Universidad de Matanzas

Facultad de Ciencias Técnicas

Departamento de Informática

TRABAJO DE DIPLOMA EN OPCIÓN DEL TÍTULO DE INGENIERO INFORMÁTICO

SISTEMA DE ANÁLISIS DE VENTAS SOPORTADO EN ALGORITMOS DE MINERÍA DE DATOS

Autor: Yasniel del Pino Martín

Tutores: Dr.C. Liz Pérez Martínez

Ing. Ramsey Ricardo Busto Martínez

Matanzas, 2023

Resumen

El tema del presente estudio es el análisis de ventas del bar "Bacardi" soportado en algoritmos de minería de datos. En un entorno empresarial cada vez más competitivo, es fundamental utilizar técnicas de análisis de datos, lo cual es crucial para tomar decisiones informadas y diseñar estrategias efectivas para obtener información valiosa que ayude a impulsar el crecimiento y la rentabilidad de un negocio. El objetivo general es Desarrollar algoritmos para el análisis de ventas en el bar "Bacardí", para ello se utilizaron los algoritmos prophet y apriori. Como resultado se desarrollaron algoritmos capaces de predecir las ventas futuras y lograr la identificación de los productos en combinación que más se piden constituyendo un instrumento para la toma de decisiones de los directivos que laboran en el bar “Bacardí”.

Palabras claves: Análisis de ventas, minería de datos, bares, análisis de datos, predicción, reglas de asociación, Python.

Abstract

The subject of this study is the sales analysis of the "Bacardi" bar supported by data mining algorithms. In an increasingly competitive business environment, it is essential to use data analysis techniques, which is crucial to make informed decisions and design effective strategies to obtain valuable information to help drive the growth and profitability of a business. The general objective is to develop algorithms for sales analysis in the bar "Bacardi", for this purpose the prophet and apriori algorithms were used. As a result, algorithms capable of predicting future sales and identifying the products in combination that are most in demand were developed, thus constituting a decision-making tool for the managers working at the "Bacardi" bar.

Keyworks: Sales analysis, data mining, bars, data analysis, prediction, association rules, Python.

Índice de Contenidos

Introducción

En la era actual de la información, las empresas se enfrentan al desafío de gestionar grandes volúmenes de datos generados por sus operaciones diarias. El análisis de ventas es una práctica esencial para el éxito de cualquier negocio, incluido el sector de la hospitalidad. Los bares y restaurantes buscan constantemente formas de mejorar sus estrategias de ventas y maximizar sus ingresos. En este contexto, la Minería de Datos ha surgido como una herramienta valiosa para analizar grandes volúmenes de datos y extraer información relevante que puede impulsar la toma de decisiones.

El apoyo a la toma de decisiones significa ayudar a los niveles gerenciales de las organizaciones a reunir inteligencia, generar alternativas y tomar decisiones, contribuyendo a la estimación, la evaluación y/o la comparación de alternativas (López de Munain et al. 2014).

La utilización de algoritmos de minería de datos en el análisis de ventas del bar "Bacardí" permitirá identificar patrones y tendencias ocultas que pueden mejorar la toma de decisiones y optimizar el rendimiento del negocio. Con estas técnicas se podrán hacer predicciones de las ventas de un día o un determinado tiempo, además, de poder predecir las cantidades de ventas de un determinado producto, permitiendo una mejor gestión de inventario y una mayor rentabilidad para el negocio.

Sin embargo, en el contexto específico de los bares y restaurantes, la adopción de sistemas de análisis de ventas impulsados por minería de datos aún no se ha generalizado en nuestro país. Aunque existen diversas herramientas y tecnologías disponibles, gran parte de estos establecimientos no aprovechan las ventajas de estos sistemas para optimizar sus operaciones y estrategias comerciales.

Lo anteriormente descrito deriva en el siguiente **problema científico**: ¿cómo desarrollar algoritmos de minería de datos que permitan analizar las ventas y predecir comportamientos futuros en el bar "Bacardí"?

Para dar solución al problema antes expuesto se traza como **objetivo general:** desarrollar algoritmos para el análisis y la predicción del comportamiento de ventas en el bar "Bacardí"

Los **objetivos específicos** definidos para dar cumplimiento al objetivo general son:

1. Analizar los trabajos más importantes relacionados con algoritmos para el análisis de las ventas.
2. Diseñar las propuestas de los algoritmos.
3. Implementar las propuestas de los algoritmos.
4. Validar las propuestas mediante la realización de experimentos.

Para el desarrollo de la investigación se utilizaron diversos **métodos y técnicas** tales como:

- Dentro de los métodos teóricos:
  - Método de **análisis histórico – lógico**: permitió estudiar la trayectoria y desarrollo de los algoritmos para el análisis de ventas existentes.
  - Método de **análisis y síntesis**: este se precisó durante la revisión bibliográfica y el análisis de los resultados, permitiendo descomponer lo complejo en sus partes y cualidades, la división del todo en sus múltiples relaciones para luego unir las partes analizadas, descubrir las relaciones y características generales entre ellas.
  - Método **inductivo - deductivo**: su uso fue necesario tanto en la revisión bibliográfica, como en el análisis de los resultados, permitiendo arribar a conclusiones que se infirieron a partir de propiedades y relaciones existentes entre los elementos que conforman el fenómeno objeto de estudio.
- Como métodos empíricos, utilizados por medio de las siguientes técnicas:
  - **Observación**: permitió entender lo que verdaderamente se necesitaba para el análisis de ventas y se obtuvo la información primaria acerca de lo investigado.

- **Entrevistas**: aportaron datos esenciales a la investigación puesto que el entrevistado es la persona dueña del establecimiento, por lo que conoce bien el negocio.

Entre los **aportes** de la investigación se destacan:

- el **teórico-investigativo**, al integrar los algoritmos tradicionales más utilizados por autores relacionados con el análisis de ventas a través de diferentes pasos que permiten orientar metodológicamente la secuencia de acciones lógicas a desarrollar para la conformación de los algoritmos; y los elementos a tener en cuenta para la continuidad de la investigación.
- el **práctico**, al desarrollar una herramienta informática que asista la toma de decisiones y viabilice la actividad.
- el **económico**, al elaborar algoritmos capaces de predecir cómo se comportarán las ventas, posibilitando tomar decisiones que maximicen la gestión de las ventas.

Resultados esperados:

Algoritmos que apoyen en el proceso de análisis de las ventas, realizando predicciones e identificando las combinaciones de productos que más se piden. Obtención de una herramienta que constituya un instrumento para la toma de decisiones de los directivos que laboran en el bar “Bacardí”.

Atendiendo a lo planteado anteriormente, la investigación queda estructurada en tres capítulos, conclusiones, recomendaciones y referencias bibliográficas, según sigue:

- Un primer capítulo donde se recoge el marco teórico referencial del tema y los principales conceptos que constituyen la base teórica de la investigación, así como el análisis de las principales tendencias tecnológicas y el estudio de los antecedentes que enmarcan la problemática planteada.
- Un capítulo segundo donde se diseña la propuesta de solución a partir de la conformación de los modelos predictivos y asociativos, sobre la base de lo analizado en el capítulo primero.

- Un tercer capítulo donde se analizan los resultados obtenidos para el pronóstico de indicadores a partir de los pronósticos obtenidos.
- Un apartado de conclusiones donde se verifica el cumplimiento de los objetivos trazados al inicio de la investigación.
- Las recomendaciones en la cual se plasman una serie de propuestas encaminadas a la continuidad de esta investigación.
- Y las referencias de la bibliografía citada.

Capítulo I. Marco Teórico Referencial

1.1. Introducción del Capítulo

Para la realización de la investigación fue necesario el estudio de diferentes conceptos y recursos que resultaron de gran utilidad para realizar nuestra propuesta. En este capítulo se abordan los aspectos teóricos que ayudarán a entender la problemática planteada y que se tuvieron en cuenta para proponer la solución. Se presentan diferentes modelos, algoritmos y recursos que han sido combinados para realizar tareas de minería de datos. Se dará una breve explicación de todos los recursos utilizados, sus características y funcionamiento.

1.2. Antecedentes

La minería de datos es un proceso ampliamente utilizado, su uso va en ascenso exponencial dado que con el pasar de los años se acumulan cada vez más datos. Diversos trabajos similares a esta investigación se han realizado en diferentes campos de aplicación.

Modelo predictivo basado en Machine Learning dirigido a PYMES de venta, caso de estudio Bluefields. Entre las funcionalidades que presenta el modelo, están registrar y almacenar datos, brindar información de los procesos de negocio, realizar cálculos y predicciones, generar sugerencias de paquetes de productos e informes de la información procesada, de esta manera pretende garantizar una mejor organización y aprovechamiento de los diferentes recursos que posee el negocio (Pérez, Cuthbert, Sambola 2022).

- Predicción de la demanda usando modelos de machine learning. Para aquellas empresas dedicadas a la venta en retail o venta directa donde su portafolio de productos es muy amplio, la planeación de la demanda se convierte en un área determinante para la correcta administración del flujo de caja, rentabilidad y efectividad en ventas por varias razones: la primera de ellas es la gestión de compra de insumos por medio de negociación de precio por volumen con sus proveedores; el control de inventario donde se cuide un equilibrio entre uso efectivo del espacio de almacenamiento y reducción de obsolescencia contra la disponibilidad para distribución y por último en la

venta efectiva respetando las estacionalidades, tendencias del mercado y satisfacción del cliente. (Hincapié Herrera 2021)

- Análisis de series de tiempo en el pronóstico de almacenamiento de productos perecederos. Los autores realizaron un estudio sobre la relevancia de incorporar pronósticos en la demanda de almacenamiento en productos perecederos dentro de la cadena de frío deriva de su importancia económica y social. Este caso de estudio presenta una empresa con tendencia de crecimiento dedicada al almacenamiento de productos perecederos e incorpora técnicas de pronósticos de series de tiempo, en el volumen de ingreso y egreso de los productos en una cámara frigorífica, con el fin de estimar el volumen de almacenamiento para prever los requerimientos de instalaciones adicionales, personal y materiales necesarios para la movilidad de los productos. (Juárez, 2016)

- Predicción de ventas mediante algoritmos de aprendizaje automático. Se realiza un estudio exhaustivo de la predicción de ventas utilizando modelos de aprendizaje automático como la regresión lineal, K Vecinos más cercanos, el regresor XGBoost y el regresor de bosques aleatorios. La predicción incluye parámetros de datos como el peso del elemento, el contenido de grasa del elemento, la visibilidad del elemento, el tipo de elemento, el MRP del elemento, el año de establecimiento del punto de venta, el tamaño del punto de venta y el tipo de ubicación del punto de venta. (Bajaj et al. 2020)

- Weka es un acrónimo de Waikato Environment for Knowledge Analysis, es un entorno para experimentación de análisis de datos que permite aplicar, analizar y evaluar las técnicas más relevantes de análisis de datos, principalmente las provenientes del aprendizaje automático, sobre cualquier conjunto de datos del usuario. WEKA se distribuye como software de libre distribución desarrollado en Java. Está constituido por una serie de paquetes de código abierto con diferentes técnicas de preprocesado, clasificación, agrupamiento, asociación, y visualización, así como facilidades para su aplicación y análisis de prestaciones cuando son aplicadas a los datos de entrada seleccionados. (Corso 2009)

Existen varias herramientas de visualización de datos que se pudieran utilizar para un mejor análisis de datos, algunas de estas herramientas son las siguientes.

- Microsoft Power BI es la solución destinada a la inteligencia empresarial, que permite unir diferentes fuentes de datos (más de 65), modelizar y analizar datos para después, presentarlos a través de paneles e informes; que puedan ser consultarlos de una manera muy fácil, atractiva e intuitiva. (Rivera Resina 2018)
- Looker Studio es una herramienta de visualización de datos en línea desarrollada por Google. Se pueden crear paneles personalizados y visualizaciones interactivas que se actualizan automáticamente a medida que cambian los datos subyacentes. Como desventaja tiene que por ser una herramienta en línea no se puede conectar a fuentes de datos de origen local, además puede carecer de algunas funciones avanzadas de análisis y visualización que se encuentran en otras herramientas más especializadas, lo que puede ser una limitación para proyectos más complejos.

1.3. Minería de datos

La minería de datos se sitúa en la confluencia de la estadística, la informática y la inteligencia artificial, con el propósito fundamental de desentrañar información significativa en conjuntos masivos de datos. Este campo se enfoca en la detección de información procesable, lo que implica identificar patrones, relaciones y conocimientos ocultos en los datos recopilados.

El término "minería de datos" ha ganado popularidad, aunque a menudo se usa incorrectamente en contextos que abarcan el manejo de grandes cantidades de datos sin una comprensión profunda de su propósito central. En la esencia de la minería de datos reside el concepto de "descubrimiento", que se refiere a la capacidad de encontrar elementos novedosos o significativos en los datos. Esto se logra a través de rigurosos métodos de análisis matemático y estadístico, que permiten identificar patrones y tendencias que, de otro modo, serían difíciles o imposibles de percibir mediante métodos de exploración convencionales.

En la medida que los negocios mundiales se han hecho más competitivos, los datos cada vez cobran más vida y se han convertido en información vital y estratégica para la toma de decisiones. En tal sentido, las empresas han venido evolucionando y han querido agregarle valor a la gran cantidad de información que tienen almacenada en sus bases de datos. Para ello, se han interesado en automatizar los procesos y poder así descubrir información valiosa, que de otra manera seguiría siendo subutilizada o simplemente desperdiciada. (Martínez 2001)

Un rasgo distintivo de la minería de datos es su capacidad para abordar datos complejos y voluminosos que sobrepasan las capacidades de las técnicas tradicionales de análisis. Esto se debe a la sofisticación de los algoritmos y herramientas de procesamiento que se utilizan en este campo.

El propósito general de la minería de datos es transformar datos en conocimiento valioso. Esto implica no solo la identificación de patrones, sino también la comprensión de su relevancia y su aplicabilidad en una variedad de contextos, desde la toma de decisiones empresariales hasta la investigación científica. La minería de datos es una disciplina esencial en un mundo impulsado por la recopilación masiva de datos, que proporciona una base sólida para la toma de decisiones informadas y el avance en diversas áreas de estudio y aplicación. Para entender su significado es imprescindible conocer los términos relacionados en esta:

- Datos**:** En el ámbito de la informática, los datos representan cualquier tipo de información, número o texto que es susceptible de ser procesado por una computadora. En la actualidad, las organizaciones están acumulando volúmenes cada vez mayores de datos, en diversos formatos y almacenados en múltiples bases de datos.
- Información**:** La verdadera riqueza de los datos radica en la capacidad de identificar patrones, asociaciones y relaciones entre ellos. Esto permite extraer información valiosa. Por ejemplo, el análisis de datos de transacciones en puntos de venta puede proporcionar información acerca de qué productos se venden y cuándo se venden.

- Conocimiento: la información puede ser convertida en conocimiento acerca de los patrones históricos y las tendencias futuras. Por ejemplo, la información resumida sobre las ventas de supermercados minoristas puede ser analizada a la luz de los esfuerzos de promoción para facilitar el conocimiento del comportamiento de compra del consumidor. Por lo tanto, un fabricante o distribuidor puede determinar qué elementos son los más susceptibles a los esfuerzos de promoción.

Según (Riquelme Santos, Ruiz, Gilbert 2006), “los datos en bruto raramente son beneficiosos directamente”. Su verdadero valor se basa en: (a) la habilidad para extraer información útil la toma de decisiones o la exploración, y (b) la comprensión del fenómeno gobernante en la fuente de datos.

