Universitat Oberta de Catalunya uoc.edu

Gestión del inventario en una empresa del sector farmacéutico mediante algoritmos de Machine Learning.

Jordi de Mas Jaumot
Máster universitario en Ciencia de Datos
Trabajo final de máster

Directora de TFM: **Dra. Lorena Polo Navarro**
Profesor responsable de la asignatura: **Dr. Jordi Conesa Caralt**

Fecha Entrega: 10.01.2021

Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada 3.0 España de Creative Commons

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Gestión del inventario en una empresa del sector farmacéutico mediante algoritmos de Machine Learning.*
Nombre del autor:	*Jordi de Mas Jaumot*
Nombre del consultor/a:	*Lorena Polo Navarro*
Nombre del PRA:	*Jordi Conesa Caralt*
Fecha de entrega (mm/aaaa):	01/2021
Titulación::	*Máster universitario en Ciencia de Datos*
Área del Trabajo Final:	*Trabajo Final de Máster*
Idioma del trabajo:	*Español*
Palabras clave	*Inventory level prediction, Machine Learning, Neural Networks, Deep Learning*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

El TFM versa sobre la gestión avanzada del inventario de forma que se pueda predecir el nivel de stock óptimo a partir de algoritmos de machine learning.

La predicción que se realice ha de ser capaz de asegurar que el nivel de stock permitirá dar un buen servicio al cliente, evitando roturas de stock y la consiguiente pérdida de ventas y, reduciendo en lo posible los costes derivados de la gestión del inventario.

Los datos tomados como base para este proyecto son del sector farmacéutico.

La predicción del stock óptimo se realiza en un único paso, evitando la necesidad de conocer de antemano la previsión de la demanda y sin tener que identificar la

distribución de la probabilidad de la misma. El nivel de inventario óptimo se calcula directamente analizando los datos disponibles: histórico de ventas, histórico de stock real, datos de promociones, calendario de festivos, etc.. Aunque en la industria farmacéutica, concretamente en los fabricantes, existen diversos tipos de stock (materia prima, WIP, productos semielaborados y productos terminados), los datos proporcionados para el proyecto permiten centrar el trabajo en los productos terminados.

Adicionalmente se tienen en cuenta diversos factores que influyen en el resultado de la predicción como: la estacionalidad o patrones semanales que muestran una mayor venta en determinados días de la semana.

El desarrollo se realiza en base a algoritmos de machine learning, analizando distintos tipos de ellos para decidir cuál proporciona una mayor precisión, con el fin de obtener un método alternativo de gestión del stock y asegurar un mejor servicio al cliente con el mínimo coste para la empresa.

Abstract:

Master’s dissertation is about advanced inventory management so that the optimal stock level can be predicted based on machine learning algorithms.

Prediction must be able to ensure that the level of stock will allow a good customer service, avoiding a stockout situation and the consequent loss of sales and reducing as far as possible the costs derived from inventory management.

Data taken as the basis for this project is from the pharmaceutical sector.

The prediction of the optimal stock is carried out in a single step, avoiding the need to know in advance the demand forecast and the distribution of its probability. The optimal inventory level is calculated directly by analysing the available data: sales

history, real stock history, promotion data, holiday calendar, etc. Although in the pharmaceutical industry, specifically in manufacturers, there are several types of stock (raw materials, WIP, semi-finished products and finished products), the data provided for the project allows to focus on the finished products.

Additionally, some factors with influence in the result of the prediction are considered, such as: seasonality or weekly patterns that show a higher sales level on certain days of the week.

The development is carried out based on machine learning algorithms, analysing different types of them to decide the one that provides better performance in order to obtain an alternative method of stock management and ensure better customer service with the minimum cost for the company.

A mi esposa Lourdes, a mi hija Anna y a mi hijo Jordi.

Con todo mi amor.

Reconocimientos

En primer lugar quiero agradecer a mi Directora de Proyecto, la Dra. Lorena Polo Navarro, por haber aceptado dirigir mi Trabajo Final de Máster. Por haberme dado su apoyo constante y sus sabios consejos y, por haberme mostrado el camino en los momentos de duda. Lorena, gracias por todo.

También quiero agradecer a todo el profesorado que ha impartido las diferentes materias de este Máster Universitario en Ciencia de Datos, desde la primera asignatura que me introdujo a este fascinante mundo de los datos y hasta esta última, el Trabajo Final de Máster, cuyo responsable es el Dr. Jordi Conesa Caralt. Las enseñanzas de todos ellos me han abierto un amplio campo de conocimiento y generado una gran curiosidad por profundizar más en este universo de datos. Gracias a todos.

Por último pero no por ello menos importante, a mi familia, que me ha brindado su apoyo incondicional, me ha dado soporte y ánimos en los momentos difíciles y ha tenido una paciencia infinita. Os quiero.

Índice

1. Introducción 1
1.1 Descripción y justificación del interés y la relevancia del trabajo 1
1.2 Explicación de la motivación personal 2
1.3 Definición de los objetivos 3
1.4 Descripción de la metodología empleada 3
1.5 Planificación del proyecto 5
2. Estado de la cuestión 6
2.1 Breve introducción conceptual a la gestión de stocks 6
2.2 Revisión de la literatura 9
2.2.1 Trabajos basados en métodos clásicos de gestión de inventario 9
2.2.2 Trabajos basados en métodos Machine Learning 11
3. Diseño e implementación 16
3.1 Recogida inicial de datos y almacenamiento 16
3.2 Preparación, limpieza y preprocesado de los datos 18
3.3 Análisis de los datos de ventas 25
3.3.1 Características del análisis 25
3.3.2 Correlación entre variables 30
3.3.3 Análisis componentes de la serie temporal 31
3.4 Preprocesado de los datos de stocks 32
3.4.1 Clusterización y eliminación de outliers 32
3.4.2 Eliminación automática de valores anómalos (outliers) 35
3.5 Análisis predictivo con algoritmos Machine Learning 38
3.5.1 Selección de algoritmos para el análisis predictivo 39
3.5.2 Red Neuronal Artificial multicapa con Embeddings 39
3.5.3 Gradient Boosting Regressor 45
3.5.4 Random Forest Regressor 48
3.5.5 Decision Tree Regressor 50
3.5.6 Light Gradient Boosting Machine Regressor 51
3.5.7 Extreme Gradient Boosting Regressor 52
3.5.8 Análisis comparativo de resultados 57
4. Conclusiones 59
4.1 Lecciones aprendidas 59

4.2 Logro de los objetivos planteados 59
4.3 Planificación y metodología 60
4.4 Líneas de trabajo futuro 61
5. Bibliografía 62

Lista de figuras

Figura 1 - Planificación del proyecto 5
Figura 2 - Fichero inicial de ventas con datos añadidos 19
Figura 3 - Fichero inicial de stocks con datos añadidos 20
Figura 4 - Comprobación de valores nulos en datos de ventas 22
Figura 5 - Estadísticos básicos de la variable udsVenta 22
Figura 6 - Outliers detectados en datos de ventas (vista clústers) 23
Figura 7 - Clústers identificados en datos de ventas 24
Figura 8 - Outliers detectados en serie temporal de ventas 24
Figura 9 - Ventas diarias por producto 25
Figura 10 - Evolución ventas diarias producto 1 26
Figura 11 - Ventas totales por producto 26
Figura 12 - Ventas totales por días de la semana 27
Figura 13 - Ventas totales por laborable / festivo 27
Figura 14 - Ventas totales por número de semana en el mes 28
Figura 15 - Ventas totales por mes 28
Figura 16 - Ventas totales con promoción vs. sin promoción 29
Figura 17 - Ventas totales por período (año/mes) 29
Figura 18 - Heatmap - Correlación entre variables de datos de ventas 30
Figura 19 - Descomposición de componentes de la serie temporal 31
Figura 20 - Patrón de estacionalidad 32
Figura 21 – Número de clústeres según el método del codo 33
Figura 22 - Observaciones de productos según stock, venta y promoción 34
Figura 23 - Contribución de los principales componentes 35
Figura 24 - Número de observaciones en cada clúster 35
Figura 25 - Outliers detectados en clústers 36
Figura 26 - Visualización de los clústeres 37
Figura 27 - Anomalías detectadas en la serie temporal 37
Figura 28 - Datos de entrenamiento 40
Figura 29 - Resumen estructura red neuronal 41
Figura 30 - Comparación datos test y predicción 42
Figura 31 - Muestra de resultados de la predicción 42
Figura 32 – NN - Valores reales vs. predicción - Producto 2 43
Figura 33 – NN - Valores reales vs. predicción - Producto 5 43
Figura 34 - Precisión estimada en Gradient Boosting Regressor 46

Figura 35 - Desviación train y test en Gradient Boosting 46
Figura 36 - GBR - Valores reales vs. predicción - Producto 4 47
Figura 37 - RF - Búsqueda de parámetros óptimos 48
Figura 38 - RF - Valores reales vs. predicción - Producto 4 49
Figura 39 - DT - Valores reales vs. predicción - Producto 4 50
Figura 40 - LGBM - Valores reales vs. predicción - Producto 4 52
Figura 41 - XGB - Valores para tuning hiperparámetros 54
Figura 42 - XGB - Comparación real vs. predicción todos los productos 54
Figura 43 - XGB - Valores reales vs. predicción - Producto 2 55
Figura 44 - XGB - Valores reales vs. predicción - Producto 4 55
Figura 45 - XGB - Valores reales vs. predicción - Producto 11 56
Figura 46 - XGB - Valores reales vs. predicción - Producto 13 56

Lista de tablas

Tabla 1 - Resumen de la literatura consultada y sus modelos de machine learning.... 15
Tabla 2 - Ficheros de datos iniciales proporcionados..... 17
Tabla 3 - Variables contenidas en los ficheros originales..... 18
Tabla 4 - Muestra resultados predicción red neuronal..... 44
Tabla 5 - Resultados totales predicción red neuronal..... 44
Tabla 6 - Resultados totales predicción Gradient Boosting Regressor..... 47
Tabla 7 - Resultados totales predicción Random Forest Regressor..... 49
Tabla 8 - DT - Resultados totales predicción Decision Tree Regressor 51
Tabla 9 - LGBM - Resultados totales predicción LGBM Regressor..... 52
Tabla 10 - XGB - Resultados totales predicción XGB Regressor 57
Tabla 11 - Comparativa resultados algoritmos ML..... 57

1. Introducción

1.1 Descripción y justificación del interés y la relevancia del trabajo

El presente trabajo de fin de máster trata un proyecto de Data Science en el ámbito industrial y más concretamente como subámbito, la gestión avanzada del inventario usando algoritmos de Machine learning.

El desarrollo y resultado del proyecto resulta interesante para las empresas, puesto que aporta una nueva forma de gestionar el stock que permite conseguir un ahorro de costes en la cadena de suministro, al tiempo que se asegura la continuidad de la producción y satisfacer la demanda de los clientes. Mantener el stock adecuado implica que no se dediquen al inventario más recursos monetarios que los necesarios, lo que redundará en una mayor rentabilidad.

En la actualidad, el stock necesario de cada producto se calcula en dos pasos. En primer lugar se realiza una previsión de la demanda y después, se calcula el stock óptimo necesario para, en función de esa demanda estimada, dar el mejor servicio a los clientes. La propuesta de este trabajo de fin de máster consiste en ofrecer una alternativa que permita realizar la estimación del stock óptimo en un sólo paso.

La predicción del stock óptimo se hace partiendo del histórico de stock y teniendo en cuenta una serie de variables como: venta, promociones, factor estacional, etc. De esta manera, la previsión de la demanda deja de ser necesaria.

Diversos autores han realizado investigaciones sobre nuevos métodos para gestionar el inventario, como Šustrová (2016) que ha creado un modelo de red neuronal artificial (ANN) para predecir el tamaño de pedido de materiales que se debe realizar, de forma que se mejora el ciclo de pedidos en la empresa. O He (2013) que propone un nuevo algoritmo de retropropagación que ha usado

en un conjunto de datos de una compañía de componentes de automoción para predecir el nivel de inventario, aunque el documento indica que todavía existen algunos problemas que deben ser resueltos, como decidir el número de capas ocultas y la optimización de la estructura entera de la red neuronal.

También hay autores que han usado otro tipo de algoritmos de machine learning, como Praveen, Prateek, Pradyumna, Pragathi y Madhuri (2020) que proponen un modelo de regresión XGBoost que usa árboles de decisión, para realizar predicciones de la demanda, ya que cuando los datos son desestructurados, las redes neuronales superan a otros algoritmos predictivos, pero con datos estructurados y tabulados los algoritmos basados en árboles de decisión se consideran más adecuados.

En este trabajo de fin de máster se busca el aspecto innovador que permita realizar la predicción en un sólo paso y para ello se analizan diferentes algoritmos de machine learning con el fin de hallar aquel que nos proporcione una mayor precisión en la predicción del stock futuro y al mismo tiempo permita reducir los costes derivados de la gestión del inventario.

1.2 Explicación de la motivación personal

Tengo mucho interés en hacer el TFM sobre este tema ya que conozco el caso de una empresa con problemas en la gestión de su inventario, a la que le cuesta mucho lograr una reducción de sus stocks de baja rotación. Además, esa empresa también tiene dificultades a la hora de hacer una previsión de la demanda con un cierto rigor. Por esta razón creo que la posibilidad de hacer un trabajo de fin de máster consistente en lograr una predicción del stock óptimo de un modo directo (obviando el cálculo de la previsión de la demanda), puede ser de gran ayuda para las empresas que se encuentren en ese mismo caso.

Considero que el uso de la inteligencia artificial y la posibilidad de lograr modelos predictivos eficientes, permitirá una gran mejora en la gestión las empresas y contribuirá a su éxito.

1.3 Definición de los objetivos

El objetivo principal de este trabajo de fin de máster es construir un modelo predictivo basado en algoritmos de machine learning que permita pronosticar el stock futuro óptimo de cada material en un solo paso, a fin de que los costes derivados del stock puedan reducirse.

Del mismo modo se establecen una serie de objetivos secundarios de carácter más específico:

- Revisión de bibliografía para comprobar los resultados obtenidos por otros autores sobre la misma problemática.
- Preprocesamiento de los datos
    - Tratamiento de valores nulos
    - Normalización de datos
    - Evaluación de outliers
- Realizar un clustering de referencias con el fin de encontrar artículos con el mismo comportamiento de demanda, de forma que se pueda identificar un modelo para cada cluster.
- Analizar el tipo de algoritmo que mayor precisión proporciona.
- Comparar resultados obtenidos por el algoritmo machine learning vs. el resultado del stock actual de la empresa, tanto en términos de precisión como en términos económicos.

1.4 Descripción de la metodología empleada

Por lo que se refiere a herramientas software, la metodología empleada en este trabajo se basa en el tratamiento del conjunto de datos mediante la programación en Python. Para comprobar el desarrollo de las diferentes etapas de tratamiento de los datos (carga, preprocesamiento, modelado, etc) se utiliza un libro jupyter (.ipynb) de Python. También se utiliza Excel en la preparación inicial de los ficheros.

La redacción de la memoria se realiza con Microsoft Word, la presentación con Microsoft PowerPoint y para la planificación del proyecto se utiliza GanttProject.

Se trata de un proyecto nuevo construido desde cero por lo que la utilización de las mencionadas herramientas es considerada la opción más adecuada.

Por otro lado, en cuanto a las diferentes etapas que se deben cubrir en este proyecto, se distinguen las siguientes:

- Carga de datos: originalmente recogidos en ficheros .csv o Excel, y que se convierten en dataframes para su uso en el libro jupyter mediante el lenguaje Python.

- Preprocesamiento de los datos: de forma que permita dejarlos limpios y con la mejor calidad para poder realizar las tareas de análisis y la construcción de modelos. En esta etapa se realiza el tratamiento de los valores nulos, la normalización de datos, la evaluación de outliers, etc.

- Clusterización: se efectúa un proceso de clustering de los datos con el fin de determinar referencias con un mismo comportamiento de demanda y así poder identificar un modelo más apropiado para cada cluster.

- Construcción de modelos alternativos: se realiza la construcción de modelos alternativos y al análisis de los resultados obtenidos para poder seleccionar el que proporcione una mayor precisión.

- Análisis de resultados del algoritmo seleccionado: los resultados obtenidos por el algoritmo de machine learning seleccionado se comparan con el resultado del stock actual de la empresa, tanto en términos de precisión como en términos económicos. A partir de ese punto se pueden realizar una revisión del modelo de forma que se pueda afinar más en el resultado obtenido.

- Conclusiones: una vez completado todo el proceso se extraen las conclusiones definitivas.

- Redacción de la memoria: se realiza durante toda la extensión del trabajo de fin de máster, desde su inicio, con la explicación de la selección del proyecto, hasta el final, una vez realizado todo el trabajo de análisis de los datos y la obtención de resultados.

1.5 Planificación del proyecto

El proyecto se va a realizar básicamente en el ordenador en el que se ha instalado la plataforma Anaconda que permite trabajar con el lenguaje Python y el uso de los Jupyter notebooks.

La planificación temporal del TFM es la siguiente:

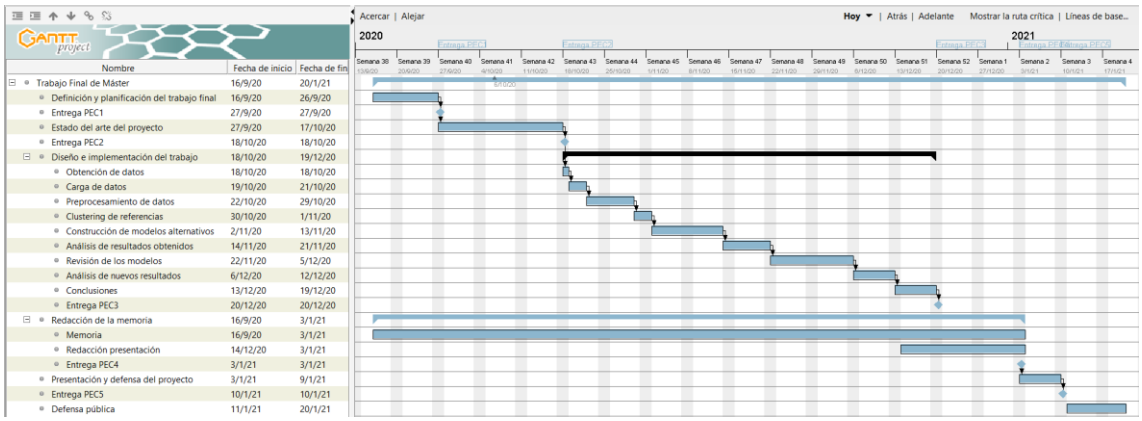

Figura 1 - Planificación del proyecto

2. Estado de la cuestión

El objetivo de este capítulo es hacer una revisión de los artículos publicados referentes al objeto de la presente investigación.

En la actualidad las empresas trabajan en un entorno VUCA: volátil, incierto, complejo y ambiguo. La competencia es feroz y se desarrolla en un escenario donde los consumidores demandan productos de una mayor calidad, los márgenes se reducen y se requiere de mucha más rapidez en la entrega. Todo ello conlleva que las empresas deban abordar un proceso de optimización de sus negocios buscando un mayor rendimiento a sus inversiones.

Las fórmulas adoptadas son variadas, pero una de las que se considera más importante y con mayor impacto en la rentabilidad es la gestión eficiente del inventario. El objetivo es asegurar un buen servicio al cliente sin la necesidad de disponer de un stock demasiado grande que provoque gastos excesivos. Es decir, se busca un equilibrio entre ambos factores.

