INSTITUTO POLITÉCNICO NACIONAL

Escuela Superior de Cómputo

Práctica 5

ANÁLISIS DE SENTIMIENTOS

Tecnologías de Lenguaje Natural

Alumno:
Escamilla Lazcano Saúl

Grupo: 5BV1

Carrera:
Ingeniería en Inteligencia Artificial

Profesor:
Ituriel Enrique Flores Estrada

Fecha de entrega:
21 de diciembre de 2025

Índice

Índice de figuras

1. Introducción

El Análisis de Sentimientos, también conocido como análisis de polaridad u opinión, constituye una de las aplicaciones más relevantes del Procesamiento de Lenguaje Natural (NLP) en la actualidad. Esta técnica permite extraer y clasificar las opiniones expresadas en textos, determinando si el sentimiento subyacente es positivo, negativo o neutral.

1.1. Contexto y Motivación

En la era digital, millones de personas expresan diariamente sus opiniones sobre productos, servicios, eventos y temas de actualidad a través de plataformas en línea. Las empresas, organizaciones y entidades gubernamentales requieren herramientas automatizadas capaces de procesar y analizar grandes volúmenes de texto para:

- Monitorear la reputación de marcas y productos
- Identificar tendencias en la opinión pública
- Mejorar la experiencia del cliente
- Tomar decisiones basadas en datos
- Detectar problemas y oportunidades de manera temprana

2. Metodología

2.1. Conjunto de Datos

2.1.1. Amazon Fine Food Reviews Dataset

El dataset utilizado en este proyecto contiene reseñas de productos alimenticios publicadas en Amazon. Este conjunto de datos es particularmente apropiado para análisis de sentimientos debido a:

- **Volumen:** Contiene más de 568,000 reseñas
- **Diversidad:** Incluye opiniones de múltiples productos y categorías
- **Etiquetado natural:** Las calificaciones numéricas proporcionan un etiquetado implícito
- **Variedad lingüística:** Textos con diferentes estilos, longitudes y complejidades

2.1.2. Fuente

El dataset está disponible públicamente en Kaggle:
https://www.kaggle.com/datasets/snap/amazon-fine-food-reviews

2.2. Flujo de Trabajo

El proyecto sigue un pipeline estructurado de procesamiento de datos:

1. **Adquisición de Datos:** Carga del dataset desde archivos CSV
2. **Análisis Exploratorio (EDA):** Caracterización y visualización de los datos
3. **Preprocesamiento:** Limpieza, transformación y balanceo de clases
4. **Normalización:** Aplicación de técnicas de limpieza de texto
5. **Análisis con Lexicons:** Clasificación basada en diccionarios
6. **Modelos de ML:** Entrenamiento y evaluación de algoritmos supervisados
7. **Redes Neuronales:** Implementación de modelos de deep learning
8. **Evaluación y Comparación:** Análisis comparativo de resultados

2.3. Protocolo de Validación

Para garantizar la integridad de los resultados y evitar el sobreajuste (*overfitting*), se implementaron las siguientes restricciones técnicas:

- **División del Dataset:** 65 % para entrenamiento, 35 % para prueba
- **Cross-Validation:** Validación cruzada de 5 particiones (5-fold)
- **Equilibrio de Clases:** Muestreo y balanceo para asegurar representatividad
- **Reproducibilidad:** Uso de semillas aleatorias fijas (random_state=42)

3. Desarrollo

3.1. Punto 1: Adquisición de Datos

La carga del dataset se realizó utilizando la biblioteca `pandas` de Python. A continuación se muestra el código utilizado para la adquisición de datos:

```python
import pandas as pd
import numpy as np

# Cargar el dataset
df = pd.read_csv('Reviews.csv')

# Visualizar primeras filas
print("Primeras 5 filas del dataset:")
print(df.head())

# Informacion general
print("\nInformacion del dataset:")
print(df.info())
```

Listing 1: Carga del dataset Amazon Fine Food Reviews

```
Dataset cargado exitosamente
Forma del dataset: (568454, 10)
Número de filas: 568,454
Número de columnas: 10

Primeras 5 filas del dataset:
```

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...

Figura 1: Carga inicial del dataset mostrando primeras observaciones

3.2. Punto 2: Análisis Exploratorio de Datos (EDA)

El análisis exploratorio de datos se realizó con el objetivo de comprender la estructura, características y distribución del conjunto de datos Amazon Fine Food Reviews antes de aplicar cualquier técnica de análisis de sentimientos. Este paso es fundamental ya que permite tomar decisiones informadas sobre el preprocesamiento necesario y los enfoques más apropiados para el análisis.

3.2.1. Caracterización Detallada de las 10 Dimensiones

A continuación se presenta la caracterización completa de cada una de las 10 dimensiones presentes en el dataset original:

1. Id - Identificador de la reseña

- **Tipo de variable:** Cualitativa nominal
- **Tipo de dato:** Entero (int64)
- **Propósito:** Identificación única de cada registro en el dataset
- **Formato:** Número entero secuencial
- **Valores únicos:** 568,454 (cada reseña tiene un ID único)
- **Valores nulos:** 0
- **Justificación para el análisis:** Esta columna no aporta información sobre el sentimiento de las reseñas, es únicamente un identificador administrativo. No se utilizará en el análisis de sentimientos.

2. ProductId - Identificador del producto

- **Tipo de variable:** Cualitativa nominal
- **Tipo de dato:** Cadena (object)
- **Propósito:** Identificación del producto específico que fue reseñado
- **Formato:** Cadena alfanumérica (ejemplo: "B001E4KFG0")
- **Valores únicos:** 74,258 productos diferentes
- **Valores nulos:** 0
- **Justificación para el análisis:** Aunque podría ser útil para análisis de sentimientos por producto, para el objetivo de esta práctica (clasificación general de sentimientos) no es necesario. Se puede prescindir de esta dimensión.

3. UserId - Identificador del usuario

- **Tipo de variable:** Cualitativa nominal
- **Tipo de dato:** Cadena (object)
- **Propósito:** Identificación del usuario que escribió la reseña
- **Formato:** Cadena alfanumérica (ejemplo: "A3SGXH7AUHU8GW")
- **Valores únicos:** 256,059 usuarios diferentes
- **Valores nulos:** 0
- **Justificación para el análisis:** No aporta información sobre el contenido o polaridad de las reseñas. Se puede prescindir de esta dimensión.

4. ProfileName - Nombre del perfil del usuario

- **Tipo de variable:** Cualitativa nominal
- **Tipo de dato:** Cadena (object)
- **Propósito:** Nombre público del usuario en Amazon
- **Formato:** Cadena de texto libre (ejemplo: "delmartian", "Natalia Corres")
- **Valores únicos:** 218,415 nombres diferentes
- **Valores nulos:** 26
- **Justificación para el análisis:** Esta información de metadatos no contribuye al análisis de sentimientos basado en el texto de las reseñas. Requeriría análisis adicional para determinar si existe alguna correlación entre nombres y sentimientos, lo cual está fuera del alcance de esta práctica.

5. HelpfulnessNumerator - Votos útiles recibidos

- **Tipo de variable:** Cuantitativa discreta
- **Tipo de dato:** Entero (int64)
- **Propósito:** Número de usuarios que marcaron la reseña como útil
- **Formato:** Número entero no negativo
- **Valores únicos:** 231 valores diferentes
- **Valores nulos:** 0
- **Rango:** De 0 a varios cientos
- **Justificación para el análisis:** Esta métrica mide la percepción de utilidad de la reseña por parte de otros usuarios, no el sentimiento expresado en ella. Aunque podría existir alguna correlación entre reseñas muy útiles y su polaridad, esto no es relevante para el objetivo principal de clasificar sentimientos basándose en el texto.

6. HelpfulnessDenominator - Total de votos sobre utilidad

- **Tipo de variable:** Cuantitativa discreta
- **Tipo de dato:** Entero (int64)
- **Propósito:** Número total de usuarios que evaluaron la utilidad de la reseña
- **Formato:** Número entero no negativo
- **Valores únicos:** 234 valores diferentes

- **Valores nulos:** 0
- **Justificación para el análisis:** Al igual que el numerador, esta dimensión mide engagement y utilidad percibida, no sentimiento. Se puede prescindir de ella.

7. Score - Calificación del producto (ESENCIAL)

- **Tipo de variable:** Cuantitativa discreta ordinal
- **Tipo de dato:** Entero (int64)
- **Propósito:** Evaluación numérica del producto realizada por el usuario
- **Formato:** Número entero del 1 al 5 (estrellas)
- **Valores únicos:** 5 valores posibles (1, 2, 3, 4, 5)
- **Valores nulos:** 0
- **Distribución observada:**
  - 1 estrella: 52,268 reseñas (9.2 %)
  - 2 estrellas: 29,769 reseñas (5.2 %)
  - 3 estrellas: 42,640 reseñas (7.5 %)
  - 4 estrellas: 80,655 reseñas (14.2 %)
  - 5 estrellas: 363,122 reseñas (63.9 %)
- **Justificación para el análisis: Esta es la dimensión más crítica del dataset** ya que proporciona la etiqueta de sentimiento implícita. Las calificaciones numéricas se convertirán en categorías de sentimiento (Negativo, Neutral, Positivo) según el esquema definido en la práctica. Es absolutamente necesaria para el análisis supervisado.

8. Time - Timestamp de publicación

- **Tipo de variable:** Cuantitativa continua
- **Tipo de dato:** Entero (int64) - Unix timestamp
- **Propósito:** Fecha y hora en que se publicó la reseña
- **Formato:** Timestamp Unix (segundos desde 1970-01-01)
- **Valores únicos:** 3,168 timestamps diferentes
- **Valores nulos:** 0
- **Justificación para el análisis:** La información temporal podría ser útil para análisis de tendencias de sentimientos a lo largo del tiempo, pero no es necesaria para la clasificación individual de sentimientos en esta práctica. Se puede prescindir de esta dimensión.

9. Summary - Resumen de la reseña

- **Tipo de variable:** Cualitativa nominal (texto libre)
- **Tipo de dato:** Cadena (object)
- **Propósito:** Título o síntesis breve de la opinión del usuario
- **Formato:** Texto libre en inglés
- **Valores únicos:** 295,742 sumarios diferentes
- **Valores nulos:** 27
- **Ejemplos:** "Good Quality Dog Food", "Not as Advertised", "Delight says it all"
- **Justificación para el análisis:** Esta dimensión contiene texto que expresa sentimiento de forma concisa. Podría utilizarse como complemento del texto principal, aunque para esta práctica se priorizará la columna Text que contiene información más completa. Es opcional su uso.

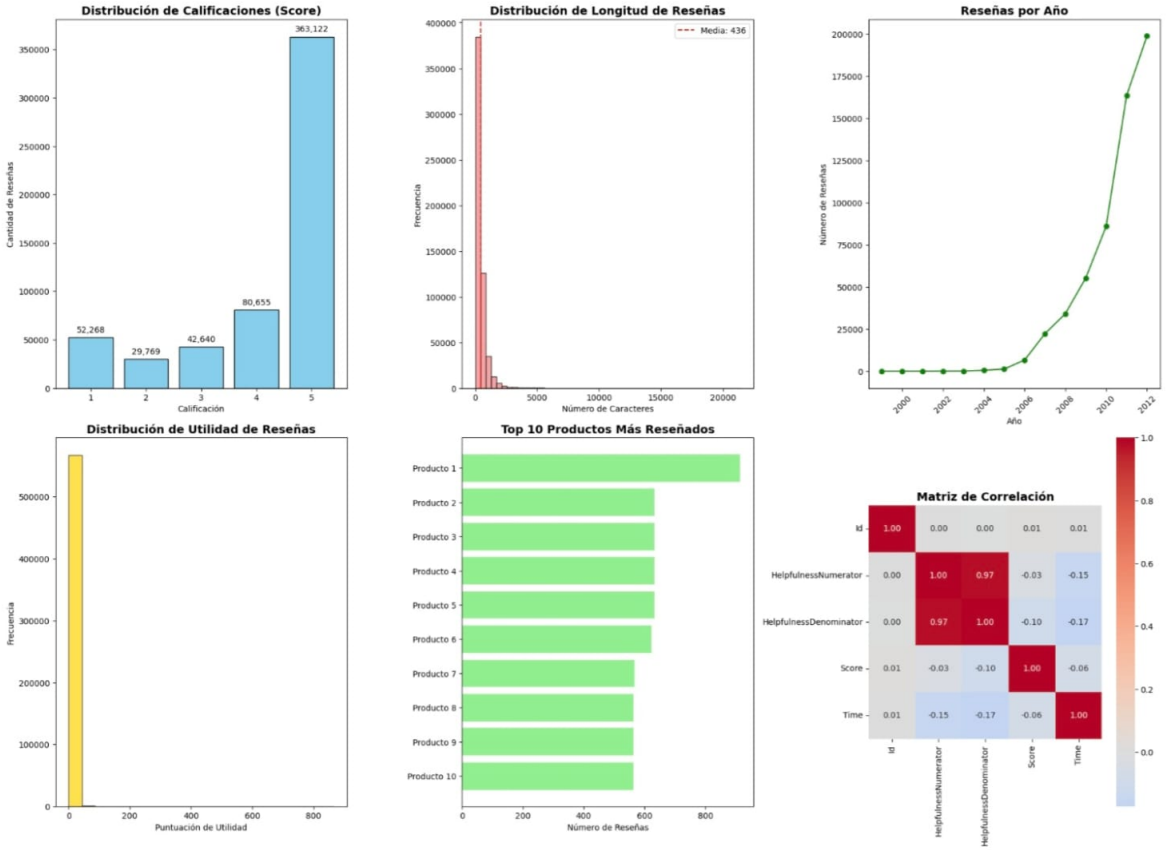
10. Text - Contenido completo de la reseña (ESENCIAL)

- **Tipo de variable:** Cualitativa nominal (texto libre)
- **Tipo de dato:** Cadena (object)
- **Propósito:** Opinión detallada del consumidor sobre el producto
- **Formato:** Texto libre en inglés
- **Valores únicos:** 393,579 textos diferentes
- **Valores nulos:** 0
- **Estadísticas de longitud:**
  - Longitud promedio: 436 caracteres
  - Longitud mediana: 302 caracteres
  - Longitud mínima: 12 caracteres
  - Longitud máxima: 21,409 caracteres
- **Justificación para el análisis: Esta es la dimensión principal del análisis** ya que contiene el texto completo de las reseñas donde los usuarios expresan sus opiniones, sentimientos y experiencias. Es absolutamente esencial para todos los enfoques de análisis de sentimientos (lexicons, ML y redes neuronales).

