Universidad Rafael Landivar

Facultad de Ingeniería en Informática y Sistemas

Inteligencia Artificial Sección 2

Ingeniero Rolando Valdés

CLASIFICADOR DE SENTIMIENTOS CON NAÏVE BAYES (TWITTER)

Nilssen Christopher Chinchilla Galicia 1254016

Daniela José Morales Ayala 1168321

Guatemala, 23 de abril del 2025

INTRODUCCIÓN

Este proyecto tiene como propósito desarrollar un clasificador de sentimientos para textos de Twitter, utilizando el algoritmo Naïve Bayes implementado desde cero. El modelo se entrena con dos conjuntos de datos públicos: Sentiment140 y Twitter Tweets Sentiment (Kaggle), que contienen tweets etiquetados como positivos, negativos o neutrales.

El sistema incluye un proceso de preprocesamiento detallado, orientado a limpiar y normalizar el texto, eliminando elementos irrelevantes y estandarizando el lenguaje informal. La implementación considera unigramas y bigramas para mejorar la representación de los datos.

El modelo se entrena localmente y se expone a través de una API desarrollada con Flask. Adicionalmente, se proporciona una interfaz web donde el usuario puede ingresar texto y visualizar el resultado de la clasificación junto con las probabilidades por clase.

DEFINICIÓN DEL PROBLEMA Y OBJETIVOS

- **Problema:**
  Detectar automáticamente el sentimiento de tweets en inglés.
- **Objetivo General:**
  Desarrollar un clasificador Naïve Bayes capaz de identificar sentimientos positivos, negativos y neutrales en tweets reales, usando un enfoque de entrenamiento profesional y preprocesamiento afinado.

- **Objetivos Específicos:**
  - Combinar datasets reales (Sentiment140 + Kaggle).
  - Aplicar preprocesamiento inteligente que maneje sarcasmo, repeticiones y lenguaje informal.
  - Implementar el algoritmo Naïve Bayes desde cero.
  - Servir el modelo mediante una API con Flask.
  - Evaluar el modelo con métricas estándar.

DESCRIPCIÓN DEL DATASET UTILIZADO

Se utilizaron dos fuentes de datos:

- Sentiment140: 1.6 millones de tweets con etiquetas 0 (negativo), 2 (neutral), 4 (positivo).
- Twitter Tweets Sentiment (Kaggle): ~16,000 tweets etiquetados como positivo, negativo o neutral.

Ambos datasets fueron combinados, normalizados y limpiados para formar un conjunto de entrenamiento robusto y representativo.

DESCRIPCIÓN DEL PREPROCESAMIENTO APLICADO

- Conversión a minúsculas.
- Eliminación de URLs, menciones, hashtags y emojis.
- Reducción de repeticiones de letras ("soooo" → "so").
- Normalización de lenguaje informal y expresiones ("lol" → "laugh", "wtf" → "anger").
- Extracción de unigramas y bigramas.
- Eliminación de stopwords (palabras vacías).

NAÏVE BAYES Y JUSTIFICACIÓN

El algoritmo se implementó desde cero, sin usar librerías como sklearn. Se utilizaron:

- Suavizado de Laplace.
- Log-probabilidades para evitar underflow.
- Cálculo de probabilidades por clase con softmax.
- Extracción de contribuciones de tokens al resultado final.

Esto permite una interpretación clara, control del comportamiento y ajuste del modelo a datos reales de Twitter.

RED BAYESIANA: MODELO DEL PROBLEMA A RESOLVER

El modelo Naïve Bayes se basa en la siguiente suposición:

$$P(\text{clase} \mid \text{palabras}) \propto P(\text{clase}) * \Pi\, P(\text{palabra\_i} \mid \text{clase})$$

Se asume independencia condicional entre las palabras dado el sentimiento. Esto permite estimar la probabilidad de cada clase de forma eficiente.

