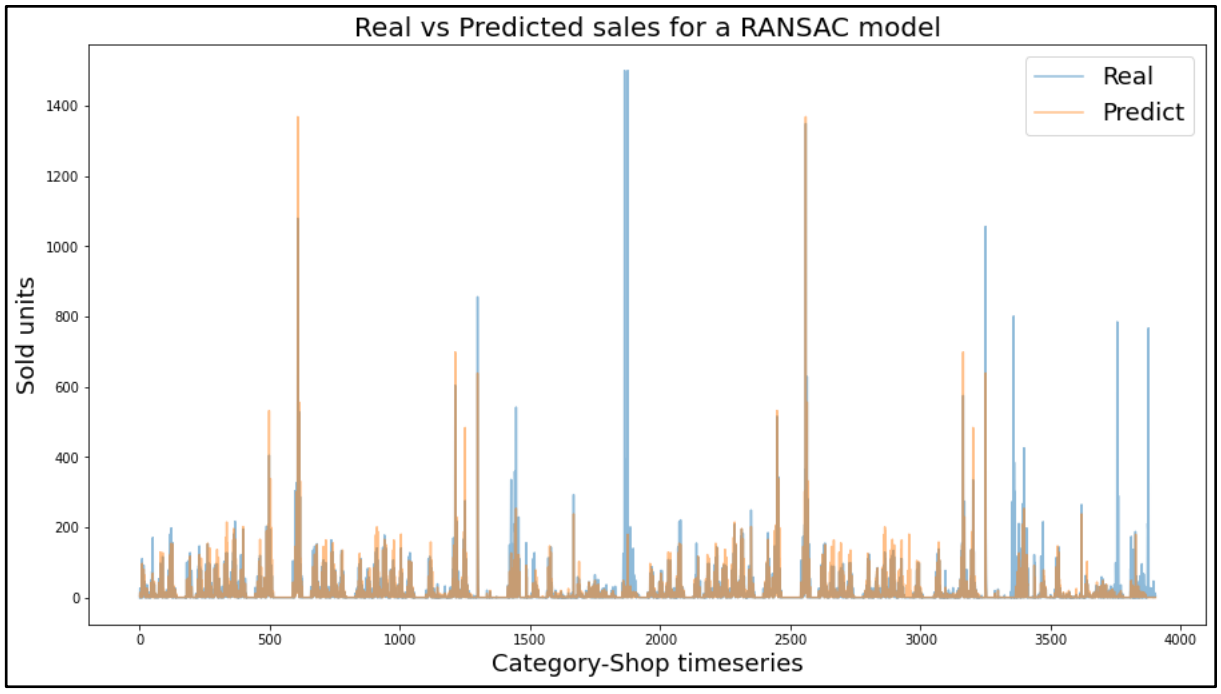

Figura 22. Valores de ventas reales y predichos mediante el modelo RANSAC para los datos de validación.

.

En el caso del modelo RANSAC se obtuvieron resultados muy similares con los diferentes *lookback*; en términos generales, se puede observar una mayor similitud en las predicciones respecto a los valores originales en los primeros valores, con enormes diferencias en los picos comparados con los valores reales. Es probable que estas diferencias se deban a que el modelo RANSAC considere como outliers los picos de ventas en los meses de diciembre y los excluya para el modelo final.

Random Forest Regression:

Para este caso, y a partir del Grid Search CV, se obtuvo una profundidad máxima de 20 y 150 estimadores. Los resultados del entrenamiento, comparados con los valores reales se presentan en la figura 23.