Sobre este tema se ha escrito largo y profusamente, siendo de especial interés las publicaciones científicas que abordan diversas soluciones desde diferentes puntos de vista.

2.1 Breve introducción conceptual a la gestión de stocks

Definimos el inventario o stock como la cantidad de materiales de los que se dispone para permitir la continuidad del proceso productivo y para satisfacer la demanda de los clientes.

En la gestión empresarial se intenta que el inventario sea lo más pequeño posible de forma que requiera una inversión mínima y al mismo tiempo permita ofrecer un buen servicio al cliente, lo que requiere que exista un equilibrio entre ambos aspectos.

En la gestión de inventario se deben tener en cuenta diferentes tipos de costes:

- Costes de almacenamiento
    - Amortización del almacén o alquiler del mismo
    - Seguros
    - Equipos de manipulación de mercancías
    - Impuestos
    - Energía
    - Mermas
    - Obsolescencia
    - Etc.

- Costes de emisión de pedidos
    - Costes administrativos
    - Mano de obra
    - Teléfono
    - Material informático
    - Etc.

- Coste de adquisición (compra o fabricación)

- Coste de ruptura de stock
    - Externos (ventas no atendidas, pérdida de imagen, etc.)
    - Internos (parada de equipos, horas extra, subcontratación, etc.)

El comportamiento de los costes es variable según la estrategia que la compañía decida aplicar al inventario. Si se desea disponer de un amplio inventario que permita satisfacer siempre al cliente, los costes de almacenamiento aumentarán. Si en cambio, se quiere mantener muy pequeño, los costes de emisión de pedidos serán mayores y posiblemente se produzcan roturas de stock.

Otro factor que también se debe tener en cuenta es el plazo de suministro necesario que incluye:

- Confección del pedido
- Transporte
- Preparación

- Espera
- Inspección
- Etc.

El tiempo necesario para cada uno de ellos determina la antelación con la que hay que hacer el pedido de reposición para asegurar que la mercancía esté disponible y poder dar servicio al cliente.

La combinación de ambos factores, minimizando costes y asegurando la disponibilidad se convierte en la principal actividad para la gestión del inventario.

En cuanto a las estrategias para la gestión del inventario, García (2011) hace la siguiente distinción en función de si son para una demanda continua o para una demanda discreta:

- Modelos básicos en demanda continua
    - Gestión por punto de pedido (se compra cuando se alcanza un determinado nivel de stock)
    - Gestión por aprovisionamiento periódico (se compra cada cierto tiempo prefijado)
    - Modelo de múltiples puntos de pedido (para niveles de variabilidad elevados en la previsión de la demanda)
    - Modelo de las dos cajas (se depositan dos cajas en el almacén, cuando se consume una se lanza el pedido para comprar otra)
    - Modelo (S-1, S) o de reaprovisionamiento continuo (cuando se produce una venta se repone la cantidad vendida)

- Modelos básicos en demanda discreta
    - Por períodos
    - Modelo de programación matemática (optimizando y generando el plan más barato)
    - Modelo Wagner-Within de grafos (solo se pide cuando el stock queda a cero. La cantidad pedida debe permitir esa circunstancia)

2.2 Revisión de la literatura

2.2.1 Trabajos basados en métodos clásicos de gestión de inventario

A lo largo de los años se han aplicado diferentes metodologías para optimizar la gestión de los stocks, sea mediante una clasificación de las referencias según su importancia o haciendo análisis de los datos históricos que permita llegar a una predicción precisa del comportamiento del inventario en el futuro.

Los métodos clásicos de gestión del inventario basan su trabajo en diferentes aproximaciones con el fin de buscar su optimización. Los más habituales son:

- la basada en la definición de un stock de seguridad, en la que simplemente se añade una cierta cantidad a la demanda prevista para conseguir el nivel de inventario necesario.

- La que calcula el inventario óptimo a partir de una estimación de la incertidumbre de la demanda y el aprovisionamiento, teniendo en cuenta los costes de mantenimiento del inventario y de la rotura de stock.

- La que prioriza el nivel de servicio al cliente de forma que siempre debe haber stock suficiente para atender la demanda.

A finales del siglo pasado se consideraba como un fenómeno natural que el rendimiento de una persona ocupado en una tarea repetitiva mejoraba con el tiempo y la idea se asociaba a la denominada curva de aprendizaje. Dado que se creía que el fenómeno del aprendizaje tenía implicaciones económicas y de toma de decisiones en un sistema de gestión de inventarios, Peterson y Silver (1979), se realizó un estudio que investigaba el efecto del aprendizaje tanto en la cantidad óptima de producción como en el mínimo coste total de inventario Salameh, Abdul-Malak y Jaber (1993). En este estudio se presentó un modelo modificado de inventario de producción que incorporaba el efecto del

aprendizaje que se puede conseguir durante la producción en el sistema de inventario de costo total por unidad de tiempo y cantidad óptima de producción.

Uno de los primeros sistemas aplicados era la clasificación ABC de los ítems a partir del coste unitario de cada referencia y de su demanda anual. A pesar de que han aparecido otros métodos, su uso es defendido por Ravinder y Misra (2014) que recomiendan que los libros de texto incorporen sus hallazgos y métodos en las discusiones sobre el tema. En su trabajo presentan un análisis de la situación de la investigación basada en ABC multicriterio y hacen referencia a los documentos encontrados al realizar una revisión de la literatura sobre el tema.

Otro sistema clásico muy común es el establecimiento de un stock de seguridad que permita asegurar el servicio al cliente, aún a costa de incurrir en mayores costes. Nieto (2014) realizó un trabajo que buscaba encontrar el stock de seguridad óptimo que permitiera disminuir los costes de mantenimiento del inventario. Para ello, realiza un análisis entre tres métodos:

- disponer de 15 días de existencias de todos sus productos
- pronosticar el stock a partir de un análisis del patrón de la demanda
- hallar el stock adecuado tomando como base el promedio ponderado y la desviación de la serie de tiempo

Los tres métodos se analizaron midiendo los costes de almacenamiento y el nivel de servicio para cada producto estudiado.

El estudio se realizó sólo sobre cinco referencias, con lo que los resultados no podían ser concluyentes y se recomendaba realizar el mismo ejercicio sobre un número considerablemente mayor de artículos.

El sistema ABC se ha seguido utilizando en diversos estudios para las fases iniciales de diversos trabajos de investigación, aunque éstos incluyeran en sus fases posteriores, procesos mediante algoritmos de Machine Learning. Se pueden ver algunos casos a continuación.

Partiendo de que el sistema ABC se basaba en el coste unitario y en la demanda anual, Puente, Priore, Pino y de la Fuente (2003) propusieron un nuevo sistema que pudiera manejar un mayor número de atributos de entrada para clasificar los inventarios de una empresa farmacéutica. Los atributos

pasaron a ser cinco: coste unitario, demanda anual, coste del pedido, plazo de entrega y criticidad del proceso, los cuales conformaban los datos que alimentaban la red neuronal artificial para la clasificación ABC multiatributo del inventario de la compañía. La arquitectura de la ANN finalizaba en una capa de salida con tres neuronas, una para cada tipo de stock (A, B o C), que quedaban activas según el material fuera clasificado. Se obtuvieron elevados porcentajes de acierto.

Por su parte, Cardona, Orejuela y Rojas (2018) lo aplicaron como primera fase de la estrategia que diseñaron para gestionar el inventario de materias primas en una empresa del sector de alimentos concentrados. En su trabajo toman como base la variabilidad de la demanda y el nivel de servicio deseado para definir el stock de seguridad de cada ítem.

2.2.2 Trabajos basados en métodos Machine Learning

Las redes neuronales se han convertido en uno de los métodos más ampliamente utilizados para lograr predecir el comportamiento del inventario futuro. Khalilzadeh (2008), Quijaite y Apaza (2014), Shoujing Zhang, Qin, Hu, Qing Zhang, Dong y Zhao (2020), Praveen, Farnaz y Hatim (2019), He (2013) y Šustrová (2016), entre otros.

El modelo de NN con backpropagation (BPNN) es uno de los más ampliamente usados por su simple estructura y su habilidad para aprender. La principal característica del algoritmo BP es la propagación hacia adelante de los datos y la propagación hacia atrás de los errores, de forma que la red vuelve a aprender del error cometido.

El mismo tipo de red neuronal con backpropagation fue utilizada por He (2013) pero añadiendo una compensación de error (error offset) al algoritmo de BP que mejoraba en precisión de la predicción a otros modelos. Además, la arquitectura diseñada era algo diferente y consistía en tres capas: capa de entrada, capa oculta y capa de salida, con 9 parámetros de entrada y uno de salida. En su investigación buscaba predecir el nivel de inventario teniendo en cuenta no sólo la demanda sino también los costes asociados a la gestión de

inventario. Su trabajo partía de los siguientes datos: coste de almacenamiento, coste de pedido, coste de escasez del producto (rotura de stock), coste de transporte, nivel de demanda, nivel de suministro, cantidad de sustitutos, tiempo de espera y nivel de servicio. En esta investigación resulta interesante la aplicación de artículos sustitutos cuando no hay stock de algún material.

Šustrová (2016) buscaba un modelo de red neuronal para determinar la cantidad óptima de pedido y optimizar así la cantidad de stock, partiendo tan sólo de seis datos para cada artículo: demanda actual mensual, demanda próximos tres meses, demanda tres meses siguientes a los tres próximos, nivel actual de inventario, precio de compra unitario y coste de transporte unitario. Para ello creó y probó varios tipos de red neuronal para averiguar qué modelo de arquitectura era más adecuado para realizar la predicción. Fueron 16 modelos de red neuronal con función de aprendizaje TRAINGDX o TRAINSCG, con función de transferencia TANSIG o LOGSIG y en los que se varió el número de neuronas de la capa oculta. Para escoger la más adecuada se tomó la que tenía un menor MSE (Mean Squared Error) y un coeficiente de determinación R^2 más cercano a 1. Finalmente se observó que el modelo de ANN Feed-forward backpropagation, con función de entrenamiento TRAINGDX, función de transferencia TANSIG y una arquitectura 6-8-1 era la más adecuada para la predicción.

En cambio Barbosa, Pestana y Goliatt (2017) intentan mejorar el nivel de servicio al cliente identificando los materiales con mayor probabilidad de no poder servirse (rotura de stock). Su intención es identificarlos en un período corto de tiempo, posibilitando que la empresa pueda reaccionar. En este estudio se introduce el concepto de “Imbalanced learning” que trata el problema de desequilibrio existente cuando en un conjunto de datos existen muy pocas instancias de una clase comparado con las del resto. Ejemplos de estos casos se podrían encontrar en proyectos para la detección del fraude o en diagnosis médica.

Por su parte, Shoujing Zhang et al. (2020) introducen el concepto del grado de importancia de las piezas al realizar un estudio de la gestión del inventario de las piezas de repuesto en una empresa de mantenimiento de camiones pesados y de esta forma, decidir las piezas más críticas que debían estar en el inventario. Para ello combinaron una BPNN con un algoritmo de clustering

mejorado. Usaron PCA para reducir la dimensionalidad de los datos y K-Means como método de clustering. La clasificación se realizó en base a los datos históricos de mantenimiento y los datos de inventario. Al utilizar Machine Learning se consiguió evitar el impacto subjetivo (experiencia de los inventory managers) durante el proceso de evaluación de la importancia de cada pieza.

La utilización de clustering puede ser interesante para agrupar artículos cuya demanda tenga el mismo comportamiento y esto permitiría identificar un modelo para cluster.

Per ha habido también modelos no basados en redes neuronales como Praveen K.B. et al. (2020) que usaron el modelo de regresión XGBoost (Extreme Gradient Boosting) para predicciones de la demanda. XGBoost es un algoritmo de Machine Learning que usa árboles de decisión y entorno Gradient Boosting. Decidieron usarlo porque partían de información muy estructurada y tabulada y para ese tipo de información estructurada se considera que los árboles de decisión dan mejores resultados. Con este método consiguieron reducir el stock y el capital dedicado a ello, en pequeñas y medianas empresas. Para realizar el entrenamiento del modelo usaron early-stopping-runs con el fin de evitar el overfitting. La ejecución del modelo proporcionaba un informe con la predicción de la demanda de las referencias para las siguientes dos semanas.

Otro aspecto a tener en cuenta es la estacionalidad ya que la demanda de ciertas referencias no se comportará de la misma forma durante todo el año. Del mismo modo, se debe prestar atención a los patrones semanales que permiten ver si existe variación en la demanda en diferentes días de la semana. Estos aspectos deberán ser estudiados para decidir si puede tener sentido su aplicación en el proyecto actual.

A continuación se muestra una tabla resumen de los modelos Machine Learning referenciados.

Autor/Año	Variable objetivo	Features utilizadas	Técnicas usadas	Técnicas con mejores resultados
Puente, Priore, Pino y de la Fuente (2003)	Clasificación ABC de los materiales	Coste unitario Demanda anual Coste de pedido Plazo de entrega Criticidad del proceso	ANN para la clasificación ABC multiatributo para la gestión de inventarios.	ANN
Khalilzadeh (2008)	Nivel de inventario	Datos de los 36 últimos meses de cada artículo	Red neuronal híbrida para la predicción a partir de series temporales en base a los datos de los 36 últimos meses de cada artículo.	Juntando los datos de los artículos y los de los artículos con similar comportamiento de ventas o con alta correlación.
He (2013)	Nivel de inventario	Coste de almacenamiento Coste de pedido Coste de escasez Coste de transporte Nivel de demanda Nivel de suministro Cantidad de sustitutos Tiempo de espera Nivel de servicio	BPNN mejorada con error offset añadido	El algoritmo mejorado supera significativamente el algoritmo estándar y otros modelos BP mejorados, tanto en tasa de convergencia como en precisión de la predicción.
Quijaite y Apaza (2014)	Coste total anual	Cantidad a pedir Punto de reorden Inventario inicial Tiempo de entrega Costos por orden Costo de almacenamiento Costo por venta perdida Semana Inventario virtual Inventario actual físico Orden recibida Unidades recibidas Demanda	Uso de OptQuest, software comercial diseñado para optimizar sistemas complejos. Dicho software funciona en base a una combinación de estrategias: búsqueda dispersa, algoritmos genéticos, búsqueda tabú y redes neuronales.	Con la técnica usada se logró reducir el costo total en un 37% respecto a su valor inicial.

Autor/Año	Variable objetivo	Features utilizadas	Técnicas usadas	Técnicas con mejores resultados
Šustrová (2016)	Cantidad óptima de pedido	Demanda actual mensual Demanda prox. 3 meses Demanda 3 meses siguientes (3+3) Nivel (valor) actual de inventario Precio de compra unitario Coste de transporte unitario	Se probaron 16 NN con función de aprendizaje TRAINGDX o TRAINSCG y con función de transferencia TANSIG o LOGSIG. El número de neuronas en la capa oculta también se varió.	Una red neuronal con función de entrenamiento TRAINGDX, función de transferencia TANSIG y una arquitectura 6-8-1 es la más adecuada para la predicción mediante ANN.
Barbosa, Pestana y Goliatt (2017)	Identificar materiales con mayor probabilidad de rotura		Algoritmo ML de clasificación. Se probaron diversos métodos de ensemble: Random Forest (FOREST), Gradient Tree Boosting (GBOOST), Blagging (BLAG).	Gradient Tree Boosting (GBOOST).
Praveen et al. (2020)	Predicción de la demanda		Uso del modelo de regresión XGBoost (Extreme Gradient Boosting) para predicciones de la demanda.	Se reduce el exceso de stock y la escasez del mismo ya que los stocks son pedidos en base a la demanda.
Shoujing Zhang et al. (2020)	Grado de importancia de la pieza de repuesto		Se usa un algoritmo de clustering mejorado (Mahalanobis distance clustering algorithm) combinado con una BP NN. Antes del algoritmo mejorado se usa PCA (Principal Component Analysis) para realizar una reducción de dimensionalidad de datos y K-Means como método de clustering.	Se obtienen resultados precisos sin factores subjetivos que afecten a la evaluación de la importancia de la pieza de repuesto.

Tabla 1 - Resumen de la literatura consultada y sus modelos de machine learning

3. Diseño e implementación

Se ha seguido el proceso habitual y con las etapas propias de un proyecto de ciencia de datos. No todas las fases definidas se han sucedido de forma secuencial sino que en algún momento se ha tenido que volver a reprocesar datos, usando en algunos casos nuevas herramientas que permitieran pulir tanto los conjuntos de datos en sí como el análisis que de ellos se debía abordar.

Asimismo se ha tratado la problemática del proyecto utilizando diversos algoritmos de machine learning, de forma que se pudieran observar diferentes modos de tratar el caso y comprobar cuál de ellos resulta más eficiente.

Seguidamente se describen los diferentes pasos realizados para la implementación del proyecto.

3.1 Recogida inicial de datos y almacenamiento

Los datos iniciales han sido proporcionados por una empresa del sector farmacéutico que los ha cedido a la directora de este TFM para poder utilizarlos en él.

El conjunto de datos en su totalidad ha sido anonimizado allí donde ha sido necesario, eliminando descripciones de productos y cualquier otra información que pudiera servir para averiguar la empresa de donde proceden. Del mismo modo, a las cifras correspondientes a la cantidad de stock, a las unidades vendidas y a la previsión de ventas, se les ha aplicado un coeficiente multiplicador de forma que también ayudara a garantizar la confidencialidad de los datos.

Los ficheros de datos proporcionados para este proyecto son los siguientes:

Fichero	Descripción
01_TablaVentas_Farma	Unidades vendidas de cada producto. Período: 29/06/2015 al 30/10/2020. Total registros: 98.590 Sin datos vacíos.
02_TablaPromos	Promociones de venta de los productos. Período: 09/12/2015 al 31/12/2020. Existe alguna promoción con fecha fin en 31/12/2100 que podría indicar que es una promoción permanente. Total registros: 5.979 Contiene datos NULL y vacíos.
03_TablaStock_Farma	Unidades en stock de los productos. Período: 05/06/2019 al 04/11/2020 Total registros: 23.599
04_PrevisionEmpresa	Previsión de ventas de los productos. Período: 30/01/2019 al 23/08/2020 Total registros: 28.995
05_Festivos	Calendario de festivos anual. Período: 2016 a 2019 Total registros: 48

Tabla 2 - Ficheros de datos iniciales proporcionados

De todas maneras estos ficheros no componen el conjunto de datos definitivo ya que se ha tenido que trabajar sobre ellos completando la información y creando algún nuevo fichero que permitiera trabajar en el proyecto con mayor comodidad y sentido.

Las variables contenidas en cada uno de los ficheros se describen a continuación:

Fichero	Variables
01_TablaVentas_Farma	Fecha (AAAAMMDD) Producto Unidades vendidas
02_TablaPromos	Identificador de la promoción Fecha de inicio de la promoción (DD/MM/AAAA) Fecha de fin de la promoción (DD/MM/AAAA) Semana de inicio Semana de fin Producto Precio tarifa Precio en promoción Cantidad Tipo de promoción
03_TablaStock_Farma	Fecha (DD/MM/AAAA) Producto Unidades en stock
04_PrevisionEmpresa	Producto Fecha (DD/MM/AAAA) Previsión de ventas
05_Festivos	Descripción del festivo Día de la semana (Lunes, Martes, Miércoles,…) Fecha (DD/MM/AAAA) Texto (constante “FESTIVO”)

Tabla 3 - Variables contenidas en los ficheros originales

3.2 Preparación, limpieza y preprocesado de los datos

El conjunto de datos original consta de cinco ficheros, cuatro de ellos en formato csv y uno, el correspondiente a los festivos, en formato Excel. Para el tratamiento inicial de preparación de los ficheros se usará Excel y una vez estén listos para su uso en un libro ipynb (Jupyter notebook) se transformarán a formato csv.