Explicación de la evaluación del modelo (métricas utilizadas y resultados obtenidos)

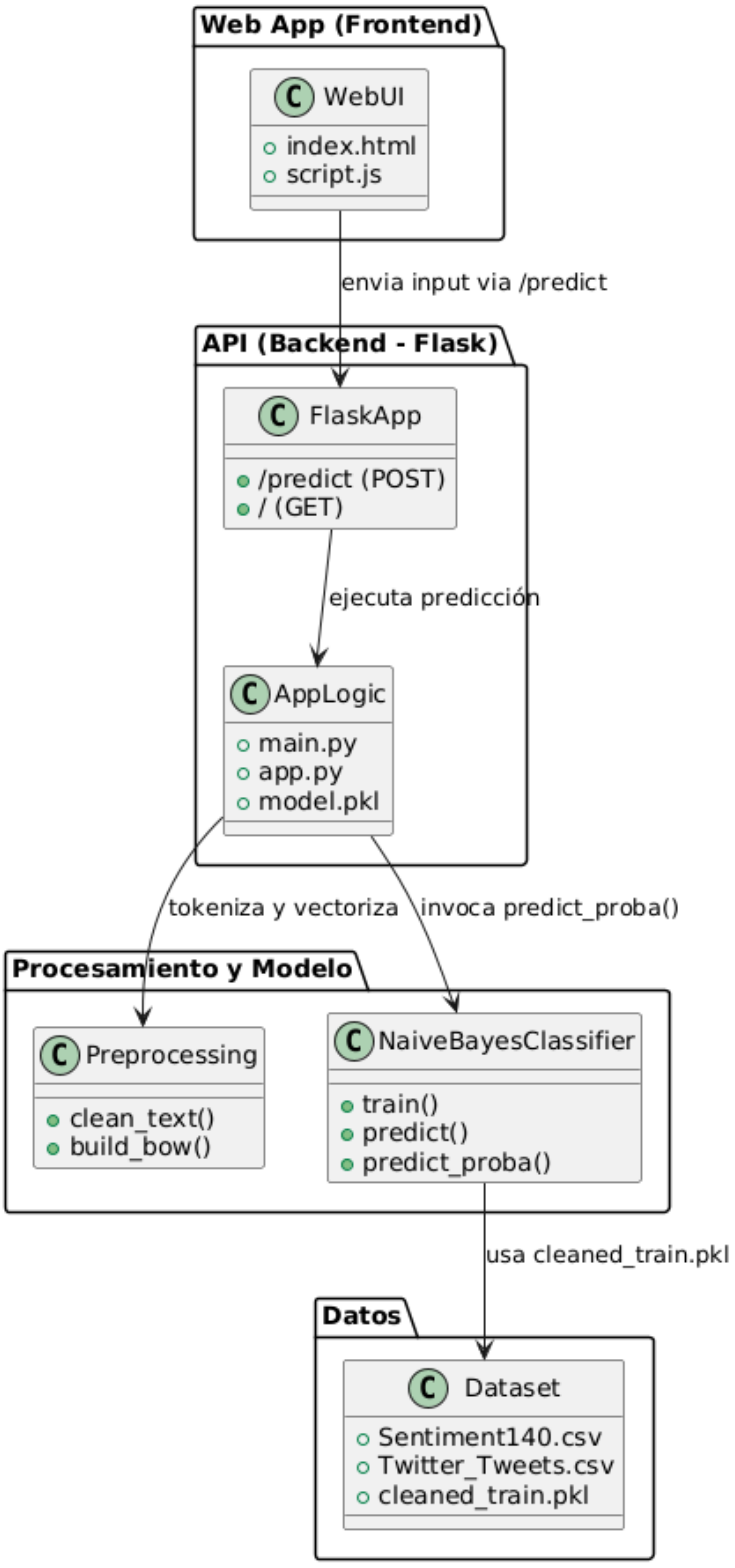
Se utilizó una partición de prueba para evaluar la precisión del modelo:

- Accuracy global: ~77%.
- Reporte de clasificación: incluye precision, recall y f1-score por clase.
- Matriz de confusión: generada con seaborn.

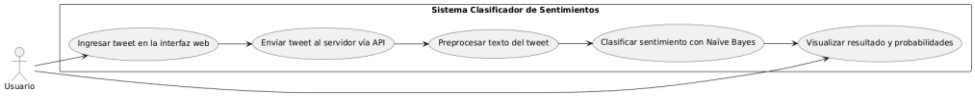
Estos resultados validan que el preprocesamiento detallado y los datos reales permiten un desempeño competitivo del Naïve Bayes.