Ciertamente, la minería de datos bebe de la estadística, de la que toma las siguientes técnicas (Oded, M.; Lior, R., 2010):

- Análisis de varianza, mediante el cual se evalúa la existencia de diferencias significativas entre las medias de una o más variables continuas en poblaciones distintas.
- Regresión: define la relación entre una o más variables y un conjunto de variables predictoras de las primeras.
- Análisis de agrupamiento o clustering: permite la clasificación de una población de individuos caracterizados por múltiples atributos (binarios, cualitativos o cuantitativos) en un número determinado de grupos, con base en las semejanzas o diferencias de los individuos.
- Análisis discriminante: permite la clasificación de individuos en grupos que previamente se han establecido, permite encontrar la regla de clasificación de los elementos de estos grupos, y por tanto una mejor identificación de cuáles son las variables que definan la pertenencia al grupo.
- Series de tiempo: permite el estudio de la evolución de una variable a través del tiempo para poder realizar predicciones, a partir de ese conocimiento y bajo el supuesto de que no van a producirse cambios estructurales.

Un proceso típico de minería de datos consta de los siguientes pasos generales:

1- Selección del conjunto de datos, tanto en lo que se refiere a las variables objetivo (aquellas que se quiere predecir, calcular o inferir), como a las variables independientes (las que sirven para hacer el cálculo o proceso), como posiblemente al muestreo de los registros disponibles.
2- Análisis exploratorio de los datos, en especial los histogramas, diagramas de dispersión, presencia de valores atípicos y ausencia de datos (valores nulos).
3- Transformación de los datos, se realizará de diversas formas en función del análisis previo, con el objetivo de prepararlo para aplicar la técnica de minería de datos que mejor se adapte a los datos y al problema, a este paso también se le conoce como preprocesamiento de los datos.
4- Seleccionar y aplicar la técnica de minería de datos, se construye el modelo predictivo, de clasificación o asociación.
5- Extracción de conocimiento, mediante una técnica de minería de datos, se obtiene un modelo de conocimiento, que representa patrones de comportamiento observados en los valores de las variables del problema o relaciones de asociación entre dichas variables. También pueden usarse varias técnicas a la vez para generar distintos modelos, aunque generalmente cada técnica obliga a un preprocesado diferente de los datos.
6- Interpretación y evaluación de datos, una vez obtenido el modelo, se debe proceder a su validación comprobando que las conclusiones que arroja son válidas y suficientemente satisfactorias. En el caso de haber obtenido varios modelos mediante el uso de distintas técnicas, se deben comparar los modelos en busca de aquel que se ajuste mejor al problema.

Si el modelo final no superara esta evaluación el proceso se podría repetir desde el principio o, si el experto lo considera oportuno, a partir de cualquiera de los pasos anteriores. Esta retroalimentación se podrá repetir cuantas veces se considere necesario hasta obtener un modelo válido. Una vez validado el modelo, si resulta ser aceptable (proporciona salidas adecuadas y/o con márgenes de error admisibles) éste ya está listo para su explotación. Los modelos obtenidos por técnicas de minería de datos se aplican incorporándolos en los sistemas de análisis de información de las organizaciones.

1.3.1. Técnicas de minería de datos

Como ya se ha comentado, las técnicas de la minería de datos provienen de la inteligencia artificial y de la estadística, dichas técnicas, no son más que algoritmos, más o menos sofisticados que se aplican sobre un conjunto de datos para obtener unos resultados.

Las técnicas más representativas son:

Regresión Lineal**:** En estadística, LR es un enfoque lineal para modelizar la relación entre una respuesta de escala y una o más variables explicativas. El caso de una variable explicativa se denomina LR, mientras que para más de una variable explicativa, se denomina LR multivariable (Zhang et al. 2021).

Series Temporales: Las series temporales se utilizan para analizar la relación causal entre variables que evolucionan con el tiempo y se influyen mutuamente. Son comunes en análisis financiero, pronóstico del clima, seguimiento de datos de salud y cualquier situación en la que se deba comprender cómo cambian las variables a lo largo del tiempo

Reglas de Asociación**:** Las reglas de asociación se emplean para descubrir patrones o hechos que ocurren en conjunto dentro de un conjunto de datos. Son especialmente útiles en análisis de mercado, recomendación de productos y otras áreas donde se buscan relaciones entre elementos.

Árboles de Decisión**:** Los árboles de decisión son modelos de predicción utilizados en inteligencia artificial y análisis predictivo. Se crean a partir de una base de datos y representan decisiones lógicas basadas en condiciones sucesivas. Son similares a diagramas de flujo que ayudan a categorizar y resolver problemas mediante la evaluación de condiciones en secuencia.

Redes Neuronales**:** Las redes neuronales son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales. Estas redes consisten en una interconexión de unidades llamadas neuronas, que trabajan juntas para generar una respuesta de

salida. Se utilizan en tareas como reconocimiento de patrones, procesamiento de imágenes, procesamiento de lenguaje natural y más.

1.3.2. Modelos de predicción

El análisis predictivo es una rama de la minería de datos que se enfoca en descubrir información que permita anticipar tendencias y comportamientos futuros. A menudo, nos interesa predecir situaciones desconocidas que ocurrirán en el futuro, pero el análisis predictivo también se puede aplicar al pasado y al presente. Por ejemplo, puede ayudarnos a identificar posibles sospechosos después de haber ocurrido un crimen o una transacción fraudulenta con tarjeta de crédito.

La esencia del análisis predictivo radica en identificar conexiones entre las variables que explican y las variables que predicen, utilizando datos del pasado como base para proyectar lo que sucederá en el futuro. Sin embargo, es fundamental tener en cuenta que la confiabilidad y utilidad de las predicciones dependerán en gran medida de la calidad de los datos y de las suposiciones utilizadas en el proceso de análisis.

En el ámbito de la investigación, es fundamental entender que existen tres aspectos clave en el modelado predictivo:

La muestra de datos: Esta parte se refiere a los datos que se recopilan para representar de manera precisa el problema que se quiere resolver. Estos datos deben mostrar relaciones conocidas entre las entradas y las salidas.

El aprendizaje del modelo: Aquí, se crea un algoritmo que se ajusta a los datos recopilados. Es importante destacar que este modelo debe ser capaz de ser utilizado en el futuro repetidamente para hacer predicciones.

Las predicciones: Este es el uso del modelo previamente entrenado para realizar predicciones sobre nuevos datos, donde el resultado no es conocido de antemano.

Sin embargo, aunque el análisis predictivo es una herramienta poderosa, es importante ser consciente de sus posibles desventajas en un contexto de investigación:

- Errores en la fase de entrenamiento y prueba pueden magnificarse en las predicciones futuras.
- Los datos iniciales utilizados para entrenar el modelo pueden no ser representativos de toda la población estudiada, lo que podría llevar a resultados sesgados.
- El modelo puede no ser capaz de detectar diferentes tipos de datos que difieren del conjunto de entrenamiento original.

En términos de tipos de modelos predictivos, existen dos categorías principales:

Modelos de clasificación: Estos modelos se utilizan para predecir la pertenencia a una categoría o clase, como determinar qué clientes son propensos a abandonar un servicio. Los resultados de estos modelos son binarios, a menudo en forma de 0 y 1, junto con un grado de probabilidad.

Modelos de regresión: Estos modelos permiten predecir un valor numérico, como el beneficio que generará un cliente en los próximos meses. En lugar de clasificar en categorías, estos modelos predicen valores continuos.

1.3.3. Algoritmos de asociación

En la era de la información, la capacidad de extraer conocimiento valioso de grandes conjuntos de datos se ha convertido en un activo estratégico para empresas y organizaciones de todo tipo. Dentro de la minería de datos, los algoritmos de asociación desempeñan un papel fundamental al revelar conexiones valiosas entre elementos en conjuntos de datos.

Este tipo de técnicas se emplea para establecer las posibles relaciones o correlaciones entre distintas acciones o sucesos aparentemente independientes; pudiendo reconocer como la ocurrencia de un suceso o acción puede inducir o generar la aparición de otros. (López, Herrero 2006)

Estos algoritmos excavan profundamente en datos transaccionales, como historiales de compras, registros de navegación en línea y datos de transacciones financieras, para revelar reglas de asociación y patrones frecuentes.

Estos algoritmos son particularmente útiles en aplicaciones de análisis de mercado, recomendación de productos y detección de comportamientos de clientes. Algunos ejemplos de algoritmos de asociación son los siguientes:

Apriori**:** El algoritmo Apriori es uno de los algoritmos de asociación más conocidos. Se utiliza para descubrir reglas de asociación, que son patrones que indican la coocurrencia de elementos en transacciones. Por ejemplo, en un conjunto de datos de compras, Apriori podría revelar que "Si un cliente compra pan, es probable que también compre mantequilla". Estas reglas de asociación se utilizan comúnmente en sistemas de recomendación y en la colocación de productos en tiendas.

FP-Growth (Crecimiento frecuente de patrones)**:** FP-Growth es otro algoritmo de asociación que se utiliza para encontrar patrones frecuentes en conjuntos de datos. Este algoritmo es eficiente y suele ser más rápido que Apriori en grandes conjuntos de datos. FP-Growth crea una estructura de árbol compacta para representar los patrones de manera eficiente.

Eclat (Transformación de clase de equivalencia**)**: Eclat es otro algoritmo de asociación que se utiliza para encontrar conjuntos de elementos frecuentes. A diferencia de Apriori y FP-Growth, Eclat no utiliza estructuras de árbol, lo que lo hace especialmente útil para conjuntos de datos densos y grandes.

1.3.4. Prophet

Prophet es un método para predecir datos de series temporales basado en un modelo aditivo en el cual las tendencias no lineales se ajustan con una estacionalidad anual, semanal y/o diaria, además de considerar la influencia de los días festivos. Funciona mejor con series temporales que poseen una fuerte componente estacional y varias temporadas de datos históricos. (Cobo Cano 2021)

Cuando se tienen series de tiempo, es elemental invertir tiempo en analizar la variable que se desea pronosticar. Se deben entrar a considerar: la estacionariedad, la estacionalidad, las distribuciones y las relaciones de características externas para el diseño de la arquitectura de cualquier modelo. (Lopera Arango 2022)

Según (Galmés Mifsud 2019), Prophet es un algoritmo de código abierto desarrollado en febrero de 2017 por Facebook. Pertenece a la familia de algoritmos GAM (modelos aditivos generalizados). Se propone un modelo de regresión modular combinado con parámetros interpretables que pueden ser ajustables intuitivamente por un analista con un conocimiento avanzado en series temporales.

En este algoritmo se usa un modelo que se pueden descomponer en tres componentes principales, las cuales se estiman por separado.

$y(t) = g(t) + s(t) + h(t) + e(t)$

- g(t): tendencia
- s(t): componente estacional
- h(t): las vacaciones (holidays)
- El término e (épsilon) se refiere al término del error. Se hace el supuesto paramétrico de que este término se distribuye de forma Normal.

Por otro lado, la selección del parámetro "period" se utiliza para definir la forma y duración de la estacionalidad que se desea capturar en los datos. Ajustar estos valores depende de la naturaleza específica de los datos y la complejidad de la estacionalidad que se intenta modelar. En última instancia, la adaptación de estos parámetros se realiza con el objetivo de lograr un modelo que se ajuste de manera óptima a la dinámica temporal de los datos.

El parámetro **seasonality_mode** en Prophet se utiliza para especificar si los componentes estacionales deben considerarse de manera "aditiva" o "multiplicativa". La elección entre estos dos modos depende de cómo las estacionalidades interactúan con la serie temporal.

- **Aditivo (seasonality_mode='additive'):** En este modo, se asume que la estacionalidad tiene un efecto constante y se suma directamente a la tendencia. Este modo es apropiado cuando el impacto de la estacionalidad no depende del nivel actual de la serie temporal.
- **Multiplicativo (seasonality_mode='multiplicative'):** En este modo, se asume que la estacionalidad tiene un efecto proporcional al nivel de la serie temporal. Es decir, la estacionalidad se multiplica por la tendencia en lugar de sumarse. Este modo es más apropiado cuando el impacto de la estacionalidad es proporcional al nivel de la serie temporal.

1.3.5. Apriori

Los algoritmos de reglas de asociación tienen como objetivo encontrar relaciones dentro un conjunto de transacciones, en concreto, *items* o atributos que tienden a ocurrir de forma conjunta.

Uno de los métodos de minería de datos de descubrimiento de reglas de asociación más conocidos y utilizados es el algoritmo Apriori. La regla de asociación y el algoritmo Apriori son dos algoritmos muy destacados para encontrar una serie de conjuntos de elementos que aparecen con frecuencia a partir de datos de transacciones almacenados en bases de datos. El cálculo se realiza para determinar el valor mínimo de soporte y confianza que producirá la regla de asociación. (Santoso 2021)

Su enfoque innovador para descubrir patrones de comportamiento en conjuntos de datos grandes ha contribuido significativamente al análisis de datos y la toma de decisiones en diversas disciplinas. La fortaleza principal del algoritmo radica en su capacidad para identificar conjuntos de ítems frecuentes, aquellos elementos que aparecen juntos con una alta frecuencia en una base de datos.

Este enfoque apriori permite descubrir asociaciones entre elementos que pueden no ser obvias a simple vista, revelando relaciones ocultas y proporcionando información valiosa sobre patrones de comportamiento. Además, el algoritmo Apriori ha demostrado ser altamente escalable, lo que lo hace adecuado para conjuntos de datos masivos.

Aunque puede haber desafíos en términos de uso eficiente de la memoria y optimización, el algoritmo Apriori sigue siendo una herramienta esencial en la caja de herramientas de los profesionales de la minería de datos, contribuyendo significativamente a la comprensión y extracción de conocimiento en grandes conjuntos de datos.

1.4. Tendencias Tecnológicas

Es necesario para el desarrollo de un producto informático que satisfaga una necesidad existente, utilizar las herramientas y tecnologías adecuadas. Esto es esencial porque la elección de estas herramientas y tecnologías depende en gran medida del tipo de problema que estás tratando de solucionar. Por lo tanto, elegir las herramientas adecuadas es fundamental para el éxito de tu proyecto, ya que te permitirá abordar eficazmente la necesidad existente que estás tratando de resolver con tu producto informático.

1.4.1. Lenguaje de Programación

Lenguaje Python:

Python es un lenguaje de programación que fue creado a finales de la década de 1980 por Guido van Rossum. Se ha convertido en uno de los lenguajes de programación más populares y ampliamente utilizados en todo el mundo. Python es un lenguaje de programación de código abierto, lo que significa que es gratuito, libre y de código abierto, lo que permite a cualquiera utilizarlo y contribuir a su desarrollo. Es una de las principales opciones para la minería de datos y el análisis de datos debido a su amplia gama de bibliotecas y herramientas especializadas.

La librería Pandas es un conjunto de herramientas para el análisis y manipulación de datos rápida, flexible y muy potente, desarrollado sobre el lenguaje Python. Pandas permite la preparación, manipulación, limpieza, normalización y transformación de los datos para su análisis. Además, permite combinar conjuntos de datos. También, proporciona métodos para eliminar o rellenar valores faltantes. Igualmente, permite realizar agrupaciones a partir de

uno de los ejes del conjunto de datos, entre otras funcionalidades avanzadas. (Rivas, Castillo 2022)

Algunas de las características clave de Python incluyen:

- Sintaxis legible y elegante**:** Python es conocido por su sintaxis limpia y legible, que facilita la escritura y comprensión del código. Esto lo convierte en un lenguaje de programación ideal tanto para principiantes como para programadores experimentados.
- Amplia comunidad y ecosistema de bibliotecas**:** Python cuenta con una gran comunidad de usuarios y una amplia variedad de bibliotecas y módulos disponibles. Estas bibliotecas abarcan una amplia gama de aplicaciones, desde desarrollo web y científico hasta aprendizaje automático y automatización.
- Orientado a objetos**:** Python es un lenguaje de programación orientado a objetos que permite a los programadores crear objetos y clases para organizar y estructurar su código de manera eficiente.
- Amplias aplicaciones en ciencia de datos y aprendizaje automático: Python se ha convertido en la elección preferida en campos como la ciencia de datos y el aprendizaje automático debido a bibliotecas populares como NumPy, Pandas, Matplotlib, Scikit-Learn y TensorFlow.
- Integración con bases de datos y fuentes de datos**:** Python se integra bien con una amplia variedad de sistemas de bases de datos y fuentes de datos, lo que facilita la extracción y transformación de datos desde diferentes fuentes.
- Desarrollo web y aplicaciones**:** Python es ampliamente utilizado en el desarrollo web y la creación de aplicaciones web gracias a frameworks como Django y Flask, que simplifican el proceso de desarrollo.

