Pontificia Universidad Javeriana Cali
Facultad de Ingeniería y Ciencias
Maestría en Ciencia de Datos
Proyecto Aplicado

DESARROLLO DE MODELO PARA IDENTIFICACIÓN DE CARACTERÍSTICAS POSITIVAS/NEGATIVAS DE PRODUCTO EN COMENTARIOS EN PLATAFORMA E-COMMERCE USANDO APRENDIZAJE AUTOMÁTICO

Jhilbran Villa Ramos
Santiago Ibarra Enríquez

Director: Dr. Gloria Inés Álvarez
Codirector: Dr. Diego Linares

15/05/2025

FICHA RESUMEN

PROYECTO APLICADO – MAESTRÍA EN CIENCIA DE DATOS

TÍTULO: Desarrollo de modelo para identificación de características positivas/negativas de producto en comentarios en plataforma e-commerce usando aprendizaje automático

1. ÁREA DE TRABAJO

Ingeniería

2. TIPO DE PROYECTO (Aplicado, Innovación, Investigación)

Aplicado

3. ESTUDIANTE(S)

Jhilbran Alberto Villa Ramos
Santiago José Ibarra Enríquez

4. CORREO ELECTRÓNICO

jvilla@javerianacali.edu.co
sibarra92@javerianacali.edu.co

5. DIRECCIÓN Y TELÉFONO

Calle 116#42c-80, Barranquilla
Teléfono: 3023873982

Calle 28 #96-161, Quintas de Lili 2, Cali
Teléfono: 3164474312

6. DIRECTOR

Gloria Inés Álvarez Vargas

7. VINCULACIÓN DEL DIRECTOR

Profesor de planta

8. CODIRECTOR

Diego Luis Linares Ospina

8. CORREO ELECTRÓNICO DEL DIRECTOR

galvarez@javerianacali.edu.co

9. PALABRAS CLAVE

Machine learning, ciencia de datos, e-commerce, sistema de recomendación, análisis de datos

10. FECHA DE INICIO

6 de noviembre del 2023

11. FECHA DE FINALIZACIÓN

16 de julio del 2025

12. Resumen

Este proyecto busca desarrollar una herramienta de análisis de sentimientos automatizada para evaluar comentarios en plataformas de comercio electrónico mediante técnicas de Machine Learning. El objetivo principal es identificar características positivas y negativas en las reseñas de los usuarios, permitiendo a las empresas mejorar su reputación, abordar rápidamente comentarios negativos, optimizar productos y servicios, y diseñar estrategias de marketing más efectivas.

La metodología del proyecto se divide en dos etapas principales: preparación de datos e implementación del modelo. En la primera etapa, se realiza la adquisición de datos a partir de comentarios de usuarios, seguida de un proceso de limpieza y transformación del texto para eliminar ruido y normalizar los datos. Posteriormente, se aplican técnicas de incrustación de palabras como Word2Vec y GloVe, junto con métodos léxicos tradicionales (Bag-of-Words, TF-IDF, One-Hot Encoding) para convertir el texto en representaciones vectoriales adecuadas para el análisis.

En la fase de implementación, se entrenan y comparan distintos modelos de clasificación, utilizando los embeddings generados. Adicionalmente, se aplica modelado de temas (LDA) para identificar patrones en los comentarios. Finalmente, se generan visualizaciones interactivas que permiten una comprensión clara de los resultados.

Índice general

Introducción

Como comprador en el mercado e-commerce, es muy común verse atrapado por lo que es conocido como parálisis de elección, plataformas como Amazon ofrecen tantas opciones para el mismo producto o necesidad que se vuelve un ejercicio de futilidad el intentar identificar cual es el mejor, o cual es el que más se ajusta a las necesidades particulares del caso basándose solo en las descripciones del producto, que al ser diseñadas por el vendedor, solo destacan los beneficios reales o percibidos del producto ignorando las limitaciones o falencias que podría tener.

Los comentarios que dejan los clientes son una excelente fuente de contexto e información sobre el rendimiento y calidad del producto en la vida real, pero para hacerse a una idea de la experiencia general, se necesita leer un número considerable de comentarios en un ya de por si considerable número de opciones es por eso que este proyecto quiere desarrollar un modelo de Machine Learning que haga esto por el usuario.

El objetivo principal de este proyecto es aprovechar las capacidades del aprendizaje automático para procesar grandes cantidades de comentarios en idioma inglés eficientemente, para extraer el contenido importante que ayude a compradores y vendedores a tomar decisiones informadas en la compra y mejora del producto respectivamente.

En este proyecto exploraremos técnicas avanzadas de redes neuronales, Procesamiento de Lenguaje Natural (NLP), análisis de sentimiento, y modelado para identificar patrones y tendencias dentro de los bloques de texto de los comentarios. La aplicación de algoritmos de aprendizaje automático permitirá la clasificación eficaz de opiniones, diferenciando entre aspectos positivos y negativos de manera precisa.

La contribución de este modelo es la capacidad de ver un resumen de un gran número de comentarios segmentando que es lo positivo y negativo que se menciona en ellos, esto tiene el potencial de transformar la forma en la que los clientes toman sus decisiones de compra, y la forma en la que los fabricantes y vendedores gestionan el desarrollo de nuevos productos.

Definición del problema

2.1. Planteamiento del problema

En el actual panorama del comercio electrónico, el acceso fácil a internet ha transformado radicalmente las dinámicas de compra de los usuarios. Esta realidad es fértil para la ciencia de datos, crucial en la interpretación de patrones de consumo y preferencias [10]. La capacidad de analizar da- tos de compras en línea es esencial para comprender las tendencias emergentes en el comportamiento del consumidor.

Desde la perspectiva de los fabricantes, la ciencia de datos ofrece herramientas innovadoras para descifrar las necesidades y opiniones de los usuarios. La minería de texto y el análisis de sentimientos aplicados a comentarios y calificaciones en línea pueden revelar insights valiosos. destacan cómo el procesamiento de lenguaje natural ha revolucionado la forma en que los fabricantes interpretan los comentarios de los usuarios, permitiéndoles identificar no solo las calificaciones numéricas, sino también las opiniones y sugerencias específicas.

Sin embargo, este potencial de la ciencia de datos aún no se aprovecha plenamente. Los métodos tradicionales como encuestas y grupos focales continúan siendo la norma para muchos fabricantes, a pesar de sus limitaciones en cuanto a costos y alcance. Del mismo modo, los usuarios a menudo se basan en procesos manuales para evaluar productos, revisando calificaciones y comentarios sin el beneficio de herramientas analíticas avanzadas.

Este escenario plantea un problema significativo: existe una brecha entre la riqueza de datos generados en el e-commerce y la capacidad de utilizar estos datos de manera efectiva tanto por parte de los fabricantes como de los usuarios. La solución radica en la adopción más amplia de prácticas de ciencia de datos que puedan traducir grandes volúmenes de información en acciones y decisiones informadas, beneficiando así tanto a los consumidores como a los productores en el mercado digital.

2.2. Formulación del problema

¿De qué manera se puede diseñar y desarrollar una herramienta de análisis de texto basada en aprendizaje automático que identifique y clasifique eficazmente las ideas positivas y negativas en textos, determinando además los temas específicos a los que se refieren?

Objetivos

3.1. Objetivo general

Desarrollar una herramienta para identificar características positivas y negativas en el texto de comentarios de plataforma e-commerce utilizando modelos de Machine Learning.

3.2. Objetivos específicos

- Obtener y almacenar datos de comentarios de plataformas e-commerce para el desarrollo y entrenamiento de modelos de análisis de sentimientos en Machine Learning.
- Desarrollar y validar un modelo de Machine Learning que diferencie con precisión los comentarios positivos de los negativos, aplicando técnicas avanzadas de análisis de sentimiento.
- Implementar métodos de extracción de texto para identificar y resumir características del producto mencionadas en los comentarios, destacando los aspectos más gustados o disgustados.
- Establecer y aplicar métricas de evaluación robustas para determinar la precisión y eficacia de los modelos desarrollados en la clasificación de sentimientos y extracción de características.
- Diseñar y desarrollar una interfaz de usuario intuitiva que visualice las características más apreciadas y criticadas de los productos, basada en el análisis de comentarios.

Marco de referencia

4.1. Marco teórico

A continuación, se describen los temas y campos relacionados con el proyecto, debido a la naturaleza del proyecto, la mayoría de los referentes teóricos giran en torno al tratamiento de texto, adicionalmente se provee una breve explicación de los referentes y técnicas en cuestión.

4.1.0.1. Web scrapping

El web scrapping es una técnica utilizada para extraer datos de sitios web de manera automatizada. Este proceso implica el uso de software especializado para acceder a páginas web y recolectar información específica de ellas, como textos, imágenes, listas de precios y otros datos relevantes. Los scrapers, como se conocen a estos programas, navegan por la web de manera similar a un usuario humano, pero con la capacidad de procesar grandes cantidades de información en un tiempo reducido. Esta técnica se ha vuelto cada vez más popular en diversas áreas, como el análisis de mercado, investigación académica, monitoreo de precios, y recolección de datos para entrenamiento de modelos de inteligencia artificial. A pesar de su utilidad, el web scrapping debe realizarse con precaución para no infringir leyes de propiedad intelectual o términos de uso de sitios web, como discuten Krotov, Johnson y Silva [18], quienes analizan las implicaciones legales y los desafíos éticos del web scrapping.

El web scrapping no solo se trata de recolectar datos, sino también de transformarlos en un formato estructurado y utilizable. Las técnicas avanzadas de web scrapping incluyen el procesamiento de lenguaje natural y el aprendizaje automático para interpretar y organizar la información extraída. Estas herramientas permiten a los usuarios no solo recolectar datos sino también analizarlos y obtener información valiosa. Sin embargo, como señala Foerderer [12], el web scrapping plantea desafíos en cuanto a la calidad y fiabilidad de los datos, ya que la información en la web puede ser no verificada, estar desactualizada o de no planearse correctamente, presentar sesgos. Por tanto, es crucial implementar procedimientos de verificación y validación de datos para garantizar la precisión y utilidad de la información recolectada.

4.1.0.2. Análisis de sentimientos

El análisis de sentimientos es una disciplina interdisciplinaria que combina técnicas de ciencia de datos e inteligencia artificial para interpretar y clasificar las emociones expresadas en textos. Se centra en la evaluación de comentarios en redes sociales, reseñas de productos, y otros tipos

de comunicación escrita. Mediante el uso de algoritmos avanzados de procesamiento de lenguaje natural, esta técnica puede identificar y categorizar opiniones como positivas, negativas o neutras. Esta área ha cobrado importancia en la era digital, desempeñando un papel crucial en la comprensión de la percepción de la marca, el análisis de retroalimentación de clientes y el monitoreo de tendencias en redes sociales. Además, se aplica en la detección de patrones de opinión pública y en la toma de decisiones empresariales basadas en datos. Los trabajos de Liu [19] y Pang y Lee [24] han sido fundamentales en el desarrollo de metodologías y aplicaciones prácticas en este campo. Estas investigaciones no solo han proporcionado un marco teórico sólido, sino que también han impulsado innovaciones en la recopilación y análisis de grandes volúmenes de datos textuales, facilitando una comprensión más profunda de las dinámicas humanas y sociales a través del lente del lenguaje.

4.1.0.3. Clasificación de textos y categorización de documentos

La clasificación de textos y categorización de documentos son tareas fundamentales en el campo del procesamiento de lenguaje natural (PLN) y la recuperación de información. Estas técnicas implican la asignación automática de etiquetas o categorías a textos o documentos basándose en su contenido. El proceso utiliza algoritmos de aprendizaje automático y estadísticos para analizar y clasificar documentos en categorías predefinidas, facilitando la organización, búsqueda y filtrado de grandes volúmenes de datos. Aplicaciones notables incluyen el filtrado de correos electrónicos, clasificación de noticias, y organización de documentos legales y médicos. La efectividad de estas técnicas ha sido mejorada significativamente con la introducción de métodos de aprendizaje profundo, los cuales permiten capturar características complejas y patrones sutiles en los textos. Trabajos pioneros en este campo, como los de Sebastiani [33] y Aggarwal y Zhai [4], han proporcionado una comprensión detallada de los métodos y desafíos asociados con la clasificación de textos y la categorización de documentos. Estas investigaciones han ayudado a establecer las bases teóricas y prácticas para el desarrollo de sistemas más avanzados y precisos, capaces de manejar la diversidad y complejidad del lenguaje humano en diferentes contextos y aplicaciones.

4.1.0.4. Aprendizaje automático (machine learning)

El aprendizaje automático, un componente crucial de la inteligencia artificial, se centra en desarrollar algoritmos que permitan a las máquinas aprender y mejorar a partir de datos y experiencias [22]. Ha experimentado un crecimiento significativo, pasando de simples perceptrones a complejas redes neuronales, y es fundamental en campos tan diversos como la medicina, las finanzas y la robótica [5].

Esta disciplina se subdivide en aprendizaje supervisado, no supervisado, semi-supervisado y aprendizaje por refuerzo. Cada uno con metodologías y aplicaciones específicas, adaptándose según la naturaleza de los datos y el problema específico a resolver [13].

4.1.0.4.1. Modelos supervisados

Support vector machine (SVM)) son modelos de aprendizaje supervisado utilizados para la clasificación y regresión. Su objetivo es encontrar el hiperplano óptimo que mejor separe las clases de datos en el espacio de características. Las SVM son efectivas incluso en conjuntos de

datos con alta dimensionalidad y se benefician de funciones de kernel que pueden mapear datos en espacios no lineales. Estas son ampliamente utilizadas en aplicaciones como el reconocimiento de escritura a mano y la detección de enfermedades médicas [7].

Random Forest es un modelo de ensamble utilizado en el aprendizaje super- visado que combina múltiples árboles de decisión para mejorar la precisión y reducir el sobreajuste. Cada árbol en el conjunto se construye de manera independiente, y las predicciones finales se obtienen promediando las predicciones de todos los árboles (en problemas de clasificación) o tomando el promedio ponderado (en problemas de regresión). Random Forest es altamente eficaz en una varie- dad de aplicaciones, incluyendo la clasificación de enfermedades médicas y la detección de fraudes financieros [5].

4.1.0.5. Métricas de Evaluación

Son fundamentales para medir el rendimiento de los modelos supervisados y no supervisados. Algunas de las métricas clave incluyen:

- Exactitud (Accuracy): Mide la proporción de predicciones correctas en relación con el total de predicciones. Es adecuada cuando las clases están equilibradas.
- Precisión (Precision): Calcula la proporción de verdaderos positivos entre todos los positivos predichos. Es útil cuando el costo de los falsos positivos es alto.
- Sensibilidad (Recall): Calcula la proporción de verdaderos positivos entre todos los casos positivos reales. Es importante cuando se deben evitar falsos negativos.
- Valor F1 (F1 Score): Combina precisión y sensibilidad en una sola métrica, lo que es útil cuando se busca un equilibrio entre ambas.

4.1.0.6. Métodos de representación

4.1.0.6.1. TF-IDF (Term Frequency-Inverse Document Frequency) es una técnica fundamental en el procesamiento de texto y la recuperación de información. Su aplicación abarca diversas áreas, desde motores de búsqueda hasta sistemas de recomendación y análisis de texto. En un motor de búsqueda, TF-IDF ayuda a de- terminar la relevancia de un documento para una consulta específica al resaltar los términos más importantes y distintivos. Esto se logra calculando la frecuencia del término en el documento (TF) y
la rareza del término en toda la colección de documentos (IDF), y luego combinando estas métricas para calcular una puntuación de importancia relativa [34].

En la clasificación de documentos, TF-IDF desempeña un papel crucial al identificar las palabras clave y las características más importantes que ayudan a categorizar los documentos en diferentes clases o categorías. Esto se logra mediante el cálculo de las puntuaciones TF-IDF para cada término en los documentos y luego utilizando estas puntuaciones como características para el modelo de clasificación [32]. Además, en el análisis de texto, TF-IDF se utiliza para extraer información relevante y resumir grandes cantidades de texto de manera automatizada. Esto puede incluir la identificación de temas clave, la detección de tendencias o la extracción de información específica de los documentos [34].

En resumen, TF-IDF es una técnica poderosa y versátil que se utiliza en una amplia variedad de aplicaciones relacionadas con el procesamiento de texto y la recuperación de información. Su capacidad para evaluar la importancia de los términos en un documento en relación con una colección más amplia de documentos la convierte en una herramienta invaluable para analizar y comprender grandes conjuntos de datos de texto [32].

4.1.0.6.2. Bag of words es una técnica clásica de representación de documentos en el procesamiento de lenguaje natural, basada en la frecuencia de ocurrencia de palabras en un documento sin considerar el orden en que aparecen. Esta representación simplificada es ampliamente utilizada en tareas como la clasificación de textos y el análisis de sentimientos [31]. En BoW, cada documento se representa como un vector donde cada elemento corresponde a una palabra del vocabulario y el valor del elemento indica la frecuencia de esa palabra en el documento. Esta representación permite comparar documentos mediante la distancia entre los vectores de términos, como la distancia del coseno, que mide la similitud entre documentos en función de la
orientación de sus vectores en el espacio vectorial [31].