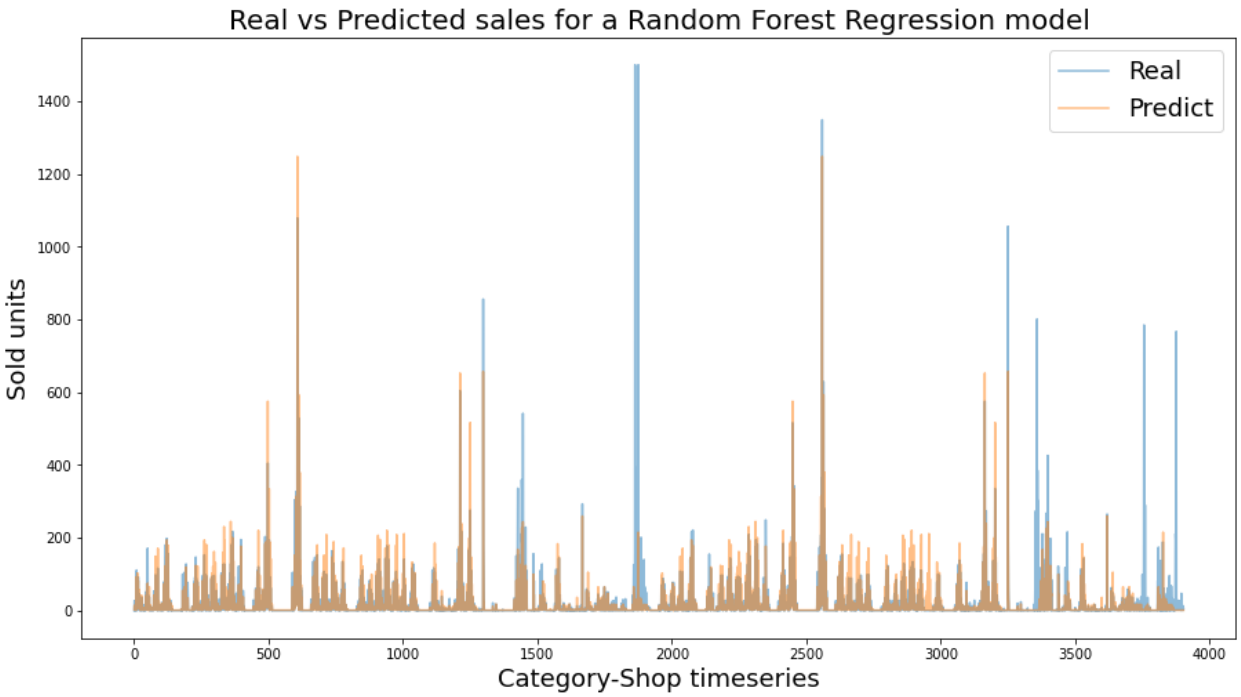

Figura 23. Valores de ventas reales y predichos mediante Random Forest Regression para los datos de validación.

Red neuronal recurrente:

Aplicando grid search para la búsqueda de hiper parámetros con la arquitectura de la red neuronal recurrente definida en la sección 4.3, se encontró que con diferentes números de neuronas para las diferentes capas, el RMSE no cambiaba demasiado, sin embargo, al variar la función de activación, el desempeño de la red neuronal si cambia considerablemente, el resultado se muestra en la figura 24.