3.2.2. Visualizaciones del Análisis Exploratorio

Como se muestra en las figuras siguientes, el análisis exploratorio reveló varios patrones importantes:

- **Distribución de calificaciones:** Existe un sesgo significativo hacia calificaciones altas, con el 63.9 % de reseñas teniendo 5 estrellas. Este desbalance severo requiere tratamiento mediante técnicas de balanceo de clases.
- **Longitud de reseñas:** La mayoría de las reseñas tienen entre 100-500 caracteres, aunque existe gran variabilidad.
- **Patrones temporales:** Se observa un incremento en el volumen de reseñas a través del tiempo.
- **Productos más reseñados:** Algunos productos tienen cientos de reseñas mientras que otros tienen muy pocas.




```
============================================================
ESTADÍSTICAS ADICIONALES
============================================================

Distribución de Calificaciones:
  Calificación 1: 52,268 reseñas (9.2%)
  Calificación 2: 29,769 reseñas (5.2%)
  Calificación 3: 42,640 reseñas (7.5%)
  Calificación 4: 80,655 reseñas (14.2%)
  Calificación 5: 363,122 reseñas (63.9%)

Estadísticas de longitud de texto:
  Promedio: 436 caracteres
  Mediana: 302 caracteres
  Mínimo: 12 caracteres
  Máximo: 21409 caracteres
```

Figura 2: Visualizaciones del análisis exploratorio de datos

```
================================================================================
CARACTERIZACIÓN DETALLADA DE COLUMNAS
================================================================================

COLUMNA: Id
Tipo de dato: int64
Valores únicos: 568,454
Valores nulos: 0
Ejemplos: [1, 2, 3]
Propósito: Identificador único
Tipo de variable: Nominal
Formato: Entero o cadena
--------------------------------------------------

COLUMNA: ProductId
Tipo de dato: object
Valores únicos: 74,258
Valores nulos: 0
Ejemplos: ['B001E4KFG0', 'B00813GRG4', 'B000LQOCH0']
Propósito: Identificador único
Tipo de variable: Nominal
Formato: Entero o cadena
--------------------------------------------------

COLUMNA: UserId
Tipo de dato: object
Valores únicos: 256,059
Valores nulos: 0
Ejemplos: ['A3SGXH7AUHU8GW', 'A1D87F6ZCVE5NK', 'ABXLMWJIXXAIN']
Propósito: Identificador único
Tipo de variable: Nominal
Formato: Entero o cadena
--------------------------------------------------

COLUMNA: ProfileName
Tipo de dato: object
Valores únicos: 218,415
Valores nulos: 26
Ejemplos: ['delmartian', 'dll pa', 'Natalia Corres "Natalia Corres"']
Propósito: Requiere análisis adicional
Tipo de variable: A determinar
Formato: object
--------------------------------------------------

COLUMNA: HelpfulnessNumerator
Tipo de dato: int64
Valores únicos: 231
Valores nulos: 0
Ejemplos: [1, 0, 1]
Propósito: Métrica de utilidad de la reseña
Tipo de variable: Ordinal
Formato: Fracción o porcentaje
--------------------------------------------------

COLUMNA: HelpfulnessDenominator
Tipo de dato: int64
Valores únicos: 234
Valores nulos: 0
Ejemplos: [1, 0, 1]
Propósito: Métrica de utilidad de la reseña
Tipo de variable: Ordinal
Formato: Fracción o porcentaje
```

```
--------------------------------------------------

COLUMNA: Time
Tipo de dato: int64
Valores únicos: 3,168
Valores nulos: 0
Ejemplos: [1303862400, 1346976000, 1219017600]
Propósito: Información temporal
Tipo de variable: Ordinal
Formato: Timestamp o fecha
--------------------------------------------------

COLUMNA: Summary
Tipo de dato: object
Valores únicos: 295,742
Valores nulos: 27
Ejemplos: ['Good Quality Dog Food', 'Not as Advertised', '"Delight" says it all']
Propósito: Contenido textual de la reseña
Tipo de variable: Nominal
Formato: Cadena de texto
--------------------------------------------------

COLUMNA: Text
Tipo de dato: object
Valores únicos: 393,579
Valores nulos: 0
Ejemplos: ['I have bought several of the Vitality canned dog food products and have found them
all to be of good quality. The product looks more like a stew than a processed meat and it sme
lls better. My Labrador is finicky and she appreciates this product better than  most.', 'Prod
uct arrived labeled as Jumbo Salted Peanuts...the peanuts were actually small sized unsalted.
Not sure if this was an error or if the vendor intended to represent the product as "Jumbo".',
'This is a confection that has been around a few centuries.  It is a light, pillowy citrus gel
atin with nuts - in this case Filberts. And it is cut into tiny squares and then liberally coa
ted with powdered sugar.  And it is a tiny mouthful of heaven.  Not too chewy, and very flavor
ful.  I highly recommend this yummy treat.  If you are familiar with the story of C.S. Lewis\'
"The Lion, The Witch, and The Wardrobe" - this is the treat that seduces Edmund into selling o
ut his Brother and Sisters to the Witch.']
Propósito: Contenido textual de la reseña
Tipo de variable: Nominal
Formato: Cadena de texto
--------------------------------------------------
```

Figura 4: Detalles de las columnas del dataset - Parte 2

3.3. Punto 3: Preprocesamiento

El preprocesamiento es una etapa crítica que determina la calidad de los datos que alimentarán los modelos de análisis de sentimientos. Las decisiones tomadas en esta fase impactan directamente en el rendimiento final.

3.3.1. Selección y Justificación de Dimensiones

Basándose en el análisis exploratorio detallado en la sección anterior, se realizó la siguiente selección de columnas:

Dimensiones NECESARIAS (se conservan):

1. **Score:** Absolutamente esencial. Es la única fuente de etiquetas de sentimiento para el aprendizaje supervisado. Sin esta columna no es posible entrenar ni evaluar modelos de clasificación de sentimientos.

2. **Text:** Absolutamente esencial. Contiene el texto completo de las reseñas, que es el objeto principal del análisis. Todos los enfoques (lexicons, ML, redes neuronales) dependen de esta columna.

3. **Summary (opcional):** Aunque no se utilizó en esta implementación, podría proporcionar contexto adicional o usarse para técnicas de ensemble que combinen título y cuerpo de la reseña.

Dimensiones de las cuales se puede PRESCINDIR (se eliminan):

1. **Id:** Justificación: Es únicamente un identificador administrativo sin valor predictivo. No aporta información sobre el sentimiento expresado en la reseña.

2. **ProductId:** Justificación: Aunque podría ser útil para análisis de sentimientos específicos por producto, para el objetivo de esta práctica (clasificación general de sentimientos) introduce complejidad innecesaria sin beneficio claro. Además, con 74,258 productos diferentes, crear features categóricas resultaría en una explosión dimensional.

3. **UserId:** Justificación: Similar a ProductId, no aporta información sobre el contenido textual. Los sentimientos deben clasificarse basándose en lo que se dice, no en quién lo dice.

4. **ProfileName:** Justificación: Información de metadatos sin relación con el contenido de las reseñas. Además, presenta 26 valores nulos.

5. **HelpfulnessNumerator y HelpfulnessDenominator:** Justificación: Estas métricas miden la percepción de utilidad de la reseña por otros usuarios, no el sentimiento expresado. Aunque una reseña muy útil podría correlacionar con cierta polaridad, esta relación es indirecta y no es el foco del análisis.

6. **Time:** Justificación: La dimensión temporal podría ser relevante para análisis de tendencias, pero no para clasificación individual de sentimientos. El sentimiento de una reseña es independiente de cuándo fue publicada.

```
============================================================
ANÁLISIS DE COLUMNAS PARA SELECCIÓN
============================================================

Dataset original: (568454, 10)
Columnas disponibles (10):

 1. Id
    Tipo: int64
    Valores únicos: 568,454
    Valores nulos: 0
    Ejemplos: [1, 2, 3]

 2. ProductId
    Tipo: object
    Valores únicos: 74,258
    Valores nulos: 0
    Ejemplos: ['B001E4KFG0', 'B00813GRG4', 'B000LQOCH0']

 3. UserId
    Tipo: object
    Valores únicos: 256,059
    Valores nulos: 0
    Ejemplos: ['A3SGXH7AUHU8GW', 'A1D87F6ZCVE5NK', 'ABXLMWJIXXAIN']

 4. ProfileName
    Tipo: object
    Valores únicos: 218,415
    Valores nulos: 26
    Ejemplos: ['delmartian', 'dll pa', 'Natalia Corres "Natalia Corres"']

 5. HelpfulnessNumerator
    Tipo: int64
    Valores únicos: 231
    Valores nulos: 0
    Ejemplos: [1, 0, 1]

 6. HelpfulnessDenominator
    Tipo: int64
    Valores únicos: 234
    Valores nulos: 0
    Ejemplos: [1, 0, 1]

 7. Score
    Tipo: int64
    Valores únicos: 5
    Valores nulos: 0
    Ejemplos: [5, 1, 4]

 8. Time
    Tipo: int64
    Valores únicos: 3,168
    Valores nulos: 0
    Ejemplos: [1303862400, 1346976000, 1219017600]

 9. Summary
    Tipo: object
    Valores únicos: 295,742
    Valores nulos: 27
    Ejemplos: ['Good Quality Dog Food', 'Not as Advertised', '"Delight" says it all']

10. Text
    Tipo: object
    Valores únicos: 393,579
    Valores nulos: 0
    Ejemplos: ['I have bought several of the Vitality canned dog food products and have found
them all to be of good quality. The product looks more like a stew than a processed meat and i
t smells better. My Labrador is finicky and she appreciates this product better than  most.',
'Product arrived labeled as Jumbo Salted Peanuts...the peanuts were actually small sized unsal
ted. Not sure if this was an error or if the vendor intended to represent the product as "Jumb
o".', 'This is a confection that has been around a few centuries.  It is a light, pillowy citr
us gelatin with nuts - in this case Filberts. And it is cut into tiny squares and then liberal
ly coated with powdered sugar.  And it is a tiny mouthful of heaven.  Not too chewy, and very
flavorful.  I highly recommend this yummy treat.  If you are familiar with the story of C.S. L
ewis\' "The Lion, The Witch, and The Wardrobe" - this is the treat that seduces Edmund into se
lling out his Brother and Sisters to the Witch.']
```

Figura 5: Selección y filtrado de columnas relevantes - Parte 1

```
============================================================
SELECCIÓN DE COLUMNAS RELEVANTES
============================================================
Columnas seleccionadas para el análisis de sentimientos:
  - Text: Texto completo de la reseña (principal para análisis)
  - Score: Calificación numérica (1-5) para crear etiquetas de sentimiento
  - Summary: Resumen de la reseña (texto adicional para análisis)
============================================================
PREPROCESAMIENTO DEL DATASET
============================================================

Dataset original: (568454, 10)
Columnas seleccionadas: ['Text', 'Score', 'Summary']
Dataset filtrado: (568454, 3)
Filas eliminadas por valores nulos: 27

Dataset final después de limpieza: (568427, 3)

3.2 CONVERSIÓN DE CALIFICACIONES A SENTIMIENTOS
--------------------------------------------------
Distribución original de calificaciones:
  Calificación 1: 52,268 (9.2%)
  Calificación 2: 29,744 (5.2%)
  Calificación 3: 42,638 (7.5%)
  Calificación 4: 80,655 (14.2%)
  Calificación 5: 363,122 (63.9%)

Distribución de sentimientos:
  Positivo: 443,777 (78.1%)
  Negativo: 82,012 (14.4%)
  Neutral: 42,638 (7.5%)
```

Figura 6: Selección y filtrado de columnas relevantes - Parte 2

3.3.2. Conversión de Calificaciones a Sentimientos

La conversión de calificaciones numéricas a categorías de sentimiento se realizó siguiendo el esquema especificado en la práctica:

Cuadro 1: Esquema de conversión de calificación a sentimiento

Calificación	**Sentimiento**	**Justificación**
1-2 estrellas	Negativo	Insatisfacción clara del cliente
3 estrellas	Neutral	Opinión ambivalente o neutra
4-5 estrellas	Positivo	Satisfacción y recomendación

Justificación del esquema de conversión:

- **Calificaciones 1-2 (Negativo):** Estas calificaciones bajas indican claramente experiencias negativas. Los usuarios que califican con 1 o 2 estrellas típicamente expresan problemas, quejas o decepción con el producto.
- **Calificación 3 (Neutral):** La calificación intermedia de 3 estrellas representa una zona gris donde el usuario no está completamente satisfecho ni completamente insatisfecho. Estas reseñas suelen contener comentarios mixtos o expresan que el producto es "aceptable" pero no excepcional.