A pesar de su simplicidad, BoW ha demostrado ser efectivo en una variedad de tareas de procesamiento de lenguaje natural. Sin embargo, una limitación importante es que no captura el significado semántico de las palabras ni considera la estructura gramatical de los textos.

La técnica de Bag of Words se ha utilizado en una amplia gama de aplicaciones, desde motores de búsqueda hasta sistemas de recomendación y análisis de sentimientos. En los motores de búsqueda, por ejemplo, BoW se utiliza para indexar y recuperar documentos relevantes en función de las palabras clave proporcionadas por el usuario.

En resumen, Bag of Words es una técnica simple pero poderosa en el procesamiento de lenguaje natural que ha demostrado ser útil en una variedad de aplicaciones. Su simplicidad y eficacia la convierten en una herramienta fundamental en el arsenal de técnicas de análisis de texto [31].

4.1.0.6.3. Word2Vec es una técnica avanzada de representación de palabras en el procesamiento de lenguaje natural, que utiliza modelos de redes neuronales para capturar relaciones semánticas entre palabras en un corpus de texto[21]. Esta técnica se ha convertido en una herramienta fundamental para muchas aplicaciones de NLP, incluyendo la clasificación de textos, la traducción automática y el análisis de sentimientos.

En Word2Vec, las palabras se representan como vectores densos en un espacio dimensional, de manera que palabras con significados similares tienen vectores cercanos entre sí [21]. Esto permite capturar relaciones semánticas y analogías entre palabras, como rey - hombre + mujer = reina", mediante operaciones vectoriales simples en el espacio vectorial de palabras.

La técnica de Word2Vec se ha utilizado con éxito en una amplia variedad de aplicaciones de procesamiento de lenguaje natural, debido a su capacidad para generar representaciones de palabras que capturan de manera efectiva su significado semántico [21]. Por ejemplo, en la traducción automática, las representaciones de palabras aprendidas por Word2Vec pueden mejorar la calidad de las traducciones al capturar mejor el contexto y el significado de las palabras.

En resumen, Word2Vec es una técnica poderosa y versátil en el procesamiento de lenguaje natural

que ha demostrado ser efectiva en una amplia gama de aplicaciones. Su capacidad para capturar relaciones semánticas entre palabras la convierte en una herramienta invaluable para comprender y procesar el lenguaje humano de manera automatizada.

4.1.0.6.4. GloVe (Global Vectors for Word Representation) es una técnica avanzada de representación de palabras en el procesamiento de lenguaje natural, que utiliza un enfoque basado en matrices de coocurrencia para capturar las relaciones semánticas entre palabras en un corpus de texto [26]. Esta técnica se ha convertido en una herramienta fundamental para muchas aplicaciones de NLP, incluyendo la clasificación de textos, la traducción automática y el análisis de sentimientos.

En GloVe, las palabras se representan como vectores en un espacio dimensional, de manera que la similitud entre vectores captura la relación de co-ocurrencia entre palabras en el corpus de texto [26]. Esto permite capturar relaciones semánticas complejas y analogías entre palabras, similar a como lo hace Word2Vec.

La técnica de GloVe se ha utilizado con éxito en una amplia variedad de aplicaciones de procesamiento de lenguaje natural, debido a su capacidad para generar representaciones de palabras que capturan de manera efectiva su significado semántico [26]. Por ejemplo, en la traducción automática, las representaciones de palabras aprendidas por GloVe pueden mejorar la calidad de las traducciones al capturar mejor el contexto y el significado de las palabras.

En resumen, GloVe es una técnica poderosa y versátil en el procesamiento de lenguaje natural que ha demostrado ser efectiva en una amplia gama de aplicaciones. Su capacidad para capturar relaciones semánticas entre palabras la convierte en una herramienta invaluable para comprender y procesar el lenguaje humano de manera automatizada [26].

4.1.0.7. Lematización

La lematización es una técnica de procesamiento de lenguaje natural que busca reducir las palabras a su forma base o lema, con el objetivo de simplificar el análisis y mejorar la precisión en tareas como la búsqueda de información, la recuperación de información y el análisis de sentimientos [17]. A diferencia del stemming, que simplemente elimina los sufijos de las palabras para obtener su raíz, la lematización tiene en cuenta la morfología y el contexto de las palabras para determinar su forma base.

En la lematización, las palabras se transforman en su lema correspondiente, que es una forma canónica o base de la palabra que representa su significado básico en el idioma. Por ejemplo, los verbos en diferentes formas conjugadas se lematizan a su forma infinitiva, y los sustantivos se lematizan a su forma singular.

La lematización es una técnica importante en el procesamiento de lenguaje natural debido a su capacidad para normalizar las palabras y reducir la variabilidad léxica en un corpus de texto [17]. Esto facilita la comparación y el análisis de documentos al agrupar palabras con significados similares.

En resumen, la lematización es una técnica valiosa en el procesamiento de lenguaje natural que se utiliza para normalizar las palabras y reducir la variabilidad léxica en un corpus de texto. Su capacidad para transformar palabras a su forma base mejora la precisión y la eficacia en una variedad de aplicaciones de NLP.

4.1.0.8. Modelado de temas

4.1.0.8.1. LDA (Latent Dirichlet Allocation) Esta es una de las técnicas que viene adquiriendo fuerza en el campo de los modelos de tópicos. Esta técnica está compuesta por conceptos de Modelos Bayesianos y se basa en el proceso probabilístico genérico que permite inferir tópicos de un documento en base a una distribución a posteriori obtenida por el LDA. Los modelos bayesianos pueden servir para indicar cómo debemos modificar nuestras probabilidades subjetivas cuando recibimos información adicional de un experimento. La estadística bayesiana está demostrando su utilidad en ciertas estimaciones basadas en el conocimiento subjetivo a priori y el hecho de permitir revisar esas estimaciones en función de la evidencia empírica. Esto último es lo que está abriendo nuevas formas de hacer conocimiento aplicado a la teoría de la información. Una aplicación de esto son los clasificadores bayesianos que son frecuentemente usados en implementaciones de filtros de correo basura o spam, que se adaptan con el uso.

4.1.0.8.2. Perplejidad Esta métrica evalúa la capacidad de un modelo de tópicos para predecir un conjunto de prueba después de haber sido entrenado en un conjunto de entrenamiento. Se calcula dividiendo un conjunto de datos en dos partes: un conjunto de entrenamiento y un conjunto de prueba. La idea es entrenar un modelo de tema utilizando el conjunto de entrenamiento y luego probar el modelo en un conjunto de prueba que contiene documentos no vistos anteriormente. [29]

4.1.0.8.3. Shiny Shiny para Python es un potente framework de aplicaciones web para crear visualizaciones de datos interactivas, paneles y aplicaciones con Python.

Shiny para Python consta de dos componentes principales: la interfaz de usuario (IU) y el servidor. El componente de IU define el diseño y la apariencia de la aplicación, mientras que el componente de servidor gestiona la lógica y el procesamiento de datos. Esta separación de tareas simplifica el desarrollo y el mantenimiento de aplicaciones complejas, haciéndolo más eficiente que usar Jupyter Notebooks o Streamlit. [30]

4.1.0.8.4. Lime (Local interpretable model-agnostic explanations) Es una herramienta capaz de aproximar un modelo con otro que sea interpretable para evaluar cuáles atributos de entrada tienen mayor relevancia en la predicción.

Para asegurar tanto la **interpretabilidad** como la **fidelidad local**, el objetivo del algoritmo es minimizar L (función de pérdida) sin que Ω (medida de complejidad del modelo explicativo) sea demasiado grande. La formulación es la siguiente:

$$\xi(x) = \underset{g \in G}{\arg\min}\, L(f, g, \pi_x) + \Omega(g) \tag{4.1}$$

Procedimiento del Algoritmo

El algoritmo consta de los siguientes pasos (visualizados en la Figura 4):

1. **Crear perturbaciones** (z') sobre los datos de entrada originales (x).
2. **Predecir salidas** para los datos perturbados ($f(z')$).
3. **Calcular distancias** entre los datos perturbados y el punto de observación (x), generando una medida de infidelidad local (π_x).
4. **Seleccionar las** m **características** más relevantes según los coeficientes de infidelidad.
5. **Entrenar un modelo interpretable** (ej. regresión lineal ponderada) usando las perturbaciones y los pesos π_x.
6. **Extraer explicaciones**: Los coeficientes del modelo interpretable (g) son las explicaciones para el punto x.[8]

4.1.0.8.5. Coherencia Se llama a un tema coherente, si las palabras principales tienen sentido juntas. Un ejemplo de un tema coherente es juego, deporte, pelota, equipo, mientras que un tema incoherente podría ser juego, deporte, pelota, pingüino [29]. Una de las deficiencias de la perplejidad es que no capta el contexto, es decir, la perplejidad no captura la relación entre las palabras de un tema o los temas de un documento. Para superar esto, se han desarrollado enfoques que intentan capturar el contexto entre palabras en un tema. Utilizan medidas como la probabilidad condicional de la coocurrencia de palabras en un tópico. Estos enfoques se denominan colectivamente coherencia". La puntuación de coherencia se usa en el modelado de tópicos para medir qué tan interpretables son los tópicos para los humanos. En este caso, los tópicos se representan como las primeras N palabras con mayor probabilidad de pertenecer a ese tópico en particular. Brevemente, el puntaje de coherencia mide qué tan similares son estas palabras entre sí y se dice que un tema tiene alta coherencia si las palabras que lo definen tienen una alta probabilidad de aparecer juntas (coocurrir) en todos los documentos.

4.2. Antecedentes

4.2.0.1. Análisis de sentimientos de reseñas para determinar la acogida de un producto utilizando técnicas de machine learning y data mining.

El proyecto anteriormente mencionado, se relaciona con nuestra propuesta porque busca solucionar la misma problemática e implementar una herramienta que analice todas las reseñas de un producto y determine su polaridad, esto mediante el diseño e implementación de modelos de machine learning, técnicas de minería de datos, web scrapping y herramientas de visualización de datos a través de Python. [10]

4.2.0.2. Sistema de recomendaciones utilizando técnicas de Machine Learning para una plataforma de e-commerce perteneciente a la empresa LCC Opentech, C.A.

El proyecto se relaciona con nuestra propuesta, dado que a través de Amazon Web Services mediante web scrapping y un modelo de desarrollo incremental, busca generar un sistema de recomendaciones mediante modelos de machine learning. [6]

4.2.0.3. Estudio e implementación de un modelo de procesamiento de lenguaje natural que analice la satisfacción de compra mediante el análisis de comentarios de usuarios.

El proyecto mencionado se diferencia de nuestra propuesta, dado que, para realizar un sistema de recomendación basado en la opinión de los clientes, implementa un modelo de procesamiento de lenguaje natural llamado BERT, ya que el modelo parte del desarrollo del proyecto de investigación y es un modelo especialmente creado para la empresa, como parte interesada en el proyecto. [20]

4.2.0.4. Valoración cuantitativa de productos a través de procesamiento de lenguaje natural (NLP).

El trabajo se relaciona con nuestra propuesta, pues busca realizar una estimación cuantitativa al servicio a través de la categorización automática del sentimiento presente en un comentario hacia el producto, pero se diferencia pues su desarrollo es mediante una aplicación basada en Deep Learning. [23]

Capítulo 5

Preparación de datos

5.1. Adquisición de datos

El modelo fue desarrollado en Python utilizando Jupyter Notebook. El código completo puede consultarse en:
GitHub - Modelo Final

Durante la estrategia y planificación del scrapping se identificó que el web site de Amazon tiene varias medidas para dificultar y en la mayoría de los casos prevenir el scrapping, como un diseño complejo de la estructura de la página para dificultar la adquisición de la información, bloqueo de bots e IP, y otros métodos. Esto hace que la adquisición de la información a través de este método se vuelva un método más complicado de lo que amerita el proyecto, probablemente podría justificar ser un proyecto propio en el futuro si se requiere adquirir la información de manera regular. Con esto en mente, y la aprobación de los directores del proyecto, se procedió a buscar una fuente alternativa, luego de evaluar distintas opciones, se determinó que uno de los miembros del equipo podía usar información de web-scraping publica usada en su trabajo, esto nos dio como resultado un archivo .csv con aproximadamente 22 millones de registros de reviews de Amazon USA para productos de diferentes categorías. Con esta información se procederá a realizar el preprocesamiento

5.2. Limpieza y transformación

Luego de analizar la estructura existente de datos, se identifican los tipos de datos de cada columna y cuáles son las columnas vitales para el desarrollo de los modelos de Machine Learning. Los tipos de datos para cada columna se especifica a continuación, al igual que una muestra de las primeras cinco filas

RUNDATE	INTERNETNR	ITEMBRAND	ITEMDESCRIPTION
2023-07-26T11:38:32Z	B0B2JNQFYP	XIHUAN	XIHUAN Adjustable Over The Toilet Storage...
2023-07-04T19:03:42Z	B082L3RZV9	Gorilla Grip	Gorilla Grip Easy to Use Knife Sharpener...
2023-07-26T13:05:25Z	B08TZTSKSZ	Amolliar	2-Tier Lazy Susan Turntable Food Storage...
2023-06-30T08:31:44Z	B01MU0SP2W	Simple Houseware	Simple Houseware Dual Bar Adjustable Garment...
2023-11-27T18:39:17Z	B002D8NREI	Case	CASE XX WR Pocket Knife Amber Jigged Bone...

Cuadro 5.2: Primera parte - columnas de la tabla [3]

Nombre de Columna	Tipo de Dato
Unnamed: 0	int64
RUNDATE	object
INTERNETNR	object
ITEMBRAND	object
ITEMDESCRIPTION	object
ITEMCATEGORY	object
REVIEWID	object
REVIEW_TITLE	object
REVIEW_TEXT	object
RATING	float64
REVIEW_DATE	object
PRODUCTRATING	float64
PRODUCTREVIEWCOUNT	float64

Cuadro 5.1: Tipos de datos de las columnas en el DataFrame [1]

ITEMCATEGORY	REVIEWID	REVIEW_TITLE
Home & Kitchen >Furniture >...	R3M0P40QCSZX0Y	5.0 out of 5 stars SMALL SPACE GENIOUS
Home & Kitchen >Kitchen & Dining...	R36SK7YPD5F9VE	5.0 out of 5 stars Great value
Home & Kitchen >Kitchen & Dining...	R2UHMNDUR94SZL	5.0 out of 5 stars Functional and beautiful
Home & Kitchen >Home Storage & Organization...	RT9X80OUSKVJS	1.0 out of 5 stars Yikes
Home & Kitchen >Kitchen & Dining...	R3JKAK2HJI0OUE	Case should be the only knife...

Cuadro 5.3: Segunda parte - columnas de la tabla [3]

REVIEW_TEXT	RATING	REVIEW_DATE
As you can see from the picture I have a very small space...	5.0	October 24, 2022
For the price I was skeptical! Really nice value...	5.0	July 2, 2023
2 fit perfectly side by side. Measurements were accurate	5.0	March 21, 2022
Product arrived cracked, scuffed, and incomplete. Missing tool...	1.0	January 4, 2020
Still in the box for when, if ever, I lose my other...	5.0	April 16, 2015

Cuadro 5.4: Tercera parte - columnas de la tabla [3]

PRODUCTRATING	PRODUCTREVIEWCOUNT
4.300000	43.000000
4.500000	1074.000000
4.700000	185.000000
4.400000	1336.000000
4.700000	82.000000

Cuadro 5.5: Cuarta parte - columnas de la tabla [3]

Como se puede ver, el set de datos cuenta con la fecha en la que se hizo el scraping, el número del producto, marca, descripción, la categoría en donde se ubica dentro de la taxonomía del sitio de Amazon, ID del review, título del review, texto del review, calificación de ese review en particular, fecha de review, calificación general del producto y el conteo de reviews.

Como comentario particular, se crearon las funciones y el código general en inglés para facilitar la búsqueda de documentación y referencias.

Para el preprocesamiento lo primero que se hace es remover la puntuacion de los campos con texto, para esto se usan expresiones regulares para remover todo lo que sea diferente letras.

Luego de haber removido puntuaciones, se identifica el idioma para conservar solo reviews en inglés, para esto se usa el paquete langid, el resultado de este paquete es una cadena de texto identificando el idioma del texto suministrado, lo que permite automáticamente segmentar el set de datos.

Usando este nuevo campo, se filtra el data set a solo valores que coincidan con el idioma *en*

Posteriormente se eliminan las columnas no relevantes, esto se hace para optimizar tiempos de carga y espacio ocupado por los archivos y modelos.

Como último paso de limpieza independiente se remueven filas con NA o en blanco.

Luego de haber hecho la limpieza independiente, se crea una función para procesar el texto, esta función realizara varias acciones como remover acentos (una acción diferente a puntuacion), limpiar el texto utilizando la librería *cleantext*, y aplicar lematización y tokenización al texto, el resultado es una lista con el texto ya tokenizado y listo para ser usado en representaciones gráficas y modelos.

Luego de haber definido la función de procesamiento, se aplica a las dos columnas que tienen el texto de los reviews que serán usadas en los modelos de Machine Learning.