Lenguaje R:

R fue creado en 1992 en Nueva Zelanda por Ross Ihaka y Robert Gentleman (Ihaka [1998]). La intención inicial con R, era hacer un lenguaje didáctico, para ser utilizado en el curso de Introducción a la Estadística de la Universidad de

Nueva Zelanda. Para ello decidieron adoptar la sintaxis del lenguaje S desarrollado por Bell Laboratories. (Santana Sepúlveda, Mateos Farfán 2014)

Es un lenguaje con licencia GNU, es decir, es libre, gratuito y abierto. En resumen, lo puede usar cualquiera y no es propiedad de nadie. R funciona con paquetes gratuitos, como las librerías en otros lenguajes, y puedes descargar y usar esos paquetes.

Este lenguaje posee diversas características sobresalientes. En primer lugar, brinda la capacidad de generar gráficos con base en LaTeX, lo que facilita la creación de representaciones visuales de datos. Además, ofrece una amplia gama de herramientas estadísticas, incluyendo modelos tanto lineales como no lineales, diversas pruebas estadísticas y algoritmos para la clasificación y agrupamiento de datos. Una característica destacada es su programación orientada a objetos (POO), lo que permite a los usuarios crear sus propias funciones y objetos de manera modular. También se integra sin problemas con diversas bases de datos, lo que facilita el manejo de información. Por último, es importante destacar que este lenguaje puede ser utilizado con fines matemáticos, lo que lo convierte en una herramienta versátil para abordar una amplia gama de tareas analíticas y computacionales.

Durante la investigación y tras analizar las diversas oportunidades que ofrecen las opciones estudiadas, se tomó la decisión de utilizar el lenguaje de programación Python. Esta elección se basa en una serie de ventajas clave que ofrece Python en el contexto de la minería de datos y el análisis. En primer lugar, Python cuenta con una licencia de código abierto, lo que significa que es un lenguaje libre, gratuito y completamente abierto para su uso por cualquier persona, sin restricciones de propiedad. Además, Python dispone de una rica colección de bibliotecas y paquetes gratuitos que amplían su funcionalidad en diversas áreas, lo que incluye bibliotecas como NumPy, Pandas, Matplotlib, Scikit-Learn y TensorFlow para la manipulación de datos, visualización y aprendizaje automático.

Python también se beneficia de una comunidad informática vibrante y comprometida que contribuye constantemente a su desarrollo y mejora. Los

usuarios pueden publicar paquetes que extienden su funcionalidad, lo que multiplica sus capacidades y lo hace altamente adaptable a las necesidades específicas de minería de datos. Además, Python ofrece la capacidad de crear gráficos de manera eficiente mediante bibliotecas como Matplotlib, Seaborn y Plotly, lo que facilita la visualización de datos y la comunicación de hallazgos. En términos de análisis estadístico, Python es altamente versátil, con la biblioteca Scikit-Learn que proporciona una amplia gama de algoritmos de aprendizaje automático y bibliotecas como StatsModels que admiten análisis estadísticos avanzados. La integración de Python con distintas bases de datos, incluyendo MySQL, simplifica la extracción y manipulación de datos desde diversas fuentes.

1.4.2. Entorno de Desarrollo Integrado (IDE)

Jupyter Notebook

Jupyter Notebook es un entorno interactivo y de código abierto ampliamente utilizado por científicos de datos, ingenieros y profesionales de diversas disciplinas. Esta elección se basa en una serie de ventajas clave que ofrece en el contexto de la programación, la exploración de datos y la presentación de resultados.

Jupyter Notebook admite varios lenguajes de programación, como Python, R, Julia y otros, lo que lo convierte en una herramienta versátil para el análisis de datos y la programación científica. Los usuarios pueden crear documentos enriquecidos que combinan código ejecutable, visualizaciones, texto narrativo y ecuaciones matemáticas, lo que facilita la comunicación efectiva de resultados y hallazgos.

La capacidad de ejecutar y modificar código en tiempo real dentro de Jupyter Notebook permite una experimentación ágil y la exploración interactiva de datos. Esto es especialmente valioso para investigaciones científicas, análisis de datos y desarrollo de modelos de aprendizaje automático.

Jupyter Notebook es altamente extensible, lo que permite a los usuarios crear extensiones personalizadas y widgets para satisfacer sus necesidades

específicas. Además, ofrece un ecosistema de complementos que amplían su funcionalidad.

La facilidad para crear visualizaciones y gráficos, gracias a bibliotecas como Matplotlib y Seaborn, facilita la presentación visual de datos y resultados. También permite la exportación de documentos en varios formatos, incluyendo PDF, HTML y más.

Visual Studio Code:

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft que se ha convertido en una herramienta esencial para desarrolladores de software en todo el mundo. Esta elección se basa en una serie de ventajas clave que ofrece en el contexto de la programación y desarrollo de aplicaciones.

VS Code es un entorno de desarrollo altamente versátil que admite una amplia gama de lenguajes de programación, incluyendo pero no limitado a JavaScript, Python, C++, Java, Ruby y muchos más. Esto lo convierte en una opción atractiva para desarrolladores que trabajan en diversas plataformas y tecnologías.

Una característica destacada de VS Code es su extensibilidad. Los usuarios pueden personalizar el editor mediante la instalación de extensiones que agregan funcionalidades específicas, como depuración, integración con sistemas de control de versiones, resaltado de sintaxis y mucho más. Esto permite que cada desarrollador adapte el entorno a sus necesidades individuales.

La interfaz de usuario intuitiva y la integración con herramientas de desarrollo populares, como Git, hacen que VS Code sea fácil de usar y eficiente para tareas de desarrollo cotidianas. Además, ofrece un conjunto robusto de características para depuración, autocompletado de código y navegación en el código fuente.

Visual Studio Code es de código abierto y está disponible de forma gratuita, lo que lo hace accesible para desarrolladores de todos los niveles, desde principiantes hasta profesionales experimentados. Su comunidad activa y en

constante crecimiento contribuye a su desarrollo continuo y a la creación de nuevas extensiones.

1.4.3. Sistema gestor de Base de Datos

Un sistema de gestión de base de datos (SGBD) es un conjunto de software que permite administrar, organizar y manipular de manera eficiente una base de datos. Proporciona una interfaz para crear, modificar y consultar la base de datos, garantizando la integridad, seguridad y disponibilidad de los datos almacenados. Además, ofrece herramientas para realizar copias de seguridad, recuperación ante fallos y optimización del rendimiento de las consultas (Connolly y Begg, 2014). Entre los gestores de base de datos se encuentran PostgreSQL, SQLite y MySQL.

PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos relacional de código abierto y robusto. Proporciona un entorno de base de datos completo con soporte para consultas SQL avanzadas, transacciones, integridad de datos, replicación y escalabilidad. PostgreSQL se destaca por su estabilidad, rendimiento y capacidad para manejar grandes volúmenes de datos. Es ampliamente utilizado en diversas aplicaciones y entornos, desde pequeñas empresas hasta grandes organizaciones. Ofrece un completo soporte SQL, transacciones ACID y una gran escalabilidad y rendimiento. Además, se destaca por su capacidad para garantizar la integridad referencial y aplicar restricciones, así como por su flexibilidad en la definición de funciones almacenadas y procedimientos. Con un amplio conjunto de tipos de datos avanzados, PostgreSQL se adapta a diversas necesidades. Su enfoque en la seguridad y el control de acceso, junto con su reputación como una de las bases de datos más avanzadas, lo convierten en una opción sólida para proyectos de todos los tamaños y complejidades (Group, 2023).

MySQL

Se ha convertido en el SGBD de código abierto más MySQL es un sistema de gestión de bases de datos relacional de código abierto. Es ampliamente utilizado en el desarrollo de aplicaciones web y es conocido por su rendimiento,

confiabilidad y facilidad de uso. Destaca por su escalabilidad, rendimiento y soporte multiplataforma. Con amplio respaldo para diferentes lenguajes de programación y características de seguridad avanzadas, MySQL se adapta a diversos entornos de desarrollo y garantiza la integridad de los datos. Además, ofrece funcionalidad extensible a través de complementos y una eficiente gestión de transacciones con soporte para transacciones ACID. Con estas características, MySQL se posiciona como una opción confiable y flexible para aplicaciones de alto rendimiento y requerimientos específicos. (Corporation, 2023).

SQLite

SQLite es un sistema de gestión de bases de datos relacional de código abierto que se caracteriza por ser ligero, autónomo y de uso sencillo. A diferencia de otros sistemas de gestión de bases de datos, SQLite se implementa como una biblioteca embebida en la aplicación, lo que facilita su integración en una amplia variedad de proyectos y plataformas. SQLite es compatible con la mayoría de las características estándar de SQL y es utilizado en una amplia gama de aplicaciones, desde dispositivos móviles hasta aplicaciones de escritorio. Ofrece soporte para transacciones ACID, lo que garantiza la integridad de los datos, y permite la ejecución eficiente de consultas y operaciones en entornos con grandes volúmenes de datos. (Consortium, 2023).

Elección del sistema gestor de base de datos

Al analizar las características de los diferentes gestores de bases de datos anteriormente mencionados se determina lo siguiente:

PostgreSQL posee un elevado consumo de memoria RAM, este aspecto en ocasiones reduce los tiempos de respuesta del sistema y en computadoras de prestaciones limitadas, además, para realizar su despliegue de forma local se requieren de conocimientos avanzados, por lo cual no es una opción viable, Por otro lado SQLite tiene limitaciones en cuanto al tamaño de datos, variables y su rendimiento puede verse afectado negativamente a medida que la base de datos crece en tamaño y complejidad, además, carece de funciones de seguridad y administración de usuarios, con lo cual se desecha como alternativa viable. Por último, MySQL es fácil de usar, es ligero, es potente y viene integrado en los

servidores libres. Posee mayor rendimiento, buenas utilidades de administración y está preparado para manejar grandes volúmenes de datos, además, es altamente personalizable por lo que su despliegue es fácil de realizarse en entornos locales. Por estos motivos se decide seleccionar a MySQL como el gestor de base de datos de la presente investigación.

1.5. Conclusiones del Capítulo

Después de haber realizado el estudio de la teoría y los conceptos en torno a la minería de datos, los modelos predictivos y de asociación, así como analizar los principales algoritmos de cada tipo, podemos concluir que:

1. La minería de datos es un campo interdisciplinario que se enfoca en descubrir patrones, relaciones y conocimientos ocultos en grandes conjuntos de datos. Utiliza una variedad de técnicas, algoritmos y herramientas para analizar datos con el objetivo de extraer información valiosa y tomar decisiones fundamentadas. La minería de datos se aplica en diversas industrias, desde el comercio electrónico y la atención médica hasta la gestión de inventarios y la seguridad cibernética.
2. Los modelos predictivos son técnicas utilizadas en la minería de datos y el aprendizaje automático para hacer predicciones sobre eventos futuros o valores desconocidos en función de datos históricos. Hay dos tipos principales de algoritmos predictivos: regresión y clasificación.
3. Los algoritmos de asociación son una categoría de técnicas en la minería de datos que se utilizan para descubrir patrones de coocurrencia en conjuntos de datos transaccionales. Estos algoritmos son esenciales para identificar relaciones interesantes entre elementos en datos, como compras de clientes o registros de navegación en línea.
4. Se definieron las tecnologías que mejor se ajustan a los requerimientos del problema detectado, comprobándose que el lenguaje Python y el entorno de desarrollo Jupyter Notebook para entrenar los algoritmos y validar los resultados, y Visual Studio Code como entorno de desarrollo, son las herramientas adecuadas para darle solución al mismo.

Capítulo II. Propuesta de Solución

2.1. Introducción del Capítulo

La elaboración de modelos predictivos y de asociación en el ámbito de la minería de datos implica llevar a cabo un análisis y diseño minucioso. El objetivo es desarrollar una propuesta de solución altamente efectiva para poder modelar y prever con precisión. Este proceso requiere un enfoque detallado y cuidadoso para asegurar la calidad y eficacia de los modelos resultantes.

2.2. Carga de datos en Python

Los datasets para el entrenamiento de los algoritmos que se utilizan en esta investigación provienen de la Universidad de Valencia, España. Estos datos nos ofrecen el control de las ventas de un bar de dicho país desde septiembre de 2020 hasta octubre de 2023. Para ello tenemos varios dataset (“cuentas”, “pedidos”, “pedidos por mesa”, “categoría” y “productos”) pero los que se utilizaron para el entrenamiento de los algoritmos de predicción y asociación fueron “cuentas” y “pedidos por mesa”, estos datasets están en formato csv.

Llevar a cabo una revisión exhaustiva de los datos fue esencial. Esta revisión tenía el propósito de asegurarnos de que los datos fueran precisos y confiables, evitando así cualquier influencia negativa en el rendimiento de los algoritmos. La calidad y consistencia de los datos son fundamentales para que los algoritmos de predicción y asociación generen resultados sólidos y efectivos.

Para cargar los datos se realizó mediante el uso de la librería Pandas de Python. Para ello se importó la librería Pandas, la cual posibilita la carga de datos desde diversas fuentes y formatos, Además, facilita la manipulación de datos al proporcionar herramientas para filtrar, transformar y limpiar datos de manera efectiva.

Algoritmo de predicción de ventas:

```
import pandas as pd

data = pd.read_csv("cuentas.csv")

data.head()
```

	created_at	updated_at	id	deleted_at	is_active	serial	price	discount	status	type	table_id	credit_card	is_deleted
0	2020-09-21 17:53:43.643762	2020-09-21 17:59:08.100712	86	NaN	1	D0000086	19.0	0.0	terminada	domicile	NaN	0	0
1	2020-09-21 18:25:02.725213	2020-09-21 19:21:09.618787	92	NaN	1	D0000092	7.5	0.0	terminada	domicile	NaN	0	0
2	2020-09-23 17:49:07.440856	2020-09-23 17:51:57.620671	98	NaN	1	D0000098	24.5	0.0	terminada	domicile	NaN	0	0
3	2020-09-23 18:45:01.094154	2020-09-23 20:02:45.507072	101	NaN	1	L0000101	39.0	0.0	terminada	local	1.0	0	0
4	2020-09-24 19:08:53.615412	2020-09-24 20:00:37.393200	102	NaN	1	D0000102	34.0	0.0	terminada	domicile	NaN	0	0

Figura 1. DataFrame “cuentas”. Fuente: Elaboración propia

En el código previo, se procedió a cargar el archivo CSV "cuentas", almacenándolo en la variable llamada "data". Para evaluar la estructura de esta variable, se utilizó la función por defecto data.head(). Esta función exhibe las primeras cinco filas del marco de datos, aunque es posible ajustar la cantidad de filas mostradas según sea necesario.

Algoritmo de predicción de ventas por producto y A priori:

```
import pandas as pd

data = pd.read_csv("pedidos por mesa.csv")

data.head()
```

	created_at	updated_at	id	deleted_at	is_active	qty	product_name	details	importe	qty_payed	total_price	command_id	product_id	is_deleted
0	2020-09-23 19:33:00.259877	2020-09-23 19:33:00.260791	665	NaN	1	1	Alemana	NaN	7.5	0	7.5	101	215	0
1	2020-09-29 20:24:23.521139	2020-09-29 20:24:23.521695	741	NaN	1	1	Heineken	NaN	3.0	0	3.0	128	159	0
2	2020-10-02 18:54:34.443012	2020-10-02 18:54:34.443681	901	NaN	1	2	Cerveza botella	NaN	6.0	0	6.0	150	16	0
3	2020-10-04 18:07:15.201530	2020-10-04 18:07:15.204457	902	NaN	1	1	Pan Ajo + mozzarella	NaN	4.5	0	4.5	152	48	0
4	2020-10-04 18:25:58.224908	2020-10-04 18:25:58.225457	906	NaN	1	1	Mozzarella empanada	NaN	2.5	0	2.5	153	44	0

Figura 2. DataFrame “pedidos por mesa”. Fuente: Elaboración propia

En el código anterior, se importó y almacenó el archivo CSV "pedidos por mesa" en la variable denominada "data". Luego, se inspeccionó la estructura de esta variable utilizando la función data.head(), siguiendo el mismo enfoque empleado anteriormente.