Se ha observado que en el conjunto de datos proporcionado existe una disparidad considerable en cuanto a número de registros y períodos de las series temporales, habiendo una mayor cantidad de información en los datos

de ventas que en los referidos a stock de productos. Esto se debe a que la empresa empezó a guardar mucho antes la información relativa a ventas y al cabo de unos años empezó a guardar un histórico de stocks.

Ante esta situación se decide abordar el proyecto aplicando la siguiente estrategia:

- Se utilizarán los datos de ventas (serie temporal de 5 años) para analizar el comportamiento de venta de los productos (tendencia, estacionalidad, etc.).
- Posteriormente se trabajará en base a los datos de stock teniendo en cuenta también su relación con el comportamiento de venta de los productos.

La idea es realizar la predicción del inventario convirtiendo el proyecto en un caso de aprendizaje supervisado. De esta manera se deberán quitar los últimos datos de stock para poder predecirlos a partir del resto de datos de fechas anteriores (stock, ventas, promociones, festivos, etc.).

En el caso de desear hacer predicciones más allá de los datos disponibles, es decir, más a futuro, los datos de ventas de cada producto se sustituirían por los datos de previsión de ventas que confecciona la empresa periódicamente. El método que usa actualmente es muy sencillo y basado en el promedio de ventas de los últimos 5 días teniendo en cuenta el día de la semana, es decir, la previsión de un lunes es el promedio de los últimos 5 lunes y así sucesivamente para cada día de la semana.

Lo primero que se ha realizado es juntar la información necesaria para el análisis de las ventas en un único fichero tomando datos del resto de ficheros proporcionados y añadiendo algunos nuevos que pueden ayudar a entender el comportamiento de las ventas de los productos. Este primer paso se ha efectuado utilizando Excel. El resultado es el siguiente:

fecha	fecha_f	producto	f+p	udsVenta	festivo	diaSemana	udsStock	udsPrevision	promo	idPromo	precioTarifa	precioPromo	cantidadPromo	tipoPromo	difPrecio	numSemEnMes
20150629	29/06/2015	1	421841	3347	0	1	4536	0	0	0	0	0	0		0	5
20150630	30/06/2015	1	421851	4303	0	2	4536	0	0	0	0	0	0		0	5
20150701	01/07/2015	1	421861	14820	0	3	4536	0	0	0	0	0	0		0	1
20150702	02/07/2015	1	421871	4303	0	4	4536	0	0	0	0	0	0		0	1
20150703	03/07/2015	1	421881	8127	0	5	4536	0	0	0	0	0	0		0	1

Figura 2 - Fichero inicial de ventas con datos añadidos

Se han añadido diversos datos, algunos de ellos para facilitar el trabajo en Excel y que posteriormente no se han utilizado en el proceso de la información con Python, y otros, provenientes del resto de datos proporcionados por la empresa.

Algunos de los datos añadidos se han usado posteriormente como variables categóricas. Este es el caso de:

- festivo: 0 = no festivo, 1 = festivo
- diaSemana: numeración de 1 a 7, siendo 1 el Lunes y 7 el Domingo.
- promo: 0 = sin promoción, 1 = con promoción
- numSemEnMes: número de la semana en el mes (1 a 6)

Las variables: idPromo, precioTarifa, precioPromo, cantidadPromo y tipoPromo son datos recogidos del fichero de Promociones proporcionado. La variable tipoPromo aunque podría utilizarse como una variable categórica que ayudara en el análisis, no se ha utilizado ya que en la mayoría de registros aparece vacío. El campo difPrecio es un campo calculado que se ha añadido para identificar casos en los que el precio de promoción sea más alto que el precio de tarifa habitual y que por lo tanto, podría deberse a un error.

El fichero de stock ha recibido el mismo tratamiento inicial que el fichero de ventas y su resultado es el siguiente:

fecha	producto	f+p	udsStock	udsVenta	festivo	diaSemana	udsPrevision	promo	idPromo	precioTarifa	precioPromo	cantidadPromo	tipoPromo	difPrecio	numSemEnMes	Mes
01/01/2020	1	438311	1457	0	1	3	33657	0	0	0	0	0		0	1	1
01/01/2020	2	438312	1351	0	1	3	13386	0	0	0	0	0		0	1	1
01/01/2020	4	438314	1824	0	1	3	22948	0	0	0	0	0		0	1	1
01/01/2020	5	438315	578	0	1	3	18174	0	0	0	0	0		0	1	1
01/01/2020	9	438319	17	0	1	3	7968	0	0	0	0	0		0	1	1

Figura 3 - Fichero inicial de stocks con datos añadidos

La descripción de los nuevos datos añadidos es la misma que la explicada anteriormente para el conjunto de datos de ventas. La única diferencia es que en este caso se han tenido que añadir nuevos registros con información de ventas que no aparecían en el fichero de stocks, al no haber datos de stock para determinadas fechas de la serie temporal. Esto ha provocado que haya

fechas de la serie temporal en las que la información del stock esté vacía y en cambio haya información relativa a ventas. Dichos datos son tratados como datos faltantes y se completan calculando el promedio del stock de cada producto a lo largo de la serie temporal.

También se ha tenido que completar la información correspondiente al fichero de festivos ya que sólo contenía información del 2016 al 2019 y era necesario disponer de ella para los años 2015 a 2020. Los datos faltantes se han recogido de los calendarios oficiales de la Comunidad Autónoma correspondiente publicados en internet y se han añadido al fichero. De esta forma se ha podido completar la información en cuanto a festivos en los conjuntos de datos de ventas y stock.

Una vez obtenidos los dos ficheros resultantes, se aplicarán sobre ellos diversos algoritmos de machine learning de forma que se pueda comparar cual es más efectivo para el objetivo de este proyecto de gestión de inventario.

Es importante remarcar que en todo proyecto de ciencia de datos cuando se busca soluciones y respuestas a las preguntas que van surgiendo a lo largo de él, el proceso seguido hasta dar con la solución más eficiente se convierte en iterativo.

Tal como se ha indicado en la estrategia seguida en este proyecto se inicia el tratamiento en Python con el conjunto de datos de ventas.

El fichero “01_TablaVentas_Farma” resultante después del tratamiento inicial en Excel y convertido a csv, se carga en la plataforma de trabajo (libro Jupyter), se comprueba su contenido y se añaden índices alternativos (Año, Mes y AñoMes) que facilitarán el análisis posterior.

Se comprueba si existen valores nulos en el conjunto de datos.

```
fecha_f              0
producto             0
f+p                  0
udsVenta             0
festivo              0
diaSemana            0
udsStock          2727
udsPrevision         0
promo                0
idPromo              0
precioTarifa         8
precioPromo          9
cantidadPromo        0
tipoPromo        98496
difPrecio            0
numSemEnMes          0
Año                  0
Mes                  0
AñoMes               0
dtype: int64
```

Figura 4 - Comprobación de valores nulos en datos de ventas

Aparecen cuatro variables (udsStock, precioTarifa, precioPromo y tipoPromo) que contienen valores nulos / vacíos. Dichas variables no serán utilizadas en el análisis previo del comportamiento de ventas por lo que los valores nulos o vacíos se dejarán a cero, con la excepción de las unidades de stock en el que se modifican los valores vacíos por la media del stock de los productos.

Se visualizan los estadísticos básicos de la variable udsVenta:

	udsVenta
count	98590.000000
mean	881.896480
std	1478.265695
min	0.000000
25%	299.000000
50%	541.000000
75%	991.000000
max	99441.000000

Figura 5 - Estadísticos básicos de la variable udsVenta

Se observa un valor máximo que puede ser considerado anómalo respecto al resto de los datos. Podría ser que efectivamente se hubiera producido una

venta de tal cantidad de unidades, algo así como un pedido extraordinario, pero también podría deberse a un error en la recogida de datos.

Se procederá a la detección y eliminación de outliers adaptando el método utilizado por Li (2019) consistente en la detección de valores anómalos a partir de un proceso de clusterización. Se consideran outliers aquellas observaciones que no formen parte de alguno de los clústeres o que formen clústers muy pequeños.

Para asignar cada una de las observaciones a su clúster primero se realiza un análisis PCA para determinar las variables más importantes. Para poder hacerlo se estandarizan las variables con StandardScaler. A continuación se realiza una reducción de dimensionalidad tomando las dos variables consideradas más importantes (udsVenta y udsStock) y se asigna cada observación a su clúster.

Para buscar las anomalías se calcula la distancia entre cada punto y su centroide más cercano. Aquellas observaciones con las distancias más largas se consideran un outlier. Una vez identificadas, esas observaciones se eliminan.

En la siguiente imagen se pueden observar los outliers detectados en un gráfico con la vista de los clústers.

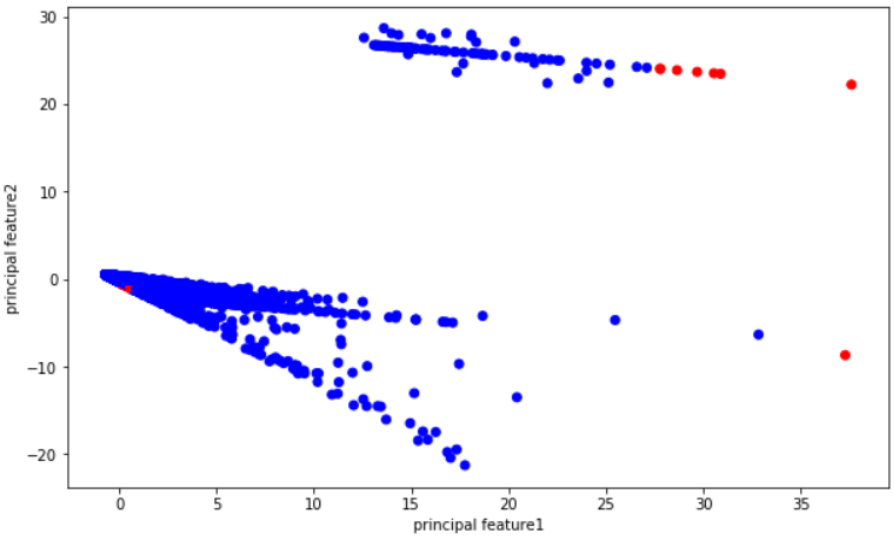

Figura 6 - Outliers detectados en datos de ventas (vista clústers)

Los clústers detectados se muestran a continuación:

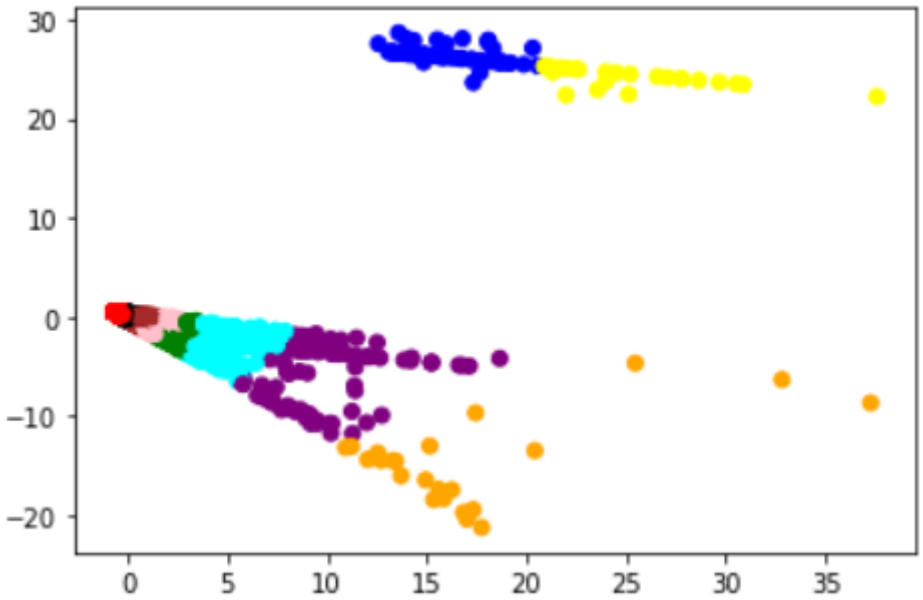

Figura 7 - Clústers identificados en datos de ventas

Y finalmente se observan los valores anómalos detectados (outliers) a lo largo de la serie temporal completa de 5 años. Se procede a su eliminación.

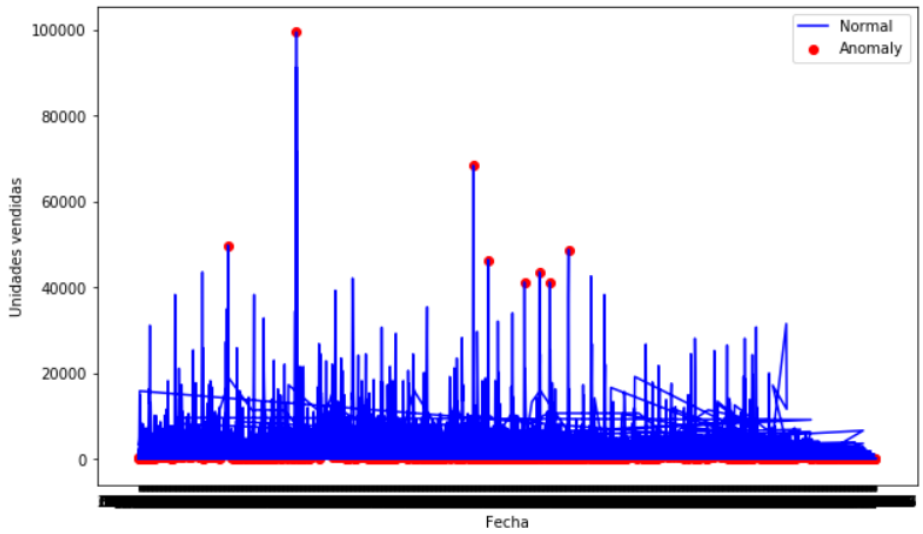

Figura 8 - Outliers detectados en serie temporal de ventas

El mismo proceso se usará para eliminar los outliers del conjunto de datos de stocks. En el capítulo correspondiente al preprocesado de los datos de stocks se puede encontrar una explicación más detallada del proceso seguido.

3.3 Análisis de los datos de ventas

Tras la fase de preprocesado de los datos se aborda una etapa de análisis inicial del conjunto de datos de venta tanto en su vertiente estadística como en la de visualización de los datos.

3.3.1 Características del análisis

El conjunto de datos de venta a analizar tiene 98.590 observaciones de 17 variables (entre originales y añadidas). Todo ello comprende un período de observaciones de 5 años para 74 productos (entre el 1 y el 99).

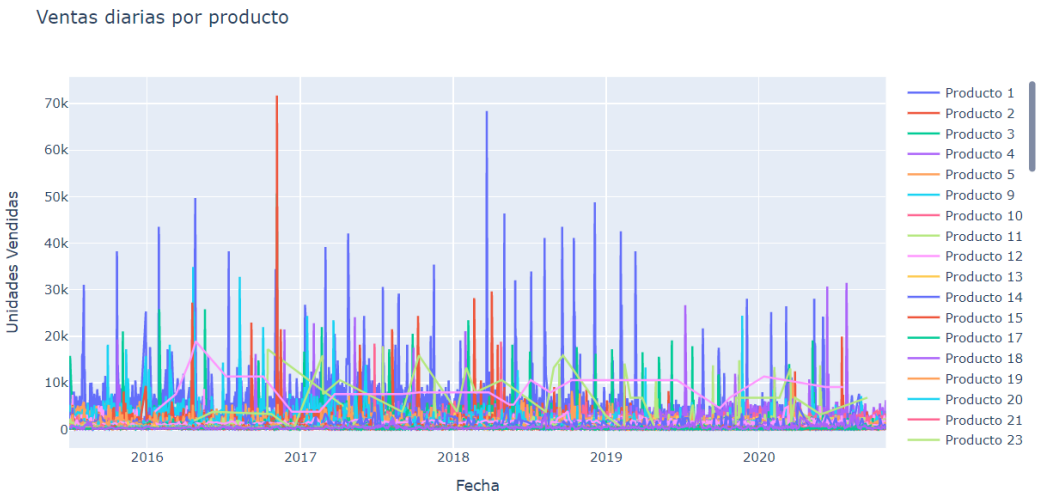

Figura 9 - Ventas diarias por producto

En el gráfico se observa la evolución de las ventas de los productos a lo largo de los 5 años de observaciones disponibles. Se puede ver que todos los productos están dentro de un rango parecido de ventas, con la excepción del producto 1 (líneas azules) en los que la cantidad de ventas es bastante mayor que en el resto para meses determinados. En el gráfico siguiente se visualiza la evolución de ventas únicamente del producto 1.

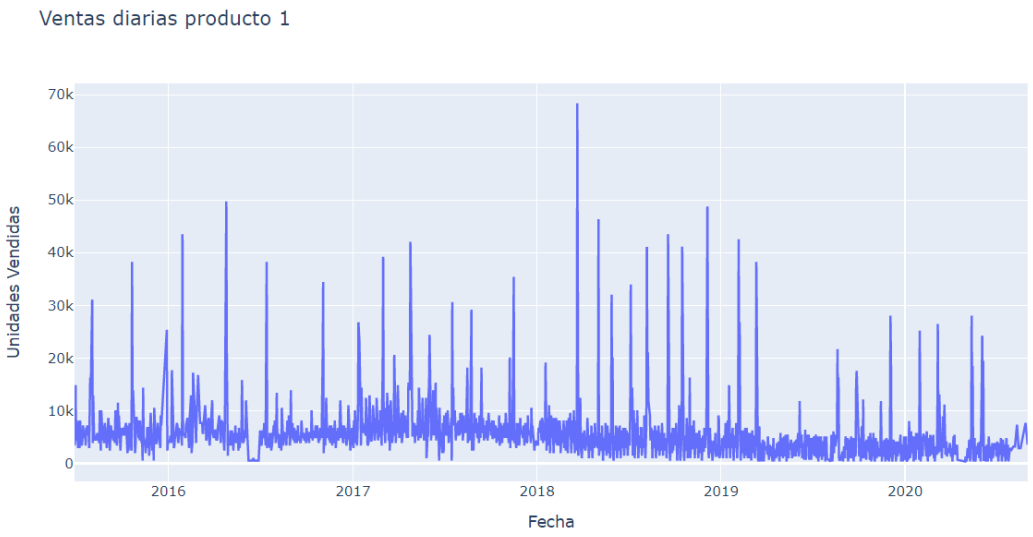

Figura 10 - Evolución ventas diarias producto 1

Se observa claramente que eliminado las ventas del producto 1, el resto de productos se venden en cantidades más similares. En el siguiente gráfico se pueden observar las ventas totales por producto y se confirma que el producto 1 es el que más ventas tiene a lo largo de la serie temporal de 5 años.