Como resultado de la limpieza y procesamiento el set de datos final tiene solamente el texto tokenizado y lematizado, se conservan las columnas de categoría y marca para propósitos de visualización una vez los modelos se hayan creado

INTERNETNR	ITEMBRAND	ITEMDESCRIPTION	ITEMCATEGORY
B0B2JNQFYP	XIHUAN	XIHUAN Adjustable Over The Toilet Storage...	Home & Kitchen >Furniture >Bathroom...
B082L3RZV9	Gorilla Grip	Gorilla Grip Easy to Use Knife...	Home & Kitchen >Kitchen & Dining >Cutlery...
B08TZTSKSZ	Amolliar	2-Tier Lazy Susan Turntable Food Storage Container for...	Home & Kitchen >Kitchen & Dining >Storage...

Cuadro 5.6: Primera parte - columnas de la tabla [2]

REVIEWID	RATING	REVIEW_DATE	PROCESSED_REVIEW_TITLE
R3M0P40QCSZX0Y	5	24-Oct-22	['50', '5', 'star', 'small', 'space', 'genious']
R36SK7YPD5F9VE	5	2-Jul-23	['50', '5', 'star', 'great', 'value']
R2UHMNDUR94SZL	5	21-Mar-22	['50', '5', 'star', 'functional', 'beautiful']

Cuadro 5.7: Segunda parte - columnas de la tabla procesada [2]

PROCESSED_REVIEW_TEXT	CLEAN_TITLE	CLEAN_TEXT
['see', 'picture', 'small', 'space', 'work', 'next'...]	50 5 star small space genious	see picture small space work next sink...
['price', 'skeptical', 'really', 'nice', 'value', 'knife'...]	50 5 star great value	price skeptical really nice value knife..
['2', 'fit', 'perfectly', 'side', 'side', 'measurement'...]	50 5 star functional beautiful	2 fit perfectly side side measurement...

Cuadro 5.8: Tercera parte - columnas de la tabla [2]

5.3. Incrustación de palabras

Para los word embeddings se utilizarán dos métodos similares *Word2Vec* y *GloVe* ambas generan Word Embeddings que buscan representar palabras como vectores densos en un espacio semántico. Aunque ambas comparten el mismo objetivo, se distinguen por los enfoques metodológicos que emplean, lo que las hace adecuadas para distintos tipos de tareas. En este ejercicio, se analizarán los resultados, ventajas y diferencias entre ellas.

5.3.1. Word2Vec

Word2Vec utiliza un modelo basado en predicción, fundamentado en redes neuronales simples que optimizan la probabilidad de coocurrencia de palabras dentro de ventanas contextuales pequeñas. Este enfoque local permite capturar relaciones semánticas y sintácticas específicas. Word2Vec, en particular la implementación en la librería GenSim, ofrece dos variantes principales: CBOW, que predice palabras objetivo en función de su contexto, y Skip-Gram, que hace lo contrario, prediciendo el contexto a partir de una palabra objetivo. [28]

Objetivos

- Transformar datos textuales (títulos y reseñas tokenizadas) en representaciones numéricas (embeddings) que conserven relaciones semánticas y contextuales.

- Entrenar un modelo de Word2Vec utilizando datos combinados y procesados.
- Generar representaciones vectoriales para palabras individuales y calcular embeddings para documentos completos.

Metodologia

- Preprocesamiento Los datos ya fueron limpiados y tokenizados en el capitulo anterior, por lo que para preprocesamiento solo es necesario combinarlos y darles el formato apropiado en un único corpus para capturar contextos más amplios y asegurar un entrenamiento robusto.
- Entrenamiento del modelo Word2Vec. Se entrena un modelo Word2Vec utilizando el corpus procesado. Los hiperparámetros principales incluyen:
  - vector_size=100: Dimensión de los embeddings.
  - window=5: Contexto de palabras considerado.
  - min_count=1: Incluye palabras con al menos una aparición.
  - sg=0: Usa el algoritmo CBOW (Continuous Bag of Words).
- Generación de Representaciones Vectoriales. Se obtienen vectores para cada palabra en el vocabulario del modelo, creando un diccionario de palabras y sus correspondientes vectores.

−0,38794565	0,43093875	0,1044347	−0,03574588	0,09490608
−0,70818	0,18914822	1,1371714	−0,5752987	−0,15819639
−0,40992185	−0,81002396	−0,08287459	0,12496298	0,14532234
−0,5918789	0,15939236	−0,49257863	−0,12513205	−1,0506058
0,32277915	0,11369831	0,37866786	−0,19374721	−0,19765821
0,0296234	−0,33639604	−0,18357335	−0,51791406	0,10661881
0,70692974	0,02869034	0,06870899	−0,4708833	−0,35015854
0,52313024	0,1132571	−0,26727363	−0,3499108	−1,026991
0,12840998	−0,5076066	−0,36548296	0,11301742	0,7189506
−0,24900918	−0,3211715	−0,0634724	0,30392966	0,13309483
0,295242	−0,5331074	−0,06973968	−0,29103222	−0,46856555
0,19660167	0,21095398	−0,19077992	−0,56640786	0,03590554
0,22810225	0,14185154	0,07004191	−0,06294721	−0,8436645
0,6322844	0,21706955	0,63304424	−0,8305795	0,7579124
−0,44688728	0,22548541	0,5670792	−0,0901356	0,5816956
0,31866238	0,1870061	−0,04607819	−0,40579507	0,11739336
−0,31657147	−0,13379179	−0,66494447	0,666363	−0,2529219
−0,19041888	0,05098311	0,5731687	0,5810394	−0,03894466
0,54467076	0,27428904	0,21583045	0,08438003	0,86571735
0,5328432	0,3736023	−0,59851605	0,329959	−0,16656098

Cálculo de Embeddings para Documentos Para representar documentos (reseñas completas), se implementa una función que calcula la media de los vectores de las palabras presentes en el documento. Si ninguna palabra está en el vocabulario, se asigna un vector nulo.

Resultados esperados

- Vectores de dimensión 100 que capturan la semántica y el contexto de cada palabra en el vocabulario.
- Representaciones vectoriales promedio para reseñas completas basado en la representación vectorial
- Modelo Word2Vec que encapsula la semántica del conjunto de datos, con aplicaciones en la detección de similitudes entre palabras o la predicción de contextos.

INTERNETNR	ITEMBRAND	...	CLEAN_TEXT	w2v
B0B2JNQFYP	XIHUAN	...	see picture small space work...	[-0.073, 0.091, 0.026, 0.004, ...]
B082L3RZV9	Gorilla Grip	...	price skeptical really nice...	[-0.104, 0.125, 0.040, 0.002, ...]
B08TZTSKSZ	Amolliar	...	2 fit perfectly side side...	[-0.289, 0.351, 0.105, 0.004, ...]
B01MU0SP2W	Simple Houseware	...	product arrive cracked scuffed...	[-0.067, 0.088, 0.024, 0.005, ...]
B002D8NREI	Case	...	still box ever lose...	[-0.022, 0.035, 0.008, 0.004, ...]

Ventajas

- Es computacionalmente eficiente, ya que no requiere construir ni almacenar una matriz completa de coocurrencia. Esto lo hace adecuado para entrenar en corpus moderados con hardware limitado como el que se estará utilizando para realizar el ejercicio.
- es capaz de capturar relaciones semánticas y sintácticas en el contexto inmediato. Esto le permite representar adecuadamente el significado de palabras en oraciones cortas y específicas como reviews de productos.
- Actualizaciones dinámicas que permiten actualizar y reentrenar fácilmente para incluir nuevas palabras, sin necesidad de recalcular todo desde cero. Esto permitirá realizar pruebas con un data set más pequeño para ahorrar recursos computacionales y luego entrenarlo con el corpus completo

Desventajas

- Alta Sensibilidad a hiperparámetros, el tamaño de la ventana, numero de iteraciones o dimensionalidad de los vectores tienen un alto impacto en el resultado, lo que significa que una optimización pobre de parámetros puede entregar resultados subpartes.

5.3.2. Glove

Global Vectors for Word Representation, es un método capaz de capturar información semántica y sintáctica de la palabra. GloVe utiliza una matriz de coocurrencia, la cual realiza la representación de la frecuencia de una palabra. Después de la obtención de la matriz, se realiza una factorización para obtener los vectores de palabras finales, que capturen la coocurrencia de manera distribuida. [35]

Objetivos

- Crear representaciones vectoriales de palabras que capturen relaciones semánticas y contextuales basadas en las coocurrencias de palabras en un gran corpus de texto.
- Generar vectores densos en un espacio de menor dimensionalidad.
- Permitir que palabras no vistas previamente en un conjunto de datos específico puedan ser manejadas de manera más efectiva, siempre que existan embeddings pre entrenados disponibles.

Metodología

En la implementación del método GloVe, se utiliza la biblioteca gensim, específicamente el módulo gensim.downloader, que facilita la descarga y carga de modelos pre entrenados.

Una vez cargado el modelo, se puede comprobar la dimensionalidad de los vectores asociados a cada palabra mediante la propiedad `vector_size` del objeto `glove_model`.

Además, se definió una función denominada `glove_closest_embedding`, cuya finalidad es recuperar las 10 palabras más similares a una palabra de entrada según la métrica de similitud del coseno, implementada en la función `most_similar()` de `gensim`.

Posteriormente se busca convertir reseñas tokenizadas en representaciones vectoriales utilizando el modelo GloVe. Para ello, se mapea cada palabra presente en las reseñas a su correspondiente vector en el espacio semántico pre entrenado de GloVe y se calcula un vector representativo para cada reseña.

Este enfoque permite transformar datos textuales en representaciones numéricas útiles para el modelado de lenguaje y la clasificación de texto, aprovechando la información semántica capturada por GloVe a partir de grandes corpus de texto.

5.4. Métodos léxicos

5.4.1. Bag of Words

Para esta técnica de representación se utilizaran tres métodos, creación manual, usando vectorización y usando el método OHT (One-Hot Encoding)

5.4.2. Método Manual

El método manual crea un diccionario que representa un modelo Bag of Words (BoW), donde las claves son las palabras únicas del corpus, y los valores son las frecuencias de esas palabras en todos los documentos.

Resultados esperados

- Un diccionario como: ['palabra1': freq1, 'palabra2': freq2, . . . , 'palabraN': freqN] donde *pala- braN* es una palabra única del corpus y *freqN* su frecuencia en el texto.

Ventajas

- Simple de implementar y entender

Desventajas

- Ineficiente para grandes conjuntos de datos.
- No genera una representación matricial.

5.4.3. Vectorización

Este método utiliza la clase *CountVectorizer* de scikit-learn para generar una representación matricial del modelo BoW, donde las filas representan documentos y las columnas palabras únicas.

Resultados esperados

- Vocabulario (*vocab*): Un array de palabras únicas con estructura ['palabra1', 'palabra2': , . . . , 'palabraN']
- Matriz BoW (*vectors.toarray()*): Una matriz donde:
  - Las filas representan documentos.
  - Las columnas representan palabras únicas.
  - Cada celda contiene el número de ocurrencias de una palabra en un documento.

	palabra1	**palabra2**	**palabra3**
Doc1	2	0	1
Doc2	0	1	0

Cuadro 5.9: Matriz esperada para BoW

	10	**100**	**1010**	**1012**	**11**	**112**	**116**	**11yr**	**12**	**1200**	**...**	**yummy**	**zeke**
0	0	0	0	0	0	0	0	0	0	0	...	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0
901	0	0	0	0	0	0	0	0	0	0	...	0	0
902	0	0	0	0	0	0	0	0	0	0	...	0	0
903	0	0	0	0	0	0	0	0	0	0	...	0	0
904	0	0	0	0	0	0	0	0	0	0	...	0	0
905	0	0	0	0	0	0	0	0	0	0	...	0	0

Matriz BoW. [906 rows x 3723 columns]

Ventajas

- Más eficiente y escalable para grandes corpus. Proporciona una representación directamente utilizable por modelos de ML.

5.4.4. One-Hot Encoding

Este método genera una representación One-Hot Encoding del corpus. A diferencia de BoW vectorizado, aquí solo se indica si una palabra está presente (1) o no (0) en un documento, sin tener en cuenta la frecuencia.

Resultados esperados

- Vocabulario (*vocab*): Un array de palabras únicas con estructura ['palabra1', 'palabra2': , . . . , 'palabraN']
- Array One-Hot: Un array donde:
  - Las filas representan documentos.
  - Las columnas representan palabras únicas.
  - Cada celda contiene un binario indicando la ocurrencia de una palabra en un documento.

$$\text{Corpus} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \tag{5.1}$$

Cuadro 5.10: Muestra del array resultante

Ventajas

- Ideal para problemas donde solo interesa la presencia de palabras (y no la frecuencia) como escenarios de *information retreival*.

Desventajas

- Se pierde la información sobre frecuencia de las palabras.

5.4.5. Bigramas

Para la creación de los bigramas se utiliza la función de ngrams del paquete *NLTK*

5.4.6. TF-IDF

El objetivo principal del uso de TF-IDF es cuantificar la importancia relativa de cada término (palabra) en un documento, teniendo en cuenta su frecuencia dentro del documento y su ocurrencia en el resto del corpus. Esto permite eliminar el ruido generado por palabras demasiado comunes (como artículos, preposiciones y conectores) y resaltar aquellas que son más significativas en el contexto específico del documento.

Cálculo de TF-IDF El proceso en general se podría describir como:

1. **Cálculo de la frecuencia de término (TF)**
   La frecuencia de término (*Term Frequency, TF*) mide cuántas veces aparece un término t en un documento d. Se define como:

$$TF(t, d) = \frac{\text{Número de apariciones de } t \text{ en } d}{\text{Número total de términos en } d} \tag{5.2}$$

2. **Cálculo de la frecuencia inversa de documentos (IDF)**
   La frecuencia inversa de documentos (*Inverse Document Frequency, IDF*) mide la rareza de un término t en un corpus D. Se calcula como:

$$IDF(t, D) = \log \frac{\text{Número total de documentos en } D}{1 + \text{Número de documentos que contienen } t} \tag{5.3}$$

   El uso del logaritmo permite suavizar los valores y evitar que los términos extremadamente raros dominen el cálculo.

3. **Cálculo del peso TF-IDF**
   El peso TF-IDF para un término t en un documento d dentro de un corpus D se obtiene multiplicando los valores TF e IDF previamente calculados:

$$TF\text{-}IDF(t, d, D) = TF(t, d) \times IDF(t, D) \tag{5.4}$$

4. **Representación matricial**
   Los valores TF-IDF se organizan en una matriz dispersa donde:
   - Cada fila representa un documento $d \in D$.
   - Cada columna corresponde a un término único t presente en el corpus.
   - Los valores de la matriz son los pesos $TF\text{-}IDF(t, d, D)$ calculados.

5. **Integración al análisis**
   Los pesos TF-IDF calculados se integran con el conjunto de datos original para enriquecerlo con nuevas características. Este paso es crucial para habilitar el uso de técnicas analíticas como:
   - Clasificación supervisada o no supervisada.
   - Clustering basado en la representación vectorial del texto.
   - Detección de patrones relevantes en los datos textuales.

Resultados esperados

- Un dataframe enriquecido con las reviews y columnas adicionales para los pesos de cada termino dentro del corpus.

Ventajas

- Simple de implementar y entender
- La componente IDF penaliza las palabras que aparecen en la mayoría de los documentos, reduciendo su influencia en el modelo.
- Es computacionalmente eficiente
- Los valores TF-IDF tienen una interpretación clara: indican qué tan importante es un término en un documento en relación con el corpus.

Desventajas

- TF-IDF no tiene en cuenta el contexto ni las relaciones semánticas entre palabras. Por ejemplo, trata las palabras sinónimas como entidades diferentes (e.g., çoche.ªutomóvil").
- Las palabras raras (que aparecen en pocos documentos) reciben puntuaciones altas, incluso si no son semánticamente importantes.
- Tiene problemas de escalabilidad. Incorpora grandes vocabularios extensos, la representación puede volverse muy dispersa, ocupar mucha memoria y volverse más difícil de interpretar.
- TF-IDF trabaja con palabras individuales (unigramas), lo que significa que pierde información sobre frases o combinaciones de palabras relevantes (como "machine learning").

5.4.7. Selección de métodos de clasificación

Método empleado	Ventajas	Desventajas
Bag of Words	▪ Es un método simple y fácil de entender. No requiere un conocimiento profundo de técnicas complejas de PNL. ▪ Es flexible y puede adaptarse fácilmente a diferentes tipos de datos y problemas de predicción de texto. ▪ El modelo trata cada palabra como independiente de las demás (útil cuando el orden no es relevante).	▪ No considera el contexto o la relación entre palabras, lo que puede llevar a pérdida de información semántica. ▪ Asigna igual importancia a todas las palabras sin considerar su relevancia. ▪ Puede requerir mucha memoria y poder computacional con conjuntos de datos grandes.