Luego de cargar correctamente los conjuntos de datos, se avanzó hacia un análisis exploratorio para obtener una comprensión más profunda de su estructura.

2.3. Análisis exploratorio de los datos

El análisis exploratorio de datos es un instrumento indispensable para realizar las primeras aproximaciones al estudio de la estructura de la información en una determinada área y detectar posibles anomalías presentes en las observaciones. (Martín Calvo 2016)

Desempeña un papel crucial en el análisis de datos permitiendo comprender a fondo los datos al revelar la naturaleza de las variables, sus distribuciones y las relaciones entre ellas, lo que es esencial para formular hipótesis y tomar decisiones informadas. Además, durante este proceso se pueden detectar problemas como valores atípicos, datos faltantes o errores en los datos, lo que facilita la corrección antes de avanzar en el modelado. También contribuye a la selección de características al identificar cuáles son las más relevantes para el problema, lo que simplifica los modelos y mejora su eficiencia. La visualización desempeña un papel crucial ya que ayuda a representar la información de manera efectiva y revela tendencias y patrones que pueden no ser evidentes en los datos en bruto.

Dado que se utilizó dos conjuntos de datos para desarrollar los algoritmos, llevamos a cabo un análisis exploratorio individual de cada uno de ellos.

Analizando el DataFrame “cuentas”:

Al utilizar la función info(), la cual es un método proporcionado por la biblioteca Pandas. Este método se emplea para obtener un resumen conciso de un DataFrame. Al invocar info() en un DataFrame, se presenta un resumen que abarca detalles como la cantidad de filas, la cantidad de columnas, los nombres de las columnas, los tipos de datos de las columnas, la cantidad de valores no nulos y el uso de memoria.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27607 entries, 0 to 27606
Data columns (total 13 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   created_at   27607 non-null  object 
 1   updated_at   27607 non-null  object 
 2   id           27607 non-null  int64  
 3   deleted_at   0 non-null      float64
 4   is_active    27607 non-null  int64  
 5   serial       27607 non-null  object 
 6   price        27607 non-null  float64
 7   discount     27293 non-null  float64
 8   status       27607 non-null  object 
 9   type         27607 non-null  object 
 10  table_id     18159 non-null  float64
 11  credit_card  27607 non-null  int64  
 12  is_deleted   27607 non-null  int64  
dtypes: float64(4), int64(4), object(5)
memory usage: 2.7+ MB
```

Figura 3. data.info() al DataFrame “cuentas”. Fuente: Elaboración propia

Después de confirmar la ausencia de valores nulos y obtener la información de que consta de 27607 filas y 13 columnas, identificamos las columnas "created_at" (fecha), "type" (tipo de pedidos) y "price" (precio) como las más relevantes y necesarias para aplicar los algoritmos de minería de datos en etapas posteriores.

Para obtener una comprensión más profunda del comportamiento de estas columnas, nos apoyamos en la librería Matplotlib. Qué posibilita la creación de diversos gráficos y visualizaciones, como gráficos de líneas, de barras, de dispersión, de torta, histogramas, entre otros. Estos gráficos resultan fundamentales para identificar patrones, tendencias y relaciones en los datos, contribuyendo así a un análisis más detallado y significativo.

Iniciamos el análisis examinando el importe total según el tipo de pedidos. Para ello, optamos por utilizar un gráfico de barras, ya que proporciona una comparación visual clara entre las diversas categorías, siendo fácil de interpretar. Este tipo de gráfico es efectivo para representar datos categóricos. Al crear dicho gráfico, pudimos observar que los pedidos de tipo local lideran en ventas, seguidos por los pedidos a domicilio y, posteriormente, los pedidos a recoger.

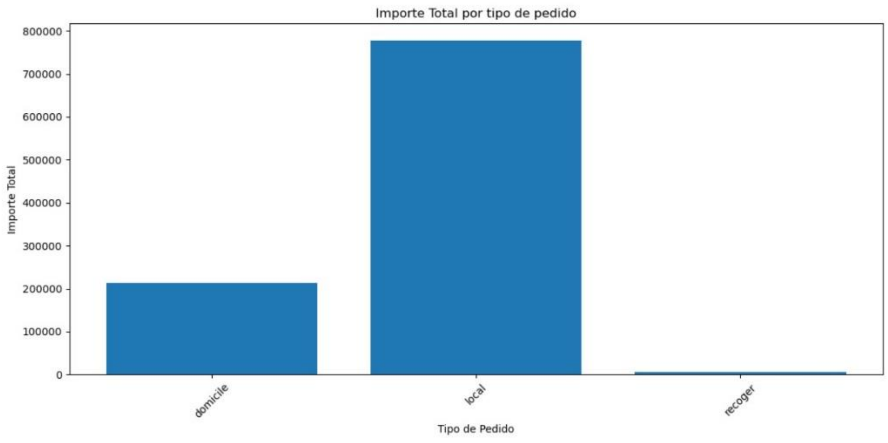

Figura 4. Importe Total por tipo de pedidos. Fuente: Elaboración propia

Para generar el gráfico mencionado, primero importamos la librería Matplotlib. Luego, llevamos a cabo una agrupación según el tipo de pedido y sumamos la columna "price" para obtener los importes totales por categoría. Una vez

obtenidos estos datos, configuramos el tamaño del gráfico, elegimos el tipo de gráfico con plt.bar (gráfico de barras) y seleccionamos los datos a visualizar. Posteriormente, añadimos etiquetas para el eje “X” e “Y”, el título del gráfico y ajustamos la rotación de los nombres en el eje “X” antes de mostrar el resultado.

```
import matplotlib.pyplot as plt

data1 = data.groupby(['type'])['price'].sum().reset_index()

plt.figure(figsize=(12, 6))

plt.bar(data1['type'], data1['price'])

plt.xlabel('Tipo de Pedido')

plt.ylabel('Importe Total')

plt.title('Importe Total por tipo de pedido')

plt.xticks(rotation=45)

plt.tight_layout()

plt.show()
```

Posteriormente, examiné el comportamiento de las ventas mediante la utilización de dos tipos de gráficos, los cuales son particularmente eficaces al visualizar la relación entre dos variables, como las fechas y las ventas. Estos gráficos resultan útiles para detectar patrones, tendencias y posibles correlaciones de manera inmediata. A través de este análisis, se evidenció que las ventas hasta abril de 2021 fueron menores en comparación con el periodo posterior a esa fecha, exhibiendo un leve incremento en este último periodo.

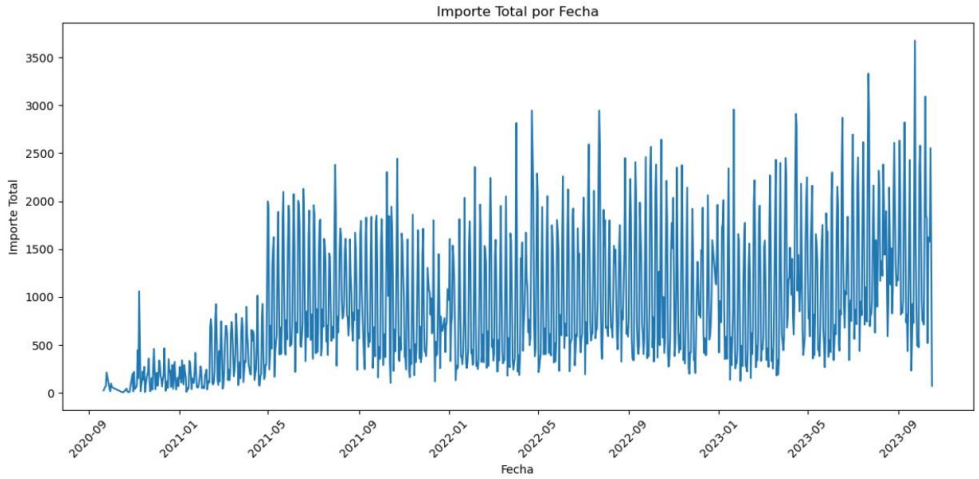

Figura 5. Importe diario gráfico de líneas (histórico). Fuente: Elaboración propia

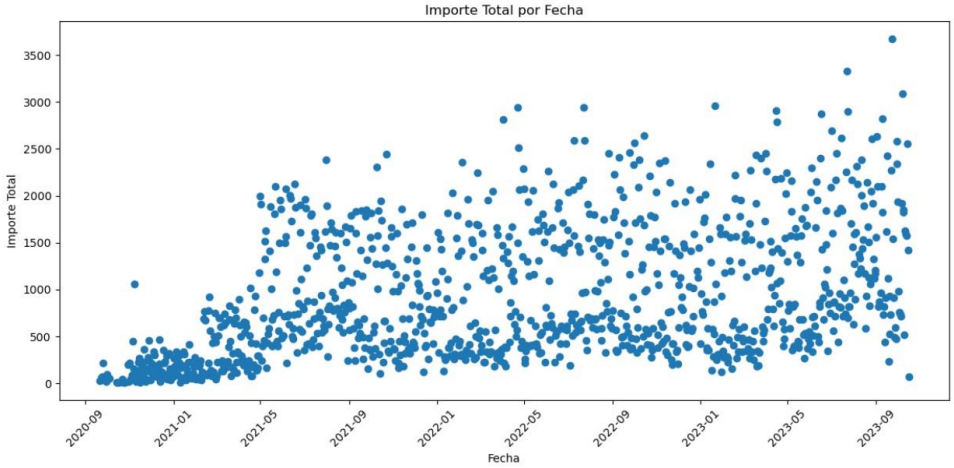

Figura 6. Importe diario gráfico de dispersión (histórico). Fuente: Elaboración propia

La creación de los gráficos mencionados sigue un proceso muy similar al del gráfico de barras. Sin embargo, en este caso, filtramos los datos utilizando la columna "created_at" y "price". Previamente, habíamos transformado la columna "created_at" al formato de solo fecha mediante dt.date, ya que inicialmente contenía información de fecha y hora, y nos interesaba mostrar solo los importes de ventas por días. Además, es esencial especificar el tipo de gráfico que queremos utilizar, ya sea plt.plot (gráfico de líneas) o plt.scatter (gráfico de dispersión), dependiendo de la representación visual deseada.

```
data['created_at'] = pd.to_datetime(data['created_at']).dt.date

data1 = data.groupby(['created_at'])['price'].sum().reset_index()

plt.figure(figsize=(12, 6))
```

plt.plot(data1['created_at'], data1['price'], linestyle='-') o plt.scatter

plt.xlabel("Fecha")

plt.ylabel('Importe Total')

plt.title('Importe Total por Fecha')

plt.xticks(rotation=45)

plt.tight_layout()

plt.show()

Analizando el DataFrame “pedidos por mesa”:

Al aplicar la función info(), se obtuvo el siguiente resumen del conjunto de datos:

```
data.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 100691 entries, 13710 to 59182
Data columns (total 14 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   created_at    100691 non-null  object
 1   updated_at    100691 non-null  object
 2   id            100691 non-null  int64
 3   deleted_at    0 non-null       float64
 4   is_active     100691 non-null  int64
 5   qty           100691 non-null  int64
 6   product_name  100691 non-null  object
 7   details       5526 non-null    object
 8   importe       100691 non-null  float64
 9   qty_payed     100691 non-null  int64
 10  total_price   100691 non-null  float64
 11  command_id    100691 non-null  int64
 12  product_id    100691 non-null  int64
 13  is_deleted    100691 non-null  int64
dtypes: float64(3), int64(7), object(4)
memory usage: 11.5+ MB
```

Figura 7. data.info() al DataFrame “pedidos por mesa”. Fuente: Elaboración propia

Ya que comprobamos que no existen valores nulos y que el conjunto de datos consta de 100691 filas y 14 columnas. Podemos identificar las columnas created_at(fecha), qty(cantidad), command_id(id de la cuenta) e importe, como las más importantes. Con el fin de obtener una comprensión más detallada de las ventas por productos, procederemos a crear un gráfico que permita visualizar aquellos productos que tienen mayores y menores niveles de venta.

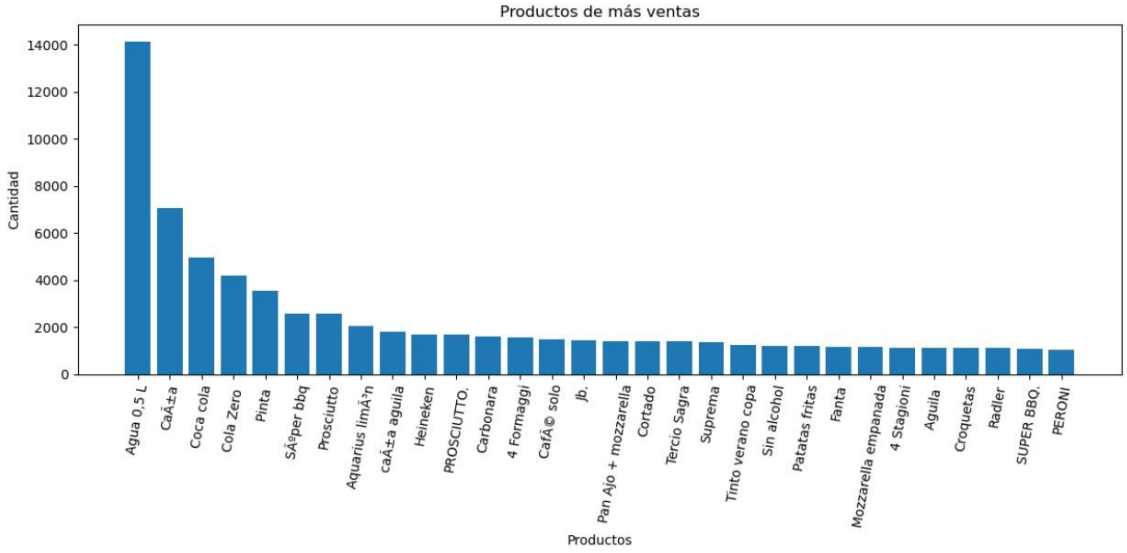

Figura 8. Productos de más ventas. Fuente: Elaboración propia

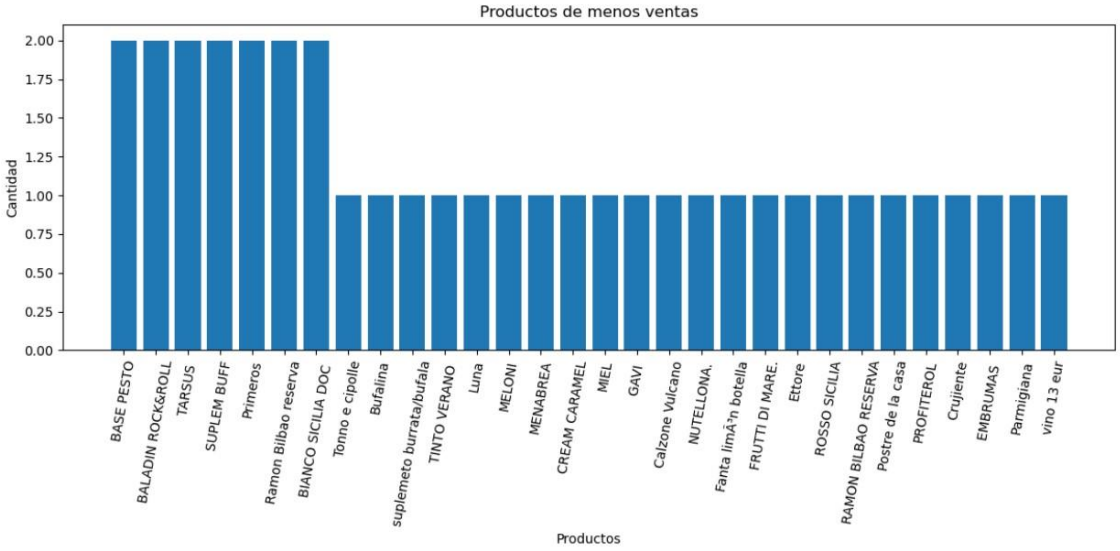

Figura 9. Productos de menos ventas. Fuente: Elaboración propia

Para generar los gráficos mencionados, en primer lugar, importamos la librería Matplotlib. Luego, realizamos una agrupación por "product_name" y sumamos la columna "qty", obteniendo así la cantidad de ventas por producto. Posteriormente, ordenamos los resultados de mayor a menor. Con estos datos ordenados, creamos el gráfico siguiendo el mismo proceso que se utilizó previamente. Para seleccionar los 30 productos más vendidos, empleamos la siguiente expresión: plt.bar(data2['product_name'][:30], data2['qty'][:30]). Similarmente, para mostrar los 30 productos menos vendidos: plt.bar(data2['product_name'][-30:], data2['qty'][-30:]). A continuación, se presenta el código utilizado para generar estos gráficos.