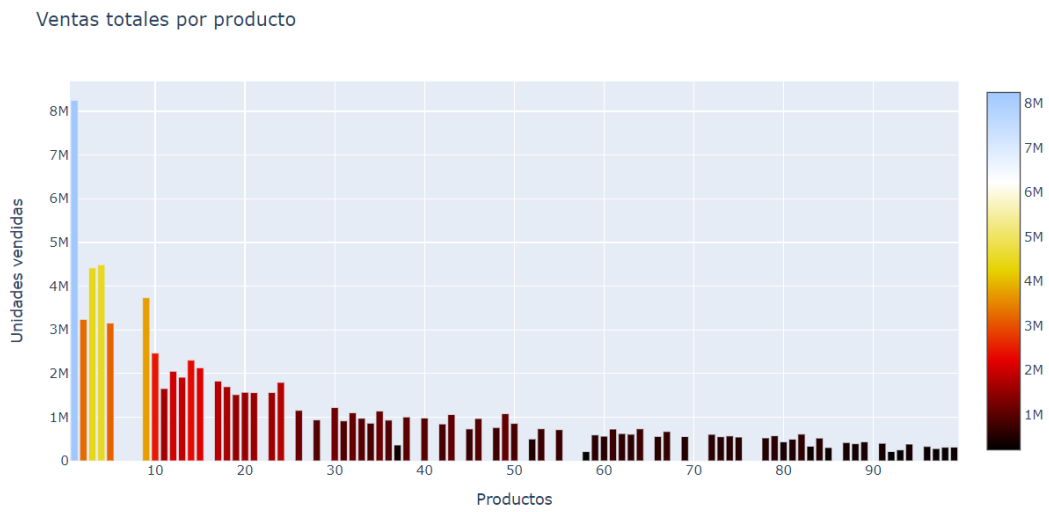

Figura 11 - Ventas totales por producto

Del mismo modo, también se confirma que hay productos que no se han vendido a lo largo de los 5 años y entre los códigos de producto 1 a 99 sólo disponemos de datos de 74.

En cuanto al comportamiento de ventas en función de los días de la semana, el Viernes es el día con mayor número de unidades vendidas (18.563.658),

seguido del Jueves (17.339.411), Miércoles (15.305.502), Martes (14.579.103) y Lunes (14.568.785), siendo Sábado (6.045.362) y Domingo (544.353) los días con menos volumen de ventas.

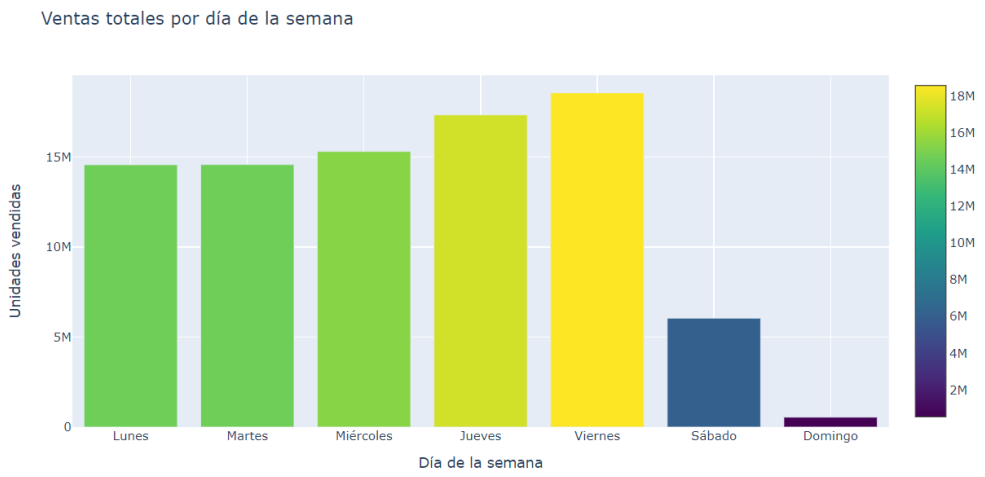

Figura 12 - Ventas totales por días de la semana

Tal como se ha visto, los días laborables (85.913.368) acumulan la mayor cantidad de unidades vendidas, mientras que en los festivos (1.032.806) las ventas disminuyen drásticamente.

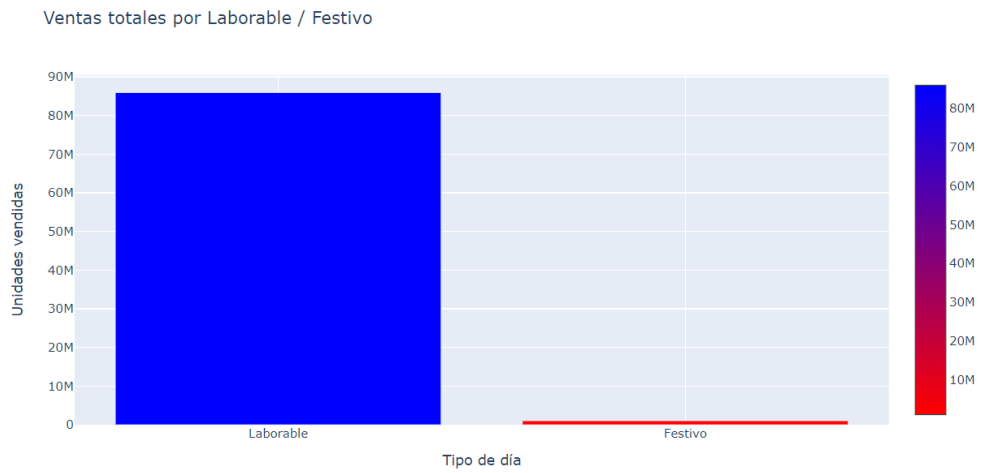

Figura 13 - Ventas totales por laborable / festivo

Otro aspecto interesante y que también se ha incluido como variable que ayude en la predicción, es ver la distribución por número de semana en el mes. El caso extremo es que en un mismo mes puede haber 6 semanas diferentes (que el día 1 fuera domingo, después 4 semanas de 7 días y los días 30 y 31 serían lunes y martes de la sexta semana del mismo mes). Atendiendo a las observaciones realizadas, se puede decir que la semana en la que se vende

más unidades es la segunda (20.826.954), seguida de la tercera (19.821.049), la cuarta (19.405.569), la quinta (16.380.607), la primera (9.448.078) y por último, la sexta (1.063.917).

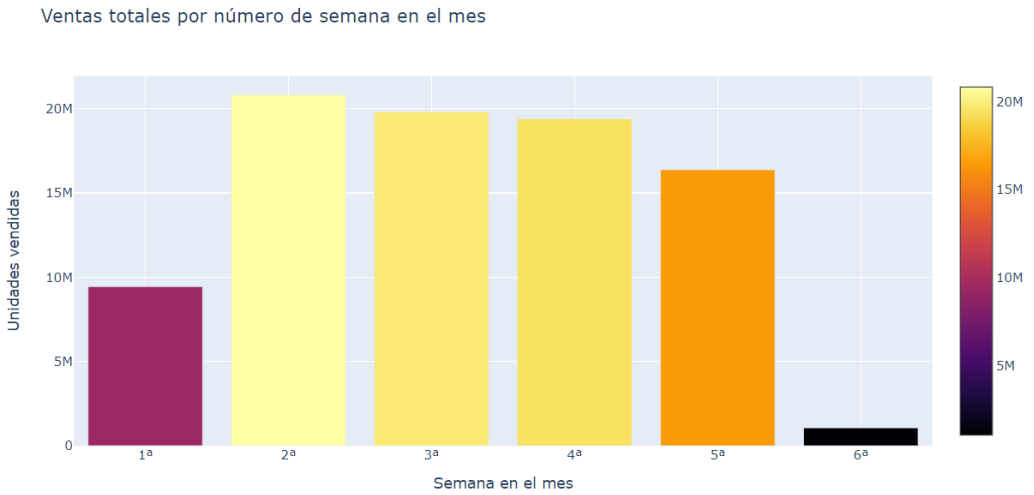

Figura 14 - Ventas totales por número de semana en el mes

Atendiendo a la distribución por meses se observa que a pesar de ser bastante similar, dentro de un rango de variación aceptable, entre todos los meses, se produce una mayor venta de productos en el segundo semestre del año. Julio (8.311.055) es el mes en el que se han vendido más unidades en esos 5 años, seguido de Octubre (8.134.287), Septiembre (8.038.661) y en la cuarta posición aparece Marzo (7.608.836) un mes del primer semestre. En la siguiente gráfica se puede ver un histograma con las ventas de cada mes.

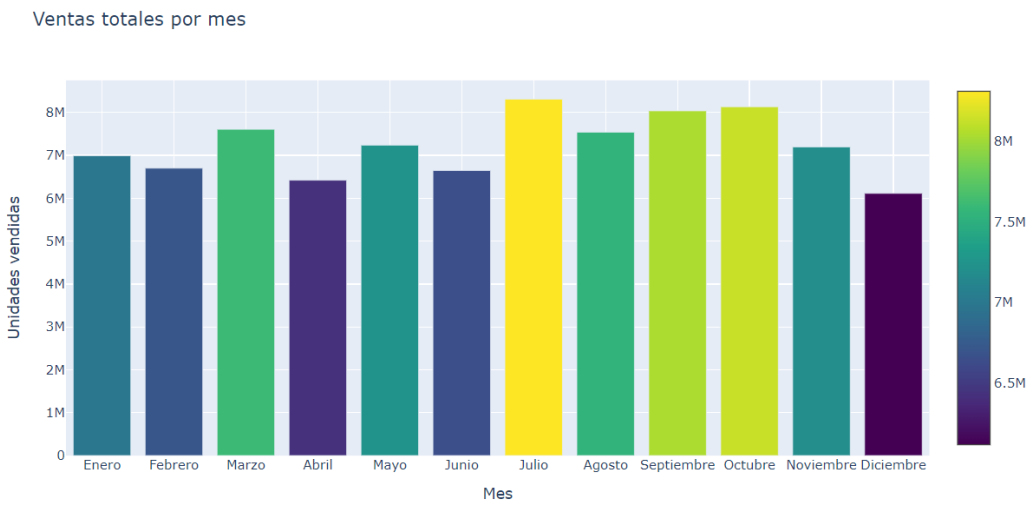

Figura 15 - Ventas totales por mes

Otro aspecto a observar es cómo afectan las promociones al comportamiento de las ventas. Para ello, se comprueba que las unidades vendidas sin estar dentro de una promoción son 85.567.389, mientras que las unidades vendidas dentro de una promoción son sólo 1.378.785, lo que significa que no parece ser un gran reclamo para el incremento de ventas.

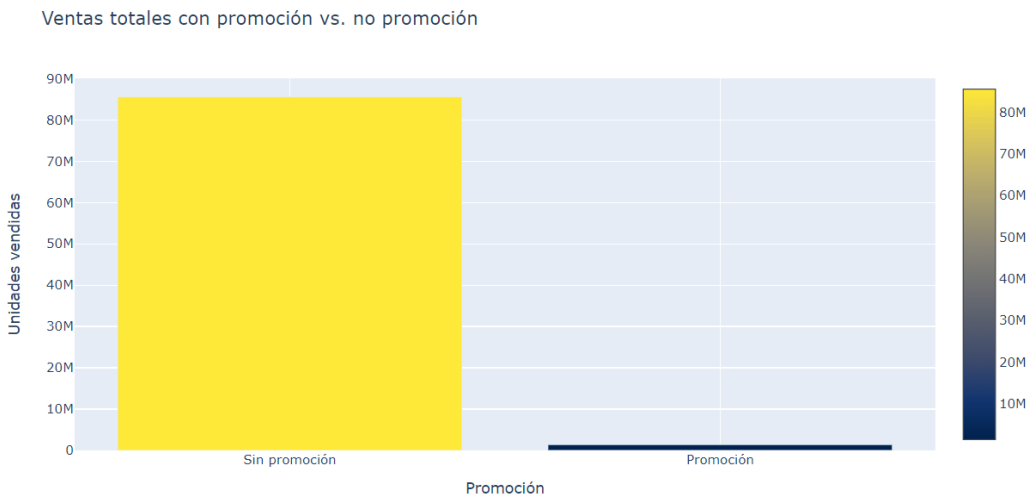

Figura 16 - Ventas totales con promoción vs. sin promoción

Finalmente, durante los aproximadamente cinco años de los que disponemos datos de ventas, las unidades vendidas en cada período (año/mes) se han mantenido bastante estables, con la excepción de algunos picos en los que la venta se ha disparado y, algunas caídas que se han producido básicamente en el año 2020. Las ventas durante los 65 períodos que componen la serie temporal se pueden ver en la siguiente gráfica:

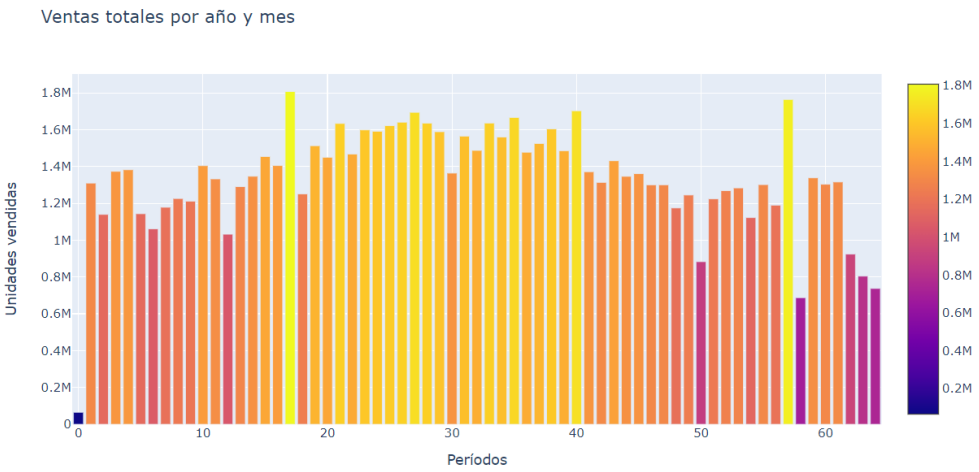

Figura 17 - Ventas totales por período (año/mes)

3.3.2 Correlación entre variables

Es interesante ver si existe correlación entre algunas variables del conjunto de datos de ventas.

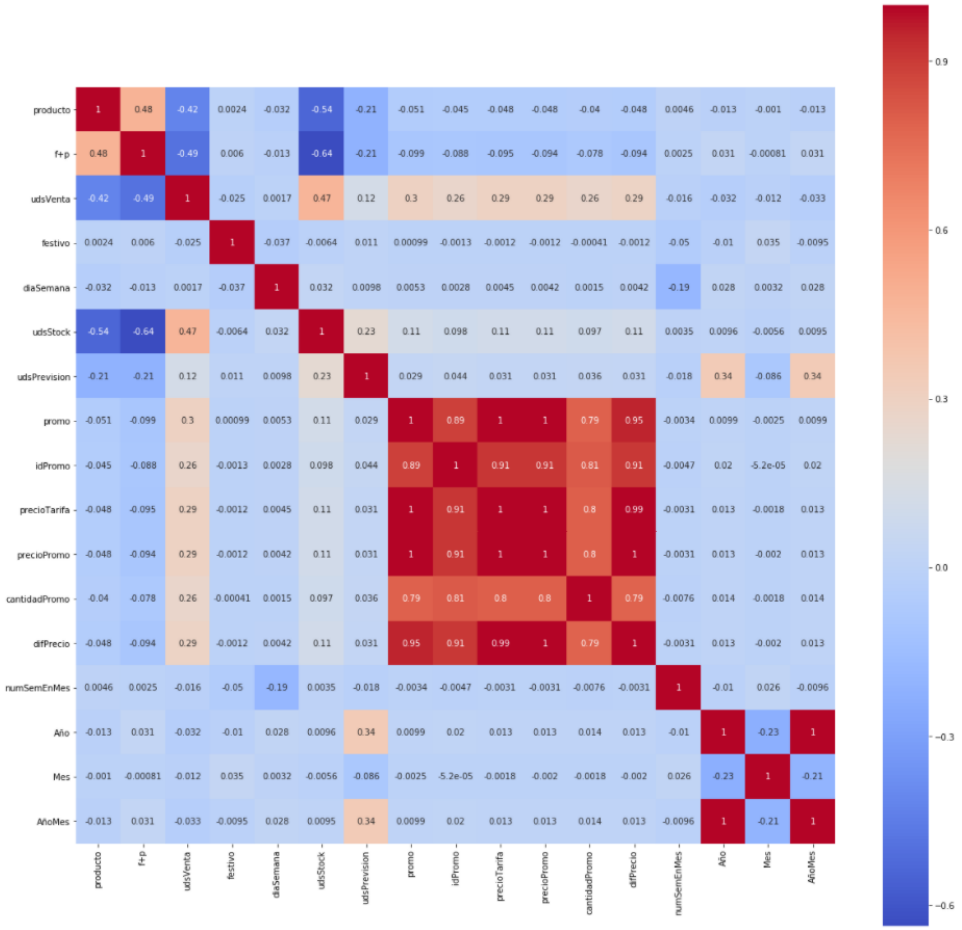

Figura 18 - Heatmap - Correlación entre variables de datos de ventas

Visualizarlo de forma gráfica a partir de un heatmap permite ver muy fácilmente la correlación entre todas las variables del dataframe y será muy útil para tener en cuenta las variables a usar en el momento de la predicción.

Se observa que las mayores correlaciones existen entre las unidades de venta (udsVenta), las unidades de stock (udsStock) y el indicador de si existe una promoción (promo) para un producto en una fecha.

Si aparecen variables fuertemente correlacionadas significa que de un modo u otro la información que proporcionan es redundante. De este modo, dadas dos variables con fuerte correlación, eliminado una de ellas podemos evitar la generación de un sesgo en la predicción de la información que contienen. Este

caso es importante tenerlo en cuenta cuando se quieren extraer conclusiones sobre la importancia de una variable en el resultado de la predicción. En este proyecto, se puede ver con claridad que todos los datos referidos a las promociones son redundantes entre sí, por lo que no se usarán en el momento en que se hagan los desarrollos aplicando los algoritmos de machine learning que se desean considerar.

3.3.3 Análisis componentes de la serie temporal

Otro aspecto importante en el análisis es descomponer la serie temporal en sus componentes para averiguar si existen efectos como: heterocedasticidad, tendencia, estacionalidad, etc.

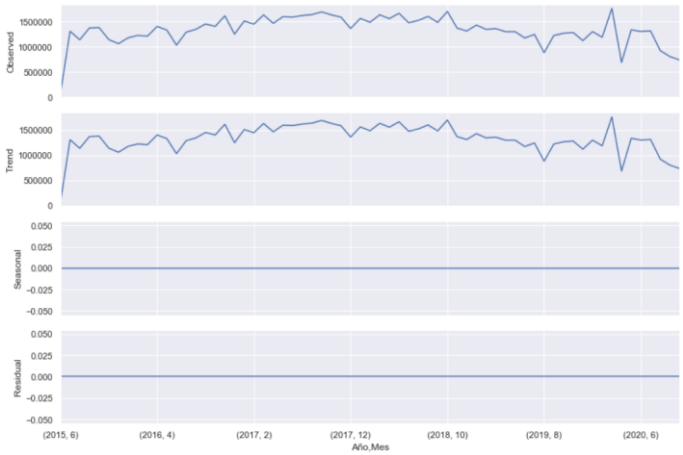

Figura 19 - Descomposición de componentes de la serie temporal

Según el resultado que podemos ver no parece existir ni una tendencia claramente marcada de aumento o disminución y tampoco un comportamiento estacional. Pero aquí se está observando el total de ventas de todos los productos y es posible que exista algún producto que se comporte de forma estacional o que tenga una tendencia clara al alza o a la baja.