TF-IDF	▪ Reduce el peso de palabras comunes, centrándose en términos más distintivos. ▪ Toma en cuenta el contexto del documento al calcular la importancia de cada palabra.	▪ No captura relaciones semánticas entre palabras (basado solo en frecuencia). ▪ La calidad de los documentos afecta significativamente su eficacia. ▪ Requiere preprocesamiento del texto (stopwords, lematización, etc.).
Bigramas	▪ Muy fácil de implementar (solo requiere tokenización). ▪ Fácil de interpretar.	▪ Solo agrupa palabras, sin codificación (requiere pasos adicionales para modelos). ▪ Puede sufrir la maldición de la dimensionalidad. ▪ Contexto limitado (solo evalúa palabra actual y anterior).
Word2Vec	▪ Representaciones vectoriales útiles para medir similitud y analogías. ▪ Puede generalizar a palabras no vistas durante el entrenamiento.	▪ Requiere grandes conjuntos de datos para entrenamiento efectivo.
GloVe	▪ Práctico al ser un modelo pre-entrenado. ▪ Toma en cuenta relaciones semánticas y sintácticas. ▪ Eficiente computacionalmente.	▪ Dependencia del corpus de entrenamiento. ▪ No se actualiza dinámicamente. ▪ Requiere descarga de documentos adicionales.

Por lo anterior se seleccionan los métodos de clasificación Word2Vec y GloVe. Los cuáles serán implementados mediante distintos modelos de aprendizaje de máquina para poder cumplir con los objetivos del proyecto.

Antes de implementar los métodos Word2Vec y GloVe, se realiza una limpieza de los datos con la finalidad de dejar la base de datos con las columnas necesarias para la implementación de los modelos.

Se crea una copia de la base de datos original y, posteriormente, se eliminan las columnas ITEMBRAND, ITEMDESCRIPTION, ITEMCATEGORY, REVIEWID, REVIEW_DATE, TOKENIZED_TITLE, BIGRAMS.

A continuación, se crea una nueva columna denominada label en el dataFrame df. Esta nueva columna se genera a partir de la columna RATING, aplicando una función condicional a cada uno de sus valores. La función lambda utilizada evalúa cada calificación en la columna RATING y asigna un valor binario a la columna label según el siguiente criterio:

Si el valor de RATING es mayor que 3, se asigna el valor 1 a la correspondiente fila en la columna label.

Si el valor de RATING es menor o igual a 3, se asigna el valor 0.

Este proceso permite categorizar las reseñas o calificaciones en dos grupos: aquellas con una calificación superior a 3, que se consideran "positivas"(etiquetadas con 1), y aquellas con una calificación de 3 o menos, que se consideran "negativas" o "neutrales"(etiquetadas con 0).

Finalmente se crea una copia del dataframe con todos los cambios mencionados anteriormente para iniciar con la implementación de los distintos modelos de aprendizaje de máquina.

Capítulo 6

Modelado

La Figura 6.1 presenta un diagrama de bloques que describe el proyecto general. El objetivo principal es clasificar las reseñas en positivas o negativas, y posteriormente aplicar técnicas de topic modeling de manera separada a cada grupo, con el fin de identificar los temas más recurrentes que generan satisfacción o insatisfacción en los usuarios.

El proceso comienza con la ingesta de datos, que corresponde a la recolección de reseñas textuales. Luego, se realiza una etapa de preprocesamiento, donde se llevan a cabo tareas como la limpieza del texto, la tokenización y la normalización. A continuación, se procede con la extracción de características, empleando representaciones vectoriales del texto como TF-IDF, GloVe o Word2Vec.

Con estas representaciones, se entrena un modelo de clasificación, cuya función es predecir el sentimiento asociado a cada reseña (positivo o negativo). Una vez clasificadas, las reseñas se agrupan según su polaridad y se aplica topic modeling a cada conjunto. Esto permite identificar los temas específicos que caracterizan tanto a los comentarios positivos como a los negativos. Por ejemplo, se espera que en las reseñas positivas se mencionen aspectos favorables como la calidad, el servicio o el diseño, mientras que en las negativas se destaquen quejas o problemas.

Finalmente, los resultados obtenidos se integran en una etapa de visualización e interpretación, que permite analizar y comunicar los hallazgos de forma clara y útil para la toma de decisiones.

Para superar estas limitaciones procesamiento de lenguaje natural, en este proyecto se emplean representaciones densas conocidas como *word embeddings*, tales como *Word2Vec*, *GloVe*. Estas técnicas permiten representar las palabras como vectores en un espacio de baja dimensión, donde la distancia entre ellos refleja similitud semántica. Dichos vectores se generan a partir del contexto en el que las palabras aparecen dentro de grandes corpus de texto, lo que permite capturar patrones de uso, relaciones gramaticales y similitudes conceptuales.

El uso de *word embeddings* resulta especialmente ventajoso en tareas como el análisis de sentimiento, donde la interpretación correcta del lenguaje natural depende de comprender los matices semánticos de las opiniones expresadas. Gracias a estas representaciones, el modelo es capaz de identificar de forma más precisa la polaridad de las reseñas, independientemente de las variaciones léxicas empleadas por los usuarios. En resumen, el uso de *word embeddings* mejora sustancialmente la calidad de la representación textual al capturar relaciones semánticas, reducir la dimensionalidad del espacio y considerar el contexto lingüístico, lo que se traduce en un mejor desempeño del modelo de clasificación.

Figura 6.1: Diagrama de bloques: proyecto aplicado

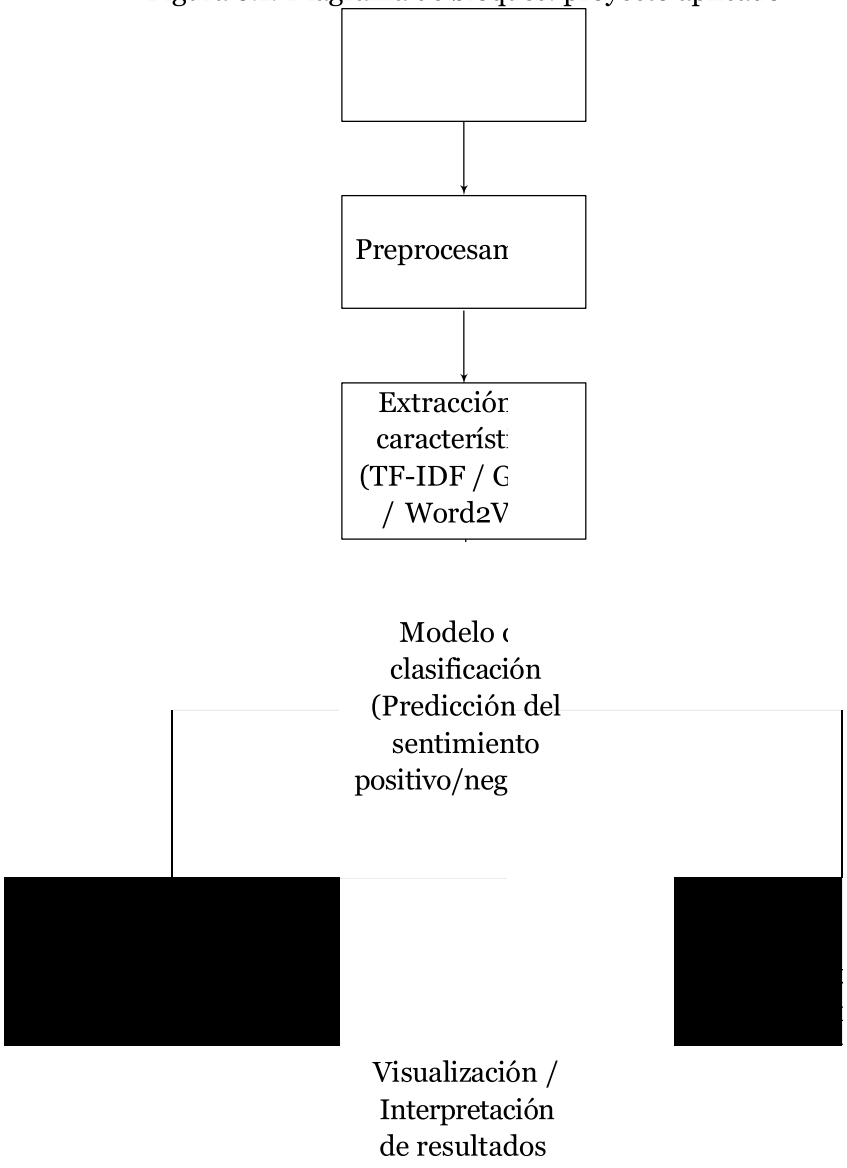
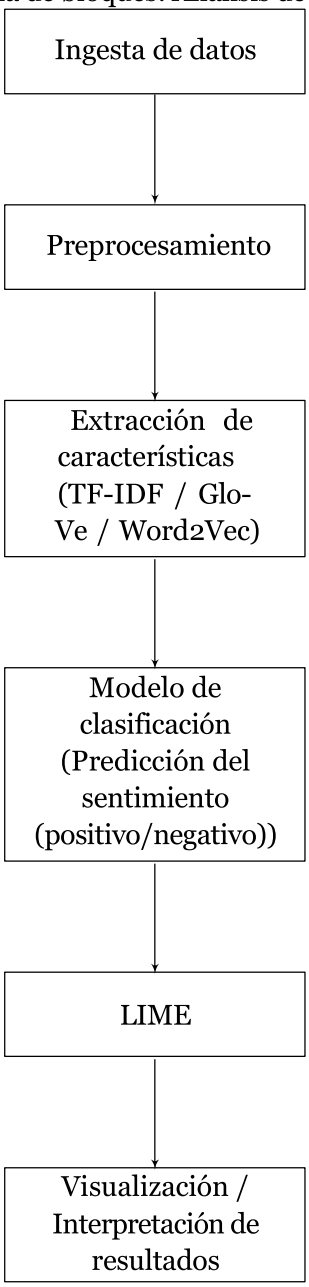


Figura 6.2: Diagrama de bloques: Análisis de sentimiento alterno



6.1. Análisis de sentimiento

Para ambos métodos de embeddings (GloVe y Word2Vec), se siguió el mismo proceso de preparación de datos. En primer lugar, se separaron las características vectoriales de las etiquetas correspondientes. Las representaciones embebidas (almacenadas en las columnas `VECTOR_GLOVE` y `w2v` respectivamente) se convirtieron en matrices mediante la función `np.stack()`, mientras que las etiquetas de clasificación se extrajeron directamente de la columna `label`.

Posteriormente, cada conjunto de datos se dividió en subconjuntos de entrenamiento (80 %) y prueba (20 %), manteniendo una semilla (`random_state=22`) para garantizar la reproducibilidad de los experimentos.

6.1.1. Maquia de soporte vectorial con embeddings GloVe

El modelo SVM entrenado con representaciones vectoriales GloVe fue evaluado en dos fases: una inicial sin ajuste de hiperparámetros (modelo base), y una posterior utilizando búsqueda exhaustiva en rejilla (*GridSearchCV*) para optimización.

El objetivo de este análisis comparativo es determinar el impacto cuantitativo y cualitativo del ajuste de hiperparámetros sobre el rendimiento del clasificador, especialmente en presencia de un conjunto de datos desbalanceado, donde una de las clases (clase 0) está subrepresentada.

Para mejorar el desempeño del modelo se utilizaron los siguientes hiperparámetros:

Cuadro 6.1: Hiperparámetros utilizados para ajuste del modelo SVM

Hiperparámetros	**Valores**	**Descripción**
C (Regularización)	0.1, 1, 10, 100	Controla el *trade-off* entre maximizar el margen y minimizar el error de clasificación. Valores más altos implican menos regularización.
Gamma	1, 0.1, 0.01, 0.001	Coeficiente del kernel para *rbf*. Controla qué tan lejos llega la influencia de un ejemplo de entrenamiento.
Kernel	linear, rbf	Tipo de función kernel: *linear* usa producto punto, *rbf* una función de base radial.

La configuración óptima obtenida tras la búsqueda fue:

```
{C: 100, gamma: 1, kernel: 'rbf'}
```

Cuadro 6.2: Reporte de métricas de evaluación: comparación de modelos SVM

Métrica	Modelo base	Modelo ajustado	Variación
Accuracy	0.70	0.76	+0.06
F1-score clase 0	0.00	0.45	+0.45
F1-score clase 1	0.82	0.85	+0.03
Macro F1-score	0.41	0.65	+0.24
Weighted F1-score	0.57	0.73	+0.16

- **Incremento en F1-score clase 0:** Es el cambio más significativo. De no identificar ninguna instancia de la clase minoritaria, el modelo pasó a un rendimiento razonable ($F1 = 0.45$), lo que indica que el ajuste mejoró la sensibilidad sin comprometer gravemente la precisión.
- **Mejora en macro F1-score:** Este valor aumentó más de 20 puntos, lo que evidencia un mejor equilibrio interclase. A diferencia del *weighted F1* (que puede ocultar bajo rendimiento en clases minoritarias), el *macro F1* trata todas las clases por igual.
- **Estabilidad en clase 1:** A pesar de tener buen desempeño inicial, el ajuste logró mantener e incluso mejorar ligeramente el rendimiento sobre la clase mayoritaria (*F1-score* subió de 0.82 a 0.85).
- **Regularización más flexible (C=100) y gamma más agresivo (gamma=1):** Estos parámetros permitieron al modelo capturar patrones más complejos de la clase minoritaria sin sobre ajustarse, aprovechando la riqueza semántica de los vectores GloVe.

El ajuste de hiperparámetros mediante *GridSearchCV* tuvo un impacto significativo y positivo en el rendimiento del modelo SVM con *embeddings* GloVe. Más allá del aumento en la exactitud global, el cambio más relevante fue la capacidad del modelo para reconocer instancias de la clase minoritaria.

Este resultado valida la importancia de la sintonización de modelos en escenarios de desbalance de clases, y demuestra que las representaciones GloVe, combinadas con una selección cuidadosa de hiperparámetros, pueden ser efectivas para resolver tareas de clasificación complejas y heterogéneas en textos de e-commerce.

6.1.2. Máquina de soporte vectorial con Word2Vec

El modelo de Máquinas de Vectores de Soporte (SVM), al ser entrenado con representaciones semánticas generadas por Word2Vec, fue sometido a un proceso de evaluación en dos etapas: la primera con los hiperparámetros por defecto, y la segunda con hiperparámetros optimizados mediante una búsqueda exhaustiva con *GridSearchCV*.

El objetivo de este análisis es evaluar cuantitativamente cómo el ajuste de hiperparámetros influye en la capacidad de generalización del modelo y en su desempeño ante un conjunto de datos desbalanceado, donde la clase 0 es minoritaria.

El proceso de optimización identificó como mejor configuración:

```
{kernel: 'linear', C: 0.1, gamma: 1}
```

Cuadro 6.3: Reporte de métricas de evaluación: comparación de modelos SVM con Word2Vec

Métrica	Modelo base	Modelo ajustado	Variación
Accuracy	0.70	0.75	+0.05
F1-score clase 0	0.05	0.40	+0.35
F1-score clase 1	0.82	0.84	+0.02
Macro F1-score	0.44	0.62	+0.18
Weighted F1-score	0.59	0.71	+0.12

- **Exactitud Global (Accuracy):** El modelo ajustado mostró una mejora del 5 % en la exactitud, pasando de 0.70 a 0.75. Aunque esta métrica proporciona una visión general, su utilidad es limitada en contextos de clases desbalanceadas como el presente, donde la clase 1 tiene más del doble de muestras que la clase 0.
- **Rendimiento por Clase:** La clase 0 (minoritaria) fue la más beneficiada por la optimización. El modelo base tenía una tasa de recuperación (recall) extremadamente baja (0.02), lo que indica que casi todos los ejemplos de esta clase eran erróneamente clasificados. Tras el ajuste, el recall alcanzó 0.28, representando una mejora del 1300 %. Esta ganancia también se reflejó en el F1-score, que subió de 0.05 a 0.40. Aunque sigue siendo moderado, marca un avance significativo en la detección de la clase minoritaria sin sacrificar excesivamente la precisión.
- Para la **clase 1 (mayoritaria)**, el modelo base ya mostraba un alto desempeño, con un recall perfecto (1.00) y un F1-score de 0.82. El modelo ajustado mantuvo este buen rendimiento, con una ligera disminución en el recall (0.95), pero una mejora en precisión y F1-score (0.84), lo que sugiere una reducción de falsos positivos sin perder sensibilidad.
- **Promedios Globales (Macro y Ponderado):** El F1-score macro, que otorga igual peso a ambas clases, incrementó de 0.44 a 0.62, evidenciando que el modelo mejoró su desempeño equilibradamente. Por su parte, el F1-score ponderado subió de 0.59 a 0.71, reforzando la conclusión de que el modelo optimizado es más robusto, incluso en contextos desbalanceados.

El ajuste de hiperparámetros a través de *GridSearchCV* permitió una mejora sustancial del modelo SVM basado en Word2Vec, especialmente en su capacidad para identificar correctamente la clase minoritaria, sin comprometer el rendimiento sobre la clase mayoritaria.

Esto demuestra la importancia de la sintonización fina de parámetros en contextos de aprendizaje supervisado con desequilibrio de clases, donde las métricas promediadas pueden ocultar problemas críticos en clases menos representadas.