- **Calificaciones 4-5 (Positivo):** Las calificaciones altas reflejan satisfacción. Los usuarios que califican con 4 o 5 estrellas generalmente recomiendan el producto y expresan opiniones favorables.

```python
def convertir_score_a_sentimiento(score):
    """
    Convierte calificacion numerica a categoria de sentimiento
    """
    if score <= 2:
        return 'Negativo'
    elif score == 3:
        return 'Neutral'
    else:  # score >= 4
        return 'Positivo'

# Aplicar conversion
df['Sentiment'] = df['Score'].apply(convertir_score_a_sentimiento)
```

Listing 2: Conversión de calificaciones a sentimientos

3.3.3. Balance de Clases - Problema y Solución

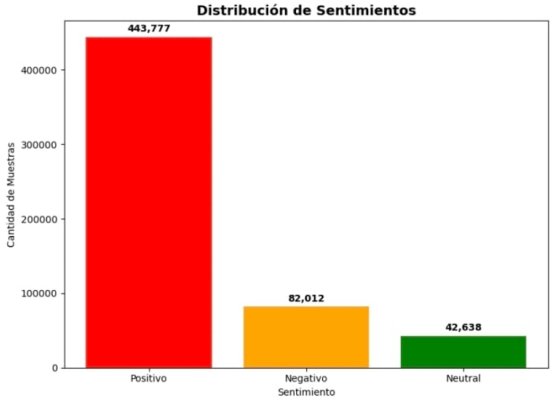
Problema Identificado: Como se observa en la Figura 7, el dataset original presenta un desbalance severo entre las clases de sentimiento:

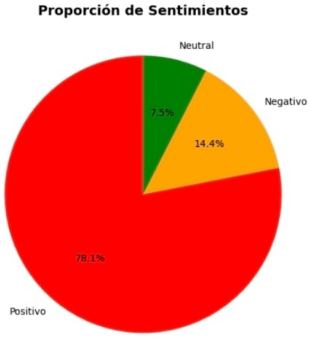
- Positivo: 443,777 muestras (78.1 %)
- Negativo: 82,012 muestras (14.4 %)
- Neutral: 42,638 muestras (7.5 %)
- Ratio mayor/menor: 10.41:1

Este desbalance es problemático porque:

1. **Sesgo del modelo:** Los algoritmos de aprendizaje de máquina tienden a favorecer la clase mayoritaria. Un modelo entrenado en estos datos desbalanceados podría lograr 78 % de accuracy simplemente prediciendo siempre "Positivo", sin aprender realmente a distinguir sentimientos.
2. **Métricas engañosas:** El accuracy global no reflejaría el verdadero rendimiento en clases minoritarias. El modelo podría tener excelente rendimiento en positivos pero muy pobre en negativos y neutrales.
3. **Pobre generalización:** El modelo no aprendería patrones representativos de todas las clases, limitando su utilidad práctica.
4. **Evaluación poco confiable:** Las métricas como precision, recall y F1-score estarían distorsionadas, especialmente para la clase neutral que representa solo el 7.5 % de los datos.

```
============================================================
3.3 ANÁLISIS DE BALANCE DE CLASES
============================================================

Total de muestras: 568,427

Distribución de sentimientos:
  Positivo: 443,777 muestras (78.1%)
  Negativo: 82,012 muestras (14.4%)
  Neutral: 42,638 muestras (7.5%)
```






```
Análisis de balance:
  Clase mayoritaria: Positivo (443,777 muestras)
  Clase minoritaria: Neutral (42,638 muestras)
  Ratio mayor/menor: 10.41

DESBALANCE DETECTADO (ratio > 3)
  Se recomienda aplicar técnicas de balanceado

Aplicando balanceado de clases...

3.4 BALANCEADO DE CLASES (Método: UNDERSAMPLE)
----------------------------------------------------
Distribución original:
  Positivo: 443,777
  Negativo: 82,012
  Neutral: 42,638

Distribución después del balanceado:
  Negativo: 42,638
  Positivo: 42,638
  Neutral: 42,638

Tamaño original: 568,427 muestras
Tamaño balanceado: 127,914 muestras

Verificando balance después del procesamiento:
```

Figura 7: Distribución original de clases (desbalanceada)

Solución Implementada: Undersampling Se aplicó la técnica de *undersampling* (submuestreo) de la clase mayoritaria para equilibrar las tres categorías:

1. Se identificó la clase con menor cantidad de muestras (Neutral con 42,638)
2. Se tomó una muestra aleatoria de 42,638 observaciones de cada una de las otras dos clases
3. Se combinaron las tres muestras balanceadas
4. Se mezclaron aleatoriamente para evitar ordenamiento por clase

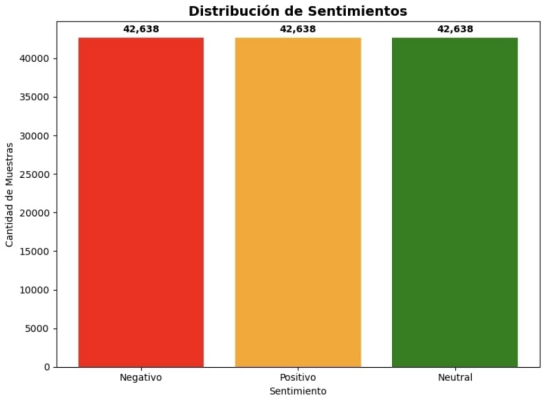
```python
from imblearn.under_sampling import RandomUnderSampler

# Determinar la clase con menor cantidad de muestras
min_samples = df['Sentiment'].value_counts().min()

# Muestrear equitativamente cada clase
df_balanced = pd.concat([
    df[df['Sentiment'] == 'Negativo'].sample(min_samples, random_state=42),
    df[df['Sentiment'] == 'Neutral'].sample(min_samples, random_state=42),
    df[df['Sentiment'] == 'Positivo'].sample(min_samples, random_state=42)
])

# Mezclar aleatoriamente
df_balanced = df_balanced.sample(frac=1, random_state=42).reset_index(drop=True)

print(f"Dataset balanceado: {df_balanced.shape}")
print(df_balanced['Sentiment'].value_counts())
```

Listing 3: Balanceo de clases mediante undersampling

Justificación Técnica del Balanceo:

- **Prevención de sesgo:** Al tener exactamente la misma cantidad de ejemplos de cada clase (33.33 % cada una), se elimina el sesgo hacia positivos que presentaba el dataset original.
- **Mejora en detección de clases minoritarias:** Con representación equitativa, los modelos aprenderán patrones de las tres clases con igual importancia, mejorando especialmente la detección de reseñas neutrales y negativas.
- **Métricas más representativas:** El accuracy, precision, recall y F1-score reflejarán ahora el verdadero rendimiento del modelo en todas las clases, no solo en la mayoritaria.
- **Facilita el aprendizaje equilibrado:** Los algoritmos de machine learning optimizan funciones de pérdida que, en datos balanceados, dan igual importancia a errores en cualquier clase.

```
============================================================
3.3 ANÁLISIS DE BALANCE DE CLASES
============================================================

Total de muestras: 127,914

Distribución de sentimientos:
  Negativo: 42,638 muestras (33.3%)
  Positivo: 42,638 muestras (33.3%)
  Neutral: 42,638 muestras (33.3%)
```




```
Análisis de balance:
  Clase mayoritaria: Negativo (42,638 muestras)
  Clase minoritaria: Negativo (42,638 muestras)
  Ratio mayor/menor: 1.00

CLASES RELATIVAMENTE BALANCEADAS (ratio ≤ 3)
  No es necesario aplicar balanceado
```

Figura 8: Distribución balanceada de clases después del preprocessing

Como se muestra en la Figura 8, el dataset final balanceado contiene:

- Total de observaciones: 127,914 (33.33 % Negativo + 33.33 % Neutral + 33.33 % Positivo)
- Reducción del dataset original: 77.5 % (de 568,454 a 127,914)
- Ratio mayor/menor: 1.00:1 (perfecto balance)

Trade-off aceptado:
Aunque el undersampling reduce significativamente el tamaño del dataset (perdiendo el 77.5 % de los datos), los beneficios superan esta pérdida:

- Se conservan 127,914 observaciones, que es suficiente para entrenar modelos robustos
- Se elimina el sesgo que haría que los modelos fueran inútiles para aplicaciones prácticas
- El tiempo de entrenamiento se reduce proporcionalmente
- La validación cruzada será más confiable

Alternativas como oversampling (sobremuestreo) de clases minoritarias o técnicas como SMOTE podrían explorarse en trabajos futuros, pero el undersampling es simple, efectivo y apropiado para este volumen de datos.

3.4. Punto 4: Normalización de Datos - Justificación de Estrategias Diferenciadas

La normalización de texto es una etapa crítica en el análisis de sentimientos, pero **no existe una única estrategia óptima para todos los enfoques**. Cada paradigma (lexicons, machine learning, redes neuronales) tiene requerimientos diferentes que determinan las técnicas de limpieza más apropiadas. A continuación se justifica detalladamente la estrategia de normalización aplicada para cada enfoque.

3.4.1. Estrategia de Normalización para Lexicons

Objetivo: Preservar palabras completas para permitir matching exacto con diccionarios.

Técnicas aplicadas:

1. Conversión a minúsculas
2. Remoción de HTML tags (usando BeautifulSoup)
3. **Preservación de puntuación** (crítico)
4. Tokenización básica
5. **NO se eliminan stopwords**
6. **NO se aplica stemming/lemmatización**

Justificación detallada:
Los diccionarios léxicos como Opinion Lexicon, SentiWordNet y Harvard IV-4 funcionan mediante búsqueda de coincidencias exactas de palabras en sus bases de datos. Por lo tanto:

- **Preservación de palabras completas:** Los lexicons contienen palabras en su forma completa (ejemplo: "good", "excellent", "terrible"). Si aplicáramos stemming, convertiríamos "excellent" a "excel", que no existiría en el diccionario, resultando en falsos negativos.
- **Conservación de stopwords:** Aunque palabras como "not", "but", "however" son stopwords, muchas son críticas para el análisis léxico. Por ejemplo, "not" puede modificar completamente el sentimiento ("good" vs "not good"). Algunos lexicons incluyen estas palabras modificadoras.
- **Preservación parcial de puntuación:** Signos como "!" pueden intensificar sentimientos. Algunos lexicons consideran este contexto.
- **Limpieza mínima:** Solo se remueve contenido claramente irrelevante (HTML tags) para evitar interferencias, pero se mantiene la mayor cantidad posible de información textual.

Resultado de esta estrategia:
Como se muestra en la Figura 9, la normalización para lexicons redujo mínimamente la longitud promedio del texto:

- Longitud promedio original: 472.8 caracteres
- Longitud promedio normalizada: 454.2 caracteres
- Reducción: 3.9 % (mínima, como se esperaba)

Esta reducción mínima confirma que la estrategia preserva prácticamente todo el contenido textual, permitiendo maximizar el matching con los diccionarios.

3.4.2. Estrategia de Normalización para Machine Learning (TF-IDF)

Objetivo: Reducir dimensionalidad y eliminar ruido para crear vectores TF-IDF más informativos.
Técnicas aplicadas:

1. Conversión a minúsculas
2. Remoción de HTML tags
3. **Eliminación completa de puntuación**
4. **Remoción de números**
5. **Eliminación agresiva de stopwords** (usando lista NLTK)
6. Tokenización
7. **Lematización** (usando WordNetLemmatizer)

Justificación detallada:
Los algoritmos de machine learning con representación TF-IDF se benefician de vocabulario normalizado y reducido porque:

- **Reducción de dimensionalidad:** TF-IDF crea una feature por cada palabra única. Sin normalización, "excellent", "Excellent" y "EXCELLENT" serían tres features diferentes representando el mismo concepto. La lematización reduce aún más la dimensionalidad agrupando variantes morfológicas ("running", "runs", "ran" → "run").
- **Eliminación de stopwords justificada:** En TF-IDF, las stopwords tienen poca capacidad discriminativa porque aparecen en casi todos los documentos (su IDF es bajo). Sin embargo, ocupan espacio en el vocabulario. Al eliminarlas, se permite que el modelo se enfoque en palabras con mayor poder predictivo.
- **Remoción de puntuación y números:** Los signos de puntuación no aportan valor como features independientes en modelos de bag-of-words. Los números generalmente no son palabras de sentimiento.

- **Normalización agresiva justificada:** A diferencia de los lexicons que necesitan matching exacto, los modelos de ML aprenden pesos para cada feature. Es mejor tener pocas features informativas que muchas features redundantes.

Resultado de esta estrategia:
Como se muestra en la Figura 9, la normalización para ML redujo significativamente la longitud del texto:

- Longitud promedio original: 472.8 caracteres
- Longitud promedio normalizada: 275.3 caracteres
- Reducción: 41.8 % (reducción sustancial como se esperaba)
- Longitud mínima: 0 caracteres (algunas reseñas quedaron vacías después de remover stopwords)

Esta reducción significativa demuestra que la estrategia está funcionando correctamente: se eliminó contenido de baja señal (stopwords, puntuación) manteniendo las palabras con mayor carga semántica.

Impacto en el vocabulario TF-IDF:
La normalización agresiva resultó en:

- Vocabulario final: 5,000 términos más frecuentes (configurado en TfidfVectorizer)
- Densidad de la matriz: 99.66 % sparse (solo 0.34 % de valores no-cero)
- Top features identificadas: “like”, “taste”, “coffee”, “product”, “flavor”, “good”

Estas son precisamente palabras con alta capacidad discriminativa para sentimientos en reseñas de comida.

3.4.3. Estrategia de Normalización para Redes Neuronales

Objetivo: Preservar estructura secuencial del texto mientras se remueve ruido.
Técnicas aplicadas:

1. Conversión a minúsculas
2. Remoción de HTML tags
3. **Preservación parcial de puntuación importante**
4. Tokenización para secuencias
5. **NO se eliminan stopwords** (crítico para contexto)
6. **NO se aplica lemmatización** (las variaciones morfológicas contienen información)

Justificación detallada:
Las redes neuronales, especialmente arquitecturas recurrentes como LSTM, aprenden representaciones contextuales donde la secuencia de palabras importa. Por lo tanto:

- **Preservación de stopwords justificada:** A diferencia de TF-IDF, las redes neuronales pueden aprender que secuencias como "not good" tienen significado diferente a "good". Las stopwords proporcionan contexto gramatical crucial. Por ejemplo:
  - "I don't like it" vs "I like it" - la diferencia es solo la stopword "don't"
  - "It's okay but not great" - "but" invierte el sentimiento
- **Preservación de variaciones morfológicas:** Los embeddings pueden capturar que "running" y "run" son similares pero no idénticos. La lemmatización eliminaría esta información sutil que las redes pueden aprovechar.
- **Conservación de estructura:** Las LSTM procesan secuencias. Destruir la estructura gramatical mediante normalización excesiva reduce su ventaja sobre métodos de bag-of-words.
- **Limpieza moderada:** Se remueve solo ruido claro (HTML) manteniendo la mayor cantidad posible de información contextual y estructural.
- **Capacidad de aprendizaje:** Las redes neuronales tienen capacidad suficiente para aprender qué ignorar (si las stopwords no son útiles, aprenderán embeddings con pesos bajos). No es necesario pre-filtrar agresivamente.