Se ha calculado individualmente si podría existir estacionalidad, calculando el patrón de la misma, sin utilizar la descomposición automática de componentes y el resultado ha sido similar al obtenido anteriormente, como se observa a continuación:

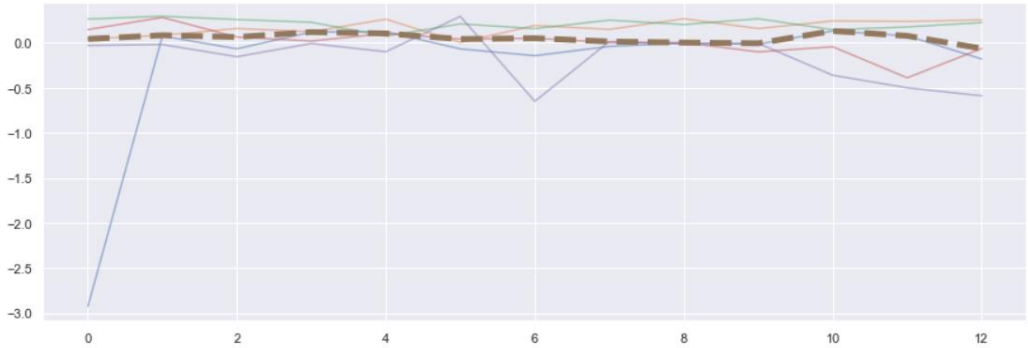

Figura 20 - Patrón de estacionalidad

3.4 Preprocesado de los datos de stocks

Tal como se había descrito anteriormente en la estrategia de este proyecto, una vez analizados los datos de ventas y comprendido el comportamiento de los productos a lo largo de la serie temporal de 5 años, el segundo paso consiste en tratar los datos de stocks de forma que se pueda predecir las unidades de stock futuras, teniendo en cuenta el histórico de ventas y su previsión, pero haciéndolo en un sólo paso, es decir, sin que sea necesario hacer una predicción de la demanda para después predecir el stock necesario. La cifra de ventas prevista ya viene dada y sobre este dato se trabaja.

3.4.1 Clusterización y eliminación de outliers

Un paso fundamental que ayudará mucho a la predicción del stock es la agrupación de productos según su comportamiento o patrón de venta. Es el proceso que se denomina clusterización o segmentación y que consiste en agrupar datos similares en función de una serie de variables.

Para efectuar este proceso se utiliza el algoritmo K-Means que está considerado como perteneciente al ámbito de los algoritmos no supervisados y se basa en la división de un conjunto de *n* observaciones en *k* grupos, teniendo en cuenta que cada una de las observaciones pertenece al grupo al que está más cercano. Los diferentes grupos están definidos por un punto denominado

centroide y es sobre el que se calcula la distancia de cada una de las observaciones.

Lo primero que se hace es definir el número de grupos o clústeres. Cada observación debe estar muy cerca de las de su mismo grupo y los grupos lo más alejados posible unos de otros, es decir, minimizar la varianza intra-clúster y maximizar la varianza inter-clúster. Además en este proyecto se usa este proceso para aprovechar y eliminar los valores que se pueden considerar anómalos (outliers). Se considera que aquellas observaciones que no formen parte de alguno de los clústeres son outliers y una vez detectados serán eliminados.

Para averiguar el número óptimo de clústeres el método que se aplica es la denominada regla del codo (Elbow curve).

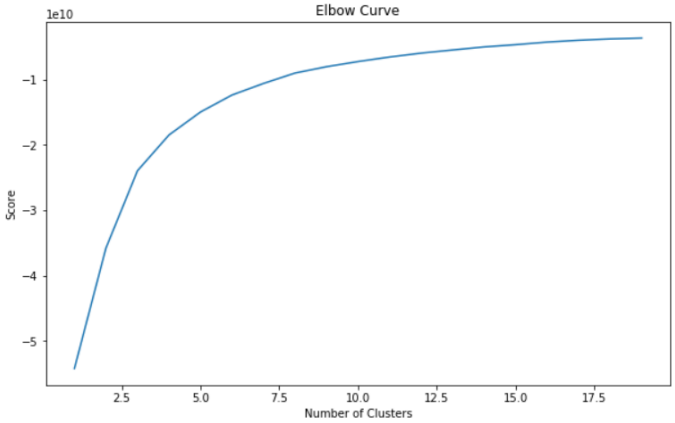

Figura 21 – Número de clústeres según el método del codo

En este caso se consideran 10 clústeres para realizar la agrupación de productos ya que añadir más clústeres no explicaría mucho más la varianza en las unidades de stock.

Las variables tenidas en cuenta para determinar el número de clústers con el método del codo son: unidades de stock, unidades de venta e indicador de si existía promoción o no. Se han elegido estas al ser las variables que tienen una mayor correlación, tal como se ha visto antes en el análisis de correlación entre variables cuando se analizaban los datos de ventas.

A continuación se puede ver un gráfico con las observaciones según las tres variables tenidas en cuenta.

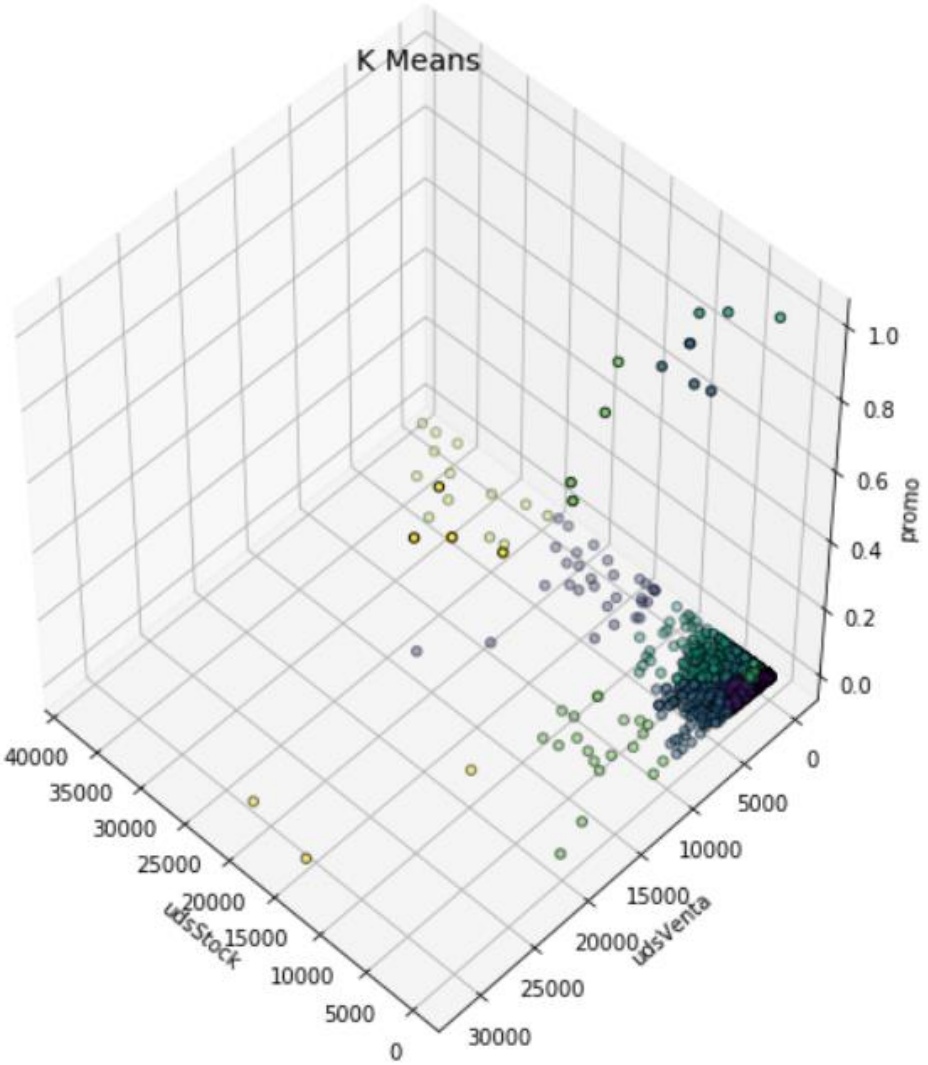

Figura 22 - Observaciones de productos según stock, venta y promoción

Queda claro que las tres variables no tienen la misma importancia para contribuir a la asignación de cada una de las observaciones al clúster más adecuado y para averiguarlo se puede visualizar la incidencia de cada uno de los componentes principales.

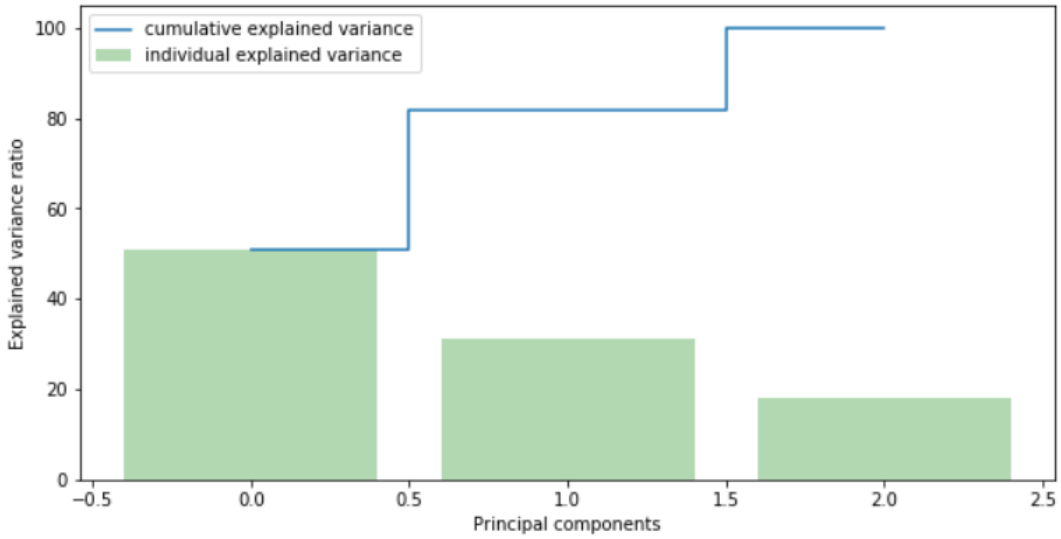

Figura 23 - Contribución de los principales componentes

Se observa que el primer componente explica aproximadamente el 55% de la varianza y el segundo explica alrededor del 27%, lo que significa que con los dos primeros alcanzamos el 82% y por lo tanto, se puede prescindir del tercero y dejar el número de componentes en 2.

Para asignar cada una de las observaciones a su clúster se toman las variables acabadas de usar para determinar la importancia y se estandarizan con StandardScaler. A continuación se realiza una reducción de dimensionalidad tomando las dos variables consideradas más importantes y se asigna cada observación a su clúster.

```
3    7172
8    4832
0    2609
5    1077
4     376
7      31
2      30
6      15
1      10
9       6
```

Figura 24 - Número de observaciones en cada clúster

3.4.2 Eliminación automática de valores anómalos (outliers)

Tal como se ha dicho anteriormente, se ha aprovechado el proceso de clusterización para detectar las anomalías (outliers). Como Li (2019) explicó, la

detección automática de outliers se ha vuelto crítica hoy en día ya que el enorme volumen de datos hace imposible la identificación manual de los datos anómalos.

En este proyecto se aplica la detección automática de outliers basada en el proceso de clustering realizado y considerando que las observaciones que caen fuera de los clústers o que pertenecen a clústers muy pequeños pueden ser potenciales anomalías.

Para buscar las anomalías se calcula la distancia entre cada punto y su centroide más cercano. Aquellas observaciones con las distancias más largas se consideran un outlier. Una vez identificadas, esas observaciones se eliminan.

En el siguiente gráfico se aprecian las anomalías detectadas (en rojo) en una vista de los clústers.

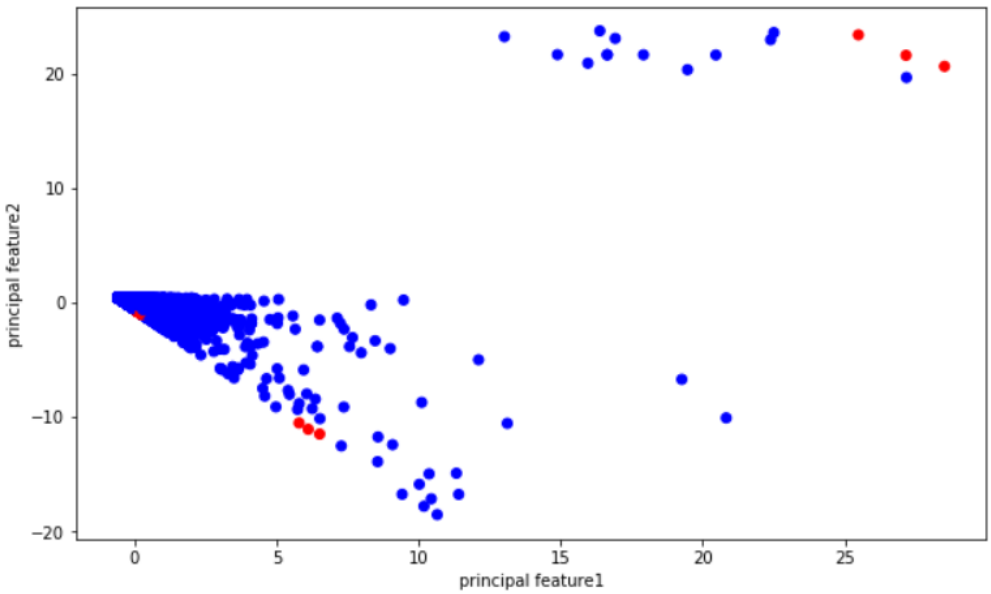

Figura 25 - Outliers detectados en clústers

La vista de los clústers identificados por colores permite ver claramente la agrupación de las observaciones.

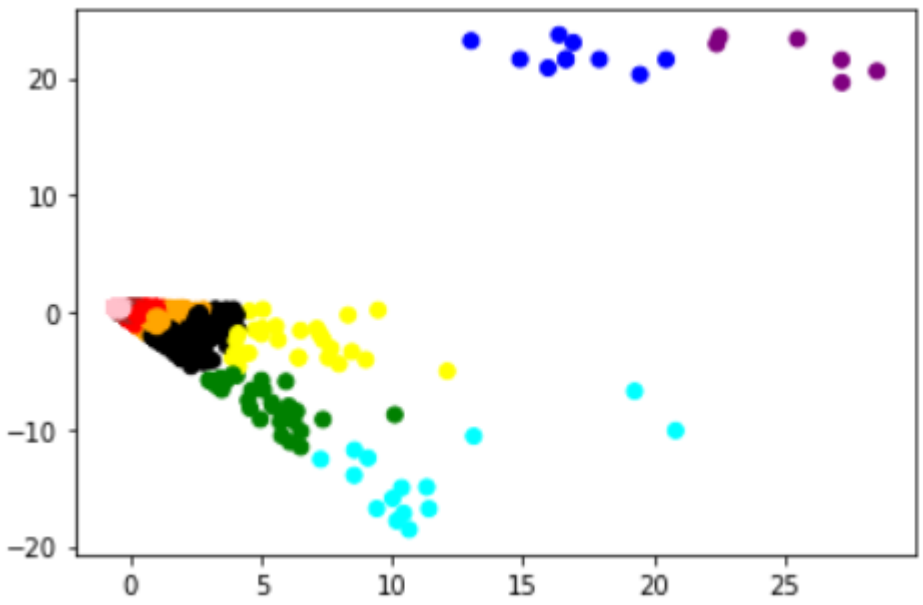

Figura 26 - Visualización de los clústeres

También se puede observar la visualización de las anomalías detectadas (outliers) en las distintas observaciones a lo largo de la serie temporal.

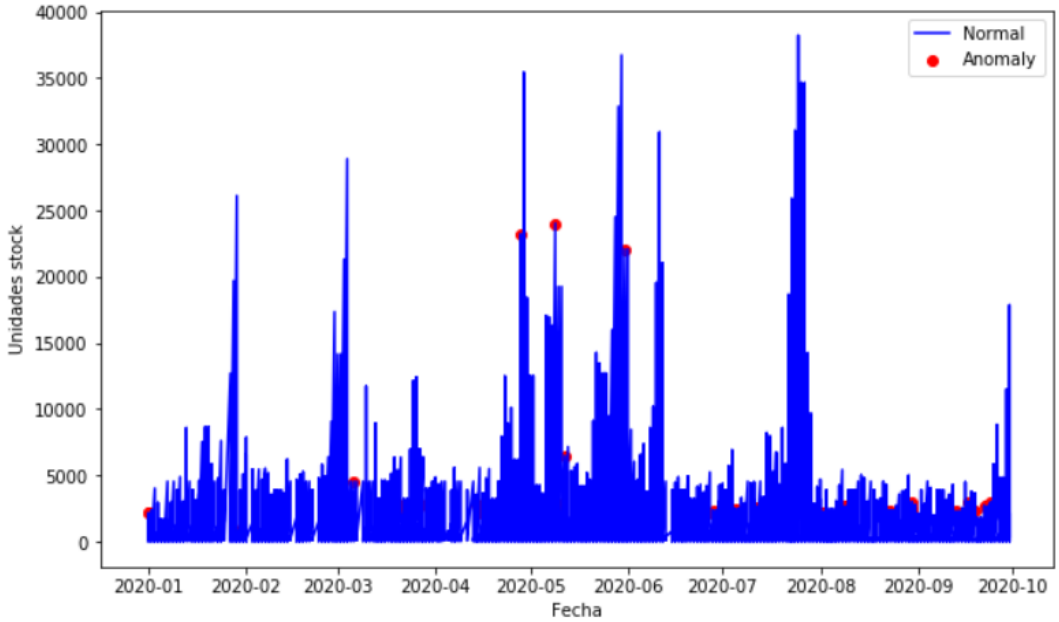

Figura 27 - Anomalías detectadas en la serie temporal

Los puntos rojos muestran las observaciones detectadas como valores extremos que deben ser eliminados. Se observa que se han detectado como anomalías tanto valores muy altos como otros que son bajos.

3.5 Análisis predictivo con algoritmos Machine Learning

Una vez ya se ha realizado el preprocesado y la clusterización de las observaciones se ha dividido el fichero en un conjunto de datos de entrenamiento y otro conjunto de datos de test. Éstos serán la base para las pruebas a realizar con los diferentes algoritmos de machine learning tomando siempre el mismo punto de partida.

Con el fin de mejorar la velocidad de proceso de los diferentes algoritmos, se ha limitado el número de observaciones en ambos conjuntos. Para los datos de entrenamiento se han tomado los datos desde el 01.01.2020 hasta el 31.07.2020, lo que significa disponer de 12.533 observaciones. En el caso de los datos de test se dispone de los datos entre el 01.08.2020 y el 30.09.2020, lo que totaliza 3.464 observaciones.

En estos dos ficheros se han eliminado aquellos datos que se han considerado no importantes para hacer las predicciones con los diferentes algoritmos. Los datos que finalmente se tienen en cuenta son los siguientes:

- Fecha
- Producto
- Unidades de stock
- Unidades de venta
- Festivo
- Día de la semana
- Promoción
- Número de la semana en el mes
- Mes
- Clúster

Todos los datos son numéricos, aunque para ciertos procesos algunos de ellos serán considerados como variables categóricas.