```
import matplotlib.pyplot as plt

data1 = data.groupby(["product_name"])['qty'].sum().reset_index()

data2 = data1.sort_values(by='qty', ascending=False)

plt.figure(figsize=(12, 6))

plt.bar(data2['product_name'][:30], data2['qty'][:30]) o (data2['product_name'][-30:], data2['qty'][-30:])

plt.xlabel('Productos')

plt.ylabel('Cantidad')

plt.title('Productos de más ventas')

plt.xticks(rotation=80)

plt.tight_layout()

plt.show()
```

2.4. Transformación de los datos

La transformación de datos representa un paso fundamental para poder aplicar los algoritmos de minería de datos. En esencia, implica la modificación o manipulación de los datos de entrada con el objetivo de hacerlos más apropiados para el análisis y la modelización. Esta fase desempeña un papel crucial al mejorar la calidad de los datos y facilitar la tarea de los algoritmos de minería de datos para identificar patrones, relaciones y tendencias de manera más efectiva.

Transformación de los datos para algoritmo de predicción de ventas:

En la implementación del algoritmo Prophet para la predicción de ventas, es esencial organizar los datos de ventas de manera que contengan dos columnas clave: 'ds', que representa la marca de tiempo, y 'y', que contiene los valores de ventas a predecir. Además, es crucial definir la frecuencia temporal de los datos, ya sea diaria, mensual u otra, para permitir que Prophet realice predicciones precisas al comprender la estructura temporal de la serie.

La identificación y gestión de valores atípicos también son aspectos críticos. La sensibilidad de Prophet a los valores atípicos puede afectar negativamente las

predicciones. Por tanto, se recomienda llevar a cabo un análisis de estos antes de aplicar el modelo, considerando técnicas para suavizar o corregir estos valores atípicos de manera efectiva.

Para aplicar el algoritmo Prophet, se realizaron transformaciones en los datos. Después de cargar los datos en la variable "data", se convirtió la columna "created_at" de formato datetime a formato date, es decir, solo fecha. Posteriormente, se llevó a cabo una agrupación por la columna "created_at" y se sumaron los valores de la columna "price". Este proceso resulta en un DataFrame con dos columnas: la primera con las fechas y la segunda con los importes.

A continuación, se ajustaron los nombres de las columnas, cambiando "created_at" por "ds" y "price" por "y", conforme a los requisitos de Prophet para entrenar el algoritmo. Posteriormente, se ordenó el DataFrame según la columna "ds". Finalmente, considerando la observación durante la exploración de datos que indicaba un aumento significativo en las ventas a partir de mayo de 2021, se filtraron los datos para incluir solo el periodo desde el 1 de mayo de 2021 hasta el 31 de mayo de 2023. Los datos posteriores a esa fecha se reservaron para la validación del algoritmo.

```
data['created_at'] = pd.to_datetime(data['created_at']).dt.date

data1 = data.groupby('created_at')['price'].sum().reset_index()

data1.columns = ['ds', 'y']

data1 = data1.sort_values(by='ds')

data2 = data1.iloc[195: 948]
```

Si aplicamos la función head() al DataFrame "data2", obtendremos el siguiente conjunto de datos:

	ds	y
195	2021-05-01	1998.47
196	2021-05-02	1914.76
197	2021-05-03	244.87
198	2021-05-04	394.14
199	2021-05-05	701.98

Figura 10. DataFrame “data2” (Algoritmo de predicción de ventas). Fuente: Elaboración propia

Transformación de los datos para algoritmo de predicción de ventas por productos:

Para realizar la predicción utilizando el algoritmo Prophet, se aplicaron transformaciones a los datos de manera similar al algoritmo anterior, aunque con algunas variaciones. En primer lugar, se almacenó en la variable "producto" el nombre del producto para el cual deseamos predecir las ventas.

Luego, se convirtió la columna "created_at" al tipo de dato "date" y se seleccionaron las filas desde la 2001 hasta la 78699, correspondientes al periodo del 1 de mayo de 2021 al 31 de mayo de 2023. A continuación, se filtraron las filas donde "product_name" es igual al nombre del producto que se quiere predecir.

Posteriormente, se procedió a agrupar por la columna "created_at" y se sumaron los valores de la columna "qty". Se cambiaron los nombres de las columnas, sustituyendo "created_at" por "ds" y "qty" por "y". Finalmente, se ordenó el DataFrame según la columna "ds". Estos pasos son esenciales para preparar los datos de manera adecuada antes de aplicar el algoritmo Prophet para la predicción.

```
producto = "Agua 0,5 L"

data['created_at'] = pd.to_datetime(data['created_at']).dt.date

data1 = data.iloc[2001: 81669]

data2 = data1[data1['product_name'] == producto]

data3 = data2.groupby('created_at')['qty'].sum().reset_index()
```

```
data3.columns = ['ds', 'y']

data4 = data3.sort_values(by='ds')
```

Si aplicamos la función head() al DataFrame "data4", obtendremos un conjunto de datos con dos columnas: "ds", que representa la fecha, y "y", que indica la cantidad de pedidos para el producto "Agua 0.5L".

	ds	y
0	2021-05-01	25
1	2021-05-02	15
2	2021-05-03	3
3	2021-05-04	6
4	2021-05-05	10

Figura 11. DataFrame “data4” (Algoritmo de predicción de ventas por productos). Fuente: Elaboración propia

Transformación de los datos para algoritmo de asociación (Apriori):

Para realizar este algoritmo, fue necesario realizar algunas modificaciones en la tabla de "pedidos por mesa". Inicialmente, se agruparon por "command_id" y "producto_name", sumando la columna "qty". Luego, mediante el método unstack(), se transformaron los niveles de índice internos en columnas, generando una tabla más amplia. Posteriormente, se restablecieron los índices del DataFrame y se llenaron los valores nulos con cero.

A continuación, se creó la función "hot_encode", la cual convierte los valores mayores a 1 en 1 y los valores menores a 1 en 0. Ya tiene una estructura adecuada para poder aplicar este algoritmo.

```
data1 = (data.groupby(["command_id", "product_name"])["qty"]
        .sum().unstack().reset_index().fillna(0).set_index("command_id"))

def hot_encode(n):
    return 1 if n >= 1 else 0

data2 = data1.applymap(hot_encode)
```

product_name	4 FORMAGGI.	4 Formaggi	4 Formaggi-	4 STAGIONI.	4 Stagioni	AGUA LLEVAR	ALBONDIGAS	ALMEJAS MARINERA	AZPILICUETA	Agua 0,5 L	...
command_id											
101	0	0	0	0	0	0	0	0	0	0	...
128	0	0	0	0	0	0	0	0	0	0	...
150	0	0	0	0	0	0	0	0	0	0	...
152	0	0	0	0	0	0	0	0	0	0	...
153	0	0	0	0	0	0	0	0	0	0	...

Figura 12. DataFrame “data2” (Apriori). Fuente: Elaboración propia

2.5. Algoritmo de asociación (Apriori):

Para desarrollar el algoritmo de asociación Apriori, comenzamos importando las funciones apriori y association_rules del módulo mlxtend.frequent_patterns. Luego, utilizando el DataFrame "data2", identificamos conjuntos frecuentes de elementos, considerando aquellos que aparecen con una frecuencia superior al 8% y tienen una longitud máxima de dos elementos.

Posteriormente, generamos reglas de asociación a partir de los conjuntos frecuentes identificados. La métrica utilizada para evaluar la fuerza de estas reglas es "lift", y solo seleccionamos aquellas que superan un umbral mínimo de 1 en la métrica de "lift". Las reglas resultantes se almacenan en un DataFrame llamado "data3", que contiene información relevante como los antecedentes, consecuentes, soporte del antecedente, soporte del consecuente y el soporte de la regla en sí.

```
from mlxtend.frequent_patterns import apriori

from mlxtend.frequent_patterns import association_rules

frq_items = apriori(data2, min_support=0.08, max_len=2, use_colnames=True)

data3 = association_rules(frq_items, metric= "lift",min_threshold=1)
[["antecedents", "consequents", "antecedent support", "consequent support", "support"]]
```

Al utilizar la función data3.head() sobre el algoritmo previamente entrenado, se exhibirán las asociaciones de productos más frecuentes. Esto brinda una visión de las relaciones más destacadas y recurrentes identificadas por este algoritmo.

	antecedents	consequents	antecedent support	consequent support	support
0	(Agua 0,5 L)	(Coca cola)	0.399766	0.177110	0.091203
1	(Coca cola)	(Agua 0,5 L)	0.177110	0.399766	0.091203
2	(Cola Zero)	(Agua 0,5 L)	0.149013	0.399766	0.080890
3	(Agua 0,5 L)	(Cola Zero)	0.399766	0.149013	0.080890

Figura 13. DataFrame “data3” (Apriori). Fuente: Elaboración propia

2.6. Creación algoritmo de probabilidad de comprar un producto cuando se ha comprado otro

Este algoritmo ofrece beneficios significativos al analizar patrones de compra y calcular la probabilidad de adquirir ciertos productos después de comprar uno específico. En primer lugar, proporciona una comprensión profunda del comportamiento del cliente, permitiendo a las empresas adaptar estrategias de marketing y ofrecer recomendaciones de productos más personalizadas. Esta información también respalda la optimización del inventario al anticipar la demanda de productos asociados, mejorando así la eficiencia en la cadena de suministro.

A nivel práctico, el algoritmo puede aumentar las oportunidades de ventas cruzadas y ventas adicionales al identificar productos que tienden a comprarse conjuntamente. Esto puede aprovecharse para diseñar promociones especiales y paquetes que incentiven a los clientes a adquirir productos relacionados.

En el proceso para realizar este algoritmo, en primer lugar, se importó la librería pandas y se cargaron los datos desde un archivo CSV mediante la función pd.read_csv de pandas. Luego, se almacenó en la variable "producto" el nombre del producto y se creó la función "hot_encode" para convertir los valores mayores a 1 en 1 y los menores de 1 en 0.

A continuación, se utilizó la función crosstab() para crear una tabla de contingencia por "command_id" y "producto_name”. Se aplicó la función "hot_encode" a la tabla de contingencia, se guardó en la variable "a" la cantidad de veces que se pidió el producto seleccionado y luego se filtraron las filas en el DataFrame "data1" donde los valores en la columna del producto seleccionado anteriormente son mayores o iguales a 1.

Posteriormente, se guardó en "data3" los productos que se pidieron al menos una vez cuando se solicitó el producto seleccionado anteriormente. Se cambiaron los nombres de las columnas por "Numero", "Nombre_del_Producto" y "Cantidad". Luego, se guardó en "data4" a partir de la fila 2, ya que la fila 1 contenía la suma total de las cuentas en las que estaba el producto.

Se ordenó por la cantidad y se creó una nueva columna "Probabilidad", que se obtuvo dividiendo la columna "Cantidad" entre la variable "a" que almacenaba la cantidad de veces que se pidió el producto, multiplicado por 100, para obtener la probabilidad de que se compre otro producto. Finalmente, se eliminó el producto ingresado en "Producto" del DataFrame.

Por último, se retuvieron las columnas "Nombre_del_Producto", "Cantidad" y "Probabilidad", y se guardaron en la variable "data6".

```
import pandas as pd

data = pd.read_csv("pedidos por mesa.csv", encoding='latin1')

producto="Agua 0,5 L"

def hot_encode(n):
    return 1 if n >= 1 else 0

contingency_table = pd.crosstab(data["command_id"],data["product_name"])
.reset_index()

data1 = contingency_table.applymap(hot_encode)

a = data1[y].sum()

data2 = data1[(data1[producto]>0)].sum().reset_index()

data3 = data2[(data2[0]>0)].reset_index()

data3.columns = ["Numero", "Nombre_del_Producto", "Cantidad"]

data4 = data3[(data3["Numero"]>0)].reset_index()

data4.sort_values(by=['Cantidad'], inplace = True, ascending= False)

data4["Probabilidad"]= data4["Cantidad"]/a*100
```

```
data5 = data4[(data4["Nombre_del_Producto"]!=producto)].reset_index()

data6=data5[["Nombre_del_Producto","Cantidad","Probabilidad"]]
```

Si aplicamos data6.head(), obtendremos la impresión del DataFrame del algoritmo antes entrenado. En este caso, el producto seleccionado fue "Agua 0,5 L". La columna "Cantidad" proporcionará la cantidad con la que se piden los otros productos cuando se solicita "Agua 0,5 L". Por otro lado, la columna "Probabilidad", como indica su nombre, mostrará la probabilidad de comprar otro producto en relación con la presencia de "Agua 0,5 L".

	Nombre_del_Producto	Cantidad	Probabilidad
0	Coca cola	1636	20.424469
1	Cola Zero	1451	18.114856
2	CaÃ±a	1386	17.303371
3	SÃºper bbq	1369	17.091136
4	Prosciutto	1337	16.691635

Figura 14. DataFrame “data3” (Algoritmo probabilidad de comprar un producto cuando se ha comprado otro). Fuente: Elaboración propia

2.7. Entrenamiento de los algoritmos

El entrenamiento del modelo es una etapa crucial en el proceso de minería de datos, donde el objetivo principal es dotar al algoritmo con la capacidad de aprender patrones y relaciones intrínsecas en los datos. Este proceso implica utilizar un conjunto de datos previamente etiquetado para permitir que el modelo ajuste sus parámetros de manera que se adapten de manera óptima a los patrones presentes en los datos. Durante este entrenamiento, el modelo busca minimizar la diferencia entre las predicciones que genera y las etiquetas reales asociadas con los datos de entrenamiento. Este ajuste se realiza a través de técnicas como la optimización de parámetros, donde se busca encontrar la configuración más adecuada que optimice el rendimiento del modelo en la tarea específica. Es fundamental encontrar un equilibrio para evitar el sobreajuste, donde el modelo se adapta demasiado a los datos de entrenamiento y pierde la capacidad de generalizar a nuevos datos. La evaluación constante del desempeño del modelo con conjuntos de datos independientes es esencial

durante este proceso para garantizar su capacidad de realizar predicciones precisas en entornos del mundo real. El éxito del entrenamiento del modelo es esencial para lograr los objetivos definidos en las primeras etapas del proceso de minería de datos.

Entrenamiento del algoritmo de predicción de Ventas:

Para entrenar el algoritmo de predicción de ventas, primero se importó la librería Prophet. Luego, se creó una instancia del modelo Prophet. Se configuró el modo de estacionalidad como 'multiplicative'.

Se agregó una estacionalidad anual al modelo, denominada 'año', con un período de 365 días, lo que sugiere un ciclo anual y el parámetro "fourier_order" se estableció en 1.

El método model.fit(data2) ajustó el modelo a los datos de la serie temporal proporcionados en "data2". Este paso implica que el modelo "aprende" de los datos históricos y ajusta sus parámetros internos para hacer pronósticos futuros.