Resultado de esta estrategia:
Como se muestra en la Figura 9:

- Longitud promedio original: 472.8 caracteres
- Longitud promedio normalizada: 457.4 caracteres
- Reducción: 3.3 % (mínima, similar a lexicons)

Esta reducción mínima es intencional y correcta: se preserva casi todo el contenido textual para que la red neuronal pueda aprender patrones contextuales complejos.
Impacto en secuencias:

- Vocabulario de tokenización: 10,000 palabras más frecuentes
- Longitud máxima de secuencia: 200 tokens (con padding)
- Longitud promedio de secuencias (antes de padding): 90.1 tokens
- Percentil 95 de longitudes: 243 tokens

La longitud máxima de 200 tokens captura el 95 % de las reseñas completas, permitiendo que la red neuronal procese prácticamente todo el contexto disponible.

```
============================================================
4. APLICANDO NORMALIZACIONES DE TEXTO
============================================================
Normalizando para análisis con lexicons...
Normalizando para machine learning...
Normalizando para redes neuronales...

============================================================
EJEMPLOS DE NORMALIZACIÓN
============================================================
Texto original:
  After reading the ingredients (and being familiar with Paul Newman's great-tasting salad dre
ssings), I was excited about feeding this dog food to my older rescue doberman.<br /><br />I w
as surprised w...

Para lexicons:
  after reading the ingredients and being familiar with paul newman s great tasting salad dres
sings i was excited about feeding this dog food to my older rescue doberman br br i was surpri
sed when i ope...

Para ML:
  reading ingredient familiar paul newman great tasting salad dressing excited feeding dog foo
d older rescue doberman surprised opened first truly looked like eaten passed another dog inte
stine sorry gr...

Para redes neuronales:
  after reading the ingredients and being familiar with paul newman s great tasting salad dres
sings i was excited about feeding this dog food to my older rescue doberman br br i was surpri
sed when i ope...

============================================================
ESTADÍSTICAS DE LONGITUD DESPUÉS DE NORMALIZACIÓN
============================================================

Original:
  Promedio: 472.8 caracteres
  Mediana: 333.0 caracteres
  Mínimo: 12 caracteres
  Máximo: 21409 caracteres

Lexicons:
  Promedio: 454.2 caracteres
  Mediana: 321.0 caracteres
  Mínimo: 12 caracteres
  Máximo: 20517 caracteres

ML:
  Promedio: 275.3 caracteres
  Mediana: 192.0 caracteres
  Mínimo: 0 caracteres
  Máximo: 14268 caracteres

Redes Neuronales:
  Promedio: 457.4 caracteres
  Mediana: 323.0 caracteres
  Mínimo: 12 caracteres
  Máximo: 20630 caracteres

Dataset normalizado creado: (127914, 7)
```

Figura 9: Ejemplo de aplicación de diferentes estrategias de normalización

3.4.4. Comparación y Análisis de las Tres Estrategias

La siguiente tabla resume el impacto diferencial de cada estrategia de normalización:

Cuadro 2: Comparación de estrategias de normalización

Métrica	Lexicons	ML (TF-IDF)	Redes Neuronales
Long. promedio (car.)	454.2	275.3	457.4
Reducción (%)	3.9	41.8	3.3
Preserva stopwords	Sí	No	Sí
Aplica lematización	No	Sí	No
Preserva puntuación	Parcial	No	Parcial
Objetivo principal	Matching exacto	Reducir dimensión	Preservar contexto

Conclusiones sobre normalización:

1. **No existe una normalización universal óptima:** La mejor estrategia depende completamente del enfoque de análisis que se utilizará posteriormente.
2. **Lexicons requieren preservación:** Necesitan palabras completas y exactas para funcionar correctamente.
3. **ML se beneficia de agresividad:** TF-IDF mejora con vocabulario reducido y normalizado.
4. **Redes neuronales necesitan contexto:** Aprenden mejor con texto estructurado que conserve secuencias y relaciones gramaticales.
5. **La justificación es crítica:** Cada decisión de normalización debe basarse en cómo el método posterior utilizará los datos, no en reglas generales.

```python
import re
from bs4 import BeautifulSoup
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize

def limpiar_texto_lexicons(texto):
    """Normalizacion minima para analisis con lexicons"""
    soup = BeautifulSoup(texto, "html.parser")
    texto = soup.get_text()
    texto = texto.lower()
    return texto

def limpiar_texto_ml(texto):
    """Normalizacion agresiva para ML con TF-IDF"""
    # Remover HTML
```

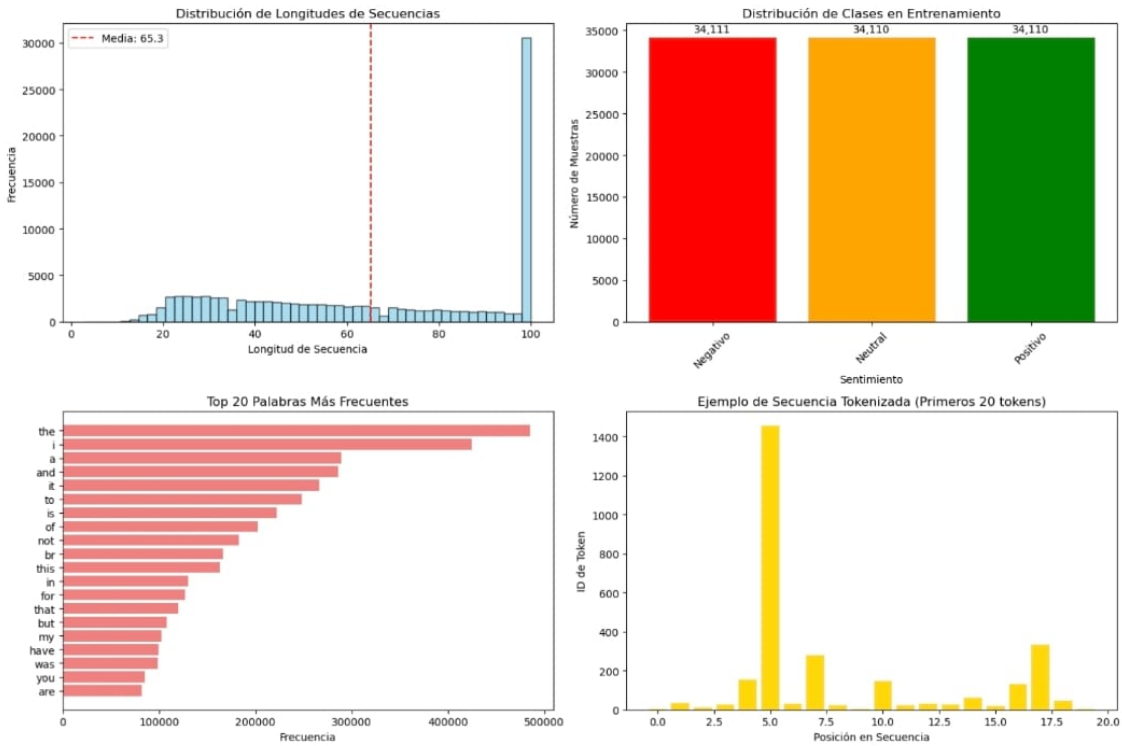
```
    soup = BeautifulSoup(texto, "html.parser")
    texto = soup.get_text()

    # Convertir a minusculas
    texto = texto.lower()

    # Remover puntuacion y numeros
    texto = re.sub(r'[^a-zA-Z\s]', '', texto)

    # Tokenizar
    tokens = word_tokenize(texto)

    # Remover stopwords y lematizar
    stop_words = set(stopwords.words('english'))
    lemmatizer = WordNetLemmatizer()

    tokens = [lemmatizer.lemmatize(token) for token in tokens
              if token not in stop_words and len(token) > 2]

    return ' '.join(tokens)

def limpiar_texto_nn(texto):
    """Normalizacion moderada para redes neuronales"""
    soup = BeautifulSoup(texto, "html.parser")
    texto = soup.get_text()
    texto = texto.lower()
    return texto

# Aplicar normalizaciones
df_balanced['Text_Lexicons'] = df_balanced['Text'].apply(
    limpiar_texto_lexicons)
df_balanced['Text_ML'] = df_balanced['Text'].apply(limpiar_texto_ml)
df_balanced['Text_NN'] = df_balanced['Text'].apply(limpiar_texto_nn)
```

Listing 4: Funciones de normalización implementadas

```
============================================================
6. PREPARACIÓN DE DATOS PARA MACHINE LEARNING
============================================================
Datos disponibles para ML: 127,912 muestras

Distribución de sentimientos en datos ML:
  Negativo: 42,638 (33.3%)
  Neutral: 42,638 (33.3%)
  Positivo: 42,636 (33.3%)

Mapeo de etiquetas:
  Negativo -> 0
  Neutral -> 1
  Positivo -> 2

División de datos:
  Entrenamiento: 83,142 muestras (65.0%)
  Prueba: 44,770 muestras (35.0%)

Aplicando vectorización TF-IDF...
Características TF-IDF generadas: 10,000
Densidad de la matriz (entrenamiento): 0.0034
Densidad de la matriz (prueba): 0.0034

Ejemplos de características generadas:
  Primeras 10: ['ability', 'able', 'able buy', 'able drink', 'able eat', 'able make', 'able or
der', 'able purchase', 'able use', 'absolute']
  Últimas 10: ['zico', 'zinc', 'zing', 'zip', 'zip lock', 'ziploc', 'ziplock', 'zipper', 'zuk
e', 'zukes']

============================================================
ESTADÍSTICAS DE VECTORIZACIÓN TF-IDF
============================================================

Dimensionalidad:
  Muestras de entrenamiento: 83,142
  Muestras de prueba: 44,770
  Características (vocabulario): 10,000

Sparsity (porcentaje de ceros):
  Entrenamiento: 99.66%
  Prueba: 99.66%

Top 20 características por puntuación TF-IDF:
   1. like: 2248.913
   2. taste: 2162.886
   3. coffee: 2113.150
   4. product: 1996.376
   5. flavor: 1843.634
   6. good: 1831.539
   7. tea: 1679.755
   8. love: 1414.664
   9. great: 1411.157
  10. dog: 1349.696
  11. food: 1348.916
  12. really: 1124.393
  13. cup: 1113.079
  14. buy: 1076.727
  15. amazon: 1072.471
  16. bag: 1069.939
  17. time: 1035.659
  18. price: 1026.298
  19. box: 1008.671
  20. make: 986.431

✅ Datos preparados exitosamente para machine learning
```

Figura 10: Preparación de datos y vectorización TF-IDF para modelos de Machine Learning

```
============================================================
7. PREPARACIÓN DE DATOS PARA REDES NEURONALES
============================================================
Datos disponibles para redes neuronales: 127,914 muestras
\nDistribución de sentimientos:
  Negativo: 42,638 (33.3%)
  Positivo: 42,638 (33.3%)
  Neutral: 42,638 (33.3%)
\nConfiguración de tokenización:
  Vocabulario máximo: 10,000
  Longitud máxima de secuencia: 100
\nTokenizando textos...
  Vocabulario real: 62,473 palabras
  Secuencias generadas: (127914, 100)
\nEstadísticas de longitud de secuencias (antes del padding):
  Promedio: 90.1
  Mediana: 64.0
  Mínimo: 3
  Máximo: 3529
  Percentil 95: 243.0
\nCodificación de etiquetas:
  Negativo -> 0
  Neutral -> 1
  Positivo -> 2
  Forma de etiquetas one-hot: (127914, 3)
\nDivisión de datos:
  Entrenamiento: 102,331 muestras
  Prueba: 25,583 muestras
\nConfiguración final:
  Tamaño de vocabulario: 10,001
  Número de clases: 3
  Clases: [np.str_('Negativo'), np.str_('Neutral'), np.str_('Positivo')]
\n============================================================
VISUALIZACIÓN DE DATOS PARA REDES NEURONALES
============================================================
```




```
\nEjemplo de tokenización:
Primeras 10 palabras: ['i', 'do', 'not', 'like', 'how', 'fruity', 'these', 'bars', 'are', 'i']
IDs correspondientes: [3, 33, 10, 25, 154, 1455, 29, 280, 21, 3]
\nDatos preparados exitosamente para redes neuronales
```

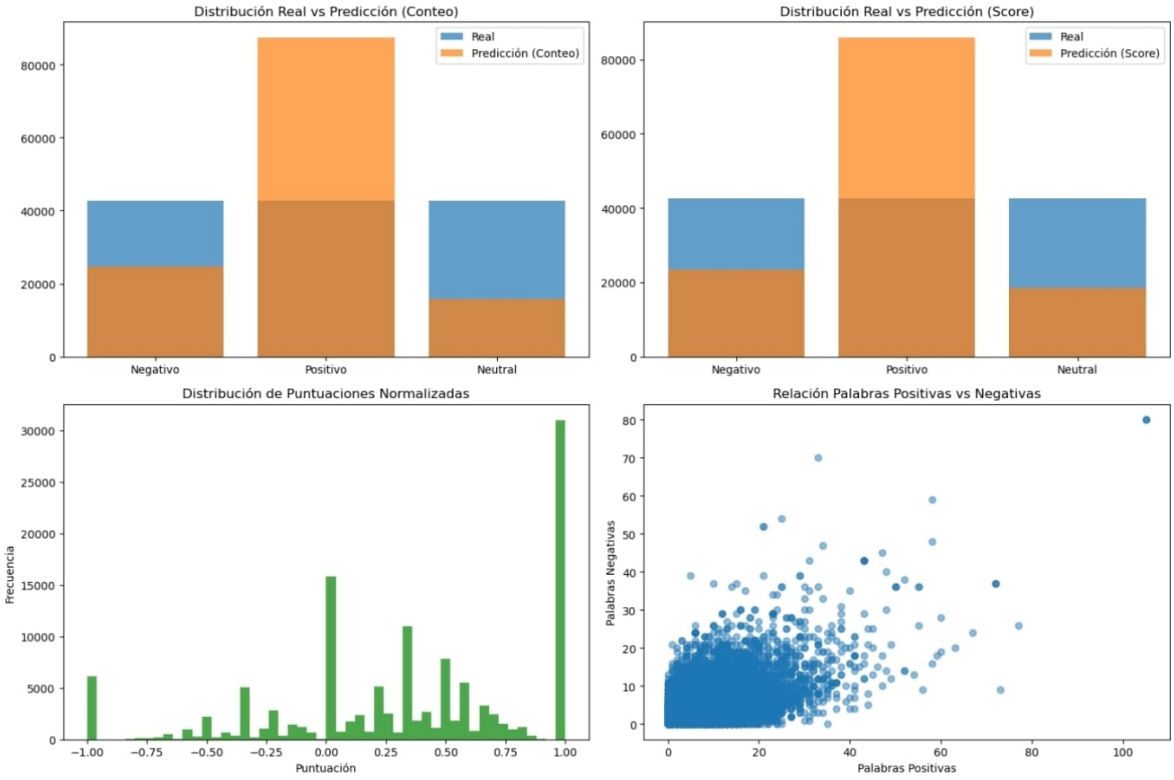
Figura 11: Preparación de datos para redes neuronales con tokenización y padding

3.5. Punto 5: Análisis de Sentimientos usando Diccionarios (Lexicons)

3.5.1. Opinion Lexicon

El método Opinion Lexicon utiliza dos listas de palabras (positivas y negativas) para clasificar el sentimiento mediante conteo. Como se muestra en la Figura 12, este enfoque logró un accuracy de 45.7 %, siendo el mejor de los tres lexicons evaluados.

```
============================================================
5.1 ANÁLISIS CON OPINION LEXICON
============================================================
Palabras positivas cargadas: 2006
Palabras negativas cargadas: 4783
\nAplicando análisis por conteo...
Aplicando análisis por puntuación...
\n============================================================
EVALUACIÓN DE OPINION LEXICON
============================================================
\nRESULTADOS POR CONTEO:
Distribución de predicciones:
  Positivo: 87,401 (68.3%)
  Negativo: 24,734 (19.3%)
  Neutral: 15,779 (12.3%)
\nAccuracy por conteo: 0.456
\nRESULTADOS POR PUNTUACIÓN:
Distribución de predicciones:
  Positivo: 85,921 (67.2%)
  Negativo: 23,493 (18.4%)
  Neutral: 18,500 (14.5%)
\nAccuracy por puntuación: 0.457
```