DIAGRAMAS

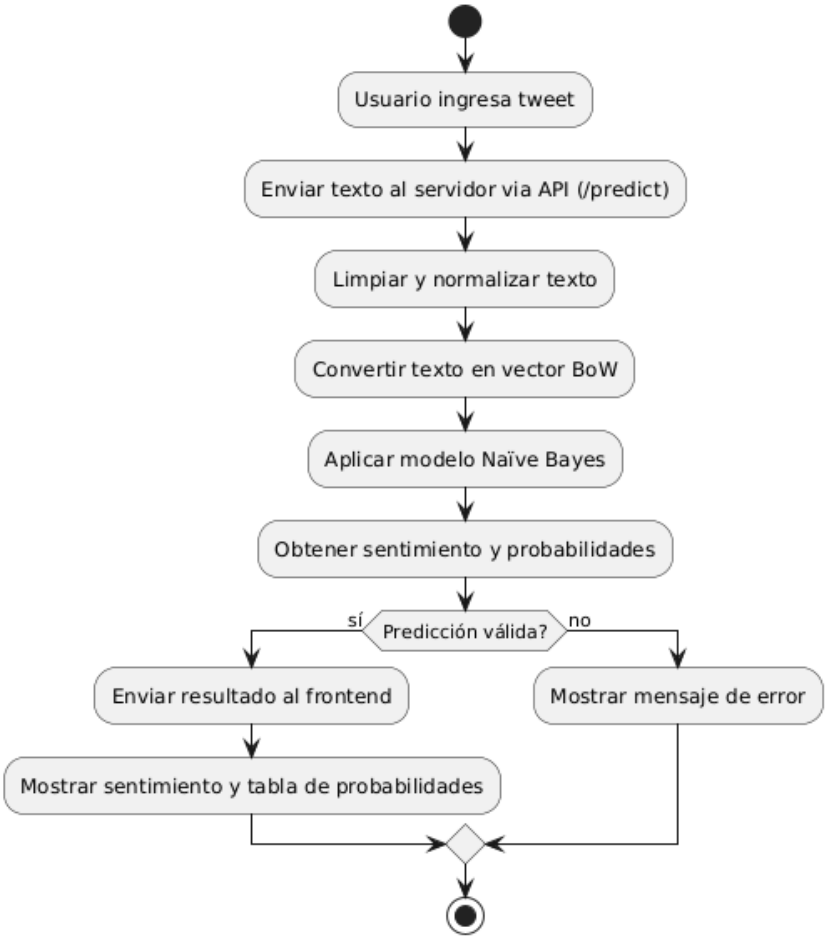
- **Arquitectura de la solución (motor de inferencia + página web):**