```
from prophet import Prophet

model = Prophet()

model = Prophet(seasonality_mode='multiplicative')

model.add_seasonality(name='año', period=365, fourier_order=1)

model.fit(data2)
```

Entrenamiento del algoritmo de predicción de Ventas por productos:

Para entrenar el algoritmo de predicción de ventas por producto, el proceso será muy similar al del algoritmo de predicción de ventas. La única diferencia es que ajustaremos el modelo utilizando el conjunto de datos "data4".

```
from prophet import Prophet

model = Prophet()

model = Prophet(seasonality_mode='multiplicative')

model.add_seasonality(name='year', period=365, fourier_order=1)

model.fit(data4)
```

2.8. Conclusiones del Capítulo

1. Se resalta la importancia del análisis exploratorio de datos como una herramienta fundamental para comprender la estructura de la información, identificar anomalías y problemas en los datos, y seleccionar características relevantes. El análisis se respalda con gráficos que visualizan el comportamiento de las ventas en relación con el tipo de pedido y a lo largo del tiempo.
2. A través del análisis exploratorio de datos, se observó un notorio aumento en las ventas a partir del 1 de mayo de 2021. Este hallazgo influyó en la decisión de obviar los datos anteriores al mencionado día al entrenar el modelo, reconociendo así la importancia de centrarse en el periodo donde se experimenta un cambio significativo en el comportamiento de las ventas.
3. Se destaca la necesidad de contar con un histórico de datos extenso para obtener predicciones óptimas en algoritmos de predicción. En este caso, se empleó un conjunto de datos que abarca información desde el 20 de septiembre de 2020.

Capítulo III. Experimentación, discusión y análisis de resultados

3.1. Introducción del Capítulo

Con la propuesta de modelos ya terminada solo queda analizarla con el fin de corroborar los resultados que se obtienen. Con el objetivo de saber si los resultados ofrecidos por el sistema son los acertados, se realizaron diferentes pruebas, con el objetivo de mejorar los resultados con cada una de ellas. Todos estos ensayos se explicarán en el presente capítulo, así como un pequeño análisis y comparación de estos.

3.2. Predicción de las ventas

Predecir las ventas es un desafío complejo y depende de varios factores, incluyendo el tipo de producto o servicio, la industria, la temporada, la economía y más.

La predicción de ventas proporciona una serie de beneficios cruciales para las empresas, estableciendo una base sólida para la toma de decisiones estratégicas y la planificación efectiva. En primer lugar, permite a las empresas anticipar y responder proactivamente a cambios en la demanda del mercado, minimizando los excesos o déficits de inventario. Esto no solo optimiza la gestión de recursos, sino que también mejora la eficiencia operativa al evitar situaciones de sobreproducción o falta de productos en el momento crucial. Además, la predicción de ventas facilita la formulación de estrategias de marketing más precisas y personalizadas, ya que las empresas pueden adaptar sus campañas a patrones identificados en el comportamiento del consumidor. Al conocer las tendencias futuras, las empresas también están mejor equipadas para aprovechar oportunidades de crecimiento y mitigar riesgos potenciales. En última instancia, la predicción de ventas contribuye a la optimización general de la cadena de suministro, la gestión financiera y la satisfacción del cliente, posicionando a las empresas de manera más competitiva en un entorno empresarial dinámico.

La predicción de ventas es una parte importante de la inteligencia empresarial moderna. Puede ser un problema complejo, sobre todo en caso de falta de datos, datos ausentes y presencia de valores atípicos. (Pavlyshenko 2019)

Modelo de predicción de ventas:

Para llevar a cabo las predicciones mediante este modelo, se inició guardando la fecha inicial del rango a predecir en la variable "start_date" y la fecha final en "end_date". A continuación, se generó un rango de fechas entre estas 2 variables, utilizando la frecuencia diaria ('D'). Este rango de fechas se transformó en un DataFrame de pandas, incorporando una columna denominada 'ds' que contiene las fechas.

Posteriormente, haciendo uso de un modelo que previamente había sido entrenado, se realizaron predicciones en el conjunto de datos futuro (future). El resultado de estas predicciones se almacenó en el DataFrame llamado "forecast".

Seguidamente, se procedió a crear un nuevo DataFrame denominado "predictions_df", el cual únicamente retiene las columnas 'ds' (fechas) y 'yhat' (predicción) del DataFrame de pronóstico “forecast”. Además, se modificó los nombres a estas columnas, designándolas como 'Fecha' y 'Predicción'. Este paso se realizó con el propósito de proporcionar una representación más clara y comprensible de los resultados obtenidos a través de las predicciones.

```
start_date = '2023-06-01'

end_date = '2023-10-05'

future = pd.date_range(start=start_date, end=end_date, freq='D')

future = pd.DataFrame({'ds': future})

forecast = model.predict(future)

predictions_df = forecast[['ds', "yhat"]].rename(columns={'ds': 'Fecha', 'yhat': 'Predicción'})
```

Ahora podemos examinar las primeras 5 filas que contienen las predicciones re alizadas.

	Fecha	Predicción
0	2023-05-01	588.968567
1	2023-05-02	707.974551
2	2023-05-03	622.442227
3	2023-05-04	727.621575
4	2023-05-05	1774.010598

Figura 15. DataFrame “predictions_df” (predicción de ventas). Fuente: Elaboración propia

Modelo de predicción de ventas por productos:

Siguiendo un proceso similar al implementado en el modelo de predicción de ventas, se llevó a cabo la predicción de ventas por productos. Se adoptó una metodología similar para establecer el rango de fechas que se deseaba pronosticar, convirtiéndolo posteriormente en un DataFrame. Las predicciones se realizaron utilizando el modelo que previamente había sido entrenado.

```
start_date = '2023-06-01'

end_date = '2023-10-05'

future = pd.date_range(start=start_date, end=end_date, freq='D')

future = pd.DataFrame({'ds': future})

forecast = model.predict(future)

predictions_df = forecast[['ds', 'yhat']].rename(columns={'ds': 'Fecha', 'yhat': 'Predicción'})
```

	Fecha	Predicción
0	2023-05-01	10
1	2023-05-02	11
2	2023-05-03	10
3	2023-05-04	12
4	2023-05-05	28

Figura 16. DataFrame “predictions_df” (predicción de ventas por productos). Fuente: Elaboración propia

3.3. Técnica de Validación para algoritmos de Series Temporales.

MAPE

Porcentaje de Error Medio Absoluto (MAPE) es una métrica utilizada en estadísticas y análisis de series temporales para evaluar la precisión de un modelo de pronóstico. Este mide la magnitud promedio de los errores en términos porcentuales.

El MAPE, es una métrica porcentual que normaliza los valores del error a la escala de la serie, y permite hacer una comparativa de las precisiones de las diferentes series. Su cálculo es sencillo y fácil de interpretar como el MAE, con la única diferencia que para el MAPE este valor se normaliza. (BLANCO, MARÍ, PEREZ-MUELAS 2019)

La fórmula para el MAPE es la siguiente:

$$\boldsymbol{MAPE} = \frac{\sum_{t=1}^{n} \left| \frac{(y_t - \widehat{y_t})}{y_t} (100) \right|}{n}$$

Dónde:

$\boldsymbol{y_t}$: Es el valor observado (valor real del indicador)

$\boldsymbol{\widehat{y_t}}$: Es el valor pronosticado (predicción del indicador)

$\boldsymbol{n}$: Es la cantidad de observaciones

El MAPE calcula el porcentaje promedio de error absoluto en relación con los valores reales. Una ventaja del MAPE es que proporciona una medida relativa de error, lo que significa que es independiente de la escala de los datos. Sin embargo, el MAPE puede ser sensible a los casos en los que los valores reales son cercanos o iguales a cero.

Al interpretar el MAPE, se busca minimizar su valor. Un MAPE más bajo indica una mejor precisión del modelo en términos porcentuales. Al igual que con cualquier métrica, es importante considerar el contexto específico del problema y las características del conjunto de datos al interpretar los resultados del MAPE.

MAE

El "Error Absoluto Medio" (MAE) es una métrica utilizada comúnmente en estadísticas y aprendizaje automático para evaluar la precisión de un modelo de

predicción. Se calcula tomando la media de las diferencias absolutas entre las predicciones del modelo y los valores reales. La fórmula para el MAE es la siguiente:

$$\boldsymbol{MAE} = \frac{\sum_{t=1}^{n} |(y_t - \hat{y}_t)|}{n}$$

$\boldsymbol{y_t}$: Es el valor observado (valor real del indicador)

$\boldsymbol{\hat{y}_t}$: Es el valor pronosticado (predicción del indicador)

$\boldsymbol{n}$: Es la cantidad de observaciones

En términos simples, el MAE representa el promedio de las diferencias absolutas entre las predicciones y los valores reales. Cuanto menor sea el valor de MAE, mejor será el rendimiento del modelo, ya que indica que las predicciones están más cercanas a los valores reales.

3.4. Análisis de los resultados de los algoritmos de predicción.

En esta sección, se lleva a cabo un análisis detallado de los resultados generados por los algoritmos y modelos de predicción implementados a lo largo de la investigación. Este análisis profundo se revela como una fase esencial para comprender la eficacia inherente de cada uno de estos, contextualizándolos en relación con la tarea específica para la cual fueron diseñados.

Algoritmo de predicción de ventas:

Se consideró inicialmente la utilización de dos modelos: Prophet y ARIMA. Se implementaron ambos modelos con el mismo rango de fechas para la predicción, y se evaluaron los resultados mediante las técnicas de validación MAPE (Error porcentual absoluto medio) y MAE (Error absoluto medio). Las predicciones se realizaron para el período comprendido desde el 1 de junio de 2023 hasta el 5 de octubre de 2023, y se compararon con las ventas reales de ese periodo para validar la precisión de ambos modelos.

Prophet

Para aplicar las técnicas de validación MAPE y MAE a este modelo, primero realizamos una unión entre los dataframes "predictions_df" y "data1". Donde "predictions_df" representa las predicciones y "data1" es el dataframe completo

cargado. Luego, eliminamos la columna “ds” para evitar tener dos columnas de fechas. Creamos una nueva columna llamada “diferencia” que representa la diferencia entre “predicción” e “importe”, asegurándonos de que todos los resultados sean números positivos mediante la función abs(). Posteriormente, renombramos la columna “y” como “Importe” y redondeamos todas las columnas del dataframe a dos lugares decimales.

Seguidamente, en la variable "y_true" almacenamos los valores de la columna "Importe" y en "y_pred" guardamos los valores de "Predicción". Luego, calculamos el MAE utilizando la librería sklearn.metrics al pasarle los valores de "y_true" y "y_pred". Además, calculamos el MAPE utilizando la librería numpy. Estos pasos nos permiten evaluar la precisión del modelo mediante dos métricas comúnmente utilizadas en la validación de modelos de predicción.

```
import numpy as np

from sklearn.metrics import mean_absolute_error

data_main = pd.merge(left=predictions_df, right=data1, how="inner", left_on="Fecha", right_on="ds")

data_main.drop(columns=['ds'], inplace=True)

data_main['diferencia'] = data_main['Predicción'] - data_main['y']

data_main['diferencia'] = data_main['diferencia'].abs()

data_main=data_main[["Fecha","y","Predicción","diferencia"]].rename(columns={'y': 'Importe'})

data_main=round(data_main, 2)

y_true = data_main['Importe']

y_pred = data_main['Predicción']

mae = round(mean_absolute_error(y_true, y_pred), 2)