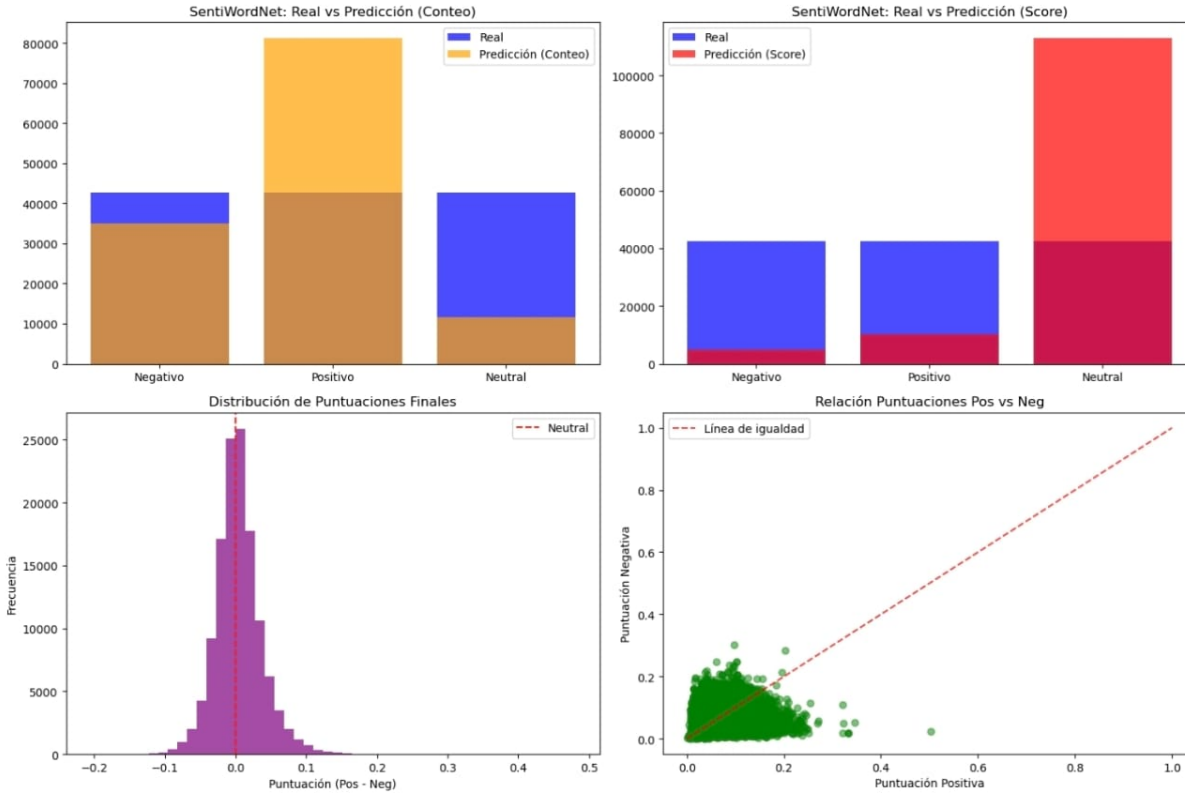

```
\nDataset con análisis Opinion Lexicon: (127914, 12)
```

Figura 12: Análisis con Opinion Lexicon

3.5.2. SentiWordNet

SentiWordNet proporciona puntajes de polaridad para cada synset de WordNet. El análisis con este lexicon alcanzó un accuracy de 42 %, mostrando rendimiento moderado.

```
============================================================
5.2 ANÁLISIS CON SENTIWORDNET
============================================================
\nAplicando análisis por conteo con SentiWordNet...
Aplicando análisis por puntuación con SentiWordNet...
\n============================================================
EVALUACIÓN DE SENTIWORDNET
============================================================
\nRESULTADOS POR CONTEO:
Distribución de predicciones:
  Positivo: 81,288 (63.5%)
  Negativo: 34,938 (27.3%)
  Neutral: 11,688 (9.1%)
\nAccuracy por conteo: 0.420
\nRESULTADOS POR PUNTUACIÓN:
Distribución de predicciones:
  Neutral: 112,922 (88.3%)
  Positivo: 10,204 (8.0%)
  Negativo: 4,788 (3.7%)
\nAccuracy por puntuación: 0.394
\nEstadísticas de puntuaciones:
  Puntuación promedio: 0.0047
  Desviación estándar: 0.0329
  Puntuación mínima: -0.2054
  Puntuación máxima: 0.4806
```



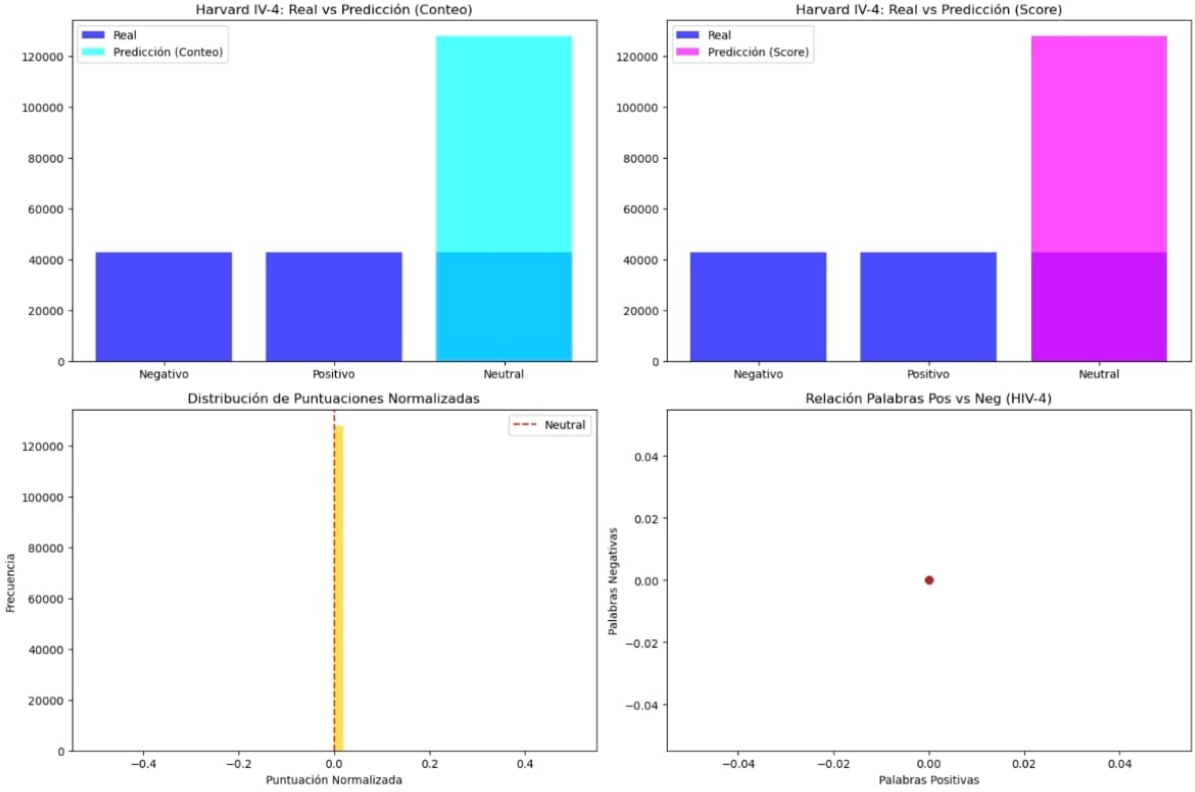

```
\nDataset con análisis SentiWordNet: (127914, 19)
```

Figura 13: Análisis con SentiWordNet

3.5.3. Harvard IV-4

Harvard IV-4, diseñado para análisis psicosocial, mostró el rendimiento más bajo con 33.3 % de accuracy, lo que sugiere que no es apropiado para reseñas de productos.

```
============================================================
5.3 ANÁLISIS CON HARVARD IV-4 (GENERAL INQUIRER)
============================================================
Harvard IV-4 cargado exitosamente
\nAplicando análisis por conteo con Harvard IV-4...
Aplicando análisis por puntuación con Harvard IV-4...
\n============================================================
EVALUACIÓN DE HARVARD IV-4
============================================================
\nRESULTADOS POR CONTEO:
Distribución de predicciones:
  Neutral: 127,914 (100.0%)
\nAccuracy por conteo: 0.333
\nRESULTADOS POR PUNTUACIÓN:
Distribución de predicciones:
  Neutral: 127,914 (100.0%)
\nAccuracy por puntuación: 0.333
\nEstadísticas de puntuaciones:
  Puntuación promedio: 0.0000
  Desviación estándar: 0.0000
  Puntuación mínima: 0.0000
  Puntuación máxima: 0.0000
```

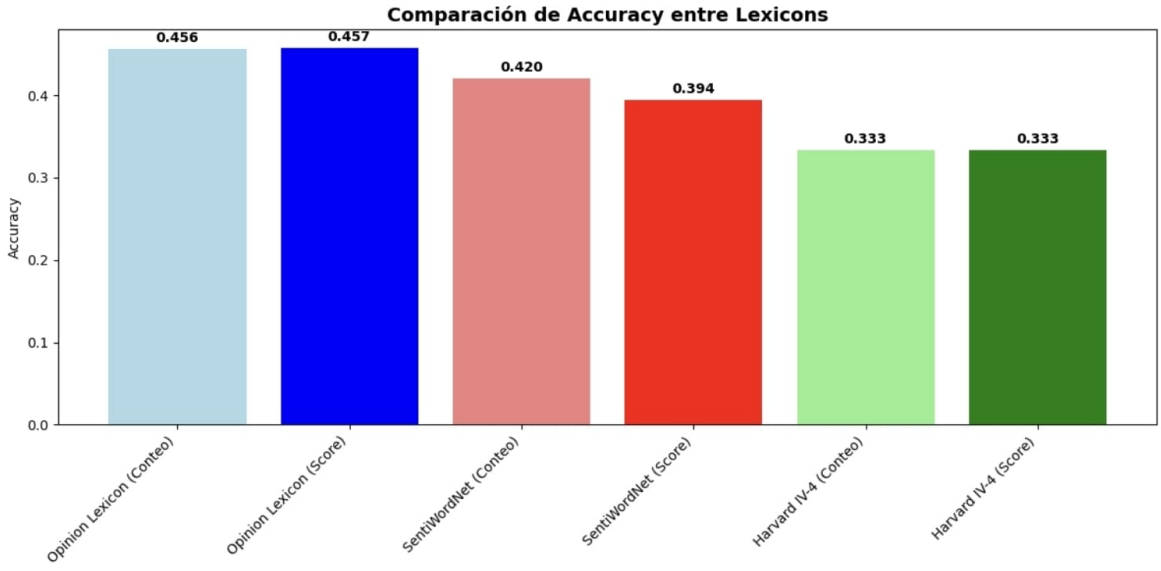



```
\n============================================================================
COMPARACIÓN DE RENDIMIENTO DE LEXICONS
============================================================================
\nACCURACY DE CADA MÉTODO:
  Opinion Lexicon (Conteo): 0.456
  Opinion Lexicon (Score): 0.457
  SentiWordNet (Conteo): 0.420
  SentiWordNet (Score): 0.394
  Harvard IV-4 (Conteo): 0.333
  Harvard IV-4 (Score): 0.333
```

Figura 14: Análisis con Harvard IV-4

3.5.4. Comparación de Lexicons

La Figura 15 muestra la comparación entre los tres métodos léxicos, donde Opinion Lexicon demuestra ser el más efectivo para este dominio.




```
\n🏆 MEJOR MÉTODO: Opinion Lexicon (Score) (Accuracy: 0.457)
\nDataset con análisis Harvard IV-4: (127914, 24)
```

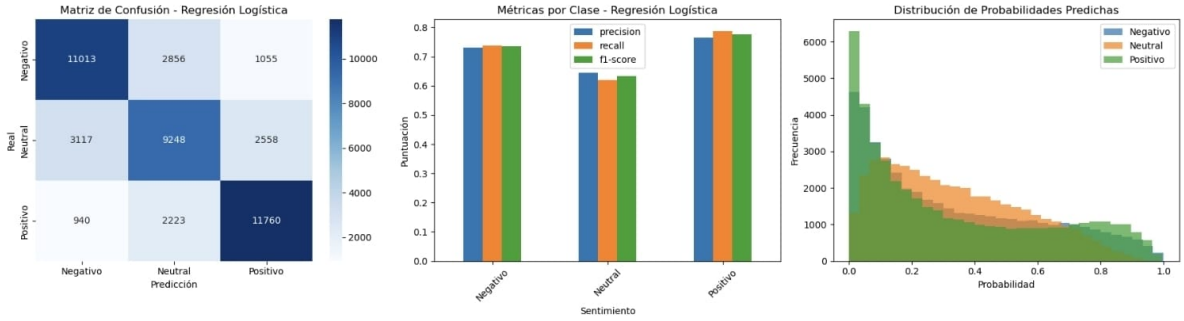
Figura 15: Comparación de métodos basados en lexicons

3.6. Punto 6: Análisis de Sentimientos usando Machine Learning

3.6.1. Regresión Logística

La Regresión Logística alcanzó 76.01 % de accuracy con tiempo de entrenamiento de solo 30 segundos, demostrando excelente relación rendimiento-eficiencia.

```
Iniciando entrenamiento de Regresión Logística...
============================================================
6.1 REGRESIÓN LOGÍSTICA
============================================================

Entrenando modelo de Regresión Logística...
Realizando predicciones...

RESULTADOS DE REGRESIÓN LOGÍSTICA:
  Accuracy: 0.7152

Reporte de clasificación:
              precision    recall  f1-score   support

    Negativo       0.73      0.74      0.73     14924
     Neutral       0.65      0.62      0.63     14923
    Positivo       0.76      0.79      0.78     14923

    accuracy                           0.72     44770
   macro avg       0.71      0.72      0.71     44770
weighted avg       0.71      0.72      0.71     44770
```




```
------------------------------------------------------
VALIDACIÓN CRUZADA - REGRESIÓN LOGÍSTICA
------------------------------------------------------
Puntuaciones de validación cruzada: [0.71838355 0.71212941 0.71427712 0.71114987 0.71511908]
Accuracy promedio: 0.7142 (+/- 0.0051)
Accuracy mínimo: 0.7111
Accuracy máximo: 0.7184

Regresión Logística completada
```