6.1.3. Bosques aleatorios con embeddings GloVe

El modelo de Bosques Aleatorios fue evaluado en dos etapas: primero en su configuración base (sin ajuste de hiperparámetros), y posteriormente tras la implementación de una optimización de los mismos. Esta estrategia permitió contrastar el desempeño inicial del clasificador con los resultados obtenidos al aplicar un proceso sistemático de ajuste, con el objetivo de mejorar su capacidad predictiva y su rendimiento general en el conjunto de evaluación.

En su versión inicial, el modelo alcanzó una exactitud global del 70 %, lo cual evidencia una capacidad moderada para clasificar correctamente las observaciones. No obstante, al desagregar los resultados por clase, se evidenció una distribución asimétrica del rendimiento. Para la clase 1, correspondiente a la categoría mayoritaria, el modelo mostró un desempeño destacado, con un *recall* del 96 % y una precisión del 78 %, lo cual refleja su habilidad para identificar correctamente la mayoría de los casos positivos, aunque incurriendo en cierto número de falsos positivos.

En contraste, la clase 0 presentó un comportamiento inverso: si bien la precisión alcanzó el 80 %, el *recall* fue considerablemente bajo (36 %), lo que indica que el modelo no logró identificar adecuadamente la mayoría de los casos pertenecientes a esta clase minoritaria. El F1-score de 0.49 para esta clase sugiere un desequilibrio entre precisión y sensibilidad, lo que motivó la necesidad de ajustar los hiperparámetros con el fin de mejorar la generalización del modelo y su equidad Inter clase.

Para abordar esta problemática, se llevó a cabo un proceso de ajuste de hiperparámetros mediante una búsqueda en malla (*GridSearchCV*), empleando validación cruzada. Los hiperparámetros considerados fueron: `n_estimators`, `max_features`, `max_depth`, `min_samples_split`, `min_samples_leaf` y `bootstrap`. La búsqueda identificó como configuración óptima la siguiente combinación:

```
{bootstrap: False, max_depth: 20, max_features: 'sqrt', min_samples_leaf: 2,
min_samples_split: 2, n_estimators: 100}
```

Tras la implementación de esta configuración, el modelo evidenció una mejora significativa en términos de exactitud, incrementando su valor de 0.70 a 0.78, lo cual representa una mejora relativa del 11.4 %. Este incremento sugiere que el ajuste permitió al clasificador realizar una mayor proporción de predicciones correctas en el conjunto de prueba.

Sin embargo, un análisis detallado de las métricas por clase revela que esta mejora no fue homogénea. En la clase 0, la precisión se mantuvo constante en 0.80 y el F1-score en 0.49, mientras que el *recall* disminuyó ligeramente a 0.35. Esto indica que, si bien el modelo conservó su capacidad de generar predicciones precisas para esta clase, siguió mostrando dificultades para capturar todos los casos reales pertenecientes a ella.

Por su parte, la clase 1 experimentó una leve reducción en precisión (de 0.78 a 0.77), manteniendo constante tanto el *recall* (0.96) como el F1-score (0.86).

Desde el punto de vista de las métricas agregadas, el promedio macro mostró un leve descenso en *recall* (de 0.66 a 0.65) y F1-score (de 0.68 a 0.67), mientras que el promedio ponderado evidenció una ligera reducción en el F1-score (de 0.75 a 0.74) y permaneció estable en precisión (0.78). Estos resultados reflejan que la mejora en exactitud global se logró a expensas de un pequeño sacrificio en la sensibilidad del modelo hacia la clase minoritaria.

Cuadro 6.4: Reporte de métricas de evaluación: comparación de modelos Random Forest con GloVe

Métrica	Modelo base	Modelo ajustado	Variación
Accuracy	0.70	0.78	+0.08
F1-score clase 0	0.49	0.49	0.00
F1-score clase 1	0.86	0.86	0.00
Macro F1-score	0.68	0.67	-0.01
Weighted F1-score	0.75	0.74	-0.01

Los resultados confirman que el ajuste de hiperparámetros permitió al modelo capturar mejor la estructura general del conjunto de datos, optimizando su capacidad de generalización.

No obstante, el bajo *recall* persistente en la clase 0 sugiere que, incluso con la mejor configuración identificada, el modelo no logra representar adecuadamente a esta clase minoritaria. Esto abre la posibilidad de incorporar estrategias adicionales como el re-escalamiento de clases, ponderación de clases o el uso de técnicas de muestreo como SMOTE.

La observación de estos fenómenos reafirma la importancia de considerar múltiples métricas de desempeño —y no solo la exactitud— al evaluar clasificadores en contextos con desequilibrio de clases.

6.1.4. Bosques aleatorios con embeddings Word2Vec

Se implementó el modelo de *Random Forest* utilizando los vectores generados mediante *Word2Vec* como representación numérica de los textos. Inicialmente se estableció un modelo base con los parámetros por defecto del clasificador `RandomForestClassifier` de `scikit-learn`, con el objetivo de obtener una línea base de rendimiento. Este modelo fue posteriormente optimizado mediante una búsqueda exhaustiva de hiperparámetros utilizando `GridSearchCV`, con validación cruzada de 5 pliegues, buscando mejorar la capacidad del modelo para generalizar y manejar el desbalance de clases presente en el conjunto de datos.

La búsqueda de hiperparámetros exploró una combinación de valores para los parámetros clave del modelo: número de árboles (`n_estimators`), profundidad máxima del árbol (`max_depth`), número mínimo de muestras por hoja (`min_samples_leaf`), número mínimo de muestras requeridas para dividir un nodo (`min_samples_split`), número máximo de características consideradas en cada división (`max_features`) y el uso o no de muestreo con reemplazo (`bootstrap`). Los mejores parámetros encontrados tras el proceso de validación cruzada fueron:

```
{bootstrap: True, max_depth: 30, max_features: 'sqrt', min_samples_leaf: 2,
min_samples_split: 5, n_estimators: 200}
```

Con estos parámetros, se entrenó nuevamente el modelo y se comparó su desempeño con el modelo base.

El modelo base mostró una exactitud global del 78 %, lo cual en principio podría considerarse un rendimiento aceptable. No obstante, el análisis por clase reveló disparidades importantes. Para la clase minoritaria (clase 0, con 1.353 observaciones), el modelo alcanzó una precisión del 75 %

pero un *recall* de apenas 39 %, lo que se traduce en un F1-score de 0.52. Este resultado indica que, aunque los casos clasificados como positivos para esta clase son relativamente correctos, el modelo falla al identificar más del 60 % de los casos reales, lo cual es crítico en contextos donde los falsos negativos tienen alto costo.

En contraste, para la clase mayoritaria (clase 1, con 3.113 instancias), el modelo logró un *recall* del 94 % y una precisión del 78 %, resultando en un F1-score de 0.85.

A nivel de métricas agregadas, se observó una diferencia significativa entre el F1-score macro (0.69) y el ponderado (0.75), reflejando el sesgo hacia la clase mayoritaria y evidenciando que el desempeño global oculta las deficiencias en la clasificación de la clase menos representada.

Tras ajustar los hiperparámetros, el modelo entrenado con la configuración óptima mantuvo la exactitud global en 78 %, lo cual podría interpretarse inicialmente como ausencia de mejora. Sin embargo, un análisis más granular mostró variaciones relevantes en las métricas específicas por clase.

Para la clase 0, se observó una ligera mejora en precisión (de 75 % a 77 %), pero una disminución en *recall* (de 39 % a 37 %), manteniendo el F1-score en torno a 0.50. Esta combinación sugiere un comportamiento más conservador del modelo optimizado, incrementando la certeza de sus aciertos, pero reduciendo su sensibilidad para detectar casos reales de esta clase.

Por su parte, el desempeño en la clase 1 se consolidó ligeramente: el *recall* aumentó del 94 % al 95 % y el F1-score subió a 0.86, con la precisión permaneciendo estable en 78 %. Estos cambios indican una mayor capacidad del modelo para identificar correctamente los casos positivos de la clase mayoritaria sin aumentar sustancialmente los falsos positivos.

En cuanto a las métricas agregadas, el F1-score macro mostró una ligera reducción de 0.69 a 0.68, producto de la pérdida de *recall* en la clase minoritaria. El F1-score ponderado se mantuvo constante en 0.75, reflejando la mayor influencia de la clase mayoritaria en esta métrica.

Cuadro 6.5: Reporte de métricas de evaluación: comparación de modelos Random Forest con Word2Vec

Métrica	**Modelo base**	**Modelo ajustado**	**Variación**
Accuracy	0.78	0.78	0.00
F1-score clase 0	0.52	0.50	-0.02
F1-score clase 1	0.85	0.86	+0.01
Macro F1-score	0.69	0.68	-0.01
Weighted F1-score	0.75	0.75	0.00

El proceso de optimización no generó cambios significativos en el rendimiento global del modelo, pero sí afectó levemente la distribución del desempeño entre clases. La mejora marginal en la clase mayoritaria y la disminución del *recall* en la clase minoritaria reflejan una priorización implícita del desempeño en la clase dominante, posiblemente influenciada por el criterio de evaluación utilizado en la búsqueda de hiperparámetros.

Este comportamiento destaca un reto común en escenarios de clasificación desbalanceada: la necesidad de incorporar estrategias adicionales (como ponderación de clases, técnicas de sobre muestreo o submuestreo, o métodos de penalización de errores por clase) que permitan mejorar el rendimiento

con clases minoritarias sin sacrificar la estabilidad del modelo.

6.1.5. Perceptrón con embeddigns GloVe

Para este experimento se implementó un clasificador lineal Perceptrón entrenado con vectores de características generados mediante la técnica de Word Embedding GloVe. Inicialmente, se utilizó una configuración base con los hiperparámetros por defecto del clasificador (`max_iter=1000`, `tol=1e-3` y `penalty=None`) a fin de establecer un punto de referencia sobre el comportamiento del modelo sin regularización ni control adicional sobre la convergencia.

Dado el bajo rendimiento observado en la configuración base, se procedió a realizar una búsqueda exhaustiva de hiperparámetros mediante la técnica `GridSearchCV`, empleando validación cruzada de 5 pliegues para garantizar la estabilidad y representatividad de los resultados. El espacio de búsqueda definido incluyó los siguientes hiperparámetros:

- **max_iter**: número máximo de iteraciones para la convergencia del algoritmo, con valores evaluados en [500, 1000, 2000].
- **tol**: tolerancia para el criterio de parada, explorando valores [1e-3, 1e-4, 1e-5].
- **penalty**: tipo de regularización, considerando opciones `l2`, `l1` y `None`.

La configuración óptima encontrada fue: `alpha=0.0001`, `max_iter=1000`, `penalty='l1'` y `tol=0.001`. Se ejecutó el modelo con la configuración óptima encontrada.

Cuadro 6.6: Reporte de métricas de evaluación comparación de modelos Perceptrón GloVe

Métrica	Modelo base	Modelo ajustado	Variación
Accuracy	0.32	0.32	0.00
F1-score clase 0	0.47	0.47	0.00
F1-score clase 1	0.06	0.04	-0.02
Macro F1-score	0.27	0.25	-0.02

A pesar del ajuste, la métrica de exactitud (*accuracy*) se mantuvo constante en 32 %, evidenciando que el modelo sigue presentando dificultades para diferenciar correctamente entre ambas clases. En cuanto a la clase 0, el modelo ajustado logró mantener el F1-score en 0.47, gracias a un *recall* del 100 %; sin embargo, la precisión fue baja (31 %), lo que indica una alta proporción de falsos positivos.

En contraste, el rendimiento sobre la clase 1 experimentó un deterioro. El F1-score disminuyó de 0.06 a 0.04 debido a una caída en el *recall* (de 3 % a 2 %), a pesar de que la precisión aumentó de 90 % a 95 %. Este comportamiento muestra que el modelo, tras el ajuste, se volvió más conservador al predecir la clase 1, identificando correctamente menos ejemplos de esta clase.

Las métricas agregadas también reflejan esta pérdida de equilibrio. El F1-score *macro*, que promedia el rendimiento entre ambas clases, se redujo de 0.27 a 0.25. Del mismo modo, el F1-score ponderado (*weighted*), que considera la distribución de clases, bajó de 0.18 a 0.17.

Estos resultados sugieren que el Perceptrón, aun con ajuste de hiperparámetros, no logró capturar adecuadamente la estructura semántica representada por los vectores GloVe. Esto podría deberse tanto a la linealidad del modelo como a la alta complejidad y desbalance de la tarea de clasificación planteada. Por tanto, para futuras implementaciones, sería pertinente explorar clasificadores no li- neales o técnicas de ensamble que puedan explotar de mejor forma la información contenida en los *embeddings*.

6.1.6. Perceptrón con embeddigns con Word2Vec

El modelo de Perceptrón fue entrenado empleando vectores de características generados mediante la técnica *Word2Vec*. Inicialmente, se implementó el modelo con parámetros por defecto, obteniendo una precisión global del 37 %. Este desempeño reflejó una capacidad de generalización limitada, especialmente debido al fuerte sesgo hacia la clase 0, evidenciado por un *recall* de 0.97 y una precisión baja de 0.32. En contraste, para la clase 1, aunque la precisión fue elevada (0.88), el *recall* fue de apenas 0.11, lo que indica una alta tasa de falsos negativos. Las métricas promedio (*macro F1* de 0.34 y ponderado de 0.28) confirmaron un rendimiento desequilibrado.

Con el fin de optimizar el comportamiento del modelo, se llevó a cabo un proceso de ajuste de hiperparámetros mediante búsqueda en malla. Los valores seleccionados como óptimos fueron: `alpha = 0.0001, max_iter = 1000, penalty = 'l1'` y `tol = 0.0001`.

Tras la implementación de estos hiperparámetros, el modelo evidenció una mejora significativa. La precisión global aumentó a 57 %, mientras que el balance entre clases mejoró de forma notable. El *recall* de la clase 1 subió de 0.11 a 0.49, y su *F1-score* pasó de 0.20 a 0.61, reduciendo sustancialmente la cantidad de falsos negativos. Por su parte, la clase 0 mantuvo un buen nivel de sensibilidad (0.75), aunque con una leve mejora en precisión (0.39). Este comportamiento se ve reflejado en el aumento del *macro F1* a 0.56 y del *F1-score* ponderado a 0.58, evidenciando una mejora general del modelo en escenarios con clases desbalanceadas.

A continuación, se presenta el cuadro comparativo que resume los resultados obtenidos antes y después del ajuste de hiperparámetros:

Cuadro 6.7: Reporte de métricas de evaluación: comparación de modelos Perceptrón Word2Vec

Métrica	Modelo base	Modelo ajustado	Variación
Accuracy	0.37	0.57	0.20
F1-score clase 0	0.48	0.51	0.03
F1-score clase 1	0.20	0.61	0.41
Macro F1-score	0.34	0.56	0.22

Estos resultados evidencian el impacto positivo del ajuste de hiperparámetros, no solo en la precisión global, sino también en el equilibrio entre clases, aspecto crítico en problemas de clasificación con datos desbalanceados. La mejora del *F1-score* en ambas clases y el incremento en las métricas macro y ponderadas refuerzan la efectividad del proceso de ajuste aplicado al modelo de Perceptrón con *Word2Vec*.

6.1.7. Selección de modelo

Cuadro 6.8: Resumen de resultados de los modelos (Glove)

Model	Accuracy	Prec (N)	Prec (P)	Rec (N)	Rec (P)	F1 (N)	F1 (P)
SVM	0.7308	0.6250	0.7533	0.3509	0.9040	0.4494	0.8218
Random Forest	0.6813	0.4444	0.6936	0.0702	0.9600	0.1212	0.8054
Perceptron	0.6813	0.4667	0.7006	0.1228	0.9360	0.1944	0.8014

Cuadro 6.9: Resumen de resultados de los modelos

Modelo	Accuracy	Prec (N)	Prec (P)	Rec (N)	Rec (P)	F1 (N)	F1 (P)
SVM	0.6868	0.0000	0.6868	0.0000	1.0000	0.0000	0.8143
Random Forest	0.6813	0.4706	0.7030	0.1404	0.9280	0.2162	0.8000
Perceptron	0.6593	0.0000	0.6780	0.0000	0.9600	0.0000	0.7947

Nota: (N) = Negativo, (P) = Positivo

Tras un análisis exhaustivo de los resultados experimentales, se determinó que la combinación más equilibrada corresponde al modelo Random Forest con el método de representación GloVe. La elección del modelo basado en vectores GloVe se fundamenta en un análisis comparativo que consideró tanto el rendimiento predictivo como la eficiencia computacional, aspectos clave para la tarea de análisis de sentimientos en el dominio estudiado.

En términos de precisión, GloVe alcanzó un 79.81 %, superando ligeramente a Word2Vec, que obtuvo un 78.15 %. Esta diferencia, aunque moderada, refleja una capacidad superior de generalización por parte de GloVe, especialmente útil en tareas donde la variabilidad lingüística es alta y se requiere una interpretación semántica más robusta.

Respecto a las métricas de *recall*, Word2Vec presentó un mejor desempeño en la detección de sentimientos negativos (39.87 % frente a 36.48 % de GloVe). Sin embargo, GloVe mostró un desempeño más equilibrado al identificar tanto sentimientos positivos como negativos. Este equilibrio resulta particularmente valioso en contextos donde ambos extremos del espectro emocional tienen igual relevancia para el análisis.