mape = round(np.mean(np.abs((y_true - y_pred) / y_true) * 100), 2)
```

Los resultados obtenidos para este modelo revelan un MAPE del 25.2% y un MAE de 293.0. Cabe destacar que estas métricas se evalúan en relación con un

promedio del importe de las ventas reales de 1369.27€. Estos resultados indican una predicción sólida, esto es especialmente destacado considerando la naturaleza potencialmente cambiante de las ventas de un bar, donde estas pueden variar significativamente. El hecho de que el modelo haya demostrado una predicción sólida bajo estas condiciones dinámicas subraya su capacidad para adaptarse y ofrecer resultados confiables incluso en situaciones comerciales que tienden a ser impredecibles.

Al ejecutar data_main.head(10) y data_main.tail(10), obtenemos las primeras y últimas 10 filas del dataframe respectivamente. Esto proporciona una visión detallada de las predicciones precisas realizadas por este modelo en diversos puntos del conjunto de datos.

	Fecha	Predicción	price	diferencia
0	2023-06-01	732.403952	426.20	306.20
1	2023-06-02	1776.227781	2040.75	264.52
2	2023-06-03	2240.431629	2300.87	60.44
3	2023-06-04	1636.388300	706.70	929.69
4	2023-06-05	649.069232	342.20	306.87
5	2023-06-06	761.260443	507.20	254.06
6	2023-06-07	681.522945	712.20	30.68
7	2023-06-08	786.725579	612.57	174.16
8	2023-06-09	1822.767584	1665.40	157.37
9	2023-06-10	2277.662321	2150.90	126.76

Figura 17. 10 primeras filas predicción de ventas (Prophet). Fuente: Elaboración propia

	Fecha	Predicción	price	diferencia
117	2023-09-26	734.107124	492.10	242.01
118	2023-09-27	656.964142	594.90	62.06
119	2023-09-28	773.833115	475.50	298.33
120	2023-09-29	1856.670552	2342.17	485.50
121	2023-09-30	2339.955914	2580.47	240.51
122	2023-10-01	1716.687401	1939.40	222.71
123	2023-10-02	697.417207	980.50	283.08
124	2023-10-03	820.324406	754.80	65.52
125	2023-10-04	745.364689	743.90	1.46
126	2023-10-05	863.285854	712.30	150.99

Figura 18. 10 últimas filas predicción de ventas (Prophet). Fuente: Elaboración propia

Se realizó una representación visual mediante un gráfico de líneas utilizando la librería Matplotlib. En este gráfico, las líneas rojas representan las predicciones del modelo, mientras que las líneas azules representan las ventas reales. Esta representación gráfica ofreció una visión comparativa y fácil de interpretar de la concordancia entre las predicciones y los datos reales a lo largo del tiempo.

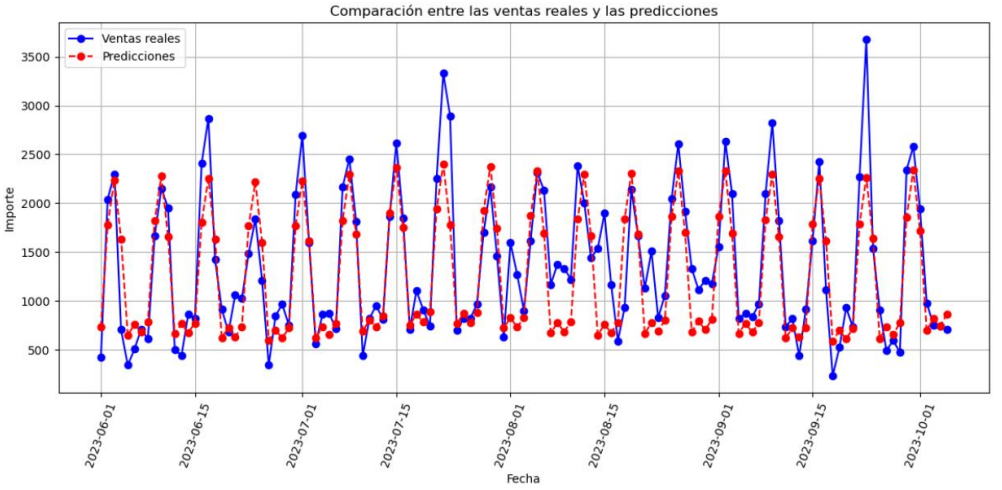

Figura 19. Comparación entre las ventas reales y las predicciones (Prophet).

Fuente: Elaboración propia

ARIMA

La elección de utilizar el modelo ARIMA con los parámetros (2,1,1) se fundamentó en pruebas y evaluaciones exhaustivas previas. Estos parámetros específicos fueron identificados como los que ofrecieron las predicciones más precisas al compararlas con otras configuraciones en el mismo rango de fechas. Obteniendo un MAPE del 47.78% y un MAE de 669.94. Al mostrar las primeras y últimas 10 filas, como se hizo anteriormente, se evidencia el comportamiento de las ventas reales y la predicción.

	Fecha	Importe	Predicción	diferencia
0	2023-06-01	426.20	661.80	235.60
1	2023-06-02	2040.75	800.49	1240.26
2	2023-06-03	2300.87	820.98	1479.89
3	2023-06-04	706.70	958.63	251.93
4	2023-06-05	342.20	733.67	391.47
5	2023-06-06	507.20	856.13	348.93
6	2023-06-07	712.20	1151.26	439.06
7	2023-06-08	612.57	766.92	154.35
8	2023-06-09	1665.40	1065.74	599.66
9	2023-06-10	2150.90	778.55	1372.35

Figura 20. 10 primeras filas predicción de ventas (ARIMA). Fuente: Elaboración propia

	Fecha	Importe	Predicción	diferencia
117	2023-09-26	492.10	1019.72	527.62
118	2023-09-27	594.90	862.82	267.92
119	2023-09-28	475.50	621.66	146.16
120	2023-09-29	2342.17	901.11	1441.06
121	2023-09-30	2580.47	811.27	1769.20
122	2023-10-01	1939.40	827.98	1111.42
123	2023-10-02	980.50	771.07	209.43
124	2023-10-03	754.80	785.69	30.89
125	2023-10-04	743.90	790.94	47.04
126	2023-10-05	712.30	902.71	190.41

Figura 21. 10 últimas filas predicción de ventas por productos (ARIMA). Fuente: Elaboración propia

La librería matplotlib fue empleada para la creación de un gráfico de líneas, proporcionando una visualización más clara y detallada del comportamiento de las ventas reales y las predicciones. Este enfoque gráfico permite una comparación efectiva entre los valores predichos por el modelo ARIMA con (2,1,1) y las ventas reales a lo largo del período evaluado.

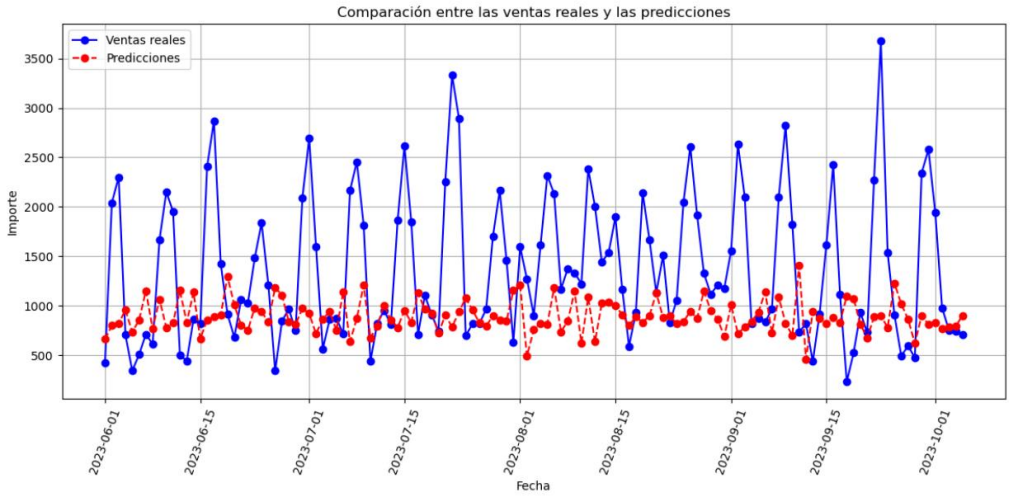

Figura 22. Comparación entre las ventas reales y las predicciones (ARIMA).

Fuente: Elaboración propia

Elección del modelo a utilizar:

La decisión de optar por el modelo Prophet en lugar de ARIMA se basa en una evaluación exhaustiva de sus resultados. Al comparar las técnicas de validación MAPE y MAE, se observa claramente que Prophet presenta un rendimiento superior. Con un MAPE del 25.2% y un MAE de 293.0, Prophet supera significativamente al modelo ARIMA (2,1,1), que registra un MAPE del 47.78% y un MAE de 669.94. Estas diferencias destacadas en la precisión de las predicciones respaldan la elección de Prophet como el modelo preferido tanto para la predicción de Ventas como el de predicción de ventas por productos.

Modelo de predicción de ventas por productos:

Para aplicar las técnicas de validación MAPE y MAE al modelo de predicción de ventas por productos, se siguió un proceso similar al empleado en el modelo de predicción general de ventas. En una fase inicial, se creó un nuevo DataFrame denominado "data5" filtrando las filas donde el valor de "producto_name" coincidía con el producto almacenado en la variable "producto". Posteriormente, se procedió a agrupar por la columna "created_at" y sumar la columna "qty" para obtener la cantidad total de veces que se pidió el producto en cada día.

A continuación, se realizó una unión entre los DataFrames "predictions_df" y "data6", donde "predictions_df" representaba el DataFrame de las predicciones y "data6" correspondía al DataFrame completo. Con el objetivo de evitar

redundancias en las fechas, se eliminó la columna "created_at". Se creó una nueva columna llamada "diferencia", que representaba la diferencia entre las predicciones y las ventas reales ("qty"). El uso de la función abs() garantizó que esta diferencia siempre fuera un número positivo.

Seguidamente, se asignaron las columnas "qty" y "Predicción" a las variables "y_true" y "y_pred" respectivamente. Finalmente, se calculó el MAE utilizando la librería sklearn.metrics, proporcionando las variables "y_true" e "y_pred". Este proceso permitió evaluar la precisión del modelo de predicción de ventas por productos mediante la métrica MAE.

```
from sklearn.metrics import mean_absolute_error

data5 = data[data['product_name'] == producto]

data6 = data5.groupby('created_at')['qty'].sum().reset_index()

data6['created_at'] = pd.to_datetime(data6['created_at'])

data_main = pd.merge(left=predictions_df, right=data6, how="inner", left_on="Fecha", right_on="created_at")

data_main.drop(columns=['created_at'], inplace=True)

data_main['diferencia'] =(data_main['Predicción'] - data_main['qty']).abs()

y_true = data_main['qty']

y_pred = data_main['Predicción']

mae = mean_absolute_error(y_true, y_pred)
```

En el caso específico de seleccionar el producto "Agua 0,5 L", que previamente se identificó como el producto más solicitado según el análisis exploratorio de datos, se obtuvo un MAE de 6.19.

Al realizar un vistazo a las primeras 10 filas del DataFrame, se puede apreciar el comportamiento detallado de las predicciones y las ventas reales para dicho producto. Este análisis más detallado proporciona una visión concreta sobre la eficacia del modelo de predicción específicamente para el producto más popular.

	Fecha	Predicción	qty	diferencia
0	2023-06-01	14	5	9
1	2023-06-02	30	42	12
2	2023-06-03	42	34	8
3	2023-06-04	26	7	19
4	2023-06-05	13	6	7
5	2023-06-06	13	1	12
6	2023-06-07	13	18	5
7	2023-06-08	14	8	6
8	2023-06-09	30	21	9
9	2023-06-10	42	32	10

Figura 23. 10 primeras filas predicción de ventas del producto “Agua 0,5 L”. Fuente: Elaboración propia

En el caso de analizar el producto "Caña", que también se identificó como uno de los más consumidos, se obtuvo un MAE de tan solo 8.01. Este valor indica que la diferencia promedio entre las predicciones y la cantidad real de ventas para el producto "Caña" es mínima, demostrando así también la precisión del modelo para este producto en particular. Al igual que en el análisis del product anterior vamos a mostrar las primeras 10 filas.

	Fecha	Predicción	qty	diferencia
0	2023-06-01	7	1	6
1	2023-06-02	12	14	2
2	2023-06-03	14	19	5
3	2023-06-04	10	1	9
4	2023-06-06	6	9	3
5	2023-06-07	6	1	5
6	2023-06-08	7	1	6
7	2023-06-09	12	17	5
8	2023-06-10	15	20	5
9	2023-06-11	11	17	6

Figura 24. 10 primeras filas predicción de ventas del producto “Caña”. Fuente: Elaboración propia

Analizando otro de los productos de mayor venta como el “Prosciutto” nos va a devolver como el MAE es de solo 2.02.

	Fecha	Predicción	qty	diferencia
0	2023-06-02	6	6	0
1	2023-06-03	7	9	2
2	2023-06-04	4	1	3
3	2023-06-09	6	6	0
4	2023-06-10	8	8	0
5	2023-06-11	4	8	4
6	2023-06-12	3	1	2
7	2023-06-13	3	2	1
8	2023-06-14	2	1	1
9	2023-06-15	2	1	1

Figura 25. 10 primeras filas predicción de ventas del producto “Prosciutto”.

Fuente: Elaboración propia

En el análisis del producto "Alemana", que no tiene una demanda tan alta y no se solicita diariamente, se observó que el modelo hace predicciones precisas para los días en que se solicita este producto. El MAE obtenido es de tan solo 0.81, indicando una discrepancia mínima entre la cantidad real de ventas y las predicciones del modelo. A pesar de la menor demanda de este producto, el modelo logra ajustarse de manera efectiva a los patrones de venta específicos, demostrando su capacidad de adaptación a diferentes niveles de demanda.

	Fecha	Predicción	qty	diferencia
0	2023-06-02	1	2	1
1	2023-06-04	1	1	0
2	2023-06-07	1	1	0
3	2023-06-09	1	1	0
4	2023-06-10	2	2	0
5	2023-06-11	1	8	7
6	2023-06-17	2	2	0
7	2023-06-19	1	1	0
8	2023-06-21	1	1	0
9	2023-06-22	1	1	0

Figura 26. 10 primeras filas predicción de ventas del producto “Alemana”.

Fuente: Elaboración propia

3.5. Análisis de los resultados de los algoritmos de asociación.

Para el análisis de los resultados de los algoritmos de asociación no vamos a utilizar ninguna técnica de validación, para ello vamos a hacer otro tipo de análisis para comprobar que los resultados que devuelve este algoritmo son los correctos.

Apriori:

Con el modelo ya creado, procedimos a filtrar en data2 las cuentas en las que se piden "Agua 0,5 L" y "Coca Cola" al mismo tiempo. Al imprimir la cantidad de cuentas en el dataframe y la cantidad de veces que se solicitan ambos productos simultáneamente, obtuvimos información sobre la frecuencia de esta combinación específica de productos en las transacciones.

```
filtro = (data2['Agua 0,5 L'] == 1) & (data2['Coca cola'] == 1)

resultados_filtrados = data2[filtro]

print("Cantidad de cuentas:",data2.shape[0])

print("Cantidad de veces que se pidió Agua 0,5 L y Coca cola:",resultados_filtrados.shape[0])
```

Esto nos proporcionará que la cantidad de cuenta es de 17938 y la cantidad de veces que se compraron ambos productos al mismo tiempo es de 1636. Por lo que si dividimos esto nos va a dar como resultado un 0.091203. Por lo que coincide con el soporte que devuelve para estos productos el algoritmo Apriori.

Probabilidad de comprar un producto cuando se compre otro:

Para este algoritmo vamos a hacer algo parecido a lo que hicimos en el caso anterior, filtramos por cuando se pide “Agua 0,5 L” y “Prosciutto” al mismo tiempo, luego imprimimos la cantidad de veces que se pidió el producto “Agua 0,5 L” y luego cuando se piden en combinación.

```
filtro = (data1['Agua 0,5 L'] == 1) & (data1['Prosciutto'] == 1)

resultados_filtrados = data1[filtro]

print("Cantidad de veces que se pidió el producto Agua 0,5 L:",data1["Agua 0,5 L"].sum())