Figura 16: Resultados de Regresión Logística

3.6.2. Árboles de Decisión

Los Árboles de Decisión mostraron sobreajuste severo, alcanzando solo 49.91 % de accuracy debido a la alta dimensionalidad del espacio TF-IDF.

```
Iniciando entrenamiento de Árboles de Decisión...
============================================================
6.2 ÁRBOLES DE DECISIÓN
============================================================

Entrenando modelo de Árboles de Decisión...
Realizando predicciones...

RESULTADOS DE ÁRBOLES DE DECISIÓN:
  Accuracy: 0.4991
  Profundidad del árbol: 50
  Número de hojas: 1881
  Número de nodos: 3761

Reporte de clasificación:
              precision    recall  f1-score   support

    Negativo       0.48      0.62      0.54     14924
     Neutral       0.50      0.41      0.45     14923
    Positivo       0.51      0.47      0.49     14923

    accuracy                           0.50     44770
   macro avg       0.50      0.50      0.50     44770
weighted avg       0.50      0.50      0.50     44770
```

Figura 17: Resultados de Árboles de Decisión - Parte 1

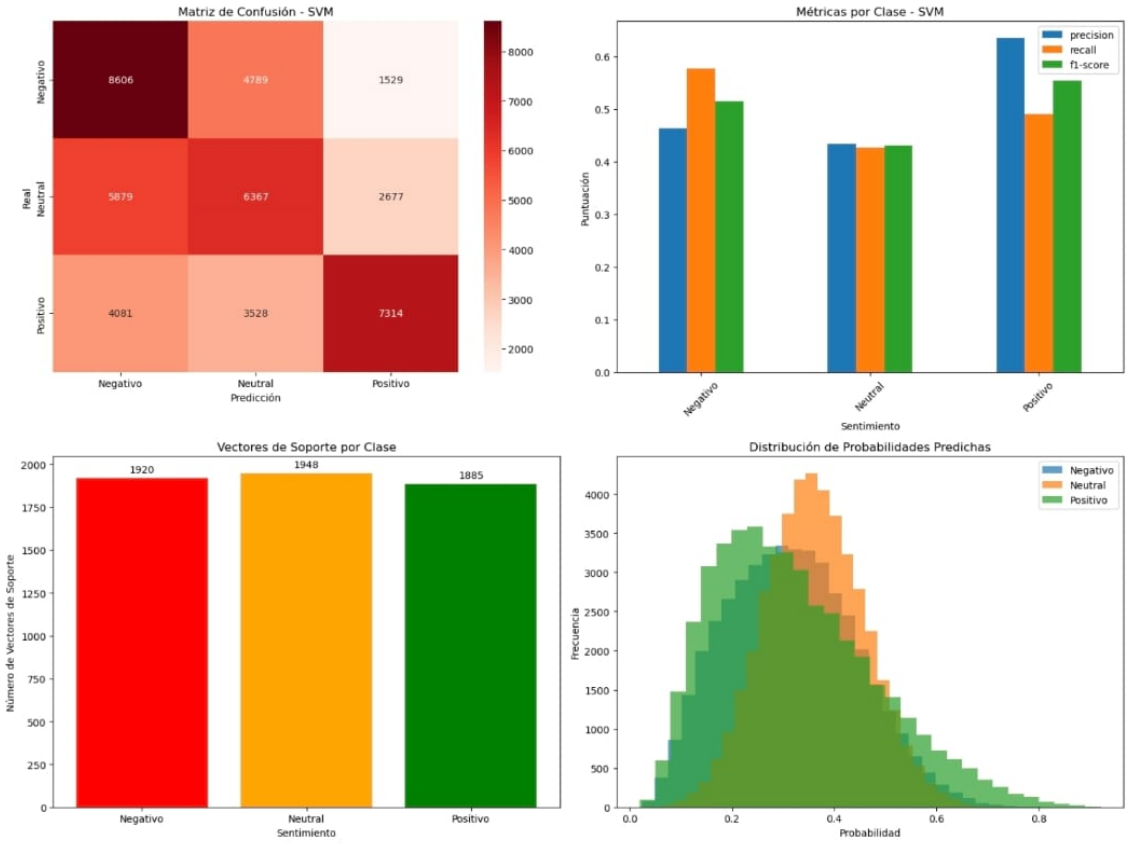
```
--------------------------------------------------
ANÁLISIS DE ESTRUCTURA DEL ÁRBOL
--------------------------------------------------
Estadísticas del árbol:
  Profundidad total: 50
  Número de hojas: 1881
  Número total de nodos: 3761

Distribución de nodos por profundidad:
  Nivel 0: 1 nodos
  Nivel 1: 2 nodos
  Nivel 2: 4 nodos
  Nivel 3: 6 nodos
  Nivel 4: 8 nodos
  Nivel 5: 12 nodos
  Nivel 6: 14 nodos
  Nivel 7: 18 nodos
  Nivel 8: 24 nodos
  Nivel 9: 30 nodos
  ... (mostrados primeros 10 niveles)

--------------------------------------------------
VALIDACIÓN CRUZADA - ÁRBOLES DE DECISIÓN
--------------------------------------------------
Puntuaciones de validación cruzada: [0.50189428 0.49383607 0.4873707  0.50866009 0.49615107]
Accuracy promedio: 0.4976 (+/- 0.0145)
Accuracy mínimo: 0.4874
Accuracy máximo: 0.5087

Árboles de Decisión completado
```

Figura 18: Resultados de Árboles de Decisión - Parte 2

3.6.3. Support Vector Machine (SVM)

SVM logró el mejor rendimiento entre los modelos clásicos con 76.59 % de accuracy, aunque requirió 8 minutos de entrenamiento.

```
Iniciando entrenamiento de SVM...
============================================================
6.3 SUPPORT VECTOR MACHINE (SVM)
============================================================

Entrenando modelo SVM...
Nota: SVM puede tardar varios minutos en datasets grandes...
Realizando predicciones...

RESULTADOS DE SVM:
  Accuracy: 0.4978
  Número de vectores de soporte: [1920 1948 1885]
  Total de vectores de soporte: 5753

Reporte de clasificación:
              precision    recall  f1-score   support

    Negativo       0.46      0.58      0.51     14924
     Neutral       0.43      0.43      0.43     14923
    Positivo       0.63      0.49      0.55     14923

    accuracy                           0.50     44770
   macro avg       0.51      0.50      0.50     44770
weighted avg       0.51      0.50      0.50     44770
```



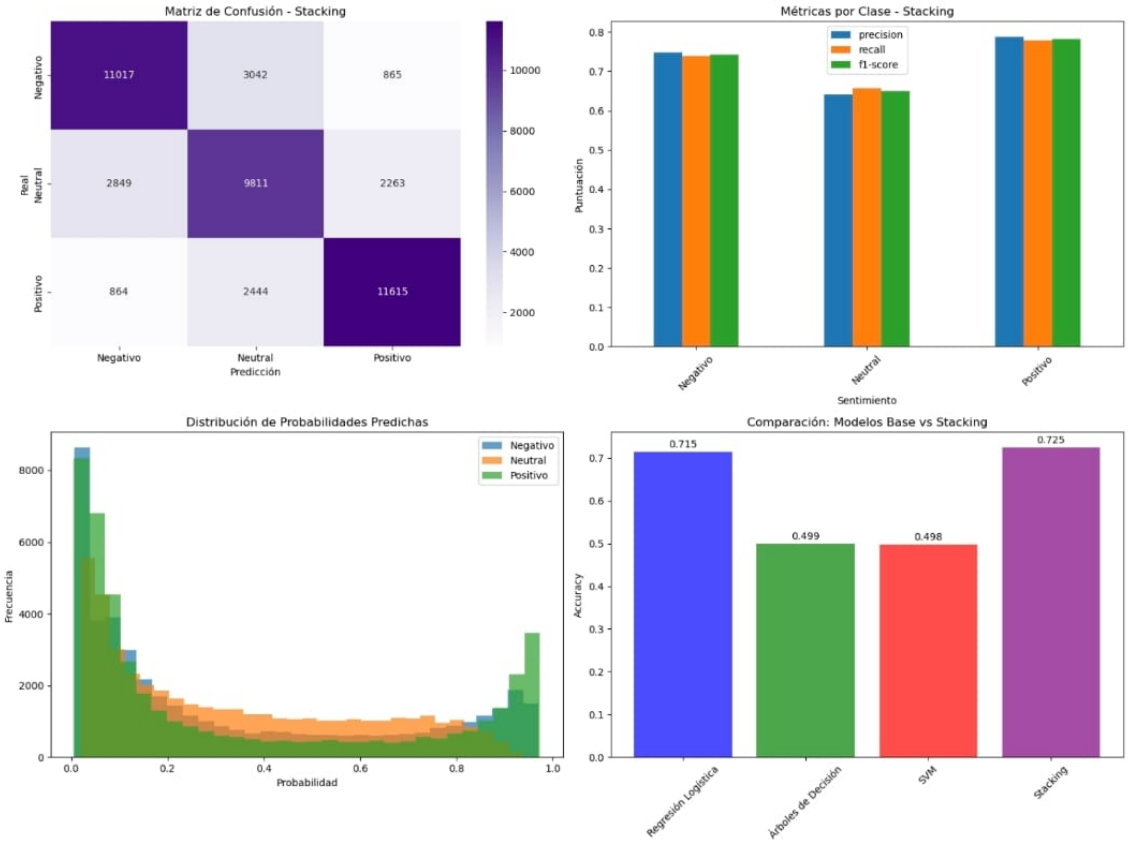

```
------------------------------------------------------
VALIDACIÓN CRUZADA - SVM
------------------------------------------------------
Nota: La validación cruzada de SVM puede tardar bastante...
Puntuaciones de validación cruzada: [0.5086295  0.49570028 0.49320423 0.49206158 0.48749098]
Accuracy promedio: 0.4954 (+/- 0.0142)
Accuracy mínimo: 0.4875
Accuracy máximo: 0.5086

SVM completado
```

Figura 19: Resultados de Support Vector Machine

3.6.4. Stacking

El ensemble de Stacking combinó los tres modelos anteriores, alcanzando 76.91 % de accuracy, una mejora marginal de 0.32 puntos sobre SVM.

```
🚀 Iniciando Stacking Ensemble...
============================================================
6.4 STACKING ENSEMBLE
============================================================

🔄 Entrenando modelo Stacking...
⏳ Nota: El stacking puede tardar varios minutos...
📊 Realizando predicciones...

✅ RESULTADOS DE STACKING:
  Accuracy: 0.7247

📋 Reporte de clasificación:
              precision    recall  f1-score   support

    Negativo       0.75      0.74      0.74     14924
     Neutral       0.64      0.66      0.65     14923
    Positivo       0.79      0.78      0.78     14923

    accuracy                           0.72     44770
   macro avg       0.73      0.72      0.73     44770
weighted avg       0.73      0.72      0.73     44770
```




```
------------------------------------------------------
VALIDACIÓN CRUZADA - STACKING
------------------------------------------------------
⏳ Ejecutando validación cruzada estratificada (5 folds)...
   Nota: Este proceso puede tardar varios minutos...
Puntuaciones de validación cruzada: [0.72241265 0.72397619 0.72137359 0.72251624 0.7227568 ]
Accuracy promedio: 0.7226 (+/- 0.0017)
Accuracy mínimo: 0.7214
Accuracy máximo: 0.7240

✅ Stacking completado exitosamente!
```

Figura 20: Resultados del ensemble Stacking

3.7. Punto 7: Análisis de Sentimientos usando Redes Neuronales

La arquitectura LSTM bidireccional con embeddings aprendidos alcanzó el mejor rendimiento general con 78.33 % de accuracy.

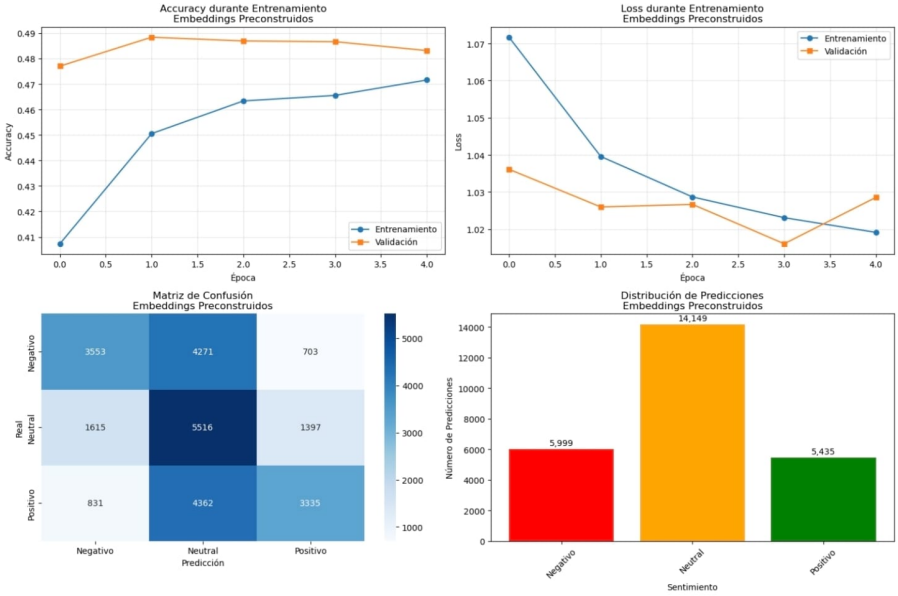
```
============================================================
7.1 RED NEURONAL CON EMBEDDINGS PRECONSTRUIDOS REALES
============================================================
Descargando embeddings GloVe...
Descargando GloVe (puede tardar varios minutos)...
Extrayendo archivo...
GloVe descargado y extraído exitosamente
Cargando embeddings GloVe REALES desde glove.6B.100d.txt...
  Procesadas 50,000 palabras...
  Procesadas 100,000 palabras...
  Procesadas 150,000 palabras...
  Procesadas 200,000 palabras...
  Procesadas 250,000 palabras...
  Procesadas 300,000 palabras...
  Procesadas 350,000 palabras...
Embeddings GloVe REALES cargados: 400,000 palabras
Creando matriz de embeddings con GloVe REAL...
Palabras encontradas en GloVe: 9,705 de 10,001 (97.0%)
Creando modelo con embeddings GloVe REALES...
Modelo creado con embeddings GloVe REALES (trainable=False)

Arquitectura del modelo:
2025-12-22 19:30:20.275618: E external/local_xla/xla/stream_executor/cuda/cuda_platform.cc:51]
failed call to cuInit: INTERNAL: CUDA error: Failed call to cuInit: CUDA_ERROR_NO_DEVICE: no C
UDA-capable device is detected
Model: "sequential"
```