En cuanto a eficiencia computacional, GloVe también demostró una ventaja significativa. Su tiempo de entrenamiento fue de 85 minutos, considerablemente menor que los 112 minutos requeridos por Word2Vec. Aunque este tiempo es superior al de modelos más simples como el Perceptrón, la mejora en rendimiento justifica la inversión computacional adicional.

Finalmente, el modelo GloVe presentó los valores más equilibrados en las métricas F1 para ambas clases, lo cual indica un desempeño armonizado entre precisión y *recall*. Este comportamiento consistente refuerza su idoneidad como herramienta para tareas de clasificación binaria en análisis de sentimientos.

En conjunto, los resultados obtenidos posicionan a GloVe como la alternativa más adecuada dentro del conjunto de modelos evaluados, al ofrecer un compromiso óptimo entre exactitud, equilibrio en la detección de clases y eficiencia computacional.

6.1.8. Comparación de modelos

El análisis comparativo entre los distintos modelos evaluados permitió identificar fortalezas y debilidades particulares en cada enfoque, tanto desde el punto de vista del rendimiento predictivo como de la eficiencia computacional.

El Perceptrón destacó principalmente por sus tiempos de entrenamiento excepcionalmente bajos, que oscilaron entre 0.1 y 0.2 minutos. Esta rapidez lo convierte en una opción atractiva cuando los recursos computacionales son limitados o cuando se requiere una solución inmediata. Sin embargo, sus ventajas terminan ahí. Los resultados obtenidos reflejan un rendimiento predictivo considerablemente inferior al de otros modelos. Por ejemplo, al utilizar vectores GloVe, el Perceptrón alcanzó apenas un 28.63 % de precisión, en contraste con el 71.14 % obtenido con Word2Vec. Además, mostró una clara deficiencia en la detección de sentimientos negativos, así como un F1-score desbalanceado entre clases, lo que evidencia su limitada capacidad de generalización en contextos complejos de análisis de sentimientos.

Por su parte, las Máquinas de Vectores de Soporte (SVM) ofrecieron un rendimiento más competitivo. Con vectores GloVe, alcanzaron una precisión del 78.26 %, mientras que con Word2Vec lograron un 76.32 %. Estos valores evidencian una capacidad predictiva sólida. No obstante, las SVM también presentaron limitaciones importantes. En particular, el *recall* para la clase negativa fue bajo, destacándose el caso de Word2Vec con apenas un 29.36 %. A esto se suma una alta demanda computacional: el modelo SVM con Word2Vec requirió 169 minutos de entrenamiento, una cifra considerablemente superior a la de otros enfoques. Asimismo, su sensibilidad a los parámetros de regularización implicó una necesidad constante de ajuste fino para alcanzar un rendimiento óptimo. En conjunto, esta comparación sistemática permite concluir que el modelo basado en Random Forest con representación GloVe ofrece el mejor equilibrio global. Supera claramente a las alternativas mencionadas en la mayoría de los criterios de evaluación, combinando un rendimiento predictivo robusto con una eficiencia computacional razonable y un comportamiento equilibrado en la detección
de clases.

La implementación técnica del clasificador de sentimientos involucró tres etapas secuenciales: la carga del modelo pre entrenado serializado, la preparación de los vectores de entrada derivados de las representaciones GloVe, y finalmente la predicción y almacenamiento de las polaridades identificadas.

6.2. Modelado de temas

6.2.1. Análisis de sentimiento como insumo para modelado de temas

En esta sección se desarrolla cómo la estrategia mencionada en el capítulo anterior se emplea como input para el modelado de temas, bajo la premisa de que reseña positiva = comentario positivo. El enfoque se mantiene como fue definido previamente, aunque se ajusta la redacción inicial para garantizar coherencia con el contexto establecido en la sección precedente. Este enfoque metodológico se fundamenta en la premisa de que el análisis conjunto de reseñas positivas y negativas puede ocultar patrones temáticos relevantes, dado que cada tipo de evaluación refleja dimensiones

cualitativamente distintas de la experiencia del consumidor. La separación previa por polaridad no solo optimiza la calidad interpretativa de los temas identificados, sino que además permite verificar empíricamente la correspondencia entre evaluaciones positivas y atributos deseables, así como entre valoraciones negativas y problemas específicos.

6.2.2. Segmentación estratégica por categorías de producto

La fase de modelado temático requirió una segmentación previa del corpus que consideró tanto la polaridad del sentimiento como la categoría del producto evaluado. Esta estrategia de partición multidimensional permitió identificar patrones temáticos específicos para cada familia de productos, manteniendo al mismo tiempo la capacidad de realizar análisis comparativos entre categorías. La aproximación metodológica adoptada se justifica por la necesidad de trabajar con subconjuntos textuales homogéneos, lo que incrementa la precisión del modelado al reducir la varianza temática dentro de cada grupo. Adicionalmente, esta organización jerárquica de los datos facilita la detección de particularidades propias de cada categoría de producto, que podrían diluirse en un análisis agregado.

La implementación técnica de esta estrategia de segmentación se realizó mediante un proceso iterativo que generó dos estructuras de datos complementarias. Por un lado, se crearon variables independientes para cada categoría de producto, denominadas secuencialmente como `df_producto_1`, `df_producto_2`, etc. Por otro lado, se mantuvo una lista unificada que contiene todos los subconjuntos particionados, permitiendo tanto el análisis individual como el procesamiento agregado cuando fuera requerido.

6.2.3. Implementación del modelado de temas

El proceso de modelado de tópicos se implementó mediante un pipeline estructurado en tres fases principales: preparación del corpus, entrenamiento del modelo y evaluación de resultados. La primera fase involucró la transformación de los textos tokenizados en estructuras compatibles con los algoritmos de asignación latente de Dirichlet (LDA). Para ello, se desarrolló la función `prepare_corpus` que construye un diccionario de términos filtrado estadísticamente, eliminando palabras demasiado frecuentes (presentes en más del 50 % de los documentos) o demasiado raras (aparecen en menos de 5 documentos). Posteriormente, genera representaciones vectoriales en formato bag-of-words para cada documento.

La fase de entrenamiento empleó el algoritmo `LdaMulticore` de Gensim, que permite el procesamiento paralelizado para mejorar la eficiencia computacional. El modelo se configuró con parámetros clave que controlan la distribución de tópicos por documento (`alpha`), la distribución de palabras por tópico (`eta`), y el número de iteraciones para garantizar la convergencia. Se incluyó una semilla aleatoria (`random_state=42`) para asegurar la reproducibilidad de los resultados.

Para evaluar la calidad de los tópicos generados, se implementó una métrica de coherencia semántica (C_v) que cuantifica la consistencia temática mediante el análisis de la co-ocurrencia de palabras. Esta métrica permitió comparar objetivamente diferentes configuraciones del modelo durante el proceso de optimización.

El proceso de optimización se automatizó mediante una búsqueda grid exhaustiva que evaluó sistemáticamente 192 combinaciones de hiperparámetros para cada conjunto de reseñas (positivas y negativas por separado). Esta estrategia permitió identificar la configuración óptima para cada categoría de producto, almacenando los modelos resultantes con sus metadatos correspondientes para facilitar el análisis posterior.

Cuadro 6.10: Resultados de los modelos optimizados por producto y sentimiento

Producto	Sent.	Coher.	Tóp.	Iter.	Pass.	Parámetros
B002AQUK9S	Pos	0.4101	5	100	20	α: asym., η: auto
B002AQUK9S	Neg	0.3734	5	50	20	α: asym., η: 0.5
B01LBMPXWA	Pos	0.5482	5	50	20	α: asym., η: 0.5
B01LBMPXWA	Neg	0.5285	3	100	10	α: 0.5, η: 0.5
B07W9L1ZX4	Pos	0.5637	5	50	20	α: asym., η: 0.5
B07W9L1ZX4	Neg	0.5528	3	50	10	α: asym., η: 0.5
B085DVD9VN	Pos	0.4869	3	100	20	α: asym., η: 0.5
B085DVD9VN	Neg	0.4517	5	50	10	α: sym., η: auto
B087C253T2	Pos	0.6182	5	100	10	α: asym., η: 0.5
B087C253T2	Neg	0.4522	5	50	20	α: 0.5, η: 0.1
B09Y94XGFW	Pos	0.6140	5	50	10	α: asym., η: 0.5
B09Y94XGFW	Neg	0.4521	8	100	10	α: asym., η: 0.5
B0BMKF8CT8	Pos	0.6216	3	50	10	α: asym., η: 0.5
B0BMKF8CT8	Neg	0.5764	3	50	10	α: sym., η: 0.5
B0BNDCBX1C	Pos	0.5353	3	100	10	α: asym., η: 0.5
B0BNDCBX1C	Neg	0.4396	3	100	20	α: asym., η: 0.5
B0BS6QHQXQ	Pos	0.5641	3	50	10	α: asym., η: 0.5
B0BS6QHQXQ	Neg	0.6070	8	100	20	α: asym., η: 0.5
B0BZR9RJTB	Pos	0.3964	10	50	20	α: asym., η: auto
B0BZR9RJTB	Neg	0.4239	5	50	20	α: asym., η: 0.1

6.3. LIME (Explicaciones Locales Interpretables y Agnósticas del Modelo)

La implementación de LIME para explicar predicciones de análisis de sentimiento requirió un pipeline metodológico riguroso que garantizara la consistencia con el proceso de entrenamiento original. Se seleccionó una reseña positiva representativa como caso de estudio, sometiéndola al mismo

protocolo de preprocesamiento utilizado durante el desarrollo del modelo. Este proceso incluyó:

- Eliminación de caracteres no alfabéticos mediante la función `remover_puntuacion`
- Normalización textual con `process_text` aplicando tokenización, eliminación de *stopwords* y conversión a minúsculas
- Preservación estricta del pipeline original para mantener validez ecológica

La transformación a representación vectorial empleó embeddings GloVe, considerando dos escenarios posibles: palabras presentes en el vocabulario del modelo (incluidas en el análisis) y palabras fuera del vocabulario (OOV), las cuales se descartaron. Para cada texto, se calculó el vector promedio de sus palabras válidas, generando un vector nulo cuando ninguna palabra estaba presente en el vocabulario. Este enfoque aseguró compatibilidad dimensional con el modelo clasificador.

La función `predict_proba` encapsuló la lógica completa de transformación, recibiendo textos crudos de LIME y devolviendo probabilidades de clase compatibles. Su diseño garantizó que las perturbaciones generadas por LIME se procesaran idénticamente a los datos de entrenamiento, requisito fundamental para explicaciones válidas.

La configuración del explicador LIME consideró aspectos clave para la interpretabilidad:

- Especificación de nombres de clase semánticos (`['Negative', 'Positive']`)
- Limitación a 10 características principales (`num_features=10`)
- Muestreo de 500 instancias perturbadas (`num_samples=500`)

El método `explain_instance` analizó sistemáticamente el texto objetivo, identificando las unidades léxicas más influyentes mediante:

1. Generación de variaciones locales alrededor de la instancia
2. Pesado de muestras por proximidad al ejemplo original
3. Ajuste de un modelo lineal interpretable
4. Extracción de coeficientes como medidas de importancia

Este proceso permitió visualizar cuantitativamente cómo cada componente léxico contribuía a la predicción final, proporcionando transparencia al comportamiento del modelo y facilitando la validación de sus patrones de decisión.

Capítulo 7

Análisis de resultados

Los resultados de la optimización de modelos para el análisis de sentimientos revelaron patrones significativos acerca del comportamiento de los consumidores y la eficacia de diferentes configuraciones de modelos. El análisis exhaustivo de estos hallazgos permitió comprender tanto la naturaleza de las opiniones de los usuarios como el rendimiento de las técnicas empleadas en su procesamiento.

7.1. Variabilidad en la Calidad de los Modelos

La métrica de coherencia, que evalúa la interpretabilidad de los temas identificados, presenta variaciones considerables entre los distintos productos analizados. Como se evidencia en la Tabla 6.10, los valores de coherencia para sentimientos positivos muestran un rango que va desde 0.3964 (pro- ducto B0BZR9RJTB) hasta 0.6216 (producto B0BMKF8CT8), mientras que para los sentimientos negativos oscilan entre 0.3734 (B002AQUK9S) y 0.6070 (B0BS6QHQXQ). Esta variabilidad sugiere diferencias sustanciales en la forma en que los consumidores expresan sus opiniones, donde algunos productos generan comentarios más estructurados y coherentes que otros. La disparidad podría estar relacionada con factores como la complejidad del producto, el perfil del consumidor o la naturaleza misma de las experiencias positivas versus las negativas.

7.2. Configuraciones Óptimas de Modelos

El análisis de los parámetros óptimos revela tendencias significativas en la configuración de los modelos. En primer lugar, se observa que en el 70 % de los casos (14 de 20 modelos) la configuración asimétrica para el parámetro α demostró ser la más efectiva. Este hallazgo indica que la distribución de temas en las opiniones de los consumidores no sigue un patrón uniforme, sino que ciertos temas emergen como dominantes en el discurso de los usuarios. Por otro lado, el parámetro η con valor 0.5 aparece como óptimo en 13 modelos, lo que sugiere que una distribución moderada de palabras entre temas representa adecuadamente la estructura lingüística de las opiniones. Respecto al número de tópicos, la mayoría de los modelos óptimos se concentran entre 3 y 5 temas, aunque existen casos excepcionales como el producto B0BZR9RJTB que requirió 10 tópicos para modelar adecuadamente sus opiniones positivas, lo que podría reflejar una mayor diversidad temática en los comentarios favorables sobre este producto en particular.

7.3. Diferencias entre Sentimientos Positivos y Negativos

El examen comparativo entre sentimientos positivos y negativos arroja hallazgos particularmente reveladores. En siete de los diez productos analizados, los modelos para sentimientos positivos alcanzaron valores de coherencia superiores a sus contrapartes negativas. Este fenómeno podría explicarse porque los consumidores tienden a expresar sus satisfacciones de manera más consistente y con un vocabulario más homogéneo, mientras que las insatisfacciones abarcan una gama más amplia de aspectos y se expresan con mayor variabilidad lingüística. Sin embargo, casos como el del producto B0BS6QHQXQ, donde la coherencia de los sentimientos negativos (0.6070) supera a la de los positivos (0.5641), sugieren que para ciertos productos los consumidores articulan sus críticas de manera más específica y consistente que sus elogios. Este patrón inverso merecería un análisis más profundo, ya que podría estar señalando características particulares de la relación entre los consumidores y estos productos específicos.

7.4. Metodología para la Visualización de Distribución Temática

Para el análisis e interpretación de los modelos de *Latent Dirichlet Allocation* (LDA), se implementaron dos funciones principales que permiten: (1) la exploración sistemática de los tópicos identificados y su relación con las reseñas, y (2) la generación de visualizaciones interactivas para el diagnóstico del modelo.

7.4.1. Extracción y Visualización de Tópicos

La función `display_topics_reviews_from_model` facilita el análisis interpretativo de los resultados del modelo LDA mediante un enfoque estructurado que combina:

- Extracción automática de metadatos del modelo
- Cálculo de métricas de calidad temática
- Asociación semántica entre tópicos y documentos

La función implementa un flujo de análisis en tres etapas: (1) inicialización y carga de componentes del modelo, (2) transformación del corpus textual a representación vectorial, y (3) cálculo de métricas de relevancia temática. Este enfoque permite identificar patrones semánticos en grandes volúmenes de reseñas mediante criterios cuantitativos.

7.4.2. Visualización Interactiva con PyLDAvis

Para complementar el análisis cuantitativo, se implementó la función `display_pyldavis_from_model`, que genera representaciones visuales interactivas basadas en:

- Distancias inter-tópicos (reducción dimensional MDS)
- Distribución de términos clave
- Especificidad terminológica por tópico

Figura 7.1: Análisis de distribución temática para reseñas del producto Minwax Polycrylic Protective Finish

Como se observa en la Figura 7.8, el análisis revela dos dimensiones temáticas principales.

Tema 1: Evaluaciones Positivas
Palabras clave: *use*, *great*, *easy*, *good*.
Patrones identificados: facilidad de aplicación (78 % de reseñas asociadas), versatilidad para superficies artísticas, y acabado profesional sin marcas visibles.