print("Cantidad de veces que se pidió el producto Agua 0,5 L y Prosciutto al mismo tiempo:",resultados_filtrados.shape[0])
```

Con esto comprobaremos si la columna cantidad está correcta al igual que el cálculo de probabilidades, esto nos va a devolver que el “Agua 0,5 L” se pidió 7171 veces, y en combinación con el “Prosciutto”, ambos se pidieron 1337. Por

lo que coincide por lo devuelto por este algoritmo, si dividimos la probabilidad de la combinación entre las que se pidió agua solamente y lo multiplicamos por 100, nos va a dar que la probabilidad de que se pida “Prosciutto” cuando se ha pedido el producto “Agua 0,5 L” es de 18.64%, por lo que coincide con el resultado del algoritmo, comprobando de esta forma que está correcta su implementación.

3.6. Módulo de Análisis de ventas

Se llevó a cabo el desarrollo de un módulo de reportes para satisfacer las demandas del cliente. Este, se creó con la finalidad de facilitar y mejorar el proceso de toma de decisiones, empleando los algoritmos que se habían estudiado con anterioridad.

Uno de los aspectos destacados de esta implementación fue la capacidad de realizar predicciones de ventas y de ventas por producto. Esto se logró mediante la aplicación del algoritmo Prophet, el cual demostró ser una herramienta efectiva para modelar y prever tendencias en datos de series temporales, como es el caso de las ventas.

Adicionalmente, incorporamos el algoritmo de asociación Apriori en el módulo. Este algoritmo posibilita el análisis de patrones de comportamiento de los clientes en función de sus historiales de compras. Además, creamos un algoritmo para calcular la probabilidad de que un cliente adquiera un producto dado que ya ha comprado otro anteriormente. Esta información resultó valiosa para comprender las relaciones entre distintos productos y diseñar estrategias que optimizasen la probabilidad de venta cruzada.

Algoritmo de predicción de ventas

Para realizar la predicción de ventas, simplemente introduces el rango de fechas que deseas pronosticar y especificas el tipo de pedidos. Esto te proporciona una tabla de dos columnas: una que muestra la fecha y la otra que presenta la predicción correspondiente. Este método te permite anticipar las ventas futuras basándote en la información proporcionada, ofreciendo una herramienta efectiva para la toma de decisiones comerciales. La simplicidad del procedimiento y la claridad de los resultados te brindan una visión rápida y precisa de lo que puedes esperar en términos de rendimiento de ventas en un período determinado.

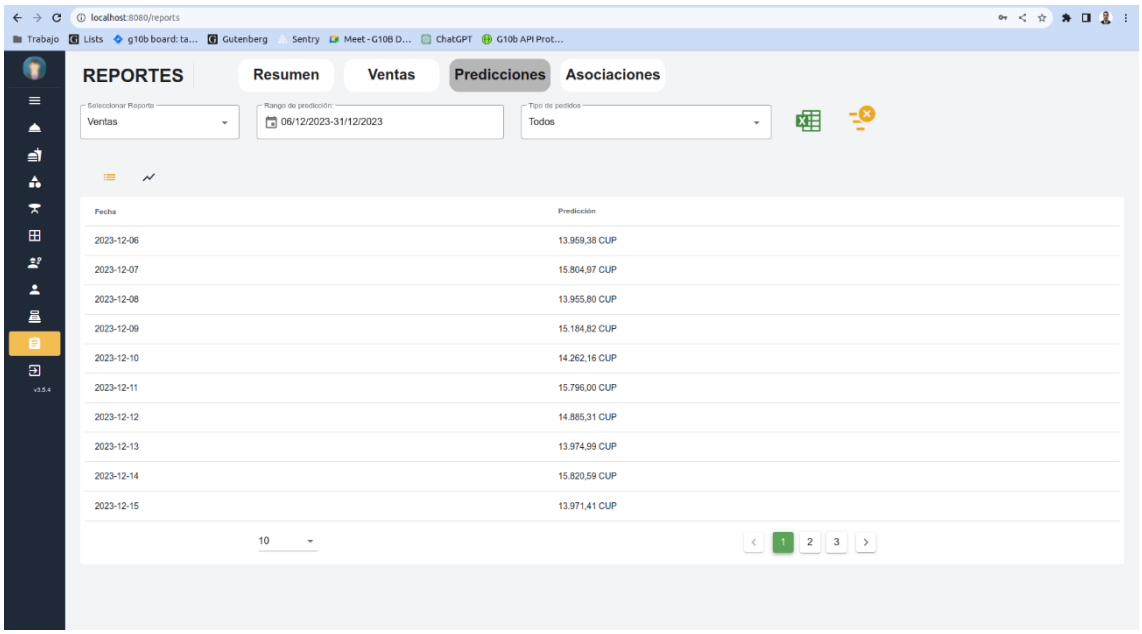

Figura 27. Vista reporte predicción de ventas. Fuente: Elaboración propia

Algoritmo de Predicción de ventas por productos

La predicción de ventas por productos sigue un proceso muy parecido al de ventas. La única variación radica en que, además de ingresar el rango de fechas y el tipo de pedidos, se añade la especificación del producto para el cual se realizará la predicción. Al ejecutar este proceso, obtendrás una tabla de dos columnas que detalla la fecha y la predicción correspondiente, teniendo en cuenta el producto específico seleccionado. Esto ofrece una manera efectiva de anticipar las ventas futuras de los productos.

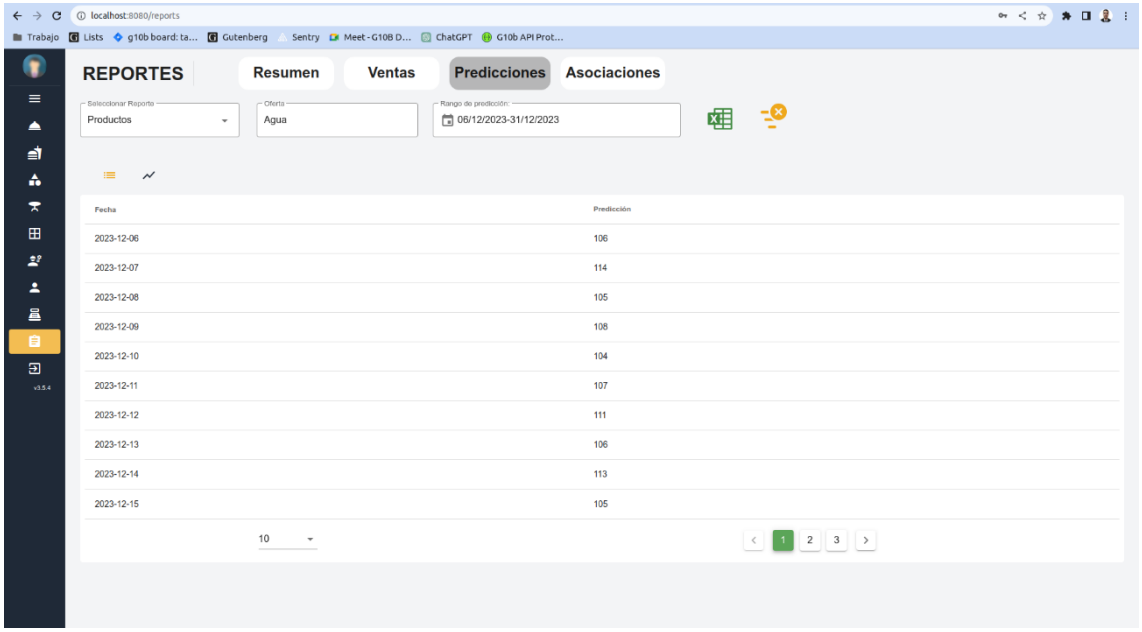

Figura 28. Vista reporte predicción de ventas por productos. Fuente: Elaboración propia

Apriori

Para implementar el algoritmo Apriori, se requiere establecer la probabilidad conjunta, la cantidad de productos y el tipo de pedido. Este proceso permite una evaluación más precisa de las relaciones entre variables, lo que resulta fundamental para identificar patrones de asociación significativos en grandes conjuntos de datos.

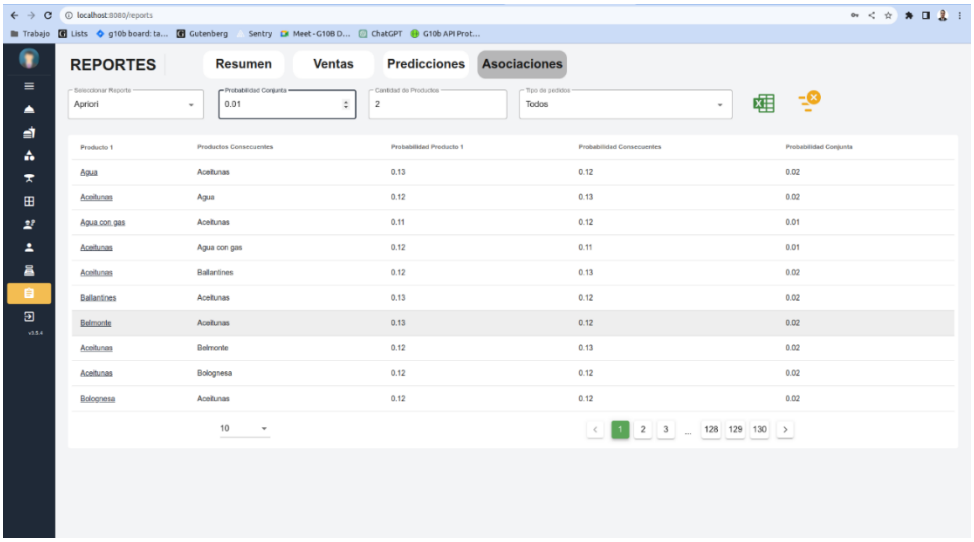

Figura 29. Vista reporte Apriori. Fuente: Elaboración propia

Probabilidad de comprar un producto cuando se ha comprado otro

Al aplicar el algoritmo de probabilidad de compra de un producto dado que se ha comprado otro, es necesario elegir tanto el producto como el tipo de pedidos. Al realizar esta selección, se generará una tabla que consta de tres columnas: la primera identificará los productos específicos, la segunda mostrará la cantidad de veces que se ha pedido dicho y la tercera columna presentará la probabilidad de que esto suceda.

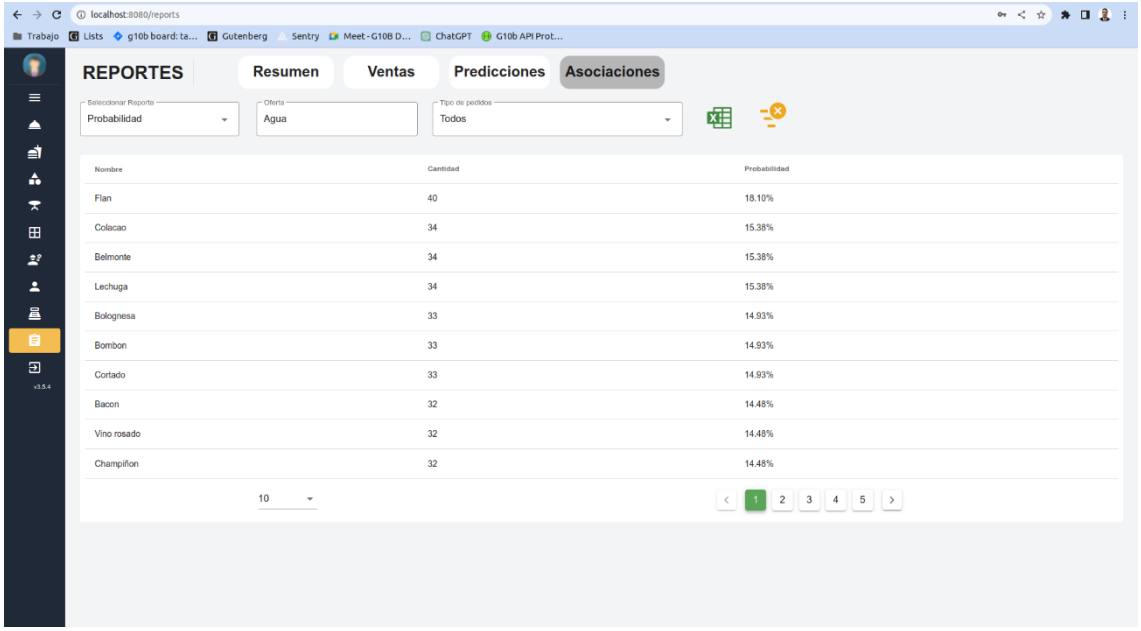

Figura 30. Vista reporte de Probabilidad de comprar un producto cuando se ha comprado otro. Fuente: Elaboración propia

Cada uno de estos informes tiene la capacidad de ser exportado a Excel. Esto significa que los resultados y datos generados por los algoritmos, ya sea la predicción de ventas, por producto, el análisis de reglas de asociación con el algoritmo Apriori, o la probabilidad de compra de un producto dado que se ha comprado otro, pueden ser transferidos fácilmente a una hoja de cálculo de Excel. Esta funcionalidad proporciona una mayor flexibilidad para que los usuarios trabajen, analicen y compartan los datos de manera conveniente, aprovechando las herramientas y funcionalidades adicionales que Excel ofrece para el análisis y la presentación de datos.

3.7. Conclusiones del Capítulo

Capítulo III. Experimentación, discusión y análisis de resultados

En este capítulo se expuso todo lo referente a la experimentación, discusión y análisis de los resultados obtenidos, una vez detallados estos aspectos se arribaron a las siguientes conclusiones:

1. En este caso, se optó por el modelo Prophet sobre el modelo ARIMA debido a sus predicciones superiores, fundamentando así su elección.
2. El Porcentaje de Error Medio Absoluto (MAPE) es una métrica esencial para evaluar la precisión de modelos de pronóstico en series temporales. Mide la magnitud promedio de errores en términos porcentuales, ofreciendo una medida relativa independiente de la escala de los datos.
3. El Error Absoluto Medio (MAE) emerge como una métrica clave en la evaluación de la precisión de modelos de predicción en estadísticas y aprendizaje automático. Su cálculo, que toma la media de las diferencias absolutas entre las predicciones y los valores reales, proporciona una medida directa de la discrepancia entre ambos.
4. Se seleccionaron como las técnicas de validación para los algoritmos de predicción el MAPE (Porcentaje de Error Medio Absoluto) y el MAE (Error Absoluto medio), aunque existen otras, estas son de las más utilizadas en modelos de series temporales.
5. Los resultados obtenidos en los modelos tanto de predicción de ventas como el de predicción de venta por productos demostraron que son altamente fiables ya que sus predicciones tienen un alto grado de precisión.
6. En el caso de los algoritmos de asociación, se demostró su utilidad al aprovechar las combinaciones de productos más solicitadas, destacando su eficacia para extraer información valiosa y mejorar la toma de decisiones.

Conclusiones Generales

Como resultado de esta investigación quedaron satisfechos los objetivos trazados arribando a las siguientes conclusiones:

1. El estudio realizado sobre los antecedentes, el estado actual de la temática, la bibliografía y documentos relacionados con el objeto de estudio, permitió aportar los elementos necesarios para dar solución a la problemática planteada.
2. Los antecedentes encontrados, vinculados al tema no le dan solución al problema planteado por lo que no es factible su utilización.
3. Se utilizaron las herramientas de software más factibles para la construcción de la solución.
4. Se logró la implementación de modelos predictivos que permitirán procesar los datos y realizar inferencias futuras.
5. Es necesario un histórico de datos amplio para lograr valores de predicción óptimos.
6. De los dos modelos utilizados para la predicción de las ventas, ARIMA y Prophet, este último hace mejores predicciones, de ahí su elección.
7. Los algoritmos de asociación demostraron que son muy beneficiosos para extraer información valiosa y mejorar la toma de decisiones.
8. Los resultados obtenidos en los modelos de predicción de indicadores comprobaron que son fiables y que sus pronósticos tienen un alto grado de precisión.
9. El hecho de que las predicciones sean muy cercanas a la realidad permite emitir criterios acertados para evaluar una situación en un espacio de tiempo determinado.

Referencias Bibliográficas

LÓPEZ DE MUNAIN, Claudia, et al. Sistemas de apoyo a la toma de decisiones. En *XVI Workshop de Investigadores en Ciencias de la Computación*. 2014.

PÉREZ, Yosmaurereen Naomi Villachica; CUTHBERT, Deyvon Kestner Ordoñez; SAMBOLA, Dexon Mckensy. Modelo predictivo basado en Machine Learning dirigido a PYMES de venta, caso de estudio Bluefields. *Ciencia e Interculturalidad*, 2022.

HINCAPIÉ HERRERA, Edwar Andrés. *Predicción de la Demanda Usando Modelos de Machine Learning*. 2021.

Juárez, e. a. (2016). *Análisis de series de tiempo en el pronóstico de la demanda de almacenamiento de productos perecederos.*

BAJAJ, Purvika, et al. Sales prediction using machine learning algorithms. *International Research Journal of Engineering and Technology (IRJET)*, 2020.

CORSO, Cynthia Lorena. Aplicación de algoritmos de clasificación supervisada usando Weka. *Córdoba: Universidad Tecnológica Nacional, Facultad Regional Córdoba*, 2009.

RIVERA RESINA, Fernando Javier, et al. Aplicación de Busines Intelligence en una pequeña empresa mediante el uso de Power Bi. 2018.

MARTÍNEZ, Beatriz Beltrán. Minería de datos. *Cómo hallar una aguja en un pajar. Ingenierías*, 2001.

RIQUELME SANTOS, José Cristóbal; RUIZ, Roberto; GILBERT, Karina. Minería de datos: Conceptos y tendencias. *Inteligencia Artificial: Revista Iberoamericana de Inteligencia Artificial*, 2006.

Oded, M.; Lior, R. *Data Mining and Knowledge Discovery Handbook*. New York, 2010.

ZHANG, Yupei, et al. Educational data mining techniques for student performance prediction: method review and comparison analysis. *Frontiers in psychology*, 2021, vol. 12, p. 698490.

LÓPEZ, José Manuel Molina; HERRERO, Jesús García. Técnicas de análisis de datos. *Aplicaciones Prácticas utilizando Microsoft Excel y WEKA*, 2006.

COBO CANO, Miriam, et al. Modelo de previsión de demanda de transporte logístico de palés. 2021.

LOPERA ARANGO, Zuleima. Predicción de las materias primas que deben presupuestarse de acuerdo a las ventas de una compañía farmacéutica. 2022.

GALMÉS MIFSUD, Antoni. *Automatic forecasting y sus aplicaciones en Big Data: una comparativa entre algoritmos*. 2019. Tesis de Licenciatura. Universitat Politècnica de Catalunya.

SANTOSO, M. Hamdani. Application of Association Rule Method Using Apriori Algorithm to Find Sales Patterns Case Study of Indomaret Tanjung Anom. *Brilliance: Research of Artificial Intelligence*, 2021.

RIVAS, Jose Gabriel Rodriguez; CASTILLO, Sofía Rodríguez. Uso de Python para el análisis de datos aplicado en la investigación. *Investigación y Ciencia Aplicada a la Ingeniería*, 2022.

SANTANA SEPÚLVEDA, Sergio, et al. El arte de programar en R: un lenguaje para la estadística. 2014.

CONNOLLY, T. M. y BEGG, C. E., 2014. Database Systems: A Practical Approach to Design, Implementation, and Management. 6th. Pearson. ISBN 978-0-273-79364-4

GROUP, P. G. D., 2023. PostgreSQL: The world’s most advanced open source database. urlalso: https://www.postgresql.org/. Consultado: 2023-05-17

CORPORATION, O., 2023. MySQL. urlalso: https://www.mysql.com/. Consultado: 2023-05-17.

CONSORTIUM, S., 2023. SQLite. urlalso: https://www.sqlite.org/index.html. Consultado: 2023-05-17.

MARTÍN CALVO, Borja. Desarrollo de una aplicación para la exploración interactiva de datos neurocientíficos. 2016.

PAVLYSHENKO, Bohdan M. Machine-learning models for sales time series forecasting. *Data*, 2019.

BLANCO, HECTOR MIRETE; MARÍ, JORDI MUÑOZ; PEREZ-MUELAS, VALERO LAPARRA. Extracción y predicción de datos de series temporales de reservas de vuelo. 2019.