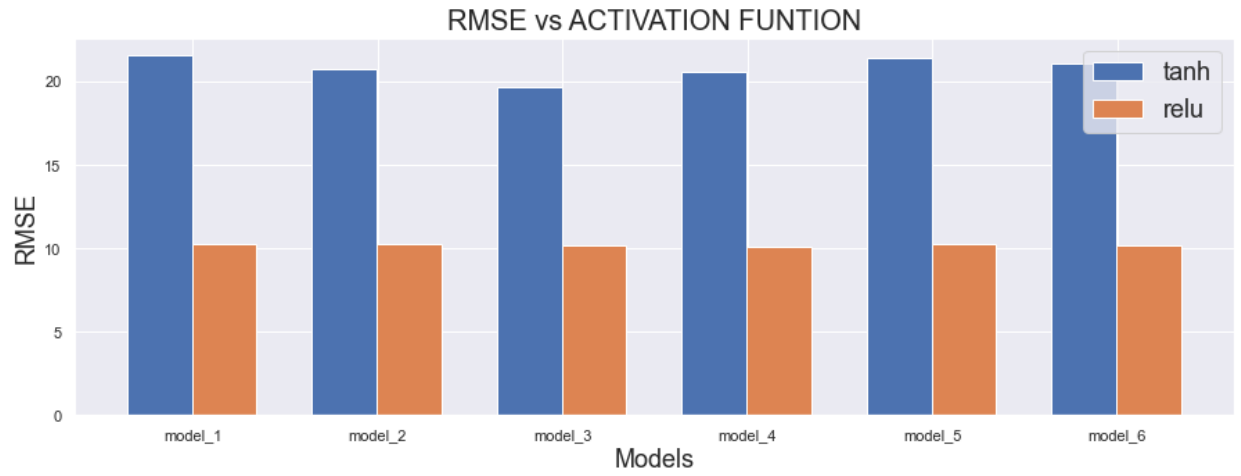

Figura 24. RMSE vs Función de activación para diferentes arquitecturas de red.

Otro hiper parámetro considerado fue la tasa de aprendizaje, debido a la naturaleza de los datos y a la forma como las redes neuronales calculan el error, en este caso para valores iguales o superiores a 1x10-3 en algunas ocasiones el RMSE presenta valores de *"Nan"*, la solución fue mantener la tasa de aprendizaje pequeña 1x10-4. La literatura también sugiere añadir parámetros de regularización a la función de pérdida.

6.3 CONSIDERACIONES DE PRODUCCIÓN

Las bases de datos de ventas en las tiendas deben estar permanentemente actualizadas para que el modelo siga funcionando en el tiempo y poder tener las predicciones de los meses posterior a la fecha más reciente, por otro lado, se debe hacer seguimiento al error entre las predicciones para los últimos meses y los valores reales una vez estos se tengan, de esta manera se podrán hacer ajustes de ser necesario.

Es importante también tener en cuenta que los valores usados para calcular el costo son asumidos, y en el momento de poner en producción se debe tener una mayor información respecto a estos costos relacionados al mantenimiento, así como los valores reales de ganancias de cada producto.

7. CONCLUSIONES

En el presente trabajo se desarrolló un modelo que permite predecir las ventas por categorías en cada tienda con un dataset dado. En el momento actual el poder predecir con mayor exactitud las ventas en periodos del futuro cercano es de gran importancia, ya que, permite hacer más eficientes los recursos disponibles, por ello cada vez más empresas y profesionales se interesan en esta área. Para elaborar un modelo capaz de predecir las ventas de forma correcta se deben tener en cuenta algunos aspectos importantes, la preparación de los datos y la eliminación de los outliers, permiten generar un modelo que captura mejor el comportamiento de las ventas. Inicialmente se buscó un modelo que permitiera predecir las ventas por cada producto, pero, teniendo en cuenta la gran cantidad de productos tenían muy poca información, se optó por un modelo simplificado del negocio, utilizando únicamente la categoría. La selección del modelo también juega un papel importante y va de la mano con los datos, ya que, para este caso y al tratarse de series temporales, es necesario dividir los datos teniendo en cuenta la variable tiempo, además, se crearon características nuevas, propias de cada venta, como los son el precio o la tienda a la cual pertenece cada producto, esto ayudó a los modelos a diferenciar y capturar el comportamiento de las ventas por cada tienda y categoría.

Una posible mejora que se puede hacer para generar modelos más confiables y que predigan mejor el comportamiento de las ventas, consiste en añadir mayor información relacionada con fechas y eventos realizados, ya que en el dataset actual existen muchos outliers, con esta nueva característica el modelo será capaz de relacionar mejor y en consecuencia podrá predecir aquellos picos que en el momento actual no puede.

8. BIBLIOGRAFÍA

Díaz Bou, P. (2020). *Desarrollo de Soporte de Redes Recurrentes en Plataforma de Entrenamiento*. España.

Hawkins, D. (2013). *Identification of Outliers*. Springer Netherlands.

Kumar, A. (2020, October 2). *RANSAC Regression Explained with Python Examples*. Data Analytics. Retrieved November 28, 2021, from https://vitalflux.com/ransac-regression-explained-with-python-examples/

Raschka, S. (Ed.). (2019). *Python Machine Learning* (V. Mirjalili, Trans.; segunda edición ed.). Marcombo. 978-84-267-2720-6