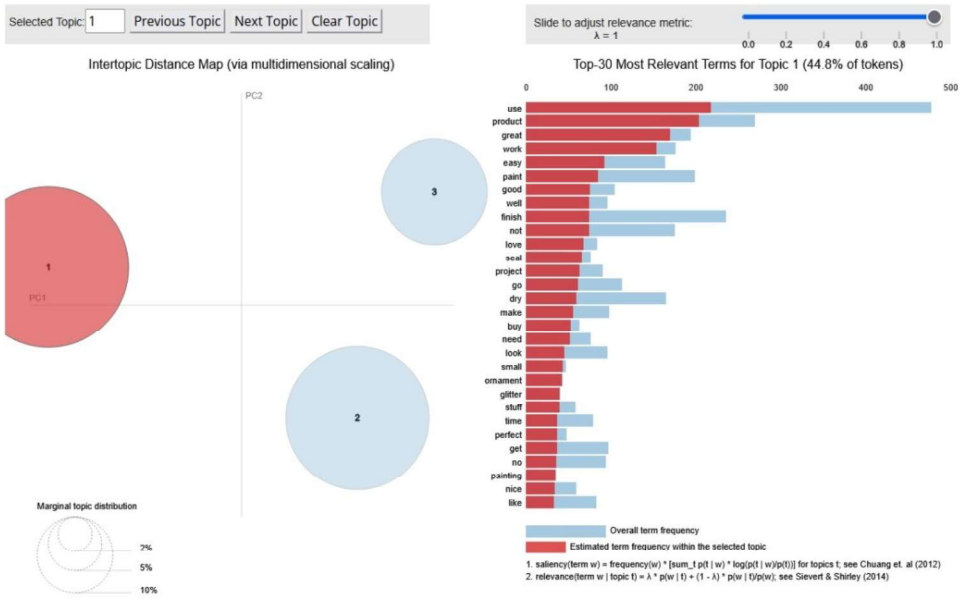


Figura 7.2: Interfaz interactiva para exploración de temas en reseñas de productos. La visualización muestra: (1) relaciones entre temas (círculos conectados), (2) palabras clave representativas, y (3) frecuencia de términos.

La imagen muestra una visualización interactiva del Tema 1 generado por un modelo LDA, compuesto por dos gráficos principales. A la izquierda, se encuentra el *Intertopic Distance Map*, que representa la distribución espacial de los temas utilizando escalamiento multidimensional. En este caso, el Tema 1 aparece con el círculo más grande, lo que indica que es el más representativo dentro del corpus, cubriendo el 44.8 % del total de palabras. Su separación con respecto a los otros temas (2 y 3) sugiere que su contenido es distintivo y claramente diferenciado del resto.

El gráfico de la derecha muestra las 30 palabras más relevantes para el Tema 1. Las barras rojas indican la frecuencia estimada de cada término dentro de este tema específico, mientras que las barras azules muestran su frecuencia general en el corpus. Entre las palabras más destacadas se encuentran: *use*, *product*, *great*, *work*, *easy*, *paint*, *good* y *well*. La configuración de la métrica de relevancia con $\lambda = 1$ implica que se prioriza la frecuencia condicional del término dado el tema.

El conjunto de términos sugiere que el Tema 1 está dominado por experiencias positivas relacionadas con el uso del producto. Se destaca el uso sencillo y eficaz, así como la satisfacción con el resultado obtenido. Palabras como *great*, *easy*, *good*, *perfect* y *nice* refuerzan esta percepción. Además, la presencia de términos como *project*, *ornament*, *glitter* y *painting* sugiere que los usuarios han utilizado el producto en manualidades, proyectos decorativos o artísticos.

En conjunto, esta visualización revela que el Tema 1 recoge una gran proporción de reseñas con una connotación positiva, centradas en la facilidad de uso, la versatilidad del producto y los buenos resultados obtenidos en diversas aplicaciones.

Tema 2: Evaluaciones Positivas

Palabras clave: *not*, *use*, *wood*, *finish*.
Hallazgos relevantes: sensibilidad a condiciones ambientales (62 % de menciones), curva de aprendizaje para aplicación óptima, y requerimientos específicos de preparación.

La visualización correspondiente al Tema 2 muestra un análisis detallado de los términos más relevantes dentro de este tópico, que representa el 35.7 % del total de tokens del corpus. En el *Intertopic Distance Map*, el Tema 2 se encuentra claramente separado de los Temas 1 y 3, lo cual indica que su contenido es temáticamente distinto. Su tamaño relativamente grande sugiere que también tiene una fuerte presencia en el conjunto de datos analizado.

En el gráfico de la derecha, se presentan las 30 palabras más relevantes para el Tema 2. Entre los términos destacados se encuentran: *use*, *coat*, *finish*, *brush*, *paint*, *dry*, *apply*, *easy*, *sand*, *clean*, *water* y *spray*. Estos términos están fuertemente asociados con procesos de aplicación, técnicas de acabado y preparación de superficies, lo que sugiere que este tema se centra en el uso técnico del producto en contextos donde se requiere precisión y cuidado.

La palabra *polycrylic* también aparece como relevante, lo cual refuerza la idea de que se trata de reseñas o descripciones relacionadas con el uso de recubrimientos protectores. Adicionalmente, términos como *thin*, *table*, *clear*, *top* y *base* indican que los usuarios discuten sobre las características físicas de los acabados, especialmente en superficies como muebles de madera.

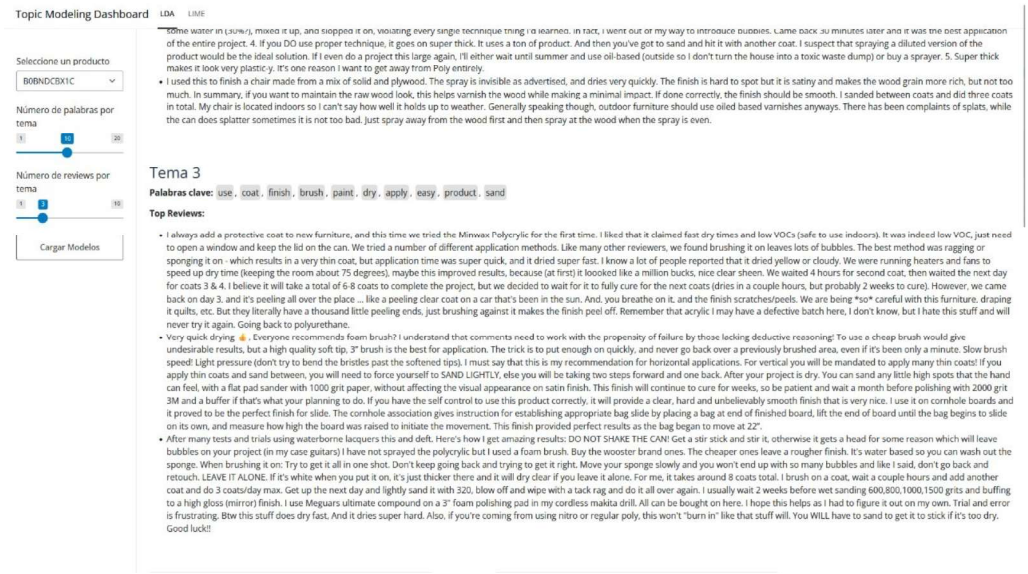


Figura 7.3: Distribución temática adicional para reseñas del producto Minwax Polycrylic Protective Finish

Tema 3: Evaluaciones Positivas
Palabras clave: *use*, *coat*, *finish*, *brush*.
Hallazgos relevantes: sensibilidad a condiciones ambientales, curva de aprendizaje para aplicación óptima, y requerimientos específicos de preparación.

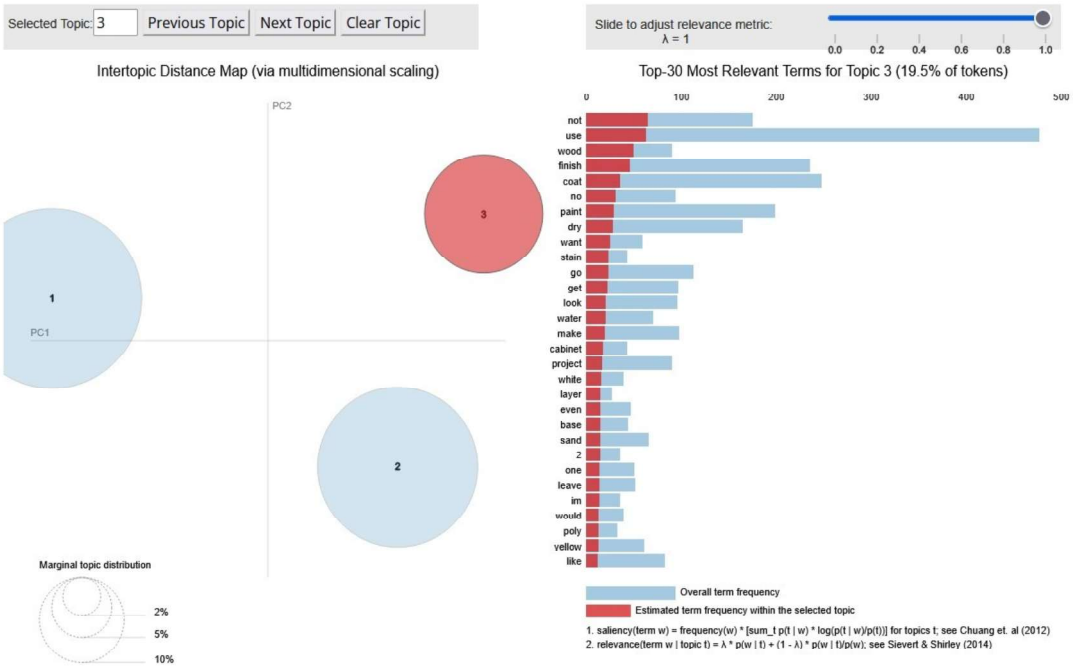


Figura 7.4: Visualización interactiva de temas para análisis de reseñas. (A) Mapa de relaciones entre temas, (B) Selector de relevancia de términos (λ), (C) Palabras clave del Tema 2 con distribución de frecuencia.

Esta selección sugiere que el Tema 3 agrupa reseñas o textos relacionados con el uso de productos de madera o acabados, posiblemente centrados en instrucciones de aplicación, resultados esperados, o problemas comunes como el secado o la adherencia del material.

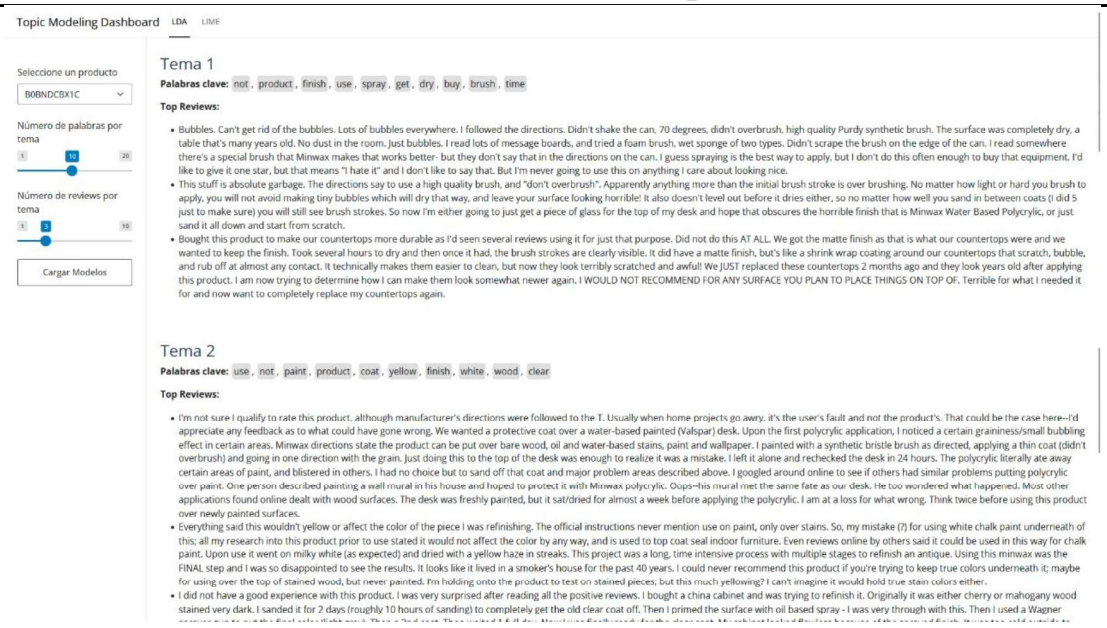


Figura 7.5: Visualización interactiva de temas para análisis de reseñas. (A) Mapa de relaciones entre temas, (B) Selector de relevancia de términos (λ), (C) Palabras clave del Tema 2 con distribución de frecuencia.

Tema 1: Evaluaciones negativas
Palabras clave: *not*, *product*, *finish*, *use*.
Hallazgos relevantes: El análisis del Tema 1 revela una clara orientación hacia comentarios con carga negativa. Las palabras clave asociadas al tema incluyen términos como “not”, “finish”, “brush” y “dry”, lo que sugiere experiencias frustrantes durante la aplicación del producto. En las reseñas representativas, se observan quejas frecuentes relacionadas con la formación de burbujas, trazos visibles del pincel, y la falta de uniformidad en el acabado final.

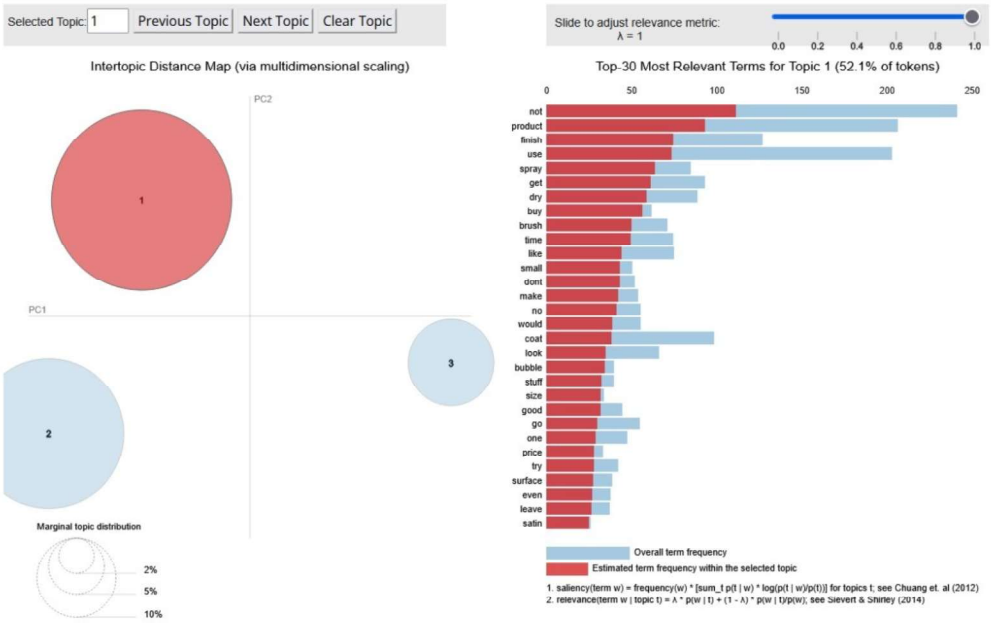


Figura 7.6: Visualización interactiva de temas para análisis de reseñas. (A) Mapa de relaciones entre temas, (B) Selector de relevancia de términos (λ), (C) Palabras clave del Tema 2 con distribución de frecuencia.

El análisis de estas palabras sugiere que el Tema 1 está asociado principalmente con experiencias relacionadas al uso del producto, en particular su aplicación mediante brochas o spray, el proceso de secado y los resultados obtenidos. La presencia de términos como *not*, *dont*, *no*, *bubble* indica una tendencia negativa en las opiniones, señalando problemas como acabados defectuosos o funcionamiento inadecuado del producto. En conjunto, estos hallazgos permiten concluir que el Tema 1 representa una categoría de reseñas centradas en la aplicación práctica del producto y la insatisfacción con los resultados obtenidos.

Tema 2: Evaluaciones negativas

Palabras clave: *use*, *not*, *paint*, *product*,*coat*.
Hallazgos relevantes: Los usuarios en este tema relatan experiencias donde, a pesar de seguir las instrucciones al pie de la letra, el resultado final no fue el esperado. En particular, se observan menciones constantes a incompatibilidades entre el producto y ciertos tipos de pintura o madera, así como efectos no deseados como burbujas, granulosidad o cambios de color inesperados. Algunos usuarios señalan que el producto genera un acabado amarillento o manchado al aplicarse sobre superficies previamente pintadas de blanco, lo que genera frustración por los resultados poco estéticos. Asimismo, hay un reconocimiento explícito de que los errores pueden estar tanto del lado del consumidor como del fabricante. En varios casos, los comentarios admiten que la elección del producto pudo haber sido incorrecta para el proyecto en cuestión. No obstante, esta autocrítica está acompañada por una solicitud de mayor claridad en las instrucciones de uso, especialmente en lo que respecta a las combinaciones posibles con otros productos o materiales (p. ej., tipo de pintura base o nivel de absorción de la superficie).