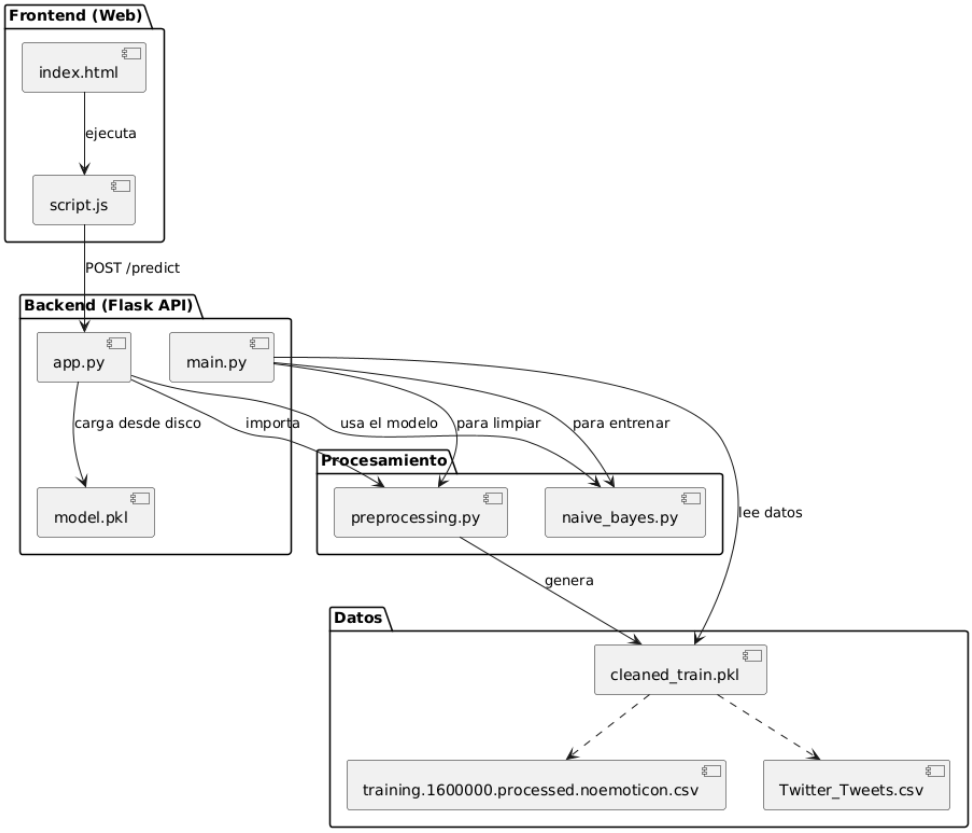
- **Diagramas de Casos de Uso:**



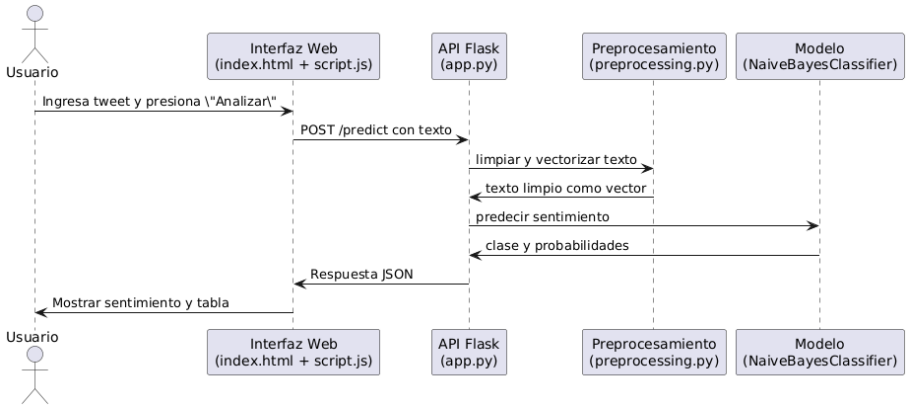
- **Diagrama de Flujo General:**



- **Diagrama de Componentes:**

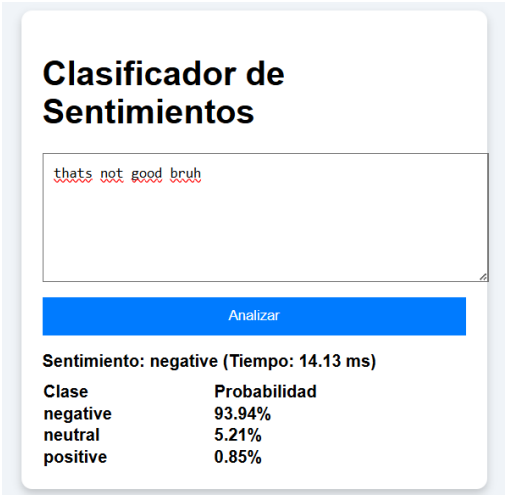


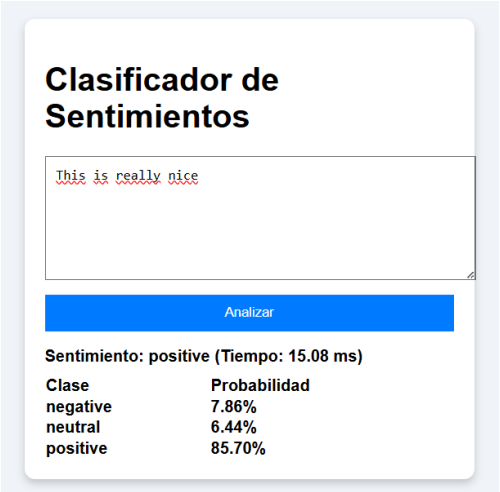
- **Diagrama de secuencias (Modelar la interacción entre el usuario, el frontend y el backend):**

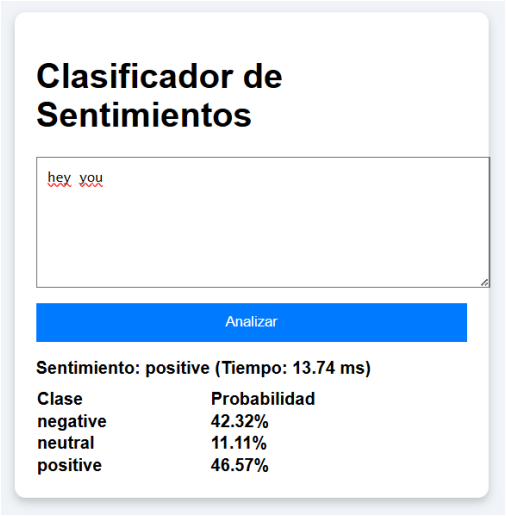


FUNCIONAMIENTO

- **Evidencias de funcionamiento:**
  - Capturas de pantalla de la web:



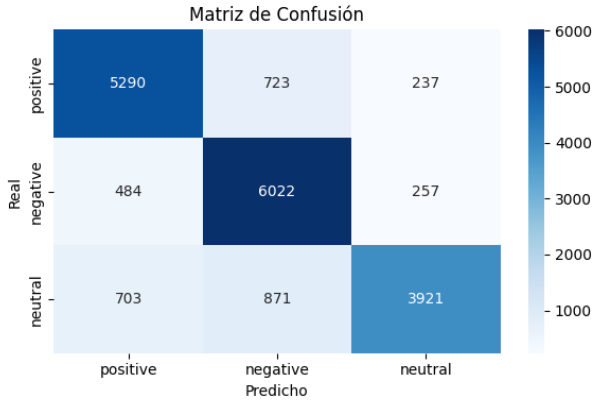




- Tweets de prueba:
    - Positivos:
        - The meeting is at 3pm.
        - I went to the store and bought bread.
        - It’s raining.
        - I had pasta for lunch.
        - We’ll talk later, I’m busy now.
        - The server was restarted successfully.
        - System is running version 3.1.2.

    - Negativos:
        - This is a complete mess.
        - Why does this always happen to me?
        - I can't stand this anymore.
        - Worst experience I've had in a while.
        - Absolutely disappointing.
        - I regret clicking that.
        - Everything broke after the update.
    - Neutrales:
        - The meeting is at 3pm.
        - I had pasta for lunch.

- **Visualización de la tabla de probabilidades por clase en la interfaz:**



- **Lectura por clase:**

  - **Clase Positive (fila 1):**
    - 5290 fueron correctamente predichos como positive
    - 723 fueron mal clasificados como negative
    - 237 fueron mal clasificados como neutral

  El modelo predice bien lo positivo, pero confunde algunos con negativo (quizás por sarcasmo o tono ambiguo).

  - **Clase Negative (fila 2):**
    - 6022 correctamente como negative
    - 484 mal como positive
    - 257 mal como neutral

  Excelente rendimiento en detectar tweets negativos. Es la clase mejor clasificada.

  - **Clase Neutral (fila 3):**
    - 3921 bien clasificados
    - 703 confundidos como positive
    - 871 confundidos como negative

  El modelo tiene más dificultad con lo neutral. Suele confundirlo con otras clases, probablemente porque el tono neutro puede parecer sutilmente negativo o positivo.

CONCLUSIONES Y APRENDIZAJES

- El modelo Naïve Bayes tuvo un buen desempeño general, especialmente al clasificar tweets negativos con alta precisión.
- Se observó mayor confusión en la clase neutral, indicando que esta categoría requiere más datos representativos y puede ser más ambigua para el modelo.
- El preprocesamiento fue determinante para mejorar los resultados. La normalización del texto, la eliminación de ruido y el uso de bigramas ayudaron a reducir errores.
- El uso de datasets reales y diversos permitió construir un modelo más robusto, capaz de manejar distintos estilos de escritura en redes sociales.
- Implementar el algoritmo desde cero permitió comprender cómo funciona internamente y cómo influyen los datos en las decisiones del modelo.
- El sistema completo, incluyendo API y frontend, facilitó la validación práctica del clasificador y permitió observar su comportamiento en escenarios reales.

CONCLUSIONES TÉCNICAS

Clase	Precision	Recall	F1-score	Soporte
positive	0,83	0,84	0,84	6250
negative	0,82	0,88	0,85	6763
neutral	0,78	0,7	0,74	5495

- La clase "positive" es bien reconocida por el modelo, con una precisión del 83% y un F1-score del 84%, lo que indica un balance adecuado entre predicciones correctas y errores.
- La clase "negative" muestra el mejor desempeño general, alcanzando un recall del 88% y un F1-score de 85%, lo que sugiere que el modelo es especialmente bueno detectando este tipo de sentimiento, incluso cuando puede haber ambigüedad o lenguaje más informal.
- La clase "neutral" es la más difícil de clasificar, con la precisión y el recall más bajos del conjunto. Su F1-score de 74% refleja una mayor tasa de confusión con las otras clases, lo que podría deberse a su ambigüedad semántica o a que muchas frases neutrales pueden tener un tono ligeramente positivo o negativo sin intención emocional fuerte.
- El soporte es balanceado, pero se puede observar que las clases "positive" y "negative" dominan en frecuencia. Esto influye en que el modelo aprenda mejor esas categorías.

REPOSITORIO GITHUB

https://github.com/Danielammmm/ProyectoIA_NaiveBayes_Tweets.git