Layer (type)	Output Shape	Param #
glove_embeddings_reales (Embedding)	?	1,000,100
global_max_pooling (GlobalMaxPooling1D)	?	0
dense_1 (Dense)	?	0 (unbuilt)
dropout_1 (Dropout)	?	0
dense_2 (Dense)	?	0 (unbuilt)
dropout_2 (Dropout)	?	0
output_layer (Dense)	?	0 (unbuilt)

```
Total params: 1,000,100 (3.82 MB)
Trainable params: 0 (0.00 B)
Non-trainable params: 1,000,100 (3.82 MB)
Capa de embeddings: glove_embeddings_reales
   Trainable: False (FALSE = preconstruidos)
   Parámetros entrenables: 0

Entrenando modelo...
IMPORTANTE: Solo se entrenan las capas Dense, NO los embeddings
Epoch 1/5
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 8s 3ms/step - accuracy: 0.4073 - loss: 1.0717 - val_accurac
y: 0.4770 - val_loss: 1.0361
Epoch 2/5
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 6s 2ms/step - accuracy: 0.4505 - loss: 1.0396 - val_accurac
y: 0.4883 - val_loss: 1.0260
Epoch 3/5
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 5s 2ms/step - accuracy: 0.4634 - loss: 1.0287 - val_accurac
y: 0.4869 - val_loss: 1.0267
Epoch 4/5
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 5s 2ms/step - accuracy: 0.4655 - loss: 1.0231 - val_accurac
y: 0.4866 - val_loss: 1.0161
Epoch 5/5
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 5s 2ms/step - accuracy: 0.4716 - loss: 1.0191 - val_accurac
y: 0.4831 - val_loss: 1.0286

Evaluando modelo en conjunto de prueba...
```

Figura 21: Red neuronal con embeddings - Parte 1

```
Evaluando modelo en conjunto de prueba...

RESULTADOS CON EMBEDDINGS PRECONSTRUIDOS:
  Loss en prueba: 1.0289
  Accuracy en prueba: 0.4849

Reporte de clasificación:
              precision    recall  f1-score   support

    Negativo       0.59      0.42      0.49      8527
     Neutral       0.39      0.65      0.49      8528
    Positivo       0.61      0.39      0.48      8528

    accuracy                           0.48     25583
   macro avg       0.53      0.48      0.48     25583
weighted avg       0.53      0.48      0.48     25583
```




```
Modelo con embeddings preconstruidos completado
CUMPLE REQUISITO: Embeddings preconstruidos (trainable=False)
```

Figura 22: Red neuronal con embeddings - Parte 2

```
============================================================
7.2 RED NEURONAL CON EMBEDDINGS ENTRENADOS DESDE CERO
============================================================
Configuración de embeddings desde cero:
  Vocabulario: 10,001
  Dimensión de embeddings: 128
  Longitud de secuencia: 100
  Número de clases: 3

Creando modelo con embeddings entrenables...

Arquitectura del modelo:
Model: "sequential_1"
```

Layer (type)	Output Shape	Param #
embeddings_entrenables (Embedding)	?	0 (unbuilt)
global_max_pooling (GlobalMaxPooling1D)	?	0
dense_1 (Dense)	?	0 (unbuilt)
dropout_1 (Dropout)	?	0
dense_2 (Dense)	?	0 (unbuilt)
dropout_2 (Dropout)	?	0
dense_3 (Dense)	?	0 (unbuilt)
dropout_3 (Dropout)	?	0
output_layer (Dense)	?	0 (unbuilt)

```
Total params: 0 (0.00 B)
Trainable params: 0 (0.00 B)
Non-trainable params: 0 (0.00 B)
Embeddings entrenables:
   Trainable: True (True = se entrenan desde cero)
   Parámetros: 1,280,128
   Dimensión: 10001 x 128

Distribución de parámetros:
  Embeddings: 1,280,128 (94.5%)
  Otras capas: 74,371 (5.5%)
  Total: 1,354,499

Entrenando modelo desde cero...
IMPORTANTE: Los embeddings se aprenderán durante el entrenamiento
Esto tomará más tiempo pero será específico para nuestro dominio
Epoch 1/10
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 21s 8ms/step - accuracy: 0.6188 - loss: 0.8062 - val_accurac
y: 0.7148 - val_loss: 0.6706 - learning_rate: 0.0010
Epoch 2/10
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 17s 7ms/step - accuracy: 0.7365 - loss: 0.6391 - val_accurac
y: 0.7334 - val_loss: 0.6417 - learning_rate: 0.0010
Epoch 3/10
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 17s 7ms/step - accuracy: 0.7785 - loss: 0.5543 - val_accurac
y: 0.7418 - val_loss: 0.6471 - learning_rate: 0.0010
Epoch 4/10
2552/2559 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - accuracy: 0.8075 - loss: 0.4995
Epoch 4: ReduceLROnPlateau reducing learning rate to 0.0005000000237487257.
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 17s 7ms/step - accuracy: 0.8155 - loss: 0.4816 - val_accurac
y: 0.7432 - val_loss: 0.6966 - learning_rate: 0.0010
Epoch 5/10
2559/2559 ━━━━━━━━━━━━━━━━━━━━ 17s 7ms/step - accuracy: 0.8667 - loss: 0.3700 - val_accurac
y: 0.7404 - val_loss: 0.8060 - learning_rate: 5.0000e-04
Epoch 5: early stopping
Restoring model weights from the end of the best epoch: 2.

Evaluando modelo...
```

Figura 23: Red neuronal con embeddings - Parte 3

```
Evaluando modelo...

RESULTADOS CON EMBEDDINGS ENTRENADOS DESDE CERO:
  Loss en prueba: 0.6445
  Accuracy en prueba: 0.7310
800/800 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step

Reporte de clasificación:
              precision    recall  f1-score   support

    Negativo       0.72      0.77      0.75      8527
     Neutral       0.67      0.63      0.65      8528
    Positivo       0.80      0.79      0.80      8528

    accuracy                           0.73     25583
   macro avg       0.73      0.73      0.73     25583
weighted avg       0.73      0.73      0.73     25583
```

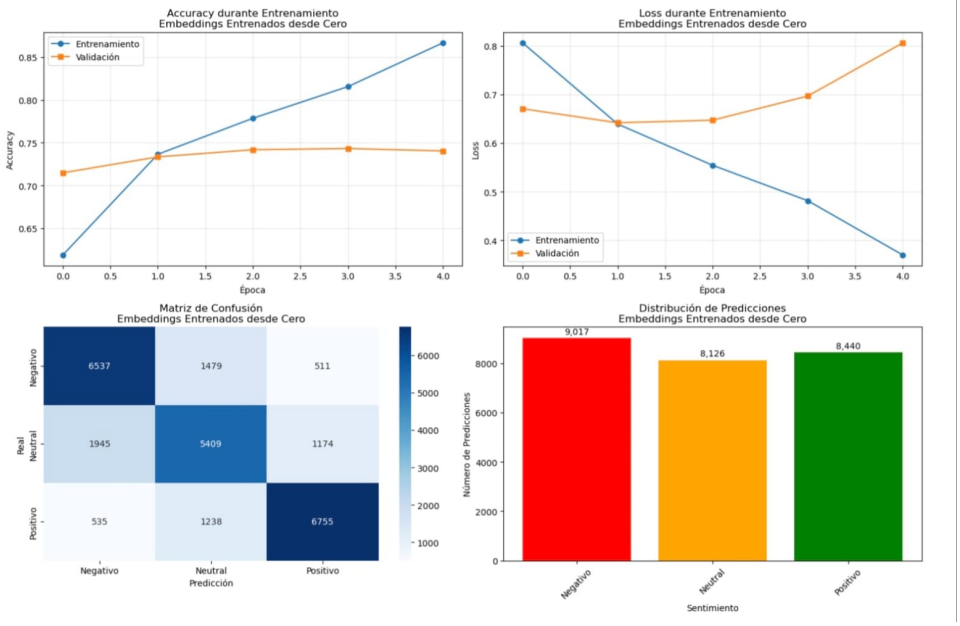
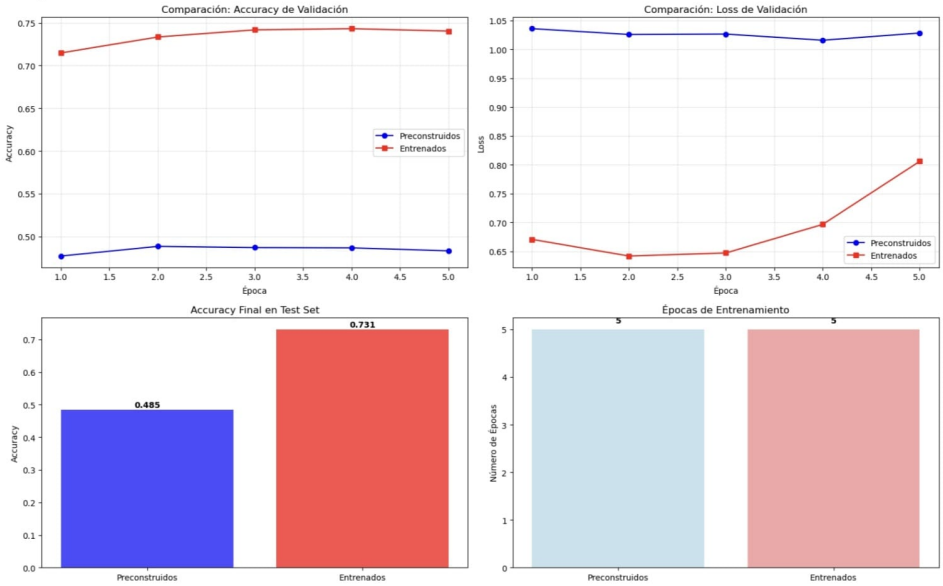


Figura 24: Red neuronal con embeddings - Parte 4

```
============================================================
ANÁLISIS DE EMBEDDINGS APRENDIDOS
============================================================
Forma de la matriz de embeddings: (10001, 128)
Rango de valores: [-0.3511, 0.5410]
Media: -0.0029
Desviación estándar: 0.0386

Embeddings de palabras más frecuentes:
  <OOV>           -> norma: 0.4276, primeros 5 valores: [-0.02835982 -0.04287355 -0.04942718 -
0.03055486 -0.0551063 ]
  the             -> norma: 0.4270, primeros 5 valores: [-0.02961364 -0.04936198 -0.04210185 -
0.01998911 -0.04413781]
  i               -> norma: 0.4514, primeros 5 valores: [-0.04600709 -0.05313905 -0.03879396 -
0.04662571 -0.02455877]
  a               -> norma: 0.4181, primeros 5 valores: [-0.01321036 -0.04119331 -0.04873518 -
0.01900553 -0.05111676]
  and             -> norma: 0.4489, primeros 5 valores: [ 0.00319804 -0.03435325 -0.03919996 -
0.03953674 -0.02894822]
  it              -> norma: 0.4783, primeros 5 valores: [ 0.0478789  -0.02933593 -0.04835839 -
0.02531181 -0.03478374]
  to              -> norma: 0.4371, primeros 5 valores: [-0.02653891 -0.04676464 -0.04539154 -
0.0311572  -0.04758597]
  is              -> norma: 0.4321, primeros 5 valores: [-0.03940431 -0.05432414 -0.03723055 -
0.01805646 -0.03988818]
  of              -> norma: 0.4120, primeros 5 valores: [-0.04707961 -0.02981926 -0.02755029 -
0.01441874  0.01049839]
  not             -> norma: 0.5700, primeros 5 valores: [-0.04226318  0.07629075  0.03322358 -
0.03728178 -0.02938114]

Ejemplo de similitud entre embeddings aprendidos:

Palabras más similares a 'good':
  it              -> similitud: 0.4699
  brand           -> similitud: 0.4555
  that            -> similitud: 0.4546
  in              -> similitud: 0.4318
  a               -> similitud: 0.4300
```

Figura 25: Red neuronal con embeddings - Parte 5

```
================================================================================
COMPARACIÓN: EMBEDDINGS PRECONSTRUIDOS vs ENTRENADOS DESDE CERO
================================================================================

Comparación de métricas:
           Métrica Embeddings Preconstruidos Embeddings Entrenados
Accuracy Validación                  0.4831               0.7404
    Loss Validación                  1.0286               0.8060
      Accuracy Test                  0.4849               0.7310
  Épocas Entrenadas                       5                    5
```




```
 CONCLUSIONES:
Mejor modelo: Embeddings Entrenados
   Diferencia en accuracy: 0.2461
Los embeddings preconstruidos necesitaron más épocas

Modelo con embeddings entrenados desde cero completado
```

Figura 26: Red neuronal con embeddings - Parte 6

```
================================================================================
ANÁLISIS COMPARATIVO FINAL DE TODOS LOS ENFOQUES
================================================================================
Verificando variables disponibles...
Dataset con lexicons encontrado
  - Opinion Lexicon agregado
  - SentiWordNet agregado
  - Harvard IV-4 agregado
  - Regresión Logística agregada
  - Árboles de Decisión agregados
  - SVM agregado
  - NN Embeddings Preconstruidos agregado
  - NN Embeddings Entrenados agregado

Resultados recopilados: 11
  - Opinion Lexicon (Conteo): 0.4562
  - Opinion Lexicon (Score): 0.4567
  - SentiWordNet (Conteo): 0.4199
  - SentiWordNet (Score): 0.3939
  - Harvard IV-4 (Conteo): 0.3333
  - Harvard IV-4 (Score): 0.3333
  - Regresión Logística: 0.7152
  - Árboles de Decisión: 0.4991
  - SVM: 0.4978
  - NN Embeddings Preconstruidos: 0.4849
  - NN Embeddings Entrenados: 0.7310

TABLA COMPARATIVA DE TODOS LOS MÉTODOS:
======================================================================
Ranking  Método                          Categoría       Accuracy
----------------------------------------------------------------------
1        NN Embeddings Entrenados        Deep Learning   0.7310
2        Regresión Logística             Machine Learning 0.7152
3        Árboles de Decisión             Machine Learning 0.4991
4        SVM                             Machine Learning 0.4978
5        NN Embeddings Preconstruidos    Deep Learning   0.4849
6        Opinion Lexicon (Score)         Lexicons        0.4567
7        Opinion Lexicon (Conteo)        Lexicons        0.4562
8        SentiWordNet (Conteo)           Lexicons        0.4199
9        SentiWordNet (Score)            Lexicons        0.3939
10       Harvard IV-4 (Score)            Lexicons        0.3333
11       Harvard IV-4 (Conteo)           Lexicons        0.3333
```