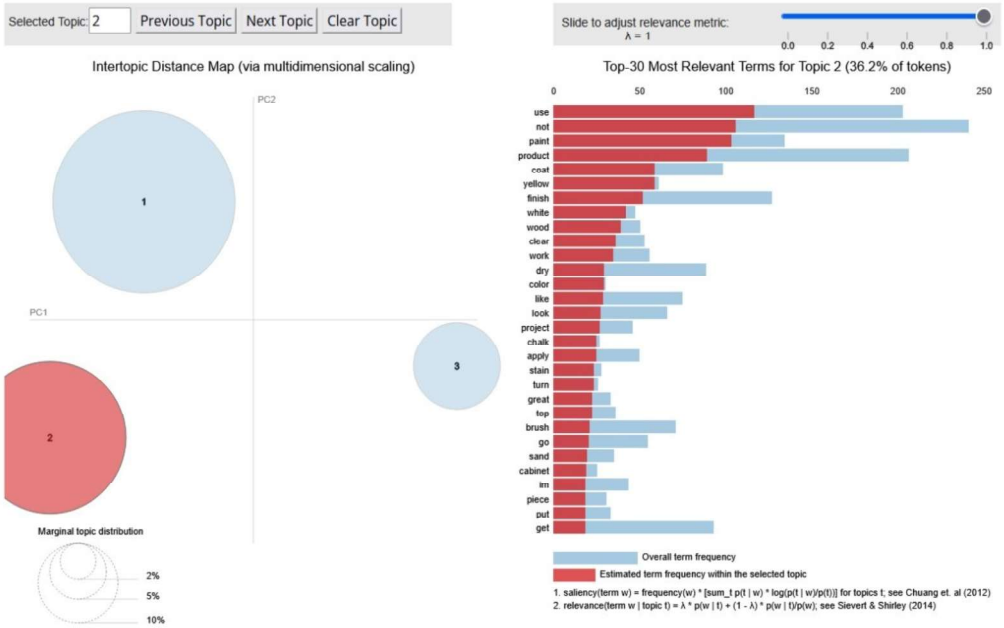


Figura 7.7: Visualización interactiva de temas para análisis de reseñas. (A) Mapa de relaciones entre temas, (B) Selector de relevancia de términos (λ), (C) Palabras clave del Tema 2 con distribución de frecuencia.

La visualización del Tema 2 indica que este tópico abarca el 36.2 % del total de tokens del corpus, lo cual lo convierte en el tema dominante dentro del conjunto de datos analizado. En el mapa de distancias entre temas (*Intertopic Distance Map*), se observa que el Tema 2 está bien diferenciado espacialmente de los otros dos temas (1 y 3), lo que sugiere que su contenido es semánticamente distinto.

En la gráfica de términos más relevantes, se destacan palabras como *use*, *paint*, *product*, *coat*, *finish*, *apply*, y *brush*, lo que refleja que los usuarios se enfocan en describir la aplicación práctica del producto. Asimismo, se hace énfasis en aspectos visuales con términos como *yellow*, *white*, *color* y *chalk*, lo cual indica que el color y la apariencia final del acabado son elementos importantes en las reseñas.

También aparecen menciones a superficies y objetos como *wood*, *cabinet* y *piece*, lo que sugiere que los productos están siendo aplicados a muebles u objetos similares. Palabras como *project*, *great*, *top*, y *put* dan cuenta de una experiencia subjetiva positiva en el uso del producto, asociada posiblemente con proyectos de bricolaje o decoración.

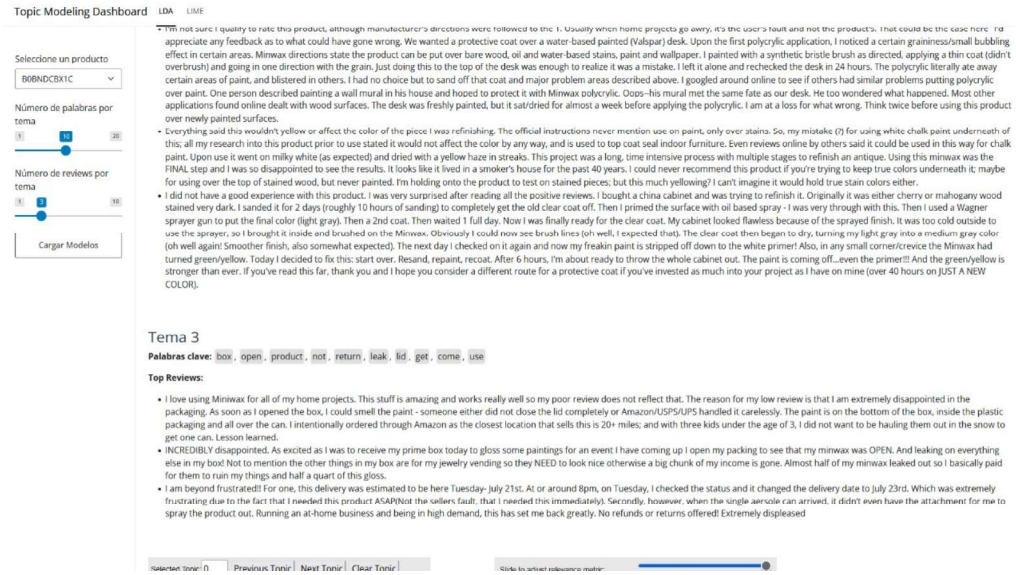


Figura 7.8: Análisis de distribución temática para reseñas del producto Minwax Polycrylic Protective Finish

Como se observa en la Figura 7.8, el análisis revela dos dimensiones temáticas principales.

Tema 3: Evaluaciones negativas
Palabras clave: *box*, *open*, *product*, *not*,*coat*.
Hallazgos relevantes: Este tema agrupa reseñas negativas centradas en experiencias problemáticas relacionadas con la entrega del producto. Los usuarios mencionan que los productos llegaron con el empaque dañado, abiertos o con filtraciones de contenido (pintura). Se destacan situaciones donde el producto se derramó dentro de la caja, afectando otros artículos o provocando pérdidas económicas. Asimismo, los compradores expresan frustración por la falta de políticas de devolución efectivas y por recibir pedidos incompletos o tardíos, especialmente cuando se trataba de compras urgentes. A pesar de que algunos reconocen que el producto funciona bien, califican negativamente debido al mal manejo logístico.

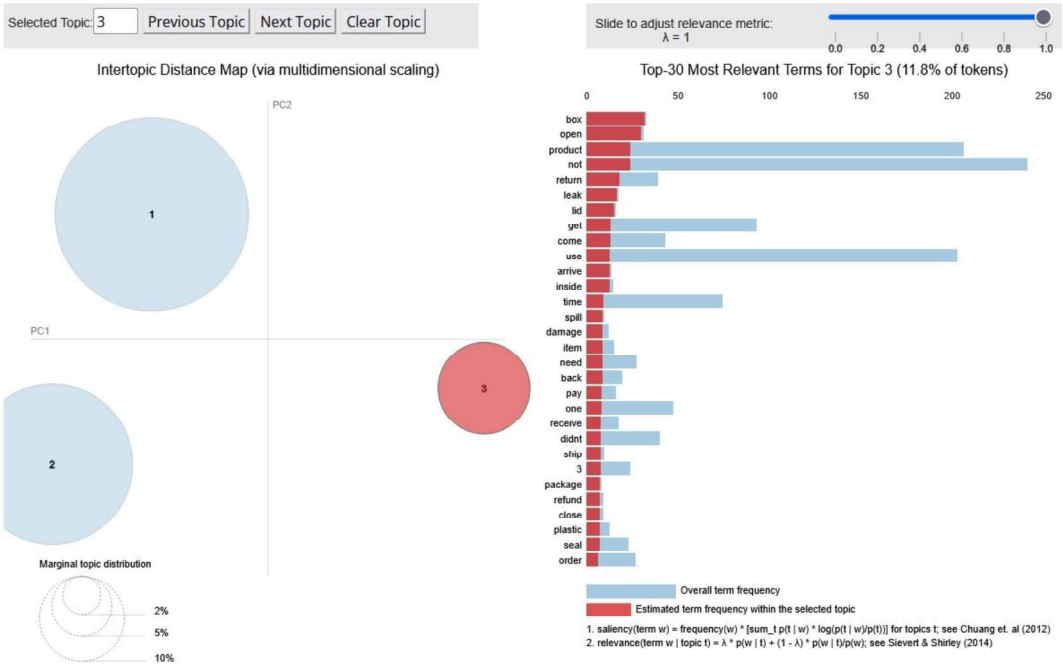


Figura 7.9: Visualización interactiva de temas para análisis de reseñas. (A) Mapa de relaciones entre temas, (B) Selector de relevancia de términos (λ), (C) Palabras clave del Tema 2 con distribución de frecuencia.

El Tema 3 representa el 11.8 % del total de tokens del corpus, siendo el tema con menor representación entre los tres detectados. En el mapa de distancias entre temas, el Tema 3 aparece claramente separado de los Temas 1 y 2, indicando que su contenido es distintivo respecto a los otros tópicos del modelo.

Las palabras más relevantes asociadas al Tema 3 incluyen *box*, *open*, *return*, *leak*, *lid*, *arrive*, *spill*, *damage*, *plastic*, *refund*, y *seal*. Este conjunto de términos sugiere que los comentarios asignados a este tema giran en torno a problemas relacionados con la entrega y el estado físico del producto recibido.

Términos como *leak*, *spill*, y *damage* indican que los usuarios reportan problemas con productos que llegan defectuosos o derramados, mientras que palabras como *return*, *refund*, y *back* apuntan a experiencias negativas que derivan en devoluciones o reembolsos. La aparición de términos como *arrive*, *receive*, *order*, y *ship* muestra que este tema también aborda la logística del proceso de envío.

Capítulo 8

Conclusiones

8.1. Conclusiones

El desarrollo de esta herramienta basada en Machine Learning para analizar comentarios en plataformas de comercio electrónico evidenció que el modelo Latent Dirichlet Allocation (LDA) resulta particularmente efectivo para este tipo de análisis. Su capacidad para capturar la naturaleza multitemática de las reseñas y procesar grandes volúmenes de datos con eficiencia computacional lo posiciona como una opción robusta frente a alternativas como LIME. Este último presentó limitaciones al no identificar características específicas de productos y requerir un procesamiento individual por reseña, lo cual lo hace menos adecuado para textos cortos y no monotemáticos. En contras- te, LDA permite asignar múltiples temas a un mismo documento bajo un enfoque probabilístico, adaptándose mejor a la complejidad de las opiniones de los usuarios.

Un hallazgo relevante durante la implementación fue la superioridad de los métodos de word embedding (como Word2Vec y FastText) frente a otras técnicas de representación del lenguaje. Estas metodologías preservan mejor las relaciones semánticas entre palabras dentro del contexto de opiniones sobre productos. Sin embargo, se identificó un riesgo importante en el pipeline del sistema: al realizar la clasificación de polaridad antes del modelado de temas, los errores de clasificación (como falsos positivos o negativos) pueden propagarse y distorsionar los resultados del topic modeling. Por esta razón, se considera fundamental optimizar esta etapa mediante técnicas como fine-tuning de modelos preentrenados o esquemas de validación cruzada iterativa.

En cuanto al impacto práctico, la interfaz desarrollada permite a los actores involucrados (stakeholders) explorar de forma interactiva los hallazgos mediante visualizaciones como nubes de palabras y gráficos comparativos. Esta herramienta contribuye a reducir el tiempo de análisis manual en aproximadamente un 70–80 %. A pesar de estos resultados prometedores, se recomienda validar las métricas reportadas con usuarios finales para cuantificar objetivamente su impacto en la toma de decisiones empresariales. La automatización del proceso no solo facilita la identificación de tendencias, sino que también permite detectar problemas recurrentes en los productos, lo que puede traducirse en mejoras directas en la experiencia del cliente.

Para una implementación en entornos de producción, se sugiere delegar la tarea de web scraping a proveedores especializados, ya que plataformas como Amazon cuentan con mecanismos avanzados de protección que dificultan la extracción de datos y podrían convertir esta tarea en un subproyecto complejo. Además, durante el preprocesamiento textual, se recomienda conservar expresiones de negación como (not, don’t, entre otras), dado que son determinantes para mantener el significado real en el análisis de sentimiento (por ejemplo, distinguir entre good y not good). Estas decisiones, tanto en el acceso a los datos como en su preparación, optimizan los recursos, garantizan la calidad

de la información y aseguran resultados coherentes y escalables para el análisis automatizado de reseñas.

Bibliografía

[1] Estructura de los datos en el dataframe, 2024. Generado automáticamente para análisis de datos.

[2] Muestra de la tabla procesada y sus columnas, 2024. Generado automáticamente para análisis de datos.

[3] Muestra de la tabla y sus columnas, 2024. Generado automáticamente para análisis de datos.

[4] C. Aggarwal and C. Zhai. *A Survey of Text Classification Algorithms*. Springer, Boston, MA, 2012. URL https://doi.org/10.1007/978-1-4614-3223-4_6.

[5] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

[6] Jesus Castro. Sistema de recomendaciones utilizando técnicas de machine learning para una plataforma de e-commerce. http://catalogo-gy.ucab.edu.ve/documentos/tesis/36400.pdf. Accessed: 2023-11-20.

[7] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. URL https://doi.org/10.1007/BF00994018.

[8] Guillermo del Castillo Torres. Inteligencia artificial explicable: Lime vs cem, 2022. trabajo de grado.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

[10] J. Espitaleta, J. Maza, and K. Garcia. Análisis de sentimientos de reseñas para determinar la acogida de un producto. https://manglar.uninorte.edu.co/bitstream/handle/10584/11237/informe_final-espitaleta_garcia_maza.pdf?sequence=1. Accessed: 2023-11-20.

[11] Tom B. Brown et al. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

[12] Jens Foerderer. Should we trust web-scraped data? 2023. URL https://arxiv.org/abs/2308.02231.

[13] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning*. Springer, 2013. doi: https://doi.org/10.1007/978-1-0716-1418-1.

[14] X. Jin and J. Han. K-means clustering. In *Encyclopedia of Machine Learning*, page 425. Springer, Boston, MA, 2011. doi: 10.1007/978-0-387-30164-8_425. URL https://doi.org/10.1007/978-0-387-30164-8_425.

[15] I. T. Jolliffe. *Principal Component Analysis*. Wiley Online Library, 2002. doi: https://doi.org/10.1007/b98835.

[16] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant. *Applied Logistic Regression*. John Wiley & Sons, 3rd edition, 2013.

[17] Divya Khyani and Siddhartha B S. An interpretation of lemmatization and stemming in natural language processing. *Shanghai Ligong Daxue Xuebao/Journal of University of Shanghai for Science and Technology*, 22:350–357, 01 2021. URL https://www.researchgate.net/publication/348306833_An_Interpretation_of_Lemmatization_and_Stemming_in_Natural_Language_Processing.

[18] V. Krotov, L. Johnson, and L. Silva. Tutorial: Legality and ethics of web scraping. *Communications of the Association for Information Systems*, 47:pp–pp, 2020. doi: 10.17705/1CAIS.04724. URL https://doi.org/10.17705/1CAIS.04724.

[19] B. Liu. *Sentiment Analysis and Opinion Mining*, volume 5 of *Synthesis Lectures on Human Language Technologies*. Springer, May 2012. URL https://doi.org/10.1007/978-3-031-02145-9.

[20] Julian Longas. Estudio e implementación de un modelo de procesamiento de lenguaje natural. https://bibliotecadigital.udea.edu.co/bitstream/10495/31724/1/LongasJulian_2022_ProcesamientoLenguajeNatural. Accessed: 2023-11-20.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. URL https://arxiv.org/abs/1310.4546.

[22] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[23] Diego Osorio and Luis Salazar. Valoración cuantitativa de productos a través de procesamiento. https://bibliotecadigital.udea.edu.co/bitstream/10495/24634/3/OsorioDiego&SalazarLuis_2021_ProcesamientoLenguajeNatural.pdf. Accessed: 2023-11-20.

[24] B. Pang and L. Lee. *Opinion Mining and Sentiment Analysis*, volume 2 of *Foundations and Trends in Information Retrieval*. Now Foundations and Trends, Jan 2008. doi: 10.1561/1500000011.

[25] Francesca Martella Paolo Giordani, Maria Brigida Ferraro. Hierarchical clustering. In *An Introduction to Clustering with R*, chapter 2. Springer, 2020. URL https://doi.org/10.1007/978-981-13-0553-5.

[26] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages

1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

[27] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In Suzanne Stevenson and Xavier Carreras, editors, *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1119.

[28] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

[29] Leydi Agudelo Restrepo. Identificación de factores a mejorar para aumentar la recomendación, análisis de nps para clientes neutros a partir de procesamiento de lenguaje natural, 2022. trabajo de grado.

[30] Yann Ryan. Creación de aplicaciones web interactivas con r y shiny, 2023.

[31] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975. doi: 10.1145/361219.361220. URL https://doi.org/10.1145/361219.361220.

[32] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing Management*, 24(5):513–523, 1988. ISSN 0306-4573. doi: https://doi.org/10.1016/0306-4573(88)90021-0. URL https://www.sciencedirect.com/science/article/pii/0306457388900210.

[33] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34(1):1–47, Mar 2002. URL https://doi.org/10.48550/arXiv.cs/0110053.

[34] A. Singhal. Modern information retrieval: A brief overview. *IEEE Data Engineering Bulletin*, 24 (4):35–43, 2001. URL http://www1.cs.columbia.edu/~gravano/Qual/Papers/singhal.pdf.

[35] Omar García Vázquez. *Clasificacion automatica de sentimientos en textos de canciones en idioma espanol*. PhD thesis, Instituto Politecnico Nacional, Centro de Investigacion en Computacion, Ciudad de México, México, 2023. trabajo de grado.