3.5.1 Selección de algoritmos para el análisis predictivo

Para realizar las tareas de predicción, que es el objetivo del proyecto, se han tenido en cuenta algunos de los algoritmos recogidos en el análisis de la literatura y que mejores resultados han obtenido.

Específicamente, Khalilzadeh (2008), He (2013) o Šustrová (2016) emplearon redes neuronales para realizar sus predicciones, Barbosa et al. (2017) usaron Random Forest y Gradient Tree Boosting para las suyas, Praveen et al. (2020) utilizó XGBOOST (Extreme Gradient Boosting) para hacer predicciones de la demanda y finalmente Shoujing Zhang et al. (2020) usó un algoritmo de clustering mejorado (Mahalanobis distance clustering algorithm) combinado con una BP NN, usando PCA (Principal Component Analysis) antes del algoritmo mejorado para realizar una reducción de dimensionalidad de datos y K-Means como método de clustering, con el fin de averiguar el grado de importancia de las piezas de repuesto.

Se han realizado pruebas con varios de estos algoritmos que se detallarán explicando el resultado obtenido.

3.5.2 Red Neuronal Artificial multicapa con Embeddings

La primera de las técnicas a utilizar va a ser una red neuronal feedforward backpropagation con Embeddings.

Los Embeddings son un método para representar variables categóricas como si fueran continuas y son incluidos como input en el modelo de red neuronal para aprendizaje supervisado. Funcionan como representaciones de categorías de forma que las categorías similares están más próximas entre sí y conforman parámetros (pesos) que se ajustan durante el entrenamiento y que permiten minimizar la pérdida en el modelo. Con los Embeddings se puede obtener una mejor predicción que si sólo se utilizara las unidades de stock. Pero en este proyecto, además de los Embeddings y las unidades de stock, se utilizarán las unidades de venta.

fecha	producto	udsStock	udsVenta	festivo	diaSemana	promo	numSemEnMes	Mes	cluster
2020-07-31	94	300	309	0	5	0	5	7	3
2020-07-31	96	292	300	0	5	0	5	7	3
2020-07-31	97	51	126	0	5	0	5	7	3
2020-07-31	98	280	206	0	5	0	5	7	3
2020-07-31	99	375	747	0	5	0	5	7	8

Figura 28 - Datos de entrenamiento

En la Figura 25 se puede ver una parte del contenido de los datos de entrenamiento. Lo que se va a hacer es definir las variables: producto, festivo, diaSemana, promo, numSemEnMes, Mes y cluster como variables categóricas (los Embeddings mencionados antes) que ayudarán a mejorar la precisión de la predicción. El resto de variables se dejarán tal cual están. Todas ellas serán tratadas por la red neuronal para calcular la predicción final.

Las variables que hemos definido como categóricas tienen valores numéricos contínuos y la red aprende al usar backpropagation ya que los errores vuelven a las capas anteriores con el fin que aprendan de ellos y en cada pasada, la red se vuelve más precisa hasta llegar a la predicción final.

La red que se ha definido consta de: input, Embeddings de las 7 variables, Reshape de los 7 Embeddings para convertirlos a un número, concatenate de los Embeddings y 6 capas fully-connected (Dense) para generar la salida. El modelo utiliza un optimizador “Adam” para modificar los parámetros de la red después de calcular los gradientes mediante backpropagation. El entrenamiento se hace en 100 epochs. La pérdida estimada tras las 100 epochs es de 0,2332 lo que significa una precisión del 76,68%.

Aunque generalmente el uso de redes neuronales requiere normalizar los datos, no se realiza en este proyecto ya que, a parte de los Embeddings, las dos variables usadas (udsStock y udsVenta) están expresadas con la misma escala y unidades y por ello no se considera necesario.

En la siguiente figura se puede ver un resumen de la estructura final de la red definida.

```
In [51]: nn.model.summary()
```

Model: "model_1"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 1)	0	
input_2 (InputLayer)	(None, 1)	0	
input_3 (InputLayer)	(None, 1)	0	
input_4 (InputLayer)	(None, 1)	0	
input_5 (InputLayer)	(None, 1)	0	
input_6 (InputLayer)	(None, 1)	0	
input_7 (InputLayer)	(None, 1)	0	
prod_embedding (Embedding)	(None, 1, 50)	5000	input_1[0][0]
fest_embedding (Embedding)	(None, 1, 1)	2	input_2[0][0]
dsem_embedding (Embedding)	(None, 1, 4)	32	input_3[0][0]
prom_embedding (Embedding)	(None, 1, 1)	2	input_4[0][0]
nsem_embedding (Embedding)	(None, 1, 3)	21	input_5[0][0]
mes_embedding (Embedding)	(None, 1, 6)	72	input_6[0][0]
clus_embedding (Embedding)	(None, 1, 5)	50	input_7[0][0]
reshape_1 (Reshape)	(None, 50)	0	prod_embedding[0][0]
reshape_2 (Reshape)	(None, 1)	0	fest_embedding[0][0]
reshape_3 (Reshape)	(None, 4)	0	dsem_embedding[0][0]
reshape_4 (Reshape)	(None, 1)	0	prom_embedding[0][0]
reshape_5 (Reshape)	(None, 3)	0	nsem_embedding[0][0]
reshape_6 (Reshape)	(None, 6)	0	mes_embedding[0][0]
reshape_7 (Reshape)	(None, 5)	0	clus_embedding[0][0]
concatenate_1 (Concatenate)	(None, 70)	0	reshape_1[0][0] reshape_2[0][0] reshape_3[0][0] reshape_4[0][0] reshape_5[0][0] reshape_6[0][0] reshape_7[0][0]
dense_1 (Dense)	(None, 128)	9088	concatenate_1[0][0]
activation_1 (Activation)	(None, 128)	0	dense_1[0][0]
dense_2 (Dense)	(None, 128)	16512	activation_1[0][0]
activation_2 (Activation)	(None, 128)	0	dense_2[0][0]
dense_3 (Dense)	(None, 64)	8256	activation_2[0][0]
activation_3 (Activation)	(None, 64)	0	dense_3[0][0]
dense_4 (Dense)	(None, 64)	4160	activation_3[0][0]
activation_4 (Activation)	(None, 64)	0	dense_4[0][0]
dense_5 (Dense)	(None, 64)	4160	activation_4[0][0]
activation_5 (Activation)	(None, 64)	0	dense_5[0][0]
dense_6 (Dense)	(None, 32)	2080	activation_5[0][0]
activation_6 (Activation)	(None, 32)	0	dense_6[0][0]
dense_7 (Dense)	(None, 1)	33	activation_6[0][0]

Total params: 49,468
Trainable params: 49,468
Non-trainable params: 0

Figura 29 - Resumen estructura red neuronal

Una vez entrenada la red se realiza el proceso de predicción utilizando los datos de test, por lo que se va a realizar la predicción del stock para los dos meses siguientes (01.08.2020 al 30.09.2020).

Se puede visualizar un gráfico con los resultados de la predicción comparado con los datos originales.

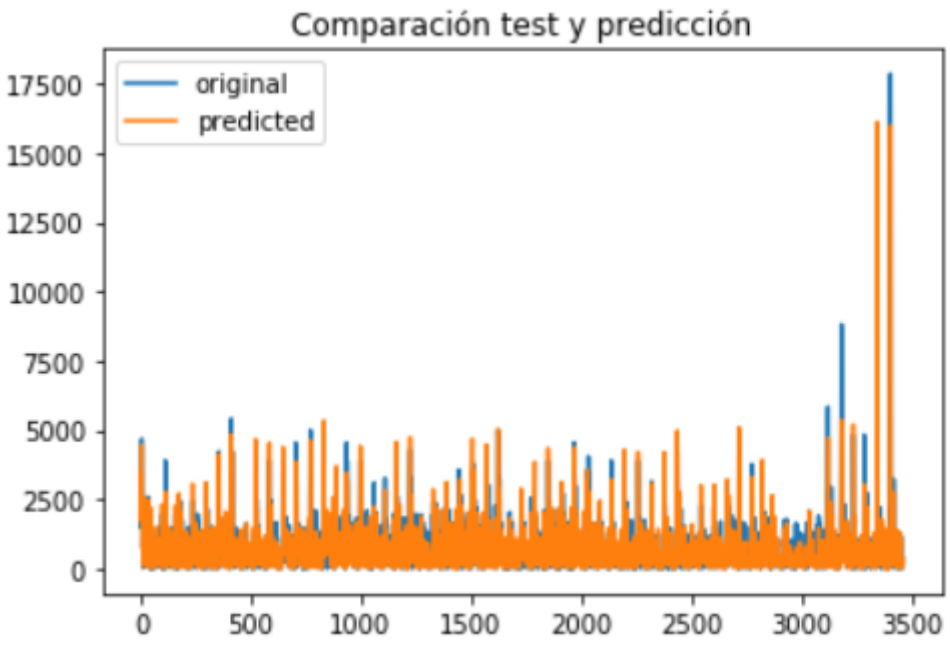

Figura 30 - Comparación datos test y predicción

Se observa bastante superposición entre ambas series pero no queda claro del todo y es mejor poder ver la comparativa para algunos materiales de forma individual.

En primer lugar vemos una pequeña muestra de los datos reales comparados con los datos de la predicción (columna stock).

fecha	producto	stock	real
2020-08-01	1	3416	3210
2020-08-01	2	1721	1524
2020-08-01	3	1953	1942
2020-08-01	4	1897	1474
2020-08-01	5	4460	4686
2020-08-01	9	1825	2453
2020-08-01	10	745	847
2020-08-01	11	843	746
2020-08-01	12	782	1301
2020-08-01	13	717	976

Figura 31 - Muestra de resultados de la predicción

El explained variance score de la predicción es de 0,82. Se aprecia bastante similitud pero queda mucho más visual si vemos la misma comparativa en formato gráfico para algunos productos.

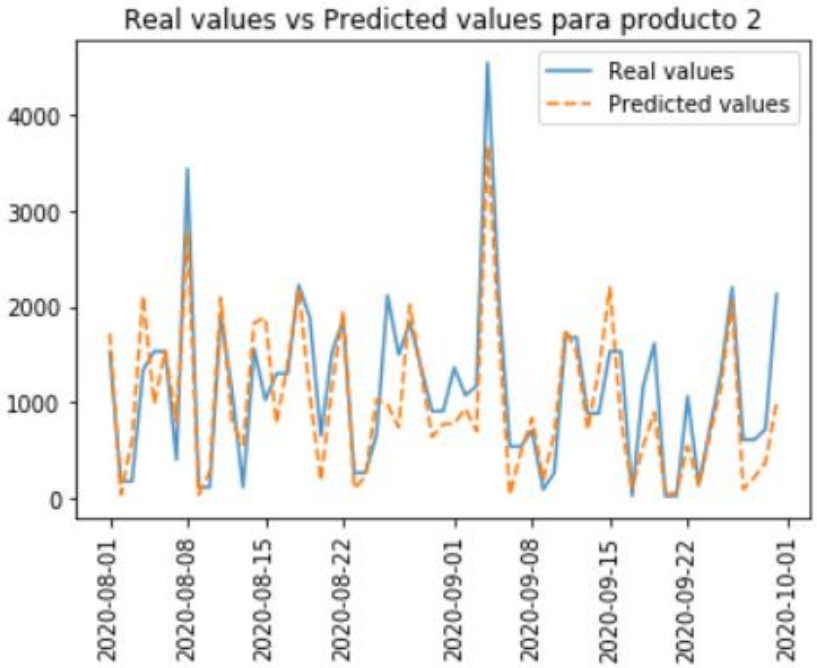

Figura 32 – NN - Valores reales vs. predicción - Producto 2

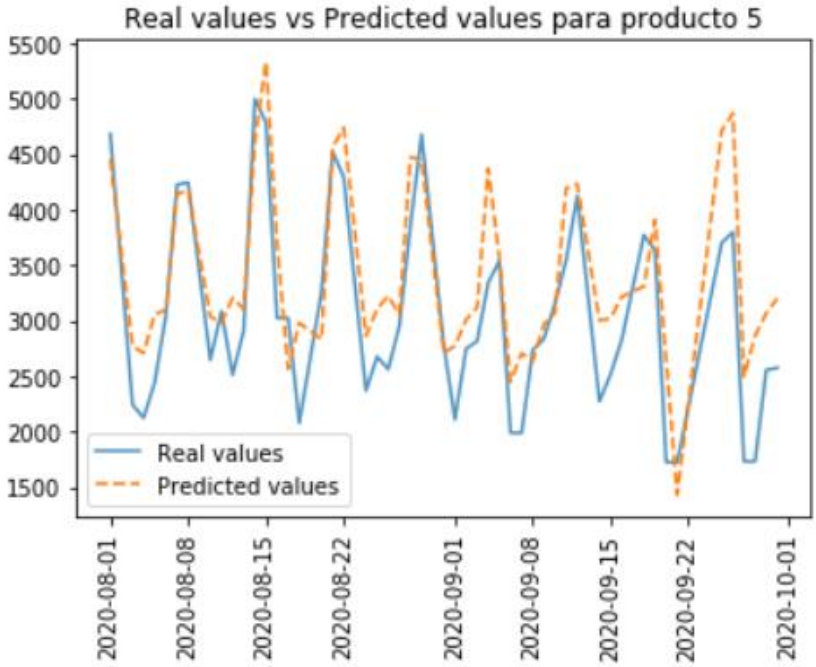

Figura 33 – NN - Valores reales vs. predicción - Producto 5

Se observa un nivel de superposición alto entre los datos reales y los de la predicción.

En el modelo presentado se tiene en cuenta la cantidad de unidades vendidas que seguro que afectan a la predicción. La empresa farmacéutica, con los niveles de stock actuales obtiene un nivel de servicio del 99%, lo cual es magnífico pero es caro ya que es posible que se esté almacenando un exceso de stock. Si el modelo desarrollado es capaz de adecuar el stock a la demanda, reduciendo el coste y conservando el nivel de servicio, será un éxito.

Veamos a continuación una tabla en la que se muestra el valor total del stock de todos los productos para las primeras fechas de la predicción.

Fecha	Stock pred.	Stock real	Valor pred.	Valor real
01/08/2020	45.385	48.801	195.130,16	211.017,09
02/08/2020	19.925	20.920	85.348,68	92.315,33
03/08/2020	32.022	31.602	135.865,84	132.995,82
04/08/2020	31.257	35.584	134.417,73	155.186,39
05/08/2020	33.101	35.343	144.684,24	156.536,40

Tabla 4 - Muestra resultados predicción red neuronal

Se puede ver que en algunas fechas el modelo ha reducido el stock pero en otras (03/08/2020) el stock se ha incrementado. Es el resultado de adecuarlo a la demanda.

Si calculamos los totales para los dos meses de predicción el resultado es el siguiente:

Fecha	Stock pred.	Stock real	Valor pred.	Valor real
TOTAL	2.185.866	2.346.909	9.294.586,83	10.134.608,55

Tabla 5 - Resultados totales predicción red neuronal

Se ha producido una reducción de 161.043 unidades de stock que significa una reducción de 840.021,72 €.

3.5.3 Gradient Boosting Regressor

Los métodos predictivos basados en árboles son unos de los que mejores resultados proporcionan y por esta razón se han vuelto muy populares.

Amat (2020) explica muy bien su funcionamiento y sus diferentes tipos. Los modelos de Gradient Boosting están formados por un conjunto de árboles de decisión individuales cuyo entrenamiento se realiza de forma secuencial, de tal forma que cada árbol nuevo trata de mejorar los errores de los árboles anteriores.

El algoritmo de Gradient Boosting es muy bueno tratando datos tabulares, lo cual se adecúa muy bien al caso del presente proyecto. Con este algoritmo tampoco se ha realizado ningún proceso de estandarización o normalización ya que los algoritmos basados en árboles no son sensibles a las magnitudes de las variables.

Para crear el modelo se han hecho diferentes pruebas de combinación de parámetros y finalmente se ha decidido usar los siguientes:

- n_estimators: 500
- max_depth: 7
- min_samples_split: 5
- learning_rate: 0.01
- los: lad

El parámetro n_estimators corresponde al número de árboles a tratar. En teoría cuanto mayor es el número de árboles mejor aprende el modelo. El valor por defecto es 100, pero tras varias pruebas se ha decidido poner 500.

La función de pérdida seleccionada es “lad” (least absolute deviations) porque es una función muy robusta que sólo se basa en el orden de la información de las variables de entrada.

Al realizar el entrenamiento, los resultados de la precisión en los datos de test y en los de validación han sido buenos, lo que ha hecho esperar una buena predicción.

```
GradientBoostingRegressor(learning_rate=0.01, loss='lad', max_depth=7,
                          min_samples_split=5, n_estimators=500)
```

```
print("Accuracy score (training): {0:.3f}".format(clf.score(train_X, train_y)))
print("Accuracy score (validation): {0:.3f}".format(clf.score(test_X, test_y)))
```

```
Accuracy score (training): 0.580
Accuracy score (validation): 0.885
```

Figura 34 - Precisión estimada en Gradient Boosting Regressor

De igual modo se puede observar que la desviación en los datos de entrenamiento y de test desciende cuanto mayor sea el número de iteraciones. En este caso cuanto más se acerca al número de estimadores definidos 500 menor es la desviación.

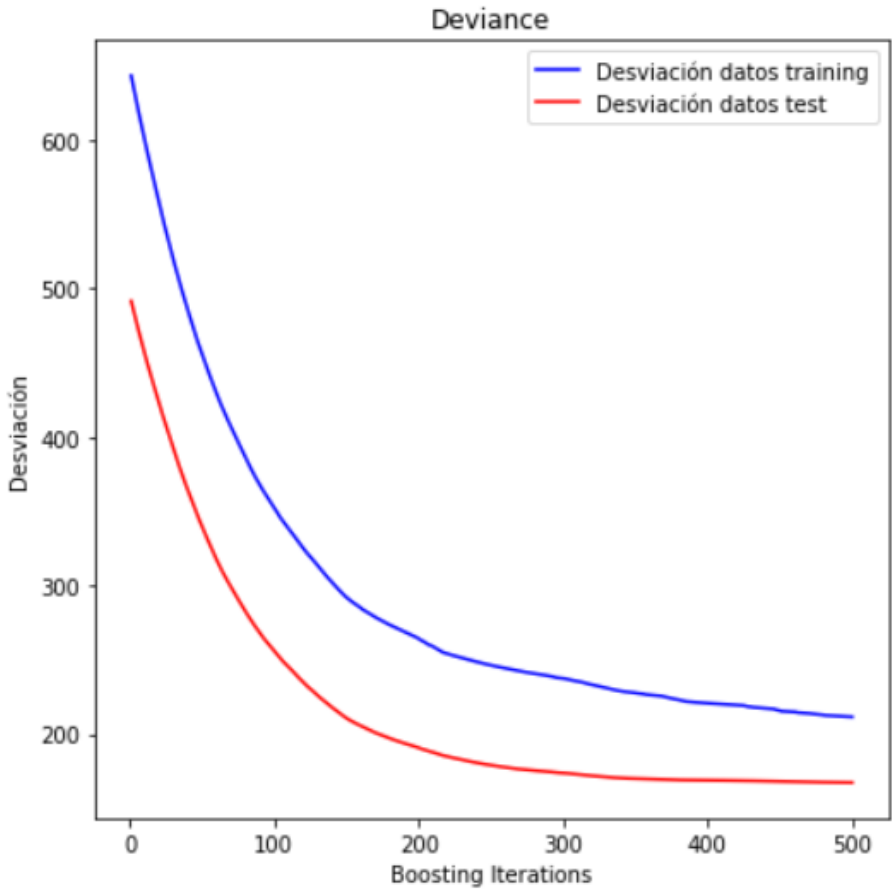

Figura 35 - Desviación train y test en Gradient Boosting

Una vez acabado de definir el modelo se realiza la predicción para los dos próximos meses partiendo de los datos disponibles.