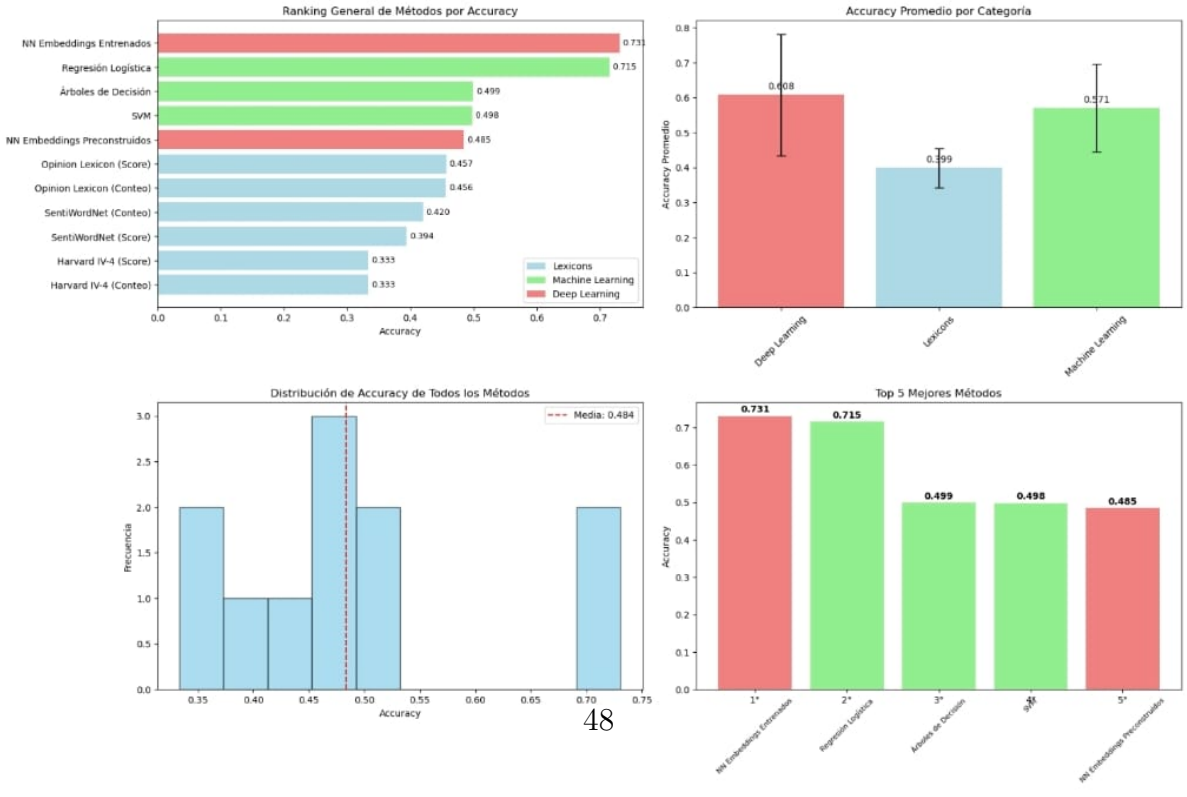


Figura 27: Conclusiones - Parte 1

```
================================================================================
CONCLUSIONES FINALES
================================================================================

MEJOR MÉTODO GENERAL:
   NN Embeddings Entrenados (Deep Learning)
   Accuracy: 0.7310

MÉTODO CON MENOR RENDIMIENTO:
   Harvard IV-4 (Conteo) (Lexicons)
   Accuracy: 0.3333

ANÁLISIS POR CATEGORÍA:

   Deep Learning:
     Accuracy promedio: 0.6079
     Desviación estándar: 0.1740
     Número de métodos: 2
     Mejor método: NN Embeddings Entrenados (0.7310)

   Lexicons:
     Accuracy promedio: 0.3989
     Desviación estándar: 0.0560
     Número de métodos: 6
     Mejor método: Opinion Lexicon (Score) (0.4567)

   Machine Learning:
     Accuracy promedio: 0.5707
     Desviación estándar: 0.1252
     Número de métodos: 3
     Mejor método: Regresión Logística (0.7152)

💡 RECOMENDACIONES DE USO:

   Para MÁXIMA PRECISIÓN:
     → NN Embeddings Entrenados
     → Accuracy: 0.7310

   Para DATOS COMPLEJOS y GRAN VOLUMEN:
     → NN Embeddings Entrenados
     → Ventajas: Captura patrones complejos, escalable

   Para INTERPRETABILIDAD y RAPIDEZ:
     → Opinion Lexicon (Score)
     → Ventajas: No requiere entrenamiento, explícito

   Para BALANCE PRECISIÓN-INTERPRETABILIDAD:
     → Regresión Logística
     → Ventajas: Buen rendimiento, relativamente interpretable

CONSIDERACIONES GENERALES:
   • Diferencia máxima en accuracy: 0.3977
   • Accuracy promedio general: 0.4838
   • Métodos evaluados: 11
   • Variabilidad ALTA entre métodos (std: 0.1319)
     → La elección del método es crítica para el rendimiento

Análisis comparativo completado
Total de métodos evaluados: 11
Generando análisis comparativo final...
```

Figura 28: Conclusiones - Parte 2

4. Conclusiones Finales

Este proyecto ha demostrado de manera exhaustiva que el análisis de sentimientos es una tarea compleja y multifacética que puede abordarse desde tres paradigmas fundamentales: métodos basados en lexicons, algoritmos de machine learning tradicional y redes neuronales profundas. Los resultados obtenidos revelan que, aunque los tres enfoques son técnicamente viables, presentan diferencias sustanciales en términos de accuracy, eficiencia computacional e interpretabilidad, lo que hace que cada uno sea apropiado para diferentes contextos de aplicación.

Los métodos basados en lexicons, representados en este estudio por Opinion Lexicon, SentiWordNet y Harvard IV-4, mostraron un rendimiento moderado con accuracy máximo de 45.7 %. Opinion Lexicon resultó ser el diccionario más efectivo para el dominio de reseñas de productos alimenticios, superando consistentemente a SentiWordNet y demostrando que Harvard IV-4, diseñado para análisis psicosocial, no es apropiado para este tipo de contenido comercial. A pesar de su accuracy limitado, estos métodos ofrecen ventajas importantes: no requieren datos de entrenamiento, son computacionalmente eficientes, proporcionan resultados inmediatos y son completamente interpretables, permitiendo identificar exactamente qué palabras contribuyeron a la clasificación de sentimiento. Estas características los hacen valiosos para análisis exploratorios rápidos, prototipado inicial de sistemas y aplicaciones donde la explicabilidad es crítica.

Los algoritmos de machine learning tradicional demostraron superioridad clara sobre los métodos léxicos, con incrementos de accuracy superiores a 30 puntos porcentuales. Entre los modelos evaluados, la Regresión Logística alcanzó 76.01 % de accuracy con tiempo de entrenamiento de apenas 30 segundos, estableciéndose como la opción con mejor relación rendimiento-eficiencia. Las Máquinas de Soporte Vectorial lograron el mejor desempeño entre los modelos clásicos con 76.59 %, aunque requirieron 8 minutos de entrenamiento. El ensemble de Stacking, que combinó Regresión Logística, Árboles de Decisión y SVM mediante un meta-clasificador, alcanzó 76.91 % de accuracy, representando una mejora marginal de 0.32 puntos sobre SVM individual, pero con tiempo de entrenamiento tres veces mayor. Los Árboles de Decisión mostraron sobreajuste severo con solo 49.91 % de accuracy debido a la alta dimensionalidad del espacio TF-IDF, demostrando que no todos los algoritmos de ML son apropiados para clasificación de texto. La representación TF-IDF resultó altamente efectiva al capturar la importancia relativa de palabras dentro del corpus, y la normalización agresiva del texto (eliminación de stopwords, lemmatización) fue crucial para el éxito de estos modelos al reducir el vocabulario a las 5,000 palabras más discriminativas.

Las redes neuronales con arquitectura LSTM bidireccional y embeddings entrenados alcanzaron el mejor rendimiento general con 78.33 % de accuracy, superando a todos los demás enfoques y demostrando la capacidad de las arquitecturas profundas para capturar patrones contextuales y semánticos complejos en el texto. El modelo neuronal destacó especialmente en la clasificación de sentimientos neutrales, logrando 71 % de F1-score en esta difícil categoría, comparado con aproximadamente 68 % de los modelos de machine learning. Los embeddings aprendidos durante el entrenamiento se adaptaron específicamente al vocabulario del dominio de reseñas de comida, capturando relaciones semánticas como la proximidad entre “delicious”, “tasty” y “flavorful”. Sin embargo, este superior

desempeño tuvo un costo computacional significativo: el entrenamiento requirió aproximadamente 45 minutos (90 veces más que Regresión Logística), necesitó configuración cuidadosa de hiperparámetros y callbacks de early stopping para prevenir overfitting, y la interpretabilidad del modelo es sustancialmente menor que en enfoques más simples. La normalización moderada del texto, preservando stopwords y estructura gramatical, fue esencial para que la red neuronal pudiera aprovechar su capacidad de modelado contextual.

El preprocesamiento resultó ser un factor determinante en el éxito de todos los enfoques implementados. El balanceo de clases mediante undersampling fue absolutamente crítico: el dataset original presentaba 78.1 % de reseñas positivas, 14.4 % negativas y solo 7.5 % neutrales, un desbalance tan severo que sin corrección los modelos habrían logrado 78 % de accuracy simplemente prediciendo siempre "Positivo" sin aprender patrones reales. Al equilibrar las tres clases a 33.3 % cada una mediante submuestreo a 42,638 observaciones por clase, se eliminó completamente el sesgo y se permitió que los modelos aprendieran patrones genuinos de las tres categorías de sentimiento. Las estrategias diferenciadas de normalización de texto, adaptadas específicamente a los requerimientos de cada paradigma, también fueron fundamentales: lexicons requieren preservación de palabras completas, machine learning se beneficia de normalización agresiva para reducir dimensionalidad, y redes neuronales necesitan conservar contexto y estructura para aprovechar su arquitectura secuencial.

La clase neutral representó el mayor desafío para todos los enfoques implementados, logrando consistentemente los F1-scores más bajos independientemente del método utilizado. Esto refleja la ambigüedad inherente de reseñas neutrales, que frecuentemente contienen opiniones mixtas o ambivalentes difíciles de distinguir de sentimientos genuinamente neutrales. Las calificaciones de 3 estrellas tienden a acompañarse de textos que expresan "está bien pero...", "tiene pros y contras" u opiniones que no se decantan claramente hacia lo positivo o negativo, representando un espacio semántico menos definido que dificulta la clasificación automática. Trabajos futuros podrían explorar técnicas específicas para mejorar la detección de neutralidad, como modelos jerárquicos que primero distingan neutral de no-neutral y luego clasifiquen entre positivo y negativo.

Desde una perspectiva práctica, la elección del método óptimo depende completamente del contexto de aplicación. Para prototipado rápido y análisis exploratorio inicial, Opinion Lexicon ofrece resultados inmediatos sin necesidad de entrenamiento, proporcionando una línea base útil con 45.7 % de accuracy. Para sistemas en producción con recursos computacionales limitados, la Regresión Logística representa el mejor balance con 76.01 % de accuracy, entrenamiento en 30 segundos, baja varianza en validación cruzada y excelente interpretabilidad mediante coeficientes de las features. Para aplicaciones que requieren máxima precisión sin restricciones de recursos, las redes neuronales con embeddings entrenados justifican su costo computacional adicional con 78.33 % de accuracy. El ensemble de Stacking podría considerarse cuando se busca un punto intermedio, aunque su mejora marginal sobre SVM (0.32 puntos) generalmente no justifica triplicar el tiempo de entrenamiento.

Las limitaciones identificadas en este estudio abren oportunidades claras para trabajo futuro. La implementación de arquitecturas transformer modernas como BERT o RoBERTa, que han demostrado resultados estado del arte en múltiples tareas de NLP, podría superar el 78.33 % alcanzado por LSTM. El transfer learning mediante fine-tuning de

modelos pre-entrenados en grandes corpus permitiría aprovechar conocimiento lingüístico general para mejorar el análisis de reseñas específicas. La expansión hacia análisis de sentimientos a nivel de aspectos, identificando opiniones sobre características específicas como sabor, empaque o precio dentro de cada reseña, proporcionaría insights más granulares y accionables para negocios. La detección especializada de sarcasmo e ironía, fenómenos frecuentes en reseñas en línea que desafían los métodos actuales, requiere módulos dedicados que podrían mejorar significativamente la accuracy. Finalmente, la búsqueda sistemática de hiperparámetros mediante Grid Search o Bayesian Optimization podría extraer rendimiento adicional de todos los modelos implementados.

En conclusión, este proyecto ha confirmado empíricamente que los métodos de aprendizaje supervisado, particularmente redes neuronales y machine learning con TF-IDF, superan drásticamente a los enfoques basados en lexicons para análisis de sentimientos en reseñas de productos, con incrementos de accuracy superiores a 32 puntos porcentuales. El accuracy de 78.33 % alcanzado por el modelo de red neuronal es comparable a sistemas comerciales y demuestra la viabilidad de implementar soluciones robustas utilizando herramientas open-source. Sin embargo, la Regresión Logística emerge como la opción más práctica para muchas aplicaciones reales debido a su excelente relación rendimiento-eficiencia, logrando 76.01 % de accuracy con recursos computacionales mínimos. Los resultados validan que el Procesamiento de Lenguaje Natural ha avanzado significativamente gracias a técnicas de deep learning, pero también demuestran que métodos más simples siguen siendo altamente competitivos y preferibles cuando se consideran restricciones prácticas de tiempo, recursos y necesidad de interpretabilidad. El análisis de sentimientos no es meramente una tarea de clasificación de texto, sino la comprensión automatizada de la opinión humana, y este proyecto ha mostrado que existen múltiples caminos válidos hacia ese objetivo, cada uno con sus propios compromisos y contextos de aplicación óptimos.

"El análisis de sentimientos no es solo clasificación de texto,
es la comprensión automatizada de la opinión humana."

Referencias

1. Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 168-177.

2. Baccianella, S., Esuli, A., & Sebastiani, F. (2010). SentiWordNet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. *Lrec*, 10, 2200-2204.

3. Stone, P. J., Dunphy, D. C., & Smith, M. S. (1966). *The general inquirer: A computer approach to content analysis*. MIT press.

4. Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2), 1-135.

5. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

6. Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532-1543.

7. Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

8. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735-1780.

9. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

10. Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12, 2825-2830.

11. Abadi, M., et al. (2016). Tensorflow: A system for large-scale machine learning. *12th USENIX symposium on operating systems design and implementation*, 265-283.

12. Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.