El explained variance score de la predicción es de 0,86.

Viendo la gráfica comparativa entre el valor real y la predicción, se puede apreciar bastante acierto, aunque no es el mismo para todos los productos.

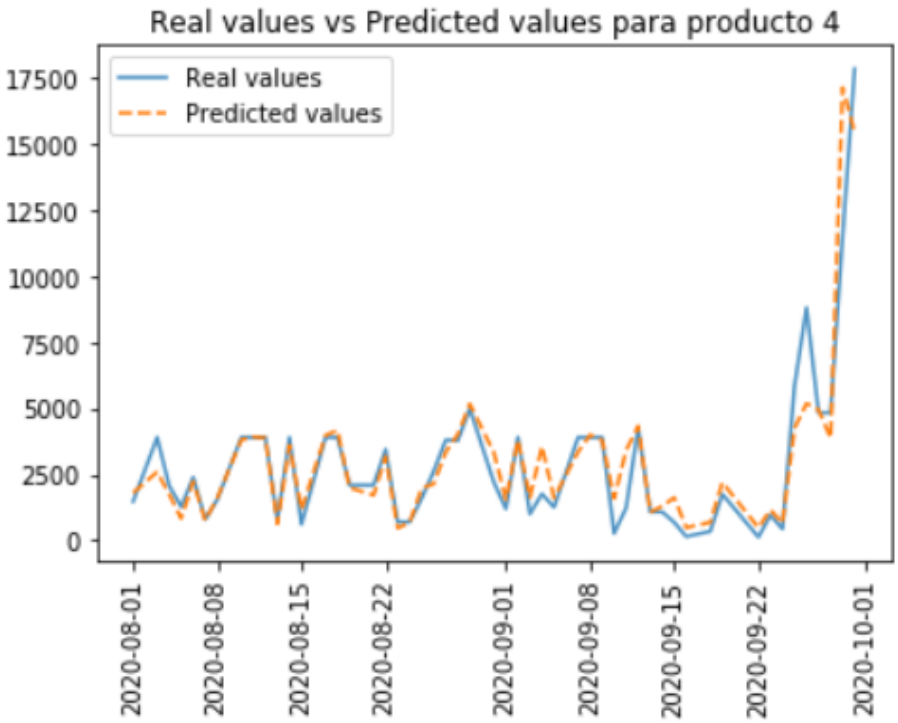

Figura 36 - GBR - Valores reales vs. predicción - Producto 4

Los resultados indican un incremento del stock al tratar de adecuarlo a la demanda.

Fecha	Stock pred.	Stock real	Valor pred.	Valor real
TOTAL	2.397.818	2.346.909	10.369.472,39	10.134.608,55

Tabla 6 - Resultados totales predicción Gradient Boosting Regressor

Se ha producido un aumento de 50.909 unidades de stock que significa un incremento en valor de 234.863,84 €.

3.5.4 Random Forest Regressor

Otro de los métodos predictivos basados en árboles que se ha analizado es Random Forest Regressor.

En muchas ocasiones es un algoritmo que tiene un rendimiento similar al de Boosting, con la ventaja de que es más simple de entrenar y ajustar. Por contra hay ocasiones en que puede tener problemas de overfitting.

Se ha realizado una búsqueda de los parámetros más adecuados mediante GridSearchCV con diferente número de n_estimators y de max_depth.

```
Precisión media: 25.97 +/- 4.91 con parámetros {'max_depth': 2, 'n_estimators': 500}
Precisión media: 25.86 +/- 4.64 con parámetros {'max_depth': 2, 'n_estimators': 1000}
Precisión media: 25.83 +/- 4.73 con parámetros {'max_depth': 2, 'n_estimators': 2000}
Precisión media: 25.93 +/- 4.65 con parámetros {'max_depth': 2, 'n_estimators': 4000}
Precisión media: 48.24 +/- 12.45 con parámetros {'max_depth': 3, 'n_estimators': 500}
Precisión media: 48.59 +/- 12.69 con parámetros {'max_depth': 3, 'n_estimators': 1000}
Precisión media: 48.37 +/- 12.59 con parámetros {'max_depth': 3, 'n_estimators': 2000}
Precisión media: 48.32 +/- 12.83 con parámetros {'max_depth': 3, 'n_estimators': 4000}
Precisión media: 62.19 +/- 15.74 con parámetros {'max_depth': 4, 'n_estimators': 500}
Precisión media: 62.22 +/- 15.88 con parámetros {'max_depth': 4, 'n_estimators': 1000}
Precisión media: 62.13 +/- 15.59 con parámetros {'max_depth': 4, 'n_estimators': 2000}
Precisión media: 62.27 +/- 15.70 con parámetros {'max_depth': 4, 'n_estimators': 4000}
Precisión media: 73.39 +/- 11.40 con parámetros {'max_depth': 5, 'n_estimators': 500}
Precisión media: 72.94 +/- 12.00 con parámetros {'max_depth': 5, 'n_estimators': 1000}
Precisión media: 73.15 +/- 11.78 con parámetros {'max_depth': 5, 'n_estimators': 2000}
Precisión media: 73.20 +/- 11.72 con parámetros {'max_depth': 5, 'n_estimators': 4000}
```

Figura 37 - RF - Búsqueda de parámetros óptimos

Los mejores resultados arrojan que la precisión media puede estar en torno al 84%, lo que es un buen indicador.

Para este algoritmo sí que se ha querido realizar una estandarización de las variables a pesar de que, al ser un algoritmo basado en árboles, no es necesario. De todos modos, se ha pretendido observar si tenía efectos positivos en el resultado del modelo.

Tras hacer el entrenamiento se comprueba que el coeficiente de determinación R^2 de la predicción da un valor de 0,99, lo que indica que el modelo se ajusta mucho a la variable que se está intentando explicar. El problema que se puede tener es que este modelo no penaliza cuando se incluyen variables explicativas que no sean significativas. Es decir, si al modelo se añaden variables explicativas que tienen poca relación con la variable objetivo, el valor de R^2

aumentará. En nuestro caso, si recordamos el heatmap de correlación de variables mostrado cuando se hacía el análisis de las series de los datos de ventas, había algunas variables con una correlación bastante débil y eso podría haber provocado el efecto descrito de aumento de R^2.

El explained variance score de la predicción es de 0,85 e indica un buen ajuste. Comprobamos la predicción para uno de los productos comparándola con el valor real del stock en los dos meses que se predicen.

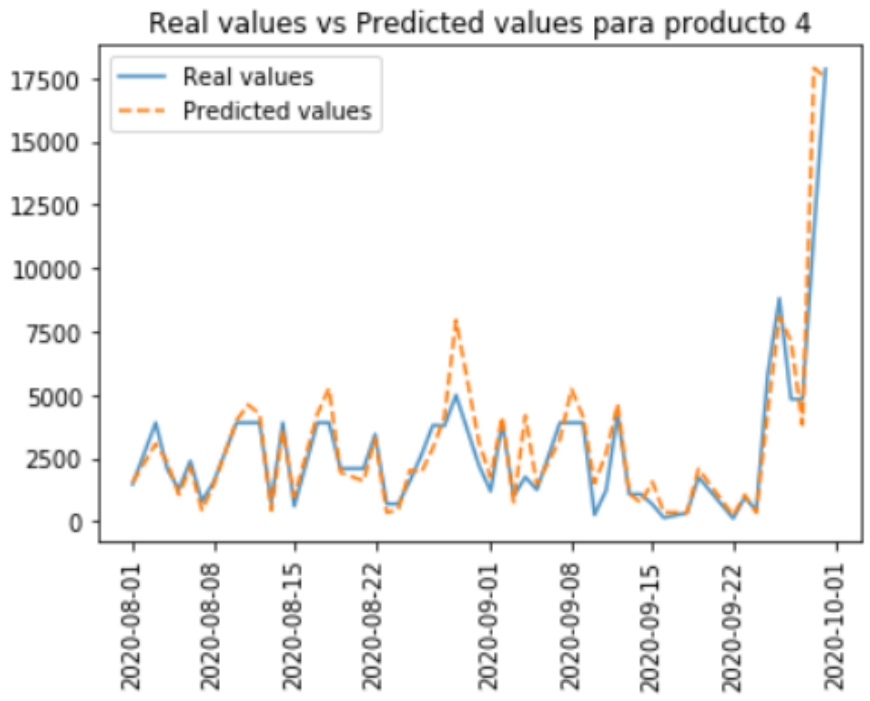

Figura 38 - RF - Valores reales vs. predicción - Producto 4

Veamos los resultados para la predicción de todos los productos para los dos próximos meses.

Fecha	Stock pred.	Stock real	Valor pred.	Valor real
TOTAL	2.509.641	2.346.909	10.793.449,20	10.134.608,55

Tabla 7 - Resultados totales predicción Random Forest Regressor

Se ha producido un aumento de 162.732 unidades de stock que significa un incremento en valor de 658.840,66 €.

3.5.5 Decision Tree Regressor

Un árbol de decisión es un modelo que llega a una predicción después de formular una serie de preguntas a los datos. Cada pregunta reduce los valores posibles hasta que el modelo llega a tener el suficiente conocimiento como para realizar una única predicción. Es el propio modelo el que determina el orden de las preguntas y el contenido de las mismas, y todas ellas esperan una respuesta de verdadero o falso que corresponden a ramas separadas en el árbol.

Drakos (2019) explica que este modelo tiene una serie de ventajas como: que es fácil de entender, que no le afectan demasiado los outliers o los valores nulos y que puede capturar relaciones no lineales. Pero al mismo tiempo, también tiene desventajas como que puede provocar overfitting o que al trabajar con variables numéricas continuas pierde información al categorizar las variables.

En este proyecto se ha probado a construir un modelo basado en Decision Tree Regressor con una profundidad máxima (max_depth) de 15 niveles. Al igual que con Random Forest, se ha realizado un proceso de estandarización con el mismo fin que en el anterior algoritmo.

El resultado ha sido bastante aproximado y el explained variance score es de 0,72. Se comparan los valores reales y la predicción para un producto.

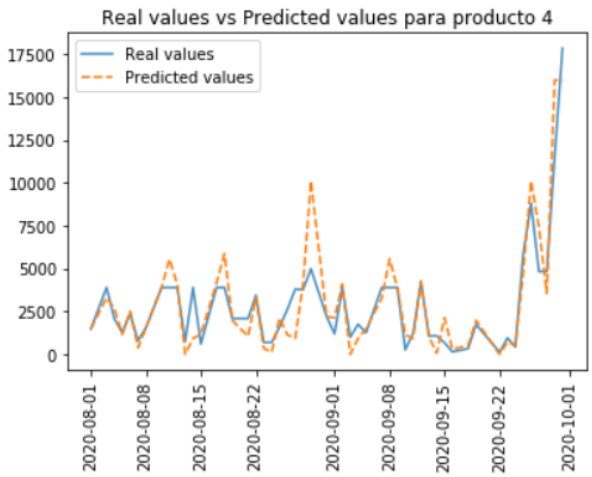

Figura 39 - DT - Valores reales vs. predicción - Producto 4

Los resultados para la predicción de todos los productos para los dos próximos meses lo podemos ver a continuación:

Fecha	Stock pred.	Stock real	Valor pred.	Valor real
TOTAL	2.540.307	2.346.909	10.922.889,67	10.134.608,55

Tabla 8 - DT - Resultados totales predicción Decision Tree Regressor

Se ha producido un aumento de 193.398 unidades de stock que significa un incremento en valor de 788.281,12 €.

3.5.6 Light Gradient Boosting Machine Regressor

Es un modelo muy popular para modelado predictivo estructurado y especialmente efectivo en datos tabulares. Al igual que otros algoritmos de Gradient Boosting también funciona ajustando los árboles de decisión minimizando el gradiente de error.

Este modelo usa también aprendizaje basado en árboles que tiene como ventajas una alta velocidad de aprendizaje, poco uso de memoria, una buena precisión y es capaz de gestionar una gran cantidad de datos. No se ha realizado proceso de estandarización.

En este proyecto se ha usado con los siguientes parámetros:

- n_estimators = 1000
- learning_rate = 0.1
- n_jobs = -1
- random_state = 0

Una vez entrenado el coeficiente de determinación R^2 (score) de la predicción era de 0,9829 lo que indica que el modelo se ajusta bastante bien. El explained variance score es de 0,75.

Veamos el resultado de la predicción de dos meses para uno de los productos.

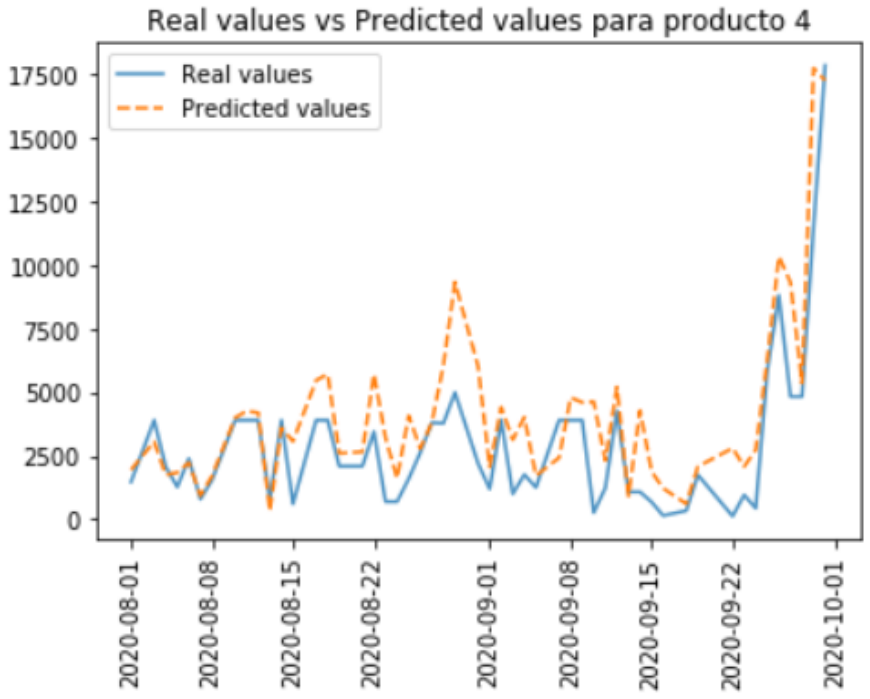

Figura 40 - LGBM - Valores reales vs. predicción - Producto 4

Los resultados para la predicción de todos los productos para los dos próximos meses lo podemos ver a continuación:

Fecha	Stock pred.	Stock real	Valor pred.	Valor real
TOTAL	2.528.728	2.346.909	10.868.300,44	10.134.608,55

Tabla 9 - LGBM - Resultados totales predicción LGBM Regressor

Se ha producido un aumento de 181.819 unidades de stock que significa un incremento en valor de 733.691,89 €.

3.5.7 Extreme Gradient Boosting Regressor

El Extreme Gradient Boosting es una implementación optimizada del algoritmo Gradient Boosting Trees. Se caracteriza por una gran velocidad de cómputo, paralelización y un buen rendimiento.

El algoritmo usa el principio del boosting que consiste en generar modelos de predicción débiles de forma secuencial, de forma que cada modelo recoge los resultados del modelo anterior hasta llegar a crear un modelo fuerte, entendiendo por fuerte uno con un mayor poder de predicción y más estable.

En el entrenamiento del modelo los parámetros se ajustan de forma iterativa buscando reducir el error. Cuando se ha entrenado un modelo, los resultados se comparan con los del modelo anterior y si son mejores, se toma el nuevo modelo como base para ir realizando los ajustes. Si los resultados son peores, se vuelve al mejor anterior y se sigue en base a él. Este proceso se realiza hasta llegar a una diferencia tan pequeña entre modelos que se considera que se ha encontrado el mejor o hasta alcanzar el máximo número de iteraciones que se han definido.

En este proyecto también se prueba el resultado de la predicción con este modelo optimizado. No se realiza proceso de estandarización.

En primer lugar se define el modelo y se comprueba el score obtenido durante el entrenamiento.

- MSE: 226127.21659520385
- RMSE: 475.52835519577997

Usando la función SMAPE (Symmetric mean absolute percentage error), una medida de precisión basada en el porcentaje de errores, se obtiene el siguiente score:

- SMAPE: 30.829875760553715

Con el fin de mejorar el rendimiento se decide realizar un tuning de hiperparámetros mediante una optimización bayesiana.

```
param_grid = {
    'max_depth': [3, 4, 5, 6, 7, 8, 9, 10, 11, 12],
    'min_child_weight': np.arange(0.0001, 0.5, 0.001),
    'gamma': np.arange(0.0,40.0,0.005),
    'learning_rate': np.arange(0.0005,0.3,0.0005),
    'subsample': np.arange(0.01,1.0,0.01),}
```

Figura 41 - XGB - Valores para tuning hiperparámetros

Se realiza el tuning con los rangos definidos en la imagen anterior y una vez obtenida la mejor combinación de hiperparámetros se vuelve a comprobar el score obtenido.

- SMAPE (train): 40.28476322881612
- SMAPE (validation): 41.04573861297119

Posteriormente se realiza ya la predicción sobre los datos de test obteniendo los siguientes resultados en forma gráfica para el total de los productos en los dos meses a predecir.

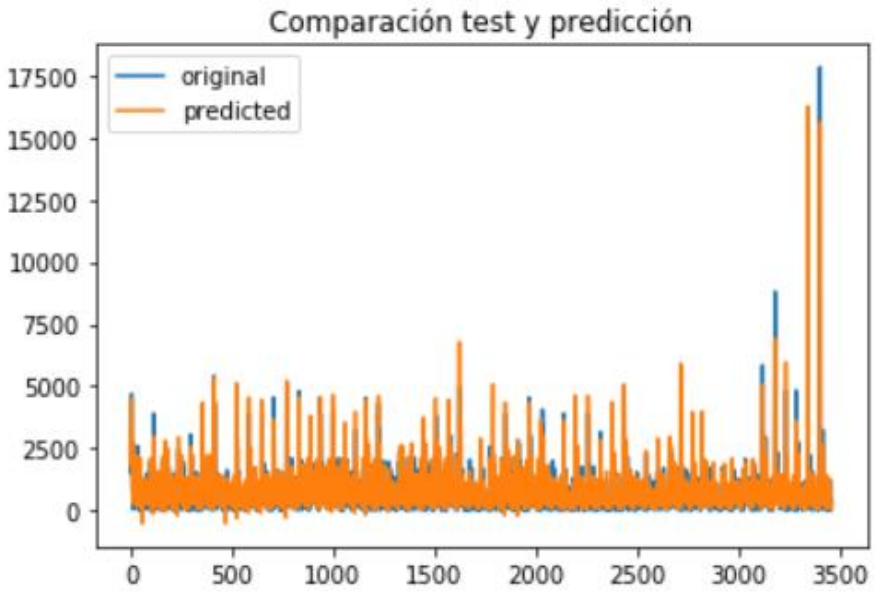

Figura 42 - XGB - Comparación real vs. predicción todos los productos

En el gráfico obtenido se puede observar bastante superposición de líneas aunque no completa, ya que se observan diferencias en las líneas de colores de la predicción que sobresalen por encima de los datos reales. En este caso

se debería considerar algún ajuste más que permitiera una mayor precisión en los resultados. El explained variance score de la predicción es de 0,80.

Seguidamente se muestra la comparativa de la predicción vs. los datos reales de algunos productos. En las diferencias que se muestren hay que pensar en el efecto de las ventas sobre la predicción de stock a futuro, para conseguir adecuar el stock a la demanda.

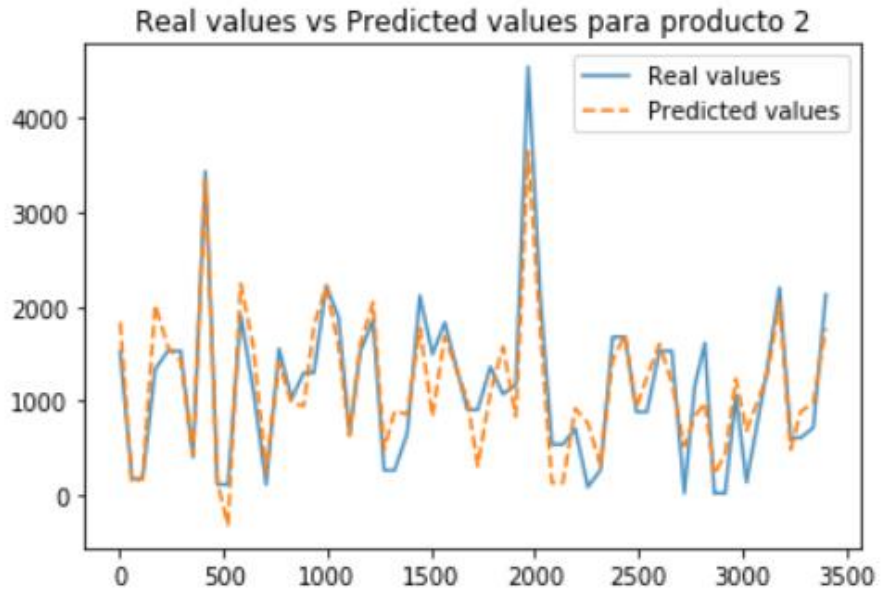

Figura 43 - XGB - Valores reales vs. predicción - Producto 2

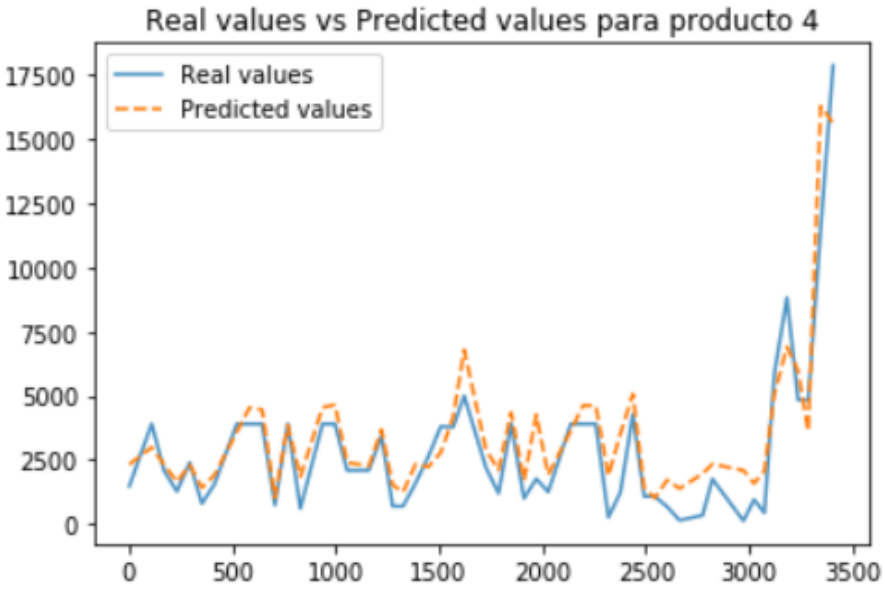

Figura 44 - XGB - Valores reales vs. predicción - Producto 4

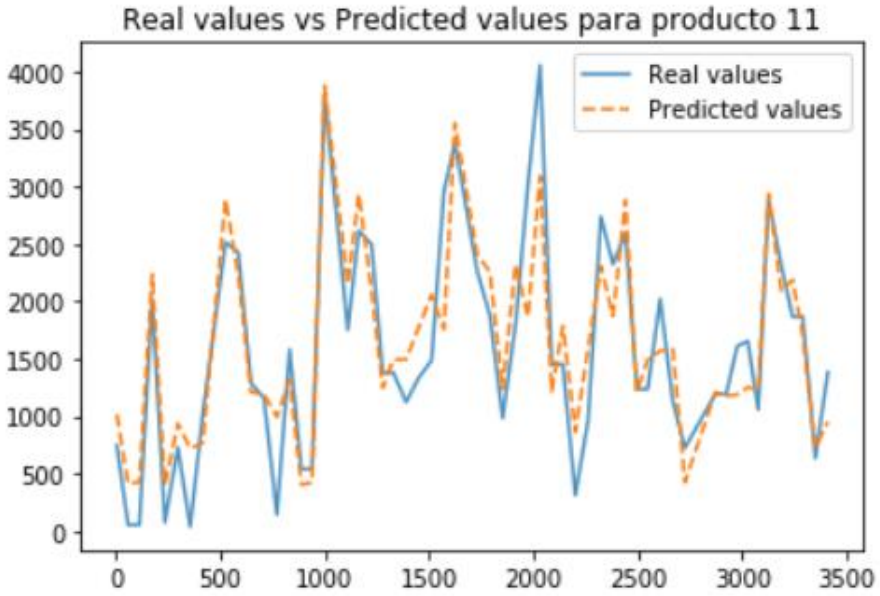

Figura 45 - XGB - Valores reales vs. predicción - Producto 11

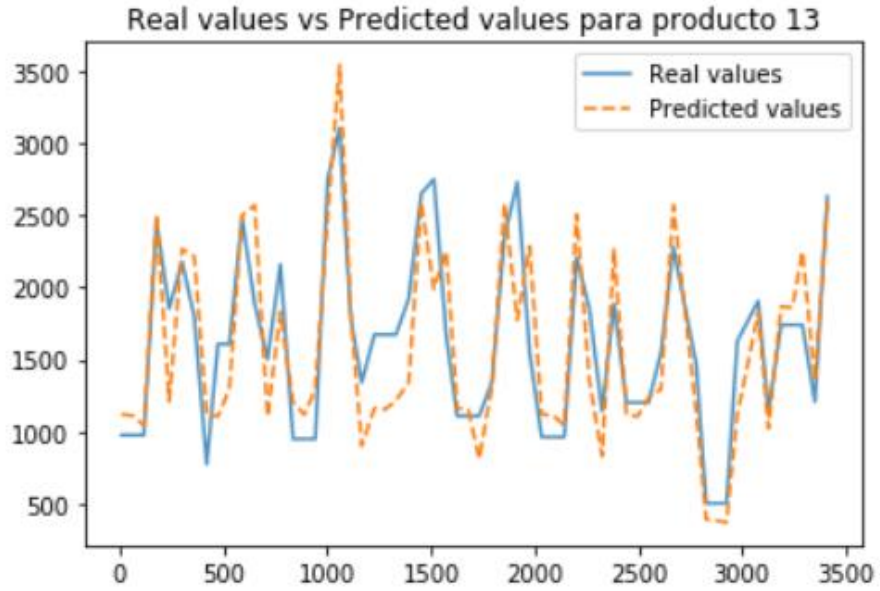

Figura 46 - XGB - Valores reales vs. predicción - Producto 13

Tal como se ha dicho existen diferencias no demasiado grandes pero debería afinarse el modelo para lograr una buena precisión que además permitiera ajustar el stock a la demanda.

Los resultados para la predicción de todos los productos para los dos próximos meses se puede ver a continuación:

Fecha	Stock pred.	Stock real	Valor pred.	Valor real
TOTAL	2.507.002	2.346.909	10.812.053,34	10.134.608,55

Tabla 10 - XGB - Resultados totales predicción XGB Regressor

Se ha producido un aumento de 160.093 unidades de stock que significa un incremento en valor de 677.444,80 €.

3.5.8 Análisis comparativo de resultados

A continuación se muestra una tabla comparativa de los resultados obtenidos con los diversos métodos, en cuanto a variación de unidades y de valor del stock, ordenados de menor a mayor % variación uds. en valor absoluto.

Método	Explained variance score	Var. uds. (abs)	% Var. uds. (abs)	Var. valor (€) (abs)	% Var. valor (abs)
Gradient Boosting	0,86	50.909	2,17	234.863,80	2,32
Extreme GB	0,80	160.093	6,82	677.444,80	6,68
Red Neuronal	0,82	161.043	6,86	840.021,72	8,29
Random Forest	0,85	162.732	6,93	658.840,70	6,50
Light GB	0,75	181.819	7,75	733.691,90	7,24
Decision Tree	0,72	193.398	8,24	788.288,10	7,78

Tabla 11 - Comparativa resultados algoritmos ML

Gradient Boosting es el método que mejor ajusta a la serie histórica de stock, es decir, el que tiene una desviación menor.

La red neuronal, que aparece en tercer lugar, es el único método cuya variación de unidades y de valor ha resultado con signo negativo, por lo que ha reducido el stock y su valor asociado. En la tabla se representa en valor absoluto ya que se quiere mostrar la desviación respecto a la serie histórica.

El ajuste del stock a la demanda puede suponer la reducción del stock de unos productos y el incremento de otros. Las cifras mostradas en la tabla muestran la variación de los datos resultantes de la predicción con los diferentes métodos sobre el inventario real para el período considerado de dos meses.

El *Explained variance score* explica la dispersión de los errores de un conjunto de datos, es decir, mide la discrepancia entre la predicción y los datos reales. Cuanto más cercano a 1 mejor es la predicción. En la tabla podemos ver que el orden de métodos mostrado guarda bastante relación con el citado indicador y por lo tanto, se puede afirmar que el *Explained variance score* es un indicador bastante confiable para decidir el método de predicción a usar.

4. Conclusiones

4.1 Lecciones aprendidas

Este proyecto ha sido diferente a todos los problemas resueltos a lo largo de toda la duración del máster. Se ha presentado un caso real con una casuística que se puede encontrar en multitud de empresas. Diversas fuentes de datos desparejadas, con datos no totalmente coordinados, en diferentes períodos de tiempo, etc. Esto ha provocado un arduo trabajo de conexión de datos para lograr un conjunto de datos lo suficientemente homogéneo con el que poder trabajar.

Se ha podido comprobar la importancia que tiene la parte de preparación, limpieza y preprocesado de los datos. Sin esta fase no sería posible hacer ningún análisis y ni mucho menos, una predicción a futuro. El tiempo dedicado a esta etapa del proyecto es un tiempo bien empleado y hay que considerar de cara a proyectos futuros, asignarle el tiempo necesario en la planificación para que después se pueda obtener un buen resultado.

Otra lección aprendida es la importancia que se debe dar a la revisión de la literatura sobre el tema. Ello permite observar los trabajos realizados previamente y los logros conseguidos, lo cual permite focalizar hacia donde se quiere dirigir el trabajo y cuáles algoritmos son los que pueden permitir conseguir mejores resultados. Además, la revisión de la bibliografía, es una fuente inagotable de información que permite ampliar en gran manera el conocimiento y produce esa sensación de tener curiosidad por conocer más.

4.2 Logro de los objetivos planteados

En este proyecto se puede decir que se ha conseguido cumplir con los objetivos planteados al inicio del mismo.

Se ha conseguido el objetivo principal, construir un modelo predictivo basado en algoritmos de machine learning que permite pronosticar el stock futuro óptimo de cada material en un solo paso, a fin de que los costes derivados del stock puedan reducirse.

Se han conseguido también los objetivos secundarios especificados en el capítulo 1 de este documento.

Y finalmente, se ha conseguido obtener con uno de los métodos analizados, Gradient Boosting, una predicción del stock futuro en un sólo paso, adecuando el stock a la demanda con sólo un 2,17% de error.

Como parte crítica, se puede apuntar que hubiera sido deseable que con los otros algoritmos probados, se consiguiera una predicción que permitiera ajustar el stock a la demanda con un porcentaje menor de error. Es algo que quedará para trabajos futuros.

4.3 Planificación y metodología

La planificación realizada al inicio del proyecto era clara y lógica. Al entrar de lleno en el proyecto se ha podido observar que algunas de las etapas definidas han requerido más tiempo del inicialmente previsto.

La revisión de la literatura ha sido una de ellas. Existen muchos documentos interesantes que leer para que después sean útiles para enfocar el proyecto, pero el tiempo disponible es algo limitado al tener también una serie de obligaciones laborales y familiares que atender.

La preparación de datos, su limpieza y el preprocesamiento también han tomado más tiempo del inicialmente previsto y en ocasiones se ha tenido que volver a repetir tareas que ya se creían superadas. Es como el mecanismo de backpropagation, volver atrás con el error para hacerlo de nuevo de forma correcta.

Las pruebas con los diferentes algoritmos han sido más o menos como se podían esperar, aunque en algunos casos hubiera sido deseable también disponer de más tiempo para afinar más los modelos.

4.4 Líneas de trabajo futuro

Considero que hay dos líneas básicas para el trabajo futuro:

- Conseguir una mayor precisión en el modelo de redes neuronales para poder mejorar el resultado obtenido. Ha sido el modelo que ha permitido ajustar el stock a la demanda obteniendo una reducción en el valor del inventario, pero su porcentaje de error ha sido mayor que con el método de Gradient Boosting. Se puede intentar afinar los parámetros, incluida la estructura de la red neuronal (número de capas, etc.), para conseguir mejores resultados.

- Lograr que el resto de algoritmos basados en árboles permitan también conseguir el objetivo inicial del proyecto. Para ello, es necesario profundizar más en el conocimiento de los hiperparámetros hasta asegurar el punto óptimo de afinación. Sería deseable alcanzar un porcentaje de error como el logrado con Gradient Boosting.

Será muy interesante ver la consecución de estos objetivos en el futuro.

5. Bibliografía

- Amat, J. (2020). *Gradient Boosting con Python*. Recuperado de: https://www.cienciadedatos.net/documentos/py09_gradient_boosting_python.html

- Cardona Tunubala, J.L., Orejuela Cabrera, J.P. y Rojas Trejos, C.A. (2018). Gestión de inventario y almacenamiento de materias primas en el sector de alimentos concentrados. *Revista EIA, 15(30), 195-208. Disponible en https://doi.org/10.24050/reia.v15i30.1066*

- de Santis, R. B., de Aguiar, E. P., and Goliatt, L. (2017). Predicting material backorders in inventory management using machine learning in 2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI) in Arequipa, Peru: Nov. 8-10, 2017, pp. 1-6, doi: 10.1109/LA-CCI.2017.8285684. Disponible en https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwi0mvfx2rvsAhURaBoKHWrSCKoQFjADegQIBxAC&url=https%3A%2F%2Fwww.researchgate.net%2Fpublication%2F319553365_Predicting_Material_Backorders_in_Inventory_Management_using_Machine_Learning&usg=AOvVaw2AuubQy3JFXsAtuVEdrwVj

- Drakos, G. (2019). Decision Tree Regressor explained in depth. Recuperado de https://gdcoder.com/decision-tree-regressor-explained-in-depth/

- García, J. P. (2011). Modelos de Gestión de Stocks: Cuándo y Cuánto Comprar. Recuperado de http://personales.upv.es/jpgarcia/LinkedDocuments/GnStocks.pdf

- He, W. (2013). An Inventory Controlled Supply Chain Model Based on Improved BP Neural Network. *Hindawi Publishing Corporation, Discrete Dynamics in Nature and Society, 2013, Article ID 537675. Disponible en http://downloads.hindawi.com/journals/ddns/2013/537675.pdf*

- Khalilzadeh, N. (2008). *Sales Prediction for Pharmaceutical Distribution Companies – A Data Mining Based Approach* (Master's Thesis). Luleå University of Technology, Sweden. Recuperado de https://www.diva-portal.org/smash/get/diva2:1019972/FULLTEXT02

- Li, S. (2019). *Time Series of Price Anomaly Detection*. Recuperado de: https://towardsdatascience.com/time-series-of-price-anomaly-detection-13586cd5ff46

- Nieto, Y. P. (2014). Aplicación de diferentes modelos para determinar el stock de seguridad óptimo en una empresa distribuidora. Recuperado de: http://hdl.handle.net/10654/10980

- Peterson, R. y Silver, E. A. (1979). *Decision Systems for Inventory Management and Production Planning*. New York: John Wiley & Sons.

- Praveen, K B et al., (2020). Inventory Management using Machine Learning. *International Journal of Engineering Research and Technology (IJERT), 9(06). Disponible en https://www.ijert.org/research/inventory-management-using-machine-learning-IJERTV9IS060661.pdf*

- Praveen, U., Farnaz, G., y Hatim, G., (2019). Inventory Management and cost reduction of supply chain processes using AI based time-series forecasting and ANN modelling in 29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2019) in Limerick, Ireland: June 24-28, 2019. *Procedia Manufacturing 38 (2019), 256-263. Disponible en https://www.sciencedirect.com/science/article/pii/S2351978920300354/pdf?md5=9d85632f37e2f793dfc609beb14d662a&pid=1-s2.0-S2351978920300354-main.pdf*

- Puente, F.J., Priore, P., Pino, R. y de la Fuente, D. (2003). La Clasificación ABC Multiatributo de Inventarios con Técnicas de Inteligencia Artificial en *V Congreso de Ingeniería de Organización en Valladolid-Burgos: 4-5 de septiembre de 2003*. Recuperado de http://adingor.es/congresos/web/uploads/cio/cio2003/Art_112.pdf

- Quijaite, J. J. S., y Apaza, G. M. (2014). Modelo de Simulación de Inventario basado en Redes Neuronales Artificiales Supervisadas y Algoritmos Genéticos para Optimizar el Stock de Medicamentos de la Clínica Ricardo Palma. *Revista Científica Ingetecno, 2(1). Disponible en http://revistas.uap.edu.pe/ojs/index.php/RCI/article/view/176/121*

- Ravinder, H. y Misra, R. B. (2014). ABC Analysis for Inventory Management: Bridging the Gap between Research and Classroom. *American Journal of Business Education – Third Quarter 2014, 7(3)*, 257-263. Recuperado de https://files.eric.ed.gov/fulltext/EJ1053624.pdf

- Salameh, M. K., Abdul-Malak, M.U., Jaber, M.Y. (1993). Mathematical modelling of the effect of human learning in the finite production inventory model. *Applied Mathematical Modelling, 17(11)*, 613-615. Recuperado de https://www.sciencedirect.com/science/article/pii/0307904X9390070W

- Shoujing Zhang, Xiaofan Qin, Sheng Hu, Qing Zhang, Bochao Dong, Jiangbin Zhao (2020). Importance Degree Evaluation of Spare Parts Based on Clustering Algorithm and Back-Propagation Neural Network, Mathematical Problems in Engineering, vol. 2020, Article ID 6161825, 13 pages, 2020. Disponible en http://downloads.hindawi.com/journals/mpe/2020/6161825.pdf

- Šustrová, T. (2016). A Suitable Artificial Intelligence Model for Inventory Level Optimization. *Brno University of Technology. Disponible en https://trends.fbm.vutbr.cz/index.php/trends/article/download/344/293*